Node.js

Software’s requirement

1. Node.js
2. VS Code

Javascript is used to develop websites that adds interactivity to achieve dynamic behaviour.

Javascript fundamentals

1. Functions
2. Variables
3. Operators
4. Loops
5. Conditional Statements
6. Objects
7. Arrays

Variables: You can use let or const keywords to create variables, avoid using var keyword

var x = 20; // not recommended

let y = 30; // block scoped, but you can modify let variables

const z = 40; // block scoped, but you can’t modify const variables

Functions: In Javascript you can create functions using function keyword

function getUsers() {

…..

}

function multiply(a, b) {  
 return a \* b;   
}

multiply(20, 30);

Arrays, forEach & map



forEach: It is only to iterate

map: it converts an array to another array.

Handling events in Javascript

DOM (Document Object Model) generates some events when you interact with the HTML element, then you can perform some task for those events which is called as event handling, you can call some functions based on the events.

document.getElementById(“id”): This is used to access an HTML element based on the id.

document.getElementsByTagName(“tag”): This is used to access all the elements having same tag name, it returns array of elements

document.getElementsByClassName(“class”): This is used to access all the elements having same class name, it returns array of elements

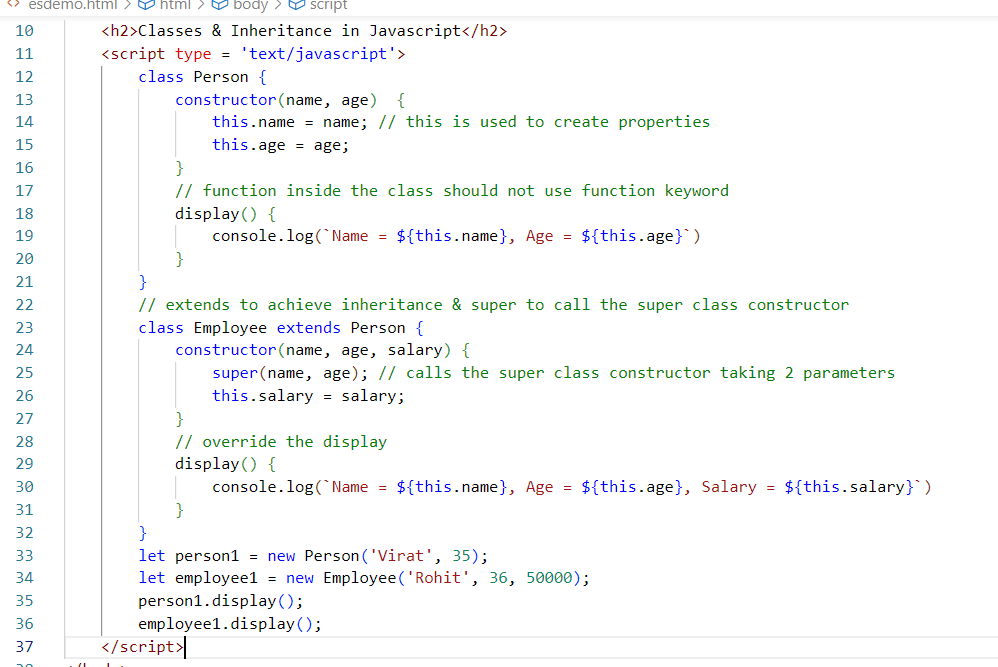
document.querySelector(“id or class or tag name”): This is also used to access the elements having either an id or class name or tag name

New features of Javascript

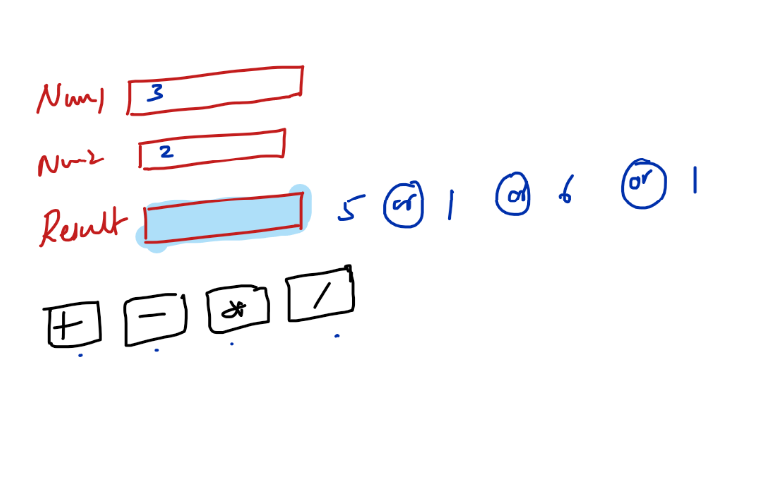
ECMA Script is the standard to specify the features for the Javascript, in ES6 they introduced lot of changes in the syntax to simplify the javascript writing

1. let, const keywords
2. class, super, extends keywords
3. arrow function – simplified form of writing the callbacks
4. template strings – back tick with ${expression}
5. rest & spread operators – to work with multiple parameters in an easy way
6. generators – to get multiple values from the function without completing the function execution
7. optional chain
8. exponential operators

Classes & Inheritance in new feature



Activity



Arrow functions

It simplifies writing callback functions by removing lot of things in the callback like function keyword, { } if its one line statement, return statement if its one line statement

Callback:

function(a, b) {   
 return a + b;  
}

Arrow function

(a, b) => a + b

(or)

(a, b) => { return a + b; }

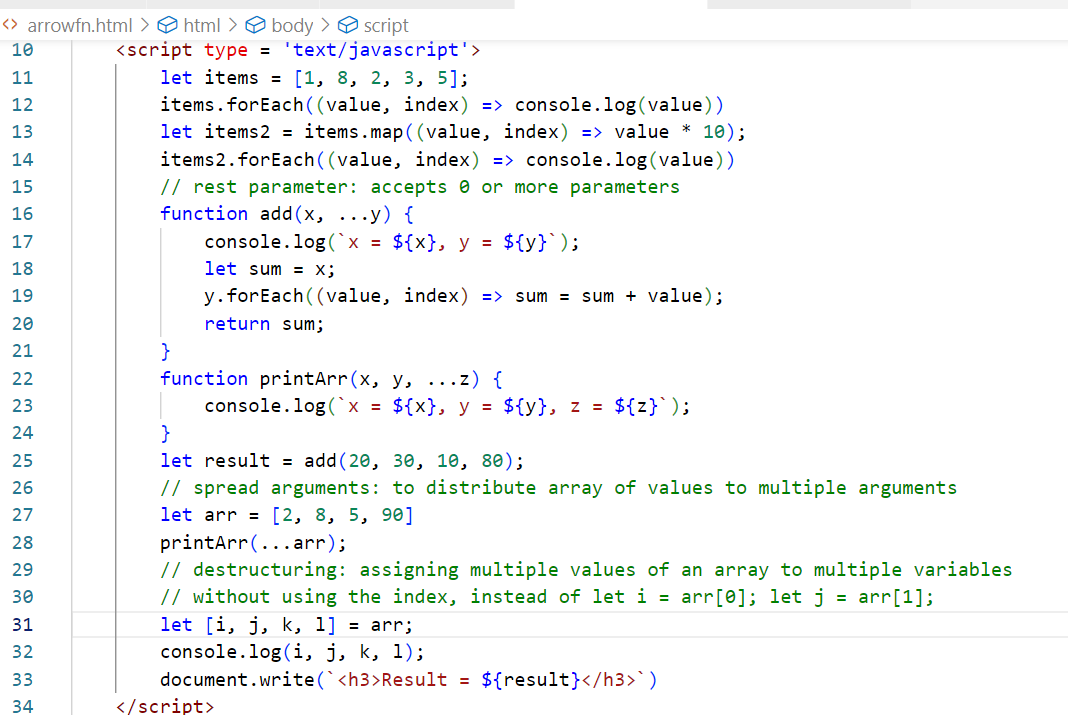
Using in forEach or map function

let items = [2, 1, 5, 4, 3];

items.forEach( (value, index) => console.log(value); }

let items2 = items.map( ( value, index) => value \* 10);

