React.js

Software’s required:

* Visual Studio Code Editor
* Node.js & NPM
* Permission to download node modules

Pre-requisites

* HTML
* CSS
* Javascript (ECMA Script new features)

Brief overview on the contents

1. React.js
2. JSX
3. ES6 & ES latest features
4. Virtual DOM
5. Components
6. Controlled Components
7. Props & States
8. HOC
9. Render Props
10. Advanced Concepts
11. Routers
12. Axios
13. Testing
14. Styling libraries
15. Redux

React.js

It is a Javascript library used to develop Single Page Applications, it uses component based approach

Components: These are independent piece of code which are reusable & they are the visible part in the web page.

What React.js uses to create single applications

* HTML
* CSS
* JSX (Superset of Javascript): This makes coding Javascript easier to create UI’s

Why JSX:

JSX stands for Javascript Extended language, it is easier to develop UI’s compare to Javascript, because Javascript is complex when it comes to writing HTML code in it.

Note: HTML, CSS & Javascript are the languages browser understands

HTML is mainly for writing presentation to the web page

CSS is mainly for adding styles to the HTML

Javascript is to add dynamic behaviour to the web page using HTML & CSS as well it provides many programming constructs like variables, functions, operators, conditional statements loops, objects and etc.

Node.js: It is a runtime environment to run the Javascript without browser so that you can use Javascript to write backend programs to perform IO operations, DB operations, Implementing Backend services

Javascript & New Features introduced in Javascript

Javascript & other Javascript run time environment follow the standard provided ECMA (European Computer Manufacturing Association) Script, so those standards are like rules every ECMA specification language must follow

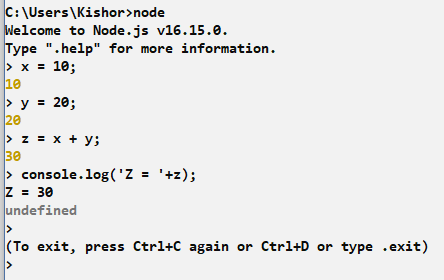
like Jscript, Java Script, Typescript, Browser, Node.js

ECMA Script provided many new features to improve the syntax of Javascript

1. Keywords like let, const, class, super, constructor, static, yield
2. Template Strings
3. Rest & Spread operators
4. Object Destructuring
5. Arrow functions
6. Generators
7. Exponential Operator
8. Trailing Commas
9. String padStart & padEnd
10. Optional Chain
11. Default parameters
12. Array includes
13. Object entries & values.

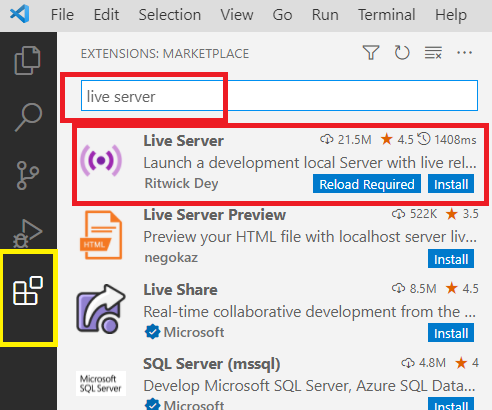
Javascripts are added in the HTML document using <script> tag and run on the browser or you can directly run the Javascript files using Node.js

Simple programs in node terminal

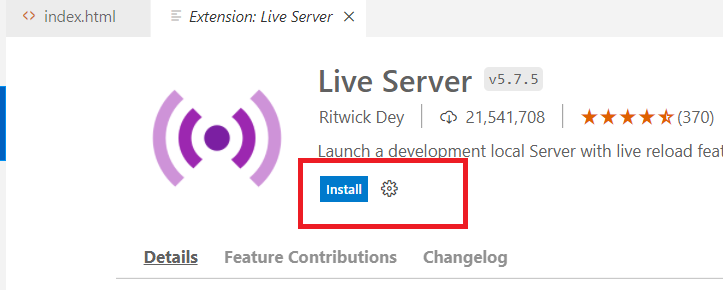


We can use Javascript files and add them in the HTML files and load it on the browser

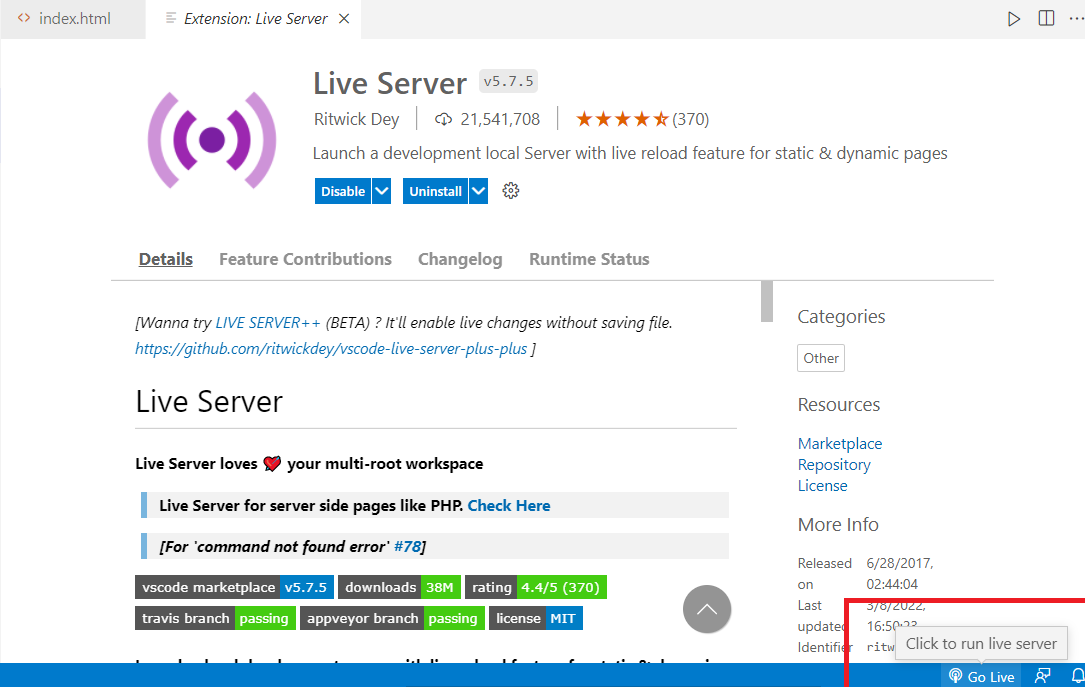
Adding Live Server plugin in the VS Code so that you get live reload feature while you save your code & don’t need to refresh



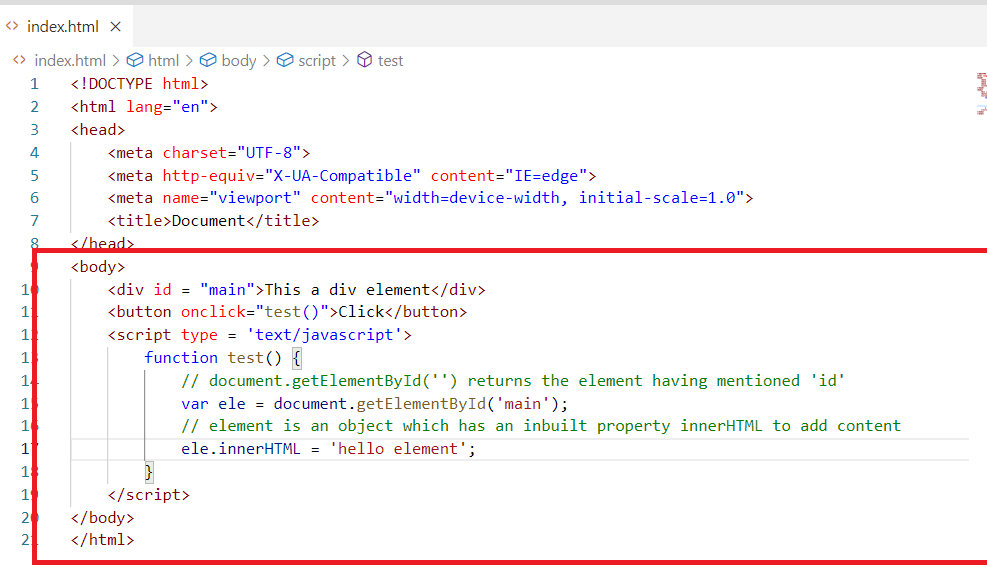
Click on Install



At the bottom right you see Go Live: Don’t click on it, instead you can select your HTML file and open with live server



index.html



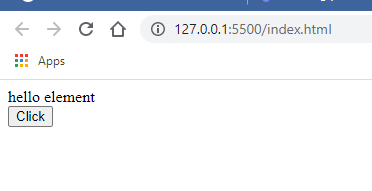
document is an inbuilt Javascript object, it provides many inbuilt properties & functions, some of the functions are getElementById(), getElementsByTagName(), write()

getElementById(): it returns the element object from which you can access many inbuilt properties of element object like innerHTML, style,

innerHTML: To access the content of the element

style: To access the style attribute of the element so that you can add CSS properties

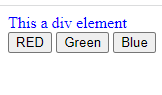
Output:



Using element object we can also add style attributes at runtime



Output:



New Features added in Javascripts

let & const keywords to declare variables: These are used to create block scoped variables,

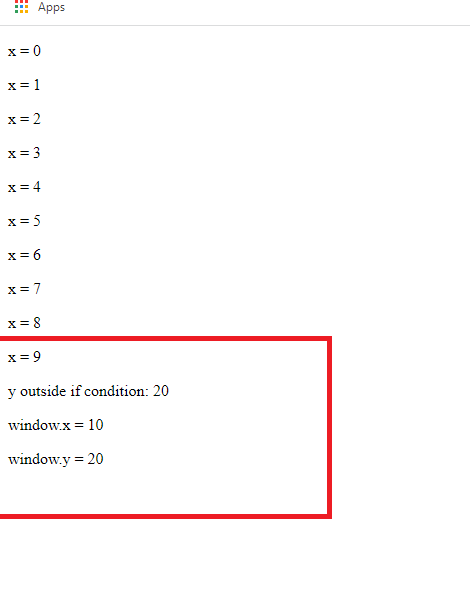
var keyword is not a block scoped variable, it creates in a global scope called window

window: It is a global object in Javascript, it is the top level object, every object is inside window object

When you declare any variable using var, then the variable will become part of window object, it wouldn’t be having any scope.



x & y are declared inside for loop, but you can access them outside the for loop as var creates the variable inside window object hence you can also access using window.x , window.y

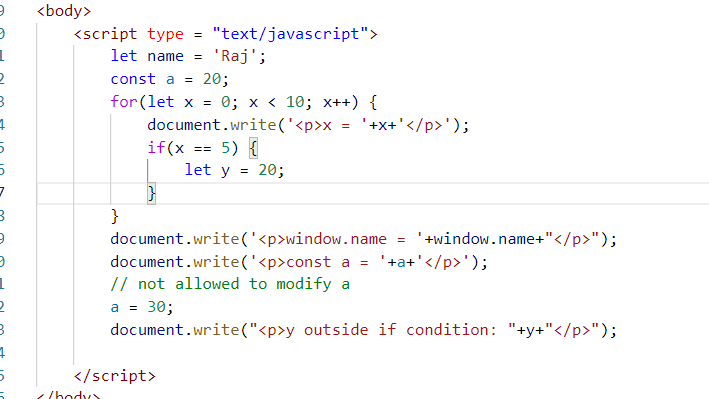


To address this problem ECMA script created introduced keywords to create block scoped variables i.e., let & const

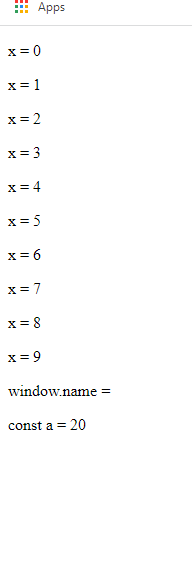
let: It is to create block scoped variables & it can be modified

const: It is to create block scoped variables & it is constant, can’t be modified

Note: let & const doesn’t add their variables to the global object, window



Output:



Template String Literals:

It is used to avoid string concatenations with many + character, so that you can create the strings with dynamic data without concatenation.

You can use `string ${expression} string`: Here the back tick can be used to create strings & the string can be added as it is however the dynamic data is wrapped in a ${data}

id = 100

qty = 4

url = “http://123.15.11.0/product/”+id+”/qty/”+qty

The above url is having many + character to concatenate string with dynamic value and also you can’t break the string inbetween without + operator

url = “http://123.15.11.0/

product/”+id+”/qty/”+qty

The above line is an error as string is not closed, it must be closed before bringing the same line to next line as below:

url = “http://123.15.11.0/” +

“product/”+id+”/qty/”+qty

To avoid this we can use a new syntax template string literals which doesn’t need any + to break the string

back tick (`) and single quote(‘): Theirs is a slight variation in back tick & single quote

url = `http://123.15.11.0/product/${id}/qty/${qty}`

Since id = 100 & qty is 4 the above URL will be

<http://123.15.11.0/product/110/qty/4>

The above URL can have a line break without any concatenation

url = `http://123.15.11.0/

product/${id}/qty/${qty}`

The above URL is still valid as back tick preserves the line break & doesn’t need any + to concatenate

Since id = 100 & qty is 4 the above URL will be

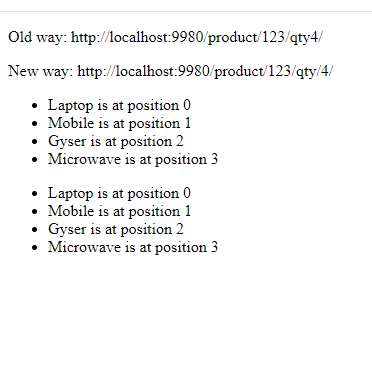
<http://123.15.11.0/product/110/qty/4>

These template strings are very much helpful in creating html elements when they have many dynamic data to be embedded.



You can observe the template strings reduces lot of concatenation.

Output:

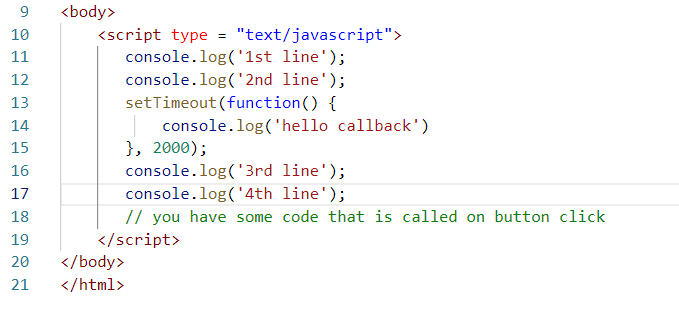


Callback:

These are special functions in JavaScript which are executed after some time but not immediately , they are usually asynchronous in nature, i.e., the next line or subsequent lines doesn’t wait for callback response

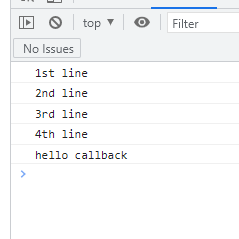
stmt1;  
stmt2;  
callbackfunction  
stmt3;  
stmt4

Here the stmt3, stmt4 is not going to wait for callback to execute, callback can be executed when it gets the opportunity



Here the setTimeout() takes callback as an argument, but the callback doesn’t block other operations for users, hence the 3rd & 4th line are executed without waiting for callback to complete, here the Front-end will not hang waiting for callback to complete, it still let user to access the UI’s.

Output:



In JavaScript you have XMLHttpRequest object which performs asynchronous operations by connecting to the backend.

XMLHttpRequest object provides some functions & properties to interact with the server side programs

open(http-method, url, boolean): Initialize the request

send(): it is to send the request

readyState: It will have state numbers of the XMLHttpRequest object from 1 to 4

1: request is initialized

2: request is sent

3: request is under process

4: response is ready to use

onreadystatechange: It is property that generates the event each time the XMLHttRequest object changes its state.

responseText: Stores the response from the backend

We have a fake online REST api to use XMLHttpRequest object to update our front end, we need to use callback here to update as we don’t have idea when the response will be ready.

<https://jsonplaceholder.typicode.com/users>

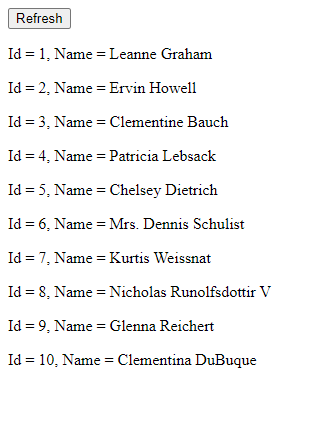


Here the callbacks are used to ensure that it will be executed when it gets time

i.e., the callback at 18th line is executed when the XMLHttpRequest objects changes its state from 1 to 4,

the another callback at 20th line is executed after 2seconds, by that time response will be ready as its used inside readyState == 4.

Output:



Note: Callbacks play an important role at the front end as they don’t block users to wait for the response.

These callbacks doesn’t have name which are called as anonymous functions, you use anonymous functions when you feel their logics are not reusable at other places.

Arrow Function:

This is a new feature in Javascript to simply writing anonymous function, it reduces writing function signature & also return statements if the anonymous function has only one line.

Usually callbacks are passed as arguments, but you will write function keyword, braces and return statement if necessary, but with arrow functions you can avoid them latelytely

Callback/anonymous function with one line having return statement:

function() {   
 return value;  
}

Arrow Functions with one line having return statement:

() => value;

Callback/anonymous function with more than one line having return statement

function() {   
 statement1;  
 return value;  
}

Arrow function with more than one line having return statement

() => {  
 statement1;  
 return value;  
}

Callback/Anonymous function of one line logic that doesn’t have return statement

function() {  
 statement1;  
}

Arrow function having one line logic that doesn’t have return statement

() => statement1;

Callback/Anonymous function of more than one line logic that doesn’t have return statement

function() {  
 statement1  
 statement2;  
}

Arrow function having more than one line logic that doesn’t have return statement

() => {  
statement1;  
statement2;  
}

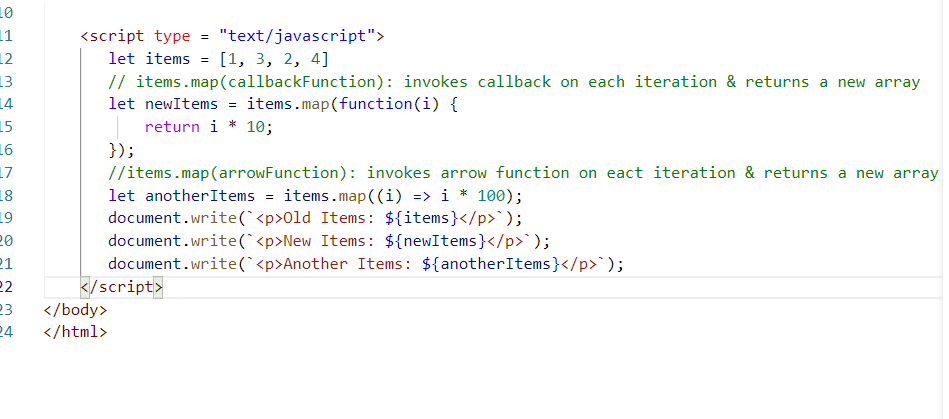
Callbacks/Anonymous function that takes argument

function(x, y) {   
  
}

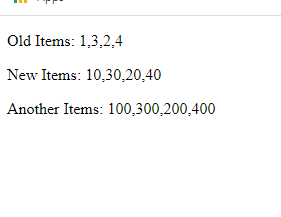
Arrow function that takes argument

(x, y) => { … }

Suppose you want to iterate an array and generate a new array in Javascript you have a map function



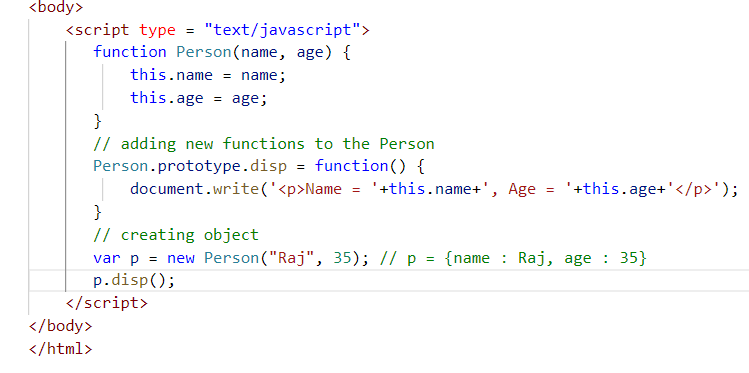
Output:



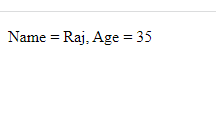
class, constructor, extends, super keyword

Earlier Javascript Functional based approach to create objects & achieve inheritance, which had a quite complex syntax:

Old approach



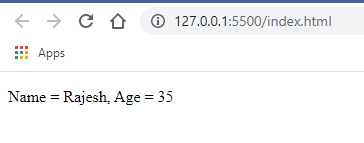
Output:



New feature of Javascript uses class & constructor to create objects & initialize objects, so that all the functions & constructors are modularized in a single unit.

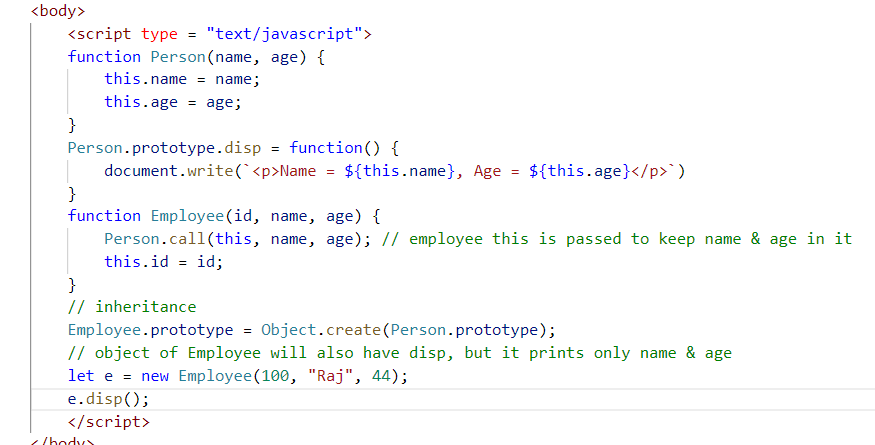


Output:

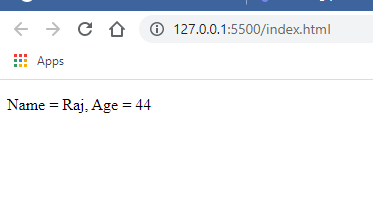


Another new feature is using extends & super to achieve inheritance

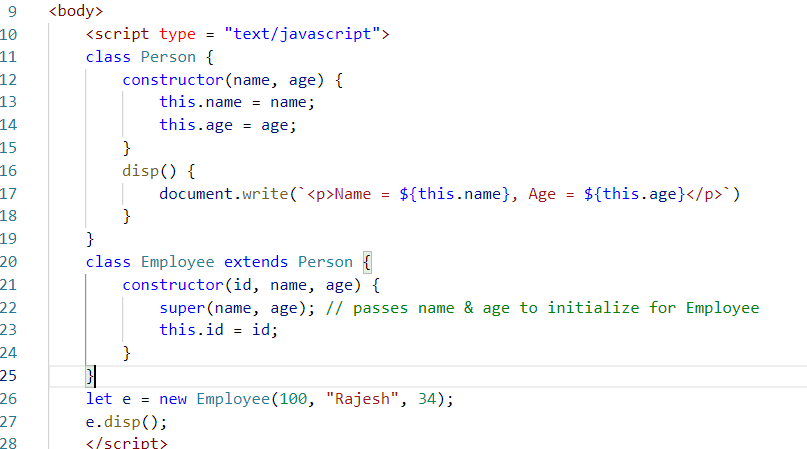
Older approach uses prototype to achieve inheritance and call() function to reuse the parent object function.



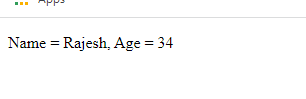
Here the properties of Employee is initialized by using Person.call() & Object.create() helps to inherit Person to Employee, hence you can access e.disp();



New syntax uses class, extends & super keyword to simplify achieving inheritance



Output:



Instruction to upload your activities in the GIT

Step1: Login to your Git account & then Fork the Repository (Need to be done only once) of the link

<https://github.com/Kishor-C/maverick-react-js-activities.git>

Step2: Copy the repository link that is forked in your account (Ensure the link has your account id) and Clone in your machine (Need to be done only once)

Step3: After you clone navigate to the repository through GIT bash terminal

Once you make changes in your repository folder like creating files & folders of your activities

Step4: Enter following git commands in the order (Ensure you are inside the repository in git terminal, you will see (master) in brackets)

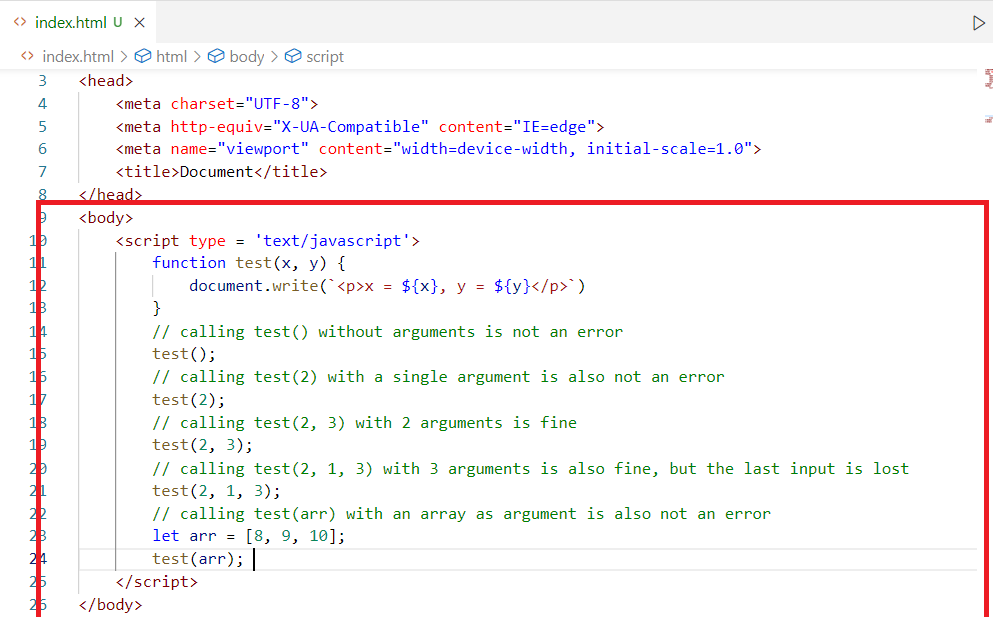
* git add . # there’s a dot after add
* git commit -m ‘some message’
* git push -u origin master

Step4: Observe in the Git Hub whether or not the changes are updated

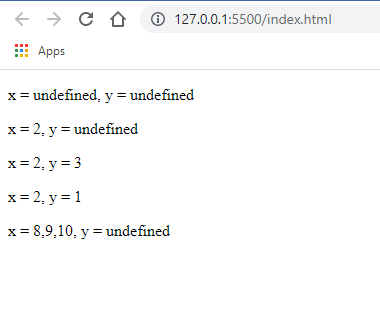
Rest & Spread operators

Javascript doesn’t through any error if you call the functions by missing any arguments or if you provide extra arguments

Below example has a test function with 2 arguments, but it can be called from any number of arguments.



test(2, 1, 3) will pass value for x as 2 & y as 1, however the data 3 is not received by any variable, which means its lost, similarly the test(arr) is an array of 3 items, but the parameter x itself accepts all the items in the array it doesn’t split the items to the number of parameters i.e., both x & y should share those items.



To avoid this problem we have rest & spread

Rest: It is a parameter with 3 dots in the beginning, which can accept 0 or more values at time of calling function

function test(x, …y) {   
}

The test function has x that accepts only 1 argument, …y is a Rest parameter which can accepts 0 or more arguments, it means y will not be undefined if in case you don’t pass argument.

Below calls to test assigns value to x & y as:

test(20, 30); // x = 20, y = 30

test(35); //x = 35, y will be empty it isn’t undefined

test(2, 3, 4, 5); //x = 2, y = [3, 4, 5]

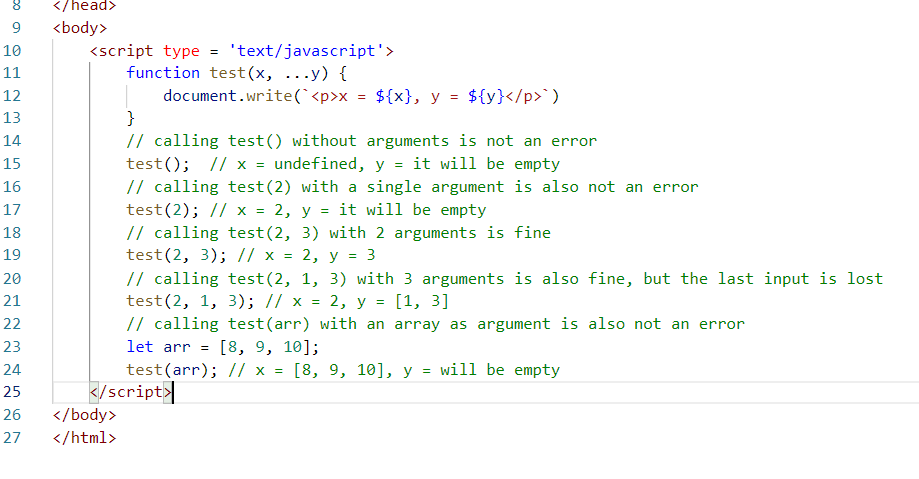
Note: Rest parameter must always be at the end and you can maximum have one Rest parameter for a function

Note: Rest parameter automatically acts like array as it can store 0 or more values, you can use for loops as well on rest parameter variable

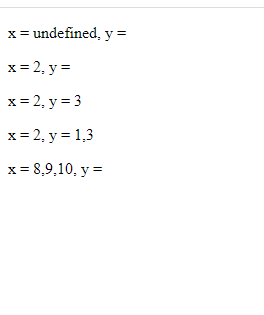
function test(…x, …y) { } // is invalid

function test(…x, y) { } // is invalid, here y is of no use as x itself accepts all arguments.

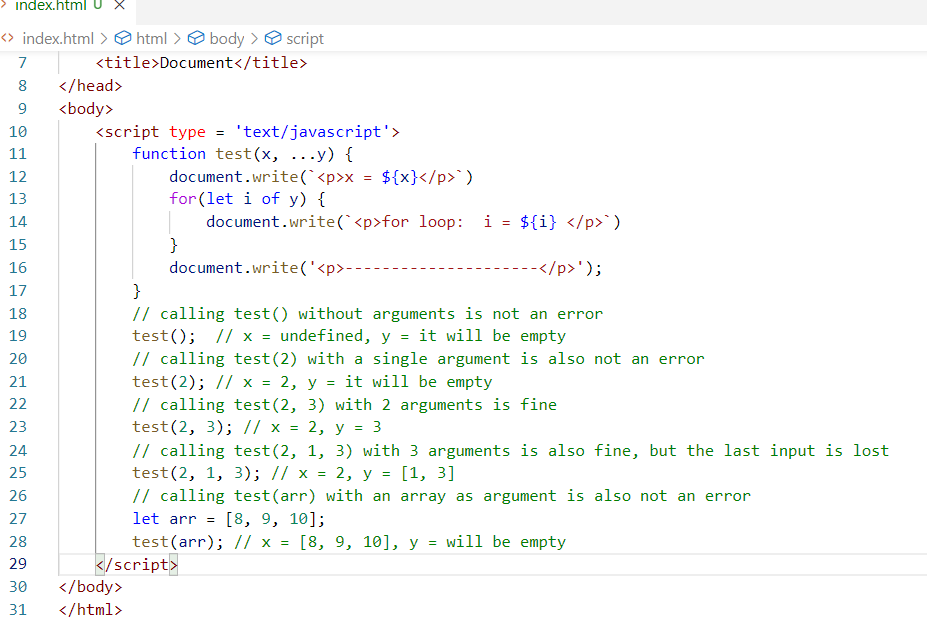
Writing above functions leads to error in Javascript.



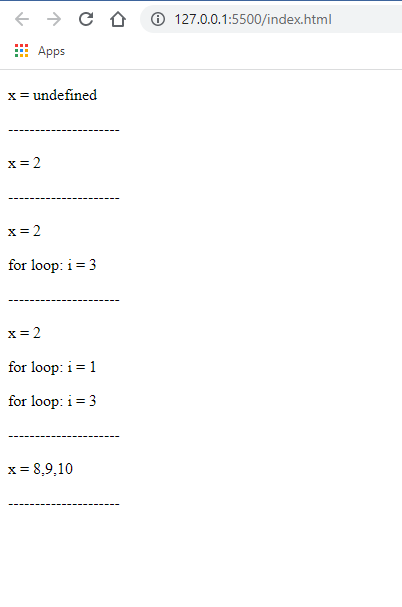
Output:



You can use for loop on Rest parameter variable

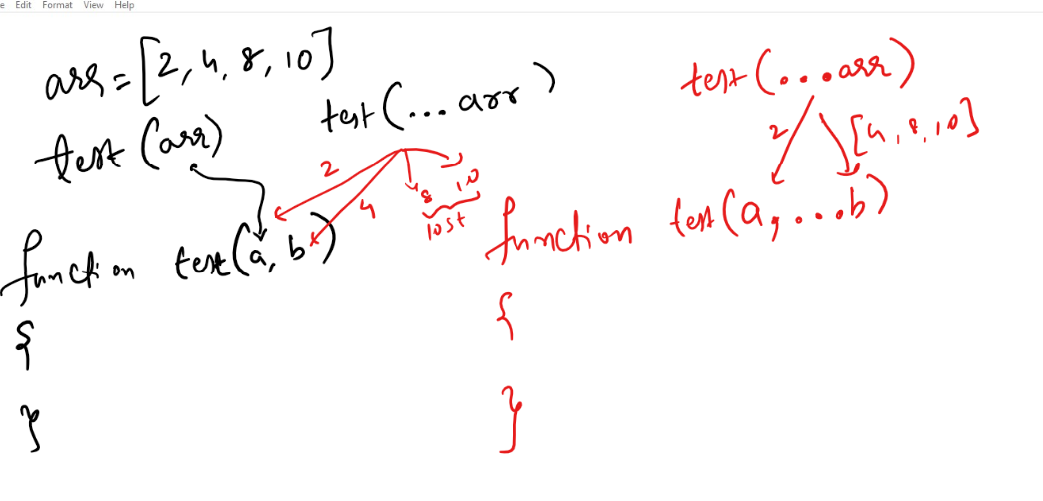


Output:

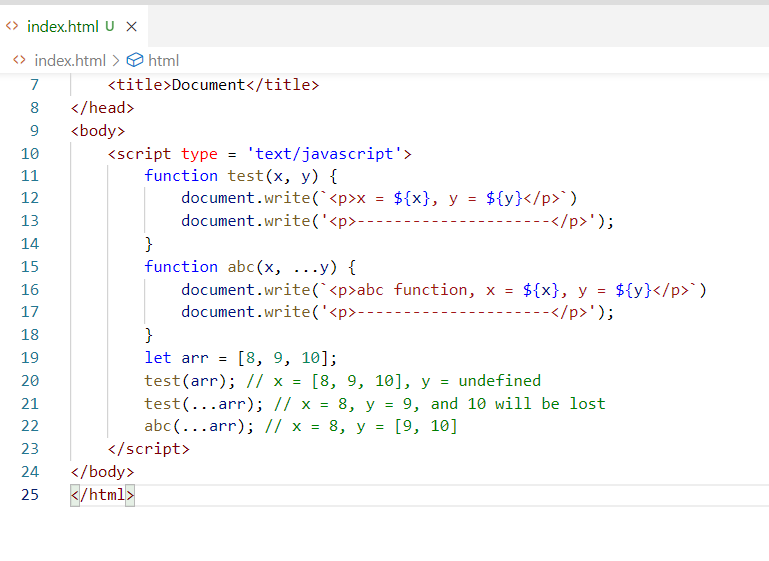


Note: Since the Rest parameter accepts rest of the values it is called Rest

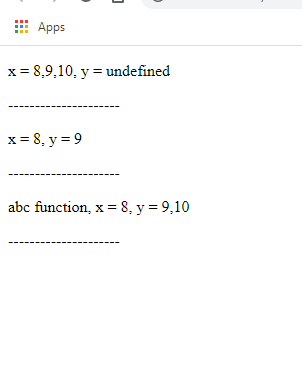
Spread operator: It is used to distribute or spread the multiple values across the parameters of the function.



Spread operator is used at the time calling the function

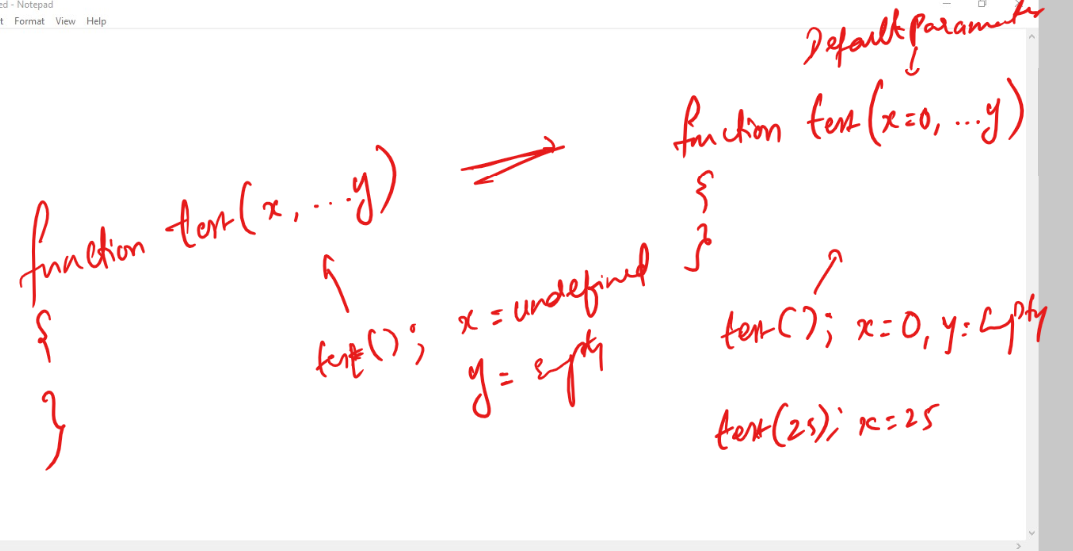


Output:



Default parameter:

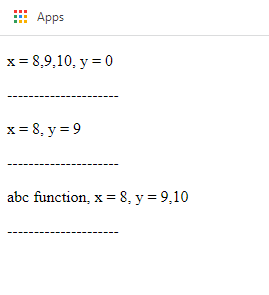
It is used to avoid undefined values to the parameters in case the parameters doesn’t get the value at the time calling



Note: Rest parameter can’t have default parameter, it is by default empty

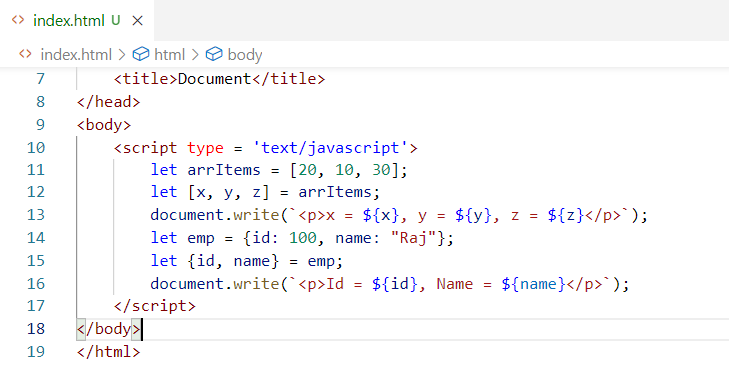


Output:

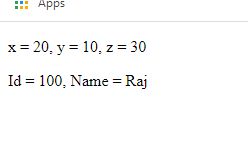


Object & Array Destructuring:

This eliminates declaring variables in multiple lines when you have a scenario where you want to assign array items to each variables & object properties to each variables



Output:



Trailing commas:

In Javascript the commas are automatically ignored when you create extra commas in case of arrays

Old javascript:

x = [1, 2, 3, ] // Used to give an error because of extra comma

New Javascript:

x = [1, 2, 3, ] // extra comma will be trailed

Exponential Operator

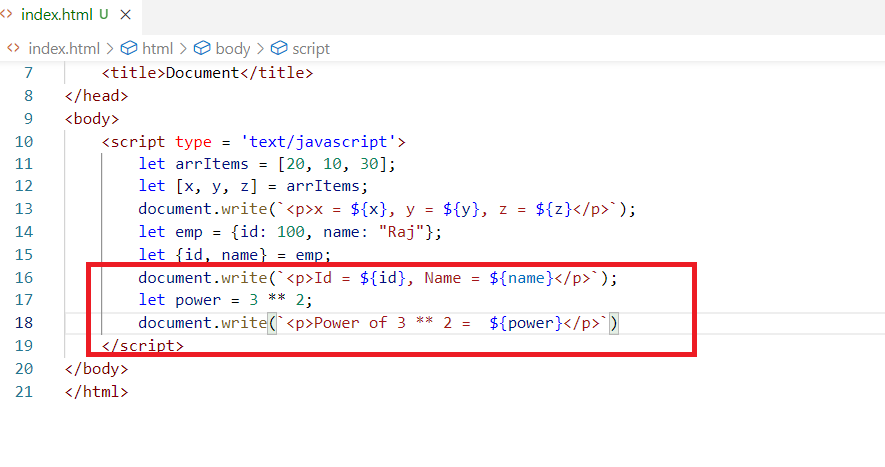
You can use \*\* (2 stars) to find the power of 2 numbers in a simple expression

Earlier:

Math.pow(3, 2): It gives 9

Now:

3 \*\* 2: it gives 9



Static members:

These are the variables or functions that you can access from class name, when you want certain functionalities to be accessed using class name you can use static

class A {   
 static test() { …}  
}

A.test(); // works fine

Array.includes:

New method to find the existence of an item in the array

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

Array.includes(1): It returns true if 1 is present else false

Earlier: You had to loop each item & compare with the item you want to search, but now you have Array.includes()

Object entries & values

These are two additional functions used to find the object properties & values

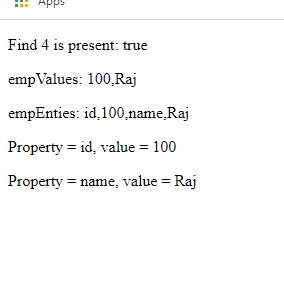
let emp = {id : 100, name : “Raj”}

Object.values(emp): It returns the values of all the properties in array format, [100, Raj]

Object.entries(emp): It returns properties & values both in the array format:   
[id, 100, name, Raj]



Output:



Optional Chain:

This avoids accessing nested properties when they don’t exist it uses ?. to access the property.

The below code iterates each user object from the array & when it tries to access address on the second iteration it gets undefined & on undefined it tries to access state and throws error



Output:



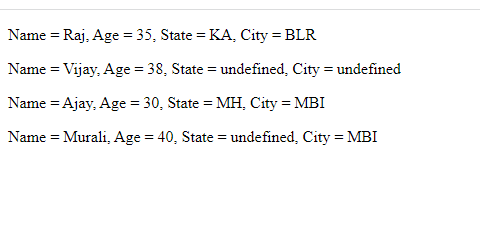
TO avoid the error you can use if conditions, but it will become complex when we have too many nested properties in an object as we need to handle undefined on each nested property.

To avoid this problem we have Optional Chain(?.), which will access the property only if the property exists, if undefined it doesn’t tries to access nested property

?. means if presents then access



Output:



padStart & padEnd functions

These adds some extra characters to the strings in the beginning or to the end depending the number of characters

padStart(): It adds the special characters in the beginning of the string

padEnd(): It adds the special characters at the end of the string

Note: It works only with strings, you can covert your data to strings to use these functions

let otp = “4523”

If you want 6 digits OTP

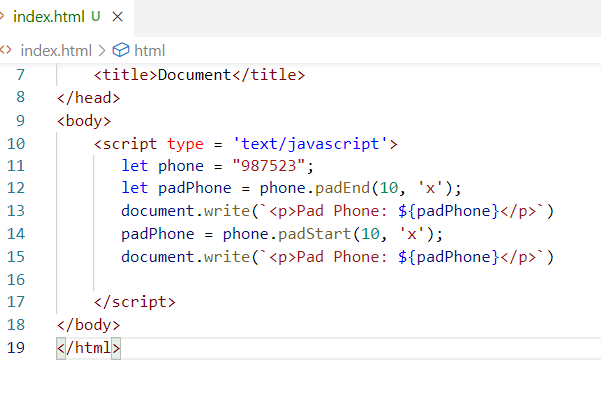
otp.padStart(6, ‘0’): It returns 004523

otp.padEnd(6, ‘0’): it returns 452300

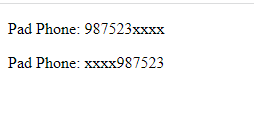
otp.padStart(4, ‘0’): it returns 4523

otp.padStart(3, ‘0’): It returns 4523

padStart() padEnd() doesn’t remove original values



Output:



Summary of new features:

* let, const, super, extends, class, static
* Template Strings: `${}`
* Arrow Functions: () => { }
* Object & Array Destructuring
* Array includes
* Object values & enties
* padStart & padEnd
* Rest & Spread operators
* Default parameters
* Exponential Operator
* Static
* Trailing commas
* Optional Chain
* padStart & padEnd

React.js

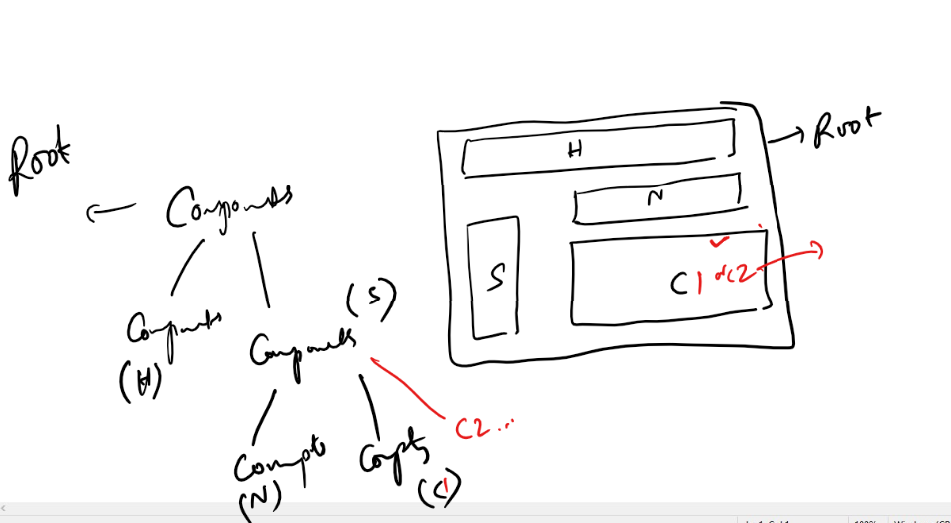
* It is Javascript library used to create Single Page Applications
* It uses components to create the UI’s
* Components: These are independent UI’s which are reusable & can be maintained independently
* React uses HTML, CSS & JSX to develop UI’s

HTML: It is for contents

CSS: It is for styling

JSX: It is an extended version of Javascript also called as Javascript + XML which simplifies writing HTML in the Javascript, it supports all the features of Javascript

Note: JSX is not understood by browser, it must be compiled by interpreter called Babel, which converts JSX to Javascript so that browser can understand.



Root Component: In React.js you will have a Root component that represents the entire document, and all the components would be part of the root component

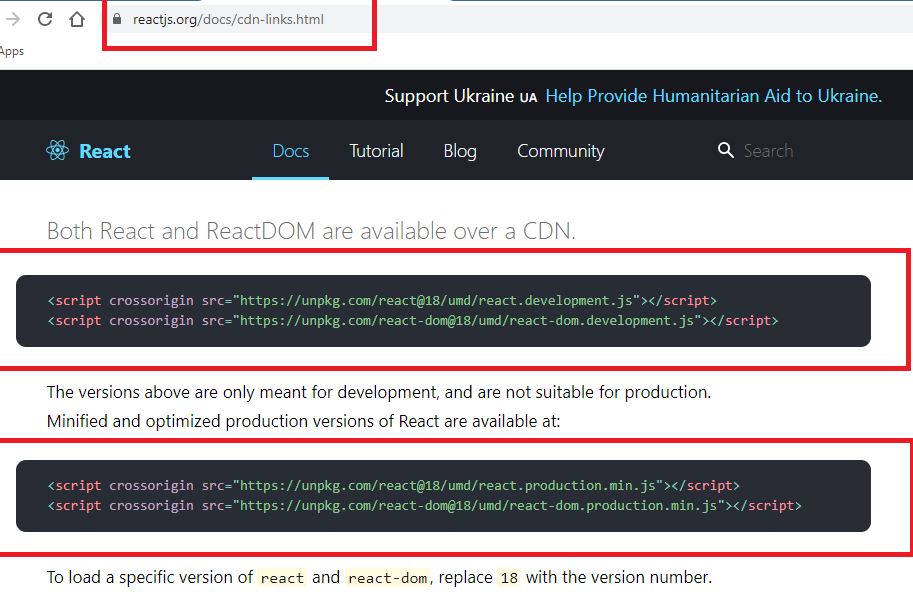
React.js provides 2 libraries to develop UI’s

1. React library: It is responsible to create components
2. ReactDOM library: It is a Virtual DOM which maps all the components to the Real DOM (Browser)

These two libraries we can get either from CDN link or React tool kit

Firstly we will use CDN link which is an online, later we can see React tool kit which is downloaded using Node.js

<https://reactjs.org/docs/cdn-links.html>



You can use either production mode or development mode libraries

Note: Apart from this you need to use babel library because you will write JSX instead of Javascript

We can also create React.js application through Javascript without JSX but it will be complex to create components

Creating React.js components with Javascript

You must have one <div> inside the HTML all the components including root components will be inside this <div>, though you have one <div> inside the HTML through components multiple <div> appears as components also like HTML elements.

If you are using Javascript to create a React component then you must use React.createElement() method

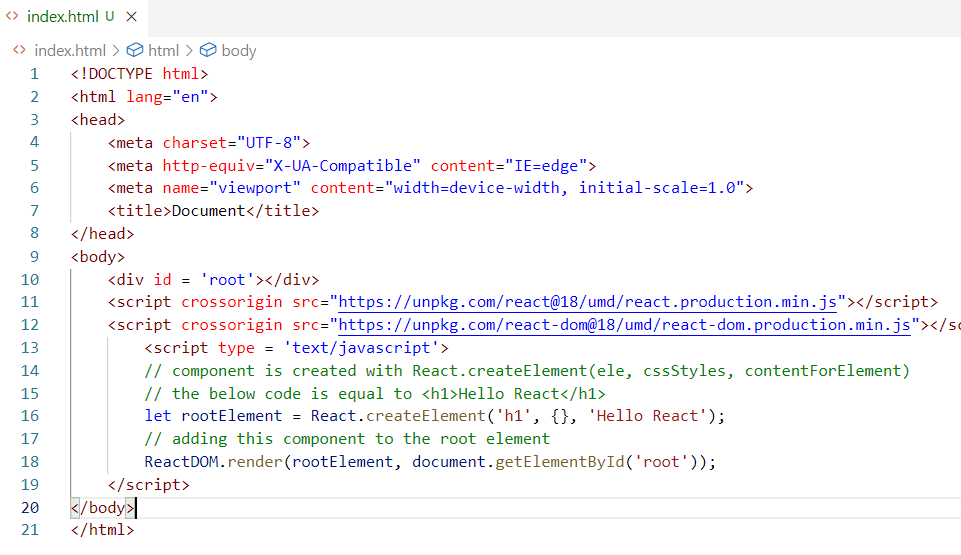
React.createElement(): It creates a component which will have an HTML element & some dynamic content

React.createElement() takes 3 arguments

1st argument is the element or component

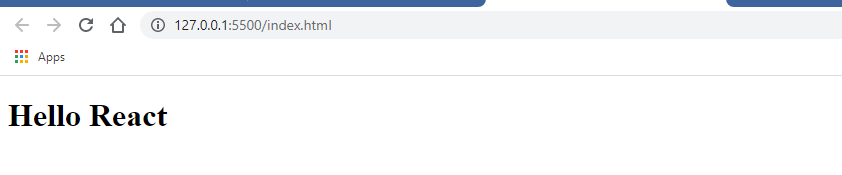
2nd argument is the styles

3rd argument is content

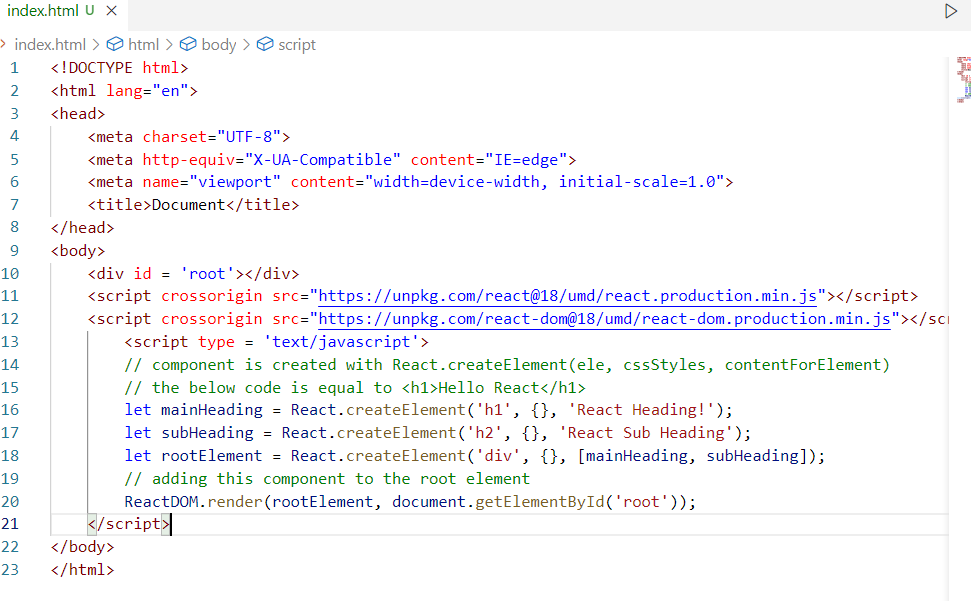


Here rootElement is the component that is added to the <div id = ‘root’> via ReactDOM.render()

Output:



Nesting the components with other components in React using Javascript



Here the mainHeading & subHeading are the components which are added to rootElement as [mainHeading, subHeading], so that it will be

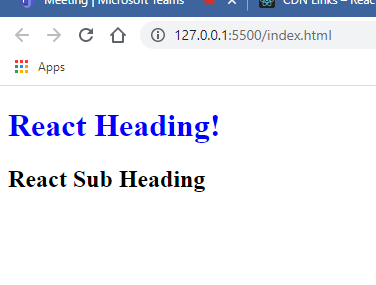
<div>   
 <mainHeading></mainHeading>  
 <subHeading></subheading>  
</div>.

Adding css classes to the components

We must use the second argument of the React.createElement(), there you can mention the CSS classes.



Output:



Currently we are using Javascript to create the React components, since It has hard coded content, it doesn’t look complex, but when you want to create complex structure like tables, lists, forms and so on it becomes complex to write, because everything is written with React.createElement()

But JSX complete avoids using this React.createElement(), because it helps to create components in HTML format in the code like <h1>Welcome to React</h2>, but you must need babel to convert it to Javascript in the form React.createElement(‘h1’, {}, ‘Welcome to React’).

In JSX

let rootElement = (<h1>Welcome to React</h1>);

In Javascript

let rootElement = React.createElement(‘h1’, {}, ‘Welcome to React’);

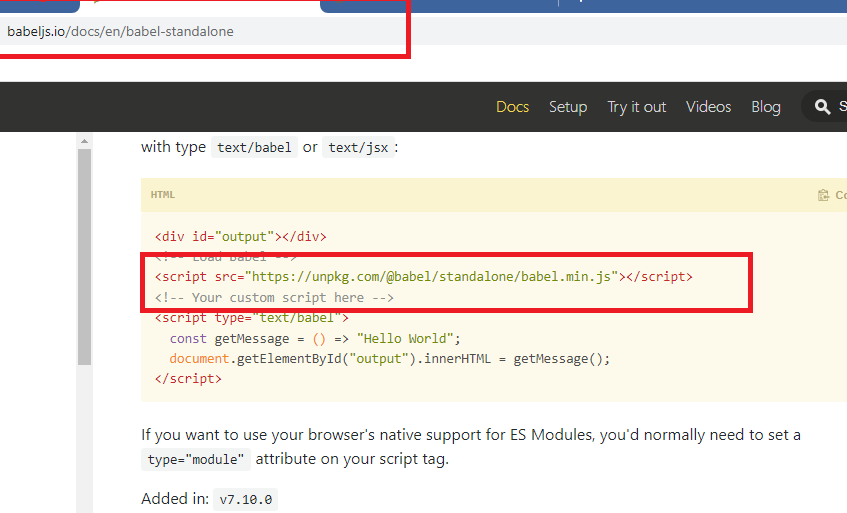
We need to add babel library to convert JSX to Javascript, but we can write JSX code and also use any Javascript expressions in JSX

Once you add babel library you must use <script> type = “text/babel”

CDN Link:

<https://babeljs.io/docs/en/babel-standalone>

Note: If use React toolkit from node.js we don’t need to use CDN link either for React or for Babel





Output:



Creating components with a Name

You can create a function that returns the content, We can use that function name to add the components in any place of our website, the function must begin with uppercase mandatorily

function App() {   
 // statements  
 return (component content);  
}

component content: It could be combination HTML, Components, Javascript Expressions, Dynamic data and so on.

We can use <App /> to add the component content



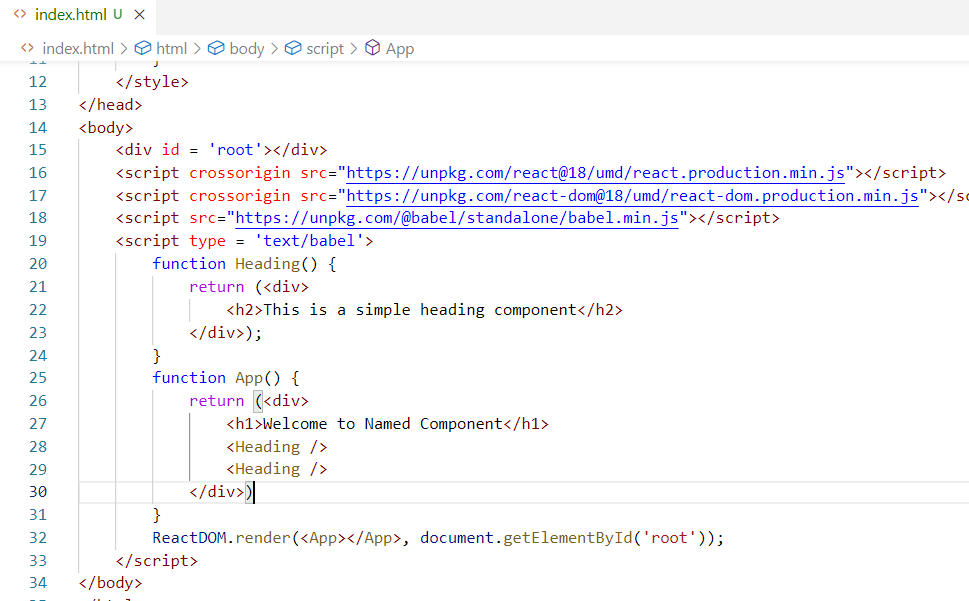
Here the <App></App> or <App /> is a component name which renders the content the function App() returns in the div#root.

Output:



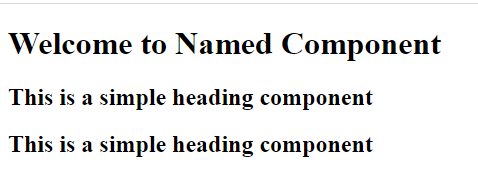
Nesting the components within another component

You just create a component function beginning with uppercase & its tag must used in another component function while returning



Here the <Heading /> component is nested inside the App component since its used twice the content of Heading() is rendered twice.

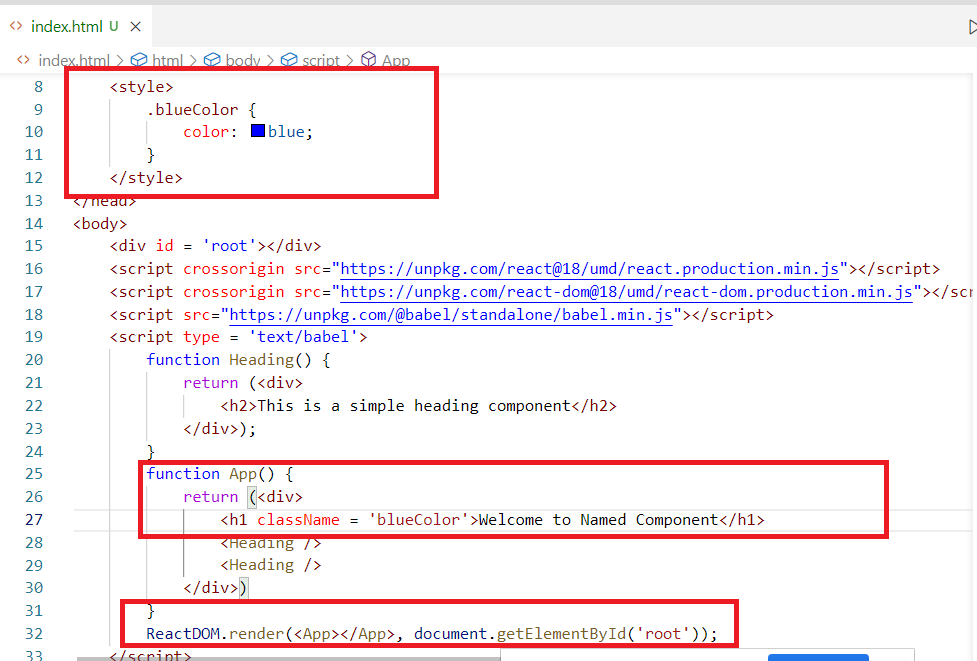
Output:



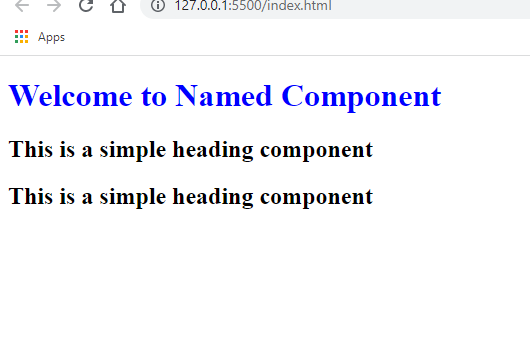
Adding CSS classes to the components or html elements part of the components

You must use the className instead of class attribute as ‘class’ is a keyword to create classes in Javascirpt, since the components are written in a script we must avoid using class keyword to apply CSS classes.

Note: You can use class keyword in the JSX to create classes but to add CSS classes you must use className



Output:



Javascript expression in JSX

You can use {} to write any Javascript expressions in the JSX, two flower braces will evaluate the Javascript expression, it could accessing the variable, calling the function, provide values to the attributes and so on.

Using variable names in the JSX



{user}: It is used to access the Javascript expression

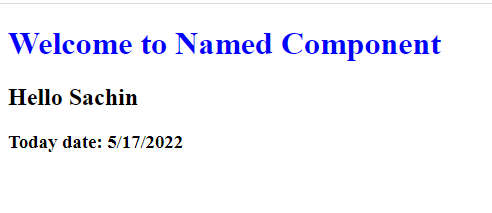
Output:



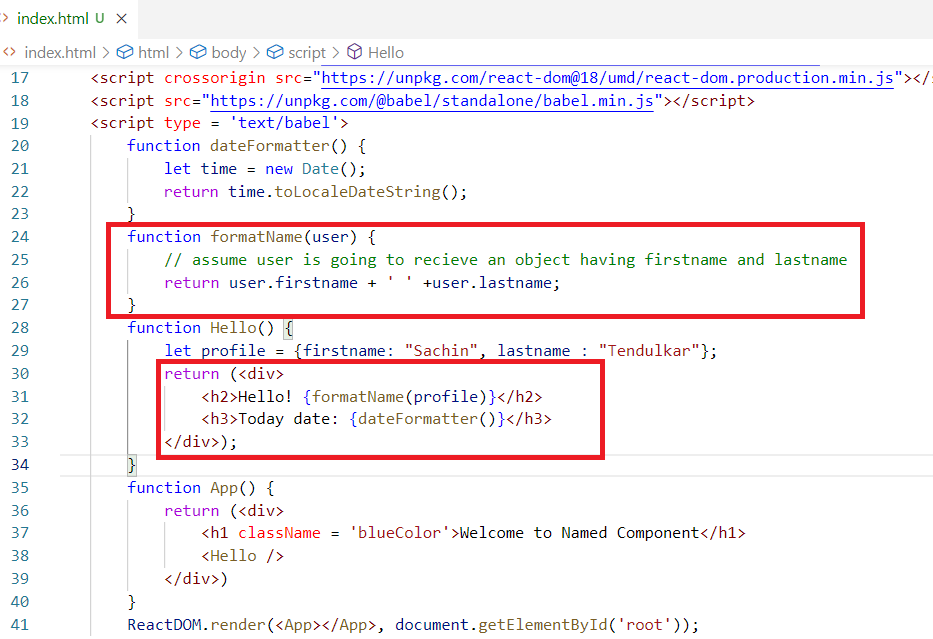
You can also call a Javascript function



Output:



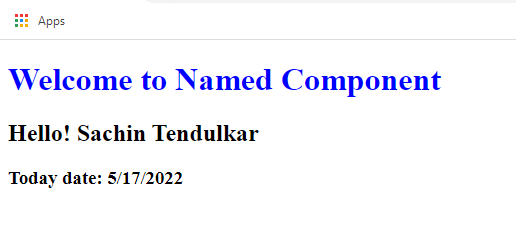
Calling Javascript functions in JSX that take arguments



Here formatName() takes an argument and access 2 properties like firstname & lastname, you can pass an object having firstname & lastname i.e., profile = {firstname: “Sachin”, lastname:”Tendulkar”}.

Since formatName() returns the concatenation of firstname & lastname as a full name, the component calls it gets the full name

Output:



You can also use attributes in JSX

Suppose you want to provide url to the <img> tag, you can use src attribute of img and assign the url to it.

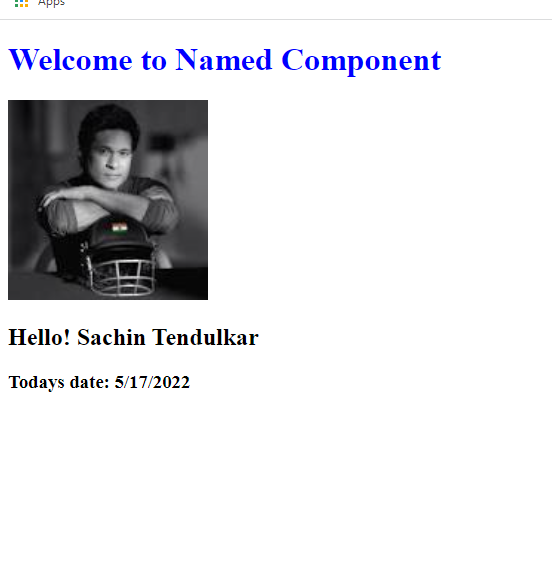
<img src = {imageUrl}></img>

If imageUrl has some url of an image then the <img> tag could able to show that image.



In Hello() function we have declared imageURL and its value is taken from the internet & the component renders image, name & date when you use <Hello />, since App() component uses <Hello /> component we could see Hello component content.

Output:



Using classes to create components

You can create named components not only through functions, even through classes, initially Javascript was not supporting classes, but later Javascript started supporting classes and made the code more organized so that functionalities belonging to the same class can be created within the class.

Hence you are allowed to create components through class name as well, but you must inherit one of the predefined class called React.Component, here React is a top level class inside that you have a Component class.

class App extends React.Component {   
 render() {   
 // this is the function that must return the content for the component

// this is automatically called when you use the component as a tag   
 }  
}

Since the code present in the class not automatically executed, the components created with class must override render() function which is called when you use the component name to display it, hence you must write the content for the component in the render()



Whenever we create components through classes we must extend it by React.Component & Override render() function, since components are visible part or UI’s we must create UI’s in the render() function & return it.



You can create components either through functions or through classes

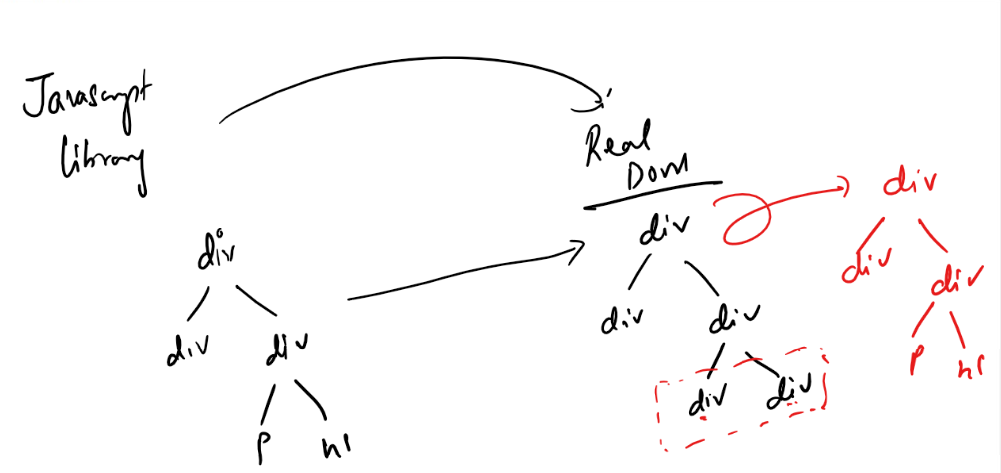
When it comes to comparison both are better, it all depends on our comfort level to go with component or functions.

Understanding about ReactDOM

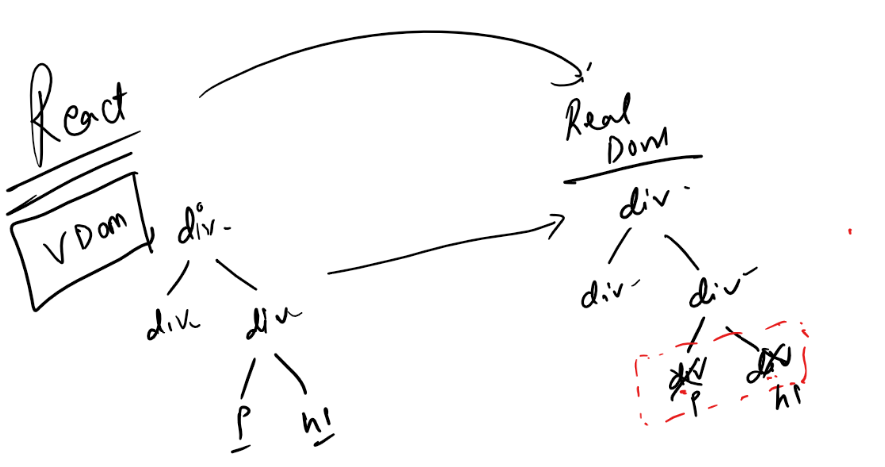
It is actually a Virtual DOM that updates the Real DOM (Browser DOM structure), React uses Virtual DOM to increase the performance at the Front-end while updating part of the web page, because most of the Javascript libraries updates the entire DOM tree structure even when part of the page updates, but for user it looks like part of the page is updated but behind the scene the entire DOM tree is reloaded it will degrade the performance at the Front-end.

Virtual DOM will first have the entire tree structure with the new content & compares with Real DOM and only updates the node that needs to be modified by keeping other nodes unchanged, this increases the performance at the front-end.

In Javascript the Real DOM is directly updated



In React.js Virtual DOM compares its new tree with the Original DOM tree and updates only those nodes need the change

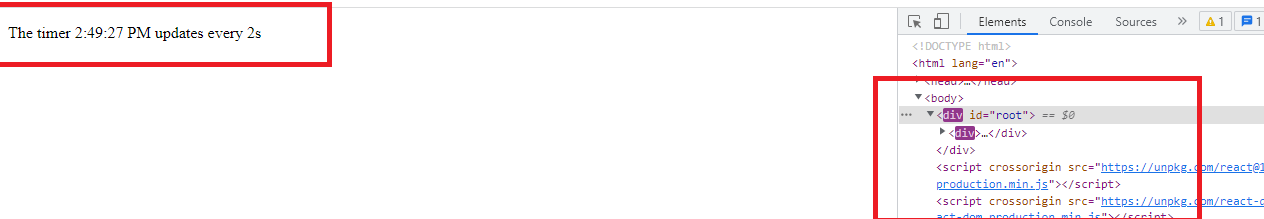


Proof that Javascript refreshes entire DOM structure

We can write a simple timer function that updates the time for every 1s and see how its looking in the front-end and behind the scene how the DOM tree is updating.



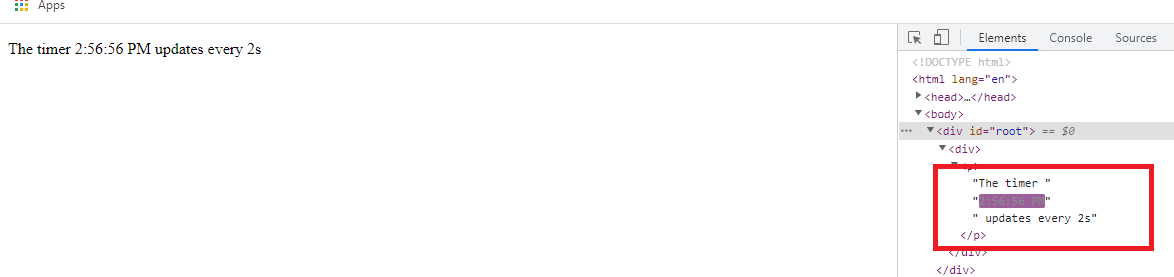
Here we are updating the div#root every 2s, though in browser we could see only seconds are updating, the DOM tree shows entire structure getting updated



However in React.js the VirtualDOM compares the current change in it with old change in the Real DOM and only updates the necessary elements keeping other elements unchanged in the Real DOM.



Here the Virtual DOM renders the component to the Real DOM every 2s, but you see the changes happening only to the new content



Till here we have used React & ReactDOM library as well as babel, because of the live server plugin the page is getting auto-reloaded, but we have another way to develop react application i.e.., by using React tool kit

React Tool Kit

It is a development tool kit which helps to create ready to run react applications with all the necessary libraries required to run in development, testing & production environment, it accelerates the development process by providing well structured project with various supporting files to develop, test & build the project

Benefits of React Tool Kit

* It provides us ready to run production grade setup for React application
* It gives a development server to give a pleasant development experience
* It will have all the necessary libraries downloaded to develop react applications like babel, react library, react dom library
* it provides various commands to easily run, test & build applications
* It provides version controlling feature so that you can update your project to the git without any setup
* It provides a ready made template for every new project which you can change as per your need
* It provides auto-reload feature where you see the changes in the output when you modify your code

Note: To use React tool kit we need to have node.js installed

How to Install React Tool Kit

There are 2 ways

1. npm install create-react-app: This locally installs the React toolkit in your machine and using it you need to create react projects, but it always needs an update when you want change the version of tool kit
2. npx create-react-app: This helps to create react projects directly without installing React Toolkit in your machine, but it connects to the internet to search for the latest react toolkit and creates the project for you, here it always gives the latest project

npm: Stands for Node Package Manager, used to download node modules from the internet could be react tool kit, angular toolkit, bootstrap js, other js libraries

npx: Stands for Node Package Execute, used to directly execute the modules without installing it locally

Note: npx, npm are the commands you get after installing Node.js

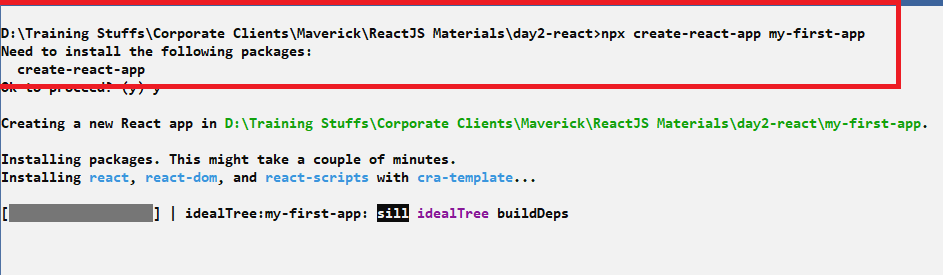
How to create react application using npx

>> npx create-react-app app-name

Here npx is the command to search the toolkit over the internet

create-react-app: it is the toolkit name that set up the environment to create & run react application

app-name: It is the project name



Once you enter the above command wait for the project to download

Once it downloads you could see a folder created and you must navigate to that folder.



You don’t have to understand all the files right now, you can see some files folders like

node\_modules: It keeps all the javascript libraries required for your project like react, react-dom, babel and etc.

public: It has an index.html, this html is the only file loaded in the browser, it has a div#root

src: It keeps all the javascript & css files

src/index.js: This file has a code that adds root component to the div#root

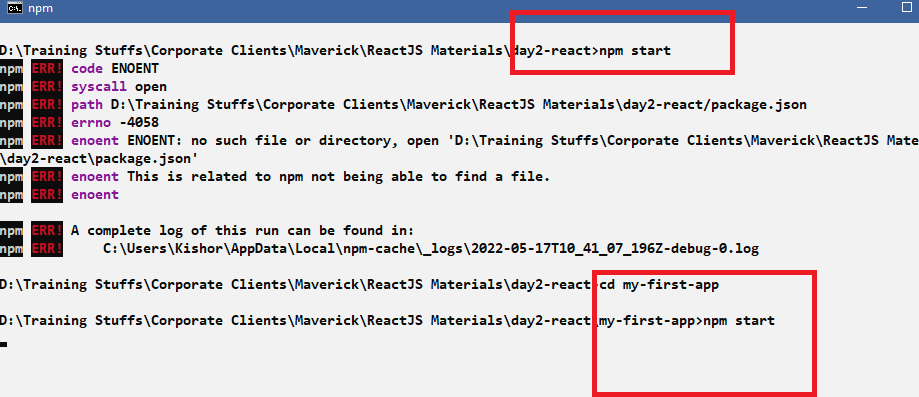
src/App.js: This file has a root component code i.e., function App()

Since it is very first project we will try to navigate to the project folder & type npm start

npm start: It is a command to launch your application, it internally runs many scripts to make sure that index.html is loaded and root component is added in your index.html, it has live development server for you

Note: You don’t have to launch index.html manually on live server from on wards

When you enter `npm start`, a development server launches the application in port 3000.



You will see a default content for the application that needs to be changed.

