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

req(http/https)----🡪

Client Server

🡨---Res(http/https) HTML/HTML5

CSS/CSS3

JavaScript

JavaScript was object based interpret scripting language which help to do validation on client side.

CSS

Types of CSS

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

Inline

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

Internal CSS syntax

<style>

Selector {property:value}

</style>

1. Universal Selector

\*{color:red}

1. Specific selector

tagName {color:red}

1. Class selector

tagName.className{color:red}

**JavaScript and ES6**

ES5 : ECMAScript : European Computer Manufacture Association : ES, it is a concept.

All browser doesn’t support ES6 features so it require to transpiler to convert ES6 JavaScript code to ES5 JavaScript

1. Babel
2. Tracesur
3. Typescript

etc

var a =10;

var a =20;

var b =30;

b=40;

let name="Ravi";

//let name="Ajay";

name="Ajay";

for(var i=0;i<100;i++){}

console.log(i)

for(let j=0;j<100;j++){}

console.log("Value is "+j);

function dis(name) {

let msg;

if(name=="Ravi") {

msg ="Welcome "+name;

}else {

msg ="Unknown "+name;

}

return name;

}

console.log(dis("Ravi"));

Types of loops

1. While loop
2. Do while loop
3. For loop
4. For in loop
5. For of loop
6. forEach() function

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

console.log("Size is "+num.length);

for(let i=0;i<num.length;i++) {

console.log(num[i]);

}

console.log("For in loop");

for(let i in num) {

console.log(i+" "+num[i]);

}

console.log("For of loop");

for(let n of num) {

console.log(n);

}

console.log("using forEach function");

num.forEach(dis);

function dis(val) {

console.log(val);

}

num.forEach(function(val){

console.log(val);

})

**ES6 Arrow functions**

function dis1() {

console.log(“dis1() function”);

}

dis1();

function addNumber(a,b) {

var sum = a+b;

return sum;

}

console.log(addNumber(10,20));

let dis2=()=>console.log(“dis2() arrow function”);

dis2();

let addNumber1=(a,b)=>a+b;

console.log(addNumber1(100,200));

let addNumber1=(a,b)=>a+b;

console.log(addNumber1(100,200));

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

num.forEach(val=>console.log(val));

**ES6 function with default parameter and default initialization**

function EmpInfo(id=0,name="Unknown",salary=8000) {

console.log("id is "+id);

console.log("Name is "+name);

console.log("Salary is "+salary);

}

EmpInfo(1,"Ravi",120000);

EmpInfo(2,"Mahesh");

EmpInfo(3);

EmpInfo();

**OOPS Concept using ES6**

ES5 JavaScript was known as Object based

Object Based Vs Object Oriented

Person is a object

function Person() {

console.log("object created");

this.pid = 100;

this.pname = "Ravi";

this.dis = function() {

console.log("id is "+this.pid+" name is "+this.pname);

}

}

var p1 = new Person();

p1.dis();

class Employee {

constructor(empId,empName) {

this.empId =empId;

this.empName = empName;

console.log("Employee object created");

}

dis(){

console.log("Employee class method");

console.log("Id is "+this.empId);

console.log("Name is "+this.empName);

}

}

let emp1 = new Employee(100,"Ravi");

emp1.dis();

let emp2 = new Employee(100);

emp2.dis();

class Employee {

constructor(id,name,salary){

this.id = id;

this.name = name;

this.salary = salary;

}

disEmp(){

console.log("id is "+this.id);

console.log("name is "+this.name);

console.log("salary is "+this.salary);

}

}

class Manager extends Employee {

constructor(id,name,salary,numberOfEmp) {

super(id,name,salary);

this.numberOfEmp = numberOfEmp;

}

disMgr() {

console.log("Number of emps are "+this.numberOfEmp);

}

}

let m1 = new Manager(1,"Ajay",45000,10);

m1.disEmp();

m1.disMgr();

**String template :**

let trainerName="Akash Kale";

let msg ="Welcome to React JS Training "

+"With ES6 "+trainerName+" Features"

let msg1 ='Welcome to React JS Training '

+'With ES6 Features'

let msg2 =`

Welcome to React JS Training

With ES6 Features

by ${trainerName}

`

let trainerName="Akash Kale";

let msg ="Welcome to React JS Training "

+"With ES6 "+trainerName+" Features"

let msg1 ='Welcome to React JS Training '

+'With ES6 Features'

let msg2 =`

Welcome to React JS Training

With ES6 Features

by ${trainerName}

`

**Structure and De- Structure Object Literals**

**Error**

import and export

**modules :** It is a collection of variables, functions classes which have the same name but different purpose. It is like a package in java or namespace in C++.

