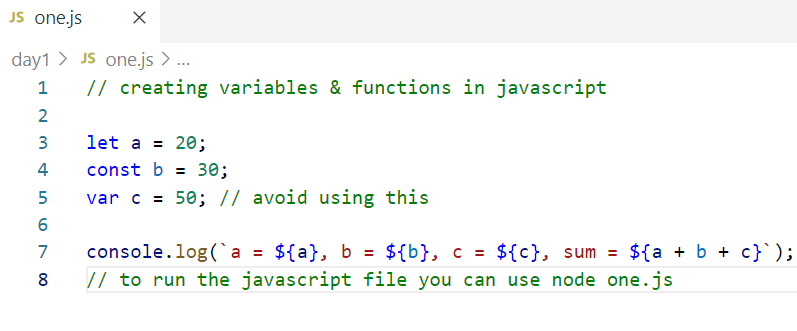
Node.js Training

Node.js: It is a runtime environment used to run Javascript programs at the backend so that you can access various backend resources likes OS, Files, Databases

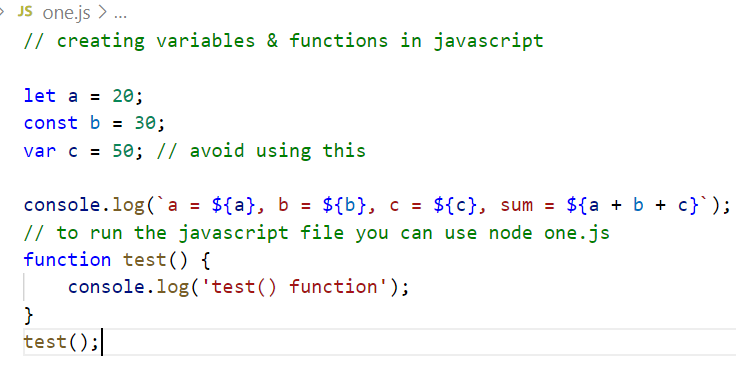
REPL Terminal: It is a node.js terminal which helps you to quickly test your javascript code for simple calculations

REPL stands for Read Evaluate Print and Loop

Understanding variables



Understanding functions



Hoisting:

Javascript loads all the function definitions and var variables at the top of the script hence you can access them before declaring them, however the initializations wouldn’t hoisted

i.e.,

console.log(x)

var x = 30;

The above line only declares x, but doesn’t initialize ‘x’;

which is why you get undefined,

Note: let & const wouldn’t be hoisted you will get error if you access a variable before declaration.

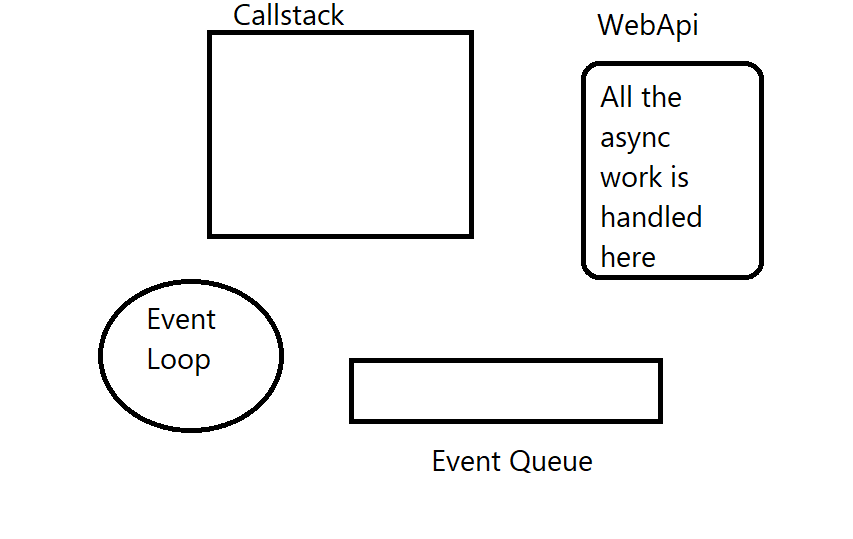
You can create javascript functions as a value but you can’t access it before if its assigned as a value.

Callback functions:

These functions are initialized first but called later based on some events like response from the server or events from some element or some actions or if timer is over, it is used mainly to perform asynchronous tasks.