Rest, Spread & Destructuring



Rest: to pass 0 or more values

Spread: to distribute multiple values to multiple parameters

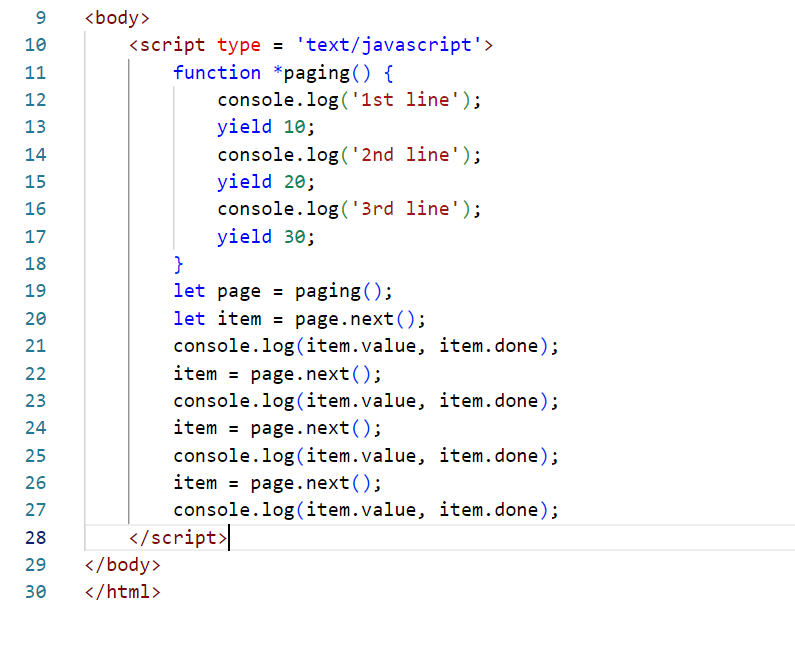
Destructuring: to assign values to multiple variables in a single line

Generators:

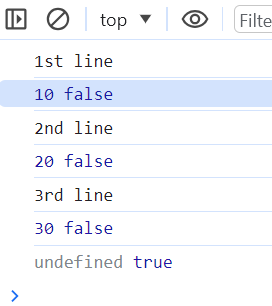
They can return multiple values on a single function call on installment

function \*foo() {  
 calling backend   
 yield data;  
 calling backend  
 yield data;  
 calling backend  
 yield data;  
}

let f = foo();  
f.next();  
f.next();  
f.next();



Output:



Optional chain

It is to access the nested property in an easier without checking for undefined condition.



Exponential operator

2 \* 3 = 6

2 \*\* 3 = 8

2 \*\* 4 = 16

Asynchronous actions

It initiates the action now but finishes later

setTimeout(callback, timing);

Ajax calls using XMLHttpRequest : call the backend api’s

let xhr = new XMLHttpRequest();

xhr.send( “api/posts” );

xhr.onreadystatechange = function() {   
 // updates the document when the response comes  
}

Promise: It is used to perform an asynchronous operation whose result could be either successful or failure

promise  
 .then( callback )  
 .catch( callback )

