 1)JSON vs. XML

|  |  |  |
| --- | --- | --- |
|  | JSON | XML |
| Stands for | *JSON* means JavaScript Object Notation. | *XML* means Extensible Markup Language. |
| History | Douglas Crockford and Chip Morningstar released JSON in 2001. | The XML Working Group released XML in 1998. |
| Format | JSON uses a maplike structure with key-value pairs. | XML stores data in a tree structure with namespaces for different data categories. |
| Syntax | The syntax of JSON is more compact and easier to read and write. | The syntax of XML substitutes some characters for entity references, making it more verbose. |
| Parsing | You can parse JSON with a standard JavaScript function. | You need to parse XML with an XML parser. |
| Schema documentation | JSON is simple and more flexible. | XML is complex and less flexible. |
| Data types | JSON supports numbers, objects, strings, and Boolean arrays. | XML supports all JSON data types and additional types like Boolean, dates, images, and namespaces. |
| Ease of use | JSON has smaller file sizes and faster data transmission. | XML tag structure is more complex to write and read and results in bulky files. |
| Security | JSON is safer than XML. | You should turn off DTD when working with XML to mitigate potential security risks. |

2)3 XML and JSON files

<?xml version="1.0" encoding="UTF-8"?>

<department>

<department>

<deptid>1</deptid>

<deptname>IT</deptname>

</department>

<department>

<deptid>2</deptid>

<deptname>CSE</deptname>

</department>

<department>

<deptid>3</deptid>

<deptname>ECE</deptname>

</department>

</department>

YEAR.XML

<?xml version="1.0" encoding="UTF-8"?>

<year>

<year>

<id>1</id>

<name>firstyear</name>

</year>

<year>

<id>2</id>

<name>secondyear</name>

</year>

<year>

<id>3</id>

<name>thirdyear</name>

</year>

<year>

<id>4</id>

<name>fourthyear</name>

</year>

</year>

STUDENT.XML

<?xml version="1.0" encoding="UTF-8"?>

<student>

<student>

<id>1</id>

<name>Preethi</name>

<dept>IT</dept>

</student>

<student>

<id>2</id>

<name>satya</name>

<dept>CSE</dept>

</student>

<student>

<id>3</id>

<name>latha</name>

<dept>ECE</dept>

</student>

</student>

DEPARTMENT.JSON

{

"departments": [

{

"id": 1,

"name": "Computer Science",

"location": "Building A"

},

{

"id": 2,

"name": "Electrical Engineering",

"location": "Building B"

}

]

}

Year JSON:

{

"years": [

{

"id": 1,

"name": "Freshman"

},

{

"id": 2,

"name": "Sophomore"

}

]

}

STUDENT.JSON

{

"students": [

{

"id": 1,

"name": "John Doe",

"department\_id": 1,

"year\_id": 3

},

{

"id": 2,

"name": "Jane Smith",

"department\_id": 2,

"year\_id": 2

}

/\* Add more students here if needed \*/

]

}

**4)Difference between authentication and authorization**

| **authentication** | **Authorization** |
| --- | --- |
| In the [authentication](https://www.geeksforgeeks.org/authentication-in-computer-network/) process, the identity of users are checked for providing the access to the system. | While in [authorization](https://www.geeksforgeeks.org/what-is-aaa-authentication-authorization-and-accounting/) process, a the person’s or user’s authorities are checked for accessing the resources. |
| In the authentication process, users or persons are verified. | While in this process, users or persons are validated. |
| It is done before the authorization process. | While this process is done after the authentication process. |
| It needs usually the user’s login details. | While it needs the user’s privilege or security levels. |
| Authentication determines whether the person is user or not. | While it determines **What permission does the user have?** |
| Generally, transmit information through an ID Token. | Generally, transmit information through an Access Token. |
| The OpenID Connect (OIDC) protocol is an authentication protocol that is generally in charge of user authentication process. | The OAuth 2.0 protocol governs the overall system of user authorization process. |
| Popular Authentication Techniques-   * Password-Based Authentication * Passwordless Authentication * 2FA/MFA (Two-Factor Authentication / Multi-Factor Authentication) * [Single sign-on (SSO)](https://www.geeksforgeeks.org/introduction-of-single-sign-on-sso/) * Social authentication | Popular  Authorization Techniques-   * Role-Based Access Controls (RBAC) * [JSON web token (JWT) Authorization](https://www.geeksforgeeks.org/json-web-token-jwt/) * SAML Authorization * OpenID Authorization * OAuth 2.0 Authorization |
| The authentication credentials can be changed in part as and when required by the user. | The authorization permissions cannot be changed by user as these are granted by the owner of the system and only he/she has the access to change it. |
| The user authentication is visible at user end. | The user authorization is not visible at the user end. |
| The user authentication is identified with username, password, face recognition, retina scan, fingerprints, etc. | The user authorization is carried out through the access rights to resources by using roles that have been pre-defined. |
| **Example**: Employees in a company are required to authenticate through the network before accessing their company email. | **Example:** After an employee successfully authenticates, the system determines what information the employees are allowed to access. |

**5)create a login screen**

<html>

<head>

<title> LOGIN</title>

</head>

<body>

<body style="background-color:B2D3C2">

<form>

<h3><b><center> LOGIN </center> </b></h3>

<table border="0" align="center">

<tr>

<td>Username : </td>

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

</tr>

<tr>

<td>Password: </td>

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

</tr>

<tr>

<td><input type="button" value="submit"></td>

</tr>

</form>

</body>

</html>

**3)create a file with dept as root,year subroot,student as element**

<?xml version="1.0" encoding="UTF-8"?>

<department>

<year>

<id>1</id>

<name>firstyear</name>

<student>

<student>

<id>1</id>

<name>Preethi</name>

<dept>IT</dept>

</student>

<student>

<id>2</id>

<name>satya</name>

<dept>ECE</dept>

</student>

</student>

</year>

<year>

<id>2</id>

<name>secondyear</name>

<student>

<student>

<id>1</id>

<name>Preethi</name>

<dept>IT</dept>

</student>

<student>

<id>2</id>

<name>satya</name>

<dept>ECE</dept>

</student>

</student>

</year>

<year>

<id>3</id>

<name>thirdyear</name>

<student>

<student>

<id>1</id>

<name>Preethi</name>

<dept>IT</dept>

</student>

<student>

<id>2</id>

<name>satya</name>

<dept>ECE</dept>

</student>

</student>

</year>

<year>

<id>4</id>

<name>fourthyear</name>

<student>

<student>

<id>1</id>

<name>Preethi</name>

<dept>IT</dept>

</student>

<student>

<id>2</id>

<name>satya</name>

<dept>ECE</dept>

</student>

</student>

</year>

</department>

**6)create user creation screen by using all elements like radio button**

<html>

<head>

<title> LOGIN</title>

</head>

<body>

<body style="background-color:B2D3C2">

<form>

<h3><b><center> LOGIN </center> </b></h3>

<table border="0" align="center">

<tr>

<td>Username : </td>

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

</tr>

<tr>

<td>Password: </td>

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

</tr>

<tr>

<td><input type="button" value="submit"></td>

</tr>

</form>

</body>

</html>

<html>

<head>

<title> Registration </title>

</head>

<body>

<body style="background-color:B2D3C2">

<form>

<h3><b><center> REGISTRATION FORM </center> </b></h3>

<table border="0" align="center">

<tr>

<td>First Name : </td>

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

</tr>

<tr>

<td>Last Name : </td>

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

</tr>

<tr>

<td>Phone number: </td>

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

</tr>

<tr>

<td>Gender : </td>

<td><input type ="radio" name="g">Female

<input type ="radio" name="g">Male

<input type ="radio" name="g">Other</td>

</tr>

<tr>

<td>

<ul>List of the subjects:</td>

<td><li>Java</li>

<li>C</li>

<li>c++</li></td>

</ul>

</tr>

<tr>

<td>Email: </td>

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

</tr>

<tr>

<td>branch: </td>

<td><select name="branch">

<option>--SELECT--</option>

<option>CSE</option>

<option>EEE</option>

<option>ECE</option>

</td>

</tr>

<tr>

<td>Address: </td>

<td> <textarea rows="5" cols="5"> </textarea> </td>

</tr>

<tr>

<td>DOB: </td>

<td><input type ="date"></td>

</tr>

<tr>

<td>HOBBIES</td>

<td><input type="checkbox" name="rcb">playing cricket</input>

<input type="checkbox" name="rcb">reading books</input></td>

</tr>

<tr>

<td>Landmark: </td>

<td><input type="text" name="input" placeholder="(optional)"></td>

</tr>

</form>

</body>

</html>

8)create a html page with google map

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Google Map Example</title>

<style>

#map {

height: 400px;

width: 100%;

}

</style>

</head>

<body>

<h2>My Google Map</h2>

<div id="map"></div>

<script>

function initMap() {

// Specify the coordinates of the center of the map

var centerCoords = {lat: 40.7128, lng: -74.0060};

// Create a new map object, specifying the DOM element for display.

var map = new google.maps.Map(document.getElementById('map'), {

center: centerCoords,

zoom: 12 // Zoom level, 0 to 21

});

// Create a marker and set its position

var marker = new google.maps.Marker({

position: centerCoords,

map: map,

title: 'New York City' // Tooltip text when marker is hovered over

});

}

</script>

<!-- Replace 'YOUR\_API\_KEY' with your actual Google Maps API key -->

<script src="https://maps.googleapis.com/maps/api/js?key=YOUR\_API\_KEY&callback=initMap" async defer></script>

</body>

</html>

9)create a html page with video file

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Video Example</title>

</head>

<body>

<h2>My Video</h2>

<!-- Replace 'video.mp4' with the path to your video file -->

<video width="640" height="360" controls>

<source src="video.mp4" type="video/mp4">

Your browser does not support the video tag.

</video>

</body>

</html>

10)create a html page with audio file

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Audio Example</title>

</head>

<body>

<h2>My Audio</h2>

<!-- Replace 'audio.mp3' with the path to your audio file -->

<audio controls>

<source src="audio.mp3" type="audio/mpeg">

Your browser does not support the audio element.

</audio>

</body>

</html>

11)create a html page to upload a file

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>File Upload</title>

</head>

<body>

<h2>Upload a File</h2>

<form action="upload.php" method="post" enctype="multipart/form-data">

<input type="file" name="fileToUpload" id="fileToUpload" accept="image/\*">

<br><br>

<input type="submit" value="Upload File" name="submit">

</form>

</body>

</html>

7)list all users,update user and delete user

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>User Management</title>

<link rel="stylesheet" href="styles.css">

</head>

<body>

<h2>User Management</h2>

<!-- List Users -->

<ul id="userList"></ul>

<!-- Update User Form (Hidden by default) -->

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

<form id="updateForm" class="modal-content">

<span class="close" onclick="closeUpdateForm()">&times;</span>

<h3>Update User</h3>

<input type="text" id="updatedUserName" placeholder="New Username" required>

<button type="submit">Update</button>

</form>

</div>

<!-- Delete Confirmation Popup (Hidden by default) -->

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

<div class="modal-content">

<span class="close" onclick="closeDeleteConfirmation()">&times;</span>

<h3>Confirmation</h3>

<p>Are you sure you want to delete this user?</p>

<button id="confirmDelete">Yes</button>

<button onclick="closeDeleteConfirmation()">No</button>

</div>

</div>

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

</body>

</html>

.modal {

display: none;

position: fixed;

z-index: 1;

left: 0;

top: 0;

width: 100%;

height: 100%;

overflow: auto;

background-color: rgba(0,0,0,0.4);

}

.modal-content {

background-color: #fefefe;

margin: 15% auto;

padding: 20px;

border: 1px solid #888;

width: 30%;

}

.close {

color: #aaa;

float: right;

font-size: 28px;

font-weight: bold;

}

.close:hover,

.close:focus {

color: black;

text-decoration: none;

cursor: pointer;

}

// Sample user data

var users = [

{ id: 1, name: 'User 1' },

{ id: 2, name: 'User 2' },

{ id: 3, name: 'User 3' }

];

// Function to display users

function displayUsers() {

var userList = document.getElementById('userList');

userList.innerHTML = '';

users.forEach(function(user) {

var li = document.createElement('li');

li.textContent = user.name;

li.innerHTML += ' <button onclick="openUpdateForm(' + user.id + ')">Update</button>';

li.innerHTML += ' <button onclick="openDeleteConfirmation(' + user.id + ')">Delete</button>';

userList.appendChild(li);

});

}

// Function to open update form

function openUpdateForm(userId) {

var user = users.find(function(user) {

return user.id === userId;

});

document.getElementById('updatedUserName').value = user.name;

document.getElementById('updateFormContainer').style.display = 'block';

}

// Function to close update form

function closeUpdateForm() {

document.getElementById('updateFormContainer').style.display = 'none';

}

// Function to update user

document.getElementById('updateForm').addEventListener('submit', function(event) {

event.preventDefault();

var newName = document.getElementById('updatedUserName').value;

// Update user logic here (e.g., call a backend API)

console.log('Updated user to:', newName);

closeUpdateForm();

});

// Function to open delete confirmation popup

function openDeleteConfirmation(userId) {

document.getElementById('confirmDelete').onclick = function() {

// Delete user logic here (e.g., call a backend API)

console.log('Deleted user with ID:', userId);

closeDeleteConfirmation();

};

document.getElementById('deleteConfirmation').style.display = 'block';

}

// Function to close delete confirmation popup

function closeDeleteConfirmation() {

document.getElementById('deleteConfirmation').style.display = 'none';

}

// Display users when the page loads

displayUsers();