# 1.Differences between JSON and XML:

| **JSON** | **XML** |
| --- | --- |
| It is [JavaScript Object Notation](https://www.geeksforgeeks.org/javascript-json/) | It is [Extensible markup language](https://www.geeksforgeeks.org/xml-basics/) |
| It is based on JavaScript language. | It is derived from [SGML](https://www.geeksforgeeks.org/what-is-sgml/). |
| It is a way of representing objects. | It is a markup language and uses tag structure to represent data items. |
| It does not provides any support for namespaces. | It supports [namespaces](https://www.geeksforgeeks.org/javascript-namespace/). |
| It supports array. | It doesn’t supports [array](https://www.geeksforgeeks.org/array-data-structure/). |
| Its files are very easy to read as compared to XML. | Its documents are comparatively difficult to read and interpret. |
| It doesn’t use end tag. | It has start and end tags. |
| It is less secured. | It is more secured than JSON. |
| It doesn’t supports comments. | It supports comments. |
| It supports only [UTF-8 encoding](https://www.geeksforgeeks.org/understanding-character-encoding/). | It supports various encoding. |

2)Create 3 XML and JSON files for department,year,student

# Department.xml

<departments>

<department>

<id>1</id>

<name>Computer Science</name>

<faculty>Engineering</faculty>

</department>

<department>

<id>2</id>

<name>Mathematics</name>

<faculty>Science</faculty>

</department>

<department>

<id>3</id>

<name>English Literature</name>

<faculty>Arts</faculty>

</department>

</departments>

# Student.xml

<students>

<student>

<id>101</id>

<name>John Doe</name>

<departmentId>1</departmentId>

<yearId>3</yearId>

</student>

<student>

<id>102</id>

<name>Jane Smith</name>

<departmentId>2</departmentId>

<yearId>2</yearId>

</student>

<student>

<id>103</id>

<name>Emily Johnson</name>

<departmentId>3</departmentId>

<yearId>4</yearId>

</student>

</students>

# Year.xml

<years>

<year>

<id>1</id>

<name>Freshman</name>

</year>

<year>

<id>2</id>

<name>Sophomore</name>

</year>

<year>

<id>3</id>

<name>Junior</name>

</year>

<year>

<id>4</id>

<name>Senior</name>

</year>

</years>

# Department.json

{

"departments": [

{

"id": 1,

"name": "Computer Science",

"faculty": "Engineering"

},

{

"id": 2,

"name": "Mathematics",

"faculty": "Science"

},

{

"id": 3,

"name": "English Literature",

"faculty": "Arts"

}

]

}

# Student.json

{

"students": [

{

"id": 1,

"name": "Alice Johnson",

"age": 20,

"major": "Computer Science"

},

{

"id": 2,

"name": "Bob Smith",

"age": 21,

"major": "Mechanical Engineering"

},

{

"id": 3,

"name": "Charlie Brown",

"age": 22,

"major": "Physics"

}

]

}

# Year.json

{

"years": [

{

"id": 1,

"year": "Freshman"

},

{

"id": 2,

"year": "Sophomore"

},

{

"id": 3,

"year": "Junior"

},

{

"id": 4,

"year": "Senior"

}

]

}

3)Create a file with depertment as root,year as subroot and student as an element

{

"departments": [

{

"name": "Computer Science",

"years": [

{

"year": "Junior",

"students": [

{

"id": 1,

"name": "Alice Johnson",

"age": 20,

"major": "Computer Science"

}

]

}

]

},

{

"name": "Mechanical Engineering",

"years": [

{

"year": "Senior",

"students": [

{

"id": 2,

"name": "Bob Smith",

"age": 21,

"major": "Mechanical Engineering"

}

]

}

]

},

{

"name": "Physics",

"years": [

{

"year": "Sophomore",

"students": [

{

"id": 3,

"name": "Charlie Brown",

"age": 22,

"major": "Physics"

}

]

}

]

}

]

