1. Write a program to Show an alert

Ans:

<body>

    <button onclick="showAlert()">CLick Me</button>

    <script>

        // Define a function to show an alert

        function showAlert() {

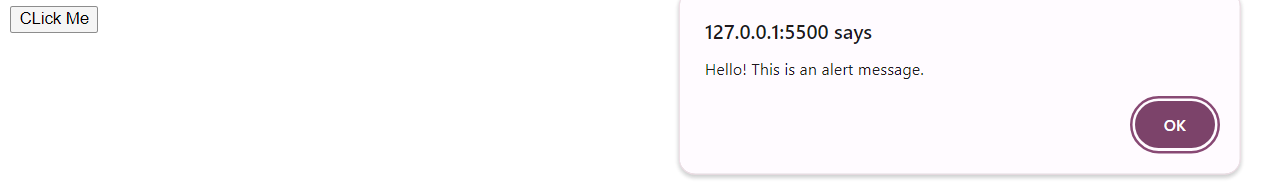
            alert("Hello! This is an alert message.");

        }

    </script>

</body>

Output :



1. What will be the result for these expressions?

Ans:

1. 5 > 4 => True

2. "apple" > "pineapple" => False

3. "2" > "12" => True

4.Undefined == null => True

1. undefined === null => Fasle

6. null == "\n0\n" => False

7. null === "\n0\n" => False

1. Will alert be shown?

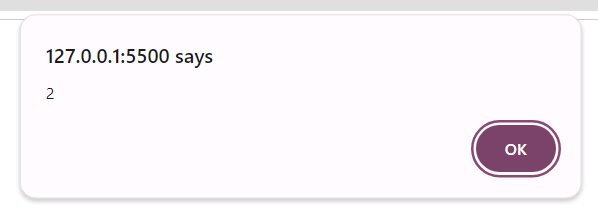
if ("0") { alert( 'Hello'); }

Ans: Yes, the alert will be shown.

In JavaScript, the condition `"0"` evaluates to `true` in a boolean context. This is because any non-empty string, including `"0"`, is considered truthy. Therefore, the code inside the `if` statement will execute, and the alert will be shown.

1. What is the code below going to output? alert( null || 2 || undefined );

Ans: The expression evaluates to true.



1. The following function returns true if the parameter age is greater than

18. Otherwise it asks for a confirmation and returns its result:

Ans:

<body>

    <span>Enter Your Age : </span>

    <input type="number" id="age1">

    <Button onclick="checkAge()">Submit</Button>

    <script>

        const Age = parseInt(document.getElementById("age1").value);

        function checkAge() {

            if (Age > 18) {

                return alert("Welcome");

            }

            else {

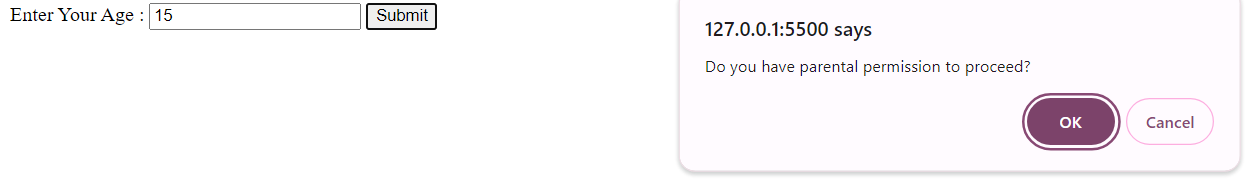
                return confirm("Do you have parental permission to proceed?");

            }

        }

    </script>

</body>



1. Replace Function Expressions with arrow functions in the code below:

Ans:

<script>

        let ask = (question, yes, no) => {

            if (confirm(question)) yes();

            else no();

        }

        ask("Do you agree?",

            () => { alert("You agreed."); },

            () => { alert("You canceled the execution."); }

        );

    </script>

1. Write the code, one line for each action:

Ans:

a) Create an empty object user. => let user = {};

b) Add the property name with the value John. => user.name=“Jhon”;

c) Add the property surname with the value Smith. => user,surname=“Smith”;