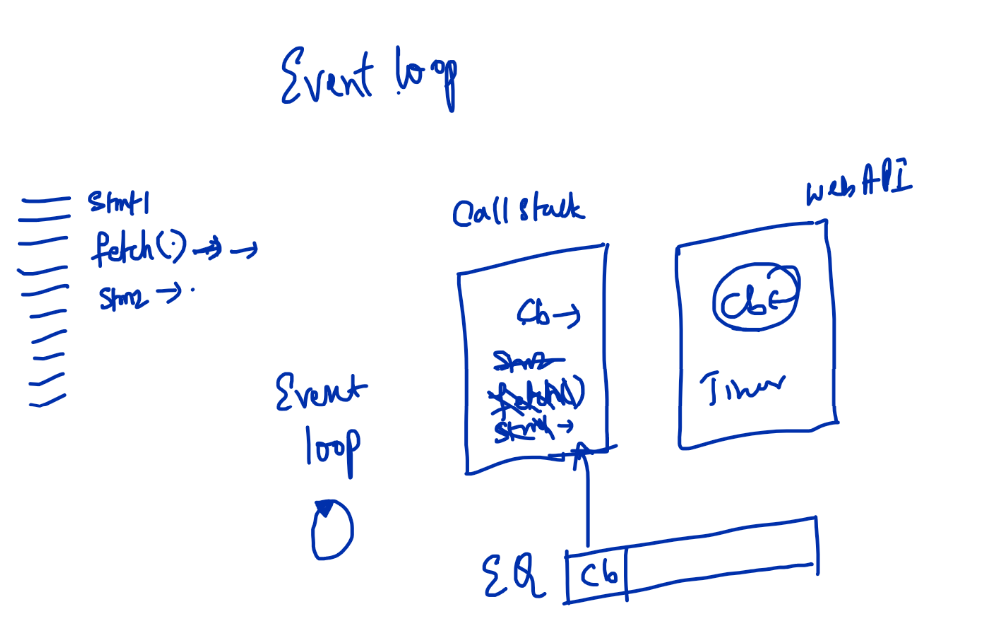
Node.js

* It provides a runtime environment to run the javascript without using browser settings
* It can be used as a tool for any kind of project

Benefits of Node.js

1. It helps to create any kind of project
2. It provides an HTTP environment so that you don’t need any external servers to host a website, earlier developers used external servers like apache tomcat, jboss, weblogic
3. Front-end developers can easily switch to write backend programs using node.js without learning a new programming language
4. It can interact with various databases like SQL & NoSQL
5. It provides NPM (Node Package Manager) from which you can download third party javascript libraries
6. NPM also provides libraries for many popular front-end technologies like React.js, Angular, Vue.js
7. Node.js can be used to develop a Fullstack application using a common language Javascript, popular fullstacks like MEAN, MERN, MEVN all use Node.js as an environment
   1. MEAN – MongoDB/MySQL, Express, Angular, Node
   2. MERN – MongoDB/MySQL, Express, React, Node
   3. MEVN – MongoDB/MySQL, Express, Vue.js, Node
8. Node.js uses Non-Blocking asynchronous callbacks to perform IO operations like reading from the network, accessing a database or a file system, instead of blocking the thread and wasting CPU cycles waiting.

Event Loop architecture



Step1: Main code is pushed to the callstack

Step2: stmt1 is pushed to the callstack & will be cleared

Step3: fetch() is pushed to the callstack & will be cleared, however it will have a callback that is handled in the WebAPI

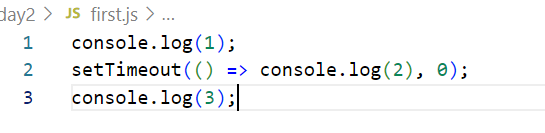
Step4: stmt2() is pushed to the callstack & will be cleared

Step5: If the asynchronous operation is completed then it will be added to the event queue and will be pushed in FIFO basis

Note: Node.js is different than the browser, because it executes javascript in the backend, you can’t use any inbuilt objects of browsers in node.js like, document, window, alert, sessionStorage, localStorage

To run the javascript you can use : node filename.js

first.js



Here the callstack executes all the 3 lines one by one, then the callback of setTimeout is added to the event queue & pushed to the callstack for execution, because of that we see callback function console.log() later.

REPL: Read Eval Print & Loop: It is a CLI provided by node.js to run some scripts to test the code, Mongodb uses this REPL in its terminal to enable admins to directly interact with the database using javascript functions

package.json: It is an heart of node.js, it contains the entire application configurations like dependencies, commands, versions and so on.

npm init: This creates package.json

npm init -fy: This also create package.json, but it doesn’t ask information of your project

Node.js modules

Modules are reusable code that you can use in any javascript file, you can reusable objects, functions, classes, variables by importing & exporting, in Node.js we have 3 types of modules

1. Local module: Which is created within the project
2. Core module: These are part of Node.js
3. Third party module: These are not part of Node.js or Local module, they are downloaded from the internet.

Local Modules: These are some variables or classes or functions which you can export in one file & import in another file

a.js

module.exports.add = function(x, y) {   
 return x + y;  
}  
module.exports.sub = function(x, y) {   
 return x – y;  
}

b.js

let add = require(‘./a.js’).add;

let sub = require(‘./a.js’).sub;

add(20, 30);

sub(30, 40);

The above code in a.js exports add & sub, and b.js imports them, however the syntax is little bit complex because of old javascript syntax, we can use export & import in an easier way but we need to add type : module in package.json, then we can write a.js & b.js as

a.js

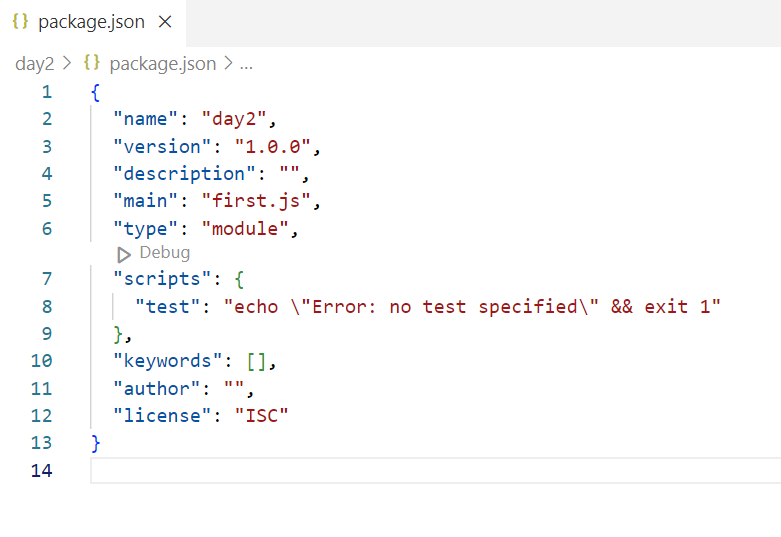
export function add(x, y) {   
 return x + y;   
}  
export function sub(x, y) {   
 return x – y ;   
}

b.js

import { add, sub } from ‘./a.js’;

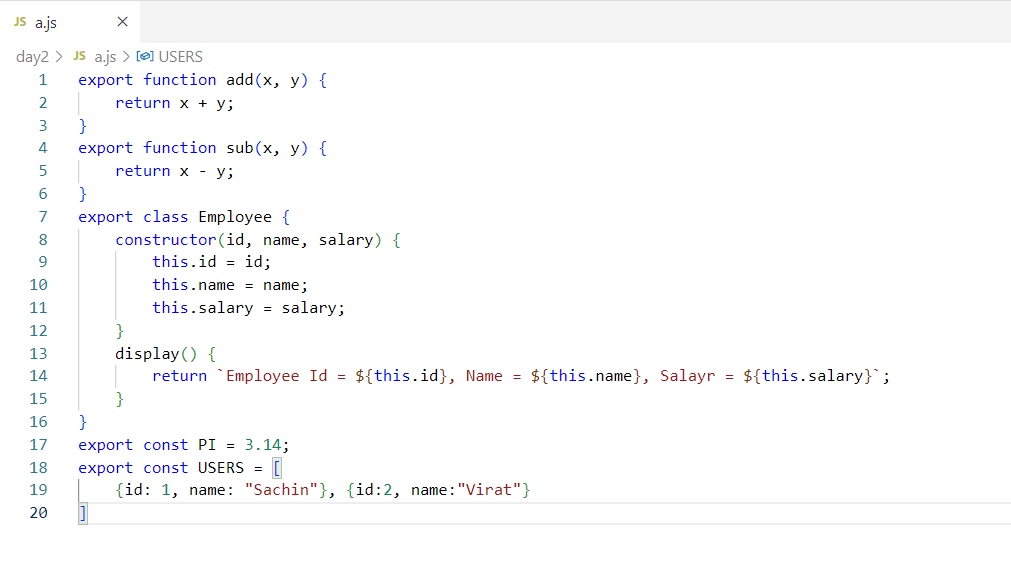
// aliasing: if we have same function in 2 or js file

import { add as a} from ‘./a.js’;  
import { add as b } from ‘./x.js’;  
a(); // calls add of a.js  
b(); // calls add of b.js



“type”:”module”: to make use of newer import & export syntax.

a.js: to export functions, variables, arrays & classes



Core Modules: In node.js there are many core modules which you can import, like

* os: It provides platform details like architecture, platform, version and so on
* http: It is to create server
* fs: It is to interact with the files to read/write data
* url: It is to query the url string when you have a client & server programs

