UNIT 3

JSON

1.INTRODUCTION

- JSON stands for **J**ava**S**cript **O**bject **N**otation
- JSON is a **text format** for storing and transporting data
- JSON is "self-describing" and easy to understand
- JSON is a lightweight data-interchange format
- JSON is plain text written in JavaScript object notation
- JSON is used to send data between computers
- JSON is language independent

JSON is built on two structures:

- A collection of name/value pairs. In various languages, this is realized as an *object*, record, struct, dictionary, hash table, keyed list, or associative array.
- An ordered list of values. In most languages, this is realized as an *array*, vector, list, or sequence.

An object is an unordered set of name/value pairs. An object begins with {left brace and ends with }right brace. Each name is followed by :colon and the name/value pairs are separated by ,comma.

JSON Example

This example is a JSON string:

'{"name":"John", "age":30, "car":null}'

It defines an object with 3 properties:

- name
- age

Each property has a value.

If you parse the JSON string with a JavaScript program, you can access the data as an object:

```
let personName = obj.name;
let personAge = obj.age;
```

The JSON syntax is derived from JavaScript object notation, but the JSON format is text only.Code for reading and generating JSON exists in many programming languages.

JSON Syntax Rules

JSON syntax is derived from JavaScript object notation syntax:

- Data is in name/value pairs
- Data is separated by commas
- Curly braces hold objects
- Square brackets hold arrays

2.JSON VS XMI

- XML is much more difficult to parse than JSON.
- JSON is parsed into a ready-to-use JavaScript object.

For AJAX applications, JSON is faster and easier than XML:

Using XML

- Fetch an XML document
- Use the XML DOM to loop through the document
- Extract values and store in variables

Using JSON

- Fetch a JSON string
- JSON.Parse the JSON string

Example

Individual examples of XML and JSON −

JSON

```
{
  "company": Volkswagen,
  "name": "Vento",
  "price": 800000
}
```

XML

```
<car>
  <company>Volkswagen</company>
  <name>Vento</name>
  <price>800000</price>
</car>
```

JSON and XML are human readable formats and are language independent. They both have support for creation, reading and decoding in real world situations. We can compare JSON with XML, based on the following factors −

Verbose

XML is more verbose than JSON, so it is faster to write JSON for programmers.

Arrays Usage

XML is used to describe the structured data, which doesn't include arrays whereas JSON include arrays.

Parsing

JavaScript's *eval* method parses JSON. When applied to JSON, eval returns the described object.

3.JSON EXAMPLE

PROGRAM

```
{
  "book": [

    {
      "id": "01",
      "language": "Java",
      "edition": "third",
      "author": "Herbert Schildt"
    },

    {
      "id": "07",
      "language": "C++",
      "edition": "second",
      "author": "E.Balagurusamy"
    }

  ]
}
```

JSON supports the following two data structures −

- **Collection of name/value pairs** − This Data Structure is supported by different programming languages.
- **Ordered list of values** − It includes array, list, vector or sequence etc.

4.JSON OBJECT

- JSON object literals are surrounded by curly braces {}.
- JSON object literals contains key/value pairs.
- Keys and values are separated by a colon.

Keys must be strings, and values must be a valid JSON data type:

- string
- number
- object
- array
- boolean
- null

Each key/value pair is separated by a comma.

- It is a common mistake to call a JSON object literal "a JSON object".
- JSON cannot be an object. JSON is a string format.
- The data is only JSON when it is in a string format. When it is converted to a JavaScript variable, it becomes a JavaScript object.

Creating Simple Objects

JSON objects can be created with JavaScript. Let us see the various ways of creating JSON objects using JavaScript −

- Creation of an empty Object −

```
var JSONObj = {};
```

- Creation of a new Object −

```
var JSONObj = new Object();
```

- Creation of an object with attribute **bookname** with value in string, attribute **price** with numeric value. Attribute is accessed by using '.' Operator −

```
var JSONObj = { "bookname ":"VB BLACK BOOK", "price":500 };
```

This is an example that shows creation of an object in javascript using JSON, save the below code as json_object.html −

```
<html>
  <head>
    <title>Creating Object JSON with JavaScript</title>
    <script language = "javascript" >
      var JSONObj = { "name" : "sample.com", "year"  : 2023};

      document.write("<h1>JSON with JavaScript example</h1>");
      document.write("<br>");
      document.write("<h3>Website Name = "+JSONObj.name+"</h3>");
      document.write("<h3>Year = "+JSONObj.year+"</h3>");
    </script>
  </head>

  <body>
  </body>
</html>
```

5.JSON ARRAY

- Arrays in JSON are almost the same as arrays in JavaScript.
- In JSON, array values must be of type string, number, object, array, boolean or *null*.
- In JavaScript, array values can be all of the above, plus any other valid JavaScript expression, including functions, dates, and *undefined.*

JavaScript Arrays

You can create a JavaScript array from a literal:

Example

```
myArray = ["Ford", "BMW", "Fiat"];
```

Creating Array Objects

The following example shows creation of an array object in javascript using JSON, save the below code as **json_array_object.htm** −