}

4)Difference between Authorization and Authentication

| **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  1. Passwordless Authentication 2. 2FA/MFA (Two-Factor Authentication / Multi-Factor Authentication) 3. [Single sign-on (SSO)](https://www.geeksforgeeks.org/introduction-of-single-sign-on-sso/) 4. Social authentication | Popular  Authorization Techniques-   * Role-Based Access Controls (RBAC)  1. [JSON web token (JWT) Authorization](https://www.geeksforgeeks.org/json-web-token-jwt/) 2. SAML Authorization 3. OpenID Authorization 4. 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

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Login</title>

</head>

<body>

<form>

<label for="username">Username or Email:</label><br>

<input type="text" id="username" name="username"><br>

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

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

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

</form>

</body>

</html>

6)Create a User Creation Screen by using all elements in it(like List, Radio button, Drop down, CheckBox)

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>User Creation</title>

</head>

<body>

<h2>User Creation</h2>

<form>

Full Name: <input type="text" name="fullname"><br><br>

Email: <input type="email" name="email"><br><br>

Gender:

<input type="radio" id="male" name="gender" value="male">

<label for="male">Male</label>

<input type="radio" id="female" name="gender" value="female">

<label for="female">Female</label><br><br>

Country:

<select name="country">

<option value="USA">USA</option>

<option value="Canada">Canada</option>

<option value="UK">UK</option>

<option value="Australia">Australia</option>

</select><br><br>

Interests:

<input type="checkbox" id="music" name="interests" value="music">

<label for="music">Music</label>

<input type="checkbox" id="sports" name="interests" value="sports">

<label for="sports">Sports</label>

<input type="checkbox" id="movies" name="interests" value="movies">

<label for="movies">Movies</label><br><br>

Comments: <br>

<textarea name="comments" rows="4" cols="50"></textarea><br><br>

<input type="submit" value="Create User">

</form>

</body>

</html>

7)List all Users,Update user and Delete user(Popup for confirmation eg:Are you sure do you want to delete)

<!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>

<script>

function deleteUserConfirmation(userId) {

if (confirm("Are you sure you want to delete this user?")) {

deleteUser(userId);

}

}

function deleteUser(userId) {

console.log("Deleting user with ID: " + userId);

}

function updateUser(userId) {

console.log("Updating user with ID: " + userId);

}

</script>

</head>

<body>

<h1>User Management</h1>

<!-- List of Users -->

<h2>List of Users</h2>

<ul id="userList">

<li>User 1 <button onclick="updateUser(1)">Update</button> <button onclick="deleteUserConfirmation(1)">Delete</button></li>

<li>User 2 <button onclick="updateUser(2)">Update</button> <button onclick="deleteUserConfirmation(2)">Delete</button></li>

<!-- Add more users here -->

</ul>

</body>

</html>

8)Create a HTML page with Google Map

<!DOCTYPE html>

<html>

<head>

<title>Google Map Example</title>

<style>

#map {

height: 400px;

width: 100%;

}

</style>

<script>

function initMap() {

var location = { lat: -34.397, lng: 150.644 };

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

center: location,

zoom: 8

});

var marker = new google.maps.Marker({

position: location,

map: map,

title: 'Hello World!'

});

}

</script>

</head>

<body>

<h3>My Google Map</h3>

<!-- The map div element -->

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

<!-- Load the Google Maps API -->

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

</head>

<body>

<h2>My Video</h2>

<!-- Replace the src attribute with your YouTube video's embed URL -->

<iframe width="600" height="337" src="https://youtu.be/t8UkfCF5reU?si=lUzURaEjKBOxcMcs" frameborder="0" allowfullscreen></iframe>

</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>Embedded Audio Example</title>

</head>

<body>

<h1>My Audio</h1>

<audio controls>

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

<source src="path/to/your/audio.ogg" type="audio/ogg">

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 Example</title>

</head>

<body>

<h1>Upload a File</h1>

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

<label for="file">Choose a file to upload:</label>

<input type="file" id="file" name="file" required>

<br><br>

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

</form>

</body>

</html>