Event loop mechanism

Node.js is a single threaded model, it runs scripts with only one thread using the callstack.



Once the asynchronous work is over the task will be added to the event queue and the task present in the event queue is loaded by event loop to the callstack for execution only if the call stack is empty, till that time the task will be there in the event queue.

Arrow functions: These are alternate way of writing callbacks

Note: Callbacks can also be synchronous, ex: the callbacks of forEach, map, filter and so on.

Writing arrow functions instead of callback

|  |  |
| --- | --- |
| Callback | Arrow function |
| function() {   return 20; } | () => 20; |
| function(x, y) {  return (x + y);  } | (x, y) => x + y |
| function(x, y) {  console.log(x + y);  return (x+ y);  } | (x, y) => {   console.log(x + y);  return (x + y); } |
| function() {  console.log(“hello”); } | () => console.log(“hello”) |
| function(x) {  return x + 20; } | x => x + 20 |

Rest & Spread operator

Rest operator accepts 0 or more arguments

function foo(x, y, …z) { }

Here …z is a rest parameter which can accept 0 or more arguments

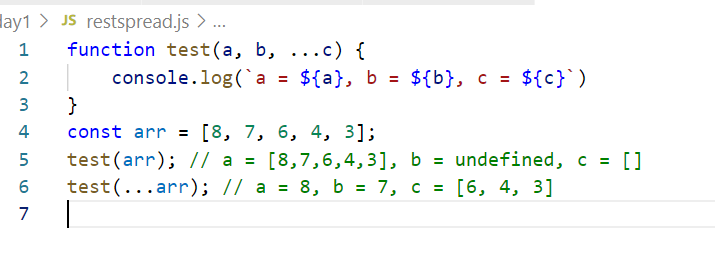
Note: You can have maximum one rest parameter & it must be the last parameter

Spread operator spreads value to multiple parameters, like each value present in the array can be assigned to separate parameters

function foo(x, y, z) { }

arr = [5, 1, 2];

foo(…arr); 5 is passed to x, 1 to y & 2 to z.



Using Rest parameter create a function that can accept a key & an array, the function must iterate the array & find the key is present or not

ex:

key = 2;

arr = [7, 8, 2, 3, 5]

search(key, …arr) // should print 2 is present

key = 9;

arr = [7,8,2,3,5]

search(key, …arr) // should print 9 is not present

== & === operator

== checks only value but not the type

=== checks both value & type

Javascript objects

let emp = { name : “Raj”, salary: 30000 }

Nested objects

let emp = { name : “Raj”, salary: 30000, address : {state : “KA”, city : “BLR”} }

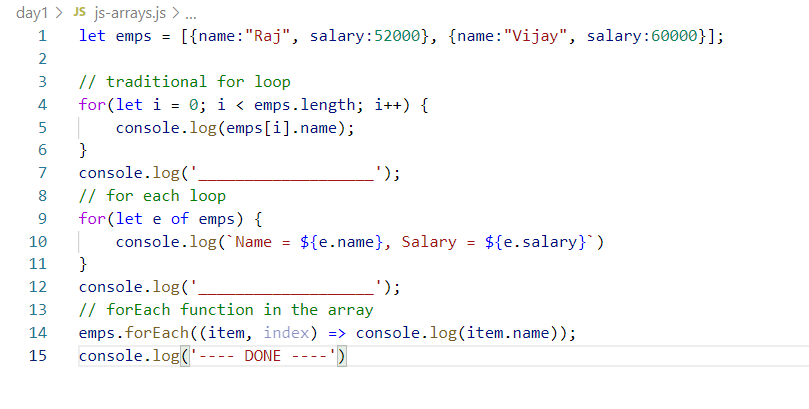
emp.address.state to access state

Javascript object array

let emps = [ {name: “Raj”, salary: 35000}, {name: ”Vijay”, salary: 40000 } ]

emps[0].name to access name of the 1st element

emps[1].name to access name of the 2nd element



Array methods: filter, map, sort, reduce, forEach, splice

forEach: It is used to iterate the elements

map: It is used to iterate the elements & generate a new array from the iterated elements, it is mainly to convert from one type to another type

filter: It is used to iterate the elements and generate a new array that matches to specific condition

sort: it used to iterate the elements & sort based on certain conditions like sort based on price, sort based on ram size

reduce: It used to iterate the elements and reduce it to a single result

splice: it used to remove the elements from the array.

Activity: