1.differences between json and xml:

|  |  |  |
| --- | --- | --- |
| feature | JSON | XML |
| SYNTAX | Lightweight, uses curly braces {}, square brackets [], and key-value pairs separated by colons : | Markup language, uses tags <tag>, </tag> and attributes |
| READABILITY | More concise, easier for humans to read | More verbose, similar to HTML structure |
| DATA TYPES | Supports strings, numbers, booleans, arrays, and objects | Everything is treated as text, with attributes for additional data |
| VALIDATION | JSON Schema | XML Schema (XSD) or Document Type Definition (DTD) |
| USAGE | Commonly used in web APIs, configuration files | Used in document storage, configuration files, and protocols like SOAP |
| EXTENSIBILITY | Less flexible for adding metadata | Highly extensible with namespaces |
| PARSING SPEED | Generally faster to parse | Slower parsing, more complex |
| TRANFORMATIONS | Not inherently supported, relies on external tools | Strong support for transformations using XSLT |
| COMMENTS | Does not support comments | Supports comments using <!-- comment --> |
| INTEROPARABILITY | Easily interoperable with JavaScript and other modern languages | Well-supported across a variety of platforms and languages, especially in enterprise environment |
| DATA STRUCTURE | Ideal for simple hierarchical data | Suitable for complex hierarchical data structures |
| HUMAN READABILITY | Designed to be easily readable by humans | Can become complex and harder to read for large documents |

2. CREATE 3 XML AND JSON FILES FOR DEPARTMENT, YEAR, STUDENT ?

XML FILES :

**1. Department.xml**

xml

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

<departments>

<department>

<id>1</id>

<name>Computer Science</name>

<head>Dr. John Doe</head>

</department>

<department>

<id>2</id>

<name>Mathematics</name>

<head>Dr. Jane Smith</head>

</department>

</departments>

**2. Year.xml**

xml

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

<years>

<year>

<id>1</id>

<name>Freshman</name>

<startYear>2021</startYear>

<endYear>2022</endYear>

</year>

<year>

<id>2</id>

<name>Sophomore</name>

<startYear>2022</startYear>

<endYear>2023</endYear>

</year>

</years>

**3. Student.xml**

xml

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

<students>

<student>

<id>101</id>

<name>Emily Brown</name>

<department>Computer Science</department>

<year>Freshman</year>

</student>

<student>

<id>102</id>

<name>Michael Green</name>

<department>Mathematics</department>

<year>Sophomore</year>

</student>

</students>

JSON Files

**1. Department.json**

json

{

"departments": [

{

"id": 1,

"name": "Computer Science",

"head": "Dr. John Doe"

},

{

"id": 2,

"name": "Mathematics",

"head": "Dr. Jane Smith"

}

]

}

**2. Year.json**

json

{

"years": [

{

"id": 1,

"name": "Freshman",

"startYear": 2021,

"endYear": 2022 },

{

"id": 2,

"name": "Sophomore",

"startYear": 2022,

"endYear": 2023

}

]

}

**3. Student.json**

json

{

"students": [

{

"id": 101,

"name": "Emily Brown",

"department": "Computer Science",

"year": "Freshman"

},

{

"id": 102,

"name": "Michael Green",

"department": "Mathematics",

"year": "Sophomore"

}

]

}

3. CREATE A FILE WITH DEPARTMENT AS ROOT , YEAR AS SUBROOT , AND STUDENT AS ELEMENT ?

XML File

xml

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

<department>

<id>1</id>

<name>Computer Science</name>

<head>K . RAMESHWARIAH</head>

<years>

<year>

<id>1</id>

<name>THANUSREE</name>

<startYear>2021</startYear>

<endYear>2024</endYear>

<students>

<student>

<id>101</id>

<name>JAYANTHI</name>

</student>

<student>

<id>102</id>

<name>Michael </name>

</student>

</students>

</year>

<year>

<id>2</id>

<name>Sophomore</name>

<startYear>2022</startYear>

<endYear>2023</endYear>

<students>

<student>

<id>103</id>

<name>Laura White</name>

</student>

<student>

<id>104</id>

<name>James Black</name>

</student>

</students>

</year>

</years>

</department>

**JSON File**

json

{

"department": {

"id": 1,

"name": "Computer Science",

"head": "Dr. John Doe",

"years": [

{

"id": 1,

"name": "Freshman",

"startYear": 2021,

"endYear": 2022,

"students": [

{

"id": 101,

"name": "Emily Brown"

},

{

"id": 102,

"name": "Michael Green"

}

]

},

{

"id": 2,

"name": "Sophomore",

"startYear": 2022,

"endYear": 2023,

"students": [

{

"id": 103,

"name": "Laura White"

},

{

"id": 104,

"name": "James Black"

}

]

}

]

}

}

4 . DIFFERENCES BETWEEN AUTHORIZATION AND AUTHENTICATION ?

