

**Study Notes for  
Interview Preparation**

**by SahebCSE**

# JavaScript

**Complete Handwritten Notes**

**PLEASE SHARE AND HELP OTHERS**



**@PrepTrain**



**@SahebCSE**

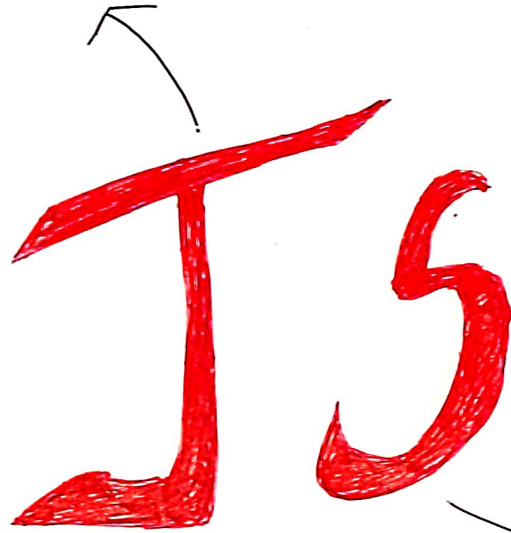


**PrepTrain**

**Do Join telegram.me/PrepTrain Community for  
latest Internships and Job Opportunitites**

- Basic Javascript
- Introduction to Javascript
  - History
  - Syntax
  - Values
  - Variables & its types

JS



Intermediate Javascript

- Scope of Javascript Variables
- Expressions
- Operator & their precedence
- Comparisons
- Conditionals
- Strings
- Arrays
- Loops

Advanced Javascript

- Functions
- Arrow Functions
- Objects
- Object Properties
- Object Methods
- Classes
- Inheritance
- Asynchronous Programming
- Callbacks
- Promises
- Async & Await

# Introduction to JavaScript

Javascript is used to create :-

- Websites
- Web Applications
- Server-Side Applications Using Node Js.
- Creating Mobile Applications using Tools (React Native).
- Creating Programs for Microcontrollers and IOT.
- Creating Smartwatch applications.

Salient Features of Js →

- High Level
- Dynamic
- Loosely Typed
- Multi-paradigm
- Dynamically Typed
- Dynamic
- Interpreted

Syntax:

White Space:

Javascript does not consider white space meaningful. Spaces and line breaks can be added in any fashion, although you will most likely keep a well defined style and adhere to what people commonly use.

Case Sensitive:

JavaScript is case-sensitive. Variable "HELLO" and "hello" are different.

Literal:

Values Written in the source code are called Literals.

## Identifiers

An identifier is a sequence of characters that can be used to identify a function, variable or an object. It can start with a letter, dollar sign or an underscore.

## Comments

These are most important part of any program. They help to transfer the idea of code and describe the code to others.

Any line can be made comment by adding `//` in front of the line at start.

## Semicolons

Every line in a Javascript program is optionally terminated using semicolons. I said optionally because Javascript interpreter is smart enough to introduce semicolons for you.

## Values

A `"hello"` string is a value. A number `"12"` is a value.

String and Number are the types of those values. We assign values to **variables**.

## Variables

Variable is a value assigned to an identifier, so you can reference and use it later in a program.



We can declare variables in 2 ways:-

const:

Variables defined with const can't be changed later.

Let:

Variables declared with let can be changed later.

Const Variable have to be assigned at time of declaration, whereas let variables can be assigned later.

const a = 0		let a
		a = 0.

## Types OF Variables

There are two main kinds of types:-

Primitive types:

- Numbers
- Strings
- Booleans
- Symbols

Object types:

Which do not belong in primitive types.

## Variable Scope

In JavaScript, there is global scope, block scope and function scope.

If a variable is defined outside function or block it is attached to global object and has global scope, which means it's available in every part of code.

→ A variable defined as var inside a function is only inside a function.

→ A variable defined as let or const on the other hand is only visible inside a function (block) where it is defined.

## Expression

Expression is a single unit of JavaScript code that JavaScript engine evaluates.

Arithmetic expressions take a variable and result to a number.

String expressions result into a string.

## Operators

Operators allow you to get two simple expressions and combine them to form a more complex expression.

1) `const a = []`

2) `const a = Array()`

We can initialize pre-filled array also.

1) `const a = [1, 2, 3]`

2) `const a = Array.of(1, 2, 3)`

\* Array can hold values of different types.

Arrays can be multidimensional.

Length of array can be determined by `length` property.

→ Adding an element to array →  
`a.push(element)`

→ Removing an element →  
`a.pop()` → from end  
`a.shift()` → from beginning

→ Join multiple arrays →

`c = a.concat(b)`

c is new array, a and b are existing arrays.

→ Finding specific element in array →

`a.find(element, index, array) => {`  
// returns true or false

`}`

Note → Find Index can be used instead of find to return index.



## Loops

Through loops we can automate and repeat a block of code indefinitely.

### While loop

```
while(true) {  
    if (something is True) break  
}
```

### For loop

We specify three parameters initialization, the condition, increment.

```
for ( initialize; condition; increment ) {  
    [ block of code ]  
}
```

### For .. of

This is newly introduced simplified form of For loop.

```
Ex for (const value of list) {  
    [ block of code ]  
}
```



## Functions

A function is a block of code, self contained.

```
function getData() {  
    // do something  
}
```

 } Declaration

```
function getData (parameter) {  
    // do something  
}
```

 } Declaration with parameter

There can be multiple parameters.

In function we get the result by return statement.

We return some value.

We can also return a function.

## Arrow Functions

```
() => {  
    // do something  
}
```

 } Declaration

Invoking a function using variable name.

```
let getData = () => {  
    // - - - - -  
}
```

 } Declaration

```
getData()
```

 } Calling.

0/00  
0/Var

## Objects

Value that's not of a primitive type is an object.

`const car = new Object()`

`const car = Object.create()`

} Declaration

Objects are always passed by reference.

## Objects Properties

Objects have properties which is composed by a label associated with a value.

The value of a property can be of any type.

## Object Methods

Functions can be assigned to a function property, in this case they are called methods.

Syntax:—

`const variable = {  
 property: function() {`

`}  
}`

} Declaration

`variable.property()` } Calling

## Classes

```
class Person = {  
    name = "Jordan"  
}
```

} Declaration

## Inheritance

A class can extend another class and object initialized using that class inherit all the methods of both classes.

## Callbacks

A callback is a function passed as an argument to other project.

## Promise

A promise is a proxy for a value not necessarily known when the promise is created. It allows you to associate handlers with an asynchronous action's eventual success