React and Node.js Notes

1. React.js
2. Node.js
3. SQL & NoSQL database

Pre-requisites

1. HTML
2. CSS
3. Javascript (ES6 & latest)

Topics to be covered

1. HTML - basic HTML elements, Tables, Lists, Forms
2. CSS - classes, id, types of CSS
3. Javascript - basics to intermediate
4. React.js
5. Node.js
6. SQL & NoSQL

HTML: Hyper Text Markup Language, it is mainly used to display the content on the web page using HTML elements

CSS: Cascading Style Sheet, it is mainly used to style the HTML

JS: Javascript, it is mainly used to add interactivity to the web pages like dynamically changing the content, updating the color and so on.

Software requirement - for React.js & Node.js including HTML, CSS & Javascript

1. VS Code - Editor
2. Browser - Google Chrome
3. Node.js - node & npm commands
4. Permission to download the libraries from the internet

HTML:

It provides HTML elements like html, head, body, div, p, table, and etc.

Heading tags: These are used to display the headings in the web page, there are totally 6 heading tags like h1, h2, h3, h4, h5 and h6

<h1>Some Content</h1>

<h2>Some Content</h2>

Image tag: <img src = “url” width = “100p” height = “100” />

Anchor tag: It is used to create hyper links

<a href = “url”>Link Name</a>

Lists: These are the contents which are displayed in the form of lists either in the order or bullets, there are two types

1. ordered list <ol>
2. unordered list <ul>

both of these tags can display the items using <li>

<ol>

<li>Apples</li>  
 <li>Grapes</li>

</ol>

Tables: When you want to display the contents in a table form you can use <table> tag, along with <tr>, <th> & <td>

<tr>: It creates row

<th>: It creates columns in a bold font

<td>: It creates columns in a normal font

<table border = “1”>  
 <thead>  
 <tr>  
 <th>Heading1</th><th>Heading2</th>  
 </tr>  
 </thead>  
 <tbody>  
 <tr>  
 <td>Content1</td><td>Content2</td>  
 </tr>  
 <tr>  
 <td>Content1</td><td>Content2</td>  
  
 </tr>  
 </tbody>  
</table>

Forms:

When user wants to provide the inputs you can use the form

<form>  
 Enter Name <input type = ‘text’ /> <br />  
 Enter Age <input type = ‘number’ /> <br />  
 Select DOB <input type = ‘date’ /> <br />  
 <input type = ‘submit’ value = ‘Submit’ />  
 <input type = ‘reset’ value = ‘Reset’ />  
</form>

div:

It is a container tag, that can wrap multiple HTML elements together, so that you can style the container which applies to all the children’s of the div

CSS: Cascading Style Sheet

3 types of CSS

1. inline css: You can apply the styles to a particular element
2. internal css: You can apply to the entire HTML document
3. external css: You can apply style to multiple HTML documents, by creating a css file & referencing that css file

How to comment & uncomment: Highlight the lines and press Control and / together

3rd party css

bootstrap.css is one of the widely used external css, it provides 10000’s of inbuilt classes like .btn-primary, alert-danger, alert-primary, text-success, alert-success, container, row, col, table, table-striped and so on

Javascript:

It is a programming language used to provide effects to the HTML, it allows you to dynamically access HTML & CSS so that you can modify the DOM (Document Object Model) which is a tree structure formed when the HTML loaded on the browser

It provides many fundamentals like

* variables
* functions
* event handling
* objects
* arrays

Example on how to access and update HTML



let & const keyword: It is a new keyword introduced in ES6(new feature of Javascript), ES stands for ECMA Script, let & const are the keywords you can use to declare variables, you should avoid using var keyword, because it doesn’t have a scope.

let variables can be modified, however const variables can’t be modified.

How to use arrays in Javascript

Array is a container to store multiple values, in Javascript you use [] to create an array.

let numbers = [20, 30, 10, 50, 40];

let fruits = [“Apple”, “Mango”, “Grapes”];

How to use objects

let employee = {id:100, name:”Rajesh”, salary:35000};

Array of Objects

let employeeArray = [{id:200, name:”Raj”}, {id:300, name:”Vijay”}, {id:400,name:”Ajay”}];

Array iterations

1. It can be done using for loop.
2. It can be done using an inbuilt forEach function, that does the internal iteration
3. It can be done using an inbuilt map function, that is similar to forEach, but it returns a new array after iterating the elements, which can be used to transform the iterated elements.

