Java Script notes

* Variables
* By using the var keyword we can redeclare the variable and reassign value with same

Ex:- var x = 100

var x = 200

console.log(x)

Output: - 200

* By using the let keyword we cannot redeclare the variable with same name but we can re assign the value

Ex: - let x = 100

x = 200

console.log(x)

Output: - 200

* By using the const variable we can’t redeclare variable and we can’t reassign the value

Ex:- const x = 100

Console,log(x)

Output: - 100

* Objects
* to create an object, we use {} and give instructions

Ex: - let data = {

Full name = “Krishna Murthy”

}

Console.log (data [Full name])

Let data= {

Name: “BMW”

}

Console.log(data.name)

* #DEEP COPY#: - Deep copy will be affected where shallow copy will not be affected

Ex: - const obj2 = {...obj} //shallow copy

const obj3 = obj // Deep Copy

obj.age = 25

obj. tech = "nodejs"

console.log (Object.values(obj));

console.log(obj2)

console.log(obj3)

delete obj.age

console.log(obj)

* OPPERATORS
* + => Addition
* - =>Subtraction
* \* =>Multiplication
* = => equals
* == =>Compare the values
* === =>Strictly compare the datatypes
* ! = (or)!== => not equal to
* ++ => Increments
* -- => Decrementation
* >=== =>Greater then
* <=== =>Less then
* Logical Operators
* & logical AND checks all
* && logical AND Checks one
* If we are using && if any condition fails then it will not check rest and it will exit
* || logical OR
* | Logical OR just like &
* ! Logical NOT if this value is not there then it will come In loop
* Data Types
  + Number
  + String
  + Boolean
  + Object Datatypes
  + Arrays
* If we want to get to know the datatype then we can use the “type of” operator
* Arrays
* Array is also variable which can hold more than one value

Ex: - const number = [1,2,3,4,5,6]

* Array can hold multiple values and we can access those values by referring the index position

Console.log (arr [position])

* -- -- -- Length Method -- -- --

If we want to know the length of array, we can use this method

Ex:-console.log(arr.length)

* -- -- -- Push Method -- -- --

If we want to insert the value into the last index position, we can use Push Method

Ex: - arr. push(value)

* -- -- -- pop Method -- -- --

If we want to remove the last index position of array, we can use pop method

Ex: - arr. Pop ()

* -- -- -- Unshift Method -- -- --

If we want to add element into first index position of array we can use “unshift method”

Ex: - arr. unshift(value)

* -- -- -- Shift Method -- -- --

If we want to remove the first element from the array, we can use the shift method

Ex: - arr. shift(value)

* -- -- -- Concat Method -- -- --

If we wat to combine two single arrays to a single array we can use the Concat method

Ex: - const arr1 = [1,2,3,4,5,6]

Const arr2 = [7,8,9,10,11]

Const arrays = arr1.concat(arr2)

Console.log(arrays)

* -- -- -- for Each Method -- -- --

If we get multiple of array inside the array based on the satisfied condition we can able to delete / add index position

Ex: -const arr = [1,2,3,4,5,6,7,8,9,10]

arr. forEach((element, index) => {

if (element == 5) {

delete arr[index]

}

Console.log (element, index)

})

* -- -- -- for method -- -- --

Whenever we want to use the async and await keyword then we can use the below method

Ex: - for (let number(arr))

{

Console.log(number);

}

* -- -- -- Map Method -- -- --

Map method returns new array after performing function on each element of the array and it does

Ex: -Let numbers = [1,2,3,4,5,6,7,8,9]

Console.log (numbers. Indexof (1));

* -- -- -- Find Method -- -- --

if we want to find the particular matched element inside the array of element we use find method

Ex: -const data = [{name: ‘Krishna’, tech: ’Java’}, {name:” Naven”, tech:” node.js”}]

Let value = data. find (ele=>ele.name = ‘Krishna’)

Console.log(value)

* -- -- -- includesof Method -- -- --

If we need to check if any value is there in the array we use “includes of” if the value present, then it will show true and if it is not there then it will show false

Ex:- const arr = [1,2,3,4,5,6,7,8,9]

Console.log(numer.includes(4))

O/P:- True

* -- -- -- Sort Method -- -- --

By using sort method, we can able to arrange the numbers by ascending or descending order

Ex: - const = [2,4,3,1,5,7,4,8,6,9]

Console.log (number. Sort ())

* -- -- -- split method -- -- --

If we want to convert the string into array of elements then we can use split method

Ex: - let name = ‘WOW’

let name1 = name. split (“”)

console.log(name1)

O/P:- “W”,”O”,”W”

* FUNCTIONS: -
* A Function is a block of code designed to perform particular task. By using functions, we can reuse the functions
* When we pass the value into function is calling is called argument
* When we pass into function, we call it as parameter
* ANNONYMOUS: -
* The function without name is called the anonymous function
* FIRSTCLASS FUNCTION: -
* Node js is an open-source cross platform runtime environment
* Nodejs allows you to run javascript on the server satisfied
* By default, Nodejs will run in the asynchronous manner
* SET TIMEOUT FUNCTION: -

If we want to call the function particular time, we can use set timeout function

* SET INTERVEL FUNCTION: -

If we want to call the function multiple time, we can use set interval function

-------------------------------------------------------------------------

###” Await” expressions are only allowed within async functions and at the top levels of modules###

$$$if we want to execute the code in synchronous format async keyword$$$

------------------------HTTP METHODS-----------------------------------------------------------------------------------------------------

* POST METHOD: - when we want to insert any data in database, we use post method
* GET METHOD: - if we want to fetch any data from database, we use get method
* PUT METHOD: - if we want to update multiple fields then we can use put method
* PATCH METHOD: - if we want to update one or two fields then we use patch method
* DELETE METHOD: - if we want to delete any data then we can use delete method

----------------------------------------------------------------------------------------------------------------------------------Git Commands------------------------------------------------------------------------------------------------------------------------------

* If we want to know the current branch then we can use the command “git branch”
* If we want to know the changes in the current branch then we use the command “git status”
* If we want to commit the required files then we can use “git add (file path) / git add .”
* If we want to commit the new files then we can use “git add -f (file path)”
* If we want to commit the code in to the branch then we can use the “git commit -m “Random Message””
* If we want to push the code into the branch then we can use “git push”
* If we want to create a new branch from the one branch we use “git checkout -b <branch name>”

-------------------------------------------------------------------------------------------------------------------------------Event Loop--------------------------------------------------------------------------------------------------------------------------------------

Event loop allows nodejs to perform non-blocking input/output/ operations even if it is single thread data. Nodejs runs on a single threaded but it can take multiple request at a time and whatever the request gets completed those things returned parallelly. When we start the nodejs project it initializes the event loop

------------------------------------------------------------------------------------------------------------------------------EVENT LOOP DIAGRAM-----------------------------------------------------------------------------------------------------------------------

>> Timers

(EX: - Set time out functions)

>>Pending callbacks

(EX: - IPO operations)

>> Idle

(EX: - I/O events)

>>Checks

(EX: - set immediate functions)

>>Closed callbacks

(EX: - event emitters, socket. On)

\*Event loop executes the tasks from the call stack whenever the call stack whenever the call stack is empty then only it will take the task from the event cube\*

-----------------------------------------------------------------------------------------------------------------------------JSON Web Token--------------------------------------------------------------------------------------------------------------------------------

JSON Web Token

JWT token is used to validate the API’s by using the authentication method

JWT token will be generated by 3 parts 1st header 2nd Signature 3rd payload

By using JWT token we can protect the API’s

Ex: - "data": "eyJhbGciOiJIUzI1NiJ9.a3Jpc2huYUBnbWFpbC5jb20.Np9OmLfQj44QrBdHWc7jUnjl7Pkbvy9zoTLc4V-VACU"

---------------------------------------------------------------------------------------------------------------------------------PROMISES-----------------------------------------

--------------------------------------------------------------------------------------------

* In nodejs many operations are asynchronous(non-blocking)

Ex: fs.readfile for reading the data from the database

* Querying the database for avoiding the callback Hell will use promises

--------------------------------------------------------------------------------------------------------------------------------PROMISES.ALL--------------------------------------------------------------------------------------------------------------------------------

In the promise. All function all the functions will run parallelly

Promise. All method will be resolved when all the promises got succussed

If any one promise got failed it will be rejected immediately

-----------------------------------------------------------------------------------------------------------------------PROMISE.ALLSETTLED--------------------------------------------------------------------------------------------------------------------------------

Promise. allsettled all the function will run parallelly

Promise. allsettled will execute promises whether it might be resolved

Or rejected

---------------------------------------------------------------------------------------------------------------------------------PROMISE.RACE-------------------------------------------------------------------------------------------------------------------------------

Promise. race method will execute all the method parallelly

Promise. Race method it will return the first settled promise whether it might resolve or rejected

------------------------------------------------------------------------------------------------------------------------------------PROMISE.ANY-----------------------------------------------------------------------------------------------------------------------------

Promise. Any method will execute all the functions simultaneously

But the promise. Any will return the first resolved promise only

------------------------------------------------------------------------------------------------------------------------------------CLOSURE------------------------------------------------------------------------------------------------------------------------------------

>> closure gives the access of outer function variable to an inner function

>>closure remembers the outer function variable even after completing the execution of function

---------------------------------------------------------------------------------------------------------------------------CALLBACK FUNCTION----------------------------------------------------------------------------------------------------------------------------

>> callback function is a function which is passed as an argument to another function

>>In nodejs callback functions are widely used for asynchronous operations

Ex: reding “.txt” files

>>Instead of waiting for a task to finish nodejs continues executing the code and the call back function runs when the task completes

>>by using call back function we can prevent the bocking of code execution

---------------------------------------------------------------------------------------------------------------------- Steps to install setup---------------------------------------------------------------------------------------------------------------------------------

* VS CODE
* Goto browser and search: “ <https://code.visualstudio.com/download>”
* Hit enter and download based on your system configuration  
  if you are using mac download mac version  
  if you are using windows download windows version
* Wait for the download to finish and after downloading goto downloads in system and click on “VSCodeUserSetup-x64” file
* Give permissions and install
* MONGO DB
* Goto browser and search:” <https://www.mongodb.com/products/tools/compass>”
* Then Goto Products>>Community Edition>> then click download community
* It starts download
* After download goto downloads in system and click on “mongodb-windows-x86\_64-8.0.12-signed.”
* Then give permissions and install
* NODE JS
* Goto browser and search:” <https://nodejs.org/en/download>”
* Then click on “windows.msl” installer and download
* It starts download
* After download goto downloads in system and click on “node-v22.17.1-x64”
* Then give permission and install
  + - * COMMAND PROMPT STEPS
* Open cmd as administrator
* Type “sudo apt install nodejs npm” and hit enter  
  if sudo doesn’t support then goto settings and enable sudo in developer settings
* Check if npm installed or not by “npm -v”
* Type “npm install express” and hit enter
* Type “npm install cors” and hit enter
* Type “npm install mongoose” and hit enter
* Type “npm install dotenv” and hit enter
* Type “npm install body-parser” and hit enter
* Then create a package by using these commands
* Goto file manager and copy a path where the folder is created and then open cmd and type “cd path”
* Type “npm init” and hit enter
* Give details as follows  
  name: nodejs  
  version:   
  description: NodeJSproject  
  main:  
  scripts:  
  keywords: backend  
  author: your name  
  license:
* It sows something like this   
    "name": "nodejs",

  "version": "1.0.0",

  "description": "NodeJSproject",

  "main": "app.js",

  "scripts": {

    "test": "echo \"Error: no test specified\" && exit 1"

  },

  "keywords": [

    "backend"

  ],

  "author": "your name",

  "license": "ISC",

  "dependencies": {

    "body-parser": "^2.2.0",

    "cors": "^2.8.5",

    "dotenv": "^17.2.1",

    "express": "^5.1.0",

    "mongoose": "^8.17.0",

    "node": "^20.19.4"

  }

* Click Y and now your package is created
* Now type “code .” to open vs code in that folder
* MAVEN FILE: -

nodejs/

│

├── config/

│ └── database.js

│

├── constant/

│ └── responseConstant.js

│

├── controllers/

│ └── customerController.js

│ └──ProductController.js

│ └──OrderController.js

│

├── models/

│ └── CustomerSchema.js

│ └── ProductSchema.js

│ └──OrderSchema.js

│

├── routes/

│ └── route.js

│ └──ProductRoutes.js

│ └──OrderRoutes.js

│

├── utils/

│ └──JSONWebToken.js

│ └──Middleware.js

│

├── node\_modules/

│

├── .env

├──. gitignore

├── app.js

├── controller.js

├── package.json

├── package-lock.json

├── rough.js

└── rough.txt

--------------------------------------------------------------------------------------------------------------------NODEJS INTERVIEW QUESTIONS---------------------------------------------------------------------------------------------------------------------

Q. what is nodejs?

Ans. Nodejs is an open source cross platform javascript runtime environment that allows developers to execute javascript code outside of a web browser

Q. What is npm?

Ans. Npm is a node package manager which is also default package manager for javascript runtime environment

Q. what is event loop? And how it works?

Ans.

>>Event loop allows nodejs to perform non-blocking input/output/operations even if it is single thread data

>>it is a singular thread but it can take multiple request at a time and whatever the request gets completed those things returned parallelly

>>event loop executes the tasks from the call stack whenever the call stack is empty then only it will take the tasks from the event cube

Ex: -

Console.log(“10”)

setTimeout(() => {

console.log(“20”)

},500)

Console. log (“30”)

Q. what is difference between package. Json and package-lock.json?

Ans.

|  |  |
| --- | --- |
| Package. Json | Package-lock. Json |
| >> it defines a data related to project data ( Meta Data ) ,Scripts and declares the project’s direct Dependencies | >> this file ensures deterministic installations by “locking” the exact versions of all installed packages including sub-dependencies and their resolved locations. This guaranteed that npm will always produce the same node\_modules structure regardless of when or where its executed |
| >> it is developed manually by developers or created using “npm init” | >> it is automatically generated and updated by npm when npm install or other dependency modifying commands are executed |
| >> Contains info related to dependencies and their versions and info related to project | >> it is contains detailed hierarchical representation of the entire dependency tree including the version, source, and integrity hash for every package |

