

<!DOCTYPE html>

<html>

<head>

<meta charset=utf-8 />

<title>JavaScript program to calculate multiplication and division of two numbers </title>

<style type="text/css">

body {margin: 30px;}

</style>

</head>

<body>

<form>

1st Number : <input type="text" id="firstNumber" /><br>

2nd Number: <input type="text" id="secondNumber" /><br>

<input type="button" onClick="multiplyBy()" Value="Multiply" />

<input type="button" onClick="divideBy()" Value="Divide" />

</form>

<p>The Result is : <br>

<span id = "result"></span>

</p>

</body>

</html>

function multiplyBy()

{

num1 = document.getElementById("firstNumber").value;

num2 = document.getElementById("secondNumber").value;

document.getElementById("result").innerHTML = num1 \* num2;

}

function divideBy()

{

num1 = document.getElementById("firstNumber").value;

num2 = document.getElementById("secondNumber").value;

document.getElementById("result").innerHTML = num1 / num2;

}

**How useEffect works in ReactJS ?**

import React, { useState, useEffect } from 'react';

function App() {

const [count, setCount] = useState(0);

useEffect(() => {

alert(`You clicked ${count} times`)

});

const handleUpdate = ()=> {

setCount (count + 1)

}

return (

<div>

<div>You have clicked {count} times</div>

<button onClick={ handleUpdate} >

Click me

</button>

</div>

);

}

export default App;

# How to pass data from one component to other component in ReactJS ?

//Parent.js

import React from 'react'

import Child from './Child';

const Parent = () => {

const data = "Hello Everyone";

return(

<div>

<Child data={data}/>

</div>

);

}

export default Parent;

//Child.js

import React from 'react';

const Child = (props) => {

return(

<h2> {props.data} </h2>

);

}

export default Child;

//App.js

import React from 'react';

import "./index.css";

import Parent from './Parent'

const App = () => {

return (

<div className="App">

<Parent/>

</div>

);

}

export default App;

//JavaScript program to swap two variables

//take input from the users

let a = parseInt(prompt('Enter the first variable: '));

let b = parseInt(prompt('Enter the second variable: '));

// addition and subtraction operator

a = a + b;

b = a - b;

a = a - b;

console.log(`The value of a after swapping: ${a}`);

console.log(`The value of b after swapping: ${b}`);

// program that checks if the number is positive, negative or zero

// input from the user

const number = parseInt(prompt("Enter a number: "));

// check if number is greater than 0

if (number > 0) {

console.log("The number is positive");

}

// check if number is 0

else if (number == 0) {

console.log("The number is zero");

}

// if number is less than 0

else {

console.log("The number is negative");

}

// program to check an Armstrong number of three digits

let sum = 0;

const number = prompt('Enter a three-digit positive integer: ');

// create a temporary variable

let temp = number;

while (temp > 0) {

// finding the one's digit

let remainder = temp % 10;

sum += remainder \* remainder \* remainder;

// removing last digit from the number

temp = parseInt(temp / 10); // convert float into integer

}

// check the condition

if (sum == number) {

console.log(`${number} is an Armstrong number`);

}

else {

console.log(`${number} is not an Armstrong number.`);

}

// program to find the factorial of a number

function factorial(x) {

// if number is 0

if (x == 0) {

return 1;

}

// if number is positive

else {

return x \* factorial(x - 1);

}

}

// take input from the user

const num = prompt('Enter a positive number: ');

// calling factorial() if num is positive

if (num >= 0) {

const result = factorial(num);

console.log(`The factorial of ${num} is ${result}`);

}

else {

console.log('Enter a positive number.');

}

// program to check the number of occurrence of a character

function countString(str, letter) {

let count = 0;

// looping through the items

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

// check if the character is at that position

if (str.charAt(i) == letter) {

count += 1;

}

}

return count;

}

// take input from the user

const string = prompt('Enter a string: ');

const letterToCheck = prompt('Enter a letter to check: ');

//passing parameters and calling the function

const result = countString(string, letterToCheck);

// displaying the result

console.log(result);

// program to pass a function as a parameter

function greet() {

return 'Hello';

}

// passing function greet() as a parameter

function name(user, func)

{

// accessing passed function

const message = func();

console.log(`${message} ${user}`);

}

name('John', greet);

name('Jack', greet);

name('Sara', greet);

Ref: <https://www.programiz.com/javascript/examples>

# How to Access the File System in Node.js ?

const fs = require('fs');

/\* The fs.writeFileSync method is used

to write something to the file, but if

the file does not exist, it creates new

files along with writing the contents \*/

fs.writeFileSync('./testfile', 'This is a file');

var file\_content = fs.readFileSync(

'./testfile', 'utf8').toString();

console.log(file\_content);

/\* The fs.appendFileSync method is used

for updating the data of a file \*/

fs.appendFileSync('./testfile', " Updated Data");

file\_content = fs.readFileSync(

'./testfile', 'utf8').toString();

console.log(file\_content);

/\* The fs.unlinkSync method are used to delete

the file. With passing the file name \*/

fs.unlinkSync('./testfile');

# Create a Calculator Node.js Module with functions add, subtract and multiply. And use the Calculator module in another Node.js file.

// Returns addition of two numbers

exports.add = function (a, b) {

    return a+b;

};

// Returns difference of two numbers

exports.subtract = function (a, b) {

    return a-b;

};

// Returns product of two numbers

exports.multiply = function (a, b) {

    return a\*b;

};

var calculator = require('./calculator');

var a=10, b=5;

console.log("Addition : "+calculator.add(a,b));

console.log("Subtraction : "+calculator.subtract(a,b));

console.log("Multiplication : "+calculator.multiply(a,b));

# How to create node.js web applications

var http = require("http");

http.createServer(function (request, response) {

// Send the HTTP header

// HTTP Status: 200 : OK

// Content Type: text/plain

response.writeHead(200, {'Content-Type': 'text/plain'});

// Send the response body as "Hello World"

response.end('Hello World\n');

}).listen(8081);

// Console will print the message

console.log('Server running at http://127.0.0.1:8081/');

# Express routers allow us to serve different HTTP methods such as GET, POST, PUT, DELETE, HEAD. how to create a router.

const express = require('express');

const app = express();

const router = express.Router();

router.get('/home', (req,res) => {

res.send('Hello World, This is home router');

});

router.get('/profile', (req,res) => {

res.send('

Hello World, This is profile router

');

});

router.get('/login', (req,res) => {

res.send('

Hello World, This is login router

');

});

router.get('/logout', (req,res) => {

res.send('

Hello World, This is logout router

');

});

app.use('/', router);

app.listen(process.env.port || 3000);

console.log('Web Server is listening at port '+ (process.env.port || 3000));

Ref: <https://codeforgeek.com/node-js-tutorial-step-by-step/>

# Differences between Functional Components and Class Components in React

import React, { useState } from "react";

const FunctionalComponent=()=>{

const [count, setCount] = useState(0);

const increase = () => {

setCount(count+1);

}

return (

<div style={{margin:'50px'}}>

<h1>Welcome to Geeks for Geeks </h1>

<h3>Counter App using Functional Component : </h3>

<h2>{count}</h2>

<button onClick={increase}>Add</button>

</div>

)

}

export default FunctionalComponent;

----------------------------------------------------------------

import React, { Component } from "react";

class ClassComponent extends React.Component{

constructor(){

super();

this.state={

count :0

};

this.increase=this.increase.bind(this);

}

increase(){

this.setState({count : this.state.count +1});

}

render(){

return (

<div style={{margin:'50px'}}>

<h1>Welcome to Geeks for Geeks </h1>

<h3>Counter App using Class Component : </h3>

<h2> {this.state.count}</h2>

<button onClick={this.increase}> Add</button>

</div>

)

}

}

export default ClassComponent;

# ReactJS | Lifecycle of Components

import React from 'react';

import ReactDOM from 'react-dom';

class Test extends React.Component {

constructor(props)

{

super(props);

this.state = { hello : "World!" };

}

componentWillMount()

{

console.log("componentWillMount()");

}

componentDidMount()

{

console.log("componentDidMount()");

}

changeState()

{

this.setState({ hello : "Geek!" });

}

render()

{

return (

<div>

<h1>GeeksForGeeks.org, Hello{ this.state.hello }</h1>

<h2>

<a onClick={this.changeState.bind(this)}>Press Here!</a>

</h2>

</div>);

}

shouldComponentUpdate(nextProps, nextState)

{

console.log("shouldComponentUpdate()");

return true;

}

componentWillUpdate()

{

console.log("componentWillUpdate()");

}

componentDidUpdate()

{

console.log("componentDidUpdate()");

}

}

ReactDOM.render(

<Test />,

document.getElementById('root'));

# JavaScript Class Inheritance

// parent class

class Person {

constructor(name) {

this.name = name;

}

greet() {

console.log(`Hello ${this.name}`);

}

}

// inheriting parent class

class Student extends Person {

}

let student1 = new Student('Jack');

student1.greet();

## **JavaScript super() keyword**

// parent class

class Person {

constructor(name) {

this.name = name;

}

greet() {

console.log(`Hello ${this.name}`);

}

}

// inheriting parent class

class Student extends Person {

constructor(name) {

console.log("Creating student class");

// call the super class constructor and pass in the name parameter

super(name);

}

}

let student1 = new Student('Jack');

student1.greet();

Node.js - Callbacks Concept

## **Blocking Code Example**

Create a text file named **input.txt** with the following content −

Tutorials Point is giving self learning content

to teach the world in simple and easy way!!!!!

Create a js file named **main.js** with the following code −

var fs = require("fs");

var data = fs.readFileSync('input.txt');

console.log(data.toString());

console.log("Program Ended");

Now run the main.js to see the result −

$ node main.js

Verify the Output.

Tutorials Point is giving self learning content

to teach the world in simple and easy way!!!!!

Program Ended

## **Non-Blocking Code Example**

Create a text file named input.txt with the following content.

Tutorials Point is giving self learning content

to teach the world in simple and easy way!!!!!

Update main.js to have the following code −

var fs = require("fs");

fs.readFile('input.txt', function (err, data) {

if (err) return console.error(err);

console.log(data.toString());

});

console.log("Program Ended");

Now run the main.js to see the result −

$ node main.js

Verify the Output.

Program Ended

Tutorials Point is giving self learning content

to teach the world in simple and easy way!!!!!