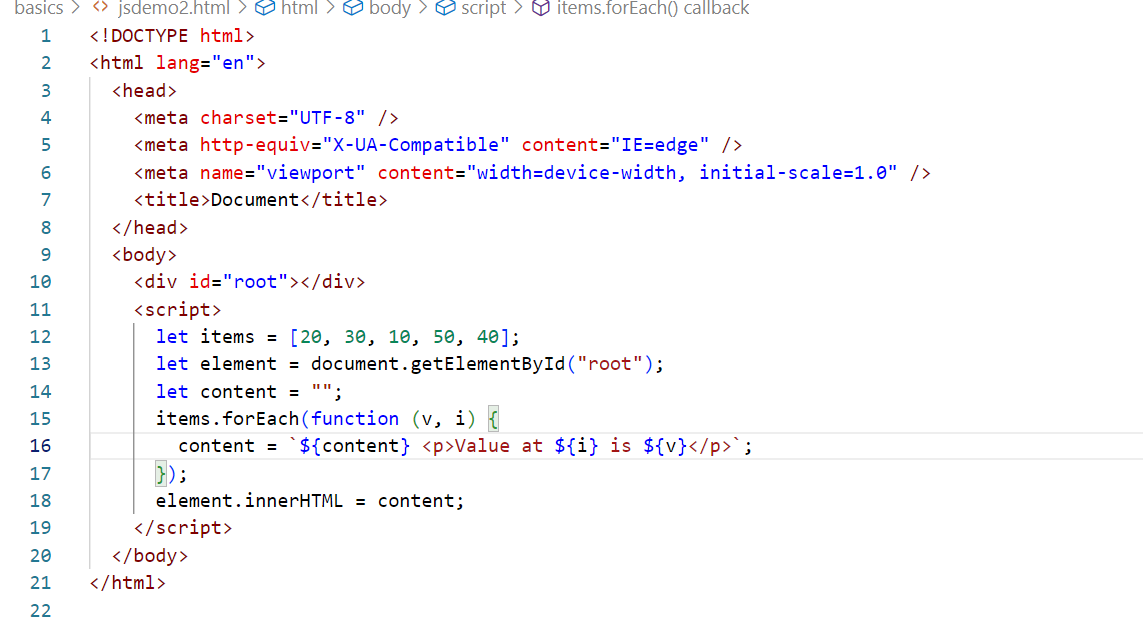
let items = [20, 30, 40, 10, 50];

items.forEach(callbackFn);

callbackFn: it is a callback function, takes 2 arguments which is an element that is iterated and the index of the iterated element.

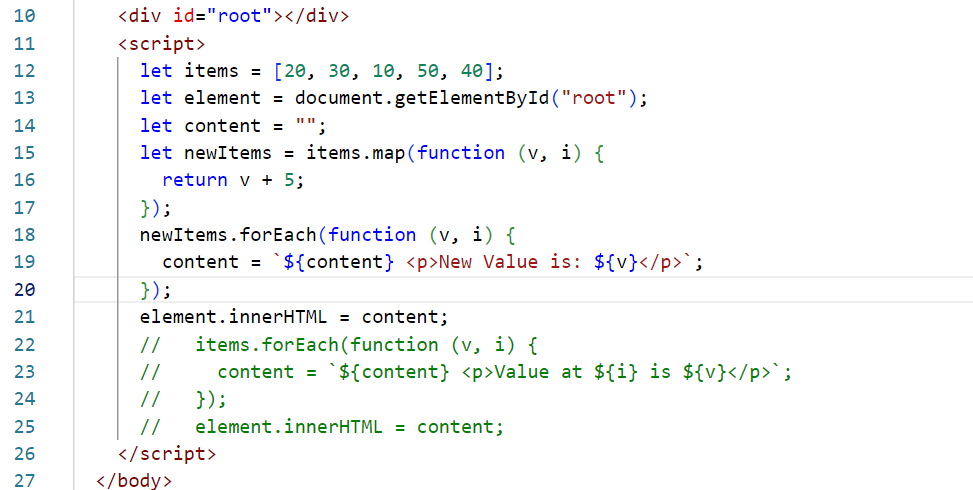
items.forEach( function(v, i) {  
 v : value of the iterated element  
 i : index of the iterated element  
});

Array iteration using forEach



Map: It is an inbuilt function to iterate an array & return each element into another array after the transformation

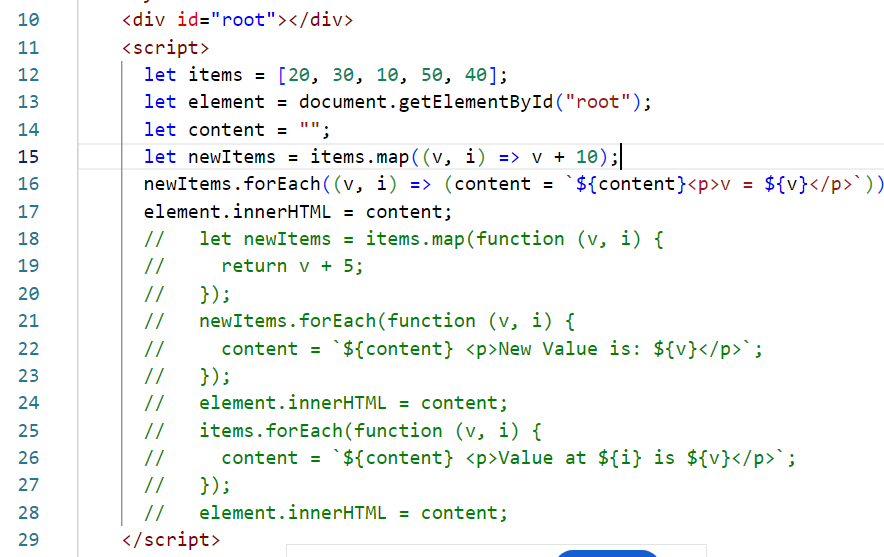
let newItems = items.map(function(value, index) { } );