|  |  |  |
| --- | --- | --- |
| ASPECT | AUTHENTICATION | AUTHORIZATION |
| DEFINATION | Verifying the identity of a user or entity | Determining what resources and actions an authenticated user is allowed to access |
| PURPOSE | To confirm that the user or entity is who they claim to be | To control access levels and permissions for resources and actions |
| PROCESS | Validation of credentials (e.g., passwords, biometrics, tokens) | Checking permissions and roles assigned to the user against the resources they are attempting to access |
| WHEN IT OCCURS | Before authorization | After authentication |
| DETERMINES | Who the user or entity is | What the user or entity is allowed to do |
| EXAMPLES | Logging in with a password, biometric scan, using an OTP | Granting access to specific files, allowing actions based on user roles (e.g., admin, editor, viewer) |
| OUTCOME | Access granted if identity is confirmed; access denied if not | Access granted or denied based on permissions; access level varies depending on user roles |
| SCOPE | Identity verification | Access control and permission management |
| DEPENDENCIES | Requires user credentials | Requires authenticated identity |
| USECASES | Logging into a system, verifying identity with a fingerprint | Controlling which users can read, write, delete, or modify resources |
| FAILURE RESULTS | Access denied due to failed identity verification | Access denied or restricted due to insufficient permissions |

5 . CREATE A LOGIN SCREEN ?