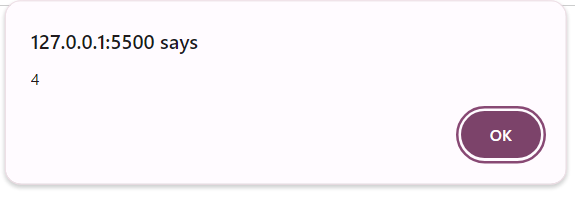
d) Change the value of the name to Pete. => user.name = “Pete”;

e) Remove the property name from the object. => delete user.name;

1. Is array copied?

Ans: If the source array and destination array are both rerefrence-type array or are both arrays of type Object, s shallow copy is performed. Array. The elements themselves or anything referenced by the elements are not copied.

* <script>
* let fruits = ["Apples", "Pear", "Orange"]; // Original array
* let shoppingCart = fruits; // Assigning the reference of fruits to shoppingCart
* shoppingCart.push("Banana"); // Modifying shoppingCart
* document.write(fruits); // Output: ["Apples", "Pear", "Orange", "Banana"]
* </script>





1. Map to names

Ans:

<script>

        let john = { name: "John", age: 25 };

        let pete = { name: "Pete", age: 30 };

        let mary = { name: "Mary", age: 28 };

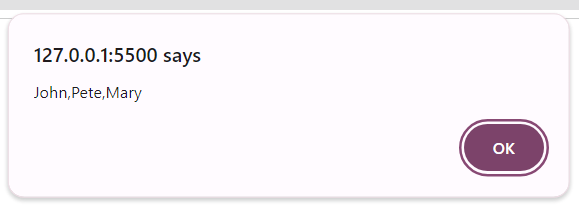
        let users = [john, pete, mary];

        let names = users.map(user => user.name);

        alert(names); // John, Pete, Mary

    </script>

Output:



1. Map to objects

Ans:

<script>

        let john = { name: "John", surname: "Smith", id: 1 };

        let pete = { name: "Pete", surname: "Hunt", id: 2 };

        let mary = { name: "Mary", surname: "Key", id: 3 };

        let users = [john, pete, mary];

        let usersMapped = users.map(user => ({

            fullName: `${user.name} ${user.surname}`,

            id: user.id

        }));

        /\*

        usersMapped = [

          { fullName: "John Smith", id: 1 },

          { fullName: "Pete Hunt", id: 2 },

          { fullName: "Mary Key", id: 3 }

        ]

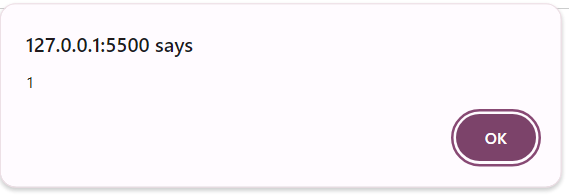
        \*/

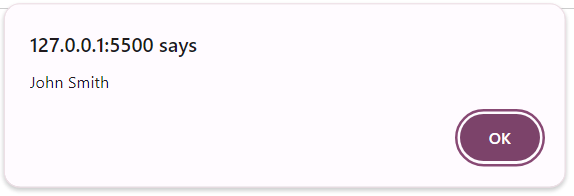
        alert(usersMapped[0].id); // 1

        alert(usersMapped[0].fullName); // John Smith

    </script>

Output :





1. Sum the properties There is a salaries object with arbitrary number of salaries. Write the function sumSalaries(salaries) that returns the sum of all salaries using Object.values and the for..of loop.If salaries is empty, then the result must be 0.

