

JS Intro
The most popular programming lang
It is the programming lang of the web

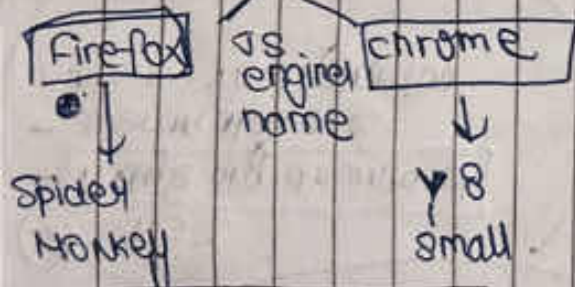
What can you do with JS

- To build Interactive webpages
- Realtime Networking apps
- Mobile Applications
- Command line tools
- Games

Where does JS code runs

- Only in browser

A browser is known to be a javascript engine that can execute js code.

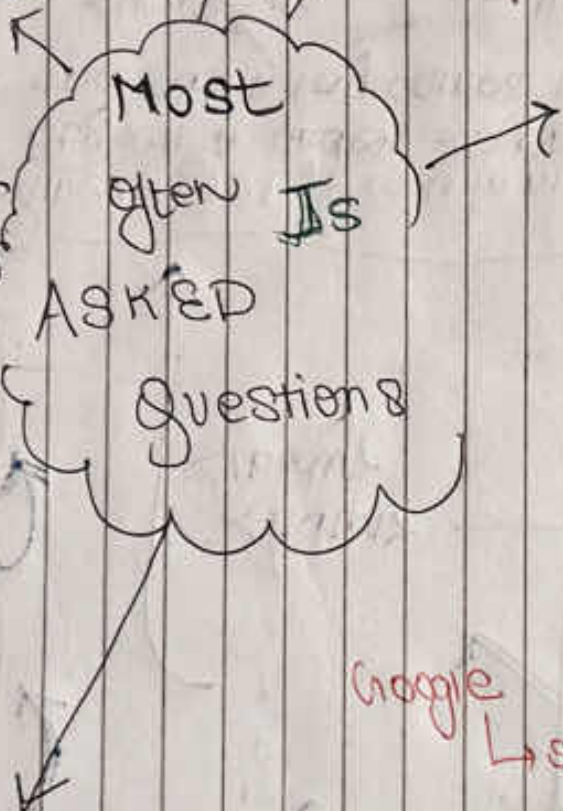


What is Node.js?

Extract the open source js attach to a C++ code and this is how Node is made (having google's js engine)

What we can build with JS?

With JS we can build the backend of our website.



What is the difference

ECMA Script	Javascript
just a specification	a prog lang it confirms this specification
it is Responsible for Defining standards	
ES6 being recent	

Google

→ right click

→ Inspect

console.log('Hello world')
hello world
2+2
4
about('yo')



1 **use code editor?**

- VScode
- sublime text
- Atom

code.visualstudio.com
cross platform
powerful editor

2 **use Node**

- node.js.org download
- we can also use chrome vs engine too

3 **use live server in vs**

- download live server extension
- open with live server for an html file.

4 **create a folder**

- on desktop
- js practise folder

5 **create an html file**

- html file in folder

6 **open with live server**

After downloading extension - start vs code after download

Where to Start

- 1 - Use code editor - VScode
- 2 - Use Node - node.js.org
- 3 - Use live server extension to run html
- 4 - create a folder in desktop
- 5 - create an html file in that folder
- 6 - run by right clicking and opening it with live server.
- 7 - Use script tag

<body>

<script>
</script>

7 **where goes js code in html**
It goes under <script> tag
the script tag comes under

<Head> or
<script>

<body>
<script>

best practice is to add it at the end of body script section

Separation of concerns

Keeping things separated → clothes kept not on the bed but in a separate cupboard

& we separate html which is all about content from javascript which is all behaviour about

index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title> Document </title>
  </head>
  <body>
    <h1> Hello world </h1>
    <script src="index.js"> </script>
  </body>
</html>
```

index.js

```
console.log('Hello world');
```

super neat
↓
here.

a separate file for js

JS in node

→ downloading node from nodejs.org

→ open up command prompt → head over to the folder created js-practice

→ now run command `node index.js`

→ we use a variable to store data temporarily
declaration `var a;` → declaring with var

variable



boxes to store items

so `let name;` brings up issues
by default variables defined their value is undefined

→ strings can be both single or double quotes.

- ① → the name can't be a reserved keyword
- ② → the name should be meaningful (we on purpose of the var)
- ③ → they cannot start with the no.
- ④ → they cannot contain a space or a hyphen (-)
- ⑤ → can use camel notation
- ⑥ → they are case sensitive
- ⑦ → to declare multiple variables

`let firstName = 'abc', lastName = 'cde';`
or
single line `let firstName = 'abc';`
single line `let lastName = 'cde';`

Rules
for Variable
Name
Selection

- ^{not} To change the value of variable we use constants → when you don't want to reassign values.

```
const interestRate = 0.3;
interestRate = 1;
```

Constants

error
will be seen on console

Primitive type

→ there are two types

primitive type

Reference type

the members
are

→ They are also known as value types too.

String Number Boolean Undefined

```
# let name = 'Mosh';  String literal
# let age = 30;       Number literal
# let isApproved = true; Boolean literal.
# let firstName = ;   Undefined
# let or lastName = null; → clear out the value.
```

Dynamic Typing

Javascript is a dynamic language

Static
language

→ statically typed

Dynamic → dynamically
typed

when we declare a variable
its ^{type} ~~value~~ can't be changed
in the future

when we declare a variable, its type can be
changed at runtime

typeof to check the type of variable. → type of name
→ "string"


```
person['name'] = 'Mary';  
console.log(person.name);
```

Which Approach is better? — Dot vs. bracket Notation.

- dot notation is a bit more concise, that should be a default choice.

- bracket notation is helpful when we don't know value at the moment

```
let selection = 'name';  
person[selection] = 'Mary';  
console.log(person.name);
```

Arrays

① When dealing with list of objects we use Array. List of products in a cart
Array is needed to store that list. List of colors a user has selected

② `let selectedColors = ['red', 'blue', 'black'];` → square brackets denotes Array literal

③ Every element has an index in the Array `console.log(selectedColors[2]);`

④ As JS is a dynamic language, so the type of variable can change at runtime. The same principle applies to arrays. ~~So the type of variable~~ So the lengths of the array as well as the type of object in array are dynamic as in they can change. eg `selectedColors[2] = 'green';` → the array length will now change
`selectedColors[3] = 1;`

⑤ Type of an array is object

Functions

one of the fundamental building blocks in JS. A set of statements that performs a task or calculates a value.

For eg: function greet() {

console.log("Hello");

↳ we do not need a semicolon at the end of the function as we are not declaring it as a variable.

↳ calling a function

greet();

Q1 = write a function that returns the square of a number.

function findSquare(n) {

return n * n;

↳ console.log(findSquare(2));

Q2 = write a function to find area of rectangle

function area(a, b) {

return ("The area of rectangle is " + a * b);

↳ console.log(area(10, 20));

Q3 = write a function to convert Celsius to Fahrenheit

function calFahrenheit(cel) {

return (cel * (9/5)) + 32;

↳ console.log(calFahrenheit(20)) // 68

Q4 = write a func to calculate area and perimeter of circle

function circleValue(rad) {

return { Peri: 2 * Math.PI * rad, Area: Math.PI * rad * rad };

↳ console.log(circleValue(10))

Q write a function to reverse a number

```
function reverseNum (num) {
  var reverse = 0;
  while (num != 0) {
    reverse = reverse * 10;
    reverse = reverse + num % 10;
    num = Math.trunc(num / 10); // remove decimal no.
  }
  return reverse;
}
console.log(reverseNum(123))
```

Q count no. of vowels in a string

```
function countVowel (str) {
  let count = 0;
  str = str.toLowerCase();
  for (var i = 0; i < str.length; i++) {
    if (str.charAt(i) == 'a') count++;
    if (str.charAt(i) == 'e') count++;
    if (str.charAt(i) == 'i') count++;
    if (str.charAt(i) == 'o') count++;
    if (str.charAt(i) == 'u') count++;
  }
  return count;
}
console.log(countVowel("Hello"));
```

Q Flatten array of arrays using JS reduce

Flatten a 2D array to 1D array by using JS reduce

```
function flattenArray (arr) {
  return arr.reduce((result, array) => result.concat(array));
}
console.log(flattenArr([[1, 2, 3], [4, 5, 6], [7, 8, 9]]))
// [1, 2, 3, 4, 5, 6, 7, 8, 9]
```


Q write a program to check
if entered no. is a palindrome

012

$$3-0-1 = 2$$

0

$$3-1-1$$

1

```
function checkPalindrome (str) {  
  for (var i=0; i<str.length; i++) {
```

```
    if (str.charAt(i) != str.charAt(str.length-i-1)) {
```

```
      return false;
```

```
    }  
    return true;
```

Q calculate simple interest based on principal Amount

```
function simpleInterest (principle, rate, time) {
```

```
  var finalAmt = principle + principle * time * rate;
```

```
  return finalAmt;
```

```
console.log (simpleInt (20000, 5, 2)) // 220000
```

Q calculate compound interest based on principal amount

```
function compoundInt (princi, rate, time) {
```

```
  var interest = princi * (Math.pow((1 + (rate/n)), (n * time)));
```

```
  return principle + interest;
```

```
console.log (compoundInt (20000, 5, 4)) // 304250
```