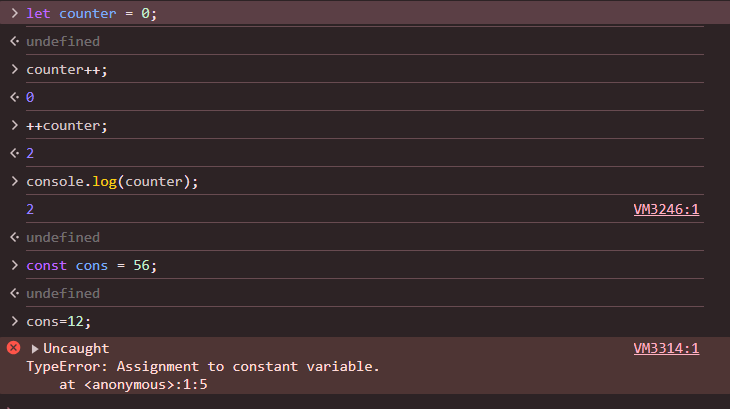
Front end Notes

# Java Script

* Programming language that specifically runs on web. easy to write on web. Powerful language used in the front end of web dev, desktop apps, and many more, Frame of js is widely used in Development such as next.js, electron , react.js,
* ECMA is an international standard organization for Java script. For Example The same program may have different results in different browsers.
* There are run type compilation in browsers for java script. How the code runs in any environment vs how it runs in browser in entirely different.

## JAVA SCRIPT VARIABLES

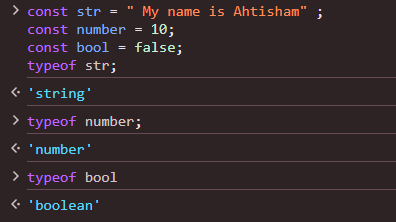
1. Const is for declaring constants in js ;
2. Let is for declaring variable which may be later reassigned;



Camel case , pascal case, snake case , lower case 🡪 all are valid in java script just like java.

Camel case is widely used in declaring variables and used in most of companies 🡪 java thing ;

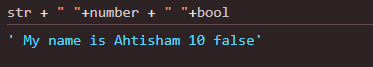
Depending upon the type of Data js automatically assigns its datatype as given below . No need to write data type while typing the variable names .



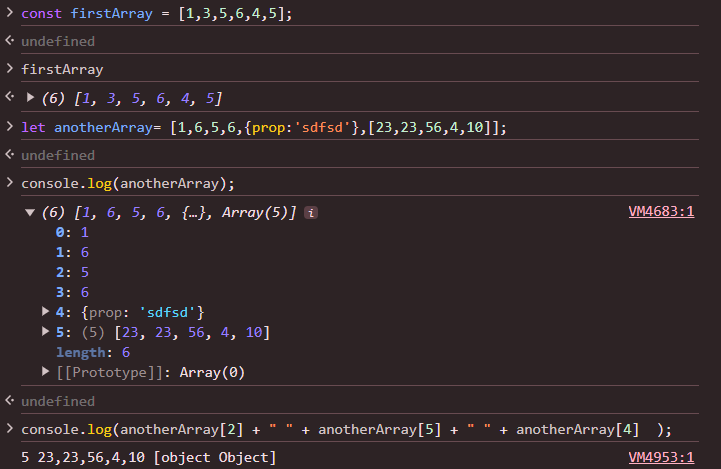
Here typeof give the data Type .

## Implicit conversion :

as shown below concatenates the number , Boolean and strings data types.



Array :



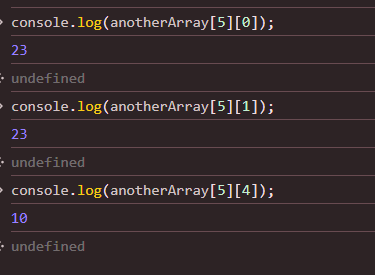
Array rules are same as other programming languages .

In the above example there are six elements :

The element at index 4 is an object

The element at index in the above array is another . Its members can be accessed by treating as 2d array

FOR EXAMPLE :



Here 5 Is the index of subarray in its super array whereas the 2nd bracket gives the index of elements in the subarray;

### In java everything is an object

# JavaScript Array Methods

## 1. map()

The `map()` method creates a new array by applying a function to each element of the original array. It doesn't modify the original array; instead, it transforms the elements and returns a new one.

