Javascript & React.js

Software’s requirement

* VS Code
* Node.js

Javascript: To add interactivity to your website, things you can do with javascript

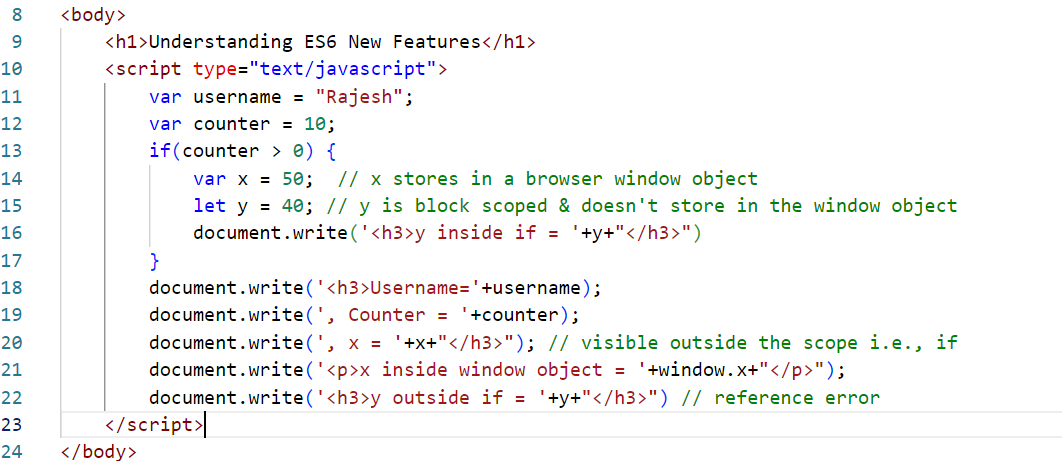
* Input validation
* Event Handling
* Pop-up boxes

New features of Javascript (ES6 new features)

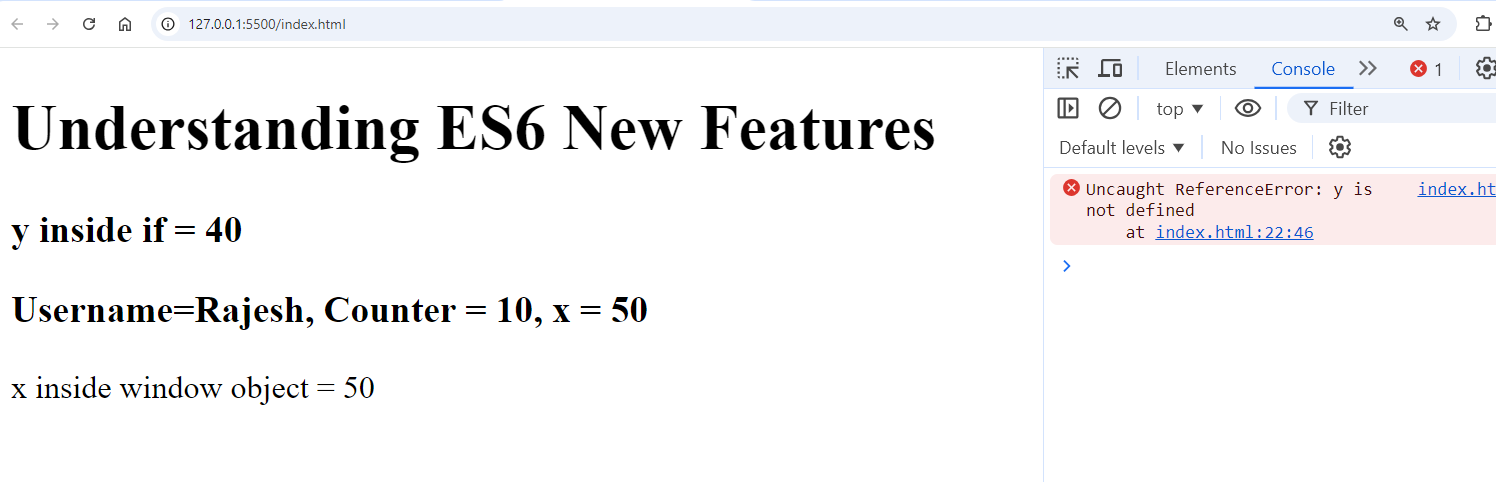
Helps to easily write the javascript code, it added lot of modern language syntax

* Keywords like let, const, class, super, extends
* Template strings : ``
* Arrow functions
* Spread & Rest Operators
* Promises
* Async / Await
* Optional Chain

let & const: They are used to create variables, but earlier javascript used var



Output:



let variables you can modify, const variables you can’t modify

let x = 20;

x = 30; // ok

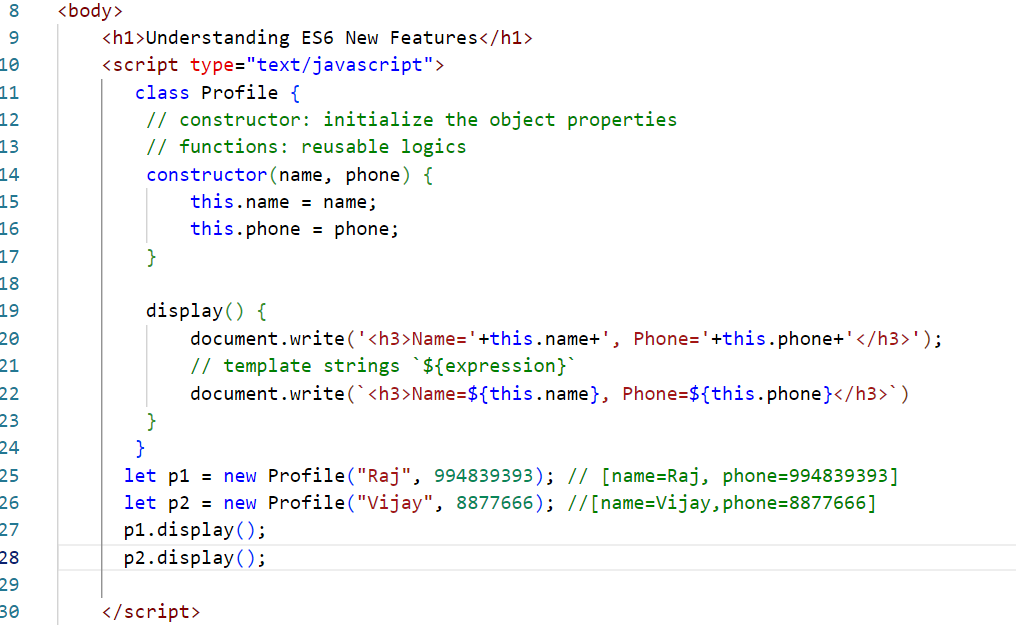
const y = 50;

y = 55; // error

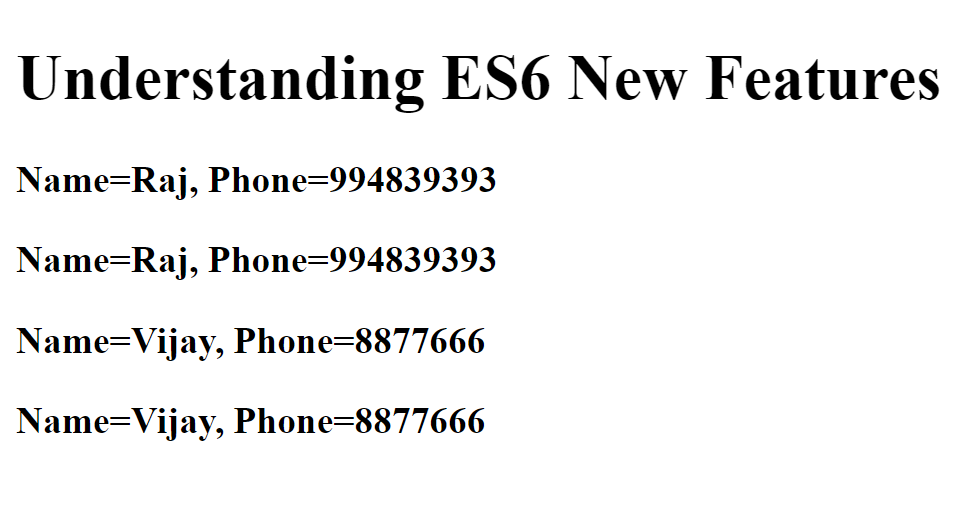
class:

It is a blueprint of an object, which tells the structure of the object.

Understanding the classes & objects

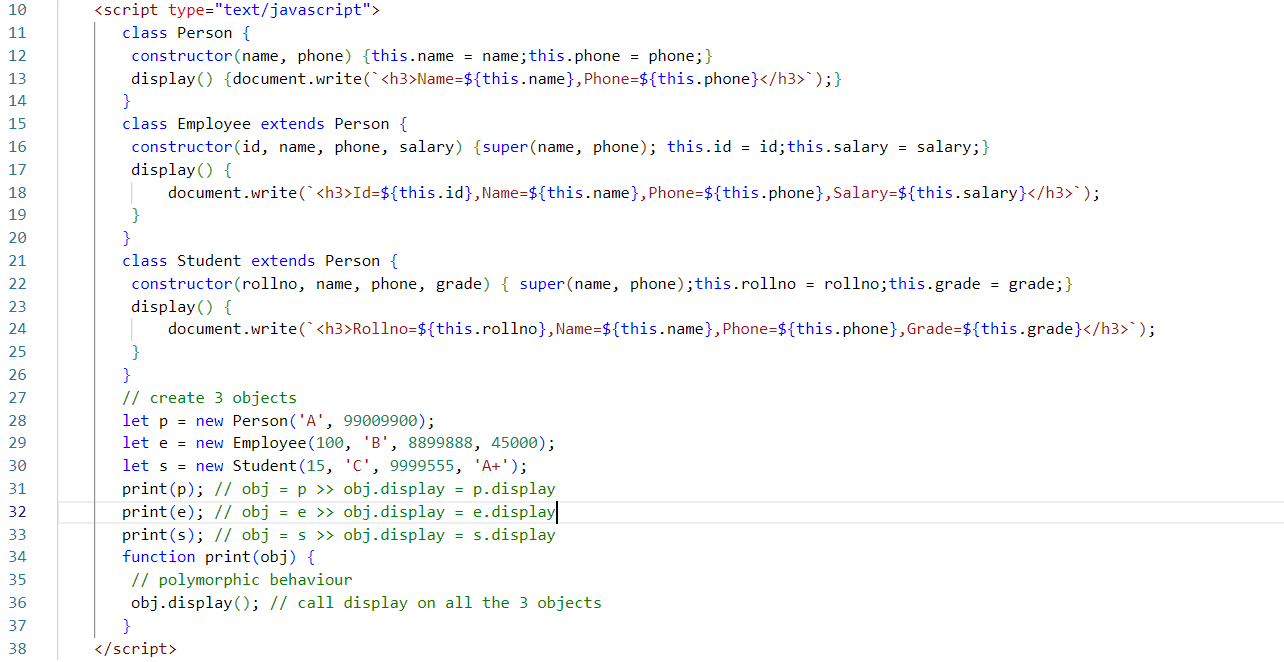


Output:

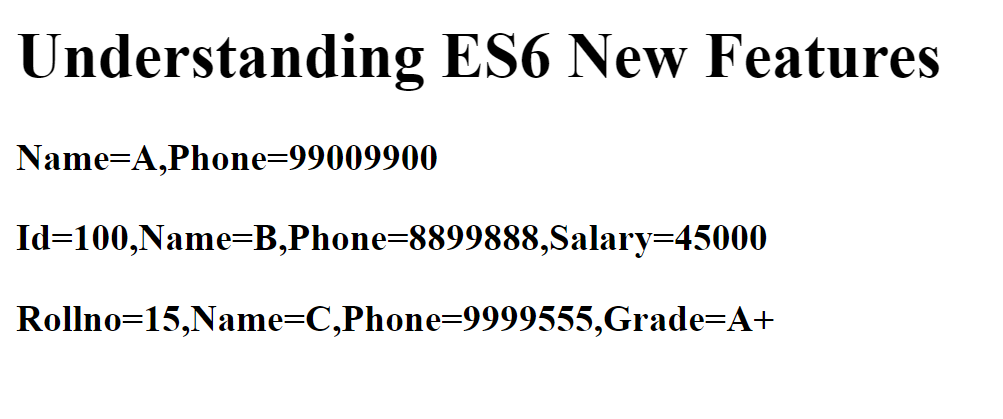


Inheritance

Process of acquiring the properties & behaviors from one object to another object.

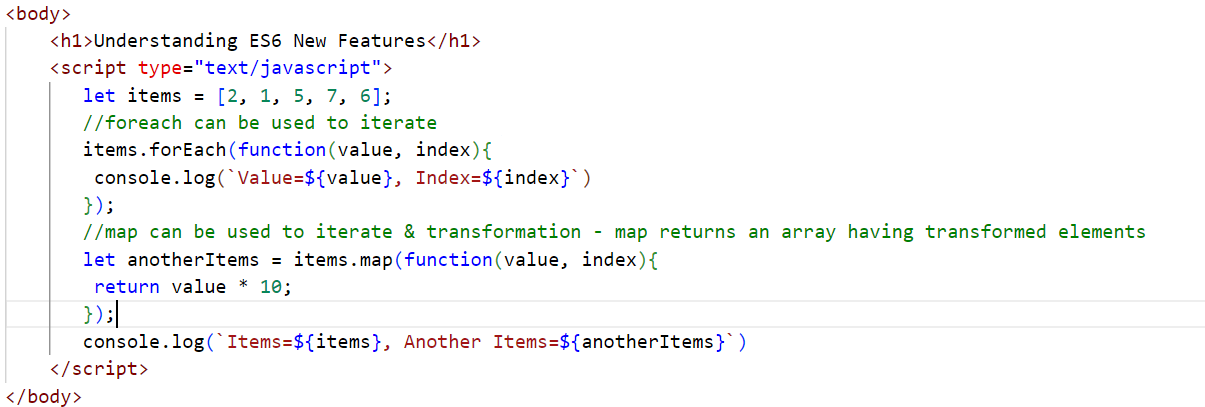


Output:

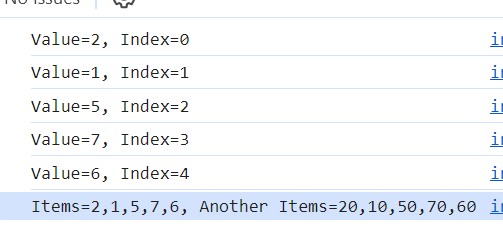


Callback functions

These are the functions which are invoked later depending on some events, like response from the server, events from user interaction, when timeout happens



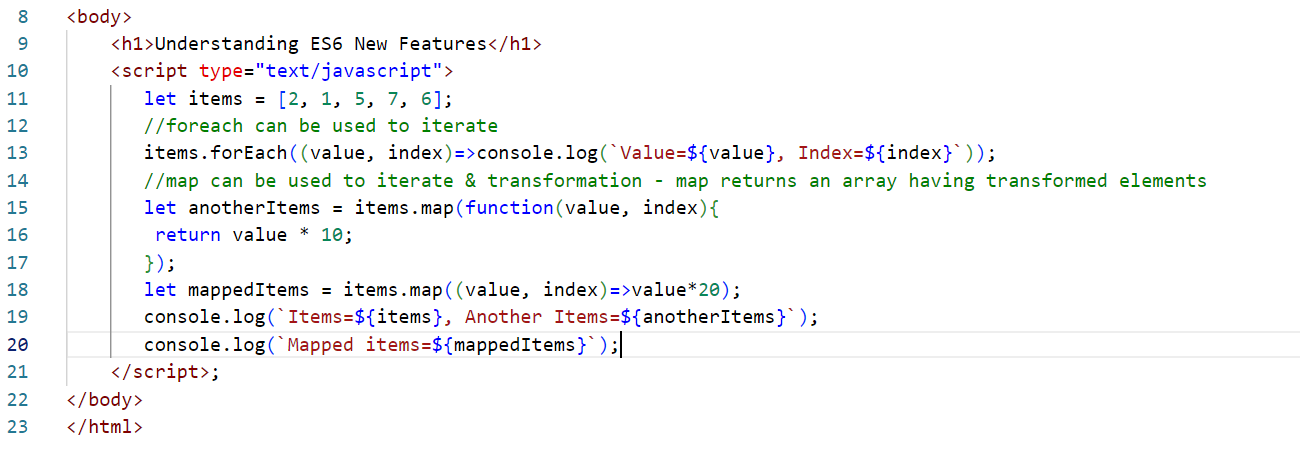
Output:



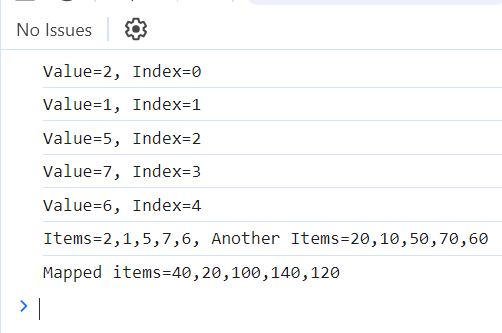
Arrow functions: These are the alternate form for the callbacks, it also simplifies the syntax

Callback:  
function(x, y) {   
 return x \* y;  
}  
Arrow function:  
(x, y) => { return x \* y ; }  
(x, y) => x \* y;

(x, y) => console.log(x, y); // this doesn’t return any thing



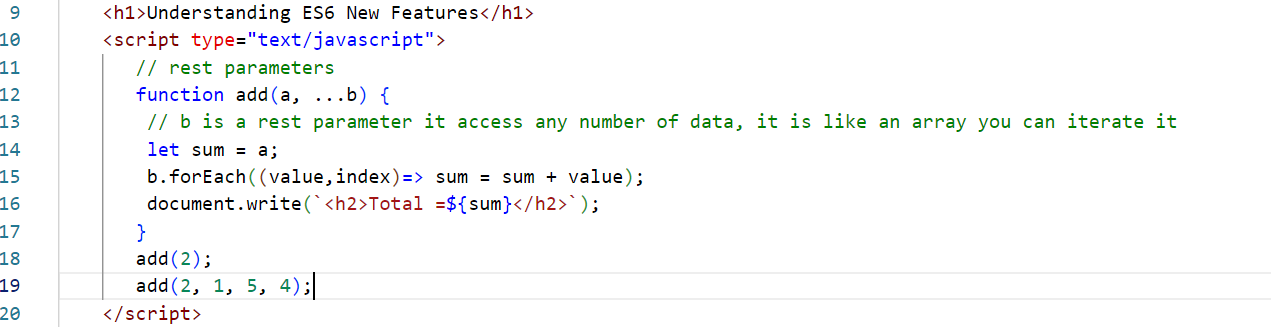
Output:



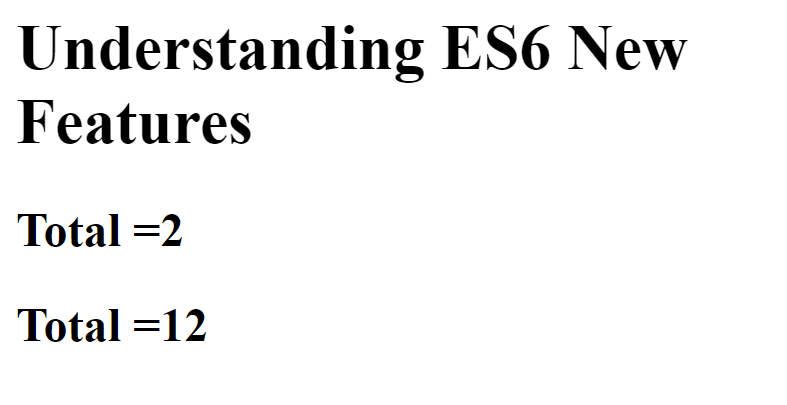
Rest & Spread operators

function add(a, …b) {   
}

Rest parameter can accept 0 or more elements



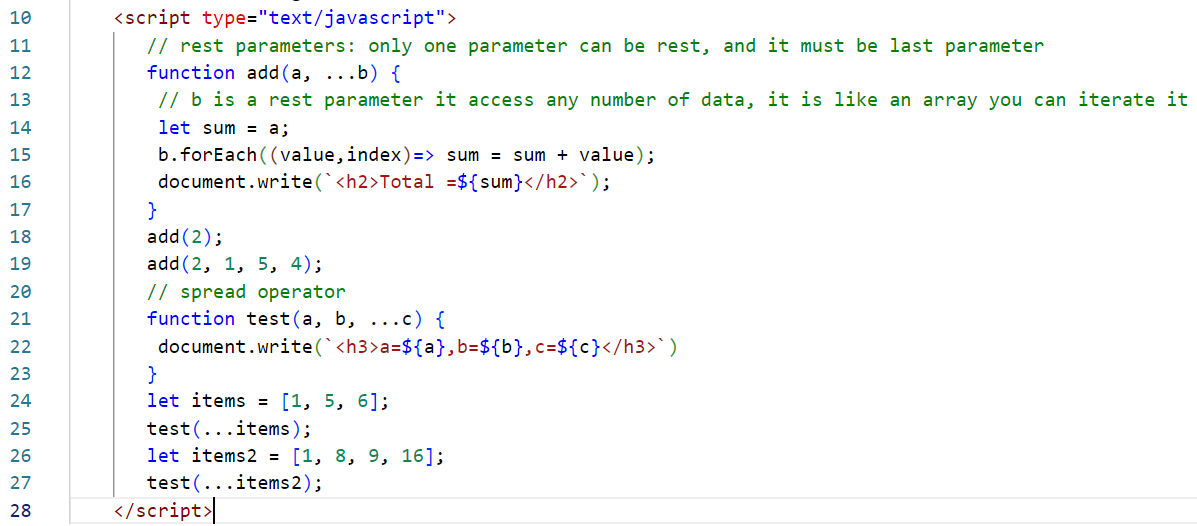
Output:



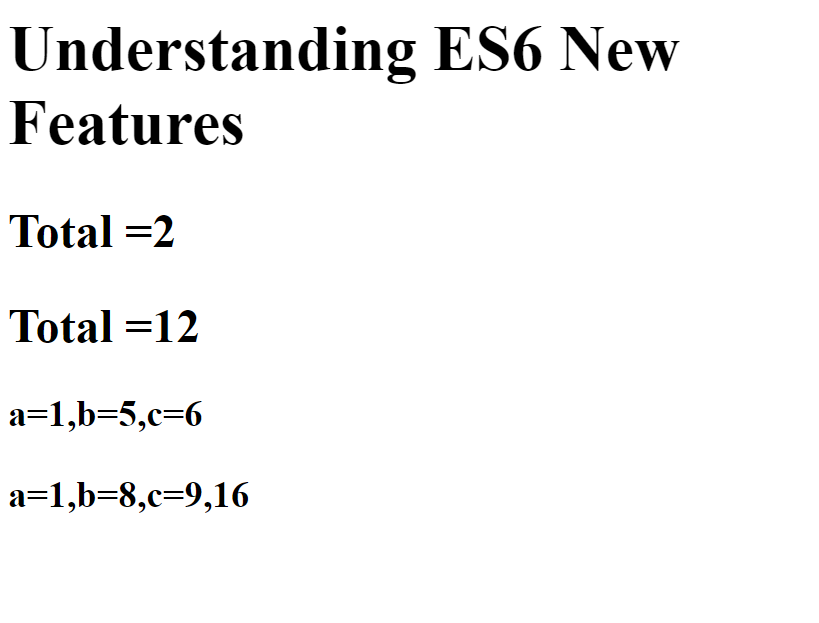
Spread parameter:  
It helps in distributing the elements into multiple parameters

let items = [2, 5, 6];

function test(a, b, c) { }  
test(…items); // a = 2, b = 5, c = 6



Output:



Destructuring: This helps in unwrapping the elements in the objects or arrays into another variables.

* Object Destructuring
* Array Destructuring

old approach

employee = {id:100, name: ”Rajesh”, salary:45000, email:”raj@g”, address : {st:”ka”,ct:”bl”} } ;

let id = employee.id;  
let name = employee.name;  
let salary = employee.salary;  
let email = employee.email;

New approach: Object destructuring

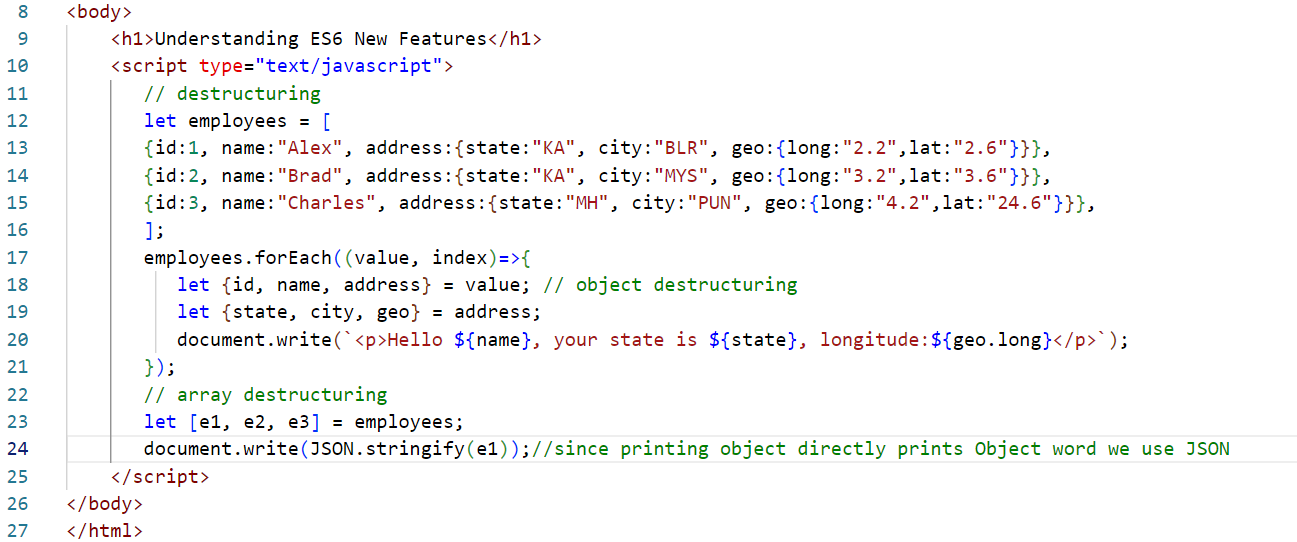
let { id, name, salary, email, address } = employee;   
let {st, ct } = address;  
console.log(st); // ka

Array Destructuring

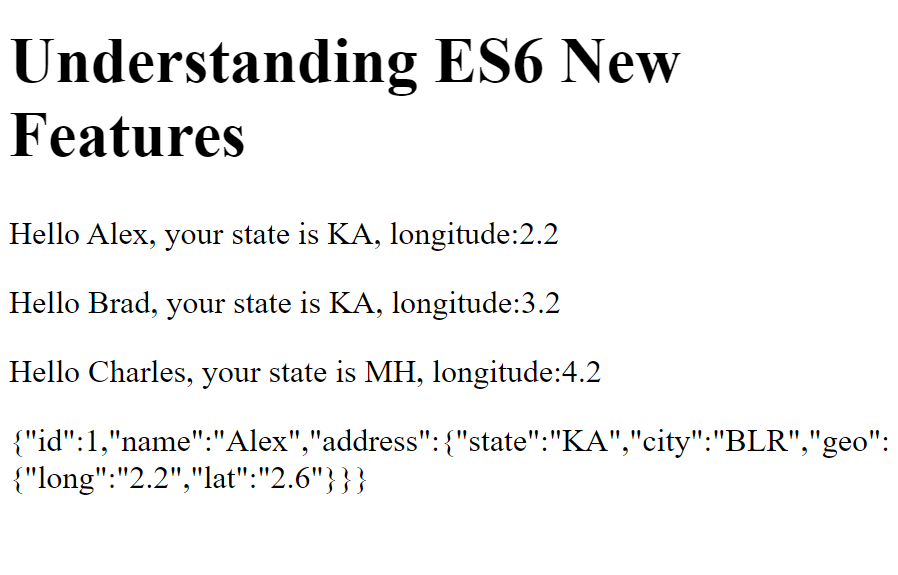
Old approach uses index

let iplTeams = [“RCB”, “CSK”, “KKR”, “GT”];  
let r = iplTeams[0];  
let c = iplTeams[1];

New approach uses Destructuring  
let [a, b, c, d] = iplTeams;



Output

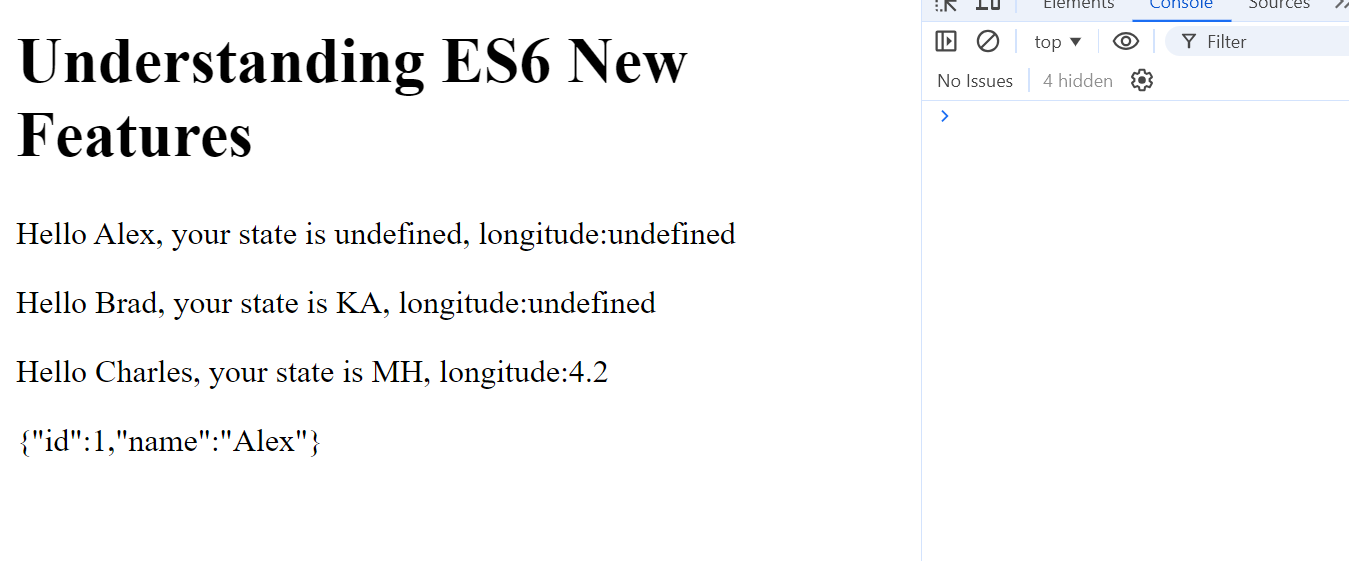


Optional Chain

Whenever you access nested properties you need to be careful because there could be chance of type errors

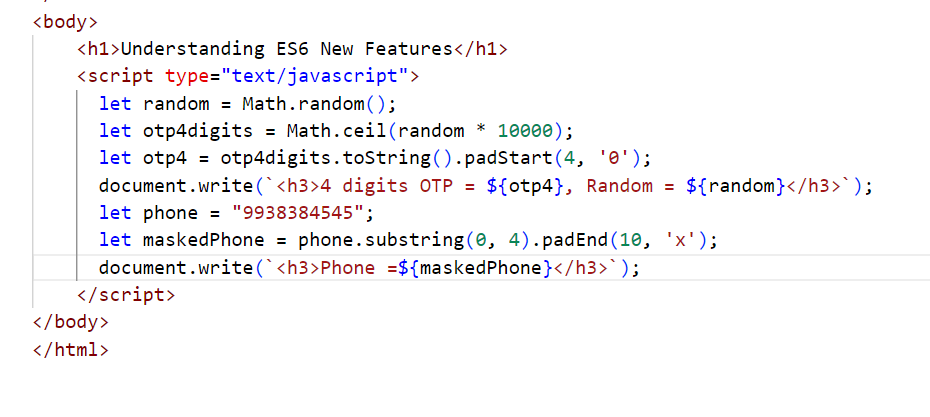


Output:

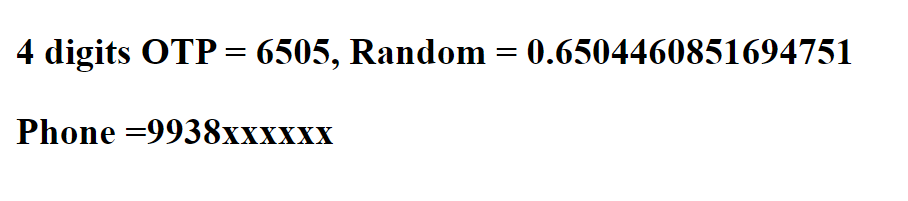


String padding start & end

9998889xxxx  
xxxx88989



Output:



canvas element:

It is mainly to create 2d shapes

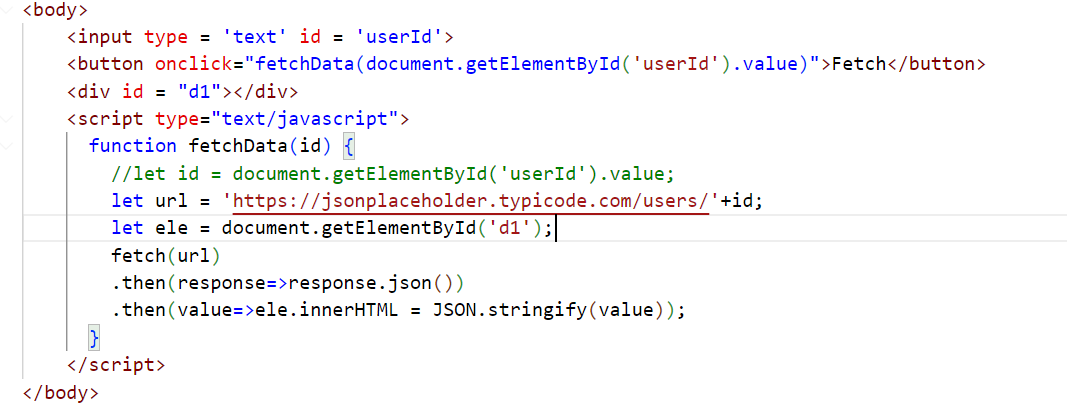
<canvas width=”400” height=”400” id = “c1”></canvas>

Promises

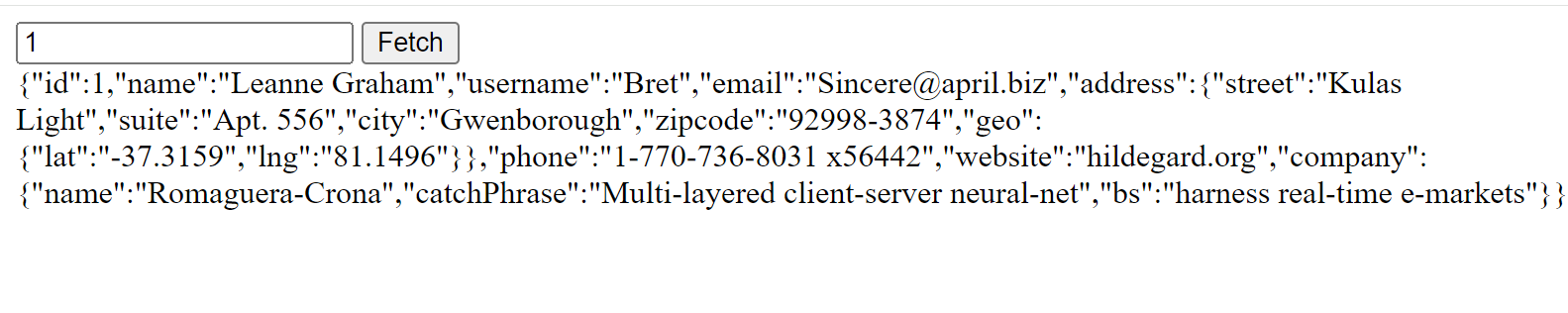
These are objects that make asynchronous requests and gets the response which can be either success or errors, if success its called promise is resolved, if error its called promise is rejected

You will use two methods of promise that takes callback functions as parameters

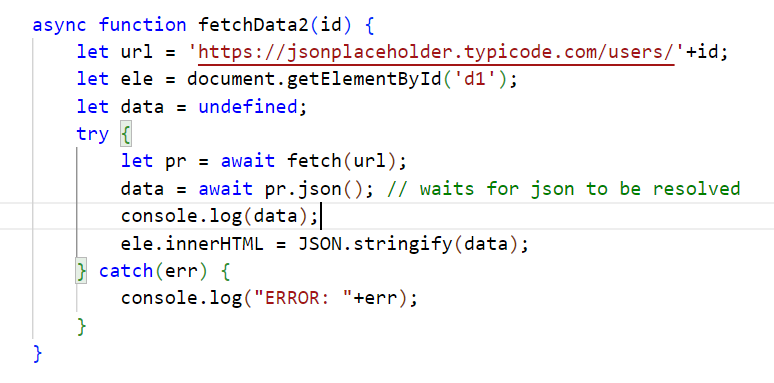
.then(callback)  
.catch(callback)



Output:



Async/Await



setters & getters

set & get keyword is used to create write & read functions

React.js

It is used to develop Single Page Applications(SPA), it is a Javascript library that helps you to create rich UI’s for your web page.

SPA:

Everything happens in one page, any changes you make that modifies only part of the page without reloading entire page

Components: These are independent UI’s which you can reuse in any part of the page.

Technologies react uses

1. HTML
2. JSX: Javascript XML: It makes you write HTML easily in the Javascript

Software’s required

1. VS Code
2. Babel toolkit -> Converts JSX to Javascript
3. Node.js & NPM

Libraries

Below are the libraries React.js uses

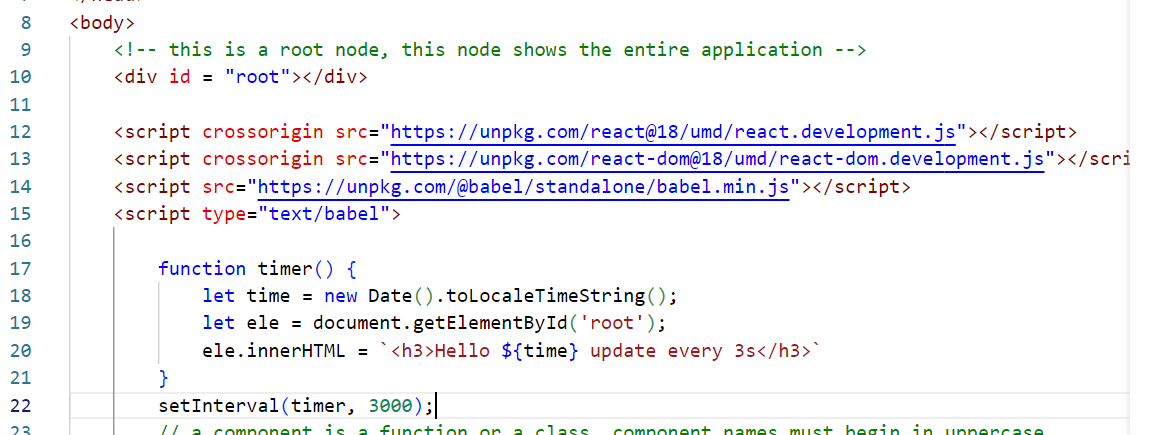
1. React: It is to create components & other react related features
2. React DOM: It is a virtual DOM
3. Babel: It is a transpiler to convert JSX to Javascript



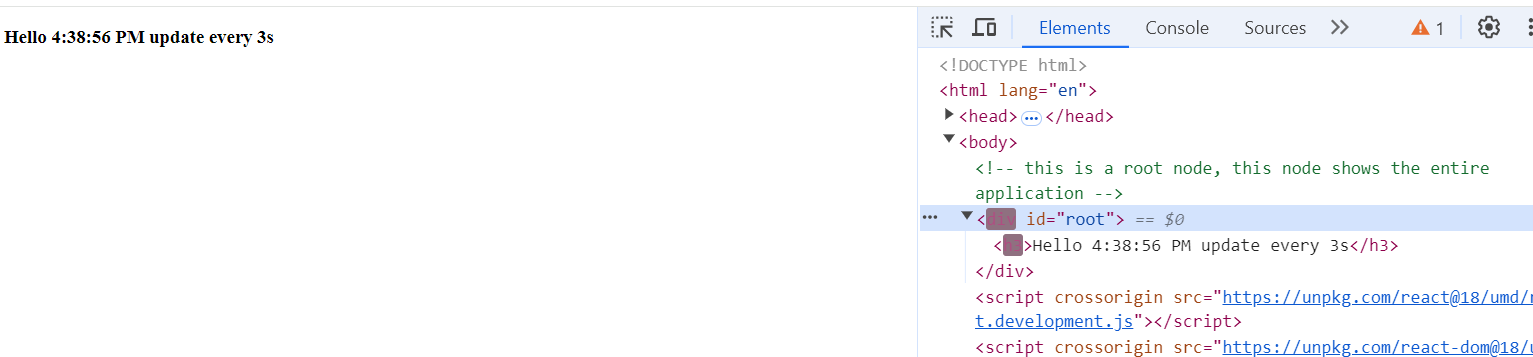
Virtual DOM

React uses this Virtual DOM to compare the content with the Real DOM of the browser and update only the changes required without reloading the DOM tree.

Without Virtual DOM



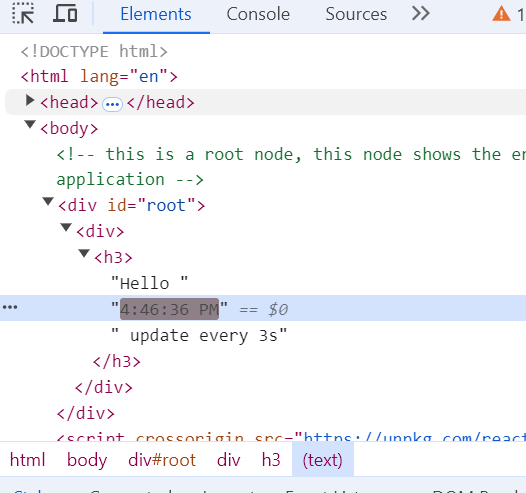
Output: You see entire DOM tree getting reloaded



With Virtual DOM



Output: You will see only time part is updating not the entire div or h3



React Tool Kit

It is used to create an industry standard react application, it gives inbuilt structure like

* node\_modules: to maintain all the libraries
* src: to keep all the source code
* package.json: to keep the project dependencies & commands to maintain the application

npx create-react-app myapp