**a.js**

export function display() {

}

export function display2() {

}

**b.js**

export function display1() {

}

export function display2() {

}

**c.js**

default export function display4() {

}

**d.js**

export function dis1() {}

export function dis2() {}

export function dis3() {}

**main.js**

**import {dislay1,display2} from ‘./b.js’**

**import {display,dispay2 as obj} from ‘./a.js’;**

**import display4 from ‘./c.js’;**

**import \* as obj1 from ‘./d.js’;**

display();

display1();

display2()

obj();

obj1.dis1();

obj1.dis2();

**DOM : Document Object Model :** DOM is a API which provided set of function and property which help to read, write and manipulate the DOM elements.

**React JS : React JS is a open source library provide by facebook to improve the “View” part from the MVC architecture.**

**React JS use component, Component is function or ES6 class which control the view or part of the view.**

**React JS use Virtual DOM Mechanism which help to update Actual DOM after all operation on Virtual DOM using render functions.**

**React JS is use to Create SPA (Single Page Application). In SPA every component work independently. Like header component, footer component navigation component, main component, banner components.**

**React 15.x base upon ES5**

**React 16.x base upon ES5 as well as ES6**

**JSX : JavaScript and XML**

**HTML code**

**<p>Welcome to React JS</p>**

**<div id="app"></div>**

**<div id="app1"></div>**

**JS**

**document.getElementById("app").innerHTML="Welcome to JavaScript";**

**/\***

**function App() {**

**return(<div>Welcome to React JS</div>)**

**}**

**\*/**

**//import {Component} from 'react';**

**class App extends React.Component {**

**render(){**

**return(<div>Welcome to React using ES6 class</div>)**

**}**

**}**

**ReactDOM.render(<App></App>,document.getElementById("app1"));**

**Html**

**<div id=”root”></div>**

**JavaScript**

**class Header extends React.Component {**

**render() {**

**return(<div>Header Component</div>)**

**}**

**}**

**class Footer extends React.Component {**

**render() {**

**return(<div>Footer Component</div>)**

**}**

**}**

**class App extends React.Component {**

**render(){**

**return(<div>**

**<Header></Header>**

**<h1>Welcome to React JS </h1>**

**Welcome to React using ES6 class**

**<Footer></Footer>**

**</div>)**

**}**

**}**

**ReactDOM.render(<App></App>,document.getElementById("root"));**

**React JS Types of Variables**

**In React JS we can declare two type of variables**

1. **state : state variables is a type of variable which hold mutable data(can changes using setState() methods/functions).**

**Using state variables we can describe the behavior of components.**

1. **props : props variables is type of variable which hold immutable data.**

**Using props variables we can pass the value from parent component to child components.**

**Data Binding :**

**Data binding provide the bridge between Component and View.**

**React JS support one way data binding**

1. **Component to View --🡪 {this.state.variableName}**

**Node JS** : Node JS is run time environment for the JavaScript library or framework. Like is JRE in Java. With the help of node js we can write Server side scripting language. Node JS provided asynchronouse non blocking IO programming on network environment.

**npm : node package manager : This command is use to**

**download the node external modules.**

**npm install –g module\_name**

**or**

**npm install module\_name**

**npm install –g create-react-app**

**create-react-app sample-demo**

**Types of Components**

1. **Stateless Components : Normally Stateless components are components created using function or arrow which takes props as a parameter and return JSX. In Stateless components we can’t use state variables as well as life cycle methods of components.**
2. **Statefull Components**

**Statefull component created using ES6 class, Statefull components contains constructor, with**

**State and props variables as well as life cycle methods.**

**React Forms**

**npm install –g create-component-react**

**React Routing : React routing is use to navigate from one component to another base upon the path.**

**Promise Object**

**Calling backend REST Service using fetch/axios**

**Life cycle hook method of React**

**Component communication**

**React with Redux**

**Promise : It is a pre-defined object provided JavaScript, expose or produce the data or emit the data as a Asynchronously.**

**var obj = new Promise(function(accept,reject){**

**accept(“Successully Done”)**

**reject(“Error Generated”)**

**});**

**obj.then(data=>console.log(data)).catch(error=>console.log(error));**

**fetch and axios : Both service help to call the Rest API. Fetch is a pre-defined function provided by JavaScript which help to call REST API. Fetch provided generic features. Axios third party api which help to call the Rest API.**

**React Life Cycle method or Hook methods :**

**React component contains set of pre-defined methods which will get call automatically base upon on the pre-defined event. Whenever we create component which contains three phase**

1. **Mounting or initialization**
2. **Updating**
3. **Unmounting or destroy**
4. **constructor : constructor get called when component object get created. Inside a constructor we have to do state initialization as well as using super pass the props to super class constructor.**
5. **render() :This method is use to render or pass the JSX data to UI(View). This method called in Mounting and updating phase.**

**Component Communication**

**App.js Parent Component**

**State variables**

**Access the variables of child component**

**Child1.js Child component**

**Access the parent component state**

**Variables**

**State variables(own variables)**