MERN Stack Training

Contents

* Web Applications
* Fundamentals of Computer Programming Language
* Algorithms & Pseudocode
* Linux
* Git
* HTML, CSS, Javascript
* React.js, Redux, React Router
* Node.js
* Express.js
* MongoDB
* Testing
* Deployment

Application: It is a computer program which perform some tasks, there are two types of applications

1. Standalone application: These are the applications which you can use only by running on particular devices like mobile applications/desktop applications
2. Distributed application: These are the applications which you can access over the internet, there are two types of distribute applications
   1. Web applications
   2. Web services

Frontend & Backend



When you divide the applications into multiple layers like view, controller & model then you can have view in various types like mobiles, web browser, devices (Swiping machine, ATM Machine, …) and also backend can have webservices that helps to take data from any applications and return the data to any applications

Technologies used to create front-end & back-end applications

For Front-end

1. HTML, CSS, Javascript
2. React.js
3. Angular Framework

For Back-end

1. Javascript - Node.js & Express.js
2. Java - Spring Boot
3. Python
4. ASP.NET

Database: It is used by the backend applications to maintain the data, some of the database are: MongoDB, OracleDB, MySQL and so on

Front-end & Back-end together forms Fullstack development, a full stack development needs multiple technologies that needs to be connected

MERN stack: MongoDB, Express.js, React.js, Node.js

Evolution of Programming Languages

MLL: It is the language computers understand

C/C++: It is an high level languages but needs to be compiled to machine code, it is platform dependent

Java: It is also an high level language and its platform independent, it is more-preferred to write backend applications

JavaScript: It is also an high level language which is understood by browser & node.js, it can be used to develop both front-end & back-end applications

Before you write your first program you must know how to write instructions to the computers hence we need to have an idea on programming fundamentals like

1. Algorithms
2. Flowchart
3. Pseudocode

Note: These are not understood by any computers, these are for programmers to help writing code in different technologies like Javascript, C, C++, Java

Softwares required for MERN stack training

1. OS: Windows or Mac or Linux (UI-based)
2. RAM: Min 4GB
3. Privileges: Permission to download softwares/install them in your machine.
4. Virtual box - to learn linux
5. Git - to share your code with other team members in your company projects
6. VSCode - editor to write javascript code
7. Node.js - runtime environment to run javascript at the backend
8. MongoDB - database to maintain the data
9. Browser

Algorithm:

* It is a step by step procedure which are followed to get the desired result for the problem statement
* It is independent from any programming language
* It is not understood by any computer/compilers
* It can be written in any ways, there’s no any defined rules it must written in some standards
* It can be used to write programs which are understood by computers

Suppose you want to add two numbers

Step1: Start

Step2: Read number1

Step3: Read number2

Step4: Add number1 and number2 and store in a result

Step5: Print the result

Step6: End

Another way to write algorithms to add two numbers

Step1: Start

Step2: Take two numbers

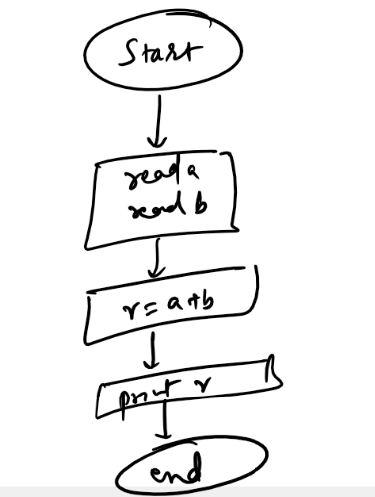
Step3: Add those two numbers and store in a result

Step4: Print result

Step5: End

Flowchart:

* It is a graphical notation of program sequence
* It is also language independent



Pseudocode:

It is also language independent, it can mimic the programming language by specifying programming constructs that needs to be used to solve problem

Note: This is also not understood by any computers

Ex: Pseudocode to add two numbers

Read x

Read y

r = x + y

print r

Pseudocode to find maximum of two numbers

Read x

Read y

if x > y then x is maximum

else y is maximum

Writing the actual programs in various languages

C program to add two numbers

void add() {   
 int x, y, z  
 printf(“Read x”);  
 scanf(“%d”, &x);  
 printf(“Read y”);  
 scanf(“%d”, &y);  
 z = x + y;  
 printf(“result = %d”, z);  
}

Java program to add two numbers

void add() {   
 int x, y, z;  
 System.out.println(“Read x”);  
 x = scanner.nextInt();   
 System.out.println(“Read y”);  
 y = scanner.nextInt();  
 z = x + y;  
 System.out.println(“result = “+z);  
}

Javascript program to add two numbers

x = prompt(“Read x”);  
y = prompt(“Read y”);

z = x + y;  
alert(z); // shows the result in dialogbox  
console.log(z);

Linux

It is a popular server OS used to deploy high end software’s, business oriented applications

* it is very powerful and faster than any UI based OS
* It is open source, it means any body can customize the Linux OS as per their need

What we must know from Linux

We must understand how to use linux commands either to open a file, write a file, open a folder, navigate to different path, install software’s

Important commands of Linux

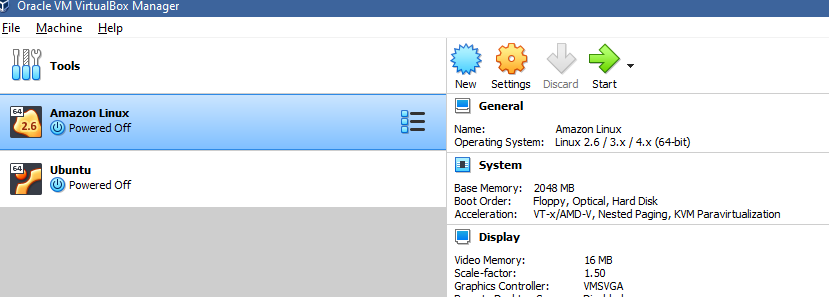
* ls
* pwd
* cat
* date
* cal
* touch
* rm
* cp
* cd

How to install linux in virtual box

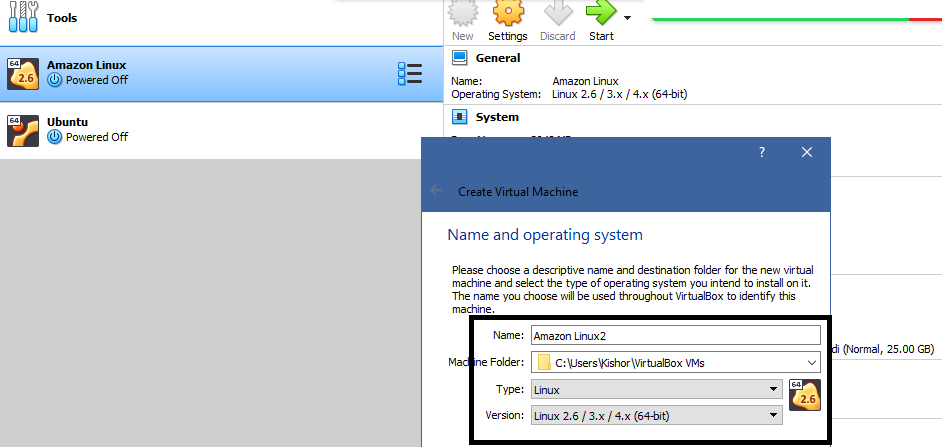
Follow this link

<https://www.how2shout.com/linux/download-amazon-linux-2-to-install-on-virtualbox/>

Step1: Open Virtual Box

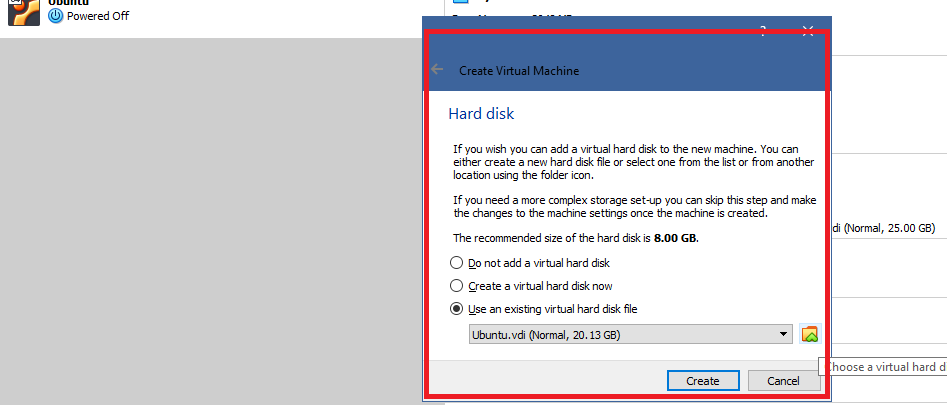


Step2: Click on New & Enter name & select type as Linux



Step3: Click Next & choose Ram size

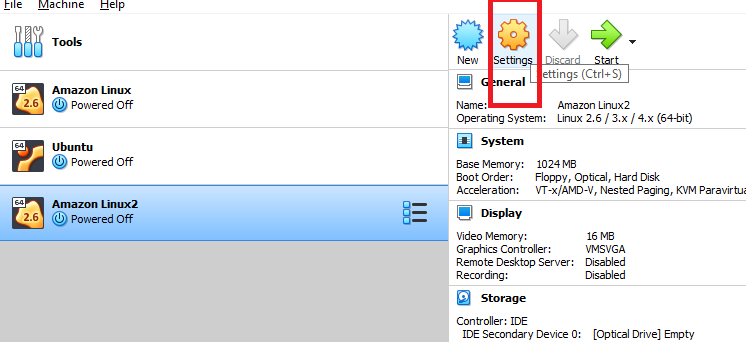
Step4: Select Use an existing virtual machine



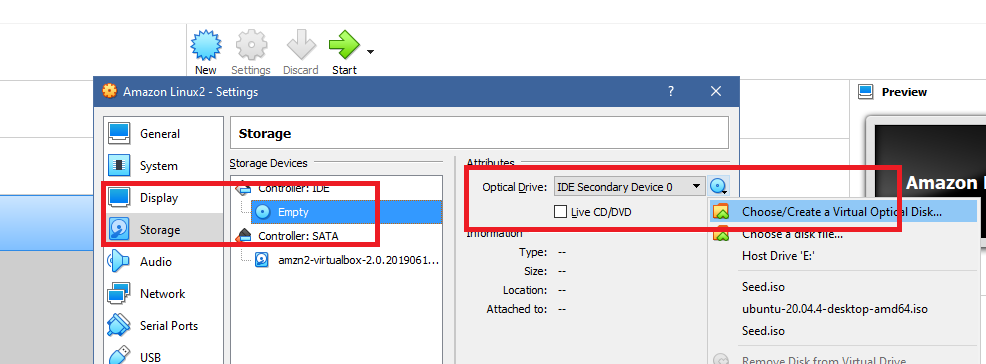
Choose virtual hard disk file, you will get an Add icon, select that you will need to browse and select the .vdi file which you have downloaded.

Step5: Click on Create

Step6: Select Settings



Step7: Choose Storage and Controller-> Empty, there you select the disc icon, where you will get choose/create a virtual optical disk



Step8: Click on Add and select seed.iso file

Step9: You can start the virtual machine, which will install linux on the virtual box

Note: Default username and password is

Username = ec2-user

Password = amazon

Note: When you enter password you may not see how many characters you are entering, just enter all the characters and hit enter key

Understanding Linux commands

pwd: Shows the present working directory

ls: Lists all the files & folders in the present working directory

mkdir: Used to create folder/directory

cd: To navigate from one folder to another folder

touch: To create files

cal: It shows the calendar

Steps:

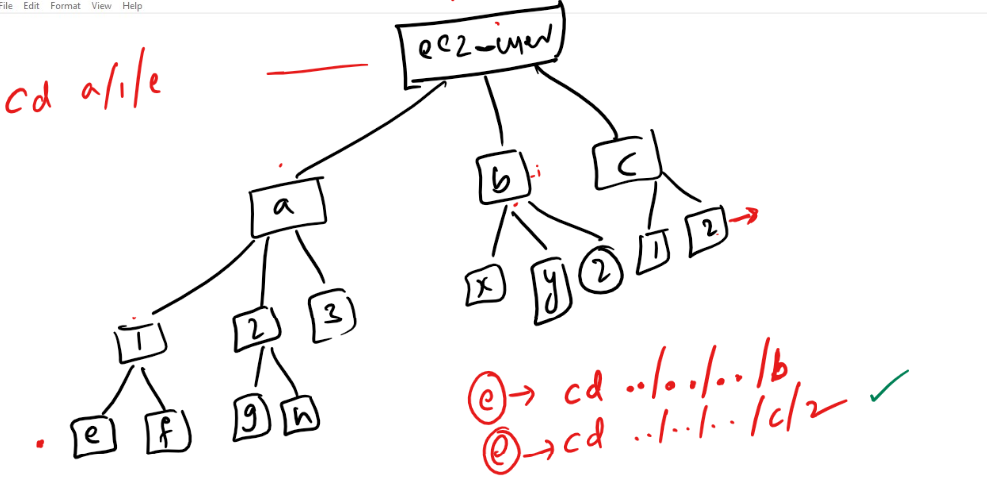
touch test.txt

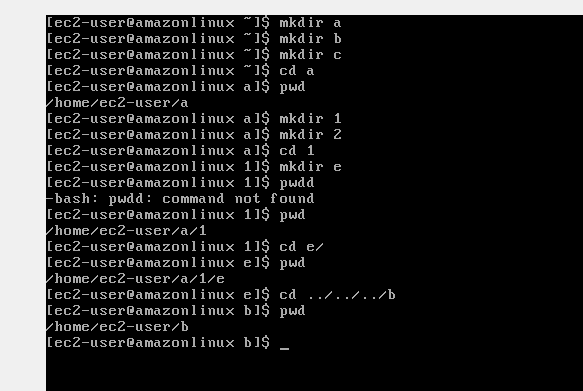
PRESS INS KEY

write some content

PRESS ESC KEY

Type :wq (Which appears at the bottom of the editor)





Day 3:

Agenda:

1. Linux grep and shell script
2. GIT

Grep: Global Regular Expression Print

It is mainly used to search the characters in the file and also you search various files & folders in your linux platform.

ex: grep pattern filename.txt



Shell Script

These are list of instructions that are stored in a .sh file, this simplifies user to keep frequently executing commands in one file and run it through script runner which is called as ‘bash’, this bash is present in /bin/sh location that can run any script files.

Note: You must begin every script file using #!/bin/sh so that the script will be run by the bash

Syntax of the script

echo “Some content”

The above line will print the expression you have written in the string

Creating variables

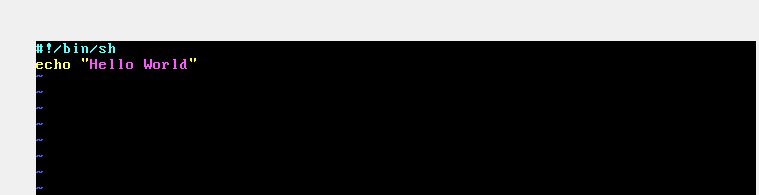
x = 20

y = 30

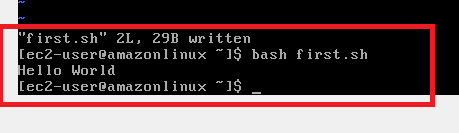
echo $x  
echo $y

The above command stores 20 to x & 30 to y and also prints their values.

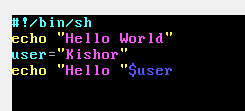
first.sh



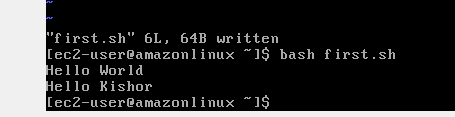
Use bash first.sh to execute the script



Storing the name in the variable

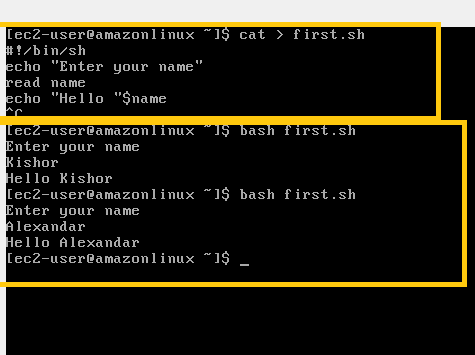


Output:



Reading the input from the user

syntax: read variable\_name



Calculating the data

#!/bin/sh

echo “Enter num1”

read num1

echo “Enter num2”

read num2

res = $(($num1 + $num2))

echo “Result = “$res

Output:

>> bash filename.sh

Summary

1. Linux commands
2. Grep
3. Shell Scripting

Git:

It is a distributed version controlling system or centralized version controlling system which keeps track everyone’s work through some version numbers that is internally generated by the git, the version number is an unique 40characters id that will help git to internally identify the changes done by the user.

Git will also help in collaborating the work done by multiple users within the same team and also it will maintain everyone’s work in a remote repository that is available to all the users and the same work will be maintained in local repository which is available locally to each user in their local machine.

Repository:

It is a working directory that will have files & folders and git version controlling folder that will track the changes, there are 2 types of repositories

1. Remote repository: It is a working directory which is present in the git hub and accessible over the internet
2. Local Repository: It is a working directory which is present in the user local machine

How git can collaborate everyone’s work

Git gives lot of commands that you can enter through Git bash terminal, some of the useful commands are add, commit, pull, push, merge, diff, log and so on.

How to download the remote repository in our local machine

Git gives a command called clone that needs to be used with the url of the remote repository

i.e., git clone remote-url

Note: The above command must be entered through git bash and also ensure the git bash opens in a folder that you can easily remember

ex:

git clone <https://github.com/Kishor-C/git-demo-2310.git>

How to update the local repository changes to the remote repository

git add filename [or] git add .

git commit -m ‘some message’

git push -u origin master [or] git push -u origin main

Note: Git throws author identity unknown on the very first time when recently installed git, hence you need to set user.name & user.email

Run below two commands

git config --global user.email “yourmail-id@domain.com”

git config --global user.name “your name”

ex:

git config --global user.email “kishor1124@gmail.com”  
git config --global user.name “Kishor-C”

Note: After this enter git commit -m ‘some message’

Summary of basic git clone & push

1. Created Remote repository
2. Cloned Remote repository: ‘git clone <<remote-url>>’
3. Created a file: ‘touch 1.txt’
4. Added to the staging area: ‘git add 1.txt’
5. Saved the changes: ‘git commit -m ‘some message’ [you need to configure user.name & user.email for the first time]
6. Push the changes: ‘git push -u origin main[or]master’ [you need to authorize the permission with git credential manager for the first time]

Branch: It is a pointer that will have history of commits done by the user, by default git gives a main branch which is also called as master branch, but we can also create custom branch which is called as feature branch

How to see the commits done by the user

git log is the command used in the local machine to see the commits done by various users

Activity:

create another file with the name 2.txt, then push it to the remote, ensure you are adding the changes to the staging area, commit and then push

Git cycle

Working Directory -> Staging Area -> Commit -> Push

Working Directory will have the changes done

Staging Area will have the changes that needs to be tracked

Commit will save all the changes in the staging area

Push will push the changes to the remote that are committed.

Some commands list

git status: It shows the changes which are tracked & untracked in the staging area

git add . : it adds all the changes to the staging area

git add file/directory: It adds the file/directory to the staging area

git log: It lists all the commits of a branch

git commit -m ‘some message’: it commits the changes tracked in the staging area.

Git Fork  
It is used to create a clone of a repository from others account in your account, the repository created in your account will be completely independent

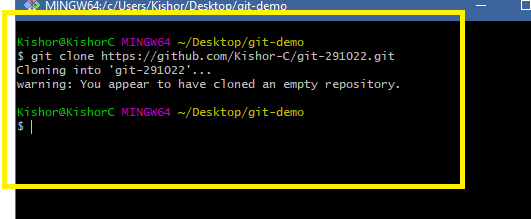
Activity

1. Clone the forked repository from your account to your local machine in a new folder which is not part of your old local repository
2. Create some files and push those changes to the remote
3. Once Done let me know

Summary of Git process

1. Firstly create a remote repository
2. Clone that remote repository in your local machine

git clone url-of-repository



Note: Ensure you are not inside an existing repository while clone a new repository

1. Navigate to the git working directory first using ‘cd’ command

cd git-291022

1. Make some changes

touch hello.txt

1. Add the changes to the staging area

git add .

1. Commit the changes with a meaningful message

git commit -m ‘created hello file’

1. Push the changes to the remote

git push -u origin branch-name >> master/main/custom-branch

Apart from these commands there are some more useful commands like

1. git log --oneline
2. git status
3. git branch
4. git checkout
5. git merge

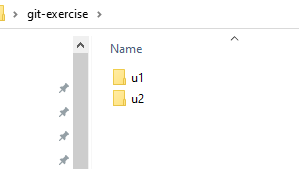
Exercise

Making changes in the same file by multiple users and updating in the remote repository

1. Create a new repository remotely with some name
2. Clone that repository in your local machine using `git clone remote-repo-url`
3. Navigate to the repository & create a new file and push that file to the remote
   1. touch 1.txt
   2. git add .
   3. git commit -m ‘create 1.txt’
   4. git push -u origin master [ or ] main

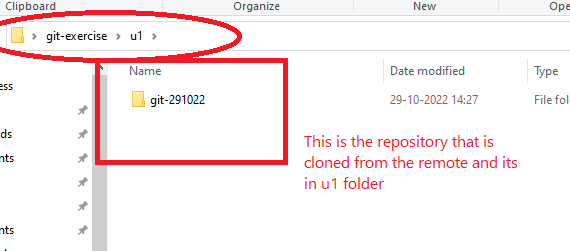
Note: Before creating 2 folders you must have atleast one commit hence you need to all the above steps

1. In local machine create 2 folders with the name u1 & u2 in a non-git repository



1. Ensure both the folders have the same repository

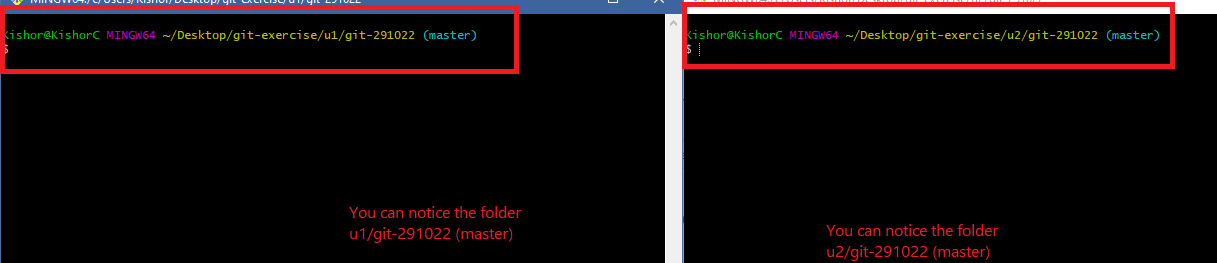
Repository in u1 >> git-291022



Repository in u2 >> git-291022

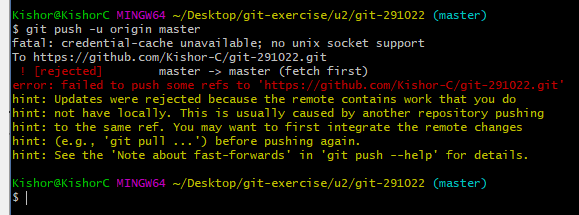


1. Open Git bash from both the folders repository

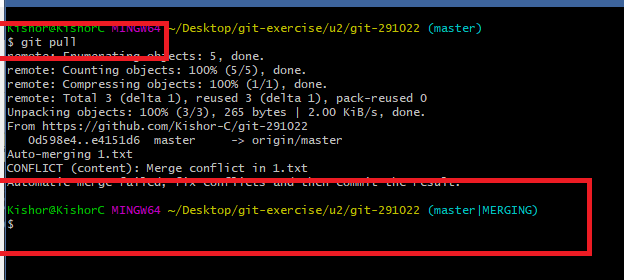


1. In u1 repository edit the file 1.txt and push that to the remote using below commands
   1. git add .
   2. git commit -m ‘modified 1.txt’
   3. git push -u origin master
2. In u2 repository edit the same file 1.txt and push that to the remote using below commands
   1. git add .
   2. git commit -m ‘modified 1.txt’
   3. git push -u origin master

Note: You will notice the push rejection

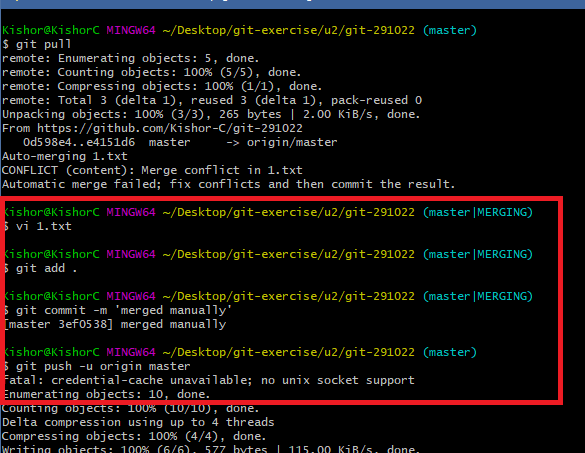


1. In u2 repository you must use `git pull`>> this will give a new branch saying auto-merge conflict



Use vi and edit the 1.txt where you will see some random characters

1. Once you edit 1.txt you need to re-add & re-commit the changes & push the work to the remote



1. Now you can see the remote & local repository uptodate
2. You can enter git pull in the u1 repository (ensure u1 has not done any new commits) to see the updates done by u2 repository

Rules we need to follow while working in the GIT

1. Never work in the master/main branch in the local repository
2. Any changes use wants do must be done in feature branch which is a copy of the master/main branch
3. Push only the feature branch from the local repository and pull the master/main branch from remote repository, it will always ensure that master/main branch in all the machines are clean and encourages the user at the server side to review everyone’s work
4. Someone at the remote repository can have a control over merging of branches, this user can also update others to resolve the merge conflicts if at the merge conflict occurs at the remote side
5. Local master must always get the changes from the Remote master
6. Local feature branch should be merged with the changes available in Local master after it pulls the changes from remote master

Note: It’s a good practice to delete the feature branch that is pushed in the remote repository, if not deleted then whoever pulls the remote will get all the branches present in the remote master, its okay if the feature branch is not deleted in the local repository, may be the use wish to work in the same branch

How to create a new branch

git branch branch-name

ex: git branch issue-runtime-error

ex: git branch branch1-user1

How to switch from one branch to another

git checkout branch-name

ex: git checkout issue-runtime-error

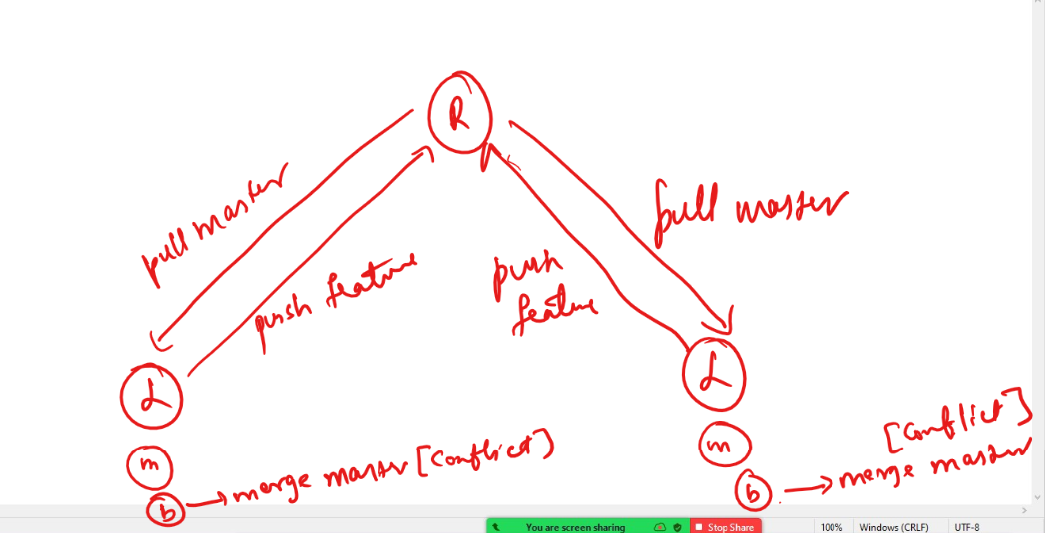
ex: git checkout branch1-user1

How to push the feature branch

git push -u origin branch-name

How to merge the local master with the feature branch

git merge master [or] main



Summary of how to work in GIT

1. Create feature branch & make changes in the feature branch and push the feature branch to the remote
2. If there’s a merge conflict in the remote then pull the remote to the local master and checkout to the feature branch, then merge the local master with the feature branch & then push the feature branch

List of commands to follow

1. git branch b1
2. git checkout b1
3. git add
4. git commit
5. git push -u origin b1
6. Go to git-hub and merge the b1 with remote master
   1. If merge is success then delete the b1 in the remote
   2. If merge fails then also delete the b1 in the remote, but in the local pull the remote master and checkout to the b1 and merge the local master with b1 using ‘git merge master’

Note: While using git pull, checkout to master/main using ‘git checkout master’ or ‘git checkout main’

Activity

1. In local repository create a feature branch with the name b1 for user1 and another branch b2 for user2
2. Make changes in each branch and push that branch to the remote & merge the branch in remote, use the command git push -u origin branch-name to push the branch to the remote
3. Try to edit the same file in both the feature branches and push the changes to the remote and observe the merge conflict you get.
4. If you get the merge conflict use ‘git pull’ in master branch & checkout to the feature branch and use ‘git merge master’ or ‘git merge main’, then resolve the merge conflict by editing the file and push the changes to the remote

Use the following commands to perform these activity

branch-creation: git branch branch-name

checkout: git checkout branch-name

.gitignore:

It will have list of files & folders that are not tracked if changes are done

HTML

Software required:

* Visual Studio Code

<https://code.visualstudio.com/>

UI Technologies: It is used to develop front-end applications, basically there are 3 main technologies we must know

* HTML
* CSS
* Javascript

HTML: Stands for Hyper Text Markup Language, it is used to display the content on the web page, it could be heading, paragraph, lists, tables, forms, buttons, text box, password box and so on

HTML are written using tags, they are understood by browser, so that it knows how to display the content

Tag: It is a name that tells how the content should look like, a tag will come in pairs most of the times, like start tag & end tag

Element: Start tag & End tag, including its content is called as HTML element

ex:

<p>some content</p>

Ex: <html>…..</html>, <p>….</p>, <h1>….</h1>

CSS: Stands for Cascading Style Sheet, it is used to style or design HTML elements, i.e., it is to style the web page

Javascript: It is a programming language for HTML & CSS, it can add dynamic behaviour to your web page

HTML Document

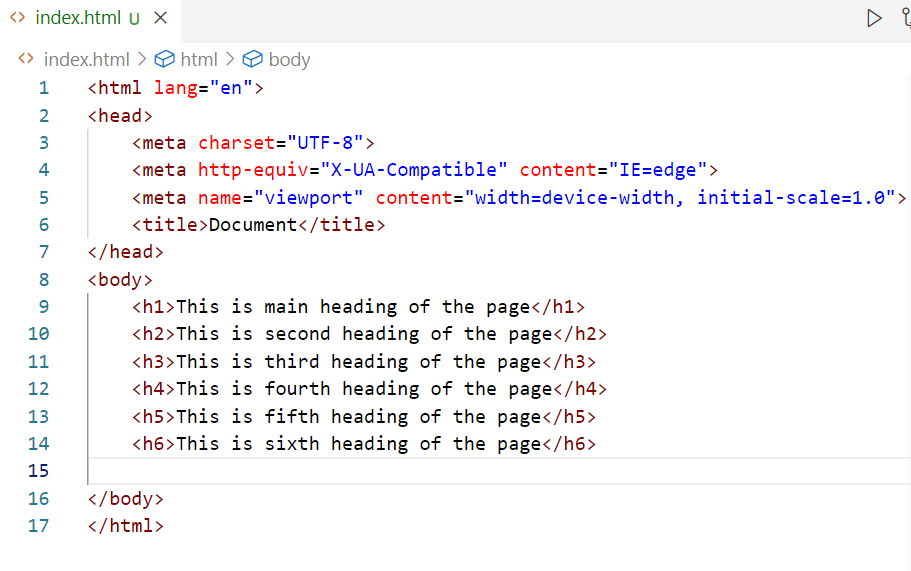
<html>

<head>  
 </head>  
 <body>  
  
 </body>  
</html>

Heading tags

There are 6 heading tags each size varies from larger font to smaller

h1 to h6



marquee: It is used to move the element

Image: You can use <img> tag, which is takes attributes like src, width, height, it is a self closing tag means you don’t have to close it

<img src = “url” width = “value” height = “value” />

Entities: These are some special characters that can be shown on the web page, there is an entity name or entity number for the special characters.

&copy; This gives copyright symbol

&reg; This gives registration symbol

&lt; This gives < symbol

&gt; This gives > symbol

&#8377; This is to display rupee symbol

&yen; This is to display Japanese currency

&pound; This is to display Euro currency

Lists: When you want to list the items you can use ordered list or unordered list, you have tag like <ol> & <ul>, both these tags take <li> as the child tag to list the items

Container tags

These are the tags which can contain other tags like child tags, container tags are mainly used to layout with the help of CSS, you just need to apply the design to the container so that it affects to all the elements inside the container

<div>  
<header>  
<footer>  
<nav>  
<section>  
<article>

All these tags display the content without any layout, but you must style these to get the effect

Table tag

It is used to create contents to the table, HTML provides <table> to create table, <tr> to create rows & <th> to create headings for the columns, <td> to show the table data

Form tag

It is used to create controls that allow user to enter input, like input box, password box, radio button, checkbox and so on

Create a form that will have below labels & controls

Enter Name <<text box>>

Enter Password <<password box>>

Gender <<radio button>> Male <<radio button>> Female

Skills <<checkbox>> HTML <<checkbox>> Javascript <<checkbox>> CSS

Educational Qualification <<drop-down-list>>

<<submit>> <<reset>>

Agenda:

* HTML 5 Features
* CSS 3

HTML Revision

* container tags - div, header, footer, nav, section
* inline tags - b, i, u, stroke
* self closing tags - hr, br, img,
* heading tags - h1 to h6
* table, list, forms

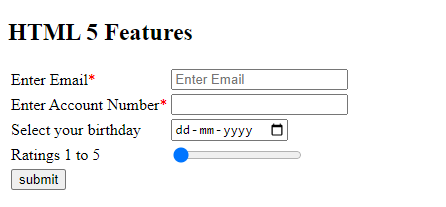
HTML 5 feature

* Added placeholder, required attributes to the form control
* Added different types to the input like email, date, number, range
* Media elements to load video & audio
* Storage - local storage & session storage (We will discuss this in Javascript)
* canvas element (we will discuss this is Javascript)

Form with number, date, email, range



Output:



Media elements

Earlier browser need to add extensions to load audio or video, but from HTML5 onwards user can use <audio> & <video> tags to load the media files

Note: audio must be .mp3 file & video must be .mp4 file

<audio width = “200” height = “200” controls >  
 <source src = “filename.mp3” type = “audio/mp3”></source>  
</audio>

<video width = “200” height = “200” controls>  
 <source src = “filename.mp4” type = “video/mp4”></source>  
</video>



CSS3:

Cascading Style Sheet is used to style the HTML elements, there are 3 types of styles you can apply

1. Inline style: You can apply for a particular HTML element
2. Internal style: You can apply to the entire HTML document
3. External style: You can apply to multiple HTML documents & styles are written outside the HTML document

CSS Properties

color, background-color, border, animation, transform, font-size, font-family, padding, margin, width, height, …

These properties take values accordingly

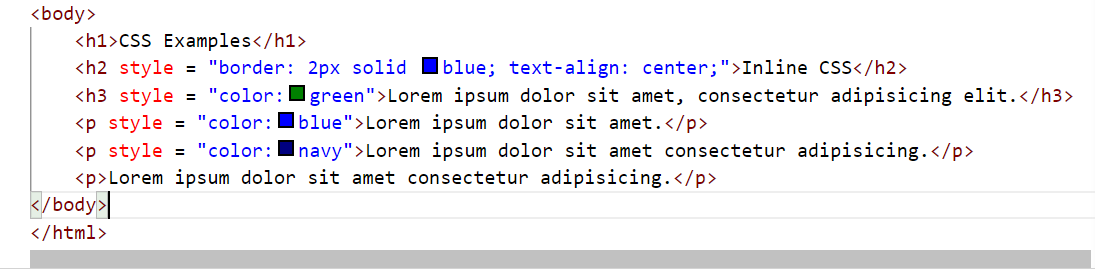
ex: color:red, border: 3px solid black

ex: font-size:10px;

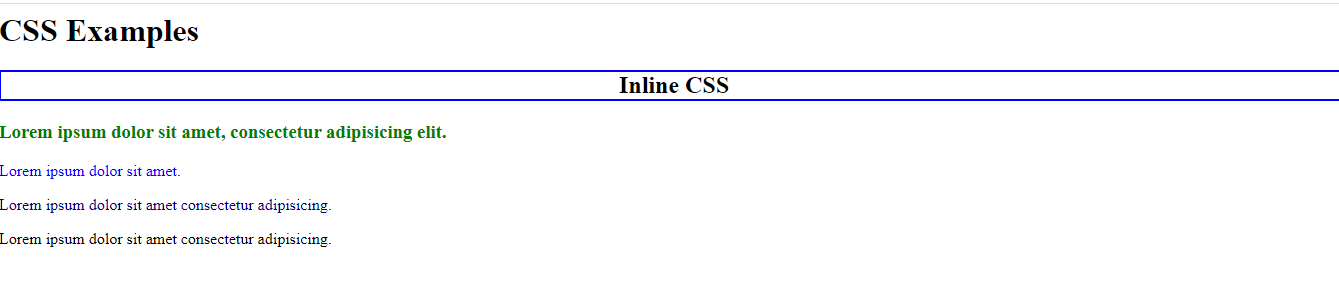
ex: padding: 3px;

ex: font-family: arial

Inline CSS: CSS can applied on a particular element using style attribute



Output:



Internal CSS

You can apply the CSS to the whole HTML document, inline CSS can override Internal CSS

<style>  
 p { color: red; background-color: yellow }  
 h1 { color : blue; background-color: yellow }  
</style>

Selectors:

tag selector: tagName { css property : value }

class selector: .className { css property : value }

id selector: #idName { css property : value }

Inline CSS: <p style = “color: blue”>Some content</p>

Internal CSS:   
<style>  
 p { color : red; }  
</style>

External CSS:

p { color : blue }

HTML code

External CSS with <link>  
Internal CSS with <style>  
Inline CSS with style attribute

Animations:

They add smooth transitional effects to the HTML elements for a particular duration

You need to use @keyframes to apply animation rules, the name that has the rules needs to be used by the HTML element

@keyframes name { rules }

selector { animation: name }

12-11-2022 Agenda

CSS Media Queries

CSS Flex Box

CSS Grid Layout

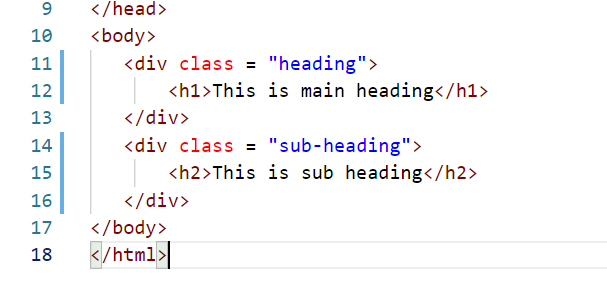
Javascript

CSS Media Queries

It is a CSS technique which applies the CSS properties to the HTML elements only if certain conditions are met.

@media selector is used to apply the condition.

HTML code

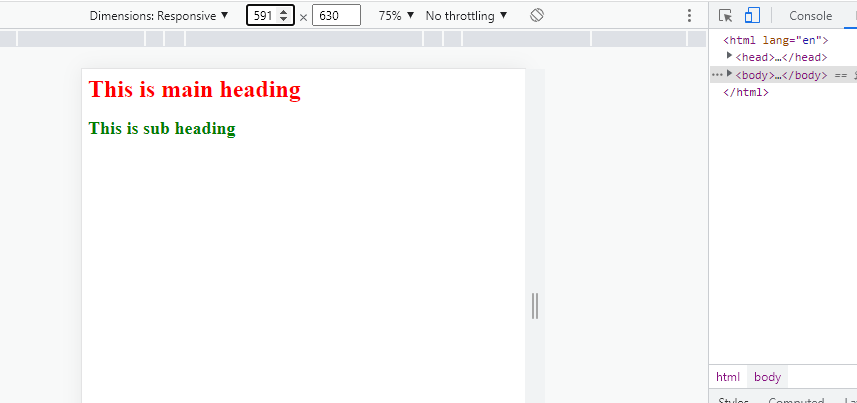


styles.css



Output:

You can see the styles are applied based on the condition when you expand or shrink your page layout

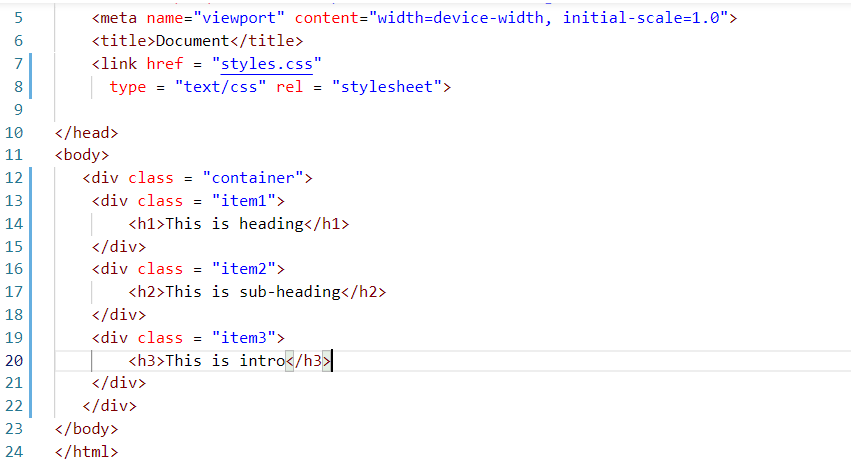


Flexbox layout:

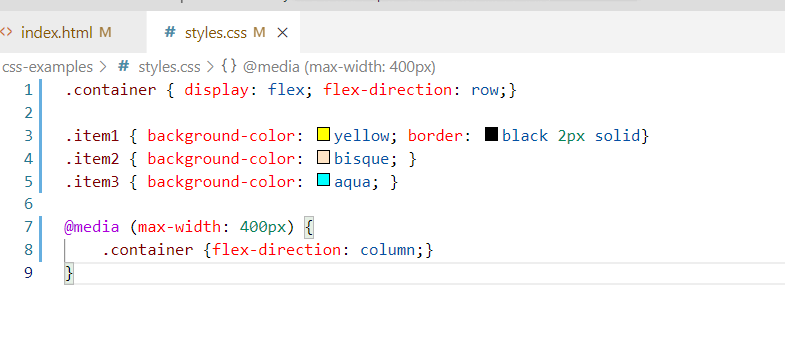
It provides an efficient way to align & distribute the items as per the device width & height.

A flex box container can expand or shrink whenever need, it can be achieved using a property called display: flex;

HTML

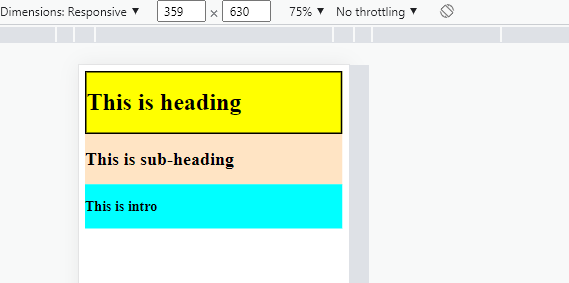


CSS



Output:

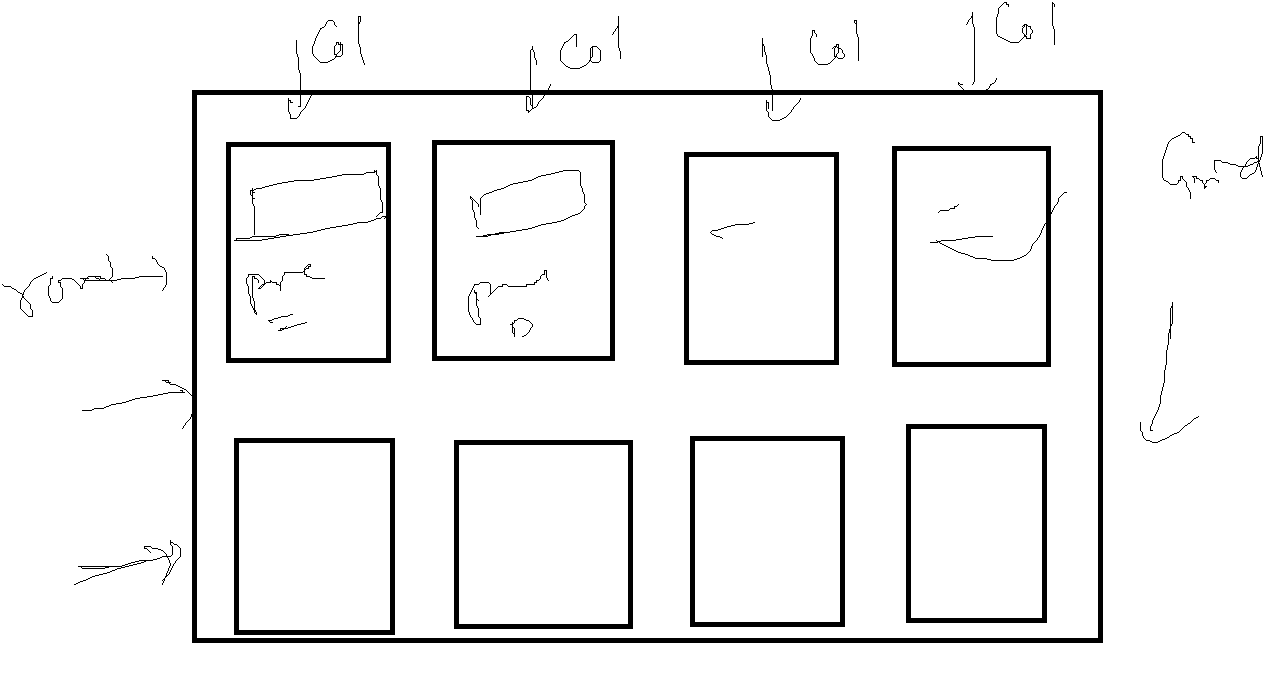
You can see the <div> inside <div class = “container”> are arranged columns wise because of @media rules



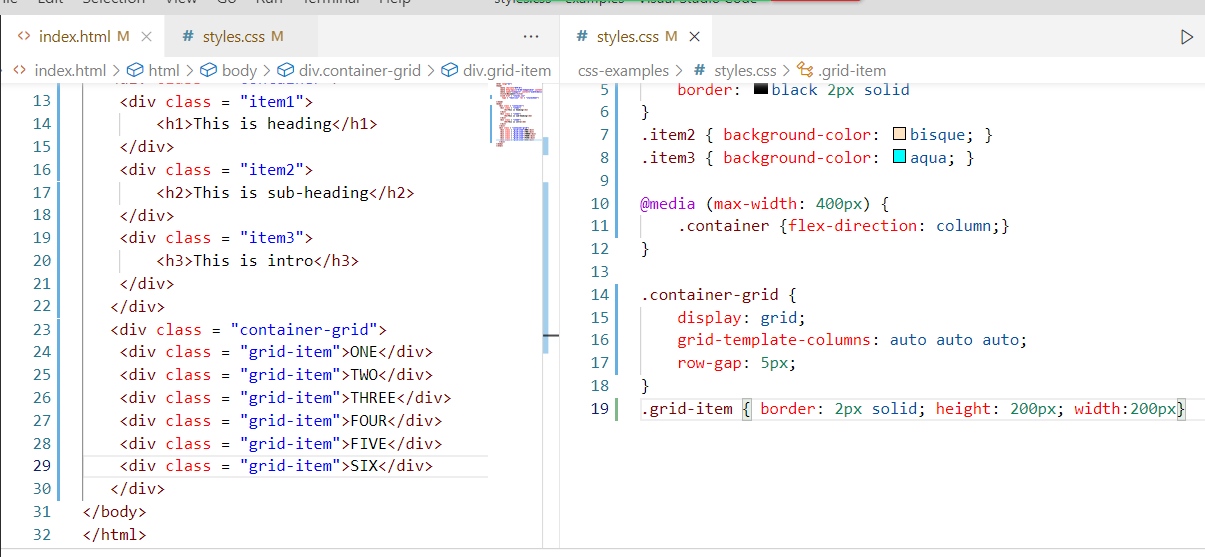
Grid system

You want to add the elements within a grid like row & col wise then you can use grid layout

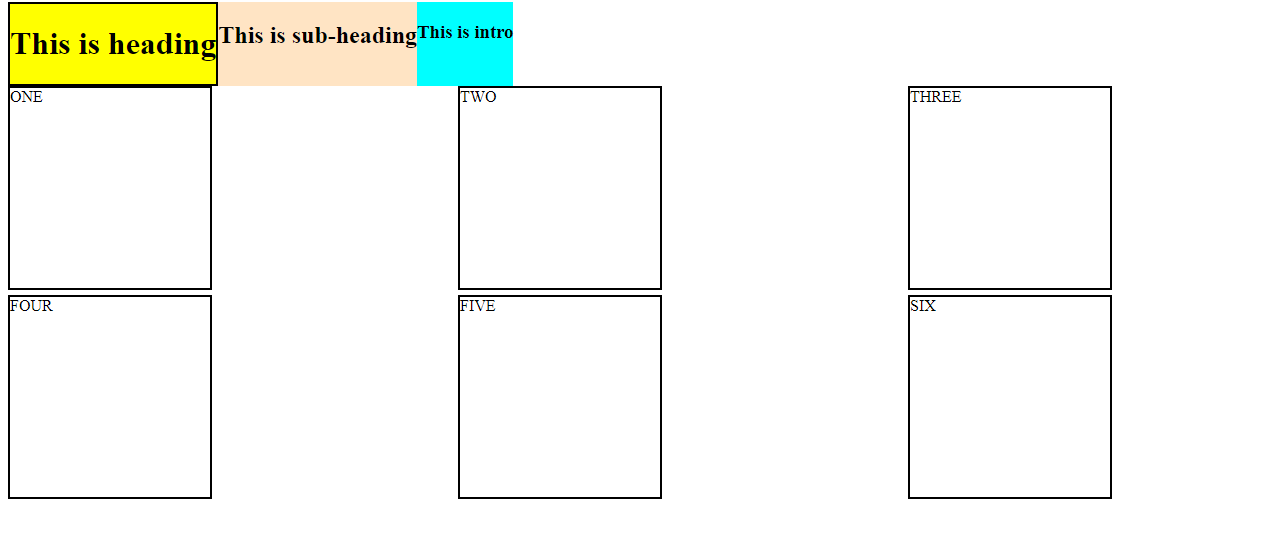
Here a container can use a property & value called display: grid and all the elements inside this container can be arranged in row & col wise



HTML & CSS



Output:



HTML: Display the content  
CSS: Style the HTML  
Javascript: It takes care of adding dynamic behaviour to the web pages, it can access HTML at runtime & can modify it, also it can apply CSS at runtime

* Javascript was used to add effects at the front-end in the early stage of Javascript
* But now Javascript can also be used to write back-end programs like connecting to database, server programs, access filesystems

index.html



Fundamentals of Javascript

* Variables
* Operators
* Conditions
* Loops
* Arrays
* Functions
* Objects
* Events

Variables:

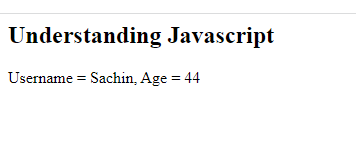
These are memory that can store some value like strings, numbers, Booleans, objects and etc.

var username = “Sachin”;

var isEmployed = true;



Output:



Operators:

Operators are special characters which can perform operations on the variables, like addition, subtraction, multiplication, division, modulus, comparison and etc.

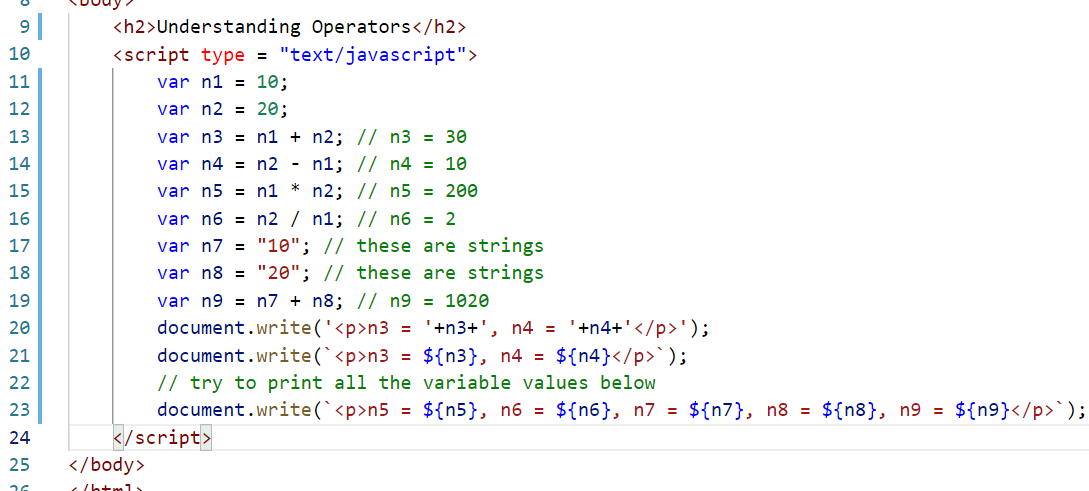
List of operators that are widely used:

+, -, \*, /, =, %

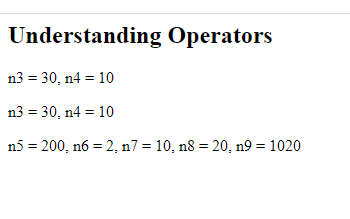
==, <=, >=, !=, ===,<, >

&&, ||

++, - - (two minus without space)



Output:



Comparison & Increment, Decrement operators

<, >, <=, >=, ==, ===, != are comparison operators

a = 20;

b = 30;

c = “20”;

d = 20

a == b; // it returns false, because a & b are not equal

a === b; // it returns false, because a & b are not equal

a == c; // it returns true, because a & c are same, but it doesn’t check types

a === c; // it returns false, because a & c types are different

a === d; // it returns true, because a & d value & types are same

a < b; // it returns true

a > b; // it returns false

var x = 20;

x++; // x will be incremented by 1 & it becomes 21

var y = 25;

y- -; // y will be decremented by 1 & it becomes 24

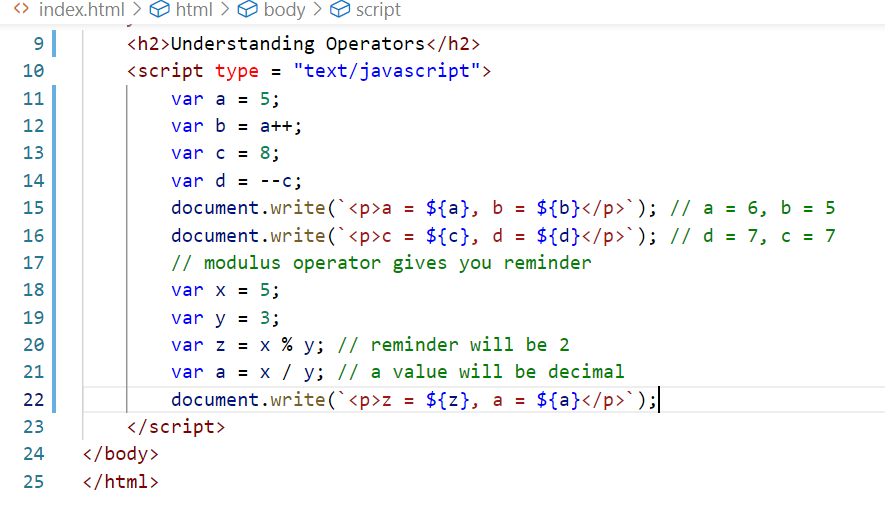
Above is post increment & decrement, but you also pre-increment & pre-decrement

var z1 = 30;

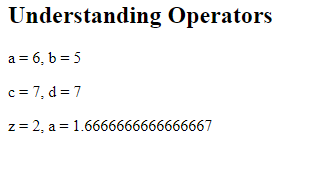
var z2 = z1++; // post increment, z2 = 30, z1 = 31

var z3 = 40;

var z4 = ++z3; // pre increment, z3 = 41, z4 = 41



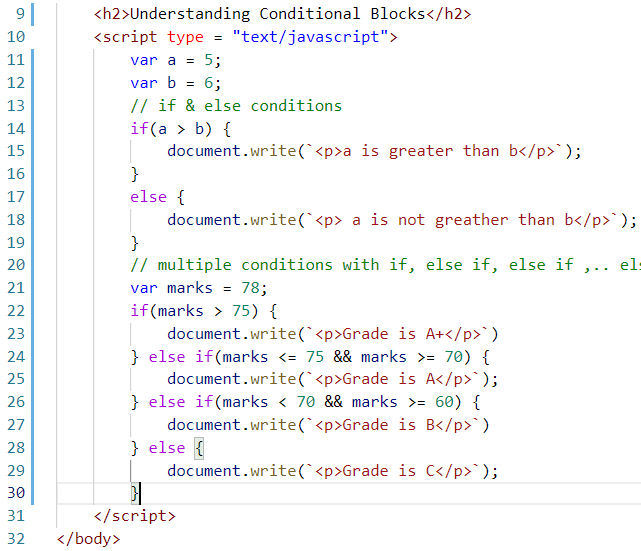
Output:



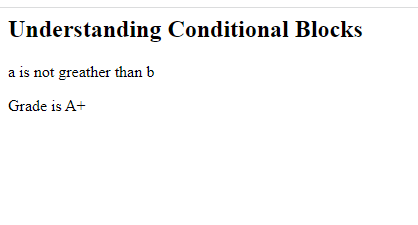
Conditional Statements

It makes set of statements to execute when certain condition is met, in Javascript below are conditional statements available

1. if
2. if & else
3. if else if else if … else
4. switch

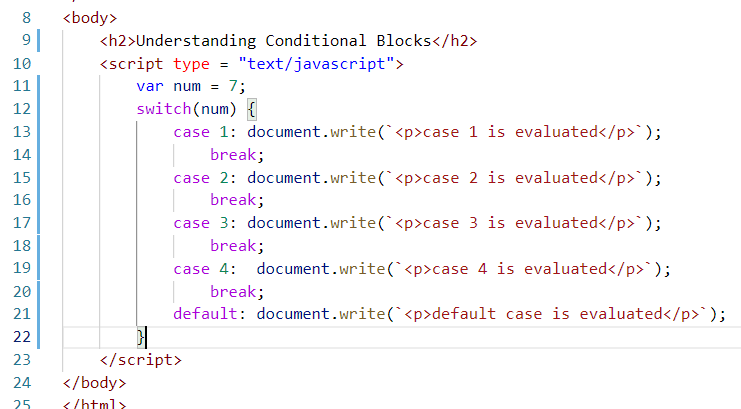


Output:

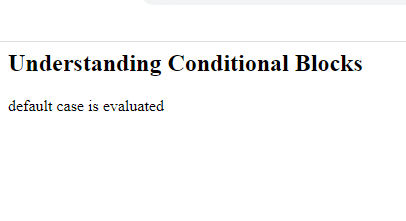


Switch statement

It is also similar to the if else if else if .. else statement, but it evaluates the conditions through case values



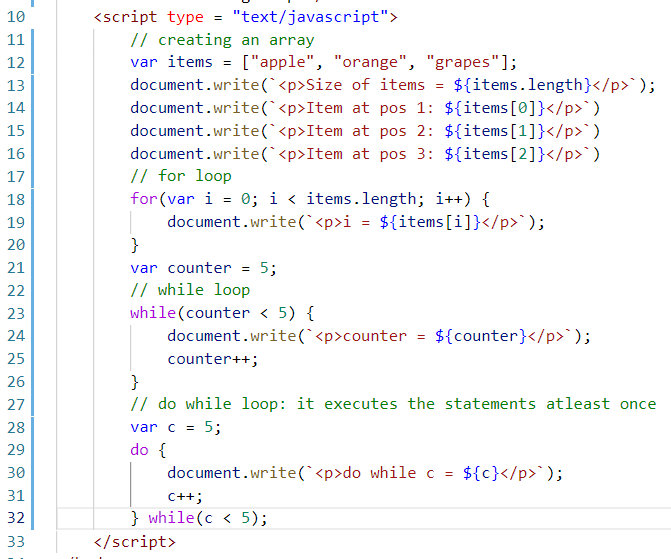
Output:



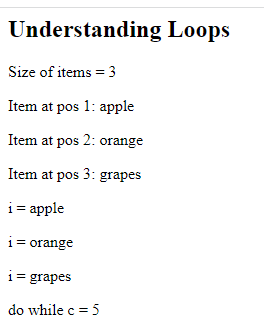
Loops:

They are used to run some statements repeatedly until the condition becomes false, there are 3 types of loops

1. for loop
2. while loop
3. do while loop



Output:

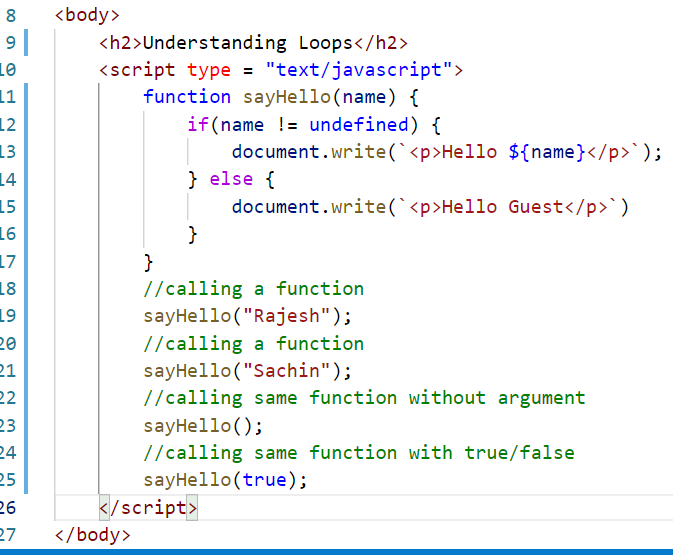


Functions:

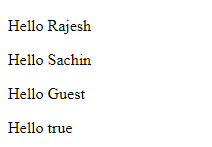
These are named blocks that can be reused whenever required, these will have logics that can be executed by calling through its name

Syntax:

function function\_name(arguments) {   
 // logics  
}

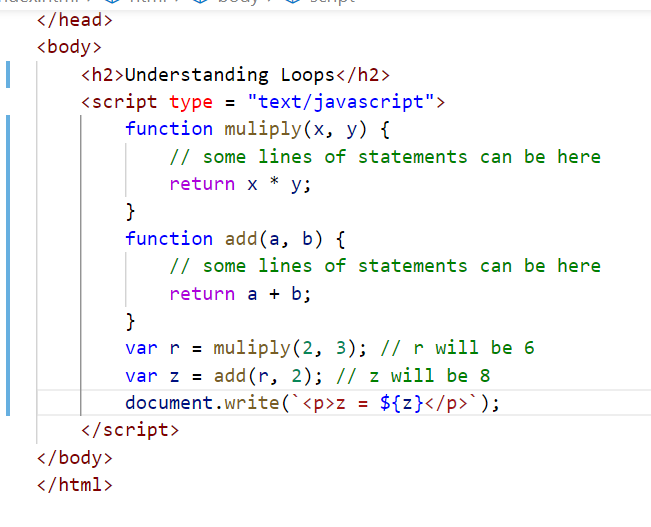


Output:

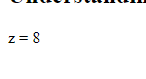


Functions with return types

Every function will not always print values, sometimes they want to process the data and give that processed data back to the caller, so the caller can use that data for any other operations



Output:



Summary

Variables - store data

Operators

Conditional Statements - if, if - else, if - else if .. else, switch

Loop - for, while, do while

Arrays

Functions

19-11-2022 Agenda (Session 11)

Anonymous Functions

Objects & Constructor function

Arrays in detail

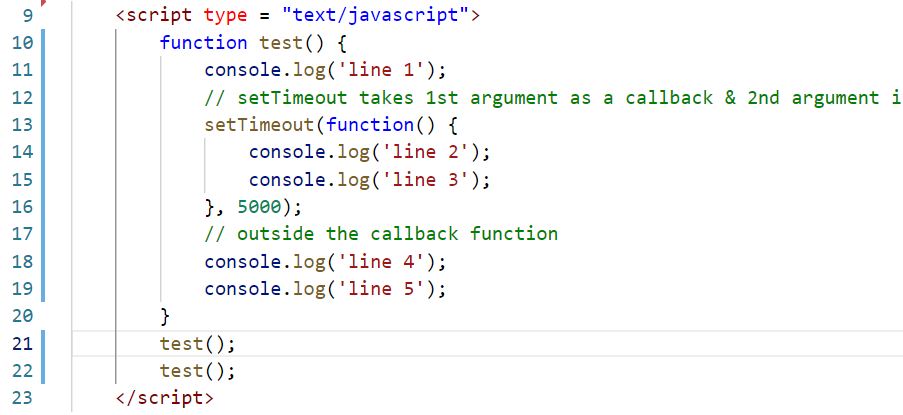
Inbuilt objects & Functions like Math, String, Date

New Features of Javascript

Anonymous functions & Callback:

These are functions which doesn’t have names, used mainly as callback functions, these are executed later based on some events probably like when timer is over, http response is available, request needs to be processed.

Callbacks are not executed as soon as they are initiated, they are executed later, these callbacks are treated as asynchronous functions as they are not executed in sequence



Here setTimeout is specifying after what time the callbacks must be executed, however in real time we must have callbacks that are executed on certain scenarios like when response is ready we want some callbacks to execute or when request is sent we want some callbacks to execute

Objects:

It is a structure that represents an entity, which will have multiple properties & values and functions as well, functions can be also part of object so that only the specific object can access the function

emp = { id: 123, name: “Raj”, salary: 35000 }

emp.id // returns 123

emp.name // returns Raj

emp.salary // returns 35000

Array of objects: These are collection of objects in one container

[“Apple”, “Mango”, “Grapes”] // this is a simple array

[ { id: 1234, name: “Raj”, salary:45000 }, { id: 8823, name: “Viji”, salary:40000 } ] // this is a complex array

Objects with functions

emp = {id: 1234, name: “Raj”, display: function() { ….

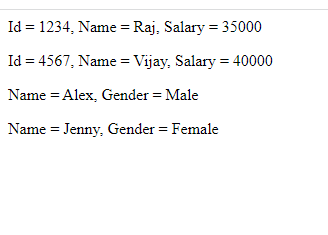
} }

function disp() { } // this can be executed directly as disp();

emp.display()



Output



Arrays:

It is a special type of datastructure which can store multiple elements, it has many inbuilt functions like push(), splice(), pop(), forEach(), map()



Output:



Some important functions in String, Date & Math

String has inbuilt functions like

toUpperCase(), toLowerCase(), length(), concat(), charAt(), indexOf()

var x = “hello\_everyone”

x.charAt(4) >> returns o

x.length() >> 14

x.toUpperCase() >> returns HELLO\_EVERYONE

// strings are immutable, the contents are not modified

x.charAt(0) >> returns h

x.concat(“\_good\_morning”) >> returns hello\_everyone\_good\_morning

Math has inbuilt functions like

pow(), random(), ceil(), floor(), abs(), min(), max() and so on

Math.pow(3, 2) >> 3 power 2 returns 9

Math.random() >> returns some random decimal number, 0 to 1 like 0.9923983492

Math.ceil(10.5) >> returns 11

Math.floor(10.5) >> returns 10

Math.min(10, 6) >> returns 6

Math.max(10, 6) >> returns 10

Math.abs(-10) >> returns 10

Math.abs(10) >> returns 10

Date inbuilt functions are like

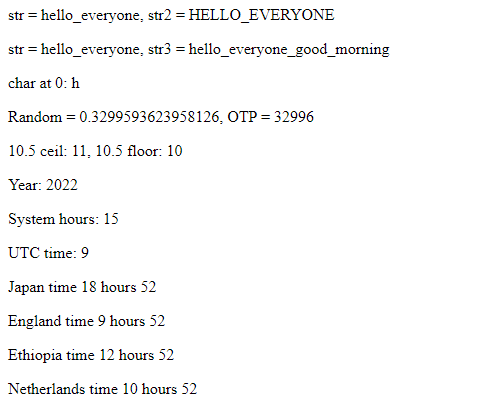
var d = new Date(); // to create

d.getYear() // returns year like 2022

d.getDate() // returns the date like 19

d.getMonth() // returns the month, but it is not advisable to use this, because month starts from 0, instead of 1, it means for November you get 10 instead of 11



Output  


ECMA Script is a specification which specifies what all the feature Javascript must support, ECMAScript is also termed as ES which has many releases with version numbers like 5, 6, 7 and so on.

ES6 has many new features specified for Javascript that simplifies writing Javascript syntax

1. New keywords like let, const to declare block scoped variables
2. New keywords like class, extends, super, constructor to support many object oriented features easily
3. Arrow functions to simplify the writing of anonymous functions
4. Template Strings to create string concatenation easily (i.e., back tick quote)
5. Object Destructuring
6. Array Destructuring
7. Rest & Spread Operator
8. Optional Chain
9. Exponential operator

let & const:

They create block scoped variables

var should be avoided because it is going to be global even if you declare within the block

if( … ) {   
 var x = 20;  
}

x is visible outside the if block also

if(…) {  
 let y = 30;

const z = 40;  
}

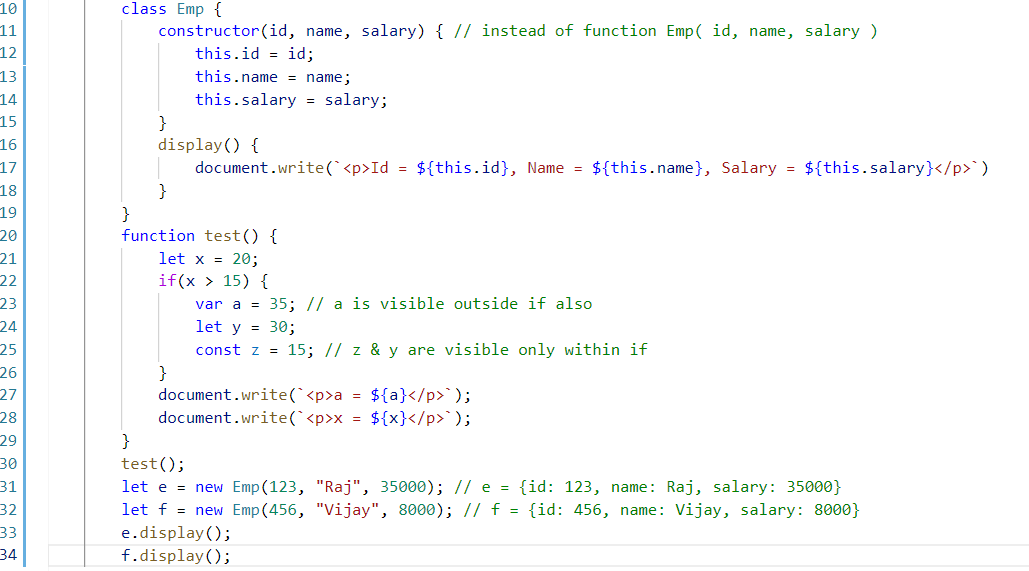
y & z both are visible within the if block, const variables are constants which can’t be modified, let variables can be modified

Creating classes

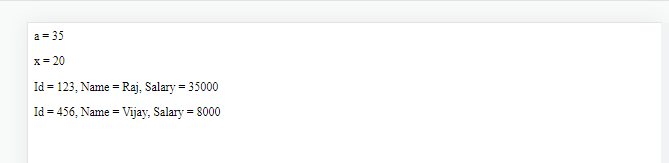
These help to simplify creating objects and their related functions easily

class Emp {   
 constructor(…) { } // constructor to Emp class

display() { … } // display function part of Emp class  
 getDepartmentCode() { … } // getDepartmentCode function part of Emp class  
}



Output:



20-11-2022 (Session 12)

* Javascript new features
* Event Handling & DOM Manipulation

New features of Javascript

Classes in Javascript

They group properties and functions of an object, so that you can access using the object, you can also inherit a class from another class using extends keyword.

Achieving Inheritance

You can use extends keyword to inherit a class from another class

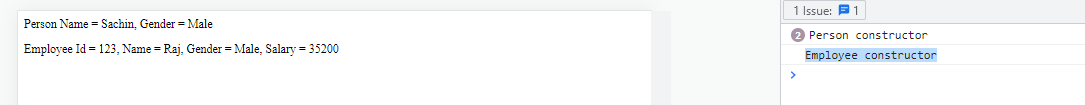
super keyword

It is used to access super class members from the sub-class, it helps to call super class constructor to initialize super class members that are inherited.

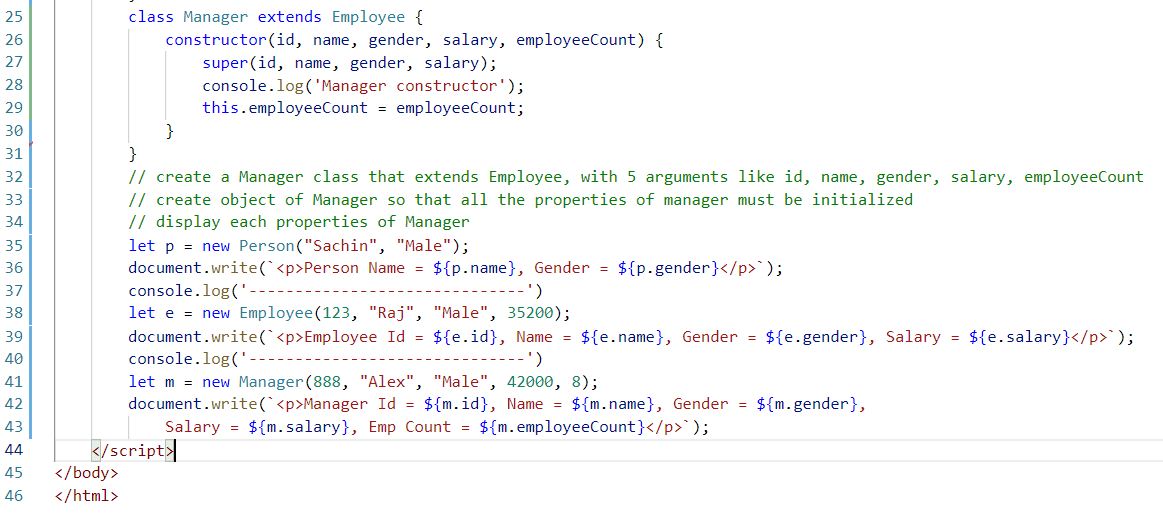
class A {   
 constructor(name, age) { // name & age initialization }   
}  
class B extends A {   
 constructor(id, name, age, salary) {   
 super(name, age);   
 // id & salary initialization  
 }   
}



Output:



Extending Employee by Manager



Template Strings;

These are strings which doesn’t need to concatenate with + to concatenate any expression, you can use back tick character

`<p>Name = ${name}</p>`

Arrow function

These will simplify writing callbacks & anonymous functions, they remove many unnecessary expressions from the functions

ex:

If anonymous class is written as

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

Then arrow function can be written as

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

If function need to have only one line then return & { } are optional

(a, b) => a + b; // returns a + b

Note: return may not be required in all the anonymous functions

function (a, b) {  
 let r = a + b;  
 console.log( r );  
}

Arrow can be written as

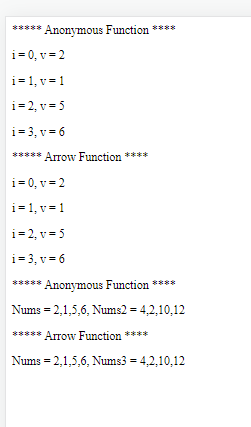
(a, b) => {   
 let r = a + b;  
 console.log( r );  
}

Simplified form can be

(a, b) => console.log(a + b); // this doesn’t return any expression, instead prints it



Output:



Destructuring arrays & objects

This feature helps us to easily access array elements & object properties, whenever you want to access array element or object properties & assign to another variable you have to create many variable declarations, destructuring helps to declare all the variables at once & assign value of array elements or object properties

Ex:

Array element assignment

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

let x = items[0];

let y = items[1];

let z = items[2]; and so on

Object property assignment

let emp = {id : 123, name : “Alex”, salary: 40000}

let id = emp.id;

let name = emp.name;

let salary = emp.salary

Array Destructuring: It helps to assign all the values of the array in a single declaration

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

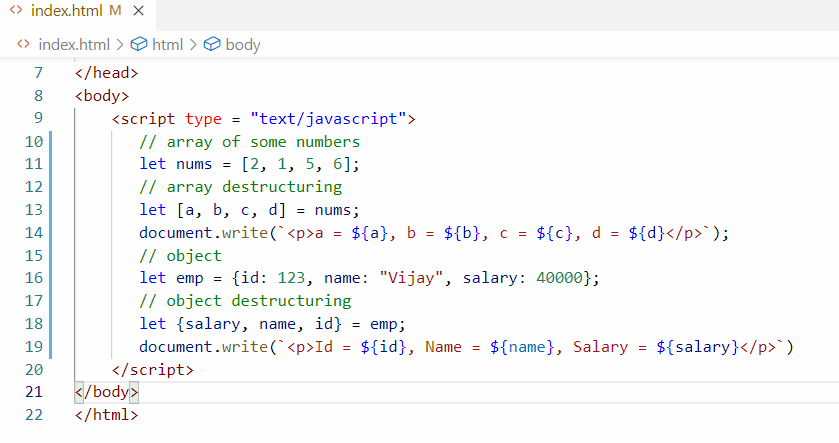
let [a, b, c, d, e] = items; // each element present in the array gets the value based on the position of the variable, means 1st variable gets value from 1st position, 2nd variable gets the value from 2nd position and so on

Object Destructuring: It helps to assign value of the object properties to the variables in single declaration, here the variables name must be same as object properties mandatorily.

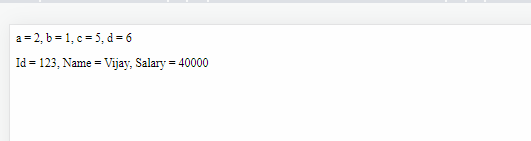
let emp = { id: 1234, name: “Alex”, salary:40000 };

You must create variables as id, name & salary, here position of the variables doesn’t matter unlike in array destructuring

let { name, id, salary } = emp; // here emp.id value is assigned to id, emp.name will be assigned to name, then emp.salary will be assigned to salary.



Output:



Rest & Spread operators

These will take care of calling the functions without losing the data

Rest operator is a parameter which can accept 0 or more arguments

function add(x, …y) {   
  
}

You can call add function by passing any number of values to y

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

add(); // x = undefined, y = []

add(2, 3); // x = 2, y = [3]

Suppose a function xyz(a, b) { } is called by passing 3 arguments like xyz(20, 30, 40) then 40 value will be lost

Spread operator is an argument that spreads value to multiple parameter of the function

function test(a, b, c) { }

let n = [10, 30, 20]

test(n); // a = [10, 30, 20], b = undefined, c = undefined

In the above function call, only a parameter accepts the entire array, we can spread value to each parameter using 3 dots while calling

test(…n); // a = 10, b = 30, c = 20

Suppose if n has more than 3 values, then other values will be lost

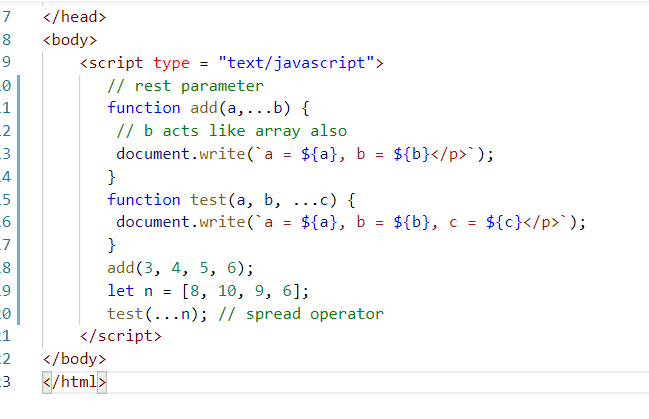
ie., if n = [10, 30, 20, 40, 50] then test(…n) will spread only to 3 parameters, which means 4th & 5th value will be lost, in that case you better use the last parameter as rest parameter in the test

i.e.,

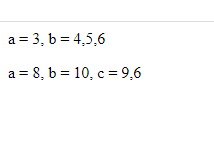
function test(a, b, …c) { }

so that if n = [10, 30, 40, 50, 20], then test(…n) will spread values to a, b & c where a will be 10, b = 30, c = [40, 50, 20]

Note: A function can have only one Rest operator & it must be at the end



Output:



Exponential operator

It is an expression that can calculate powers of 2 numbers easily using \*\* (2 stars)

Instead of Math.pow(3, 2) you can use 3 \*\* 2

Optional Chain

It is used to access the property which might be optional some times, this avoids lot of errors in the code

users = [ { name: “Alex”, gender: “Male”, address: {state:”ABC”, city: “XYZ”} },   
 { name: “David”, gender: “Male”}  
{name: “Jenny”, gender: “Male”, address: {state:”1234”, city: “HHH”}} ]

Suppose you are iterating users and accessing state & city on each iterated item, you may need to write conditions if some properties are not present

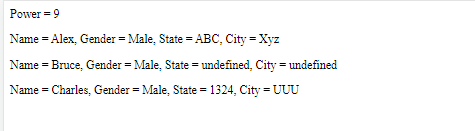
if(user.address != undefined) {   
 user.address.state  
}

There you may need to use too many conditions to check for property existence, when there many nested objects, to simplify these optional chain has been introduced, its used to check if property present or not as below:-

user?.address.state // access address only if its not undefined



Output:



Summary of ES 6 and later Features

* let, const, class, extends, super
* Arrow functions
* Template String literals
* Rest & Spread
* Exponential operator
* Optional Chain

26-11-2022 (Session 13)

Agenda

Some other features of EcmaScript

* Default arguments
* Static keyword
* String methods like padStart & padEnd

DOM Manipulation & Event Handling

* document object methods
* event object
* Different types of events generated by elements

Default argument

It is an argument that takes a default value if in case user doesn’t pass value to the function, it avoids type-errors when function expects parameter but user doesn’t pass

Earlier to avoid the errors developers used to write conditions as below

function add(x, y) {   
 // giving default values only if x & y are undefined

if(x == undefined) { x = 0; } else { x = x; } // or x = (x == undefined) ? 0 : x;  
 if(y == undefined) { y = 0; } else { y = y; }  
 result = x + y;  
   
}

add(20, 30); // here x = 20, y = 30;  
add(5); // here x = 5, y = 0

In latest feature ES, it specified easiest way of assigning default values

function add(x = 0, y = 0) {   
 result = x + y;  
}

add(); // x = 0, y = 0;

add(5); // x = 5, y = 0;

add(6, 3); // x = 6, y = 3

Static:

These are properties or functions part of a class and must be accessed through the class names, these are useful when you want some properties to be accessed without creating object

class Customer {   
 static bankName = “ABC Bank”;  
 …  
static test() { … }   
  
 display() { … }   
}  
Customer.bankName // returns ABC Bank  
Customer.test(); // test might be accessing static properites  
  
let c = new Customer(….);   
c.display(); // display might be accessing object properties



padStart & padEnd

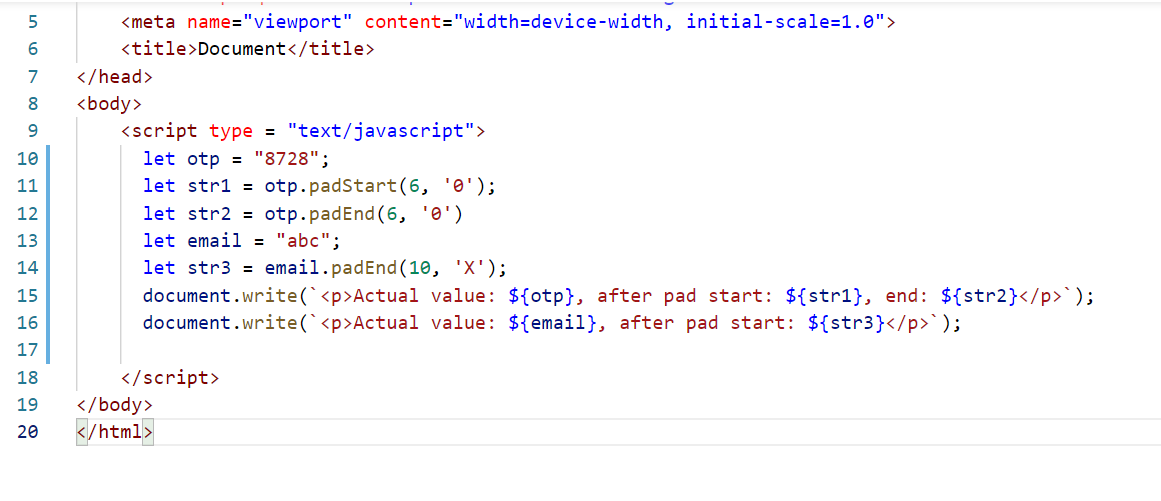
These are the functions used on strings to add some extra strings in the beginning or end of the string

hello >> 000hello >> padStart

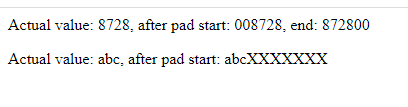
hello >> helloXXX >> paddEnd

Application >> userXXXX@gmail.com >> 9877XXXXXX

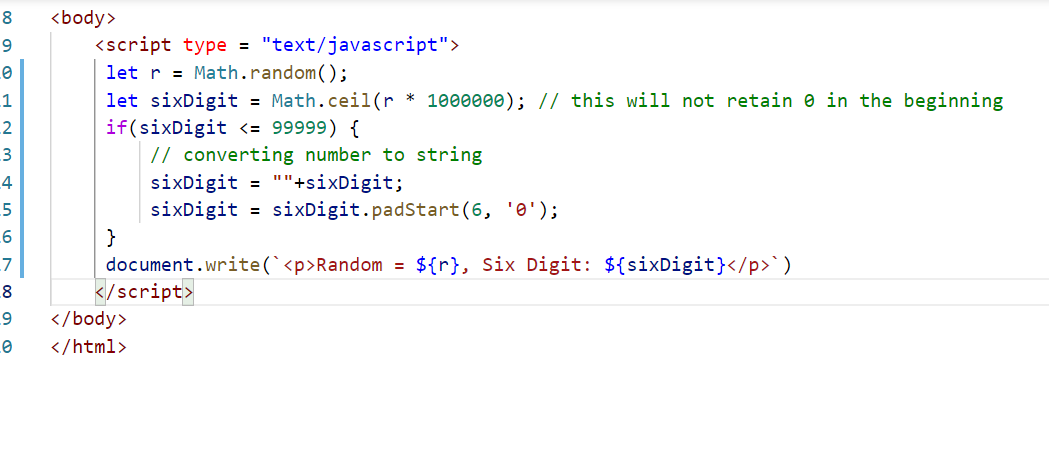
OTP >> 4589 >> 004589



Output:



Generating random number that will always display 6 digits



DOM Manipulation and Event handling

DOM stands for Document Object Model, which is created inside the browser when the HTML elements are loaded, using Javascript document object we can access any HTML element and manipulate it like adding element, removing element, adding CSS or removing CSS and so on.

Event handling: Events are the things generated by the elements when user perform some action, you can call some functions when certain events are generated which is called as event handling

HTML elements generate events like onmouseover, onmouseout, onclick, onsubmit, ondblclik, onkeyup, onkeydown and so on

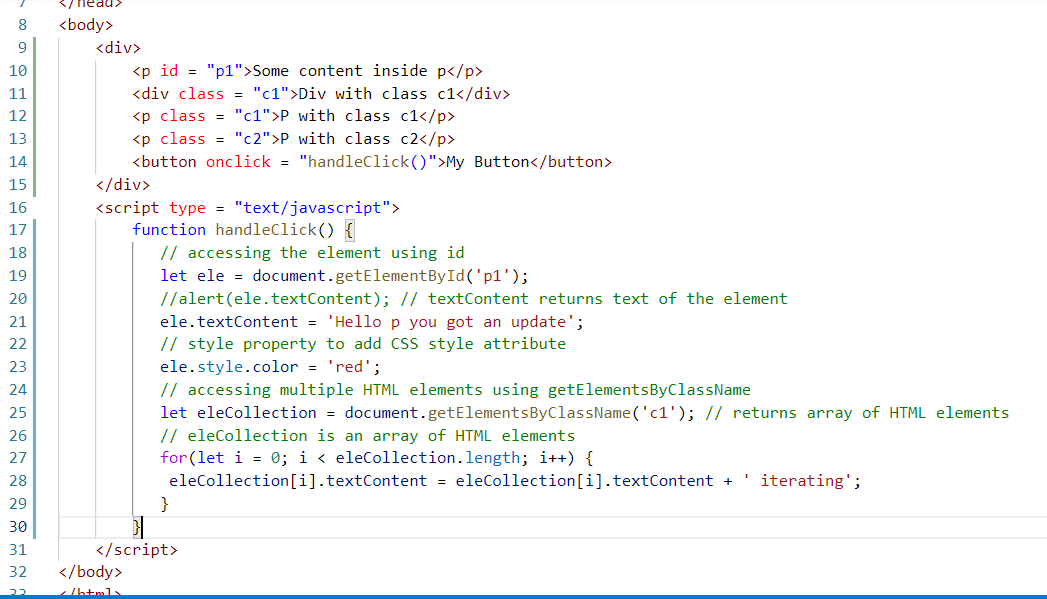
Accessing the elements

we have many methods in document object to access the element like

document.getElementById(“id”): Returns the HTML element having the id

document.getElementsByTagName(“tagName”): Returns all the HTML elements of that tag name, it returns array of HTML elements

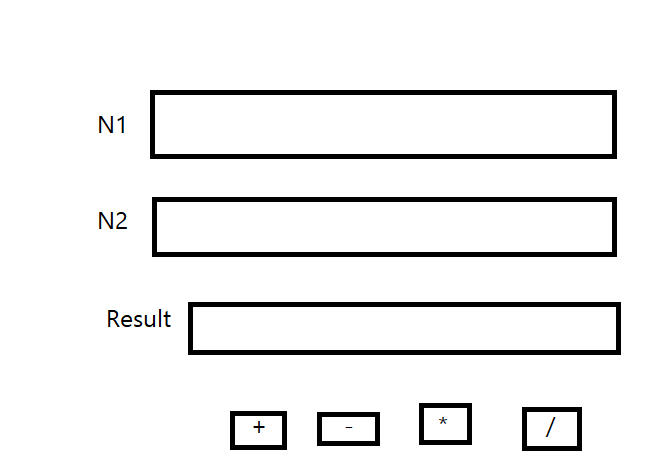
document.getElementsByClassName(“className”): Returns all the HTML elements matching to the class name, it returns array of HTML elements



Output:



Activity:



Enter value in input box N1 & N2, then upon clicking on + or - or \* or /, Result input box must show the result

Ex: If you enter 10 in N1 & 20 in N2, then Result must be 30 if you click +, -10, if its -, 0.5 if its /, 200 if its \*

Solution for add



Output

Session 14 (27-11-2022)

Agenda

Different types events in Javascript

DOM manipulation using document & event objects

Form Validation

Types of events

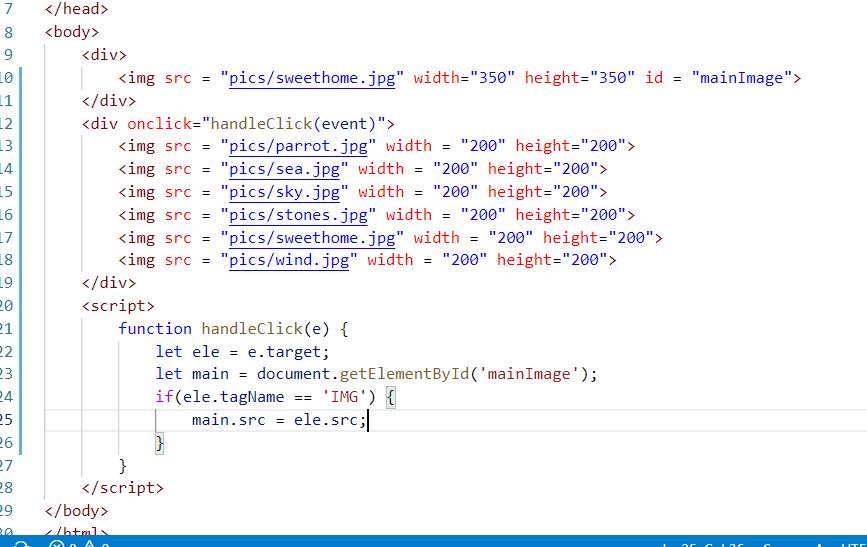
There are many events like

* onclick
* ondblclick
* onmouseover
* onmouseout
* onkeyup
* onkeydown
* onload
* onfocus
* onblur
* onsubmit

Accessing the element through events

Sometimes you want to know which particular element generated the event then you can use event property to identify the element

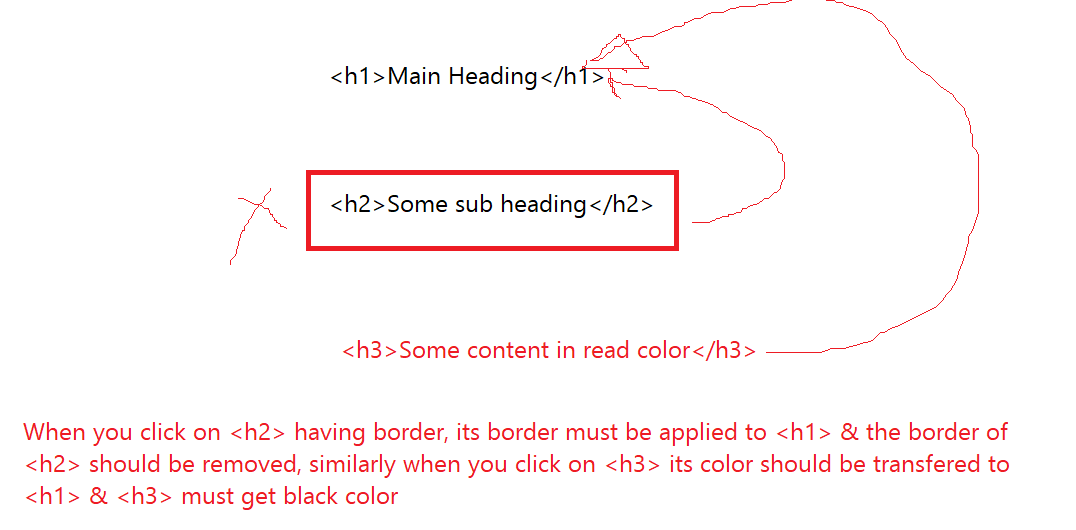
Using value of one element that generated the event and adding to another element



output:



Activity:



Summary:

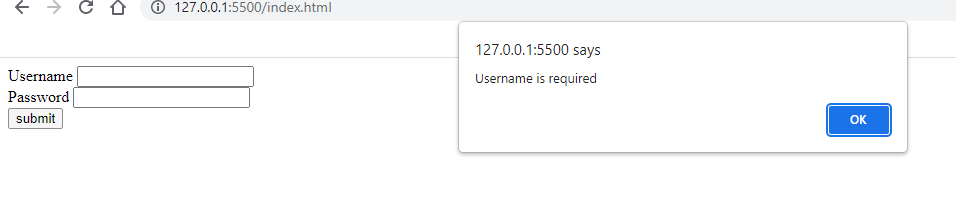
* Accessing DOM using document object methods like getElementsByTagName, getElementsByClassName, getElementById,
* Accessing DOM using event object property ‘target’, i.e., event.target

Form Validations:

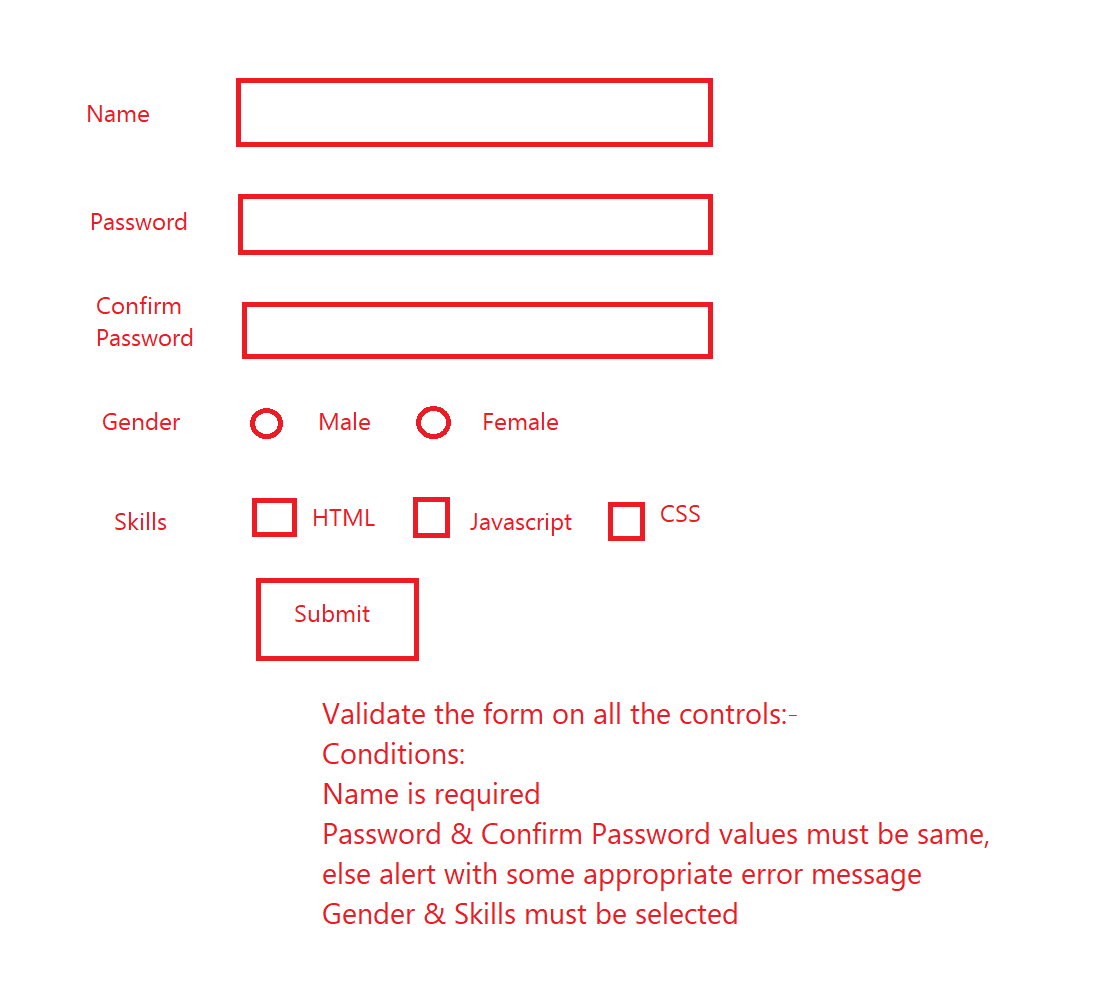
Javascript can do a client side validations on the forms so that if the input is invalid it can prevent user submitting invalid input to the server



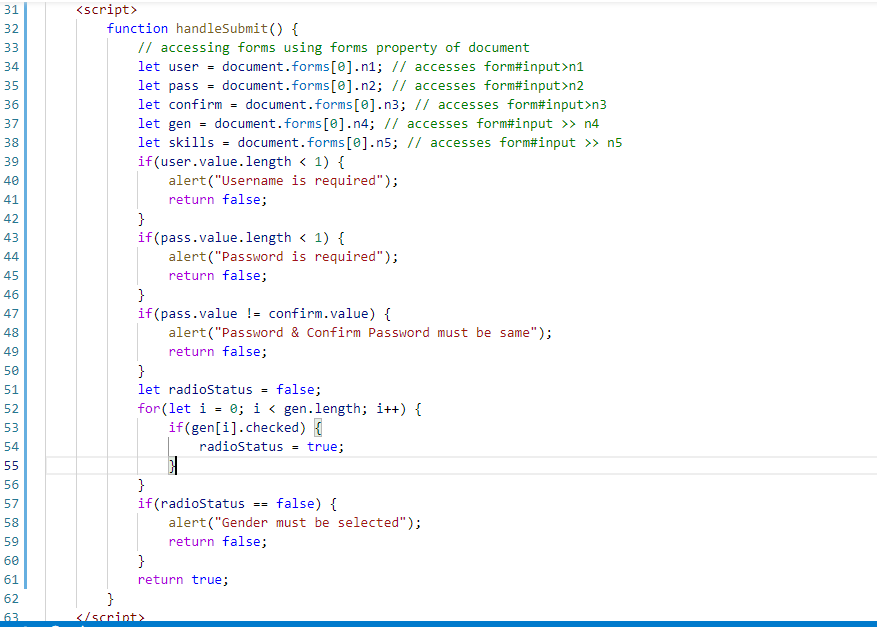
Output:



Activity

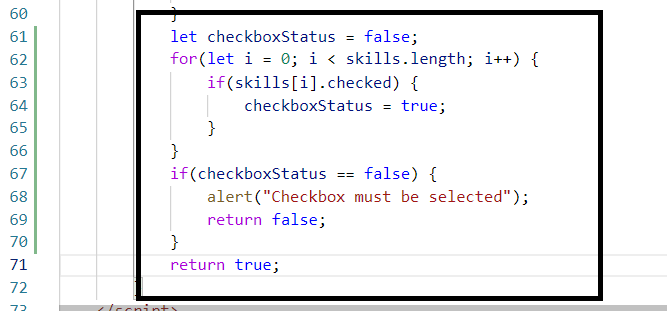


Solution:



Note: Validate checkbox in the above code

Solution to validate checkbox



Session 15 (03-12-2022)

Agenda:

* Understanding the XMLHttpRequest
* JSON importance
* fetch method
* Promises

XMLHttpRequest

It is an object in the Javascript that can perform asynchronous operation by sending request to the backend service and obtain the response data from the server

It has few properties & functions which are useful in dealing with request & response

Functions

open(http, url): This is used to initiate the request, 1st argument is HTTP method like GET, POST, PUT or DELETE, 2nd argument is the Url of the backend resource  
send(): This is used to make a request to the resource

Properties

readyState: This specifies the state of the request in terms 1, 2, 3, & 4

* 1: Request is initialized
* 2: Request is sent
* 3: Request is processing
* 4: Response is available

onreadystatechange: It is an event handler that accepts a callback, this would generate the event each time the readyState changes its value from 1 to 4

responseText: This property returns the response data send by the server, usually it will be JSON data.

responseXML: This property is used when the data is in XML format

let xhr = new XMLHttpRequest();

xhr.open(“get”, “http://api.com”); // initiate the request

xhr.send(); // send request

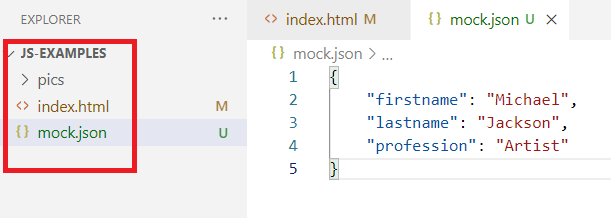
// event handler

xhr.onreadystatechange = function() {   
 // readyState is 4 then responseText will have data

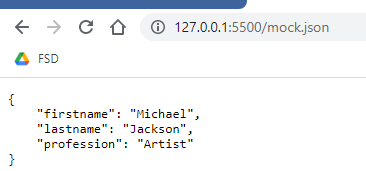
if(xhr.readyState == 4) {   
 let jsonData = xhr.responseText;  
 // convert JSON to JS  
 let jsData = JSON.parse(jsonData); // converts JSON to Javascript object  
 [ or ]

let jsData = JSON.parse(xhr.responseText)   
 }  
}

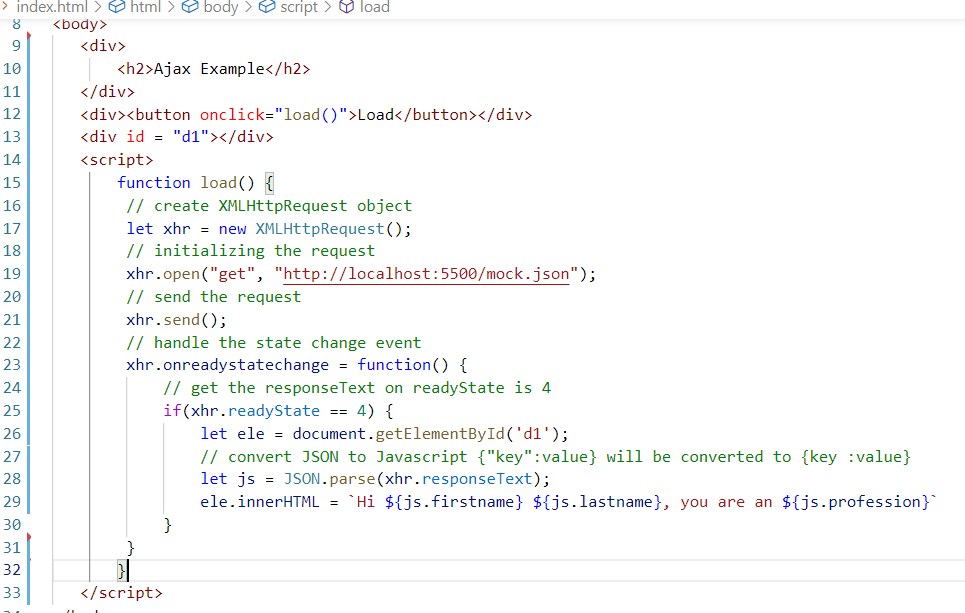
Note: Since we don’t have any backend service ready, we can create a fake JSON file and treat that itself as a real JSON data coming from the server



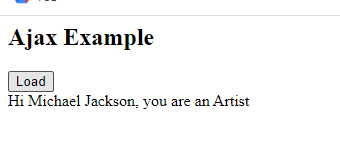
Enter the url of mock.json in the browser so that you must see these data in browser



Note: There are typically 4 HTTP methods we use while calling any backend service like get, post, put, delete. If we are able to see the data in browser then the HTTP method must be get



Output:



How to work with complex JSON data structures like an array

JSON Array data are wrapped in square brackets [ ], when you parse this JSON array you get Javascript array, you can iterate using for loop to access each data in the array.

mock.json

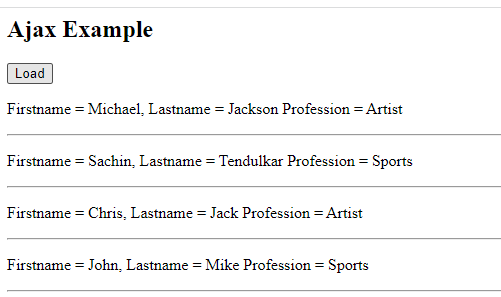


We can get this complex data and iterate in Javascript

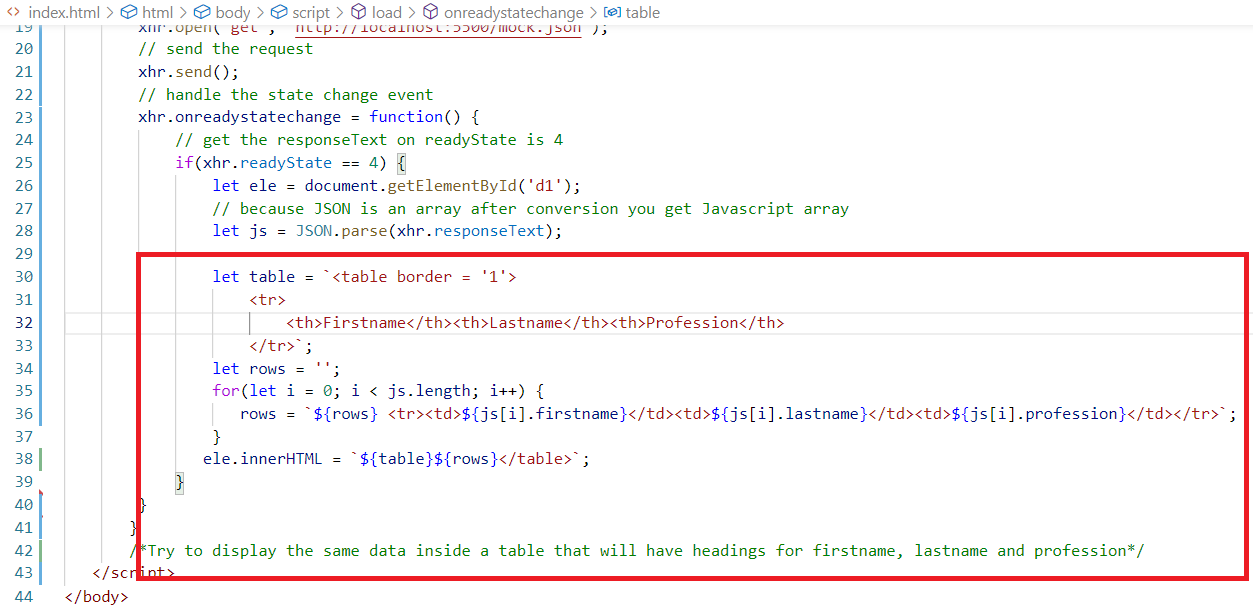
Change the load function as below

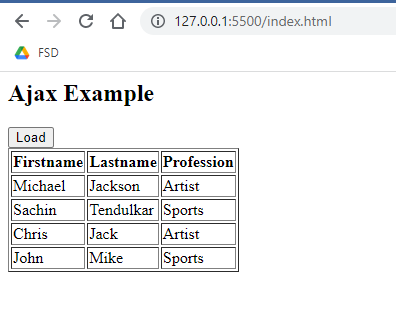


Output:



Same program that displays the items in table



Output  


Promises:

It is an object that performs an asynchronous operation which can be successful or failure, it gives you two functions .then(callback) & .catch(callback), these 2 functions are called when promise is resolved(successful) or rejected(failure)

Fetch:

It is a replacement to XMLHttpRequest to access resources at the backend, fetch is much simpler compare to XMLHttpRequest, fetch returns promise once you access the resource, this promise can be successful or failure

let p = fetch(url)

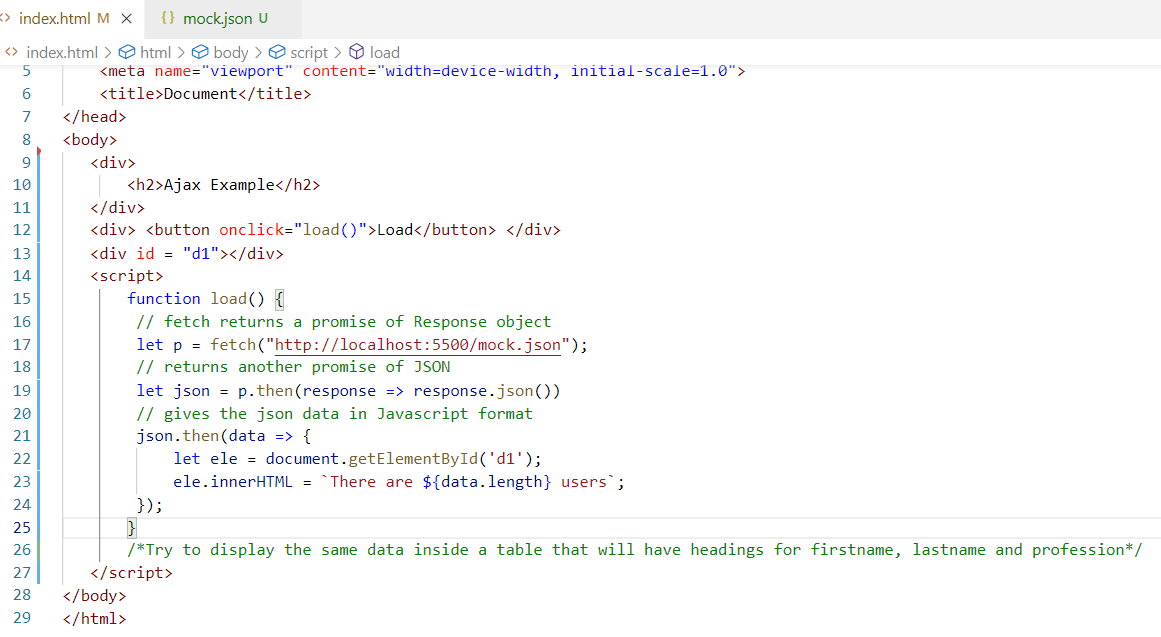
fetch will access the backend resource & returns a promise which can be resolved/rejected, you don’t have to use any onreadystatechage or readyState or responseText properties to get the data

fetch returns Promise on that you can call .then( callback ) or .catch( callback ) to get the successful or failed response

let p = fetch(url)

p.then(function(data) { .. }).catch(function(data) { });

Here the data is the response data of the promise



Note: XHR is a older approach, you can use fetch & promise which are latest approach

Note: In Promise you directly get a Javascript object after calling response.json(), which means you don’t have to use JSON.parse also

Activity:

Using fetch show the same data in a table structure

Session 16 (04-12-2022) Agenda

React.js concepts

React.js

It is a Javascript library which is used to create single page application, it uses components to create single page application.

Components: These are reusable code that creates UI (User Interface), it can be nested in other components

React.js uses HTML & JSX to create web applications

HTML: It is to display the content

JSX: It is an extended form of Javascript which simplifies writing HTML in Javascript

Note: React can also use CSS for styling the components

Writing HTML in JS

let content = ‘<h1>Some content for heading</h1>’

Writing HTML in JSX

let content = <h1>Some content for heading</h1>

Writing expression in Javascript

let name = “Alex”;

let content = `<h2>Hello ${name}</h2>`;

Writing expression in JSX

let name = “Alex”;

let content = <h2>Hello {name}</h2>

React.js uses 2 main libraries to develop applications

1. React library: It gives functionalities to create components
2. React DOM library: It renders the created components from React library on the Browser DOM

Babel: It is a library which helps to convert JSX to Javascript code, we need babel because browser understands only Javascript

Totally we need 3 libraries

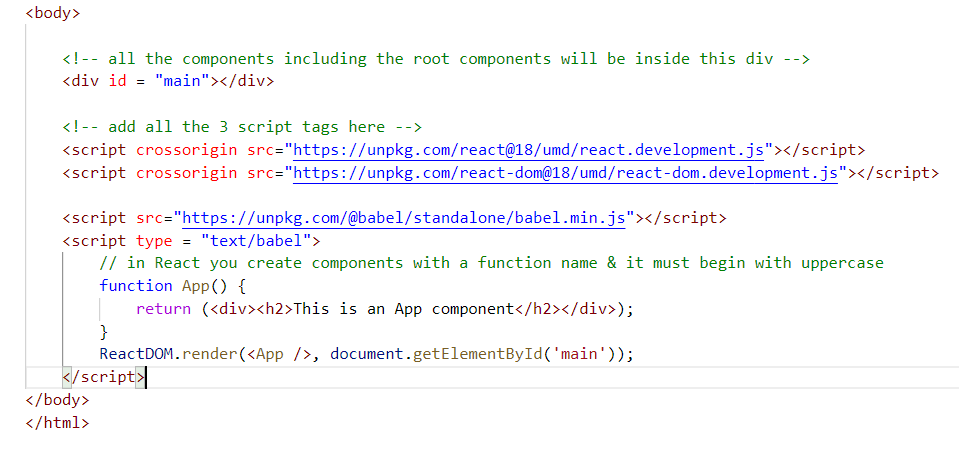
1. React library
2. React DOM
3. Babel library

There are 2 ways to get these libraries

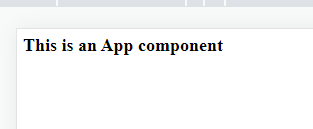
1. Accessing the online links
2. Downloading them using npm command (will be discussed later)

Root component:

It is a wrapper for all the components, every component must be inside this root component so that when you add this root component to the browser it will have all the components in it



Output:



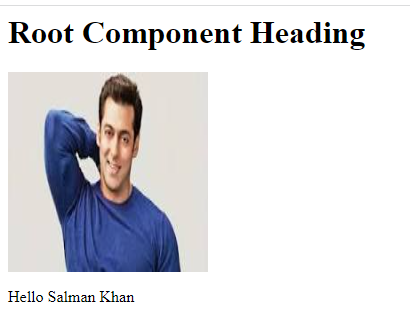
How to nest a component inside another component



Displaying the data using JSX



Output:



Using function in JSX



Virtual DOM

It is used by react to update the real DOM, it accelerates the performance at the front-end by only updating the part instead of reloading the entire DOM tree, Javascript will directly update the Real DOM and it refreshes the entire DOM tree even for minor updates, whereas in React the Virtual DOM will compare the content with Real DOM and only updates the node that need the changes.

Updating the DOM In Javascript

It updates entire DOM tree in the background

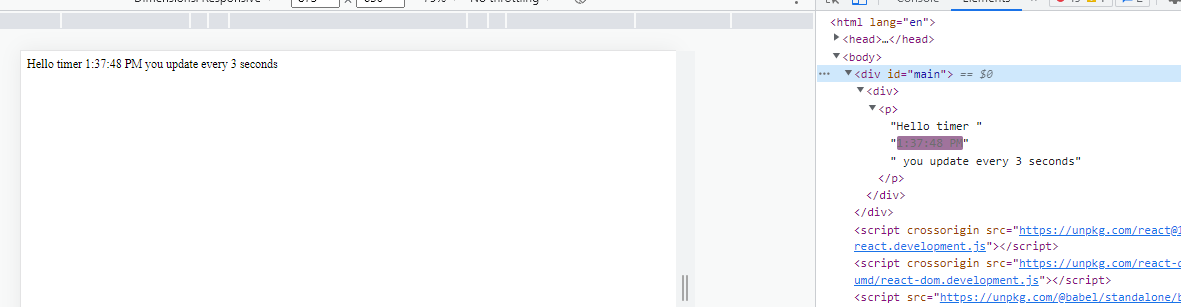
Updating the DOM in React

It updates only the node that needs to be modified and other nodes wouldn’t be reloaded in the background

ex:



Output:



Component props

props are used when a component wants to share the data to other components, any component that wants to receive the data must use props as an argument in the function

function Abc(props) { … }

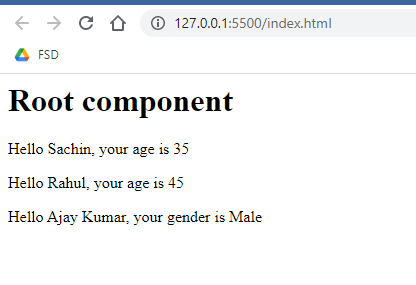
A component can share the data as below

<Abc title = “React course” user = {name:”Raj”, age:35} />

Here Abc component receives a props with title & user

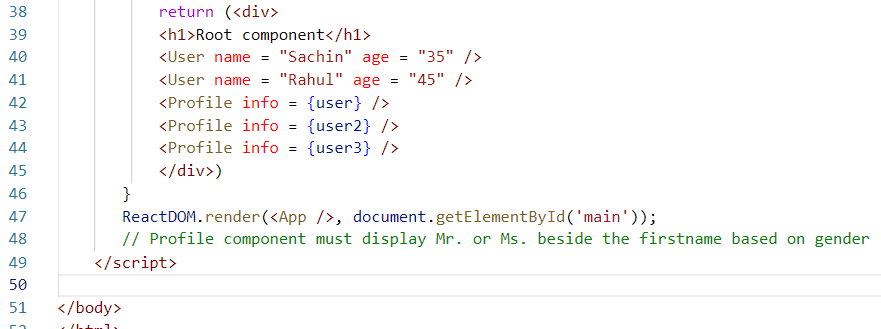


Output:

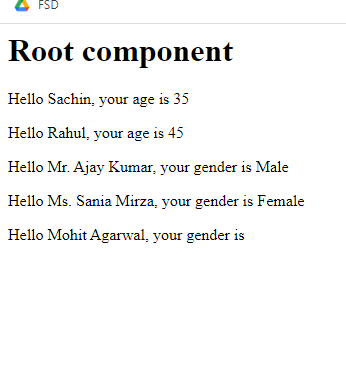


Conditional Rendering

It is used to display the content of the component based on the conditions, you can use if, else blocks to return the contents.

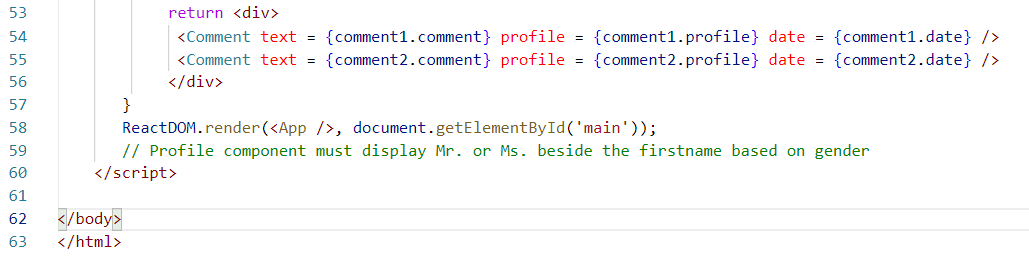


Output:

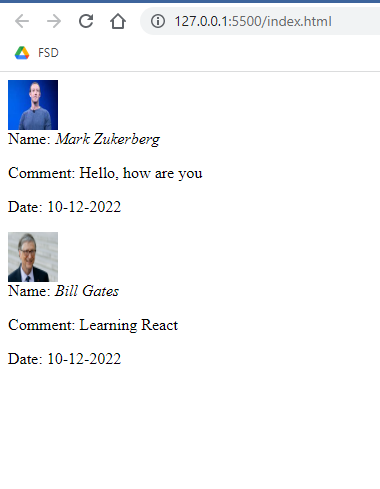


Compose components

You can create smaller components and compose them to reuse in multiple components



Output:



Session 18 (11-12-2022) Agenda

Class components

React Toolkit

Event Handling

States

Different ways of creating components

In React we can create components in two ways

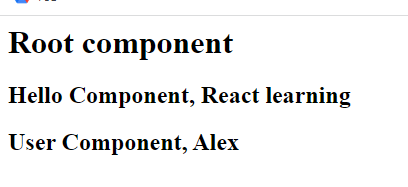
1. Function components: Which we have already discussed
2. Class components: Created by extending React.Component class

class App extends React.Component {   
 render() {   
 return <content>  
 }  
}

Note: Class based approach is more modularized as you can group the similar functionalities of the class in a single place



Output:

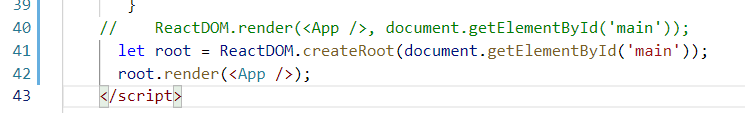


createRoot:

It is a new way of accessing the HTML element that is used to render all the components, from React18 onwards we should stop using ReactDOM.render() instead we must use createRoot and render the components

let root = ReactDOM.createRoot(document.getElementById(‘main’));

root.render(<App />);



Note: In Browser console you will not see the warning of switching to createRoot any more.

React Toolkit

It is used to quickly create react applications that will follow the best practices and the structure the industry follows, it provides following features

* Setup to develop react applications
* Live server to launch react applications
* Commands to run, build, test react applications
* Separates different types of files in different folders like HTML in one folder, JS files in another folders, all the libraries in separate folders
* It downloads all the necessary libraries to develop the react applications like React, ReactDOM, Babel and etc

How to get the React Tool kit

You can manually install the React Tool kit using NPM command and create the react application & another way is to create react application without manually installing react toolkit but you can access toolkit online while creating react application

Installing React toolkit

npm install create-react-app // install react tool kit

Creating the project from installed toolkit

create-react-app project-name // creates a new react project

[or]

Creating the project without installing the toolkit

npx create-react-app project-name

Note: npx or npm commands are available only if you install node.js in your machine



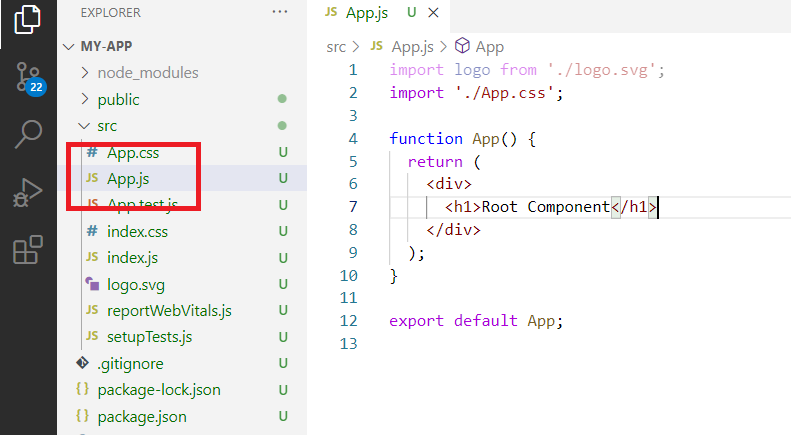
Command to launch the application

>> npm start

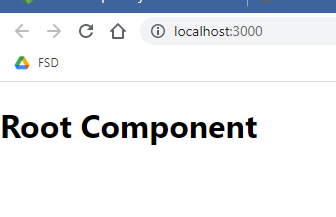
Note: Ensure you are inside the project folder in your command prompt

We need to create components in separate .js files and use them in App.js file which has a root component

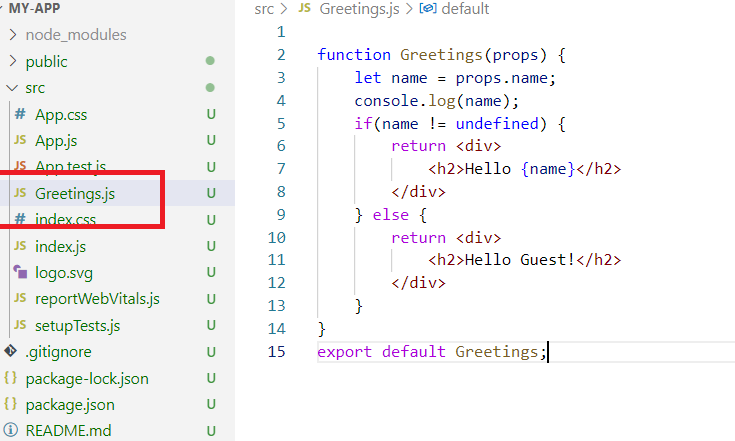
App.js



Output

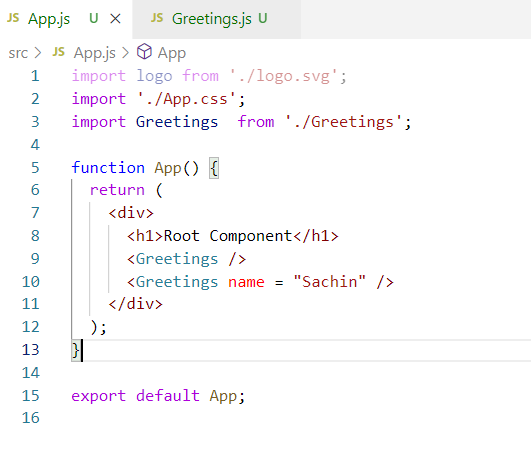


Greetings.js

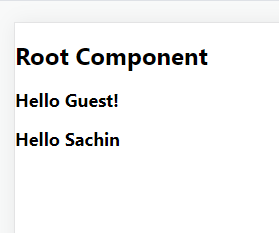


Add this Greetings component to the App.js

App.js

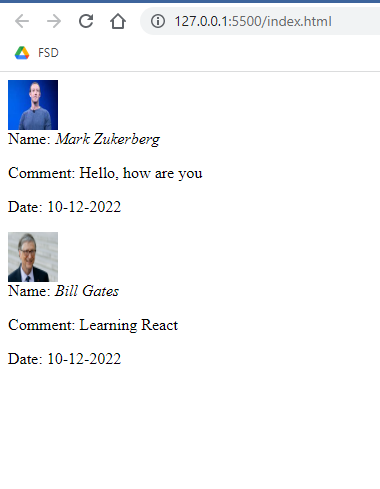


Output:



Activity:

Try the example which is already discussed in the new project, below is the output of the discussed example

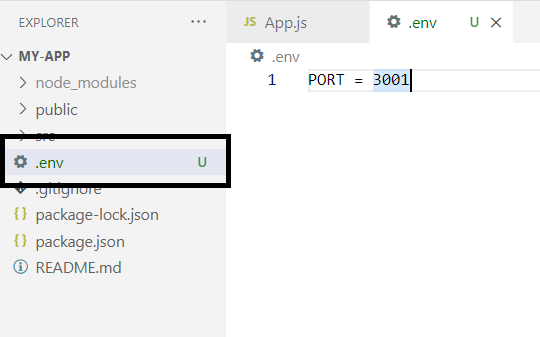


What happens when you use npm start

When you enter npm start, the npm looks package.json to identify which file needs to be loaded, from that it identifies index.js & index.html are the main file to load, index.js has a code to load root component to the element present in the index.html

How to change the port number of our application

By default the live server runs in 3000 port, but we can change it to any other ports if required, for that you need to create .env file in your project

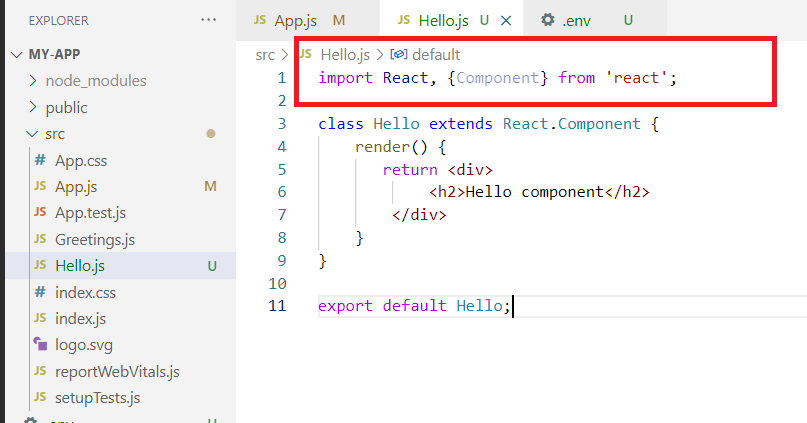


Now you can use npm start and check that application running in 3001 port.

Creating the components using classes

You must import some classes in your script

src/Hello.js



Notice that the class Hello has default export, which means it can be imported by any name i.e., import H from ‘./Hello’; [ or ] import Hello from ‘./Hello’;

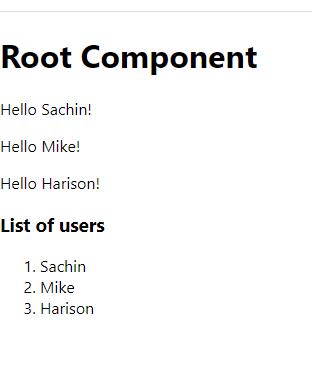
Lists and Keys

Whenever a component wants to render multiple elements present in the array we can use map function, these array are also treated as Lists, but in React whenever you are iterating over the list each iteration must be unique hence we must use key to specify each iteration is unique.

src/App.js



Output:



Applying CSS to the React components

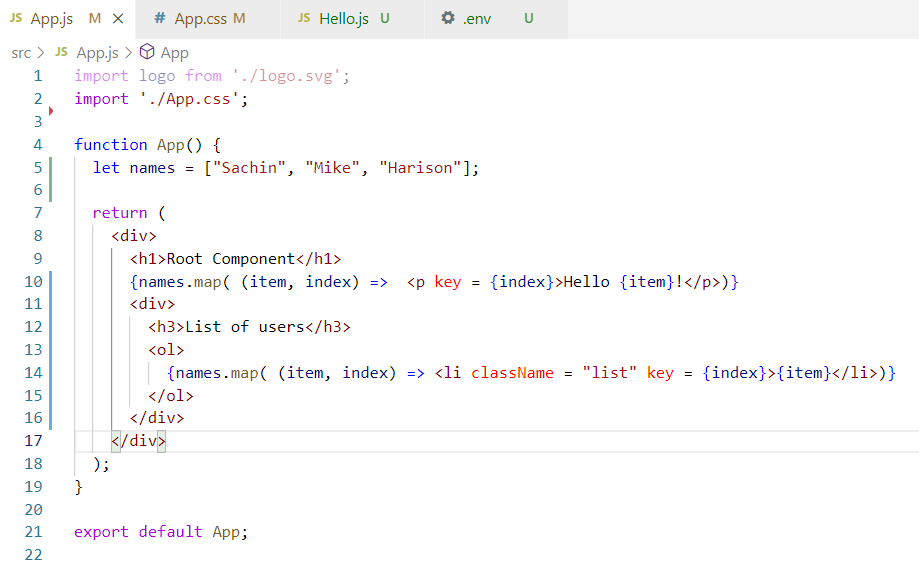
In most of the cases you will use class names of CSS in React components, however in React we must use className attribute to the HTML elements instead of using class attribute, as class is a keyword in React.

src/App.css

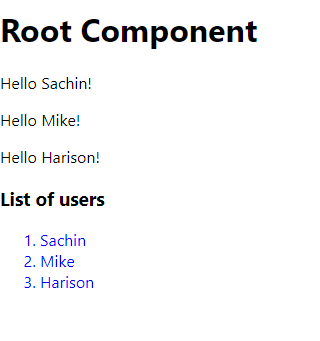


Now you must not use class attribute, you must use className to apply the CSS classes in your components

src/App.js



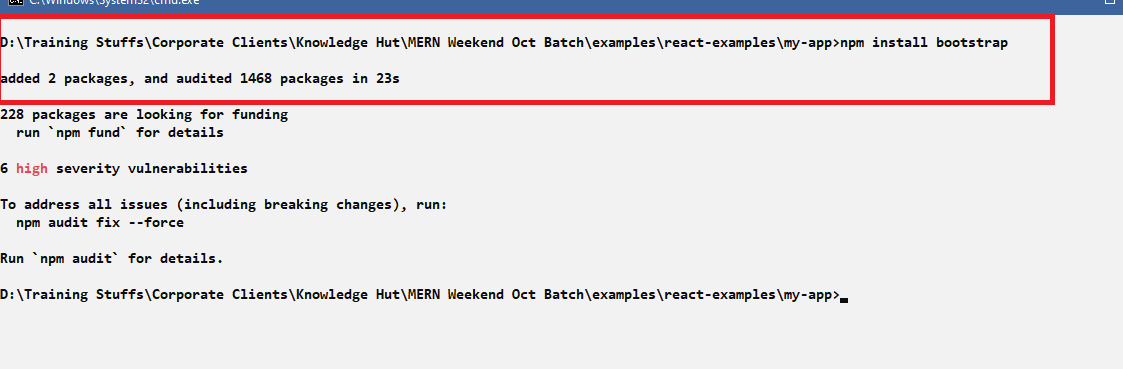
Output:



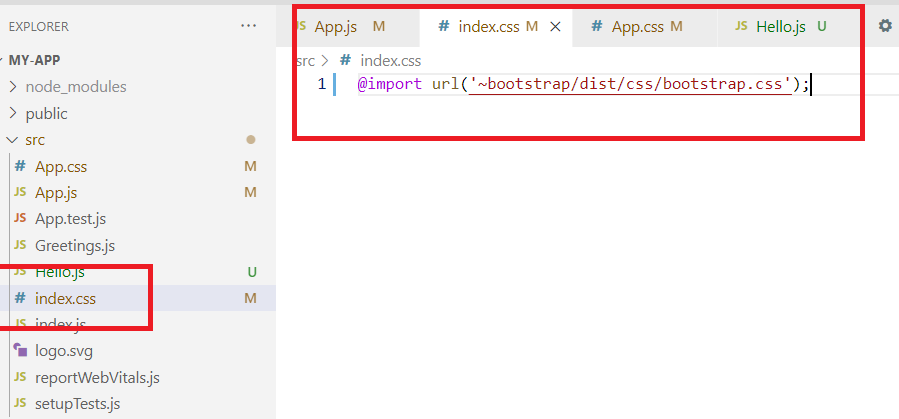
We can download bootstrap js libraries and use in our application

npm install bootstrap

Above command installs bootstrap library in node\_modules & we need to import the bootstrap css in our global css file of React application i.e., index.css



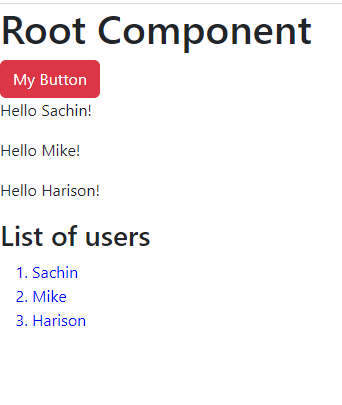
Importing bootstrap.css file in the index.css



src/App.js



Output:



Create an array of complex object and show the object properties in a table

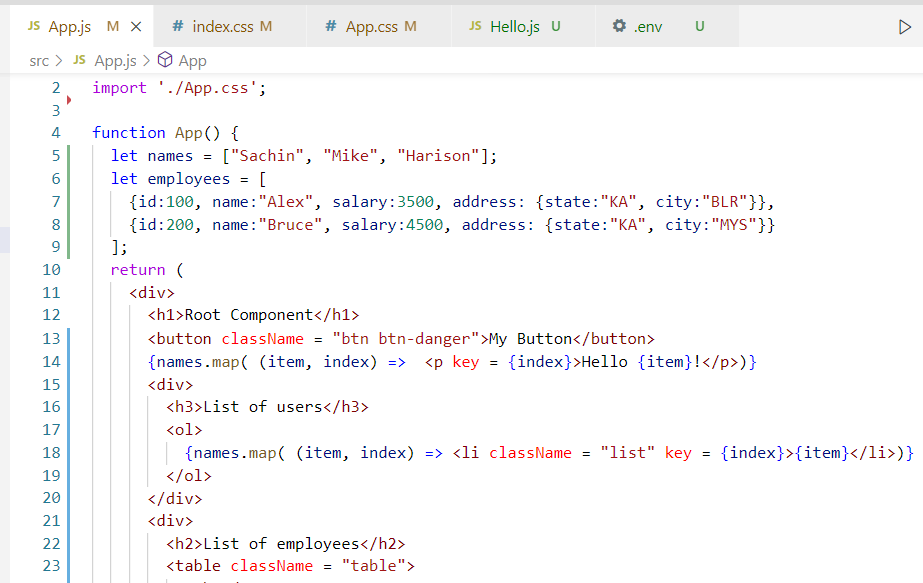
Display the below employee data in a table format using React list & key  
employees = [   
 {id: 100, name: “Alex”, salary:4500, address: {state:”KA”, city:”BLR”} },  
 {id: 200, name: “Bruce”, salary:2500, address: {state:”KA”, city:”MYS”} },  
 {id: 300, name: “David”, salary:6500, address: {state:”TN”, city:”CHN”} },   
]

Output must have a table with 5 columns as below

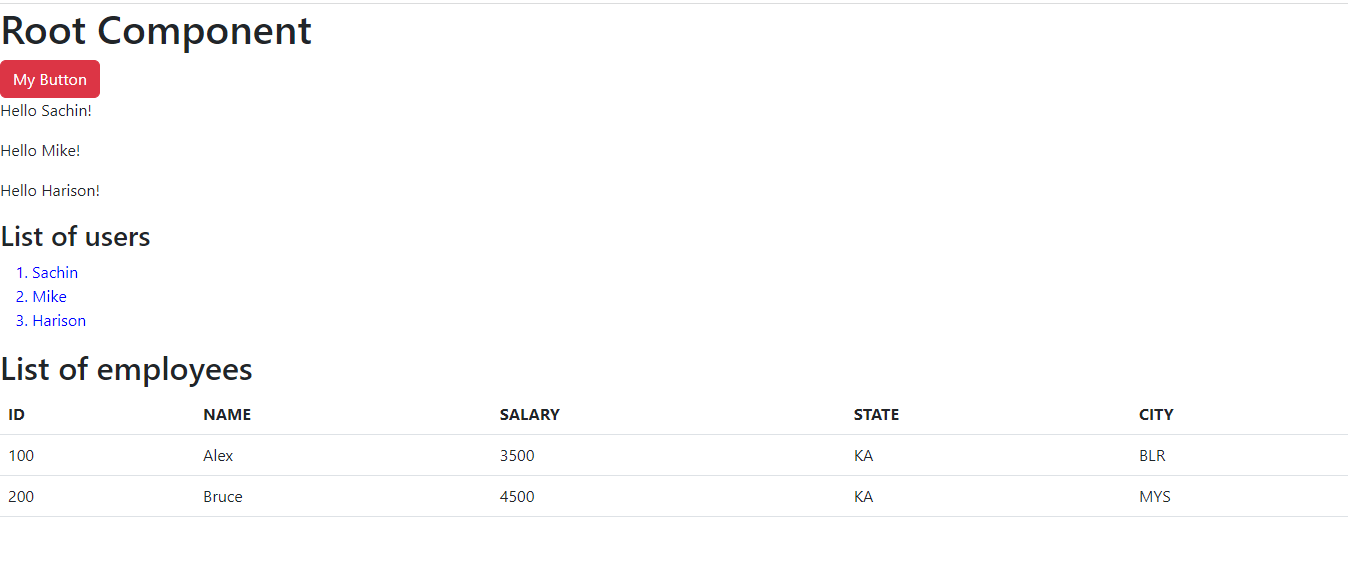
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | NAME | SALARY | STATE | CITY |
| 100 | Alex | 4500 | KA | BLR |
| 200 | Bruce | 2500 | KA | MYS |
| 300 | Daid | 6500 | TN | CHN |

Solution

App.js



Output:



Handling Events in React.js

Whenever user does some thing on the HTML elements events occurs, these events can be handled to perform actions like when you click on a button you want some action need to be performed, when you submit a form you want some action like collecting data and sending to the server needs to be performed, when you place cursor over the image you want to add some extra behavior to the image

Handling events in React.js is similar to handling events in Javascript, but the event name must use some strict case like came case

|  |  |
| --- | --- |
| Event Name | React event name |
| Click | onClick |
| Submit | onSubmit |
| Input | onInput |
| Change | onChange |

Once you handle the event you need to attach a callback, which is an event handler code that is executed when a particular event occurs.

src/App.js



We can handle data through events where these data can be updated in the UI component or it can be sent to the server side programs, for that we must use states

States

These are data in the component which can be modified, you can also share those data to other components as props

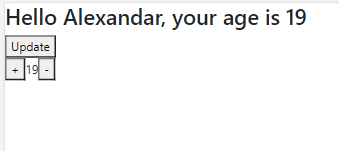
Note: Props is also a data of the component, but its read-only data, however states are updatable data

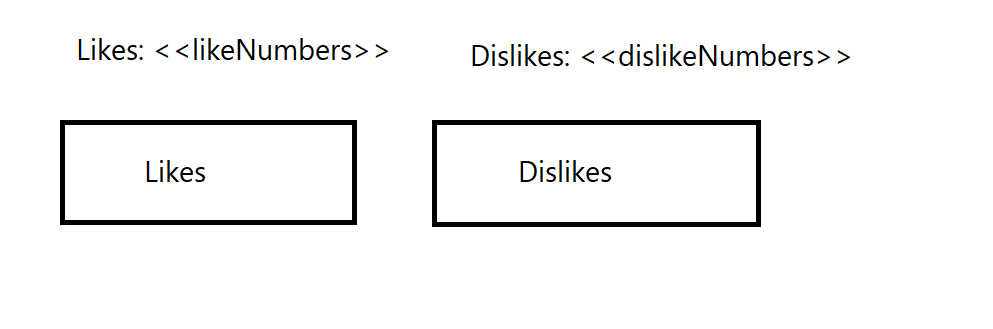
useState: It is a react hook function which helps to create states in your component

src/App.js



Output:





How to create states in components that are created using classes

It doesn’t use useState instead it uses the inherited state & setState members of React.Component

state must be initialized in the constructor & setState must be used to modify the state.

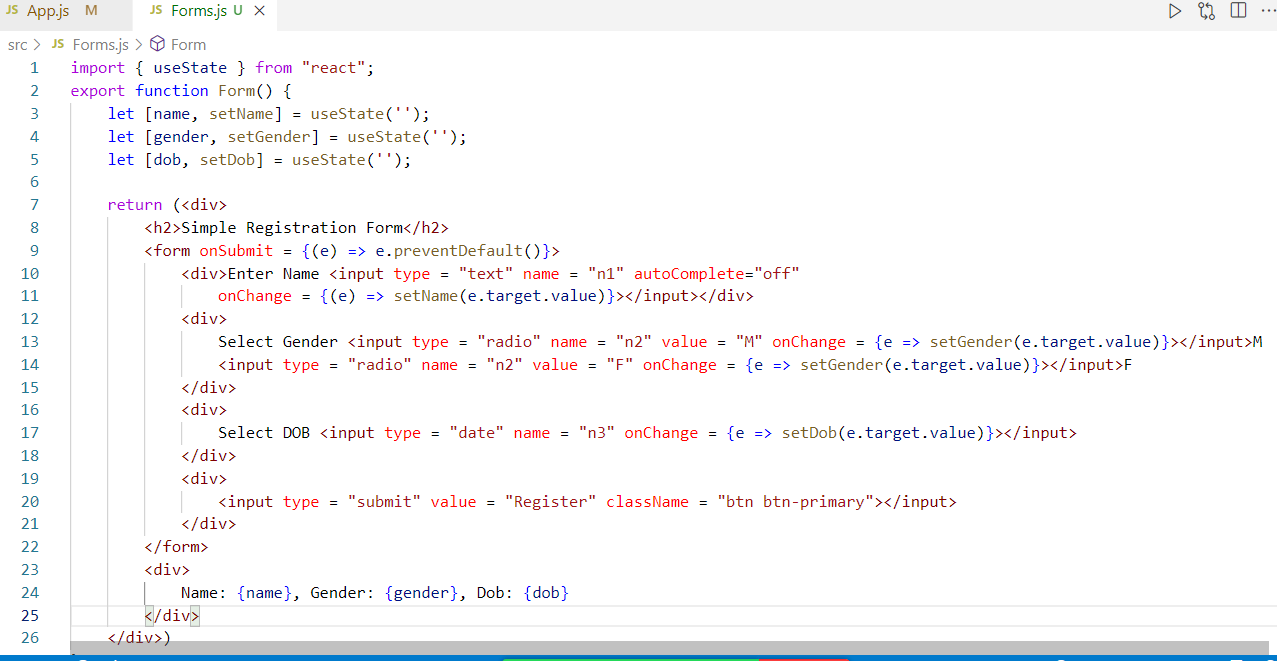
App.js



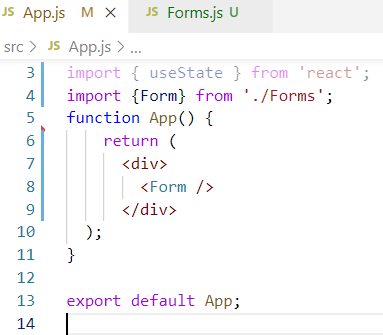
Controlled Components

These are the components which can have input controls that helps user to enter input and handle the data in the component using states

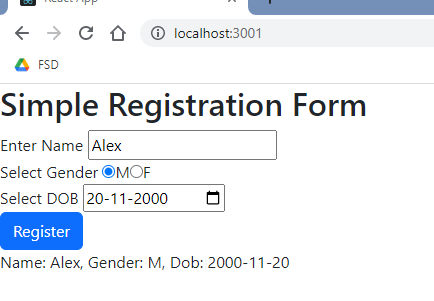
src/Forms.js



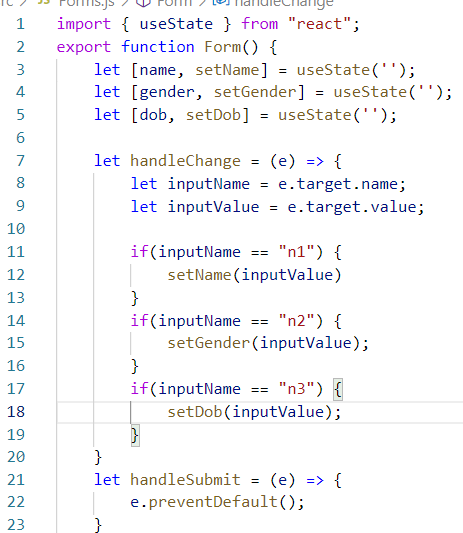
src/App.js



Output:



Handling the data in a single function without using inline function



Here we have multiple conditions to check the name of the control to update the state, however we can make this simple by completing getting rid of conditions and updating the states with the value from the form.

Updating all the states in a single line is possible when you create components using class

constructor(props) {  
 super(props);  
 this.state = {name: ‘’, age: ‘’, dob: ‘’}  
}  
handleChange = (e) => {

let inputName = e.target.name;  
let inputValue = e.target.value  
this.setState( { [inputName]: value } );

}

Session 21

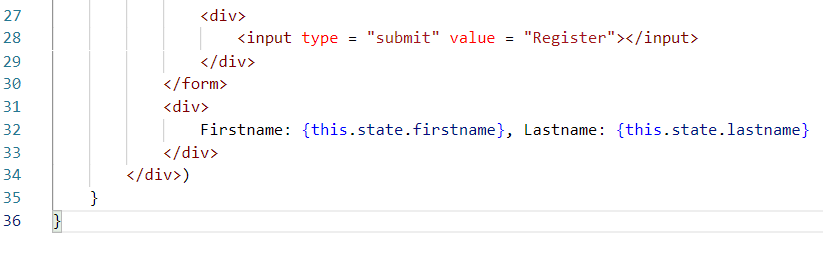
Agenda

* Simple way to update state with class components
* React Router
* Axios

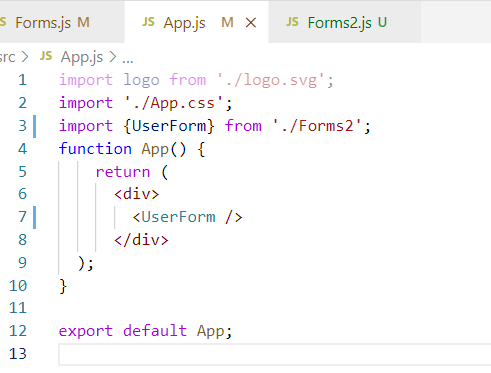
Simple way to update state with class components

Forms2.js

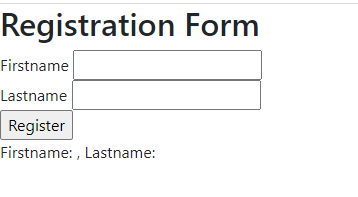




App.js



Output:



Note: Above approach updated the state without having any conditions on the properties

React Routers

They help you load multiple components within the same page without refreshing the entire page, React Router is a library which gives you predefined components to take care of routing features

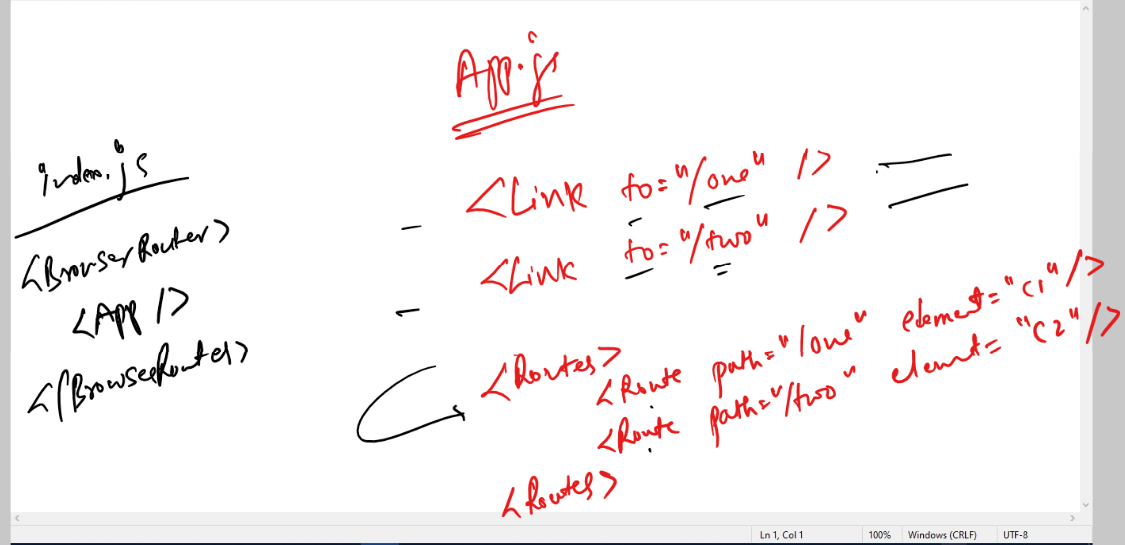
Note: React Router library we must download from the NPM

How to download

npm install react-router-dom

React Router DOM is a library that gives you predefined components like

BrowserRouter, Link, Route, Routes



BrowserRouter:

It enables routing feature to all the components, you have to keep your root component inside BrowserRouter

Link:

It creates a clickable link that specifies the url to the Routes which takes care of loading the components based on the matching url.

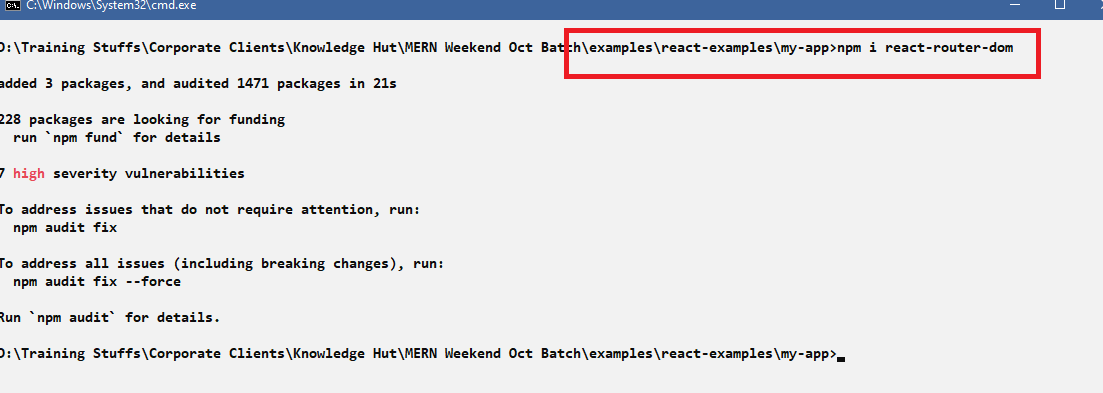
Routes & Route:

It is a container that manages all the components using Route with specific url, based on the url it gets the particular component mentioned in the Route will be shown in the page.

<Link to = “/one” />  
<Link to = “/two” />

<Routes>  
 <Route path = “/one” element = {<C1 />} />  
 <Route path = “/two” element = {<C2 />} />  
</Routes>

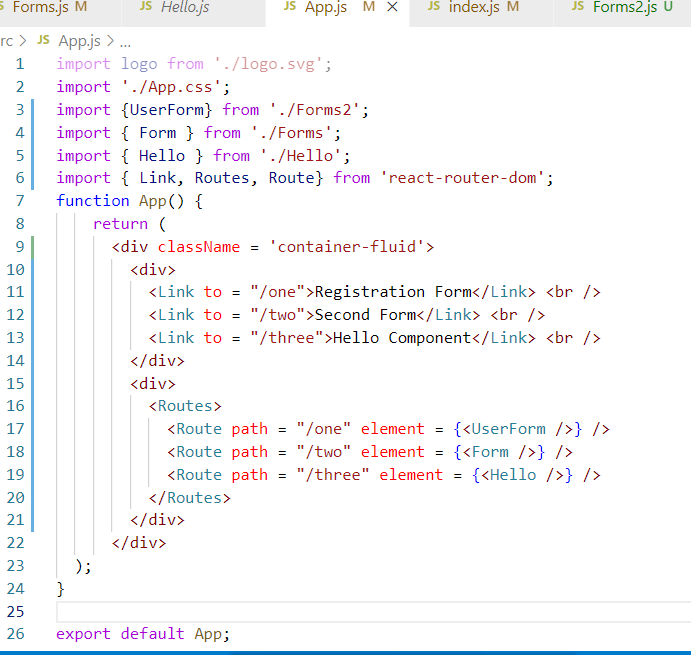
Installing React Router DOM



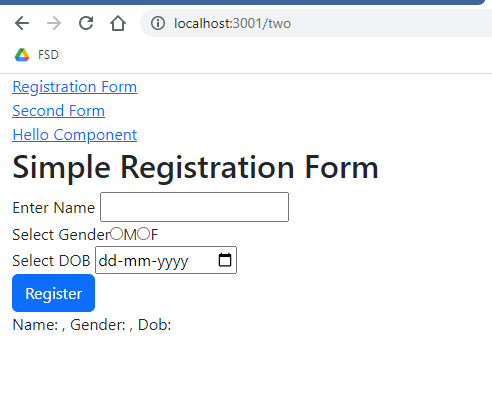
index.js



App.js



Output:



Axios

It is used to access backend API’s using HTTP methods like get, post, put & delete all these methods return promise so that you can use then() & catch() to figure out whether promise is fulfilled or rejected

It is much simpler than using fetch() API to access backend resources, since every backend resource is associated with URL & HTTP mappings, axios also provides http mapping methods like get(), post(), put() & delete() all these take URL as a parameter

axios.get(url).then(..).catch(…)

axios.post(url, data).then(..).catch(..)

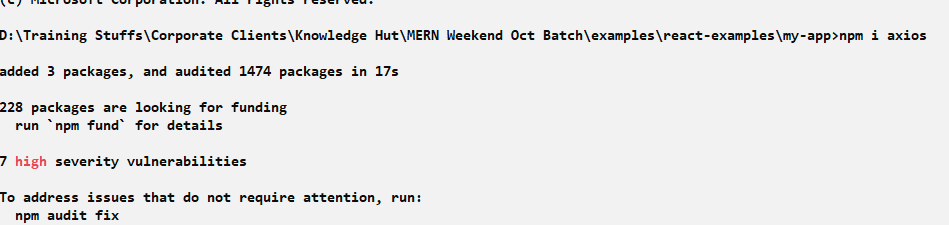
axios.put(url, data).then(..).catch(..)

axios.delete(url, data).then(..).catch(..)

Note: Axios library is not inbuilt in React, hence we must install axios library and import it from the axios module

Note: Since we don’t have any backend ready we can use JSON place holder

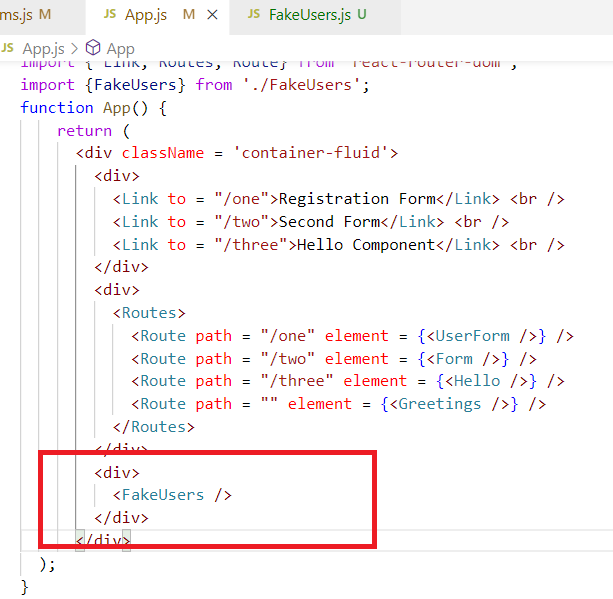
Installing axios



FakeUsers.js



App.js



Output:

