Advanced Javascript

1. Write a program to Show an alert

>> <!DOCTYPE html>

<html>

<body>

<h1>The Window Object</h1>

<h2>The alert() Method</h2>

<p>Click the button to display an alert box.</p>

<button onclick="myFunction()">Try it</button>

<script>

function myFunction() {

alert("Hello! I am an alert box!");

}

</script>

</body>

</html>

1. What will be the result for these expressions?

>>

1. Will alert be shown

>> **Yes, it will.** Any string except an empty one (and "0" is not empty) becomes true in the logical context.

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

>> <!DOCTYPE html>

<script>

"use strict";

alert( null || 2 || undefined );

</script>

**MODULE: 2 (Data Types and Objects)**

(1)Write the code, one line for each action:

1. Create an empty object user.

>> Objects can also be created using the new keyword. With the built-in Object Constructor in Javascript, new creates an empty object; or, this keyword can be used with a user-defined constructor function: with builtin Object Constructor .

<!DOCTYPE html>

<html>

<body>

<script >

const obj = {};

console.log(obj)

var o = new Object();

console.log(o)

</script>

</body>

</html>

(b) Add the property name with the value John.

<!DOCTYPE html>

<html>

<body>

<h2>JavaScript Object Properties</h2>

<p>Access a property with .property:</p>

<p id="demo"></p>

<script>

const person = {

firstname: "John",

lastname: "Doe",

age: 50,

eyecolor: "blue"

};

document.getElementById("demo").innerHTML = person.firstname + " is " + person.age + " years old.";

</script>

</body>

</html>

(c) Add the property surname with the value Smith.

>>

var person = {

firstName: 'Jade',

lastName: 'Smith'

};

addFullNameProperty(person);

console.log(person.fullName); // --> 'Jade Smith'

my code :

function addFullNameProperty(obj) {

// your code here

obj[fullName] = obj.firstName + obj.lastName;

}

(D) Change the value of the name to Pete.

1. Use the Object. keys() method to get an array of the object's keys.
2. Iterate over the array using the forEach() method and update each value.
3. After the last iteration, all the values in the object will be updated.

(E) Remove the property name from the object.

<body>

<h2>JavaScript Object Properties</h2>

<p>You can delete object properties with the delete keyword.</p>

<p id="demo"></p>

<script>

var person = {

firstname: "John",

lastname: "Doe",

age: 50,

eyecolor: "blue"

};

delete person.age;

document.getElementById("demo").innerHTML =

person.firstname + " is " + person.age + " years old.";

</script>

</body>

(2) Is array copied? let fruits = ["Apples", "Pear", "Orange"]; // push a new value into the "copy" let shoppingCart = fruits; shoppingCart.push("Banana"); // what's in fruits?

alert( fruits.length ); // ?

>> There are different methods to copy an array. You can use a for loop and copy elements of one to another one by one. Use the clone method to clone an array. Use arraycopy() method of System class.

(3) Map to names

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 = /\* ... your code \*/ alert( names ); // John, Pete, Mary

>> You have an array of user objects, each one has user.name. Write the code that converts it into an array of names.

For intence :

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 = /\* ... your code \*/

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

solution:

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(item => item.name);

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

(3) Map to objects

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 = /\* ... your code ... \*/

/\*

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

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.

let salaries = {

"John": 100,

"Pete": 300,

"Mary": 250

};

alert( sumSalaries(salaries) ); // 650

● Destructuring assignment We have an object: Write the Destructuring assignment that reads:

a) Name property into the variable name.

b) Year’s property into the variable age.

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

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

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

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

**MODULE: 3 (Document, Event and Controls)**

1. Create a program to hide/show the password

<body>

<p>Click the radio button to toggle between password visibility:</p>

Password: <input type="password" value="FakePSW" id="myInput"><br><br>

<input type="checkbox" onclick="myFunction()">Show Password

<script>

function myFunction() {

var x = document.getElementById("myInput");

if (x.type === "password") {

x.type = "text";

} else {

x.type = "password";

}

}

</script>

</body>

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

>> <button onclick="showAlert()">Show alert</button>

<script>

function showAlert() {

var myText = "This can be whatever text you like!";

alert (myText);

}

</script>

1. Create a responsive header using proper JavaScript

<!DOCTYPE html>

<html>

<head>

<meta name="viewport" content="width=device-width, initial-scale=1">

<style>

\* {box-sizing: border-box;}

body {

margin: 0;

font-family: Arial, Helvetica, sans-serif;

}

.header {

overflow: hidden;

background-color: #f1f1f1;

padding: 20px 10px;

}

.header a {

float: left;

color: black;

text-align: center;

padding: 12px;

text-decoration: none;

font-size: 18px;

line-height: 25px;

border-radius: 4px;

}

.header a.logo {

font-size: 25px;

font-weight: bold;

}

.header a:hover {

background-color: #ddd;

color: black;

}

.header a.active {

background-color: dodgerblue;

color: white;

}

.header-right {

float: right;

}

@media screen and (max-width: 500px) {

.header a {

float: none;

display: block;

text-align: left;

}

.header-right {

float: none;

}

}

</style>

</head>

<body>

<div class="header">

<a href="#default" class="logo">CompanyLogo</a>

<div class="header-right">

<a class="active" href="#home">Home</a>

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

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

</div>

</div>

<div style="padding-left:20px">

<h1>Responsive Header</h1>

<p>Resize the browser window to see the effect.</p>

<p>Some content..</p>

</div>

</body>

</html>

1. Create a form and validate using JavaScript

<!DOCTYPE html>

<html>

<head>

<script>

function validateForm() {

let x = document.forms["myForm"]["fname"].value;

if (x == "") {

alert("Name must be filled out");

return false;

}

}

</script>

</head>

<body>

<h2>JavaScript Validation</h2>

<form name="myForm" action="/action\_page.php" onsubmit="return validateForm()" method="post">

Name: <input type="text" name="fname">

<input type="submit" value="Submit">

</form>

</body>

</html>

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

<!DOCTYPE html>

<html>

<head>

<meta name="viewport" content="width=device-width, initial-scale=1">

<style>

body {font-family: Arial, Helvetica, sans-serif;}

/\* The Modal (background) \*/

.modal {

display: none; /\* Hidden by default \*/

position: fixed; /\* Stay in place \*/

z-index: 1; /\* Sit on top \*/

padding-top: 100px; /\* Location of the box \*/

left: 0;

top: 0;

width: 100%; /\* Full width \*/

height: 100%; /\* Full height \*/

overflow: auto; /\* Enable scroll if needed \*/

background-color: rgb(0,0,0); /\* Fallback color \*/

background-color: rgba(0,0,0,0.4); /\* Black w/ opacity \*/

}

/\* Modal Content \*/

.modal-content {

background-color: #fefefe;

margin: auto;

padding: 20px;

border: 1px solid #888;

width: 80%;

}

/\* The Close Button \*/

.close {

color: #aaaaaa;

float: right;

font-size: 28px;

font-weight: bold;

}

.close:hover,

.close:focus {

color: #000;

text-decoration: none;

cursor: pointer;

}

</style>

</head>

<body>

<h2>Modal Example</h2>

<!-- Trigger/Open The Modal -->

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

<!-- The Modal -->

<div id="myModal" class="modal">

<!-- Modal content -->

<div class="modal-content">

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

<p>Some text in the Modal..</p>

</div>

</div>

<script>

// Get the modal

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

// Get the button that opens the modal

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

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

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

// When the user clicks 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";

}

}

</script>

</body>

</html>

1. Use external js library to show slider
2. Prevent the browser when i click the form submit button

$(".ml-block-form").submit(function(){

var vals = $(this).serialize();

$.ajax({

url: "postpage.php",

method: "POST",

data: vals,

success: function(data) {

$("#formsubmit").val("Thank you!");

}

});

return false; // prevent from submit

});

<form class="ml-block-form" action="" data-code="myownID" method="post">

<input id="mainval" type="email" name="fields[email]" placeholder="Email\*">

<input id="hiddenval" name="ml-submit" value="1" />

<input id="formsubmit" type="submit" value="Get Updates!"/>

</form>

**MODULE: 4 (New Request)**

1. What is JSON

>> It is a text-based way of representing JavaScript object literals, arrays, and scalar data. JSON is relatively easy to read and write, while also easy for software to parse and generate. It is often used for serializing structured data and exchanging it over a network, typically between a server and web applications.

1. What is promises

>> A promise is an object that may produce a single value some time in the future : either a resolved value, or a reason that it's not resolved (e.g., a network error occurred). A promise may be in one of 3 possible states: fulfilled, rejected, or pending

(3) Write a program of promises and handle that promises also

<html lang="en">

<head>

<meta charset="UTF-8" />

<meta http-equiv="X-UA-Compatible" content="IE=edge" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title>Document</title>

</head>

<body>

<script>

let code = new Promise(function (resolve, reject)

{

var marks=50;

if(marks>=40)

{

resolve("You are pass");

}

else

{

reject("You are fail");

}

});

code.then((resp)=> document.write(resp)).catch((err)=> document.write(err));

</script>

(4) Use fetch method for calling an api

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<meta http-equiv="X-UA-Compatible" content="IE=edge" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title>Document</title>

<!-- CSS only -->

<link

href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"

rel="stylesheet"

integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3"

crossorigin="anonymous"

/>

</head>

<body>

<button onclick="fetchData()">Fetch Data</button>

<div class="container">

<div class="row" id="btn"></div>

</div>

<script>

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

.then(function (resp) {

return resp.json();

})

.then(function (data) {

return console.log(data);

})

.catch(function (err) {

console.log(err);

});

let btn = document.getElementById("btn");

function showData(newData)

{

for(const myData of newData)

{

let htmlData = ` <div class="col-md-3">

<div class="card">

<img src="${myData.image}" class="card-img-top" alt="" />

<div class="card-body">

<h5 class="card-title">${myData.title}</h5>

<p class="card-text">${myData.description.slice(0, 20)}...</p>

<p class="card-text">Price : ${myData.price}</p>

<a href="#" class="btn btn-primary">Order Now</a>

</div>

</div>

</div>`;

btn.insertAdjacentHTML("afterbegin", htmlData);

}

}

function fetchData() {

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

.then((response) => {

return response.json();

})

.then((data) => {

showData(data)

})

.catch((err) => console.log(err));

}

</script>

</body>

</html>

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