Ans:

<script>

        let salaries = {

            "John": 100,

            "Pete": 300,

            "Mary": 250

        };

        function sumSalaries(salaries) {

            let sum = 0;

            for (let salary of Object.values(salaries)) {

                sum += salary;

            }

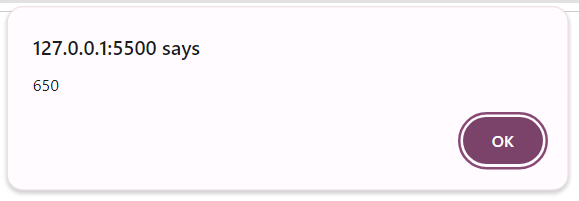
            return sum;

        }

        alert(sumSalaries(salaries)); // Output: 650

    </script>

Output :



1. Destructuring assignment We have an object: Write the Destructuring assignment that reads

Ans:

<script>

        let user = { name: "John", years: 30 };

        // a) Name property into the variable name.

        let { name } = user;

        document.write(name); // Output: "John"

        document.write("<br>");

        // b) Year’s property into the variable age.

        let { years: age } = user;

        document.write(age); // Output: 30

        document.write("<br>");

        // c) isAdmin property into the variable isAdmin (false, if no such property)

        let { isAdmin = false } = user;

        document.write(isAdmin); // Output: false

    </script>

Output :



1. Turn the object into JSON and back Turn the user into JSON and then read it back into another variable.

Ans:

<script>

        let user = { name: "John Smith", age: 35 };

        // Convert user object to JSON string

        let userJSON = JSON.stringify(user);

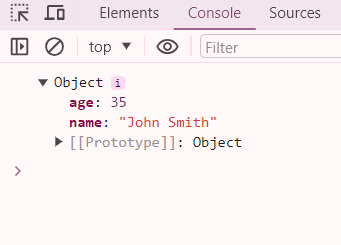
        // Parse JSON string back into an object

        let userParsed = JSON.parse(userJSON);

        console.log(userParsed); // Output: { name: "John Smith", age: 35 }

    </script>

Output :



1. Create a program to hide/show the password

Ans:

<body>

    <label for="password">Password:</label>

    <input type="password" id="password" name="password">

    <input type="checkbox" id="showPasswordCheckbox">

    <label for="showPasswordCheckbox">Show password</label>

    <script>

        const passwordInput = document.getElementById('password');

        const showPasswordCheckbox = document.getElementById('showPasswordCheckbox');

        // Add event listener to the checkbox

        showPasswordCheckbox.addEventListener('change', function () {

            // If the checkbox is checked, show the password

            if (this.checked) {

                passwordInput.type = 'text';

            } else {

                // If the checkbox is not checked, hide the password

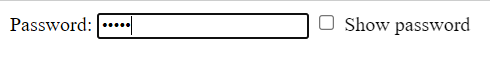
                passwordInput.type = 'password';

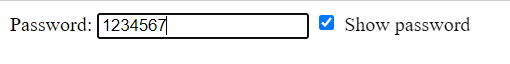
            }

        });

    </script>

Output :





1. Create a program that will select all the classes and loop over and whenever i click the button the alert should show

Ans:

<body>

    <!-- Add some elements with the class 'clickable' -->

    <button id="button">Click me!</button>

    <p class="clickable">Element 1</p>

    <p class="clickable">Element 2</p>

    <p class="clickable">Element 3</p>

    <script src="script.js"></script>

    <script>

        let clickableElements = document.querySelectorAll('.clickable');

        // Add event listener to each clickable element

        clickableElements.forEach(element => {

            element.addEventListener('click', () => {

                alert('You clicked on ' + element.innerText);

            });

        });

        // Add event listener to the button

        document.getElementById('button').addEventListener('click', () => {

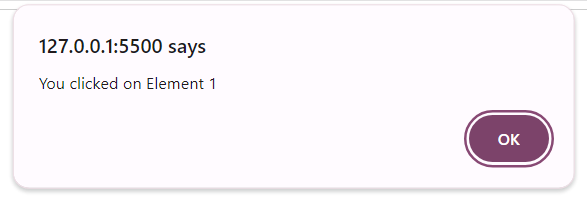
            alert('Button clicked!');

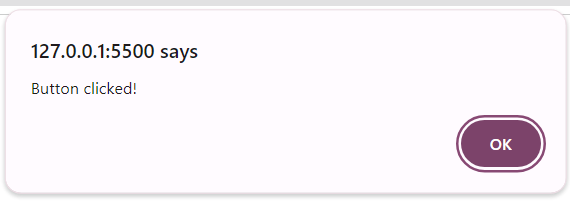
        });

    </script>

</body>

Output :





1. Create a responsive header using proper JavaScript

Ans:

<style>

    header {

        background-color: #333;

        color: #fff;

        padding: 10px 20px;

        display: flex;

        justify-content: space-between;

        align-items: center;

    }

    .hamburger {

        color: #fff;

        font-size: 24px;

        cursor: pointer;

        display: none;

        /\* Initially hide the hamburger icon \*/

    }

    nav ul {

        list-style-type: none;

        margin: 0;

        padding: 0;

        display: flex;

    }

    nav ul li {

        margin-right: 100px;

    }

    nav ul li a {

        color: #fff;

        text-decoration: none;

    }

    /\* Responsive styles for the header \*/

    @media screen and (max-width: 768px) {

        .hamburger {

            display: block;

            /\* Show the hamburger icon for smaller screens \*/

        }

        nav ul {

            display: none;

            /\* Hide the navigation menu by default for smaller screens \*/

        }

    }

</style>

<body>

    <header id="header">

        <div id="hamburger" class="hamburger">&#9776;</div>

        <nav id="nav">

            <ul>

                <li><a href="#">Home</a></li>

                <li><a href="#">About</a></li>

                <li><a href="#">Services</a></li>

                <li><a href="#">Contact</a></li>

            </ul>

        </nav>

    </header>

    <script>

        function toggleNav() {

            const nav = document.getElementById('nav');

            nav.classList.toggle('show');

        }

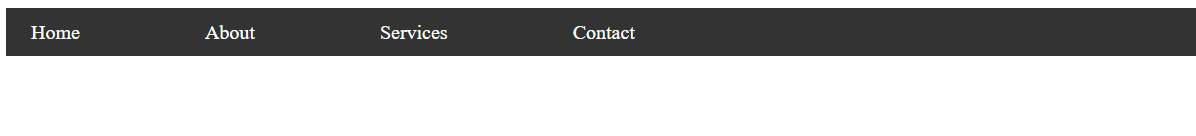
        // Event listener for the hamburger icon

        document.getElementById('hamburger').addEventListener('click', toggleNav);

    </script>

</body>

Output:



Menu Hamburger at small screen.



1. Create a form and validate using JavaScript

Ans:

<body>

    <form name="myform" action="" method="post" onsubmit="return validate(), success()">

        <table border="2" align="center">

            <tr>

                <th> Registration Page </th>

            </tr>

            <tr>

                <th>User Fist Name:-</th>

                <td>

                    <input type="text" name="ufn">

                    <span style="color:red" id="msg1"></span>

                </td>

            </tr>

            <tr>

                <th>User Name:-</th>

                <td><input type="text" name="un"></td>

            </tr>

            <tr>

                <th>Password:-</th>

                <td><input type="password" name="pass"></td>

            </tr>

            <tr>

                <th>Confirm Password:-</th>

                <td><input type="password" name="cpass"></td>

            </tr>

            <tr>

                <th>Email Address:-</th>

                <td><input type="text" name="email"></td>

            </tr>

            <tr>

                <th>Phone No:-</th>

                <td><input type="text" name="pno"></td>

            </tr>

            <tr>

                <td><input type="submit" name="submit" /></td>

            </tr>

        </table>

    </form>

</body>

<script>

        // first Make one function

        function validate()  // function name

        {

            var ufn = document.forms["myform"]["ufn"].value;

            if (ufn == "" || ufn == null)  // for null condition

            {

                //alert('Please fill out the User First Name');  // alert msg

                document.getElementById('msg1').innerHTML = "Please fill out the User First Name";

                return false;   //return false means msg show and again on same page with value not refresh page

            }

            else {

                document.getElementById('msg1').innerHTML = "";

            }

            // /^ $/

            //alpha=/^[A-Za-z]+$/

            //var phone=/^[0-9]{10,11}$/;

            //var mail=/^[a-zA-Z0-9\_]+@[a-zA-Z]+\.[a-zA-Z]{2,4}$/;

            var alpha = /^[A-Za-z]+$/;  // /^[A-Za-z]{3,8}$/

            if (!alpha.test(ufn)) {

                alert('Please fill only alpha User First Name');  // alert msg

                return false;   //return false means msg show and again on same page with value not refresh page

            }

            var un = document.forms["myform"]["un"].value;

            if (un == "" || un == null)  // for null condition

            {

                alert('Please fill out the User Name');  // alert msg

                return false;   //return false means msg show and again on same page with value not refresh page

            }

            var pass = document.forms["myform"]["pass"].value;

            if (pass == "" || pass == null)  // for null condition

            {

                alert('Please fill out the pass');  // alert msg

                return false;   //return false means msg show and again on same page with value not refresh page

            }

            if (!(pass.length <= 8 && pass.length >= 3)) {

                alert('Please,provide min 3 & max 8 char in pass');

                return false;

            }

            var cpass = document.forms["myform"]["cpass"].value;

            if (cpass == "" || cpass == null)  // for null condition

            {

                alert('Please fill out the cpass');  // alert msg

                return false;   //return false means msg show and again on same page with value not refresh page

            }

            if (pass == cpass) {

            }

            else {

                alert('Please Enter Same values');  // alert msg

                return false;   //return false means msg show and again on same page with value not refresh page

            }

            var email = document.forms["myform"]["email"].value;

            if (email == "" || email == null)  // for null condition

            {

                alert('Please fill out the email');  // alert msg

                return false;   //return false means msg show and again on same page with value not refresh page

            }

            var mail = /^[a-zA-Z0-9\_]+@[a-zA-Z]+\.[a-zA-Z]{2,4}$/;

            if (!mail.test(email)) {

                alert('Please fill proper email id');  // alert msg

                return false;   //return false means msg show and again on same page with value not refresh page

            }

            var pno = document.forms["myform"]["pno"].value;

            if (pno == "" || pno == null)  // for null condition

            {

                alert('Please fill out the pno');  // alert msg

                return false;   //return false means msg show and again on same page with value not refresh page

            }

            var phone = /^[0-9]{10,11}$/;

            if (!phone.test(pno)) {

                alert('Please fill only digits in PNO');  // alert msg

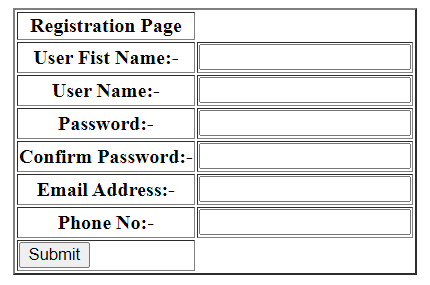
                return false;   //return false means msg show and again on same page with value not refresh page

            }

        }

    </script>

Output:



1. Create a modal box using css and Js with three buttons.

Ans:

<style>

    .modal {

        display: none;

        /\* Hide modal by default \*/

        position: fixed;

        z-index: 1;

        left: 0;

        top: 0;

        width: 100%;

        height: 100%;

        overflow: auto;

        background-color: rgba(0, 0, 0, 0.5);

        /\* Black with opacity \*/

    }

    .modal-content {

        background-color: #fefefe;

        margin: 15% auto;

        /\* Center modal vertically and horizontally \*/

        padding: 20px;

        border: 1px solid #888;

        width: 80%;

        text-align: center;

    }

    .close {

        color: #aaa;

        float: right;

        font-size: 28px;

        font-weight: bold;

    }

    .close:hover,

    .close:focus {

        color: black;

        text-decoration: none;

        cursor: pointer;

    }

    /\* Button Styles \*/

    button {

        padding: 10px 20px;

        margin: 10px;

    }

</style>

<body>

    <button id="modalBtn">Open Modal</button>

    <!-- Modal Box -->

    <div id="myModal" class="modal">

        <div class="modal-content">

            <span class="close">&times;</span>

            <p>This is a modal box!</p>

            <button id="btn1">Button 1</button>

            <button id="btn2">Button 2</button>

            <button id="btn3">Button 3</button>

        </div>

    </div>

    <script>

        var modal = document.getElementById("myModal");

        // Get the button that opens the modal

        var btn = document.getElementById("modalBtn");

        // Get the <span> element that closes the modal

        var span = document.getElementsByClassName("close")[0];

        // When the user clicks on the button, open the modal

        btn.onclick = function () {

            modal.style.display = "block";

        }

        // When the user clicks on <span> (x), close the modal

        span.onclick = function () {

            modal.style.display = "none";

        }

        // When the user clicks anywhere outside of the modal, close it

        window.onclick = function (event) {

            if (event.target == modal) {

                modal.style.display = "none";

            }

        }

        // Example event listeners for the buttons inside the modal

        document.getElementById("btn1").addEventListener("click", function () {

            alert("Button 1 clicked");

        });

        document.getElementById("btn2").addEventListener("click", function () {

            alert("Button 2 clicked");

        });

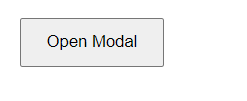
        document.getElementById("btn3").addEventListener("click", function () {

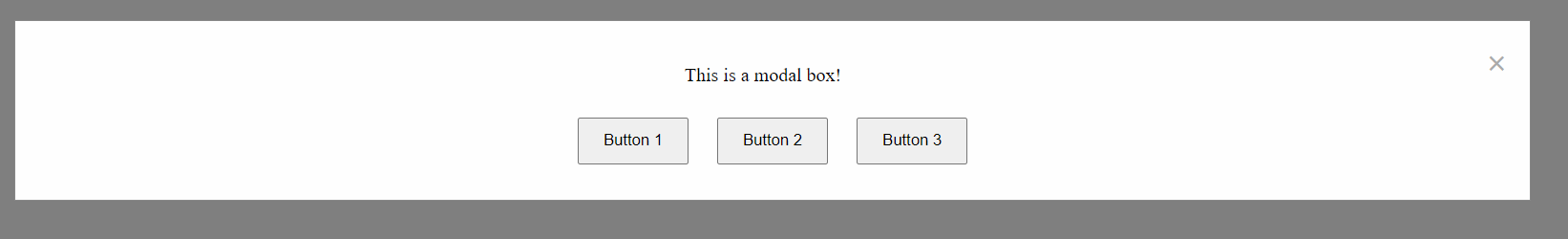
            alert("Button 3 clicked");

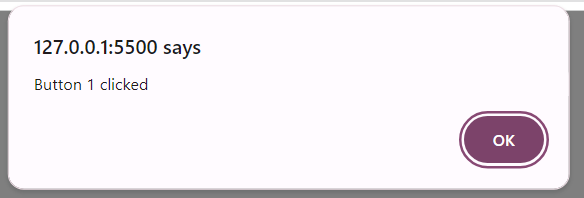
        });

    </script>

Output :







1. Use external js library to show slider

Ans:

Create a new file with .js extension and call that in html file inside <script> tag and provide link in src attribute

let slideIndex = 0;

    let slides = document.querySelectorAll('.box');

    let prev = document.querySelector('.prev');

    let next = document.querySelector('.next');

    function showSlide(index) {

        if (index < 0) {

            index = slides.length - 1;

        } else if (index >= slides.length) {

            index = 0;

        }

        slides.forEach(slide => {

            slide.style.transform = `translateX(-${index \* 50}%)`;

        });

        slideIndex = index;

    }

    function nextSlide() {

        showSlide(slideIndex + 1);

    }

    function prevSlide() {

        showSlide(slideIndex - 1);

    }

Create new main file with .html extention

<style>

        .slider{

            position: relative;

        }

        .box-container {

            display: flex;

            overflow: hidden;

        }

        .box {

            width: 50%;

            height: 90vh;

            flex:   0 0 auto;

            border: 1px solid black;

            margin: 10px;

            transition: transform 1.5 ease-out;

        }

        button {

            position: absolute;

            top: 50%;

            background: none;

            border: none;

            font-size: 24px;

            cursor: pointer;

        }

        .prev {

            left: 0px;

        }

        .next {

            right: 0px;

        }

    </style>

</body>

<div id="slider">

<div class="box-container">

    <div class="box" style="background-color: orange;"></div>

    <div class="box" style="background-color: red;"></div>

    <div class="box" style="background-color: blue;"></div>

    <div class="box" style="background-color: pink;"></div>

</div>

<button class="prev" onclick="prevSlide()">❮</button>

<button class="next" onclick="nextSlide()">❯</button>

</div>

<script src="externalSlider.js"></script>

Output :



1. Prevent the browser when i click the form submit button

Ans:

<body>

    <form id="myForm">

        <input type="text" name="username" placeholder="Username">

        <input type="password" name="password" placeholder="Password">

        <button type="submit" id="submitButton">Submit</button>

    </form>

    <script>

        var form = document.getElementById("myForm");

        // Add event listener to the form

        form.addEventListener("submit", function (event) {

            // Prevent the default form submission behavior

            event.preventDefault();

            // Additional code to handle form submission (if needed)

            document.write("Form submitted, but default behavior prevented.");

        });

    </script>

</body>

Output :

`



1. What is JSON

Ans: JSON stands for **J**ava**S**cript **O**bject **N**otation

JSON is a lightweight format for storing and transporting data

JSON is often used when data is sent from a server to a web page

JSON is "self-describing" and easy to understand

The JSON format is syntactically identical to the code for creating JavaScript objects.

Because of this similarity, a JavaScript program can easily convert JSON data into native JavaScript objects.

1. What is promises

Ans: promises allows you to associate handlers with an asynchronous action’s eventual success value or failure reason. This lets asynchronous methods return values like synchronous methods of immediately returning the final value, the

Asynchronous method returns a promise to supply the value at some point in the future.

There are three states in promises

* Pending: initial state, neither fulfilled nor rejected.
* Fulfilled: meaning that the operation was completed successfully
* Rejected: meaning that the operation failed.

1. Write a program of promises and handle that promises also

Ans:

<script>

        function delay(ms) {

            return new Promise(resolve => setTimeout(resolve, ms));

        }

        // Function that simulates an asynchronous task

        function simulateAsyncTask() {

            return new Promise(resolve => {

                // Simulate some asynchronous task

                setTimeout(() => {

                    resolve("Async task completed");

                }, 2000); // Simulating a delay of 2 seconds

            });

        }

        // Handling the promises

        async function handlePromises() {

            try {

                console.log("Starting the program...");

                // Using delay function

                await delay(1000); // Wait for 1 second

                console.log("After delay of 1 second");

                // Simulating an asynchronous task

                const result = await simulateAsyncTask();

                console.log(result);

                console.log("Program completed successfully!");

            } catch (error) {

                console.error("Error occurred:", error);

            }

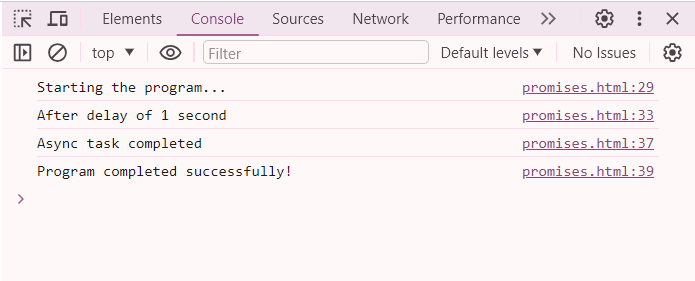
        }

        // Call the function to handle promises

        handlePromises();

