MERN Stack Training

Contents:

* Web Applications
* Fundamentals of Computer programming
* Algorithm & Pseudocode
* Linux
* Github
* HTML
* CSS
* Javascript
* Babel, Webpack
* React.js
* Node.js
* Express.js
* MongoDB
* Testing
* DevOps

Application: It is a computer program that can perform some task, there are two types of applications

1. Standalone or Desktop
2. Web Application or Distributed

Standalone application: You can use this application only after installing on your machine

ex: MS Word, Antivirus, VSCode editor and so on

Web application: These applications can be accessed over the internet without installing on your machine

ex: Bank Applications, Gmail, twitter, e-commerce website and so on.

How you can access these web applications

You need to access via browser (which itself is another application which is a standalone application that helps accessing any distributed application).



Client & Server



Both client & server are machines, where you can launch or run the applications, but servers are the one who hosts the application and client would send the request to these applications and get the response.

Web application vs Web pages

Web application can produces many web pages, it can perform various tasks and can handle the request & generate the response, these response are shown in web pages

Web application’s can show two kinds of pages

1. Static pages - These contents doesn’t change, it would be common to all the users

ex: Wikipedia

1. Dynamic pages - These contents change at runtime, it would be different for different users

ex: Facebook, Twitter, Online shopping applications, Gmail

Technologies used for static pages & dynamic pages

Static pages can be developed using HTML & CSS

Dynamic pages can be developed using various technologies like Javascript, Servlets, PHP, ASP.net, Django

Various programming languages helps you to develop dynamic web applications

1. Java: Servlets & JSP
2. JavaScript: Node.js & Express.js
3. C#: ASP.net
4. Python: Django.

URL: Uniform Resource Locator, it is used to access any web application over the internet, this is a name for a web application running in any particular server

HTTP: Hyper Text Transfer Protocol, it is used to communicate between the client and the server, it is used to exchange the data in request & response formats, which are also called as HTTP messages

HTTP messages: These are the data exchanged between server & the client, there are two types of http messages

1. Request: Sent by client
2. Response: Sent by server

Request & Response are divided into two sections

1. Header: It will have header information’s of request & response like content-type, accept, length and so on
2. Body: It will have the data, which is very much important for both client & server

History of browsers

1. WorldWideWeb Browser
2. Mosaic
3. Internet Explorer
4. Opera
5. Safari
6. Mozilla Firefox
7. Google Chrome
8. Microsoft Edge



Standalone applications are the applications that need to be installed before using

Desktop applications need to be installed in Laptops or Desktops ex: MS Word, Antivirus, VSCode and so on

Mobile applications need to be installed on your mobile phones ex: Mobile banking applications, Facebook application, Google pay, phone pay, Zomato, Swiggy and so on

Distributed applications are the applications that can be accessed over the internet, there are two types of Distributed applications

1. Web application
2. Enterprise application

Web applications are accessed only via browsers and it returns HTML as a response so that browser can easily show the content to the user

Enterprise applications are more complex applications that can be accessed by any kind of applications, they don’t provide service only for browsers, they provide service for all kind of applications like web applications, mobile applications, desktop applications, VR devices, Swiping machines, ATM machines and so on



Enterprise applications returns data not in HTML format rather in one common format called JSON (JavaScript Object Notation) that is easily readable by any applications written in any programming language, these applications are called as client applications they convert the data to the format the environment understands means, mobile applications converts the data to the format which can be displayed in the mobile screen, web applications can convert these data to the format that is displayed in the browser i.e., HTML, similarly ATM machines can convert the data in a format that can be displayed on ATM monitors and so on with Swiping devices, VR Devices.

Examples of enterprise applications

Banking applications, Google map, Facebook application, Twitter



Here the enterprise applications provide services to many client applications, ex: a banking applications are enterprise applications that can be accessed via many client applications like ATM, mobile banking apps, google pay, phone pay, paytm, swiping mahines and so on

These type of enterprise applications are allowing multiple client applications to connect and share data via webservices

Webservices: These are online services or Online API’s which are made available on the internet with URL’s for variety of client applications, webservices can convert the data in one language to JSON and vice versa.



Distributed applications are not just web applications, they are more than that

What are the other formats the webservices can use

1. JSON (More widely used is JSON)
2. XML : It is heavy weight & used rarely

Day 3:

Layered architecture

It is an architecture which is followed in every complex application as in an application there will be 1000’s of functions & you must not mix the roles of each function, every function has to perform a particular task, hence an application must have layers like

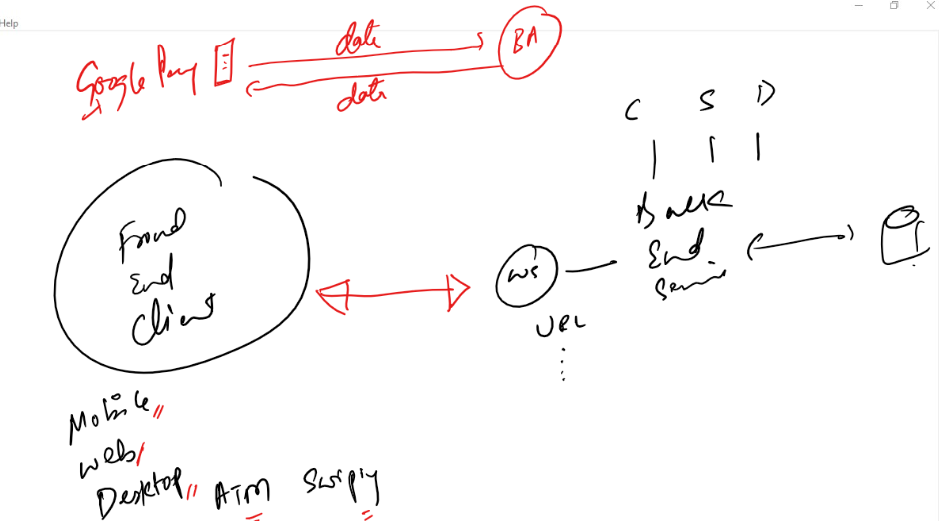
1. Presentation layer: Will have UI which user can access
2. Controller layer: Takes data from the UI and decides what is the next view the user must see and also passes the data to the UI and Business layer
3. Business layer: It is also called service layer, it applies business logic, like performing business validations before interacting with the database, it returns the data to the controller layer
4. DAO layer: DAO stands for Data Access Object, it interacts with the database and returns the data to the business layer.

Day 4

Front-end and Back-end applications

Front end applications will have only UI’s that use can access, it will not have any kind of business logics or database logics it interacts with backend webservices

Back end applications will have business logics that can interact with the database, it will have the webservices which front-end would interact with.



Ex of front end applications

1. Google pay
2. Phone pay
3. IRCTC mobile app
4. Zomato
5. Swiggy
6. Ola mobile
7. Uber mobile

Ex of Back end applications

1. Google map Api’s
2. Facebook Api’s
3. Banking application Api’s

Technologies or Programming languages that developers use to develop front end and back end applications

For front-end we have below technologies

* JavaScript
* Angular
* React.js
* Ember.js
* Vue.js
* ASP.Net

For Back end we have below technologies

* JavaScript
* Python
* Java
* C#
* C++

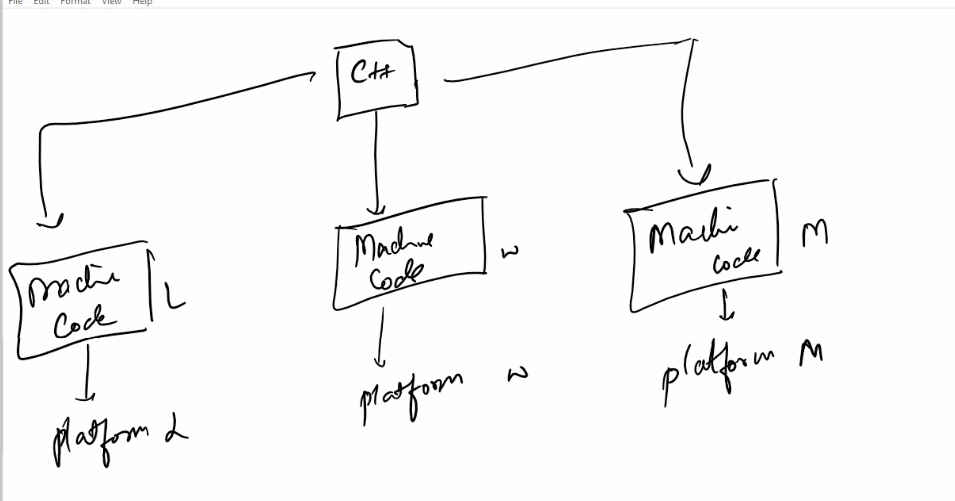
Evolution Programming languages

MLL: It is a language only Machine can understand 0’s and 1’s

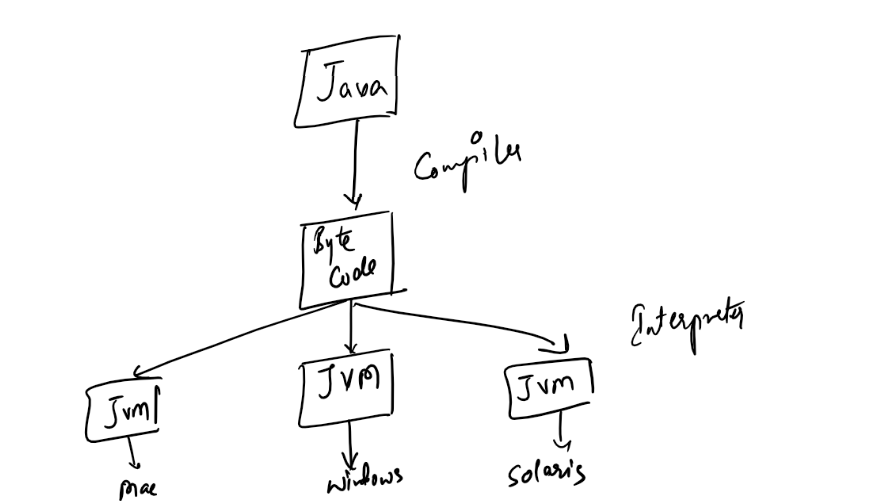
ALL: It is an Assembly Level Language, which is understood by humans, but it works only on specific processors, if you change processors, then you need to change your entire program

C: It is an High level language which is procedure oriented, but it doesn’t secure the data, as the data you create is accessible by all the procedures

C++: It is an Object Oriented Programming language, which binds the data & functions in a unit so that they are accessible only withing that unit not to the outsiders, it is platform dependent, which means the code you compile in one machine doesn’t work in another machine



Java: It is a platform independent language, it converts the source code into byte code and the byte code is converted to suitable platform specific code by JVM (Java Virtual Machine)



JavaScript: It is a language used to develop front end web pages, it is understood by all the browsers, but Java Script can also be used now for developing backend applications i.e., you can create web services with Java Script.

JavaScript needs browser to run at the front end

JavaScript needs node.js to run at the back end, node.js is a Javascript runtime environment it allows you to perform various backend operations like connecting to databases, IO operations on files, Socket IO operations (chatting applications).

Three main things a beginner must know before understanding/writing code to solve a problem

1. Algorithms
2. Flowcharts
3. Pseudocode

Algorithm:

* It is a step by step procedure which are followed to get the result for a particular problem.
* It is independent from any programming language
* It is not understood by computers/compilers, it is a design done by developers to solve the problem in English language
* Algorithms can be written in any ways, there’s no any rules that algorithms must be written in some standard way

Flowchart:

* It is also language independent
* It is used for better understanding of algorithm
* It gives the graphical representation of program sequence, it has some shapes and symbols to mention the program statements



Pseudocode:

* They are set of instructions which mimics programming language and they are also programming language independent
* It gives the idea what all the programming constructs you have to use to solve a problem, but it is still not understood by any compiler/interpreter
* Algorithms gives step by step instructions in simple English language but Pseudocode uses programming basic components/constructs like loops, conditional blocks and etc.

Writing algorithms

There’s no standard rule to write an algorithm you can write algorithm with your own statement, but it must solve the problem to give the solution

Problem:

Multiplying two numbers and display the result

Solution 1:

Step 1: Start

Step 2: Create 3 variables a, b, c

Step 3: Set value for a and b

Step 4: Multiply a and b and store the result in c

Step 5: Display c

Step 6: Stop

Alternatively you can write the above algorithm below way

Step1: Start

Step2: Take 2 numbers

Step3: Multiply the 2 number got in Step2

Step4: Print the result you got in Step3

Step5: Stop

The above algorithm is language independent, it can be implemented in any programming languages

How to implement in C

void add() {   
 int a, b, c  
 printf(“enter value for a”);  
 scanf(“a = %d“, &a);

printf(“enter value for b”);  
 scanf(“b = %d“, &b);  
 c = a \* b;  
 printf(“result c = %d “, c);  
}

How to implement in C++

void add() {  
 int a, b, c;

cout<<”Enter value for a”;  
 cin>>a;  
 cout<<”Enter value for b”;  
 cin>>b;  
 c = a \* b;  
 cout<<”Value of c = “<<c;  
}

How to implement in Javascript

function add() {

var a = prompt(“Enter value for a”);  
var b = prompt(“Enter value for b”);  
var c = a \* b;

alert(“c = “+c);  
console.log(“c = “+c);  
document.write(“c = “+c);

}

Problem: Find the area of a circle and print the area

Step1: Start

Step2: Take radius and create area variable

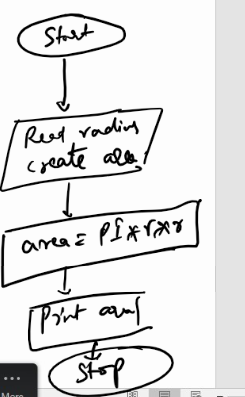
Step3: Multiply radius with radius

Step4: Multiply PI with result of Step3 and store in area

Step5: Print area

Step6: Stop

Flowchart:



pseudocode:

BEGIN  
READ radius, area  
SET radius <- value  
CALCULATE area = PI \* radius \* radius  
PRINT area  
END

Problem: Find the highest number in the two numbers

Algorithm:

Step1: Start  
Step2: Read x, y value  
Step3: check is x > y print x is greater  
Step4: else print y is greater  
Step5: End

Pseudocode:

BEGIN  
READ x, y  
IF (x > y) THEN PRINT “x is greater”  
ELSE PRINT “y is greater”  
END IF  
END

Problem: Print all the numbers from 1 to 10

Algorithm:

Step1: Start  
Step2: Initialize num to 1  
Step3: Check if(num <= 10) goto Step4 else goto Step7  
Step4: Print num  
Step5: Increment num by 1  
Step6: Goto Step3  
Step7: Stop

Pseudocode:

BEGIN  
INITIALIZE num = 1  
WHEN (num <= 10) THEN LOOP  
PRINT num  
num = num + 1  
END LOOP

Writing the program for the above Pseudocode

var num = 1;  
while(num <= 10) {  
 console.log(“num = “+num);  
 num = num + 1;  
}

Day 6:

Linux: It is one of the popular server OS which is used to deploy many applications and install servers to launch the applications, it is faster than any GUI based OS

Linux follows Unix feature, Unix is command line base OS but it is not free, however Linux is free and open source.

Open Source: You can customize as per your requirement so that it meets your organization requirement

Unix vs Linux

Unix is licensed OS, it is used in mainframes, Linux follows the same architecture of Unix but Linux is free & open source, Linux is a clone of Unix, learning linux/unix one and the same

Linux comes in 2 flavours

1. Terminal based OS: Meant for hosting servers and applications
2. GUI based: Meant for users who needs UI ex: Fedora, Ubuntu.

Installing Linux

You can install linux directly in your machine or through virtual box

Virtual box: It is a software that can host OS from the another OS, it is available for Mac, Ubuntu, Windows and other GUI based OS.

<https://www.virtualbox.org/wiki/Downloads>

Once you install either Linux terminal based or Ubuntu you will get access to the OS, but you need to use terminal of the OS to perform any operations,

Below is the snapshot of how Linux based machine looks.



Here we need to perform all the activities via commands

Some of the important commands in linux are:

date: Shows date

cal: It shows the calendar

pwd: Present Working Directory

ls: List files & folders

mkdir: It is used to create folder/directory

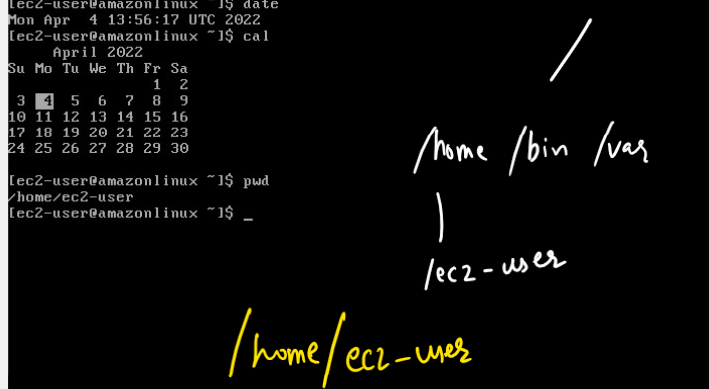
touch: It is used to create a new file

cat: It is used to concatenate files and print the files content

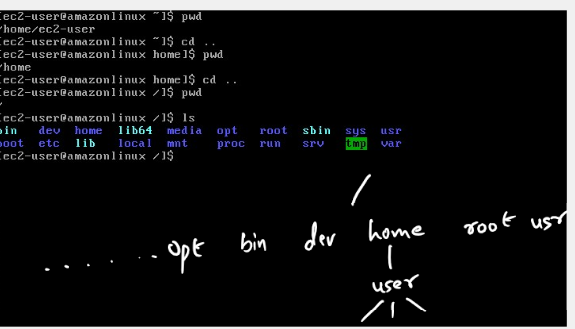
rm: It is used to remove the files & directories

mv: It is used to move or rename files & directories

In Linux/Unix you will have one root director i.e., / which is the top level directory and everything is inside this /



You can navigate backward to /home or / using cd ..



/home/ec2-user: It is the current working directory of the user, in everyone’s case the username will be different ex: /workspace/linux/playground

Some of the important commands

cal, date, cd, ls, pwd

How to create files in terminal based machines

touch filename.ext

How to edit the file

vi filename.ext

How to view the file

cat filename.ext

How to rename the file

mv oldname.ext newname.ext

ex: mv simple.txt hello.txt

Copy the files content to another file

cp sourceFile.ext destinationFile.ext

1st command: *touch hi.txt*

2nd command: *cp hello.txt hi.txt*

3rd command: *cat hi.txt*

View multiple files

cat -n sample.txt demo.txt

Writing multiple files content to a new file

cat sampe.txt demo.txt > example.txt

cat example.txt

Creating a folder

mkdir folder-name

ex: mkdir test

Copying files to the folder

cp filename folder-name

ex: cp sample.txt test

Ubuntu: It is linux based machine provides GUI, but you can use terminal in Ubuntu as well to make use of many linux related commands like installing/unistalling any softwares

There are other flavours of Linux which provides GUI

* Red Hat linux
* Fedora

How to remove the files

rm filename.ext

How to remove the folder

rm folder-name

How to remove the non-empty folder

rm -rf folder-name

-r: recursive

-f: force

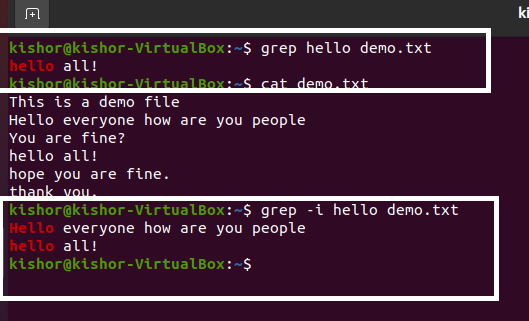
Grep: Global Regular Expression Print:

It is mainly used to search the characters in the file

grep textcharacters filename: It searches the text in a particular file

ex: grep hello demo.txt: This lists all the lines having hello in demo.txt

ex: grep -i hello demo.txt: This lists all the lines having hello and -i for case insensitive

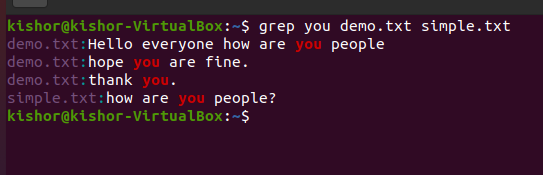


Counting the words using grep



Create multiple files and add some contents and search for a common words in the multiple files using grep

i.e., grep hello demo.txt simple.txt test.txt



How to install/remove any softwares in linux/ubuntu OS

If you want to install or remove any softwares you need to use super user command called sudo along with that you need to use installation commands

ex: sudo apt install git -y

ex: sudo apt install nodejs -y

The above commands installs git & nodejs on the linux based machine, in some linux based machine apt may not work, in that case you can use yum

i.e., sudo yum install git -y

In ubuntu it took 30s to install git

How to remove git

ex: sudo apt remove git -y

Day 8

Summary of linux commands

cat: viewing the file and also copying multiples files into another file or folder

cp: copying the files & folders

rm: removing the files & folders

rm -rf: removing the folders that are not empty

grep: It is for searching the strings in the file

cd: change director

vi: Editor

touch: to create files

mkdir: to create directories

sudo: super user do to install any softwares

sudo apt/yum install: Used to install any softwares

Note: In Linux terminal itself you can use git commands like push, pull, commit, add, status, log, merge, branch and so on

Git:

Is a distributed version controlling system (DVCS) or centralized version controlling system (CVCS), it keeps track of every changes the user does with some version id, so that user can go back to any version if required.

Git gives lot of commands to share each other codes and work in a team so that everyone in a team will have each other work/changes

Git can give you the changes back even if you have deleted your changes in your machine permanently because it maintains those changes in the remote repository.

Repository: It is a working directory in the git which will track the changes done, there are two types of repositories

1. Local Repository
2. Remote Repository

Local Repository: This is a working directory which is present in the user machine

Remote Repository: This is a working directory which is present in the git hub

How to work with Git & the repositories

1. You must have Git Hub account
2. You must have git installed to interact with the Git hub through git bash

Git bash: This is a terminal provided git to enter various git commands and interact with the Git hub

.git: This is a folder provided by git in your repository that keeps track of every commits/changes you make in the git, it is an hidden folder, it will also have a link to the remove repository

Branch: It is a pointer with history of commits, by default you will have a main/master branch.

Master branch: It is used as a stable branch for any work like for development, testing, production and so on

Useful Git commands

git clone <<url>>: It is used to clone the remote repository to your local machine

git add <<path>>: It adds the changes to the staging area

git commit -m ‘some message’: It commits the work and creates an unique id

git push -u name branch-name: It pushes the branch to the remote repository, the name here is the alias for the remote repository url, i.e., origin,

branch-name can be master or main or feature branch, -u is upload/update

git pull: It pulls the changes from the remote to the local repository

git checkout branch-name: It switches from one branch to another branch

git branch branch-name: It creates a new branch/feature branch/custom branch

Branch: It is a pointer with all the commits, It is independent from any other branch, whenever you create a branch it will clone all the commits of the current branch(master/main) where you created a branch

git merge branch-name: This is used to merge the branch into the checked out branch, this is used to manually merge in the local repository.

git branch: This lists all the branch present in your local repository

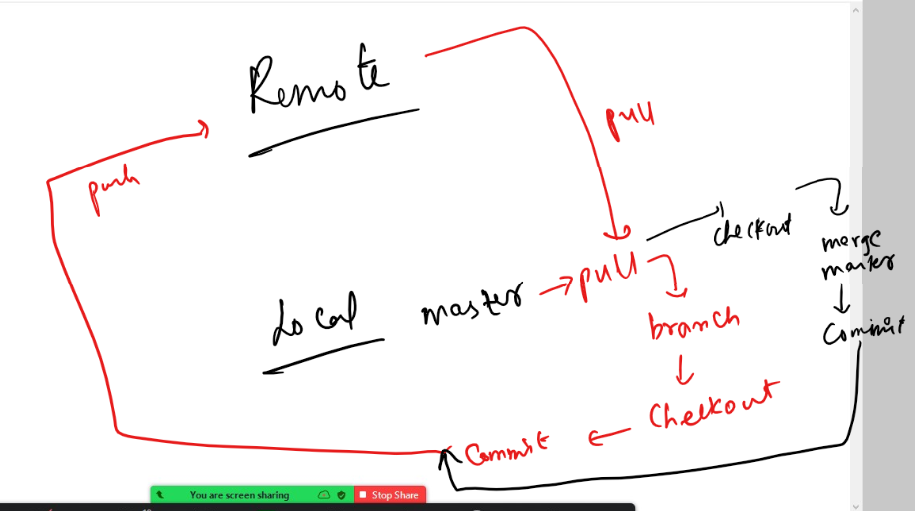
git log: This shows all the commits happened in the checked out branch

git branch -D branch-name: This deletes the branch in your local repository

The below output shows deleting the alex branch



Note: We must always work in feature branch we should never do any kind of modification in the master branch, master / main branch is always updated from the remote repository via git pull



Summary:

1. Create a new repository in your account
2. Create one folder and open git bash and enter git clone <<your repository url>
3. Do some changes in master and push that to the remote
4. Create 2 folders D1 and D2
5. Open Git Bash in D1 and enter git clone <<your repository url>
6. Open Git Bash in D2 and enter git clone <<your repository url>
7. Create a feature branch in D1 & D2 and so some changes in the feature branch and try to edit the same file and push to the remote, you will get conflict
8. When you get conflict pull the remote in your local master, checkout to feature branch, merge the master with feature and resolve the conflict.

Commands:

Clone: git clone <<url>>

Push: git push -u origin branch-name

Pull: git pull

Merge: git merge branch-name

Add: git add filename

Commit: git commit -m ‘some message’

Status: git status

Log: git log

New Branch: git branch branch-name

Checkout: git checkout branch-name

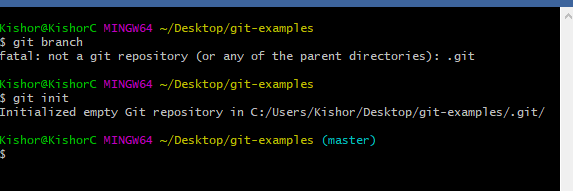
Initializing the local repository

There are two ways to initialize the local repository

1. git clone <remote-url>
2. git init

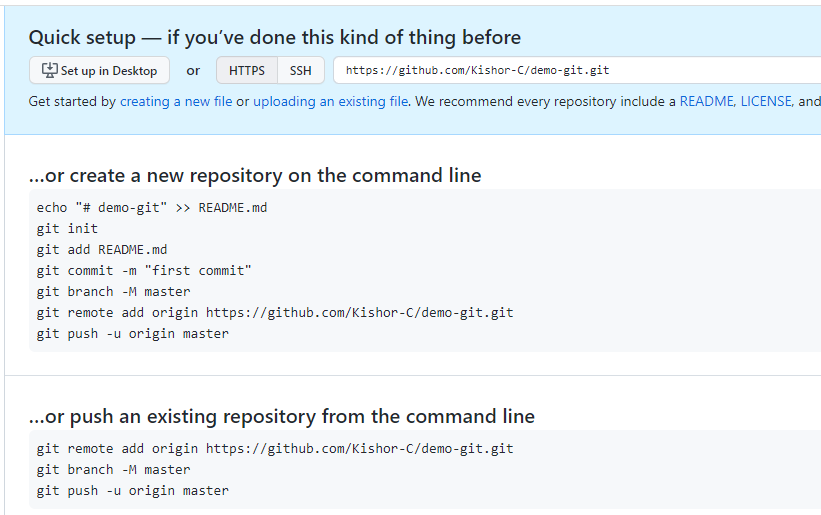
The git init creates a new local repository but it must be linked to the remote repository using git remote add <url> command.

Creating repository from the git init



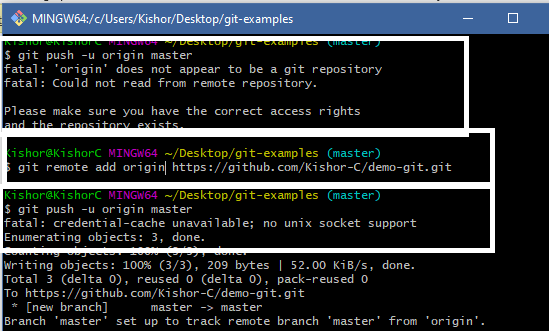
Note: You can notice the branch name is already appearing, since it is not having any commits at least you must create first commit to create any branch.

Create a remote repository and follow the instruction to link local to remote repository



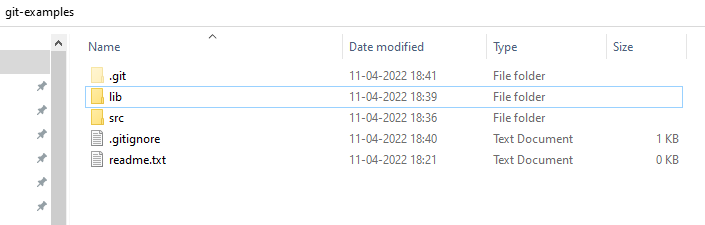
You can use git remote add origin <<url>> to link local repository to the remote repository.

Note: origin is just an alias name, you can change it if you need

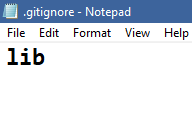


.gitignore: It is a text file that can list out all the files/folders that git should avoid tracking

You can create some dummy folders like src, lib and .gitignore, keep some files in lib & src and edit .gitignore file and enter lib so that git doesn’t track lib.

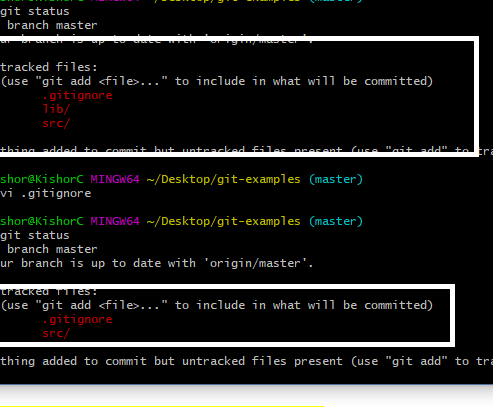


.gitignore

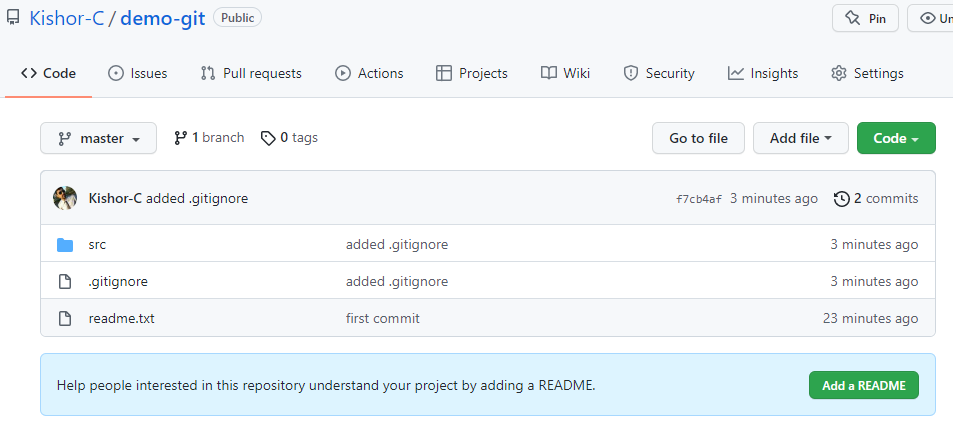


So now the Git doesn’t track lib folder, if you want multiple folders/files to be mentioned write in separate lines

Now if you use git status you see only src and .gitignore but no the lib

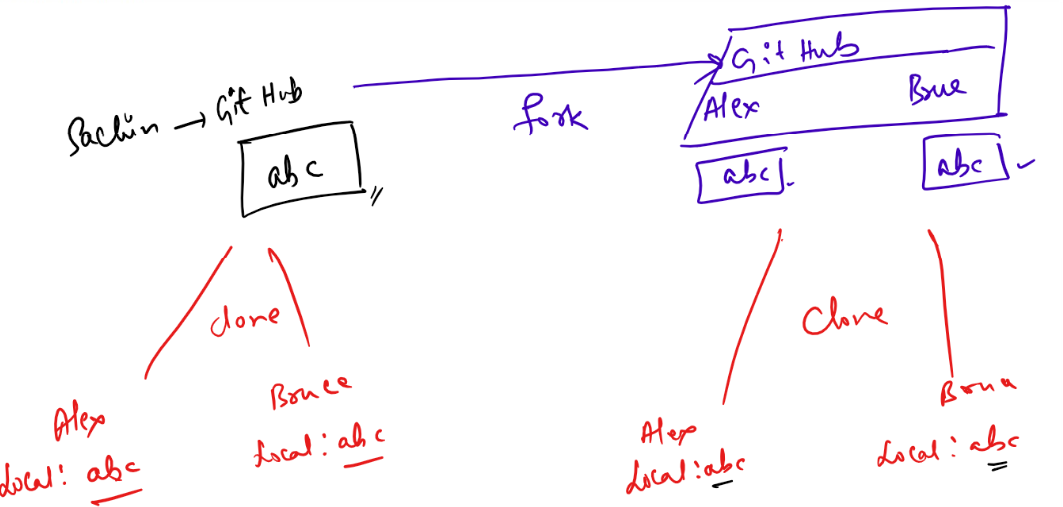


After add, commit & push you can see src, .gitignore in the remote, but lib is not pushed



fork:

It is used to clone the remote repository of one’s account to another remote account, so that you can work with that repository independently



Web application: It allows you to create web pages where user can access via URL, these pages are rendered by browser, there are the main languages understand

1. HTML
2. CSS
3. Javascript

HTML: Stands for Hyper Text Markup Language, used to display the content in your browser

CSS: Stands for Cascading Style Sheet, used to design the web page

JavaScript: It used add dynamic behaviour to the web page by accessing HTML at runtime & applying CSS at runtime, validating the user details

HTML tags

These are the tags that can have content and some may not have contents, some the basic HTML tags are

h1, h2, h3, h4, h5, h6: These are used to create heading of different size.

p: It is used to create a paragraph

img: It is used to embed images

table: It is used to create tables

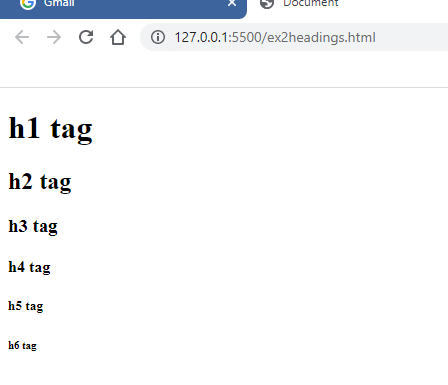
ol, ul: It is used to create lists

form: It is used to create forms

ex2headings.html



Output:

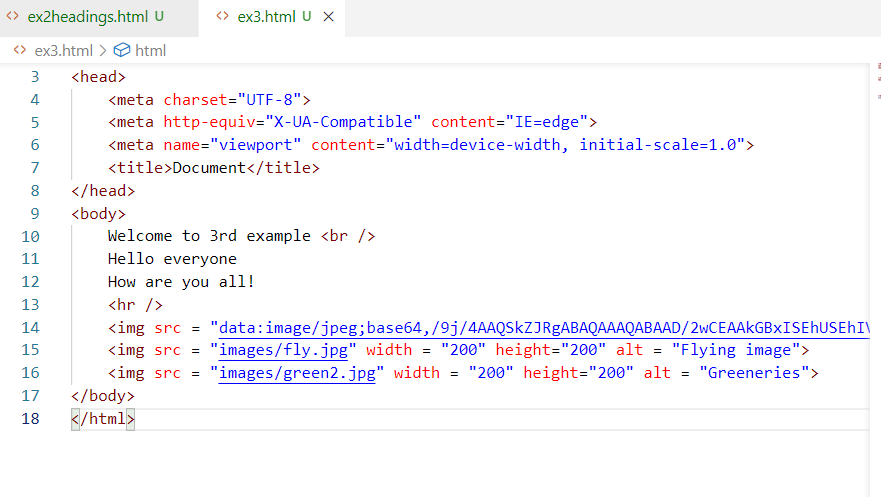


<br>, <hr> tags: These are self closing tags, they are used without content, br is a break tag which gives line break, hr is used to draw an horizontal line

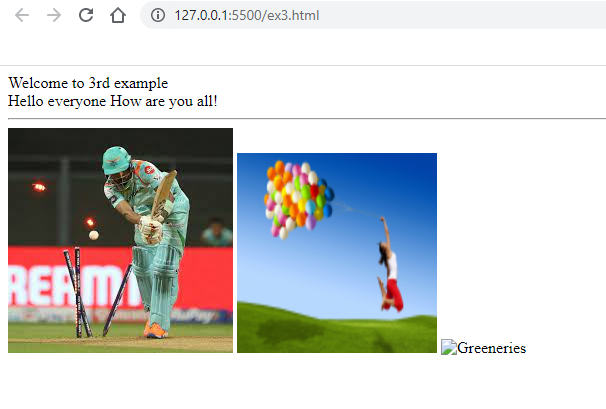
img: It is used to embed the images in the HTML, it uses some extra attributes to mention the location of the image & width & height of the image.

i.e., <img src = “url” width = “200” height = “200”>

ex2.html



Output:



Note: The alt attribute displays the message if the image is not loaded

Entity names:

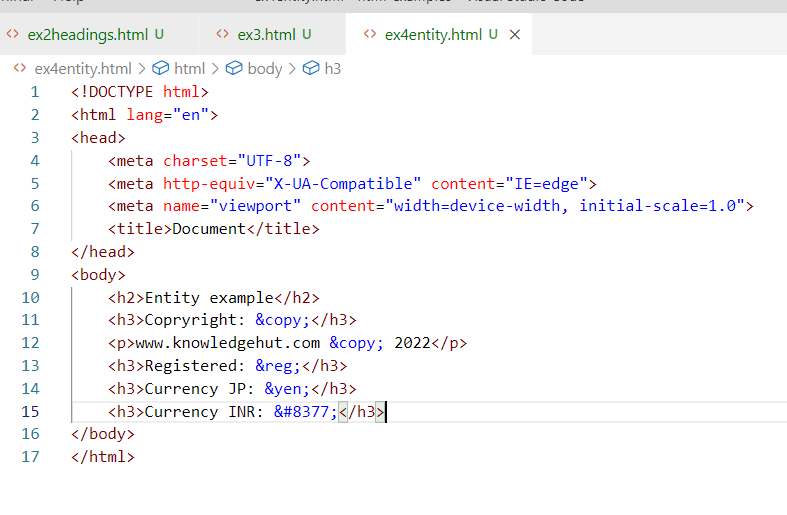
These are some special characters that browser can show though you can’t type them in the HTML document, these starts with &entity-name;

Ex: Registered mark (&reg;), trade mark(&trade;), copyright symbol(&copy;), currency symbols (&#numbers;), and so on

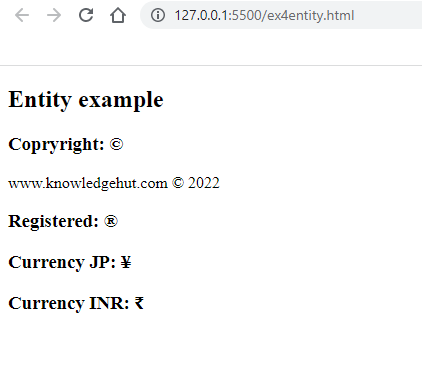
Registered mark: &reg;

And also if you want to show < or > symbols, you can use entity names like &lt; and &gt;

ex4entity.html



Output:



List tags:

There are 2 types of lists

* ordered <ol>: It will have some sequence of either number or alphabet or roman no’s
* unordered <ul>: It will have bullets like circle, disc, squar

Both these tags lists item using <li>



div: It is a container tag used to group multiple HTML elements, it can be later used to apply some css to style all the elements inside the div, it is used to layout the web sites as well, for header, footer, navigation and so on

span: It is an inline tag, so that you can have a content withing another element

ex6div.html



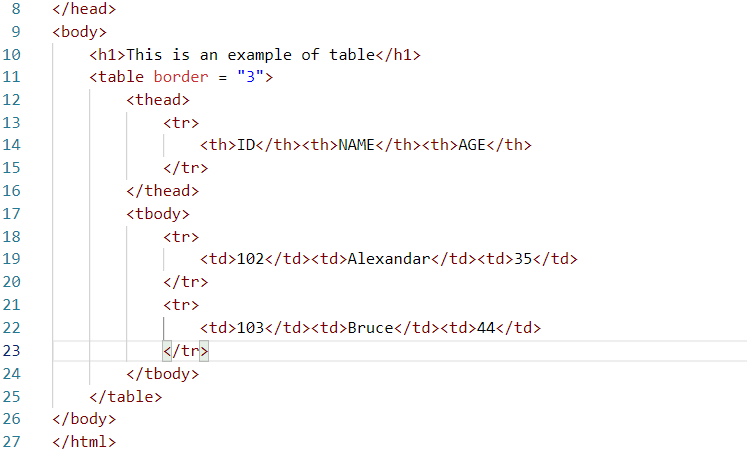
Here div groups multiple tags later you can apply css to the div which applies to all the elements indie div and span element is used to select some content within another element.

Output:



table: It is used to create table so that you can have table items in row and column, it has <tr> to create row and <th> for heading <td> for table data

ex7table.html



Output:

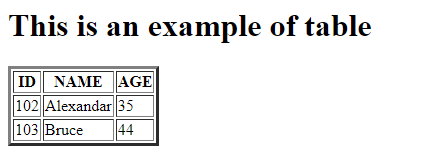
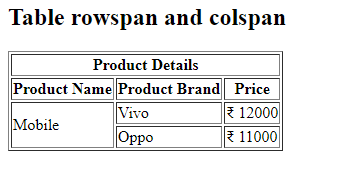


table rowspan & colspan: These are the attributes you can use to allocate some rows and columns for the table data

ex8tablespan.html



Output:



Activity:

Use rowspan and colspan for table and display the product details as below:

|  |  |  |  |
| --- | --- | --- | --- |
| Product Details | | | |
| Product Name | Product Brand | Price | |
| Rs | Ps |
| Mobile | Vivo | … | … |
| Oppo | … | … |
| Samsung | … | … |
| Laptop | Dell | … | … |
| HP | … | … |
| Acer | … | … |
| Lenovo |  |  |

Form:

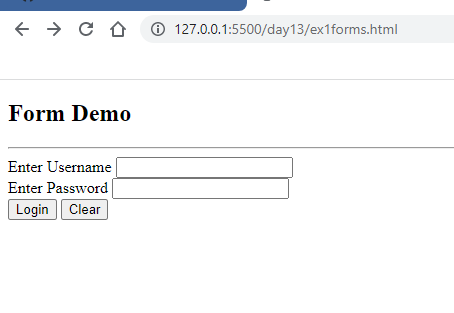
It is used to create controls where user can enter input in it, you can create controls like

textbox, password box, radio, checkbox, dropdown, textarea, button and so on.

ex1forms.html



Output:



By default form method = “GET”, which sends the data via request header, so the data will be visible to the end user, if method = “POST”, then data is sent via request body, the data sent via body wouldn’t be visible, but backend can read that data.

Get vs Post

|  |  |
| --- | --- |
| GET | POST |
| This is the default value for the form method | This must be mentioned explicitly in the form method |
| Data will be sent in the request header so the data will be visible in the URL | Data will be sent in the request body so the data wouldn’t be visible |
| It supports limited set of characters i.e., maximum it supports 256 characters | It supports unlimited characters |
| Insecure | Secured |
| Faster | Bit slower compare to GET |

GET & POST are part of HTTP methods, along with GET & POST there other HTTP methods like PUT, DELETE

Note: Form method supports only 2 http methods

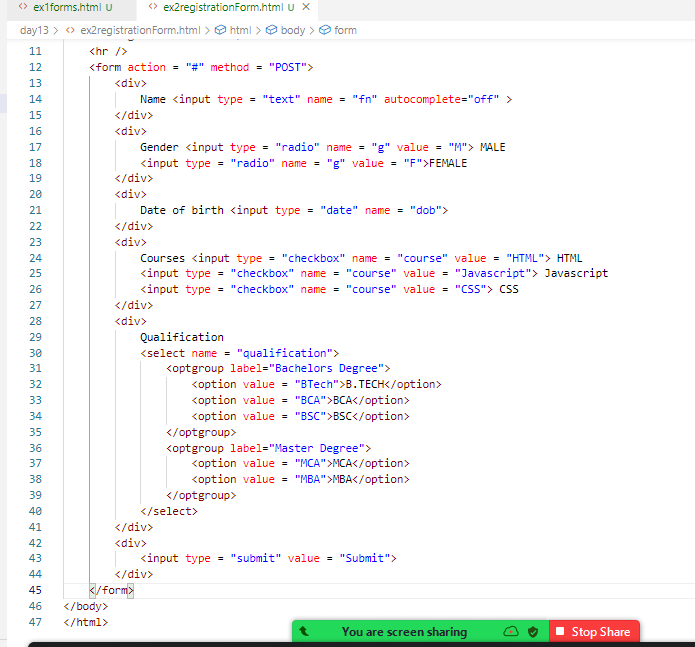
* GET
* POST

Other HTTP methods you can use programmatically , which we will learn in webservices

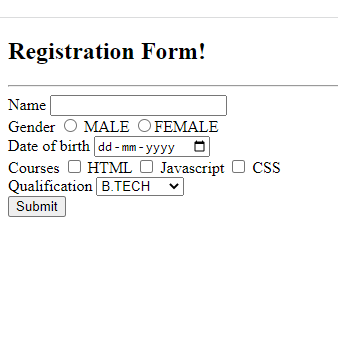
Http has some standard for GET, POST, PUT & DELETE, so both client server must use them according to the standard

1. GET: Must be used only for retrieving data
2. POST: Must be used when a new data should be created
3. PUT: Must be used when existing data must be updated
4. DELETE: Must be used when data should be deleted

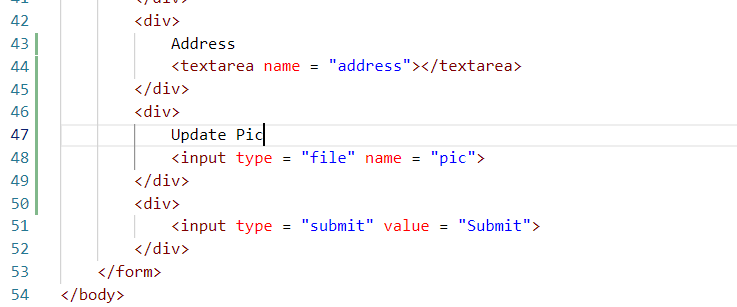
ex2registrationForm.html



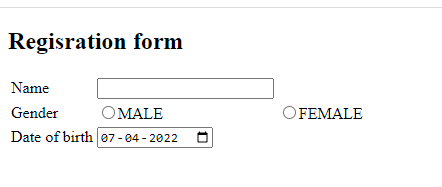
Output:



You can add upload pic button and text area as below;



But the output of the form controls are not properly aligned, to align better you can use <table> tag without border, below is the output you get if you use <table> <tr> <td> elements



CSS

Cascading Style Sheet it is used to style the HTML elements, there are three types CSS you can create

1. Inline CSS
2. Internal CSS
3. External CSS

Inline CSS: It is used to apply style on a particular element within the HTML document, it is done using an attribute called ‘style’ which takes “property: value” as value to the style, here property could be any CSS properties they are predefined & understood by browser like color, background-color, font-size, font-family, text-align, width, height, border and so on, the value is related to the property you are using ex: for color you can use red, green, blue, #FF00FF, rgb(255, 0, 0)

ex: <p style = “color: red”> This element gives red color

<p style = “color: rgb(255, 0, 0)”> This element gives red color

<p style = “color: #FF0000”> This element gives red color

Internal CSS: It applies the style to the whole HTML documents, this is used when you want an unique style for the entire HTML documents, this is achieved using <style> tag that needs to be used in the <head>

Syntax: selector { property : value; property: value,…}

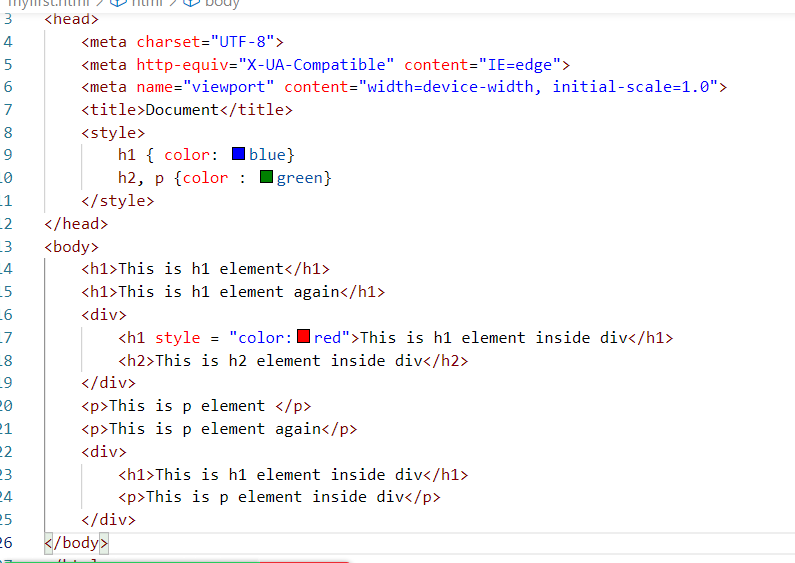
<style>  
 h1 { color : red }

h2 { color : blue; background-color : yellow}  
</style>

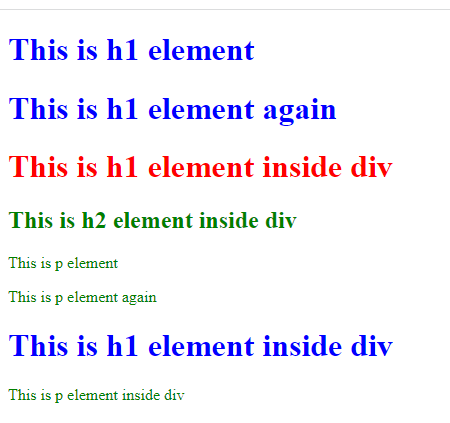
External CSS: It is used to apply common styles to multiple HTML documents, this is done by creating a .css file and that needs to be linked to the HTML document

Note: If you provide a style to a particular element in all the 3 types of CSS then preference would be given to inline first, then to internal at last to the external.

myfirst.html

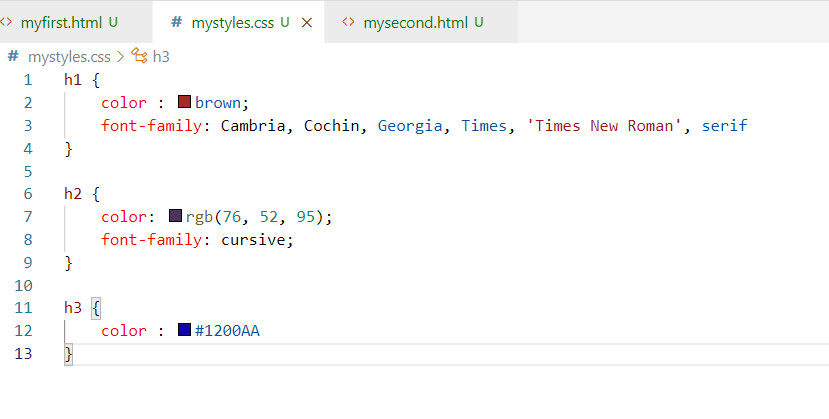


Output:

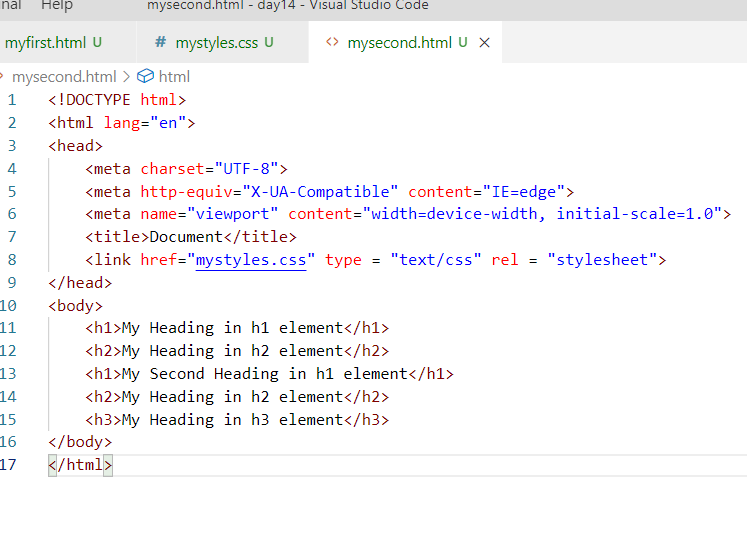


Using external CSS

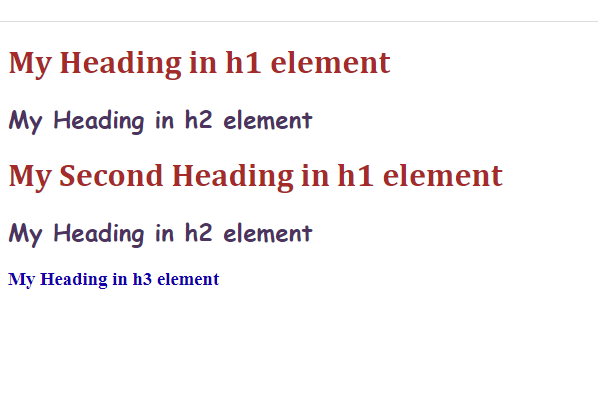
mystyles.css



mysecond.html



Output:



Different types of selectors

1. tag selector: p, h1, h2, h3, div, …
2. class selector: .name
3. id selector: #name

mystyles2.css

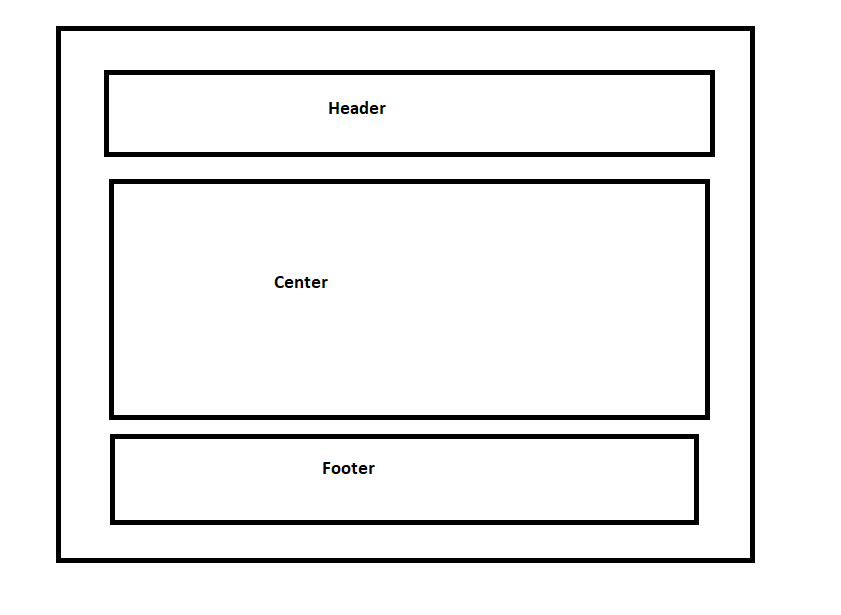


mythird.html



Activity:

1. Try above activity
2. Create a simple website that will have header, footer and center content



Apply CSS to the above layout

In HTML5 there are many container tags works like <div>, but you need to use CSS to properly align elements, some of the container tags are:-

<header>, <section>, <aside>, <nav>, <footer>

selector1 > selector2: It selects the elements that are child of selector1

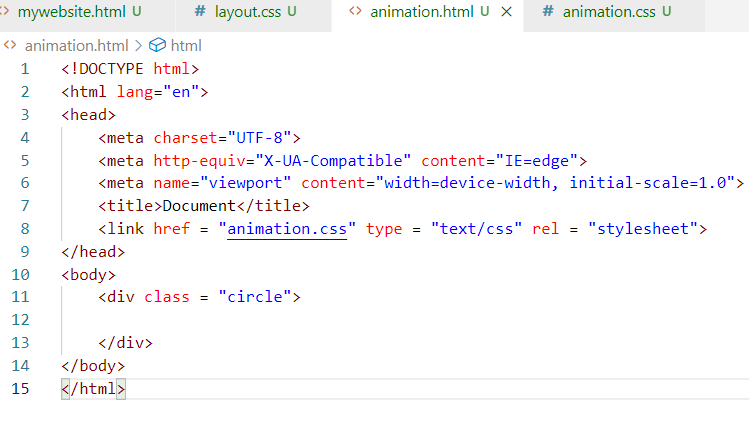
Animations in CSS:

It adds a smooth transitions and transformations to the HTML elements, to apply animations you need to use @keyframes and mention what animation you want

animation.css



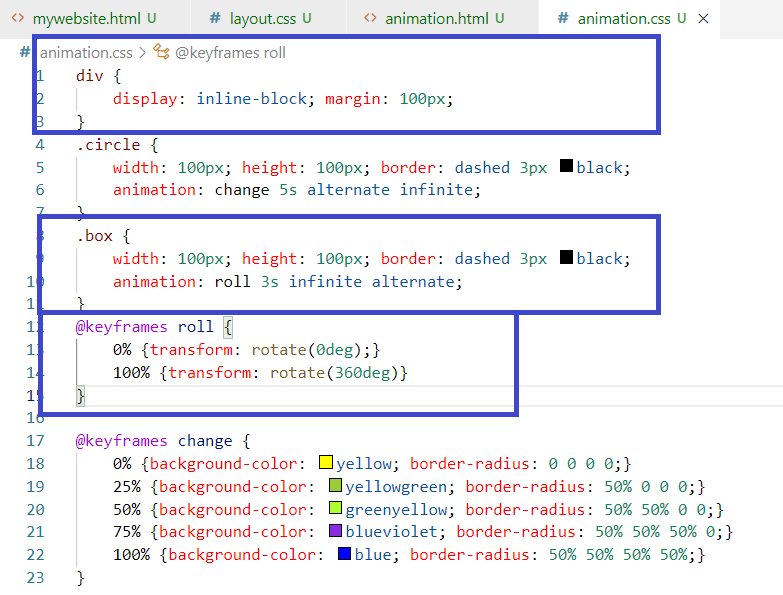
animation.html



Output:



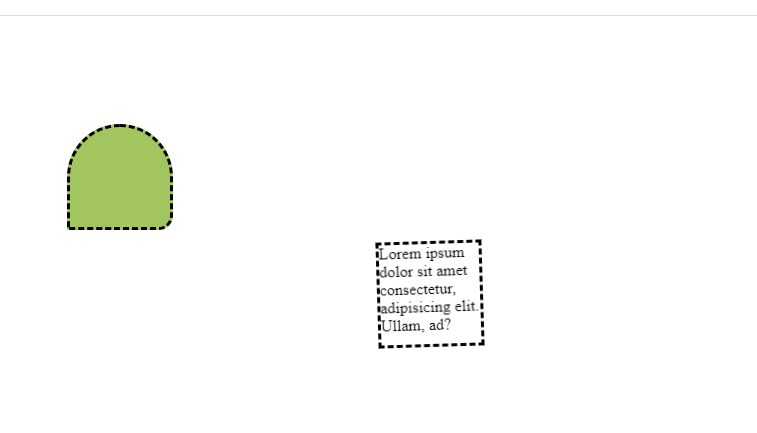
You can also make these elements to rotate using some predefined functions provided in CSS



animation.html



Output:

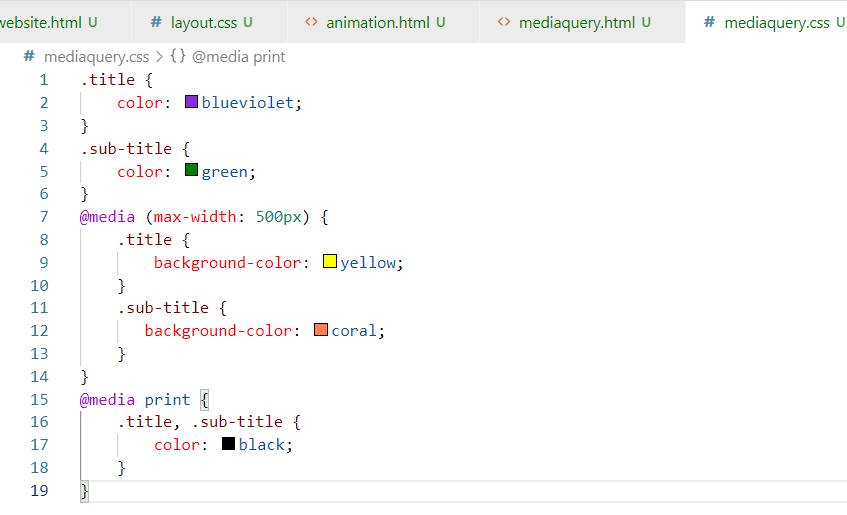


Media Query:

It is a CSS technique which applies the CSS properties only if certain condition is met.

It uses @media rules to add the CSS properties

mediaquery.css

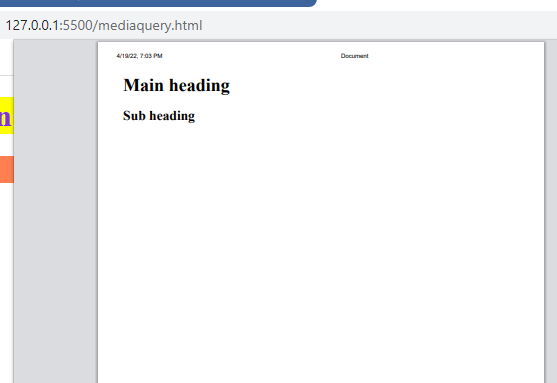


Here the media query applies when width of the screen is 500px or less than 500px and also when you take a print then it turns color to black.

Output:

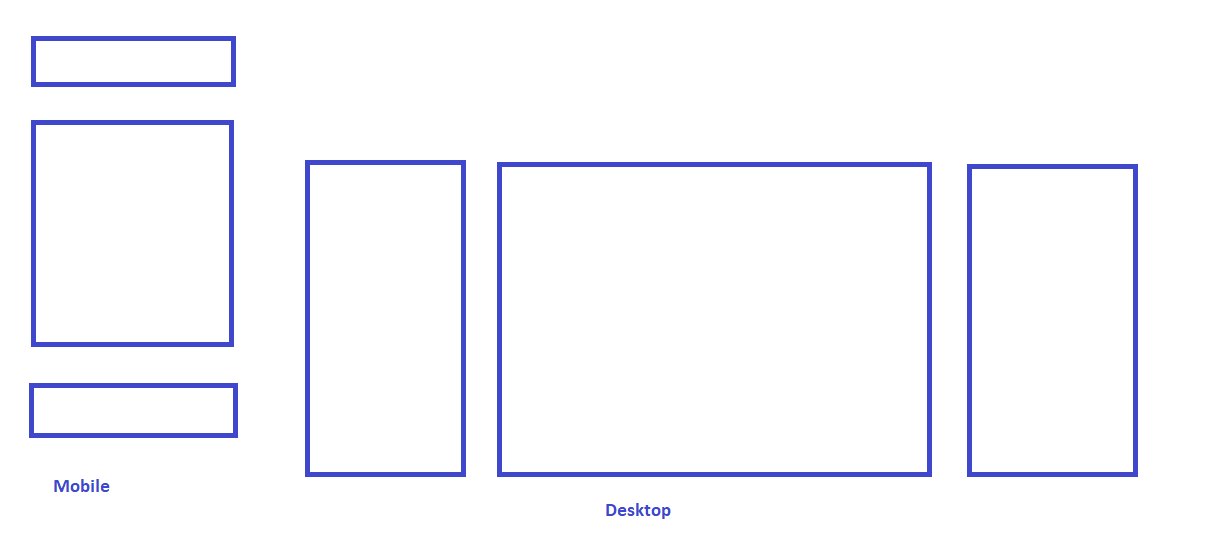


At the time of print title & sub-title will be black color



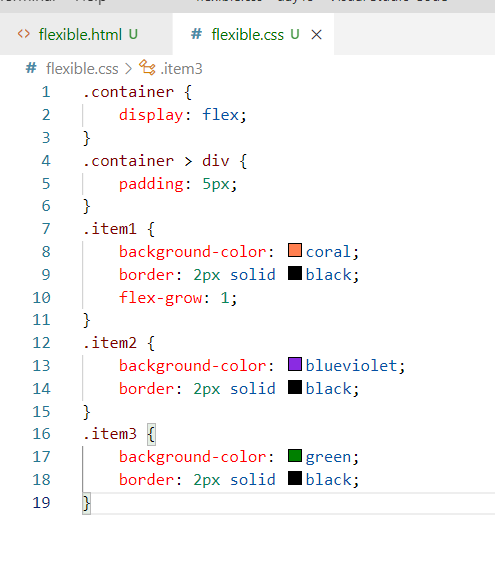
Flex box layout:

It provides an efficient way to layout items, or align, distribute space among the items in the container when their size is unknown



A flex container can expand or shrink whenever need, you need to use display: flex to add the flexible container

flexible.css



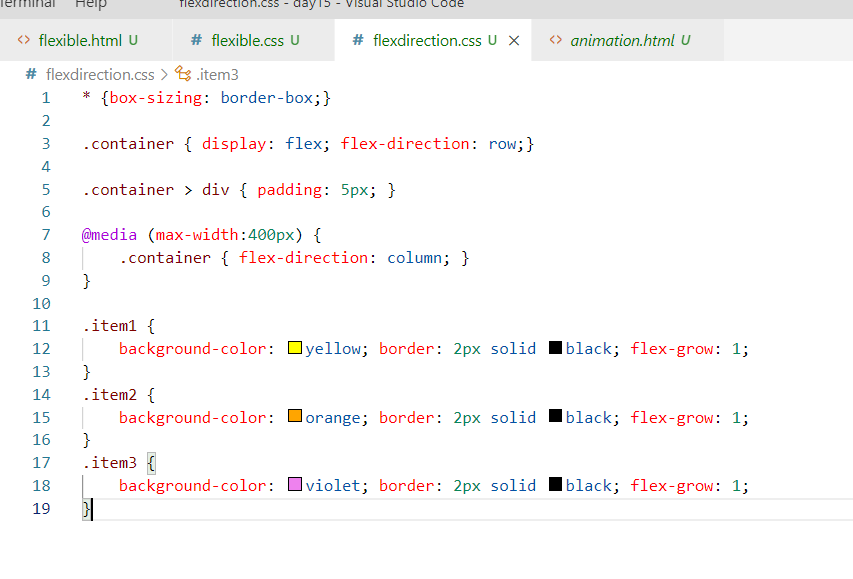
flexible.html



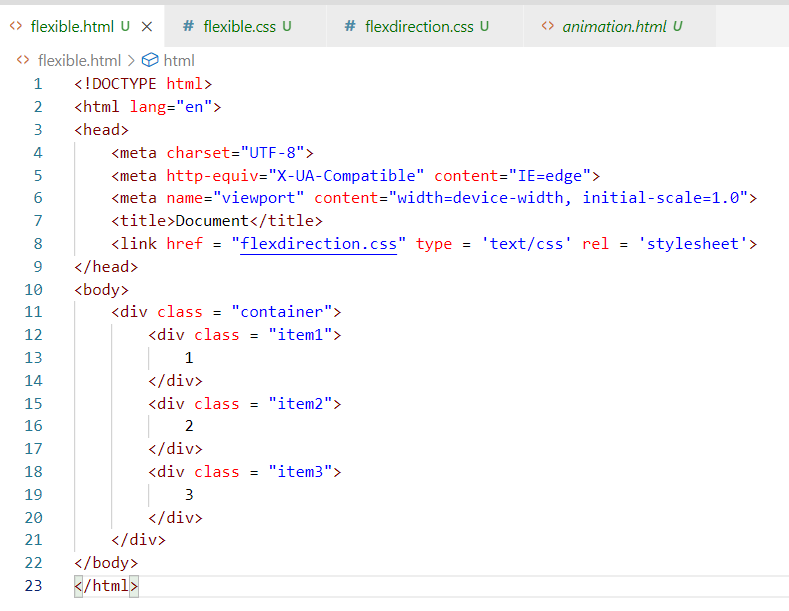
The item1 is using flex-grow:1 hence it grows as much as possible

Flex direction: It is used to array the items either in row or column, this is important when it comes to aligning the items based on the screen size, if the screen size is bigger you can arrange the items row-wise, if the screen size is smaller or if is mobile device you can arrage the items column-wise.

flexdirection.css

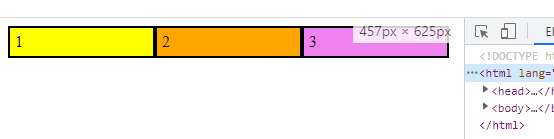


flexibile.html



Output:

When the width is > 400px



When the width is <= 400px



Grid system:

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

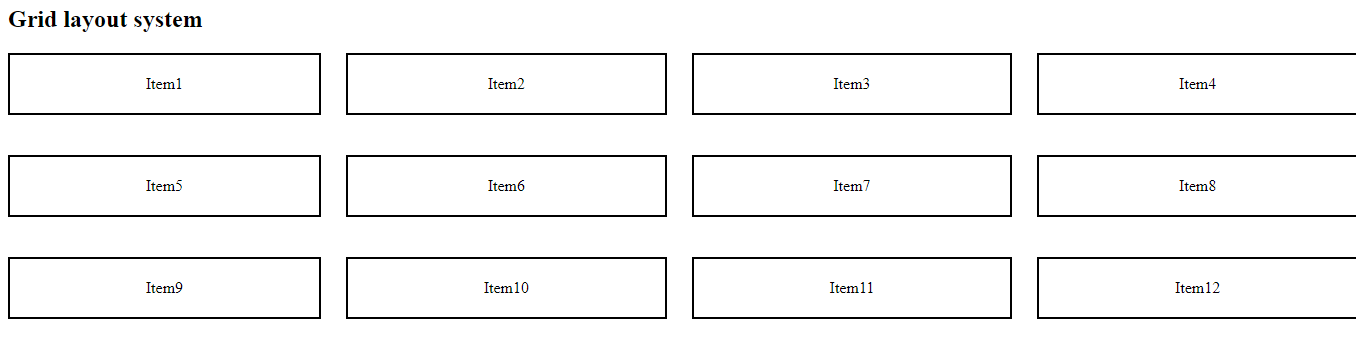
mystyles.css



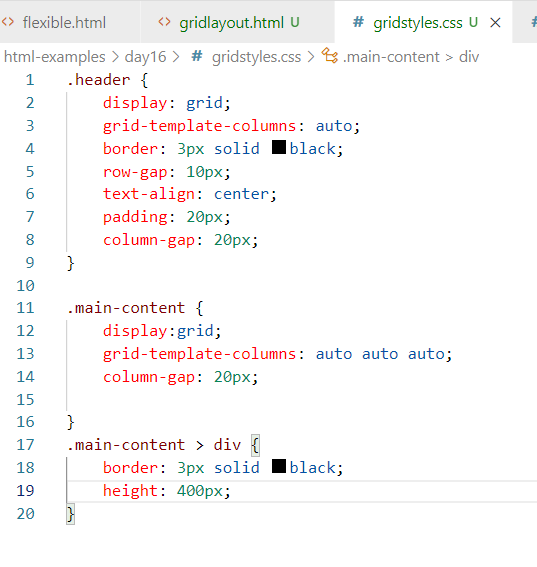
gridlayout.html



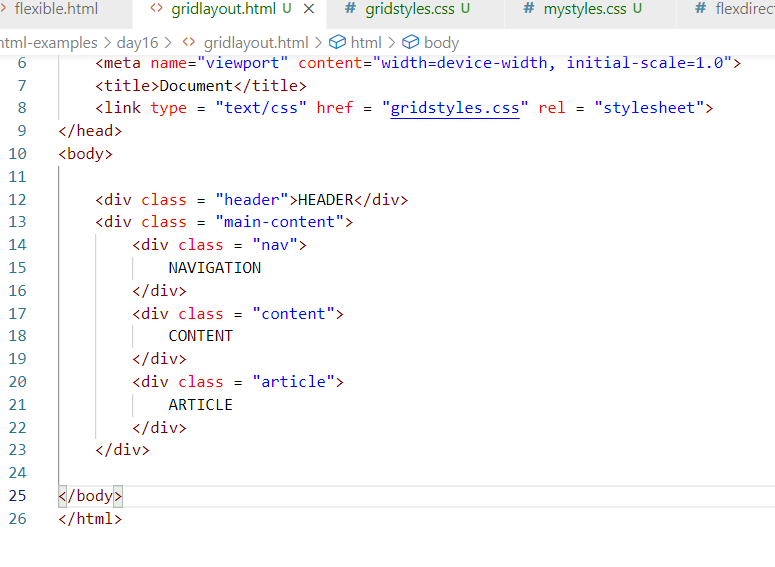
Output:



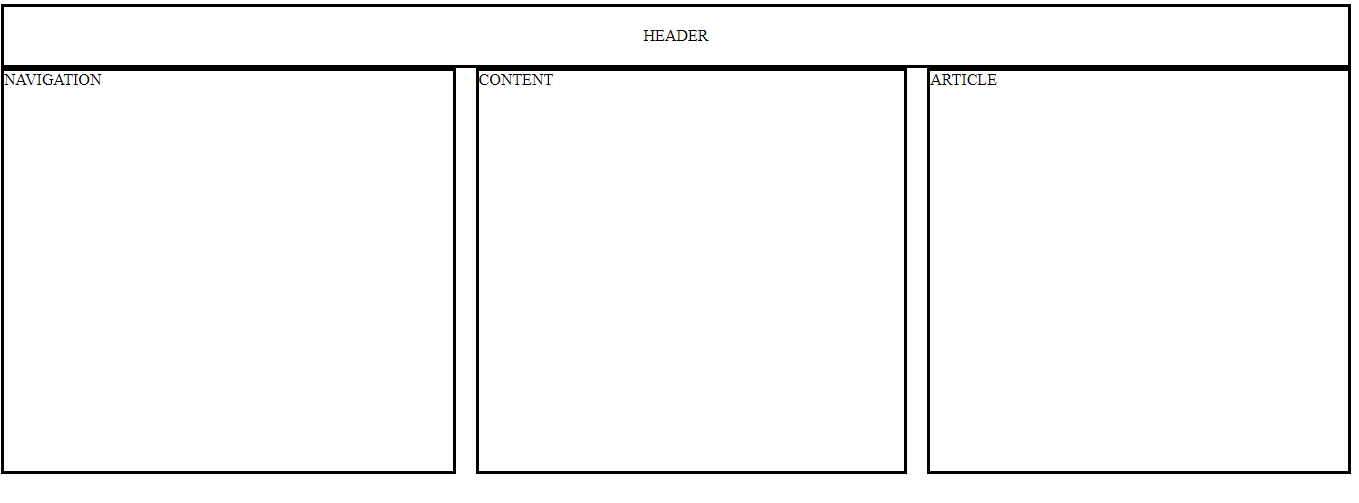
gridstyles.css



gridlayout.html



Output:



Activity: Try to reduce the width of navigation and article and increase width of content so that it looks like a property layout.

JavaScript

It is a scripting language used to write client & server side programs, you can use JavaScript at the client side for following operations:-

* Accessing HTML & CSS at runtime
* Client side validations
* Programming constructs like loops, conditions, functions, vairables
* Event Handling

The client side script is run by browser

You can use JavaScript at the server side for following operations

1. Creating webservices
2. Interacting with filesystems
3. Interacting with databases
4. Implementing business logics and so on

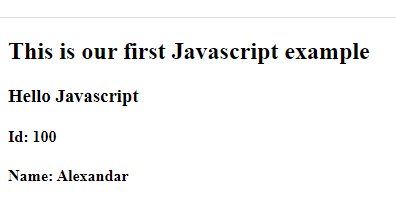
The server side script is run by node.js

You can write JavaScript inside HTML file or you can also write in a separate script file with .js extension and include in the HTML, if browser runs the HTML it also runs the script.

first.html



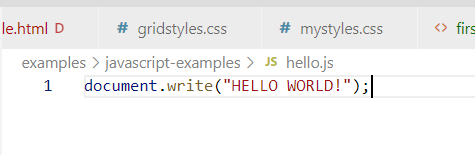
Output:



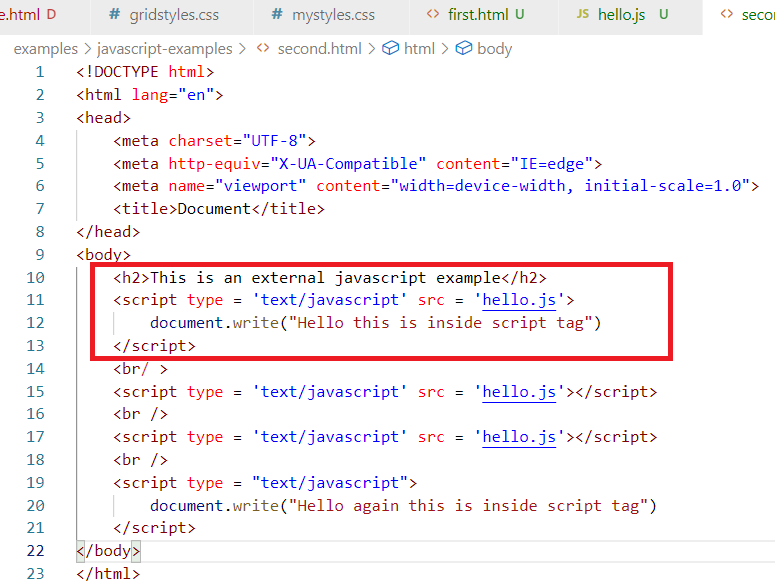
Creating external JavaScript file

You need to keep the file with .js extension & include it in the HTML file using <script>

hello.js



second.html



Note: Whenever the <script> includes the external Javascript file, the same script must now have Javascript code, if you want a javascript code inside a <script> tag then write it without ‘src’

Where can we add the <script> tag

You can add either in <body> or <head> or both, but the scripts added in the <head> is executed first and script inside <body> is executed at the time loading the content.

Note: If your script is accessing any HTML elements better keep script at the end of the <body>

Fundamentals of JavaScript

Variables

Operators

Conditional statements

Loops

Arrays

Functions

Objects

Variables: Variables are memory to store the data, you can store any kind of data like string, number, boolean, object and so on

Operators: These are used to do some operations on the variables or values for calculations, below are the Operators we have in JavaScript

=, +, -, \*, /, %, <, >, !=, <=, >=, ==, ===, ++, --

a = 10;

b = 20;

c = a \* b; // value of c will be 200

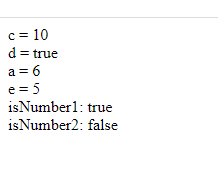
operators.js



operatorsDemo.html



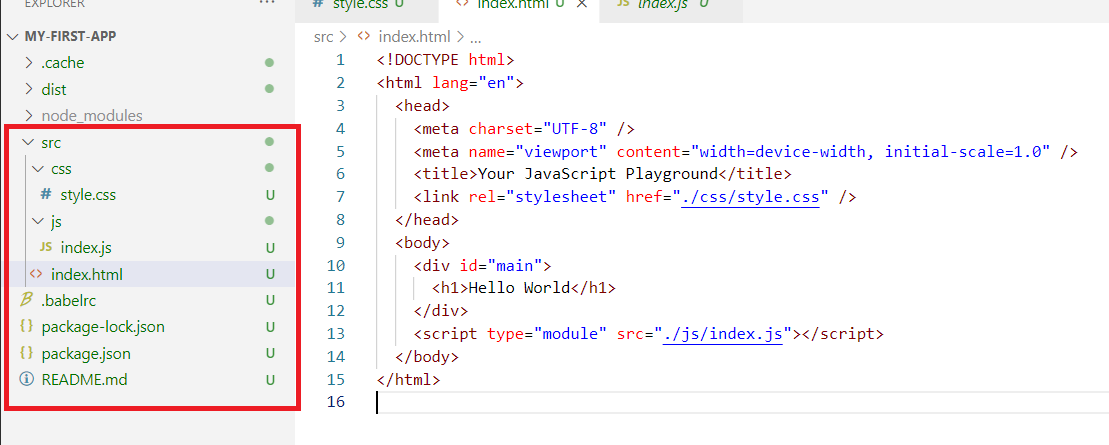
Output:



Creating a blank project that will have proper project structure with src, css folders and a live reloading feature

npx @knowledgehut/create-js-app app-name

When you create a project with the name my-first-app you can see the project folder as below:



Here the only file that will be loaded by the server is index.html

document: It is an inbuilt object in Javascript that helps you to work with DOM (Document Object Model) so that you can access HTML elements, this DOM provides below functions:

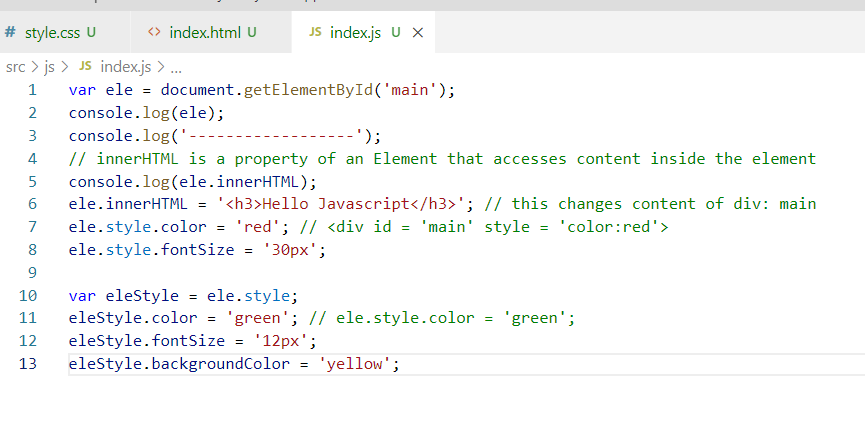
write()

getElementById()

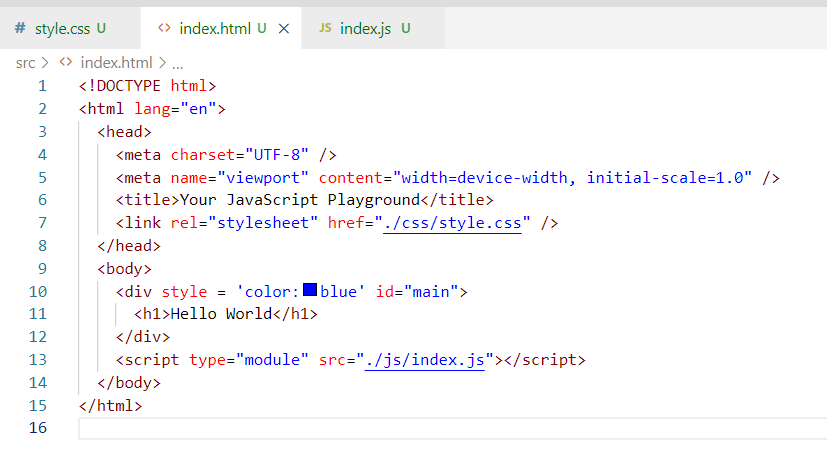
getElementsByTagName()

getElementsByClassName()

index.js

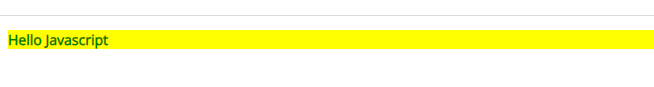


index.html



Though <div id = ‘main’> has style = ‘color:blue’ the script will change the font style to green & background color to yellow

Output:

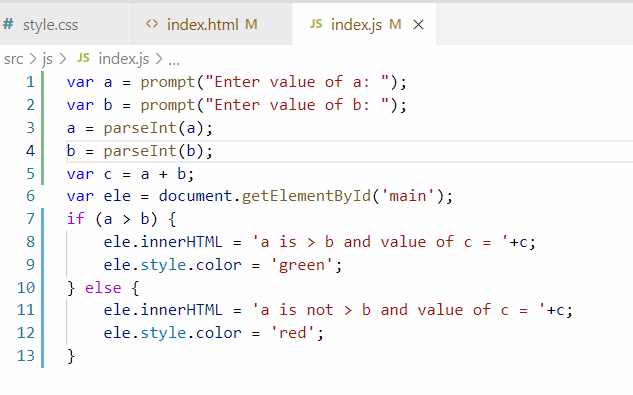


Javascript Conditional Blocks

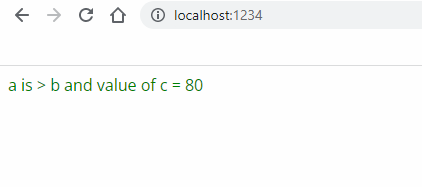
Conditional blocks are the code that are executed when the condition is true/false

* if
* if else
* if else if else if…. else
* switch

Applying if & else



Output:



Functions in Javascript

These are block of codes that can be called via function name, functions can be called whenever certain event occurs in the web page

Syntax:

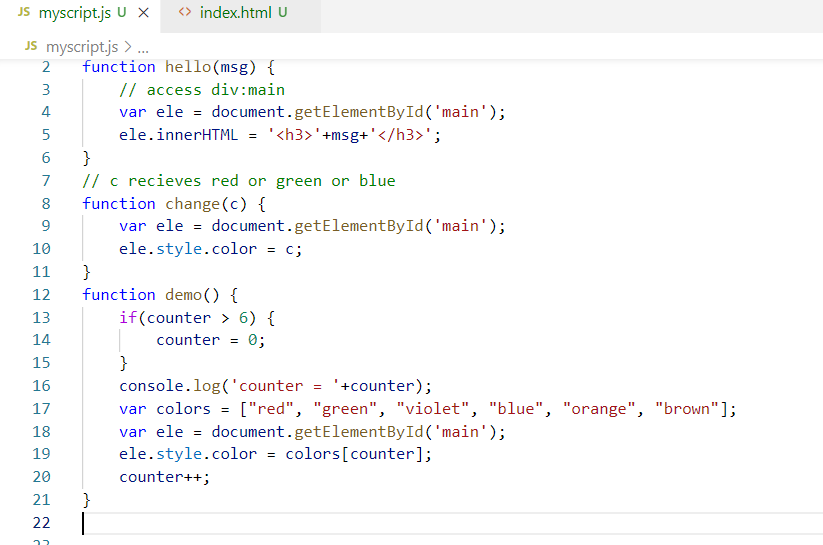
function functionName(arg1, arg2,…) {   
 // set of executable statements.  
}

Note: arg1, arg2 are the arguments or parameters the function take so that you can call the function by passing the arguments, they are optional.

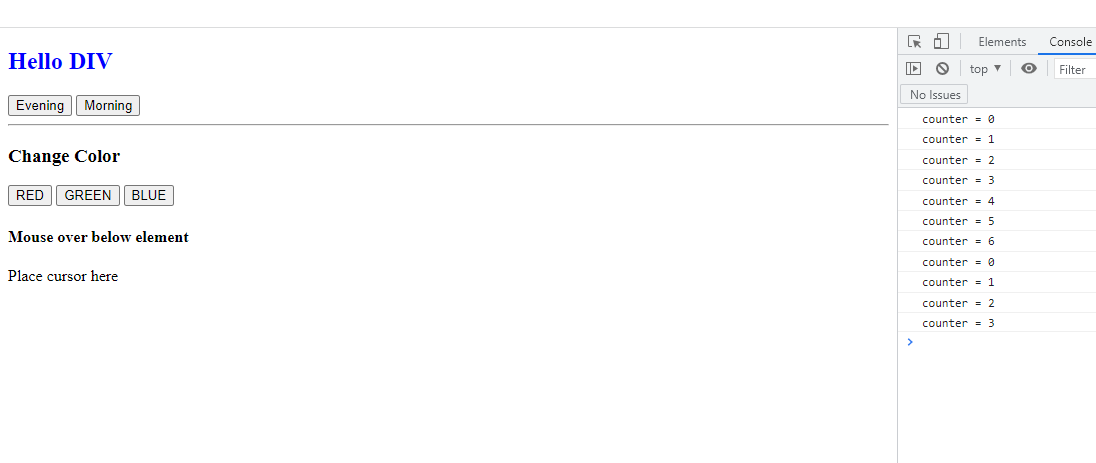
index.html



myscripts.js



Output:

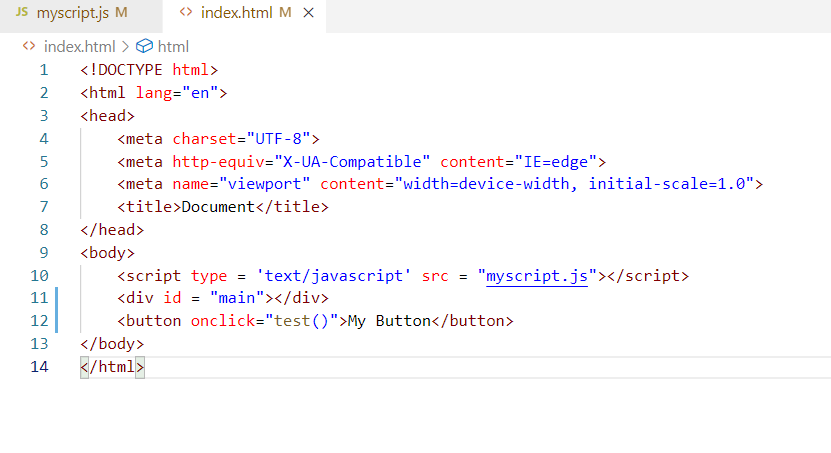


var: It is used to create variables, but it is global means its available even if you declare inside the block

let & const: let & const both are block scoped, let can be modified, however const can’t be modified



index.html

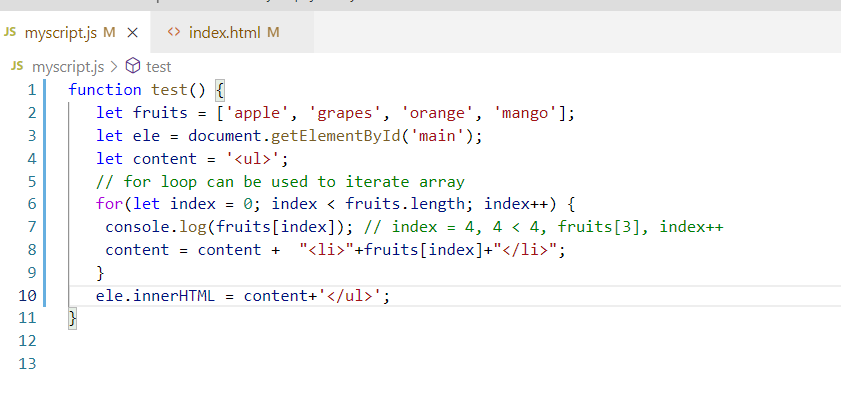


Output:

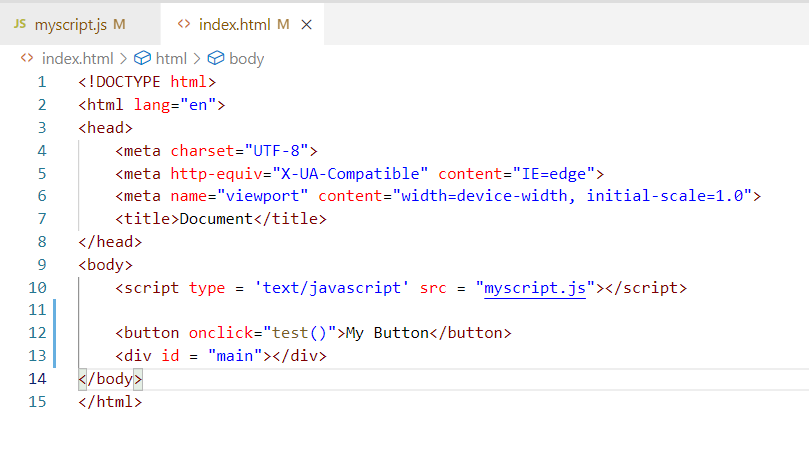


for loop, while, do while loop

myscripts.js



index.html



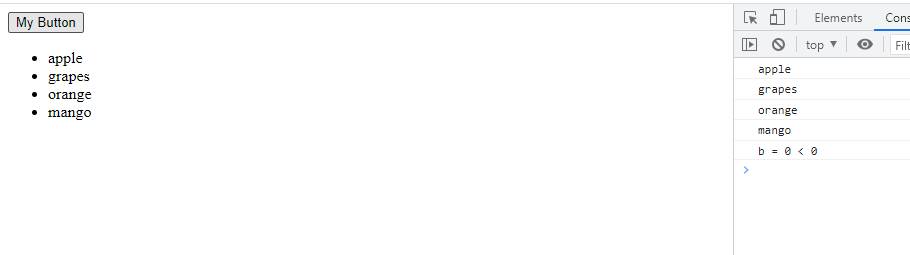
Output:



while & do-while



Output:



Activity:

Try document.getElementsByTagName() and document.getElementsByClassName() and print their content in the console and see how it works.

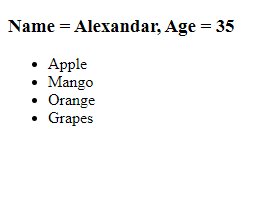
ECMAScript or ES6+: It is a standard for Javascript, Any rules ECMAScript defines JavaScript and other Javascript technologies must implement like Browser, Jscript and so on, ES specified many new features in JavaScript to improve the Javascript syntax

* let, const,
* class, extends, super, static
* Template string literals
* Arrow functions
* Rest & Spread operators
* Object & array destructuring
* Object.entries(), Object.values()
* Array.includes()
* Optional Chain
* Exponential operator
* Default parameter

Template String literals: This allows you to easily create strings and concatenate strings without breaking the strings with + operator, for this you must use back tick character to enclose strings (`), you must use ${data} to embed the data in a string



Output:



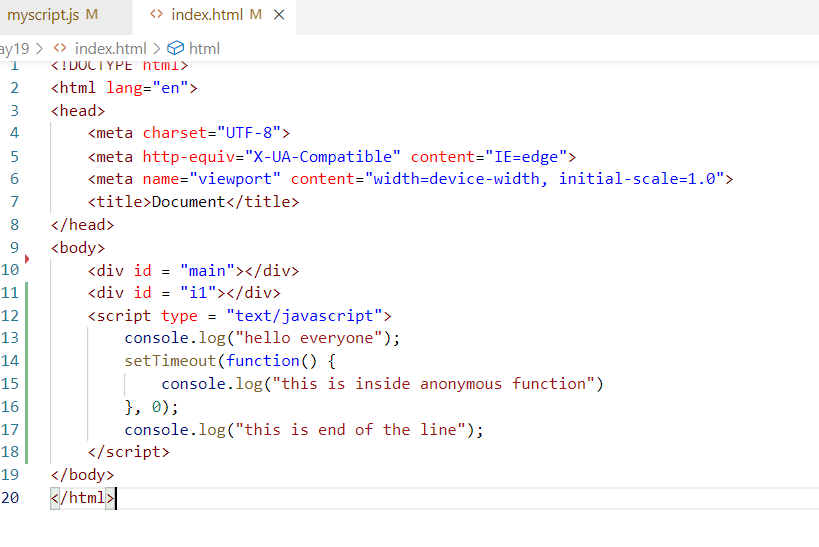
Anonymous functions:

These are the functions which doesn’t have names, but they are used only when a function is useful one time or may not want to be reusable.

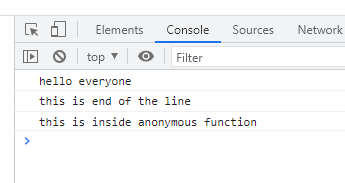
syntax:

function() {   
  
}

Ex: You can use this as a callback function, callbacks are the functions which are executed after some time



Output:



Here the setTimeout() takes 2 arguments, 1st argument is a callback that is executed after the timer is over, the 2nd argument is the timer.

The callback is a function that has no name, which is called as anonymous function, these callbacks are asynchronous in nature, they don’t wait or block other statements of script

Arrow functions: This is an alternate way of writing anonymous function, you don’t have to write a function keyword and {} if anonymous function is one line

() => {}

The above code is an alternate way for writing below anonymous function

function() {   
  
}

Suppose you write below anonymous function

function() {  
 return “hello world”;  
}

Arrow function for the above one is

() => “hello world”;

Another way to write the arrow function for above anonymous function

() => { return “hello world”; }

Note: return keyword is necessary in arrow function only if you use {}, you can use {} only if you have more than one line statements.

Understanding of arrow functions

(x, y) => x + y;

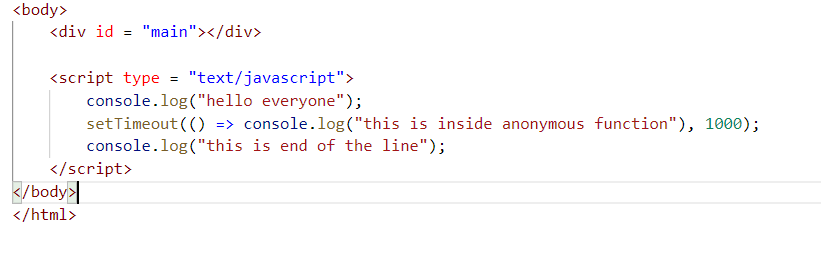
The above one takes two arguments & returns addition of 2 arguments

(x, y) => “hello”;

The above one takes two arguments & returns string

(x, y) => console.log(x, y);

The above one takes two arguments but doesn’t return any thing but prints x & y



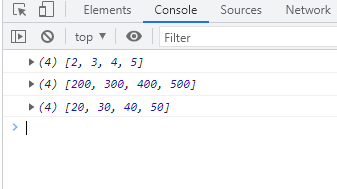
Note: The above code takes arrow function as an argument in the setTimeout()

Some uses cases where you can use anonymous or arrow functions

map(): It is a function that can iterate the array and return a new array, it takes anonymous function as an argument with 2 parameters, 1st one is the iterated item & 2nd one is the index.

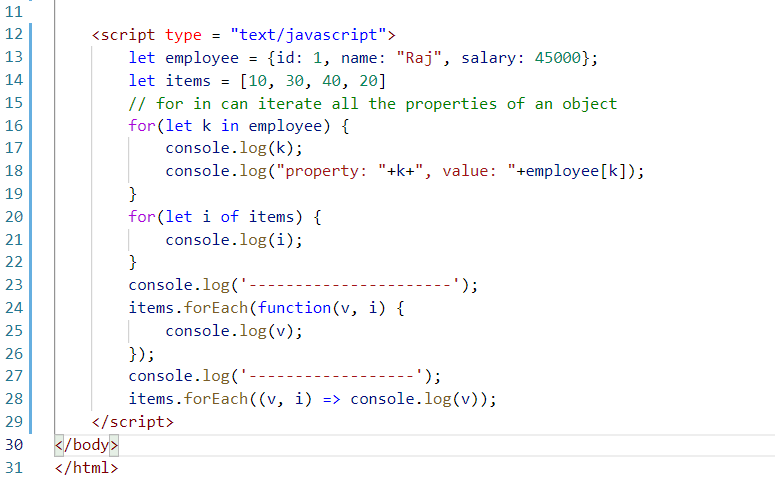


Output:

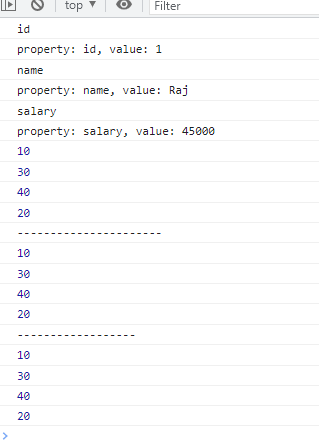


Types of for loops

* for: This is old for loop uses condition & counter
* for in: This is used to iterate the properties/keys of the object
* for of: This is used to iterate the arrays same like old for loop
* forEach: This is used to iterate the arrays same like old for loop but through anonymous function



Output:



for in vs for of vs forEach

|  |  |  |
| --- | --- | --- |
| For Of | For In | For Each |
| Iterates the array and doesn’t provide index access | It is to enumerate/get the object properties | Iterates the array and iterated variables are part of the forEach block, it provides access to index as well |

map vs forEach

map & forEach both iterates the array and both give access to index, but map returns a new array & forEach doesn’t

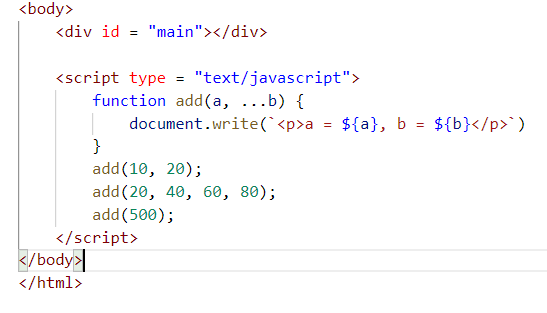
Rest & Spread operators:

Rest: It is a parameter to a function that can accept 0 or ore arguments

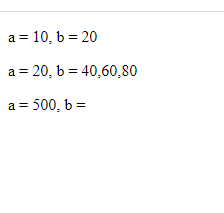
function abc(x, …y) { }

Now you can call abc by passing any number of arguments, but 1 argument is mandatory for x, and y can be empty or any number of arguments

i.e., abc(1); abc(20, 30); abc(20, 30, 40, 50)

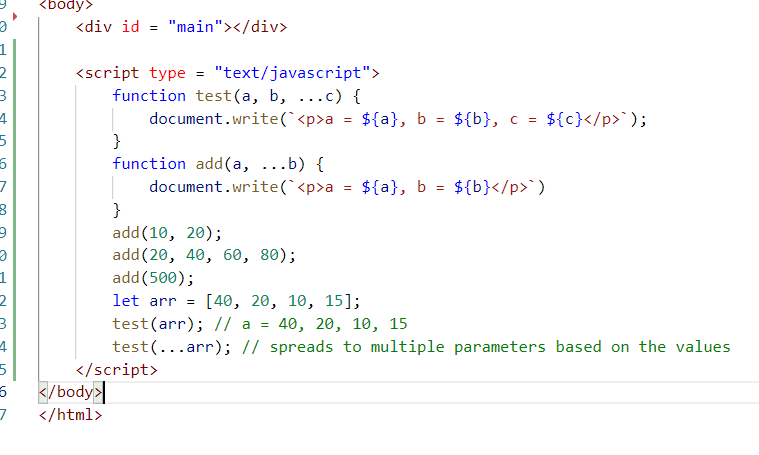


Output:



Note: Rest parameter acts like array, you can iterate it using for loop

Spread operator: It is used when you want to distribute arguments to the function parameters while calling



Output:

