Day 2

01-03-2022

Object declaration

In Java We can take the value through keyboards lot of ways

1. Scanner class.

Scanner is a pre-defined class part of util package. Package is a collection classes and interfaces.

Syntax to create the Scanner class object.

Scanner obj = new Scanner(System.in);

Day 3

02-03-2022

OOPS Concept

object : any real word entity.

Properties or state ---🡪 have

Person

Behaviour ---🡪do/does

Place

Bank

Animal

Car

Employee

Customer

class : blue print of object or template of object or user-defined data type which help describe the object.

method or function : it is use to write set of instruction to perform a specific task.

Syntax

returnType methodName(parameterList) {

}

void info() { void means to return type.

}

int getNumber() {

coding……

return 100;

}

int add(int a, int b) {

}

Syntax to create the class object.

ClassName refereceName = new ClassName();

Types of variable or fields.

1. Instance variable:
2. The variable which declared inside a class but outside a method is known as instance variable.
3. It hold default value with respective their data types : int family 🡪0, float 🡪 0.0, char 🡪 space, String null, and boolean false.
4. Instance variable we can use directly inside a all methods but method must be part of same class and it must be non static.
5. Local variable
   1. The variable which declared inside a method is known as local variable.
   2. It hold doesn’t hold default value.
   3. Scope within that method where it declared.
6. Static variable

Constructor :

Constructor is special method which help to create the object.

Points

1. Constructor have same name as class itself.
2. Constructor no return type not even void also.
3. Constructor no need to call it will call automatically when we create the object.

If we write any constructor by default empty constructor is present.

If we want we can write more than one constructor.

Encapsulation : Binding or wrapping data and code in a single unit is known as Encapsulation.

Class

**JavaBean class or POJO class(Plain Old Java Object)**

For all variable must be private and for each variable we have to provide setter and getter methods.

Setter to set the value and getter to get the value.

When we display the reference in println internally it will call toString() method of object class.

That method it return the output as [packageName.className@coede](mailto:packageName.className@coede)

So if you want meaningful output we have to override toString() method in user defined class.

Inheritance : Inheritance is use to inherits or acquire properties and behaviour of old class to new class.

class A {

}

class B extends A {

}

Types of inheritance

1. Single inheritance

Class A { }

Class B extends A{}

1. Multilevel inheritance

Class A { }

Class B extends A { }

Class C extends B{ }

Class D extends D{ }

1. Hierarchical inheritance

Class A { }

Class B extends A{ }

Class C extends A{ }

1. Multiple inheritance

Class A { }

Class B { }

Class C extends A,B{ } Wrong in Java. Java doesn’t support this type of inheritance.

It support using interface.

OOPs relationship

1. Manager/Developer Is a relationship Employee
2. Has a relationship

abstract class Employee {

id,name,salary

}

class Manager extends Employee {

numberOfEmp;

Address add = new Address();

}

class Developer extends Employee{

techName;

}

class ProjectManager extends Manager{

}

class Address {

city,state etc.

}

Has a relationship

Association

class A {

B obj = new B(); 0, 1 or many

}

class B {

A obj1 = new A(); 0, 1 or many

}

Aggregation : but is known as weak association.

class Manager {

Address ladd = new Address();

Address padd = new Address();

}

class Address {

}

Composition it is known as strong association : composition.

class Student {

StudentHistory sh = new StudentHistory();

}

class StudentHistory {

}

Polymorphism

One name many forms.

Compile time polymorphism or static binding or early binding

Method overloading

Run time polymorphism or late binding or dynamic binding

Method overriding

abstract keyword

abstract method

abstract class

Inheritance Example

**package** com;

**class** Bike {

**void** speed() {

System.***out***.println("60km/hr");

}

}

**class** Honda **extends** Bike {

**void** color() {

System.***out***.println("Gray");

}

}

**class** Pulsar **extends** Bike {

**void** color() {

System.***out***.println("Black");

}

}

**public** **class** InheritanceDemoWithAbstrat {

**public** **static** **void** main(String[] args) {

Honda hh = **new** Honda(); hh.color(); hh.speed();

Pulsar pu = **new** Pulsar(); pu.color(); pu.speed();

}

}

Method overriding

**package** com;

**class** Bike {

**void** speed() {

System.***out***.println("60km/hr");

}

}

**class** Honda **extends** Bike {

**void** color() {

System.***out***.println("Gray");

}

}

**class** Pulsar **extends** Bike {

**void** speed() {

System.***out***.println("90km/hr");

}

**void** color() {

System.***out***.println("Black");

}

}

**public** **class** InheritanceDemoWithAbstrat {

**public** **static** **void** main(String[] args) {

Honda hh = **new** Honda(); hh.color(); hh.speed();

Pulsar pu = **new** Pulsar(); pu.color(); pu.speed();

}

}

**abstract class example**

**package** com;

**abstract** **class** Bike {

**abstract** **void** speed();

**void** mailage() {

System.***out***.println("50km/lt");

}

}

**abstract** **class** Honda **extends** Bike {

**void** color() {

System.***out***.println("Gray");

}

}

**class** Pulsar **extends** Bike {

**void** speed() {

System.***out***.println("90km/hr");

}

**void** color() {

System.***out***.println("Black");

}

}

**public** **class** InheritanceDemoWithAbstrat {

**public** **static** **void** main(String[] args) {

//Honda hh = new Honda(); hh.color(); hh.speed();

Pulsar pu = **new** Pulsar(); pu.color(); pu.speed();

}

}

Final keyword

final variable : if variable is final we can’t change the value of the variable.

final int A=10;

A=20; Error

final method : if method is final we can’t override that method.

final class : if class is final we can’t extends that class.

static keyword.

interface

class extends only one class

interface extends more than one interface

class implements more than one interface

interface can’t extends or implements class

interface is known as 100% pure abstract class.

interface Abc {

fields ; public static final int A=10;

methods public abstract void dis1();

}

interface Mno {

int B=20;

void dis2();

}

interface Xyz extends Abc,Mno{

int C=30;

void dis3();

}

class Demo implements Abc,Mno {

public void dis1() { }

public void dis2() { }

}

08-03-2022

**Exception Handling**

**Java**

**Compile time error run time error**

**Syntax error**

**Run time error**

**Error Exception**

**Both are pre-defined classes part of lang package.**

**The error which generate at run time which we can’t handle it .**

**JVC crash or software or hardware issue.**

**It is a type of run time error which we can handle it.**

**Divided by zero.**

**Exception pre-defined class**

**Checked exception Unchecked exception**

**RuntimeException**

**SQLException ArithmeticException**

**IOException NumberFormatException**

**ClassNotFoundException**

5

try

catch

finally

throw

throws

Try and catch example

Custom Exception

throw vs throws

try and catch

throw keyword is use to raise or generate pre-defined or user-defined exception base upon the conditions.

Syntax

throw new Exception ()

or

throw new ExceptionSubClass();

throws keyword is use to throw the exception to caller method.

void display() throws Exception, ExceptionSubClass {

}

Collection Framework

int a=10;

int abc[];

class Employee {

id,name,salary

}

Employee emp = new Employee();

emp.id=100;

emp.name=”Ravi”;

emp.salary = 12000;

array object

Employee employees[]=new Employee[10];

Collection Framework :

Collection --🡪 interface

Set, List, Queue and Map -🡪 interfaces

Set, List, Queue internally extends Collection but Map doesn’t extends Collection.

Set : Doesn’t allow duplicate. Set classes can be order, unorder and sorted.

HashSet, LinkedHashSet, TreeSet

List : allow duplicate and it maintain the order.

Stack, ArrayList and LinkedList

Queue : First in First Out

PriortyQueue

Map : help to store information in the form of key value pairs key is unique value may be duplicate.

HashMap, LinkedHashMap, TreeMap and Hashtable

Iterator

Iterable

HashMap : it is a type of Map API. It allow to store information in key value pairs.

Key must be unique.

HashSet : it is a type of Set API. It allow to store only value. Value must unique

for(Map.Entry entry: hm.entrySet())…..is this same or different?

Generics Overview

How to access elements from nested HashMap

Heap Memory

Map

HashMap

Generic wild card

Multithreading

Program : test of instruction to perform a specific task.

Process : program in execution or time taken to execute the code.

Processor : responsible to execute the code.

Thread : small execution of a code within a process.

Multi tasking

1. Process base
2. Thread base

We can create thread using two ways

1. Extends Thread
2. Implements Runnable interface.

Concurrent API

package

git

graded project

multithreading.

Synchronization : it is a concept which help to lock or block the thread. It allow only one thread to use all resources at time. To achieve synchronization we have to use synchronized keyword with method or we can use synchronized block inside method.

wait(), notify() and notifyAll()

wait : this method is use to suspend the thread.

Notify : this method is use to resume the thread.

1. More than one thread created in same memory
2. The method must be synchronized
3. These methods are part of Objet class.

deadlock

volatile

nextLine()

week 3 contents.

Lambda Java 8 Features

IO package.

Io package (input and output operation).

Stream : flow of data or it is a abstraction between source and destination.

System.in

System.out

System.err

System.out.println(“Welcome”);

Scanner obj = new Scanner(System.in);

System is a pre-defined class part of lang package. Out is reference of PrintStream class and it is a static reference.

PrintStream ps = System.out;

ps.println(“Welcome to Java”);

System.out.println(“Welcome to Java.”);

PrintStream is a pre-defined class always refer to standard output device ie console.

System.in is consider as Standard input device reference ie keyboard.

InputStream : standard output device ie keyboard.

Scanner sc = new Scanner(System.in);

Stream

byte char

1 byte 2 byte

Input Output input Output

InputStream OutputStream Reader Writer

DataInputStream DataOutputStream InputStreamReader OutputStreamWriter

FileInputStream FileOutputStream FileReader FileWriter

BufferedInputStream BufferedOutputStream BufferedReader BufferedWriter

ObjectInputStream ObjectOutputStream PrintWriter

PrintStream

Object Serialization : storing the object into external file or converting into byte format is known as object serialization.

Object

Property : variable store only property

Behaviour : methods not store

Identity : reference name not store

Java 8 Features

From Java8 onwards interface can contains method with body. But method must be default or static.

**package** com;

**interface** Abc {

**void** dis1(); // by default public and abstract consider

**public** **default** **void** dis2() {

System.***out***.println("dis2 is a Abc interface default method");

}

**public** **default** **void** dis3() {

System.***out***.println("dis3 is a Abc interface default method");

}

**public** **static** **void** dis4() {

System.***out***.println("dis4 is a Abc interface static method");

}

}

**class** Xyz **implements** Abc {

@Override

**public** **void** dis1() {

System.***out***.println("For Abc interface dis1() method body providedd by Xyz class");

}

@Override

**public** **void** dis2() {

System.***out***.println("Xyz class overrided dis2() default method");

}

//@Override

**public** **static** **void** dis4() {

System.***out***.println("It is static Xyz own method");

}

}

**public** **class** Java8InterfaceExample {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

Xyz obj = **new** Xyz();

obj.dis1(); // overrided method

obj.dis2(); // overrided but optional

obj.dis3(); // default method

obj.*dis4*(); // calling Xyz class methodd

Abc.*dis4*(); // calling interface method

}

}

Functional interface :the interface which contains only one abstract method is known as functional interface. It can contains more than default as well as static but only one abstract method is known as functional interface.

Marker interface : The interface contains zero method or no method is known as marker interface.

Example : Serializable.

Inner classes : class within another class is known as inner class.

1. Non static inner class
2. Static inner class
3. Anonymous inner class

**package** com;

**class** Outer {

**int** a;

**void** dis1() {

System.***out***.println("Dis1 is a outer class method");

}

**class** Inner1 {

**void** dis2() {

System.***out***.println("Dis2 is a inner non static class method");

}

}

**static** **class** Inner2 {

**void** dis3() {

System.***out***.println("Dis3 is a inner static class method");

}

}

}

**public** **class** InnerClassExample {

**public** **static** **void** main(String[] args) {

Outer out = **new** Outer();

out.dis1();

// 1st approach

//Inner1 obj = new Inner1();

Outer.Inner1 in1 = out.**new** Inner1();

in1.dis2();

//2nd approach

Outer.Inner1 in2 = **new** Outer().**new** Inner1();

in2.dis2();

Outer.Inner2 in3 = **new** Outer.Inner2();

in3.dis3();

}

}

Lambda Expression : It is as Greek word. Using Lambda expression we can do functional programming in java.

Lambda expression is known as anonymous function or methods.

Lambda expression we can use with functional interface.

**package** com;

**interface** A {

**void** dis1();

//void dis2();

}

**class** B **implements** A {

**public** **void** dis1() {

System.***out***.println("B class provided body for dis1() method");

}

}

**public** **class** AnonymousInnerClassExample {

**public** **static** **void** main(String[] args) {

//1st approach

A obj1 = **new** B();

obj1.dis1();

//2nd appoach

A obj2 = **new** A() {

**public** **void** dis1() {

System.***out***.println("A interface method override by anonymous inner class - Logic 1 ");

}

};

obj2.dis1();

A obj3 = **new** A() {

**public** **void** dis1() {

System.***out***.println("A interface method override by anonymous inner class - Logic 2");

}

};

obj3.dis1();

//3rd approach

A obj4 = ()->System.***out***.println("This is lambda expression");

obj4.dis1();

}

}

Lambda Expression :

Lambda expression it return the value without return keyword.

Stream API

Pre-defined functional interfaces.

Stream API is a part of functional package and it is sub package of util package.

Collection Framework (Data Structure). If collection framework hold huge data of type integer,

Float, double, or user defined class object. if we want to apply any business logic first we have to load all data one by one using loop or iterator and then we have apply business rules.

Stream is use to load the data on demand. Stream doesn’t hold the data permanently.

Source data stream

Array

Or stream -🡪 IO1-🡪IO2-🡪O3---🡪 Terminal operator.

Collection

Intermediate operator return type is stream itself.

Terminal operator return type is non stream means void or any primitive type.

Function

apply() : Takes T is a parameter and return R value.

Predicate

test() : Take T is a parameter and return boolean value

Consumer

accept() : It take T parameter but no return type.

Supplier

get() : no parameter but return T value.

**16-03-2022**

Git

Version Control system tool : version control system tool is use to record the changes done in a project.

Local version control system : RCS : Revision control system

Centralized version control system : SVN all client machine connect to server machine once they do coding they push this code to remote repository (network folder).

Distributed version control system : local repository as well as remote repository.

Git is a open source Distributed version control system.

Which is developed using unix.

Open the git bash inside a folder

git --version

git init This command is use to make the local folder as a local repository.

git status : This command is use to check the current status of local repository

git add filename : This command is use to add the file from local file system to staging area.

Staging area : it is area provided by git where all files and folder store before commit.

git branch : it is like a pointer which hold more than one commit details.

Git by default one branch created it may be master or main.

The command to check the branch

git branch

syntax to create the user-defined branch

git branch branchname

to switch to branch

git checkout branchname

merge code from one branch to another branch

git merge branchName ( this command to merge branch code to current branch)

to link local repository to remote repository

git remote add origin URL

Week 3

RDBMS Using My SQL

JDBC

Hibernate

To download the existing remote repository in local machine you have run the command as

git clone URL

if you want to get new updated in existing repository you have to run the command as

git pull

MySQL

File System we done using IO package

1. Data redundancy (duplicate records).
2. Data consistency (format of the file or delimiter between two records).
3. CRUD Operation (Create, Read, Update and Delete).
4. Security

Data : raw fact

Information : meaningful data or processed data.

Database : storing the data in table format.

DBMS : Database Management system : it is software which help to store the data in table format.

Excel :

TrainerStudentDetails

TId TName Tech SId SName Age

1 Raj Java 100 Seeta 21

1 Raj Java 101 Meeta 22

1 Raj Java 102 Reeta 23

Dr. EF codd’s Rules 0 to 11 rules

MySQL

Oracle

Db2

SQL Server 2021

RDBMS

SQL : Structure Query Language

5 types

DRL or DQL : select clause (Data query language

DDL :Data definitional language (create, alter, drop, rename, truncate)

DML : Data manipulation language : insert, delete, and update

TCL : transactional control language (commit, rollback and save point)

DCL : granted and revoke (Data control language).

Show databases

Create database databaseName

use databaseName

show tables;

DDL

create table tableName(columnName datatype,columnName datatype);

desc sample

DML :

Insert

insert into sample values(1,'Ravi');

Delete

delete from tableName; it delete all records.

delete from sample where srno=1

Delete from sample where name =’Ravi’;

Update

update tableName set columnName = value

update sample set name = ‘Raj’;

update sample set name='Ajay Kumar' where srno=3;

DRL

select \* from tableName

Keys

Candidate keys

FK

PK

Cid,CName,Age,PhNumber,Accno,Amont,TypeOfAccount,PId,PName,Price,MgrId,Name etc

**Keys** : A key is an attribute (column or field) or combination of more than one column which is use to identify records.

Super key : an attribute or a combination of attribute that is used to identify the record uniquely.

Cid

PhNumber

AccNo

PId

MgrId

Cid,CName

Cid,PhNumber

typeOfAccount

Cid,AccNumber

Etc

Candidate key it is defined as minimal super key or irreducible super key is known as Candidate key.

Primary key : primary key is type of candidate key. Single table we can create only one column as pk. Pk doesn’t allow duplicate as well as null value

Unique key : in single table we can create more than one column as pk. It allow null value but doesn’t allow duplicate.

FK : FK is use to refer to pk of same table or different table. Fk allow only do value which present in pk it allow null value.

reate table trainer(tid int, tname varchar(10) not null, tech varchar(10) unique not null,

primary key(tid));

create table student(sid int, sname varchar(10) not null, age int, tsid int,

primary key(sid),

foreign key(tsid) references trainer(tid));

TCL :

Sub Query

DCL

21-03-2022

JDBC : Java Database Connectivity

Maven : Maven is build tool responsible to download the dependencies.

DAO : Data Access Object : Pure database logic.

Limitation of JDBC

1. Using JDBC we can’t store as well as can’t retrieve object form database means we have to convert java object into sql and vice-versa.
2. Jdbc use sql language. SQL is database dependent language.
3. Jdbc always throw checked exception. So we have to handle it mandatory.
4. JDBC doesn’t support is a and has a relationship.

ORM : Object Relation Mapping

JavaBean or POJO or Entity Relation

class Employee { Employee

id,name,salary ID,Name,Salary

}

Mapping

Employee – Employee

Id -🡪ID PK  
 name🡪Name

Salary 🡪SALARY

Using xml file or annotation

ORM : Hibernate, JPA, iBaties etc.

JPA is a technologies part of JEE (Java Enterprise Edition).

Hibernate is framework.

JPA is known as specification and Hibernate is implementation.

Configuration details (database details

Drivername,url, username, password and dialects class)

Configuration details we can provide using xml or properties or java classes.

22-03-2022

SQL : Structured Query language : SQL is database dependent. It retrieve sql query

Select \* from employee (employee table sql is not a case sensitive)

Select name from employee;

Select id from employee;

Select id,name, from employee

Select \* from employee where salary > 12000;

HQL : Hibernate Query Language : HQL is database independent. It retrieve entity class object.

select emp from Employee emp (Employee is entity class name and it is a case sensitive).

Select emp.id from Employee emp

Select emp.name from Employee emp;

Select emp.name,emp.salary from Employee emp

Select emp from Employee emp where emp.salary > 12000

23-03-2022

Hibernate Relationship

4 types one : PK many : FK

One - to – one Person Passport

Pk fk

Pk pk (shared primary key)

One – to – many Trainer Student

Project Employees

Many – to –one Students Section

Employees Departments

Many – to – many Employees SkillSet

Employees

EmpId EmpName

1. Raj
2. Ravi
3. Ramesh

SkillSet

SkillSet SkillName

100 Java

101 Python

102 Angular

Employee\_SkillSet

1. 100
2. 101
3. 102
4. 100
5. 102

Using Hibernate we are store, delete and update using method but retrieve the records

The we are depending upon HQL.

Criteria : Criteria is hibernate API we can do select operation on entity using methods.

Select name from employee sql

Select emp.name from Employee emp HQL

Oracle Sequence

create sequence myseq;

insert into employee values(myseq.nextval,'Ajay',12000);

25-03-2022

No SQL Database

Limitation of RDBMS

Schema base database

Table 🡪Employee

Number of columns and data type for that columns

Employee

Id Name Salary age City

1 Raj 12000 null null

2 Ravi 14000 null null

3 Ajay 16000 21 null

4 Mahesh 18000 null Bangalore

Trainer

PK

TId TName Tech

1 Raj Java

2 Ravi Python

Student

SID FK

SId SName Age TsId

100 Meeta 21 1

101 Reeta 22 1

No SQL

Mongo DB

Neo4j

Cassandra

HBase

Mongo DB : Mongo DB is open source document base No SQL Database which is use to store the data in the form of json.

JSON is like a Map in the form key-value pairs. Key is unique and value may be duplicate.

Java Script Object Notation.

{“id”:100,”name”:”Ravi”,age:21};

C:\Program Files\MongoDB\Server\5.0\bin

In C Drive you have to create the folder as

data --🡪 inside data folder you have to create db folder.

db ----🡪

To run the mongo db server you have to run the command as

mongod : inside a bin folder

This command to start the service

Then open another command prompt inside a bin folder and run the command as

mongo

this terminal is use to run the mongo db commands.

In mongo terminal

Cntr + L

use databasename; if database present it will switch to existing database else it will create and switch to that database.

In Mongo DB table is known as collection and row is known as document.

In Mongo DB column is as a key.

Syntax to create the collection

db.createCollection("Sample");

syntax to insert the document in collection

db.Sample.insert({name:”Ravi”})

db.Sample.insert({name:”Ram”,age:”21”});

view the document from a collection.

db.Sample.find();

db.Employee.insert({\_id:1,name:"Ravi",age:21,city:"Bangalore"});

WriteResult({ "nInserted" : 1 })

> db.Employee.insert({\_id:2,name:"Ramemsh",age:24,city:"Delhi"});

WriteResult({ "nInserted" : 1 })

> db.Employee.insert({\_id:3,name:"Ajay",age:26,city:"Mumbai"});

WriteResult({ "nInserted" : 1 })

> db.Employee.insert({\_id:4,name:"Vijay",age:25,city:"Bangalore"});

WriteResult({ "nInserted" : 1 })

> db.Employee.find();

{ "\_id" : 1, "name" : "Ravi", "age" : 21, "city" : "Bangalore" }

{ "\_id" : 2, "name" : "Ramemsh", "age" : 24, "city" : "Delhi" }

{ "\_id" : 3, "name" : "Ajay", "age" : 26, "city" : "Mumbai" }

{ "\_id" : 4, "name" : "Vijay", "age" : 25, "city" : "Bangalore" }

Retrieve the document from a collection using index positon

> db.Employee.find()[0];

{ "\_id" : 1, "name" : "Ravi", "age" : 21, "city" : "Bangalore" }

> db.Employee.find()[1];

{ "\_id" : 2, "name" : "Ramemsh", "age" : 24, "city" : "Delhi" }

> db.Employee.find()[1].name;

Ramemsh

> db.Employee.find()[1].age;

24

Retrieve the document using condition

db.CollectionName.find({condition},{filterTheField});

db.Employee.find({\_id:1});

{ "\_id" : 1, "name" : "Ravi", "age" : 21, "city" : "Bangalore" }

> db.Employee.find({city:"Bangalore"});

{ "\_id" : 1, "name" : "Ravi", "age" : 21, "city" : "Bangalore" }

{ "\_id" : 4, "name" : "Vijay", "age" : 25, "city" : "Bangalore" }

> db.Employee.find({age:{$gt:25}});

{ "\_id" : 3, "name" : "Ajay", "age" : 26, "city" : "Mumbai" }

> db.Employee.find({age:{$gt:24}});

{ "\_id" : 3, "name" : "Ajay", "age" : 26, "city" : "Mumbai" }

{ "\_id" : 4, "name" : "Vijay", "age" : 25, "city" : "Bangalore" }

> db.Employee.find({age:{$lt:24}});

{ "\_id" : 1, "name" : "Ravi", "age" : 21, "city" : "Bangalore" }

> db.Employee.find({age:{$lte:24}});

{ "\_id" : 1, "name" : "Ravi", "age" : 21, "city" : "Bangalore" }

{ "\_id" : 2, "name" : "Ramemsh", "age" : 24, "city" : "Delhi" }

Retrieve specific fields from a documents

> db.Employee.find({},{name:1});

{ "\_id" : 1, "name" : "Ravi" }

{ "\_id" : 2, "name" : "Ramemsh" }

{ "\_id" : 3, "name" : "Ajay" }

{ "\_id" : 4, "name" : "Vijay" }

> db.Employee.find({},{name:1,\_id:0});

{ "name" : "Ravi" }

{ "name" : "Ramemsh" }

{ "name" : "Ajay" }

{ "name" : "Vijay" }

> db.Employee.find({},{name:1,\_id:0,age:1});

{ "name" : "Ravi", "age" : 21 }

{ "name" : "Ramemsh", "age" : 24 }

{ "name" : "Ajay", "age" : 26 }

{ "name" : "Vijay", "age" : 25 }

Update document using conditions

db.Employee.update({\_id:1},{$set:{age:22}}) : using \_id property

WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

db.Employee.updateMany({city:"Bangalore"},{$set:{city:"Mysore"}}) : using other property

delete query

db.Employee.find();

**28-03-2022**

Connection Mongo DB database using ORM.

Hibernate / JPA provided engine ie OGM (Object Grid Mapper) which help to connect Java application to No SQL Database.

JPA use to connect the mongo db database.

Hibernate JPA

Database details.

hibernate.cfg.xml persistence.xml

Week 5 Session Plan

HTML, CSS and bootstrap

Week 6 Session Plan

JavaScript

https://[www.google.com](http://www.google.com) ---🡪 URL : Uniform resource locator

http : protocol : hyper text transfer protocol : secure : set of rules which help to communicate more than one machine.

www : world wide web

google : domain

com : commercial application

req (http/ https)-------------------------------🡪

Client Server

🡨----------res(http/https)------------------------ html or html5

Css or css3

JS (JavaScript)

Html 🡪 it is use to display the content on browser.

Css 🡪 it is responsible to display the content in proper format.

JS -🡪 it provide features to do programming on web page.

Html : hyper text makup language which help to create the web page. Html provided lot of tag or elements which help to display content on browser in different format.

Syntax of tag

<tagName> opening tag

</tagName> closing tag

1. Html
2. Head
3. Body
4. Tittle
5. P
6. Heading : h1 to h6

VS code :

Break tag : br

<br>

<br/> self closing tag

Heading tags

H1 largest

H6 smallest

Attribute : attribute is use to describe the properties of a tag.

Attribute we have to use in opening tag in the form of key-value pairs.

Value can be in single quote or double quote or without quote.

Syntax

<tagName key1=”value1” key2=’value2’ key3=value3> </tagName>

Font tag : using this tag we can change the color, size and face (style).

Html 4.x version

<!DOCTYPE html public url="https://www.gasddfsafsf.dtd">

<html>

    <head>

        <title>This is my web page</title>

    </head>

    <body>

        <p>Welcome to My Web Page</p>

    </body>

</html>

Hyper link : This tag is use to connect one page to another page

1. External hyper link
2. Internal hyper link or bookmark

External hyper link

<a href=”pageName.html”>Text</a>

Internal hyper link or book mark

<a href=”#a1”>First Topic</a>

<a name=”a1”></a>

Adding image to web page

<img src=”imageName.jpeg/gif”/>

30-03-2022

List tag : This tags are use to display the items in proper format.

1. types

Un Order list

Order list

Definition list

Table Tag :

Employee Details

Id Name Salary

100 Ravi 12000

101 Ramesh 14000

102 Ajay 16000

Forms Tags

Form tag is use to send more than one data to server.

Till html4

<input type=”text/password/radio/checkbox/file/submit/reset/button”/>

From html5

<input type=”email/number/date/time/url/color/search”/>

CSS : Css provide lot of properties and values which help to apply formatting style for web page.

With the help of CSS we can do separation of concern.

Css mainly divided into three types.

1. inline css
2. internal css or embedded css
3. external css

inline css syntax

<tagName style=”property:value;property:value;”></tagName>

Internal css or embedded css

This tag we have to write in between head tag.

<style type=”text/css”>

Selector {property:value;property:value;}

</style>

1. Universal selector : \* {property:value}
2. Specific selector : tagName {property :value}
3. Local class selector tagName.className{property:value}
4. Global class selector .className{property:value}
5. Id selector #idName{property:value}

<p class=”abc” id=”a1”>first para</p>

<p id=”a2”>second para</p>

<p class=”abc” id=”a3”>third para</p>

<p class=”xyz” id=”a4”>fourth para</p>

<h1 class=”abc” id=”a5”>First heading </h1>

<h1 class=”xyz” id=”a6”>Second heading </h1>

Class : group of tags which have same name or different names.

Two tag we can give same class name but don’t give same id.

Using id selector we can make unique ness between two tags.

External CSS file

Flex layout or Flex box :

The flex layout module of css provide an efficient way to layout and align space of ay web project.

Even when the view port and element size id dynamic or unknown.

01-04-2022

Bootstrap is a open source css web framework which help to create responsive web application as well mobile application.

Container classes

Container

Container-fluid

Alert

Bootstrap grid layout : it allow up to 12 column across the page.

Grid layout arrange the html component in row and column format.

Each row by default divided into 12 columns.

Device size

XS Extra small <576px

SM Small >=576px

Md medium >=768px

Lg large >=992px

XL Extra large >1200px

Log :

Id Operation date time

1 stored 04-04-2022 8:10pm

2 view the record 04-04-2022 8.12pm

JavaScript

JavaScript was object based interpreter scripting language.

JavaScript provided lot of pre-defined or user-defined object.

Prototype based scripting language.

<script type=”text/JavaScript”> opening tag

document.write(“Welcome to JS”);

</script> closing tag

This script tag we can write in between head tag or body tag of web page.

We can write more than one script tag inside a html page.

In JavaScript till ES5 to declare the variable we use var keyword.

Operator : Arithmetic operator, logical operator, conditional operator, increment and decrement etc.

= == ===

If statement

If else

If else if

Switch statement

Looping

While loop

Do while

For loop

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 |
| 2 | 4 | 6 | 8 | 10 |  |
| 3 | 6 | 9 | 12 | 15 |  |
| 4 | 8 | 12 | 16 | 20 |  |

05-04-2022

function : function is use to write set of instruction to perform a specific task.

1. types of function
2. pre-defined function or global function
3. user-defined function

alert(“msg”) : this function is use to display the alert message.

Prompt() : this function is use to take the value through keyboards.

parseInt(); : this function is use to convert string to integer

parseFloat(); : this function is use to convert string to float

eval(); : this function is use to convert string to number.

Confirm() : this display the pop message. Which contains two button ok and cancel. If click ok it return true and if click cancel it return false.

do {

alert 1: add 2: sub

prompt : take the choice

using conversation convert it eval or paseInt

switch() {

case 1 : add two number

case 2 : sub two number

default : wrong choic e

}

Confirm : do you want to continue.

}while()

User defined function

Normal syntax to declare the user-defined function using function keywords.

Syntax

function funtionName() {

function body;

}

1. no passing parameter and no return type
2. passing parameter and no return type
3. passing parameter and return type
4. no passing parameter and return type.

Using ES5

Using ES6

ECMA Script

06-04-2022

EC6 features

From ES6 onwards we can declare the variable using var, let and const keywords.

Var

Using var we can re-declare same variable once again with same value or different value.

Using let we can’t do.

Let

Using var we can declare global scope. Using let we can declare local or block scope.

Const : we can’t change the value.

Types of function

1. normal function
2. expression style function
3. arrow style function
4. callback : passing function name or function body or function itself to another function as a parameter is known as callback.

Array concept : in JavaScript ES5 and ES6 we can use array concept.

In JavaScript we can store any type of values.

08-04-2022

Objects : object is any real word entity.

Properties or state or field

Object

Behaviour

Two types of object.

Pre-defined object

BOM : Brower Object Model

DOM : Document Object Model

Object properties

Behaviour

Object properties

Behaviour

Object property

Behaviour

Object

User defined object

Three ways

1. literal style : using ES5
2. function style : using ES5
3. class style : from ES6 onwards.

11-04-2022

Event : event is delegation model or it provide bridge between html and JavaScript code.

In JavaScript all event start with pre-fix on followed by event name.

onClick

onDblClick button

onMouseOver

onMouseOut img

onKeyUp

onKeyDown text field or password field

onSubmit submit

onFocus

onBlur text field or password field

onChange dropdown

onLoad

onUnload body tag

etc

listener : listener is type of function which help to listen the generate events.

BOM : Browser Object Model

Window is top most object in BOM hierarchy.

window.alert() or alert()

window.prompt() or window.prompt()

DOM : Document Object Model

document is a property of window object.

document.write(“Welcome to JS”);

window.document.write(“Welcome to JS”);

index.html

html

head body

title textNode : Sample web page p textNode : welcome

meta div

style h1

script

DOM API (Document object model application programming interface). Lot programming language Java, Python, C# and JavaScript provided dom API which help to read, write and update html content dynamically.

**12-04-2022**

**Synchronous and Asynchronous**

Statement

Synchronous statement execution

Statement1

Statement2

Statement3

Asynchronous statement execution

Statement1

Statement2

Statement3

Function call

Synchronous function call

Fun1();

Fun2()

Fun3();

Asynchronous function call

Fun1();

Fun2()

Fun3();

Send request to server

Synchronous

1st req

2nd req

3rd req

Asynchronous

1st req

2nd req

3rd req

Client Server

Basic asynchronous methods

Window object provided three methods

setTime()

setInterval()

clearInterval();

**JSON : JavaScript Object Notation**

**Syntax**

**{“key1”:value1,:”key2”:”Value2”}**

**JavaScript provided pre-defined object ie JSON**

JSON.stringify(); object / json data into string.

JSON.parse(); : string to json.

13-04-2022

**XMLHttp Requests**

**Promise**

**Fetch**

AJAX : Asynchronous JavaScript and XML

Using XMLHttpRequest. It is a pre-defined object. which help to make the request as Asynchronous.

obj.readyState

readyState

0 : not initialize yet

1 : ready to send the request to page or resource

2 : request sent.

3 : request processing

4 : means ready to give the response

Promise : promise is a pre-defined object provide by JavaScript which is use to handle asynchronous event of data.

Promise can be resolved and can be rejected.

User-defined

To load the data form the promise object we have to use then() and catch(). Then and catch take callback function as a parameter. Then will call if promise resolved and catch get call if promise rejected.

Pre-defined API with promise

Fetch() : it is pre-defined function which is use to send the request to backend technologies for get as well as store the data.

Fetch method return type is promise object.

Then and catch

14-04-2022

Node JS : Node JS is run time environment for JavaScript program or library or Framework.

Before Node JS JavaScript is known as Client side scripting language. But after node JS JavaScript is known as client side as well as server side scripting language.

Using Node Js we can create server programming language like in Java Servlet, JSP or Spring framework.

node --version

npm --version :node package manager :

node js provide lot of pre-defined module.

Json-server it is a type of external module which help run static json file as a server. Which help to read and write json data through fetch function.

npm install -g json-server

or

npm install json-server

create the file employee.json

{

"employees":[

{"id":1,"name":"Ravi","age":21},

{"id":2,"name":"Ramesh","age":25},

{"id":3,"name":"Rajesh","age":28}

]

}

Open the command prompt in that location in place of file present

json-server employee.json

<http://localhost:3000/employee>s

ES7 : await and async keyword.

Await and async is replacement of then and catch when function return promise object.

18-04-2022

Scope object

HTML5 features

If we want to share the data between two pages then we can take the help scope object ie sessionStorage and localStorage object.

Both are pre-defined part of JavaScript which provided set of methods which help to set the value, get the value and remove the value.

Set the value

localStorage.setItem(“key”,value);

sessionStorage.setItem(“key”,value);

Get the value

localStorage.getItem(“key”);

sessionStorage.getItem(“key”)

remove the value

localStorage.removeItem(“key”)

sessionStorage.removeItem(“key”);

if we store the value in session scope it will store till the application close or browser close or session scope.

Session : collection of request within a particular period of a time is known as session.

If we store the value in local storage it will store permanently. We have to remove explicitly using removeItem().

Using JavaScript we can do operation on DOM ie Read, Write and Update.

jQuery : it is external JavaScript library which provided lot of pre-defined which internally connected to each others to do DOM operation very easily.

Library is not a standard. When we develop the application using library we are responsible to follow the standard.

Angular is a open source Web framework which help to develop SPA (Single Page Application).

Framework : framework internally follow standard. Design pattern : best practise. The implementation of design pattern is taken care by framework. Framework internally taken care 60% to 70% task. But it is not final product. It is template or protocol which help to develop the application very easily.

Angular is a Framework . framework are heavy. Internally follow standard.

React is Library . light weighted. Doesn’t follow standard.

Angular JS : base html, css and JavaScript mainly ES5 features.

1.0 to 1.x version

To run this application Node JS not require.

Angular Framework :

Base upon html, css, JavaScript and typescript and Node JS

2.x 13.x

19-04-2022

Typescript : typescript is a type of scripting language which support all features of ES6 with data type concept.

Typescript also known as super set of JavaScript.

But browser doesn’t support typescript directly. We have to convert TS to JS. Using typescript transpiler we have to convert ts to js.

Node JS : Node JS is run time environment for JavaScript or library or framework. Before Node JS JavaScript is known as client side scripting language. But after node js JavaScript also known as Client side as well as server side scripting language.

Using server side JavaScript we can do file handling programming, creating web application, creating rest full web service, connection database it can be mysql or mongo db etc.

Before Node JS we were not able to run the JavaScript program without taking the help of html page.

After node js we can run the Javascript using node command on console or terminal without browser.

In node js program we can’t use BOM and DOM.

Frontend technologies Backend technologies

Html,css,js, bootstrap, Jquery JEE (Servlet, JSP and EJB)

Spring Framework

Asp.net

Php

Python

Node JS

MEAN and MERN

Mongo DB / My SQL Express JS Agular Node JS

React JS

npm install –g typescript

data types :

types script data types concept.

Syntax

let/var variableName:datatype;

let/var variableName:datatype=value;

20-04-2022

In JavaScript only function name must be match doesn’t matter number of parameter list and type of parameter list must match.

JavaScript by default can return any type of value as well as not mandatory to return the value.

Optional parameter

Optional parameter declaration must from right to lift. If we want to declare the optional parameter we can declare one parameter or more than one parameter. There must be mandatory parameter between two optional parameter.

Default initialization parameter

Same rules apply for default initialization. Rather than ? we have to assign the default value.

**Rest operator or parameter**

If we make array variable as a parameter we have to array mandatory may be array hold 0 value or 1 value or many value.

Syntax

…variableName:datatype[]

In function we can use only one rest parameter and it must be last parameter in function.

OOPs concept in typescript

In JavaScript we can create user-defined object using 3 ways

1. Literal style
2. Function style
3. Class style

ES5 and ES6 in JavaScript

Typescript with class style using data type concept.

To create the constructor in typescript as well as ES6 style javascript we have to create the function with name constructor.

In Typescript we can write only one constructor it can be empty or parameter.

Typescript support access specifiers like public and private.

In typescript we can use private or public access specifier with constructor parameter variable to make the variable as instance variable.

21-04-2022

Access specifiers

Scope : let and var scope

Interface : it is type of reference type or 100% pure abstract class.

Syntax to declare the interface

Interface interfaceName {

Fields;

Methods;

}

Interface MyInterface {

fieldname:datatype; // don’t initialize

methodName:returnType; // method or function without body

}

Node JS : Node JS is a runtime environment for JavaScript programs. Before Node JS JavaScript known as client side scripting language. But after node js JavaScript also known as client side as well as server side scripting language. Using Node js we can do all concept what thing we can do with other server side programming language.

|  |
| --- |
|  |
| What are SPA |
| Introduction to Angular Framework |

introdcution to Angular :

Angular is a framework. Is JavaScript framework. Open source framework. Help to create SPA application

Single page application.

It is a part of google organization. Base upon typescript. Version of angular 2 to 13.x

It follow MVC (Model View Controller)-🡪 I will explain later on

SPA

Index.html application.html

Hyperlink

Button it can submit or other button

Using Javascript code

It will load whole dom once again.

22-04-2022

|  |
| --- |
| Angular Building Blocks |
| Angular CLI and installation of Angular |
|  |
|  |
| CLI Commands |
| Angular Project Structure |
| understanding package.json |
| How does it work? |

Angular CLI( Command Line Interface). Which help create the angular project using ng (next generation) command.

npm install –g @angular/cli

to create the angular project we use command as

ng new project-name

do you want routing yes/no : no

styles 🡪 css

if it will ask policies details : then you can give y/n

cd project-name

ng serve (it is use to run the project )

After compiled project successfully it will give url

<http://localhost:4200> angular internally provide default web server and that server run on default port number 4200

src -🡪

app.component.html -🡪 template page

app.component.css 🡪 style sheet file

app.component.ts 🡪 component : it is use to control the view or part of the view on web page.

25-04-2022

ng new project-name

cd project-name

code . (this will open the project in vs code)

ng serve –o (after compiled the project 100% it will run on default browser automatically)

Angular is component based architecture framework. Component is uses to control the view or part of the view.



@component : it is known as decorator.

Attribute

selector : “app-root” <app-root></app-root>

templatUrl:”./app.component.html”

stylesUrl:[“./app.component.css”]

app.module.ts

@NgModule :

Module : module is collection of more than one component. Module is like a package in a java.

Declaration : in this section we have to provide all component declaration details.

Import : in section we have to provide details about pre-defined or user-defined module.

BrowserModule is responsible to send or render the data on web page.

Provider : this section mainly use to provide angular service class details.

Bootstrap : This attribute hold the details about parent component which load as first component.

LoginModule CustomerModule AccountModule

LoginCreate customerCreate AccountCreate

LoginDelete customerView AcountUpdate

LoginUpdate customeerUpdate AccountDelete

Etc

Data binding

ng new angular-data-binding

routing 🡪no

styling 🡪 css

26-04-2022

ng generate component component

or

ng g c component

Data Binding

Using data binding we can share the data between view (html) or template to component or ts file.

Two types of data binding

1. One data binding
2. String interpolation : component -------------------🡪View

{{}}

{{variableName}}

{{6+8}}

{{display()}}

{{sayHello(“Ravi”)}}

ng g c string –interpolation

1. Property binding : component --------------------🡪View

[]

lname:string =”patil”;

<input type=”text” value=”Kumar”/> pure html code

<input type=”text” value=”lname”/> the value itself is lname.

<input type=”text” [value]=”lname”/> value patil

<p [innerText]=”lname”></p>

ng g c property-binding

string interpolation : we can display the value of variable without using any tag.

In String interpolation always value consider as a string.

Property binding :to display the value we require dom (HTML tags).

In property binding we can use different data types.

1. Event binding : View --------------------------🡪 Component

()

Angular use same event provided by HTML and JavaScript only difference all event name start with pre-fix on followed by event name and using those event we can call JavaScript functions. In Angular all event remove by on pre fix and wrap by (). And using these event we can call TS function which is part of Typescript class and we can call without creating the object of that class.

JS Event Angular event

onClick (click)

onDblClick (dblclick)

onChange (change)

onSubmit (ngSubmit)

etc etc

ng g c event-binding

two way data binding

event binding with string interpolation or property binding.

Passing the value from template to component.

1. Using template reference

<input type=”text” #nameRef/>

nameRef is consider as text field reference using this reference we can get the value of text field in component side.

1. Two data binding

We can achieve two way data binding using ngModel pre-defined attribute.

If we do any changes on component side it will update on view and vice-versa.

[()]

Component contains some variable

age:number =21

view or template side

<input type=”text” name=”age” [(ngModel)]=”age”/>

ng g c two-way-binding

ngModel is a pre-defined attribute part of FormsModule. So we have to import FormsModule in app.module.ts file in import section to avoid error.

28-04-2022

ng new types-of-directives

In Angular using directive we can add extra behaviour or functionality of html or dom elements.

3 types of directive

1. Component directive : component is a type of directive which help to create the create the user-defined tag with help of selector with templateUrl.

@Component({

Selector : “my-tag”,

templateUrl:”./mypage.html ”

})

class MyComponent {

set of variable and function

}

1. Structure directive : using structure directive we can add or remove dom element from html page.

\*ngIf

\*ngFor

Using ngIf and ngFor we can use if condition and looping condition in html page.

ng g c structure-directive

1. Attribute directive : using attribute directive we can apply css effect for web page

ngStyle : it is a like a style attribute with inline css

ngClass : it is a like a class attribute with internal or external css.

ng g c attribute-directive

ng new angular-forms

29-04-2022

If we want to pass the value from view to component

1. Using template reference.
2. Using two – way data binding.

If we want to send group of values from view to component.

We can pass group of value from view to component using angular forms.

Angular support two types of forms

1. Template driven form

The flow of the application from view to component.

This type of form is easy to create. The people from html and css background they prefer this type of form.

It is good for simple form.

In this form we uses ngForm and ngModel attribute. These two are pre-defined attribute we have to use in view side. These attribute are part of FormsModule. So while creating form using template driven form we have to import FormsModule in app.module.ts file in import section.

1. Model driven form or reactive form.

The flow of the application component to view.

This type of form is complex. If we want to use this type of form you must be strong in angular API.

It is good for complex form.

In this form use formGroup and formControlName attibue on view or template side. These attribute are part of ReactiveFormsModule. So while creating form using reactive form we have to import ReactiveFormsModule in app.module.ts file in import section. In this approach we have to create the Reference of FormGroup and FormControl api in component side to bind formGroup and fromControlName attribute.

According reactive form FormControl is use to bind on view side for textfield, passwordfield, emailedfield, radiobutton, checkbox etc.

FormGroup : FormGroup is a combination of more than one FromControl.

Create two component

ng g c tdf-login-page

ng g c mdf-login-page

In Template driven form we have to create the reference of form using ngForm attribute.

<form #loginRef=”ngForm”>

</form>

**02-05-2022**

Model Driven Form

Login Page

We can do the validation using template driven form as well as model driven form

Angular provided totally six attribute ie

State true false

Condition valid ng-valid ng-invalid

If we do ng-dirty ng-pristine

Changes in

Form component

If we visit the ng-touched ng-untouched

component

pattern

\d 🡪 0 to 9 one digit

\D 🡪 A to Z

[a-d] : it can be any character between a to d

? 0 time or 1 time

+ min 1 and max infinity

\* min 1 and max infinity

Angular Service :

If we write any business logic in component that logic become local to that component.

Some times if we want to use same logic for more than one component then we have take the help of angular service.

Template or html page component ts service ts

First template first component

Variable and function

Business logic

Second template second component

Variable and functions

Angular service

Divided into two types

1. User-defined service
   1. Creating user-defined service object explicitly using new keyword.
   2. Creating user-defined service class object using DI(Dependency Injection) and IOC(Inversion of Control)

IOC : Inversion of Control : It is concept or design pattern or programming pattern. In place of creating object or any resource explicitly allow to create by container (container is also known as engine or run time environment part of server). In place of creating object explicitly pull the object whenever required and use it and leave it.

DI : Dependency Injection : DI is a implementation of IOC.

types of DI

constructor base di

setter base di

interface base di

Angular support only one type of DI ie constructor base DI.

If we want to achieve DI and IOC in Angular framework

We have to make normal ts class with set of business method and on that class we have to use decorator as @Injectable.

This class default we have to provide in app.module.ts file in providers section.

Inside a component we have to pull the object using constructor.

1. Pre-defined service

HttpService

05-05-2022

ng new angular-service

Routing 🡪 no

Styling 🡪 css

Pre-defined Service : Angular Provided pre-defined service ie HttpClient which help call backend technologies REST API develop in any language it may be java, python, asp.net, php etc.

HttpClient pre-defined API part of angular which provided pre-defined method get, post, put and delete to call REST API. And all those method return type is Observable.

Observable is a part of RxJS (Reactive Programming using JavaScript).

Observable is use to handle asynchronous event of data Like same promise.

Promise load all data at time and promise can’t cancel. If function return promise to handle the promise we have to use then and catch. If promise resolved then get call else catch get call to handle rejected data.

If function return Observable to handle the Observable data we have to use subscribe function.

Subscribe function take 3 parameter

1st parameter is next() : to handle data one by one .

2nd parameter is error : if anything happen at the beginning or middle or last. Then 2rd parameter is use to handle the error.

3rd : completed : This will called if there is no error or after loaded data successfully.

Promise can’t be cancel but observable can be cancel using unsubscribe method.

Component --🡪 product -🡪 ng g c product

Service -🡪 product 🡪 ng g s product

Model -🡪 class or interface -🡪 (this class is uses to map the json data retrieve from backend technologies).

ng g interface product

or

ng g class product

we have to do the DI for HttpClient in user-defined class. HttpClient pre-defined API part of HttpClientModule so we have to import HttpClientModule in app.module.ts file in import section.

06-05-2022

Employee component create 🡪 ng g c employee

Employee service create 🡪 ng g s employee

Employee model 🡪 ng g interface employee

09-05-2022

Angular routing : it is use to move from one component another component base upon the path provided in routing file. It is like a hyperlink but using hyperlink we load whole dom but using routing we load only part of the component content base upon the path provided in routing file.

Create new angular routing project using command as

ng new angular-routing

routing 🡪 yes

styling 🡪 css

ng g c aboutus

ng g c contactus

ng g c feedback

ng g c login

ng g c home

Angular routing module provided one of the pre-defined tag ie

<router-outlet></router-outlet>. It is behave like a place holder for angular which help to load the content of web page base upon the path provided in routing file.

Async info() {

await funcitonCall();

let info1 = functionCall();

}

10-05-2022

Testing : Testing is use to find the defect or error or bugs in the application. If you run the application using any technologies indirectly you are testing the application.

Test the program working fine or not without main function or method.

Types of testing

Black box testing : Input Process Output

White box testing Input process Output

UI Testing :

Test Suite :Test suite is like a container which help to write more than one test cases. To write test suite all testing framework provided pre-defined function ie describe.

Syntax

describe(“Message”,callback);

describe(“Message”,()=>{

})

Test Case : test case is use to test the function functionality. To write test case all testing framework provided pre-defined function ie it

Syntax

It(“Message”,callback);

It(“Message”,()=>{

});

describe(“Message”,()=>{

It(“Message”,()=>{

});

It(“Message”,()=>{

})

})

expect : Inside test case we can write more than one expect function which help to check actual and expected output.

Jasmine : Jasmine is open source testing framework which provided lot of pre-defined API which help to do the testing for UI.

We can do testing Plain JavaScript using Jasmine

Jasmine with Angular Framework.

Karma : Karma is test runner for Jasmine framework. Which provide output in proper format.

Mocha : Node JS testing we use Mocha testing framework.

JEST : React JS with JEST testing framework.

Plain JavaScript testing

Jasmine provided pre-defined function (hook function).

beforeEach() : It is a pre-defined function it will call before each it function automatically.

beforeAll() : it is a pre-defined function it will call before all it function automatically. It will call only once.

afterEach() : it is pre-defined function it will call after each it function automatically.

afterAll() : it is a pre-defined function it will call after all it function automatically. It will call only once.

Angular framework internally provide the all configuration for Jasmine testing framework.

Angular user Karma as a test runner to get the output.

Angular pre-defined API ie TestBed. Using TestBed we can test angular component, service, module, pipes etc.

ng new angular-testing

Routing 🡪no

Styling 🡪 css