    </script>

Output :



1. Use fetch method for calling an api <https://fakestoreapi.com/products>

Ans:

<script>

        fetch('https://fakestoreapi.com/products')

            .then(response => {

                // Check if the response is successful (status code 200)

                if (!response.ok) {

                    throw new Error('Network response was not ok');

                }

                // Parse the JSON response

                return response.json();

            })

            .then(data => {

                // Handle the JSON data

                console.log(data);

            })

            .catch(error => {

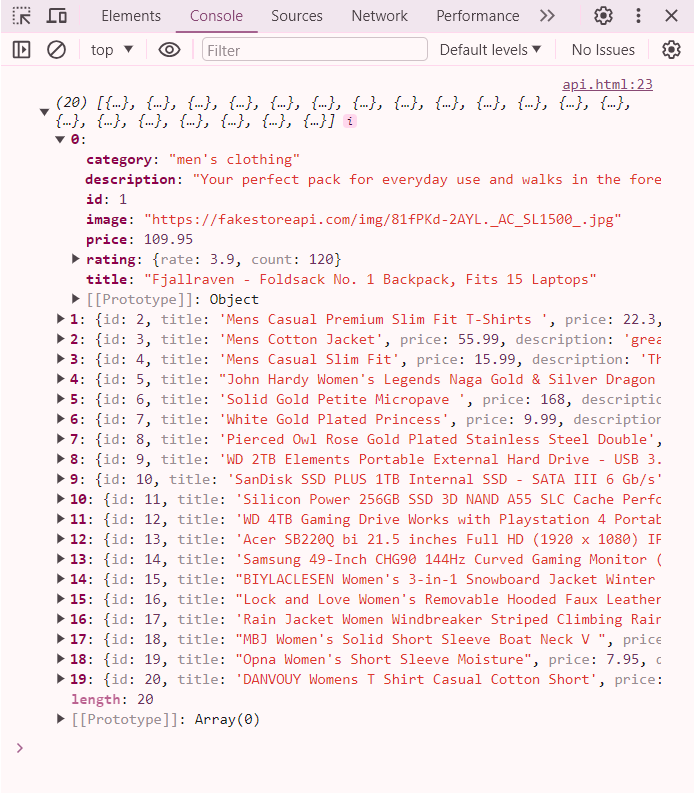
                // Handle errors

                console.error('There was a problem with the fetch operation:', error);

            });

    </script>

Output :



1. Display all the product from the api in your HTML page

Ans:

<style>

    .product {

        border: 1px solid #ccc;

        padding: 10px;

        margin-bottom: 20px;

    }

    .product img {

        max-width: 10%;

        height: auto;

    }

</style>

<body>

    <div id="products-container"></div>

    <script>

        async function fetchAndDisplayProducts() {

            try {

                // Make a GET request to the API endpoint

                const response = await fetch('https://fakestoreapi.com/products');

                // Check if the response is successful (status code 200)

                if (!response.ok) {

                    throw new Error('Network response was not ok');

                }

                // Parse the JSON response

                const products = await response.json();

                // Get the container element to display products

                const productsContainer = document.getElementById('products-container');

                // Loop through the products and create HTML elements to display them

                products.forEach(product => {

                    // Create a div element for each product

                    const productDiv = document.createElement('div');

                    productDiv.classList.add('product');

                    // Populate the div with product information

                    productDiv.innerHTML = `

                <h2>${product.title}</h2>

                <img src="${product.image}" alt="${product.title}">

                <p>Price: $${product.price}</p>

                <p>Description: ${product.description}</p>

            `;

                    // Append the product div to the products container

                    productsContainer.appendChild(productDiv);

                });

            } catch (error) {

                // Handle errors

                console.error('There was a problem with the fetch operation:', error);

            }

        }

        // Call the function to fetch and display products

        fetchAndDisplayProducts();

    </script>

Output :