arrow functions

ES6 introduced arrow functions to simplify writing the anonymous functions/callback functions

|  |  |
| --- | --- |
| Callback Function | Arrow Function |
| function(x, y) {   stmt1;  return value; } | (x, y) => {  stmt1;  return value; } |
| function (x, y) {   return value; } | (x, y) => value;  [or]  (x, y) => { return value; } |
| function(x, y) { stmt1; } | (x, y) => stmt1; |
| function(x) {  stmt1;  } | x => stmt1; |
| ex: items.forEach(function(v, i) { } ); | ex: items.forEach( (v, i) => {} ) |
| ex: items.map( function(v, i) { return v+5; }); | ex: items.map((v, i) => v + 5; ); |



Destructuring objects & arrays

Suppose you have an object as below:

let employee = { id : 100, name : “Raj”, salary:25000, desig: “Sales”, address: {state: “KA”, city:”BLR”} };

If you want to assign the property to a variable you need to access using the dot operator

let id = employee.id;  
let name = employee.name;  
let salary = employee.salary;  
let desig = employee.desig;  
let state = employee.address.state;

Using employee.name at many places would be difficult if there are too many nested properties   
With destructuring its much more simpler

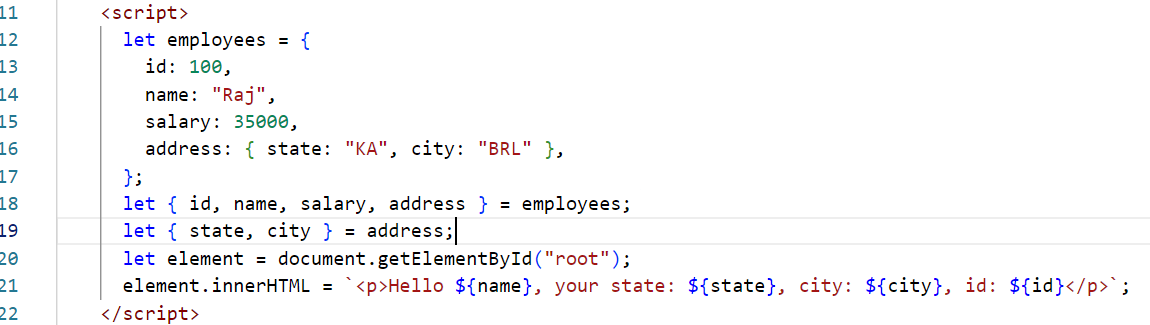
let {id, name, salary, address} = employee;  
let {state, city} = address;

you can use id, name, salary, address directly without using employee.name or employee.address

You can also destructure arrays, but use [ ].

let items = [30, 40, 10]

let [x, y, z] = items; // x = 30, y = 40, z = 10;



React.js

It is a Javascript library used to develop rich UI’s, it helps you to create reusable UI’s which are called as components to build SPA (Single Page Applications).

React.js uses a special syntax i.e., JSX

JSX: It stands for Javascript XML or Javascript Extension, this supports all the JS features along with other features to simplify writing the HTML inside the Javascript

How JSX is written

let content = <p>Some content</p>

To print the content use { content }

let employee = { id : 100, name : “Raj”, salary : 35000 }

<div>{ employee.id } { employee.name } {employee.salary}</div>

Using expressions & JS function in JSX

<p>{ 2 + 3 }</p>

<p>{ Math.pow(3, 2) } </p>// print 9

React.js uses 3 libraries to develop the application

1. React
2. React DOM
3. Babel: Converts JSX to Javascript, so that browser understands

Components: These are the visible part in the page, you can independently create the components & use with other component to create a complex component & build an application

Root component: This is the main component which represents the entire page, it will have other components in it, you need to use all the components inside the root component, because this is the only component loaded to the HTML

Online website to quickly develop create components

Codepen is the website that helps you to quickly create react components & use in the HTML.

ReactDOM: It is a library that accesses the HTML element in the DOM to map the component developed in the React

let root = ReactDOM.createRoot( document.getElementById(“root”) );

root.render( <RootComponentTag /> )

function RootComponentTag() { return complex components }

Hello World program in React