## Syntax:

let newArray = array.map(function(currentValue, index, arr) {  
 // return element to be added to the new array  
});

## Example:

let numbers = [1, 2, 3, 4, 5];  
  
// Multiply each number by 2  
let doubled = numbers.map(num => num \* 2);  
  
console.log(doubled); // Output: [2, 4, 6, 8, 10]

## Key Points:

- Returns a new array.  
- The original array is not mutated.  
- It’s useful when you want to transform each element in the array.

## 2. filter()

The `filter()` method creates a new array with all the elements that pass the test implemented by the provided function. If the condition is true, the element is kept; otherwise, it is filtered out.

### Syntax:

let newArray = array.filter(function(currentValue, index, arr) {  
 // return true to keep the element, false otherwise  
});

### Example:

let numbers = [1, 2, 3, 4, 5];  
  
// Get only even numbers  
let evens = numbers.filter(num => num % 2 === 0);  
  
console.log(evens); // Output: [2, 4]

### Key Points:

- Returns a new array with only the elements that pass the test.  
- The original array is not mutated.  
- It’s useful when you want to create a subset of an array based on a condition.

## 3. find()

The `find()` method returns the first element in the array that satisfies the provided condition. If no element satisfies the condition, it returns `undefined`.

### Syntax:

let foundElement = array.find(function(currentValue, index, arr) {  
 // return true when condition is met  
});

## Example:

let numbers = [1, 2, 3, 4, 5];  
  
// Find the first number greater than 3  
let firstGreaterThanThree = numbers.find(num => num > 3);  
  
console.log(firstGreaterThanThree); // Output: 4

### Key Points:

- Returns the first element that satisfies the condition.  
- If no element passes the test, it returns `undefined`.  
- It’s useful when you need just one item that meets a condition (the first match).

# Key Differences

|  |  |  |
| --- | --- | --- |
| Method | Description | Return Value |
| map() | Transforms each element in the array. | A new array with the transformed elements. |
| filter() | Filters elements based on a condition. | A new array with the filtered elements. |
| find() | Finds the first element that meets a condition. | The first element that satisfies the condition (or undefined). |

# Loops :

JavaScript Loops: for, for...in, and for...of

## 1. for Loop

The traditional `for` loop is the most flexible because you have complete control over how it iterates through an array. It allows you to manipulate the loop's index, which is useful when you want to skip elements, reverse the iteration, or need access to both the index and the value.

## Key Characteristics:

- Can manipulate the index (i).  
- Can break or continue based on conditions.  
- Works with any data structure that has indices (like arrays or strings).

## Example:

let arr = [10, 20, 30, 40];  
  
for (let i = 0; i < arr.length; i++) {  
 console.log(`Index: ${i}, Value: ${arr[i]}`);  
}

## 2. for...in Loop

The `for...in` loop is primarily meant for enumerating object properties, but you can also use it for arrays because arrays in JavaScript are treated as objects with numeric keys.

## Key Characteristics:

- Iterates over the keys (which are indices in an array).  
- Suitable for objects but not recommended for arrays.  
- Doesn't guarantee the order of iteration, especially for non-numeric properties.

Example:

let arr = [10, 20, 30, 40];  
  
for (let key in arr) {  
 console.log(`Index: ${key}, Value: ${arr[key]}`);  
}

## 3. for...of Loop

The `for...of` loop is introduced in ES6 (ECMAScript 2015) to work with iterable objects. Arrays are iterables, so this loop works seamlessly with them, iterating over the values directly, without concerning itself with keys or properties.

## Key Characteristics:

- Iterates over values directly (no need to worry about indices).  
- Works with any iterable, such as arrays, strings, Maps, Sets, and more.  
- Simple and concise for looping through elements of an array.

Example:

let arr = [10, 20, 30, 40];  
  
for (let value of arr) {  
 console.log(`Value: ${value}`);  
}

# Key Differences:

|  |  |  |  |
| --- | --- | --- | --- |
| Loop | Iterates Over | Output | Best Use Case |
| for | Manual index | Index + Value | Control over iteration (index, step) |
| for...in | Keys (indexes) | Index + Value | Enumerating properties of an object |
| for...of | Values | Value (no index) | Values of arrays or iterables |

# [Jquery notes](https://www.geeksforgeeks.org/jquery-cheat-sheet-a-basic-guide-to-jquery/)(click here :source GeeksForGeeks).

Animations using jquery :

Animate( {

Css code here in css format

Margin : 5px ; here second value sould be number only we cant do color : red })

Sildeup();

Slidedown();

Tooggleslide();

Fadein();

Fadeout();

Show();

Hide();

Chaining : $(“h1”).Sildeup().Slidedown().animate(); here functions are applied left to right.