**HTML (login.html)**

html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Login Screen</title>

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

</head>

<body>

<div class="login-container">

<h2>Login</h2>

<form action="login\_action.html" method="post">

<div class="input-group">

<label for="username">Username</label>

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

</div>

<div class="input-group">

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

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

</div>

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

</form>

</div>

</body>

</html>

**CSS (styles.css)**

```css

body {

font-family: Arial, sans-serif;

background-color: #f0f0f0;

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

margin: 0;

}

.login-container {

background-color: white;

padding: 20px;

border-radius: 8px;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

width: 300px;

text-align: center;

}

h2 {

margin-bottom: 20px;

}

.input-group {

margin-bottom: 15px;

text-align: left;

}

label {

display: block;

margin-bottom: 5px;

font-weight: bold;

}

input[type="text"],

input[type="password"] {

width: 100%;

padding: 10px;

border: 1px solid #ccc;

border-radius: 4px;

box-sizing: border-box;

}

button {

width: 100%;

padding: 10px;

background-color: #4CAF50;

color: white;

border: none;

border-radius: 4

6. CREATE A USER CREATION SCREEN BY USING ALL ELEMENTS IN IT [ LIKE list,radio button,drop down,check box]?

**HTML (user\_creation.html)**

html

<!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>

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

</head>

<body>

<div class="form-container">

<h2>Create User</h2>

<form action="create\_user\_action.html" method="post">

<div class="input-group">

<label for="username">Username</label>

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

</div>

<div class="input-group">

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

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

</div>

<div class="input-group">

<label for="email">Email</label>

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

</div>

<div class="input-group">

<label for="gender">Gender</label>

<div>

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

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

</div>

<div>

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

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

</div>

<div>

<input type="radio" id="other" name="gender" value="other" required>

<label for="other">Other</label>

</div>

</div>

<div class="input-group">

<label for="role">Role</label>

<select id="role" name="role" required>

<option value="admin">Admin</option>

<option value="editor">Editor</option>

<option value="viewer">Viewer</option>

</select>

</div>

<div class="input-group">

<label for="hobbies">Hobbies</label>

<div>

<input type="checkbox" id="reading" name="hobbies" value="reading">

<label for="reading">Reading</label>

</div>

<div>

<input type="checkbox" id="traveling" name="hobbies" value="traveling">

<label for="traveling">Traveling</label>

</div>

<div>

<input type="checkbox" id="gaming" name="hobbies" value="gaming">

<label for="gaming">Gaming</label>

</div>

</div>

<div class="input-group">

<label for="country">Country</label>

<select id="country" name="country" required>

<option value="usa">United States</option>

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

<option value="uk">United Kingdom</option>

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

</select>

</div>

<button type="submit">Create User</button>

</form>

</div>

</body>

</html>

**CSS (styles.css)**

css

body {

font-family: Arial, sans-serif;

background-color: #f0f0f0;

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

margin: 0;

}

.form-container {

background-color: white;

padding: 20px;

border-radius: 8px;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

width: 300px;

text-align: center;

}

h2 {

margin-bottom: 20px;

}

.input-group {

margin-bottom: 15px;

text-align: left;

}

label {

display: block;

margin-bottom: 5px;

font-weight: bold;

}

input[type="text"],

input[type="password"],

input[type="email"],

select {

width: 100%;

padding: 10px;

border: 1px solid #ccc;

border-radius: 4px;

box-sizing: border-box;

}

input[type="radio"],

input[type="checkbox"] {

margin-right: 10px;

}

button {

width: 100%;

padding: 10px;

background-color: #4CAF50;

color: white;

border: none;

border-radius: 4px;

cursor: pointer;

font-size: 16px;

}

button:hover {

background-color: #45a049;

}

7 . list all users ,update user and delete user(popup for confirmation eg: are you sure do you want to delete ) ?

<!DOCTYPE html>

<html>

<head>

<title>User Management</title>

<style>

/\* Add your CSS styles here \*/

table {

width: 50%;

margin: 20px auto;

border-collapse: collapse;

}

table, th, td {

border: 1px solid black;

}

th, td {

padding: 10px;

text-align: left;

}

#userForm {

width: 50%;

margin: 20px auto;

}

#userForm div {

margin-bottom: 10px;

}

#popup {

display: none;

position: fixed;

left: 50%;

top: 50%;

transform: translate(-50%, -50%);

border: 1px solid black;

padding: 20px;

background-color: white;

}

</style>

</head>

<body>

<h1>User Management</h1>

<div id="userForm">

<div>

<label for="username">Username:</label>

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

</div>

<div>

<label for="email">Email:</label>

<input type="email" id="email">

</div>

<div>

<button onclick="addUser()">Add User</button>

<button onclick="updateUser()">Update User</button>

</div>

</div>

<table>

<thead>

<tr>

<th>ID</th>

<th>Username</th>

<th>Email</th>

<th>Actions</th>

</tr>

</thead>

<tbody id="userTable">

<!-- User rows will go here -->

</tbody>

</table>

<div id="popup">

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

<button onclick="confirmDelete()">Yes</button>

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

</div>

<script>

let users = [];

let selectedUserId = null;

function addUser() {

const username = document.getElementById('username').value;

const email = document.getElementById('email').value;

if (username && email) {

users.push({ id: Date.now(), username, email });

renderUsers();

}

}

function updateUser() {

if (selectedUserId !== null) {

const username = document.getElementById('username').value;

const email = document.getElementById('email').value;

users = users.map(user => user.id === selectedUserId ? { id: user.id, username, email } : user);

selectedUserId = null;

renderUsers();

}

}

function editUser(id) {

const user = users.find(user => user.id === id);

if (user) {

selectedUserId = id;

document.getElementById('username').value = user.username;

document.getElementById('email').value = user.email;

}

}

function deleteUser(id) {

selectedUserId = id;

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

}

function confirmDelete() {

users = users.filter(user => user.id !== selectedUserId);

selectedUserId = null;

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

renderUsers();

}

function closePopup() {

selectedUserId = null;

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

}

function renderUsers() {

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

userTable.innerHTML = '';

users.forEach(user => {

userTable.innerHTML += `

<tr>

<td>${user.id}</td>

<td>${user.username}</td>

<td>${user.email}</td>

<td>

<button onclick="editUser(${user.id})">Edit</button>

<button onclick="deleteUser(${user.id})">Delete</button>

</td>

</tr>

`;

});

}

</script>

</body>

</html>

8 . create a HTML page with google map ?

To create an HTML page with an embedded Google Map, you'll need to use the Google Maps JavaScript API. Here's a simple example to get you started:

1. \*Sign up for a Google Maps API key\*:

- Go to the [Google Cloud Platform Console](https://console.cloud.google.com/).

- Create a new project or select an existing one.

- Enable the Maps JavaScript API for your project.

- Create credentials to obtain an API key.

2. \*Create the HTML page\*:

html

<!DOCTYPE html>

<html>

<head>

<title>Google Map Example</title>

<style>

/\* Set the size of the map div element \*/

#map {

height: 400px;

width: 100%;

}

</style>

<script>

function initMap() {

// The location you want to display (latitude and longitude)

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

// Create a map object centered at the specified location

const map = new google.maps.Map(document.getElementById("map"), {

zoom: 8,

center: location,

});

// Create a marker at the specified location

const marker = new google.maps.Marker({

position: location,

map: map,

});

}

</script>

</head>

<body>

<h1>My Google Map</h1>

<!-- The div element where the map will be displayed -->

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

<!-- Load the Google Maps API script with your API key -->

<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 ?

To create an HTML page that includes a video file, you can use the HTML5 <video> element. Here's a simple example:

html

<!DOCTYPE html>

<html>

<head>

<title>Video Example</title>

<style>

/\* Optional: Add some basic styling \*/

body {

font-family: Arial, sans-serif;

text-align: center;

margin-top: 50px;

}

video {

width: 80%;

max-width: 800px;

height: auto;

}

</style>

</head>

<body>

<h1>My Video</h1>

<!-- The video element -->

<video controls>

<source src="path/to/your/video.mp4" type="video/mp4">

Your browser does not support the video tag.

</video>

</body>

</html>

10 . create a HTML page with audio file ?

To create an HTML page that includes an audio file, you can use the HTML5 <audio> element. Here's a simple example:

html

<!DOCTYPE html>

<html>

<head>

<title>Audio Example</title>

<style>

/\* Optional: Add some basic styling \*/

body {

font-family: Arial, sans-serif;

text-align: center;

margin-top: 50px;

}

audio {

width: 80%;

max-width: 600px;

}

</style>

</head>

<body>

<h1>My Audio</h1>

<!-- The audio element -->

<audio controls>

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

Your browser does not support the audio element.

</audio>

</body>

</html>

10 . create a HTML page with audio file ?