Q. what is difference between dependency and dev-dependency?

Ans.

|  |  |
| --- | --- |
| Dependency | Dev-Dependency |
| . A dependency is a library that a project needs to function effectively | . Dev dependencies are the packages that we need during the development |
| . If package doesn’t already exist in the node\_modules directory then it is automatically added | . As we install a package, npm will automatically install the dev dependency |
| . These are the libraries we need when we run our code | . These Dependencies are needed at some point during the development process, but not during execution |

Q. Fibonacci series code?

Ans.

Function feb (n) {

If (n<=1) {

Return [];

}

If (n===1) {

Return [0];

}

Const arr = [0, 1];

For (let i=2; i<n; i++) {

Arr.push(arr[i-1] + arr[i-2]);

}

Return arr;

}

Console.log (feb (10));

Q. what is use of “type of” operator in javascript?

Ans. If we want to get to know the data type then we can use “type of” operator

Q. difference between var, let & const?

Ans. VAR: declares variable with function or global scope and allows re-declare and update within the same scope

LET: Declare variable with block scope allowing updates but not re-declaration within the same scope

CONST: Declare block-scoped variables that can’t be re-assigned after their initial assignments

Q. what is callback function?

Ans. It is a function which is passes as an argument to another function. In nodejs callback functions are widely used for asynchronous operations like reading ‘.txt’ files.

Instead of waiting for a task to finish nodejs continues executing the code and the callback function runs when the task completes and also by using function we can prevent the blocking of the code execution

Q. what is closure?

Ans. Closure gives the access of outer function variable to an inner function. Closure remembers the outer function variable even after completing the execution of a function

Q. what is difference between promise. All and promise. All settled?

Ans.

Promise. All:

=>promise. All all functions runs parallelly and in promise. all it executes if all the methods get resolved and If any one is failed then it will give error

Promise. all settled:

=>in promise. all settled all the functions runs parallelly and in promise. All settled it executes whether all the methods get resolved or rejected it executes all the functions

Q. Write the HTTP Status Codes?

Ans.

200 – OK

201 – CREATED

202 – ACCEPTED

204 – NO\_CONTENT

400 – BAD\_REQUEST

401 – UNAUTHORIZED

403 – FORBIDDEN

404 – NOT\_FOUND

409 –CONFLICT

500 – INTERNAL\_SERVER\_ERROR

502 –BAD\_GATEWAY

503 –SERVICE\_UNAVAILABLE

Q. what are the array methods?

Ans. Array is also a variable which can hold more then one value

Position method:

array can hold multiple values and we can access those values by referring the index position (ex: console. Log (arr. length))

push method:

if we want insert the value into last index position, we can use the push method (EX: arr. push(value))

pop method:

if we want to remove the last index position of an array, we can use pop method (EX: - arr. Pop ())

unshift method:

if we want to add the value to the first index position of the array we can use "unshift method"(EX: - arr. Unshift (value))

shift method:

if we want to remove the first element form the array then we can use "shift method" (EX: - arr. Shift ())

Concat method:

if we want to combine the two single arrays into one single array then we can use Concat method (EX: - const arrays = arr1.cocnat(arr2))

forEach method:

if we get multiple of array inside the array based on the satisfied condition we can able to delete/ add particular index position

for method:

whenever we want to use the async and await keyword then we can use for method (EX: - for (let number of arr){console.log(number)})

map method:

map method returns new array after performing function on each element of the array and it does

indexof method:

by using index of Method, if we need to check if any value is there in the array we can use "includes of" method if the value present, then it will show true and if it is not there then it will show false

find method:

by using find method, if we want to find the particular matched element inside the array of element we use find method

includes of Method:

if we want to find the array which include the given value then we use includes of Method

sort method:

by using sort Method, we can be able to arrange the numbers by ascending or descending order

split method:

if we want to convert the string into array of elements then we can use split method

Q. What are the MongoDB operators?

Ans.

POST Method: When we want to insert any data in database we use post method

GET Method: when we want to receive any data from the database we use get method

PUT Method: if we want to update multiple fields then we use put method

PATCH Method: if we want update one or two fields then we use patch method

DELETE Method: if we want to delete any data then we can use delete method

Q. What are the difference between Shallow Copy and Deep Copy?

Ans.

|  |  |
| --- | --- |
| SHALLOW COPY | DEEP COPY |
| Copies only the first level of the object/ array | Creates a completely independent copy of the object/arrays, including all nested levels |
| Nested objects/arrays are still references to the original | No references are shared with the original |
| Changes in nested values affect both the original and the copy | Changes in the copy do not effect the original |