OS Module



Third party module

These modules we need to install using npm

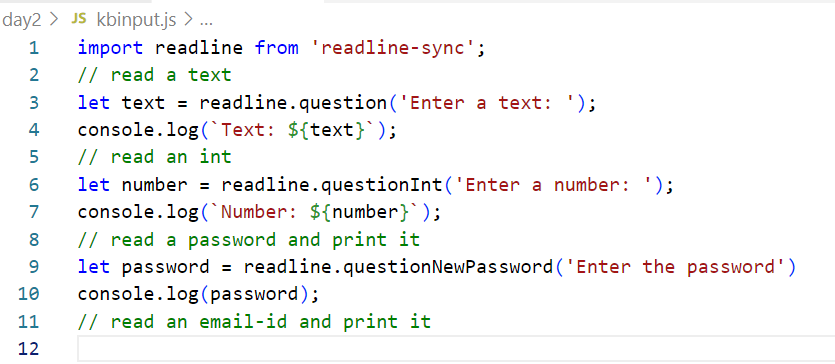
readline-sync: It is to take user input in a terminal based application

npm install readline-sync --save

or

npm i readline-sync

This gives various methods to read int, String, password, email and so on.



Array methods:

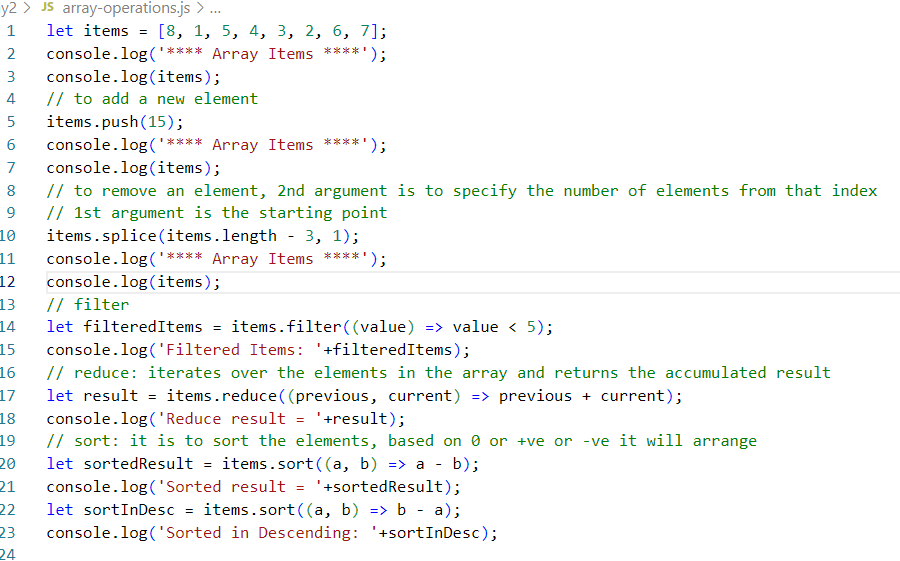
1. forEach: Iterating
2. map: transforming to another array
3. push: to add a new element in an array
4. splice: to remove an element from an array
5. filter: iterates and creates a new array by filtering on some condition
6. sort: iterates & sorts the array
7. reduce: iterates & accumulates the iterated value in a single result

Note: You can apply these methods on simple or complex types

filter(callbackFn(item)): It is used to iterate the elements and filter the element on a condition and store it in another array

reduce(callbackFn(previous, current)): It iterates the elements in the array and returns the accumulated result, and the accumulated result is provided as an argument in the next call to the callback function

sort(callbackFn(a, b)): It sorts based on the numbers compared with a & b, it could be +ve, -ve or 0



Activity:

Create 2 files

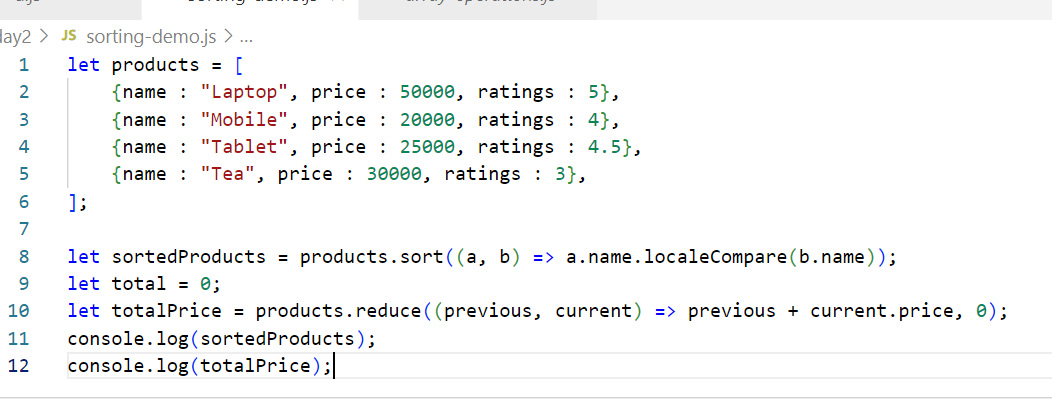
1 file will have an array to store product details like name, price, ratings, discount, all these properties must be part of a Product object, in the same file create methods that can perform following operations

1. store(product): Should accept a product object & store in an array
2. getAllProducts(): Should return all the products to the client in a sorted order based on the name
3. getAllProductsByPrice(order): Should return all the products in a sorted order by price based on the input, if order is asc then return in ascending order else return in descending order
4. getAllProductsByRatings(order): Should return all the products in a sorted order by ratings based on the input that could be either asc or desc
5. getTotalPrice(): Should return total price by accumulating all the product price

another file must call these 5 methods by taking input at runtime, these 5 methods keep in a loop such that based on some input you can call these 5 methods ex: 1: store 2: getAllProducts 3: getAllProductsByPrice 4: getAllProductsByRatings, 5: getTotalPrice, 6: Exit you must exit the loop on the input 6, until you can be in a loop to perform all the above operations.

Note: Create Product class with 4 parameters constructor that takes name, price, ratings & discount

How to sort when the data is in string form & How to calculate the total price of all the products



File Handling

fs module is present in node.js to read/write files, there are two types of methods you get

1. Synchronous read & write methods
2. Asynchronous read & write methods

import fs from ‘fs’;

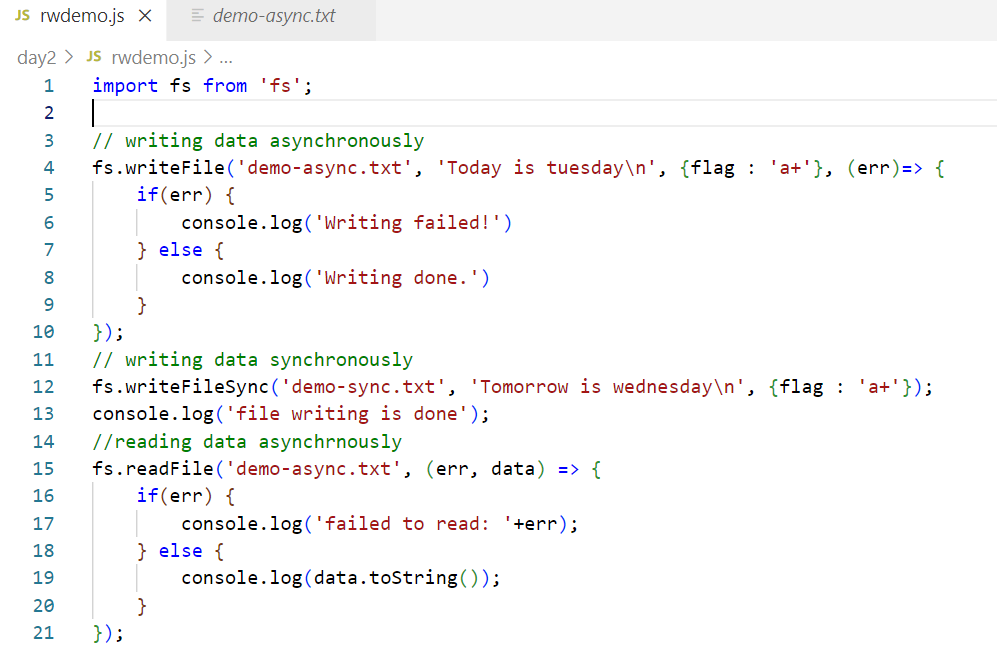
// Synchronous read/write

fs.writeFileSync(‘filename’, data);   
fs.readFileSync(‘filename’); // returns the buffer stream/byte/binary data which you can convert it into string

// Asynchronous read/write

fs.writeFile(‘filename’, data, callback);

fs.readFile(‘filename’, callback)



Reading/Writing JSON