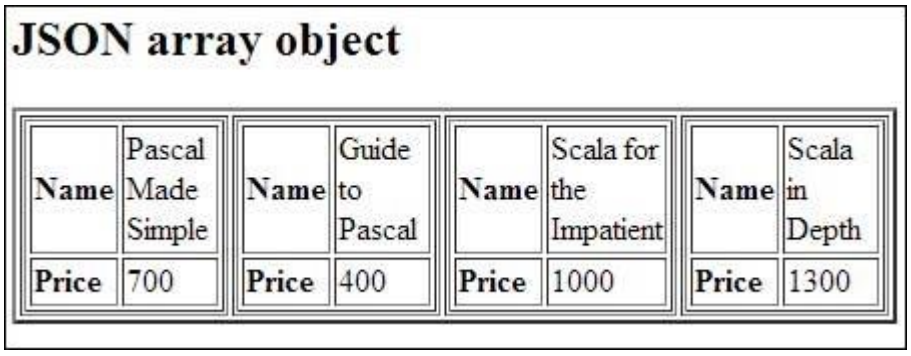
PROGRAM

```
<html>
  <head>
    <title>Creation of array object in javascript using JSON</title>
    <script language = "javascript" >
      document.writeln("<h2>JSON array object</h2>");
      var books = { "Pascal" : [
        { "Name"  : "Pascal Made Simple", "price" : 700 },
        { "Name"  : "Guide to Pascal", "price" : 400 }],
```

```
   "Scala"  : [
      { "Name"  : "Scala for the Impatient", "price" : 1000 },
      { "Name"  : "Scala in Depth", "price" : 1300 }]
   }

   var i = 0
   document.writeln("<table border = '2'><tr>");

   for(i = 0;i<books.Pascal.length;i++){
      document.writeln("<td>");
      document.writeln("<table border = '1' width = 100 >");
      document.writeln("<tr><td><b>Name</b></td><td width = 50>" +
books.Pascal[i].Name+"</td></tr>");
      document.writeln("<tr><td><b>Price</b></td><td width = 50>" +
books.Pascal[i].price +"</td></tr>");
      document.writeln("</table>");
      document.writeln("</td>");
   }

   for(i = 0;i<books.Scala.length;i++){
      document.writeln("<td>");
      document.writeln("<table border = '1' width = 100 >");
```

```
        document.writeln("<tr><td><b>Name</b></td><td width = 50>" +
books.Scala[i].Name+"</td></tr>");

        document.writeln("<tr><td><b>Price</b></td><td width = 50>" +
books.Scala[i].price+"</td></tr>");

        document.writeln("</table>");

        document.writeln("</td>");

      }

      document.writeln("</tr></table>");

    </script>

  </head>

  <body>

  </body>

</html>
```

OUTPUT



6.JSON COMMENTS

- JSON doesn't support comments. It is not a standard.
- But you can do some tricks such as adding extra attribute for comment in JSON object, for example:

```
{
   "employee": {
      "name":       "Bob",
      "salary":      56000,
      "comments":    "He is a nice man"
   }
}
```

Here, "comments" attribute can be treated as comment.

7.JSON EXAMPLE USING PHP

Function	Libraries
json_encode	Returns the JSON representation of a value.
json_decode	Decodes a JSON string.
json_last_error	Returns the last error occurred.

Encoding JSON in PHP (json_encode)

PHP **json_encode()** function is used for encoding JSON in PHP. This function returns the JSON representation of a value on success or FALSE on failure.

Syntax

string json_encode ($value [, $options = 0])

Parameters

- **value** − The value being encoded. This function only works with UTF-8 encoded data.
- **options** − This optional value is a bitmask consisting of JSON_HEX_QUOT, JSON_HEX_TAG, JSON_HEX_AMP, JSON_HEX_APOS, JSON_NUMERIC_CHECK, JSON_PRETTY_PRINT, JSON_UNESCAPED_SLASHES, JSON_FORCE_OBJECT.

common use of JSON is to read data from a web server, and display the data in a web page.

Decoding JSON in PHP (json_decode)

PHP json_decode() function is used for decoding JSON in PHP. This function returns the value decoded from json to appropriate PHP type.

Syntax

mixed json_decode ($json [,$assoc = false [, $depth = 512 [, $options = 0]]])

Paramaters

- **json_string** − It is an encoded string which must be UTF-8 encoded data.
- **assoc** − It is a boolean type parameter, when set to TRUE, returned objects will be converted into associative arrays.
- **depth** − It is an integer type parameter which specifies recursion depth
- **options** − It is an integer type bitmask of JSON decode, JSON_BIGINT_AS_STRING is supported.

Example

The PHP File

PHP has some built-in functions to handle JSON.

Objects in PHP can be converted into JSON by using the PHP function json_encode():

PHP file

```
<?php
$myObj->name = "John";
$myObj->age = 30;
$myObj->city = "New York";

$myJSON = json_encode($myObj);

echo $myJSON;
?>
```

8.JSON EXAMPLE USING JAVA

The **json.simple** library allows us to read and write JSON data in Java. In other words, we can encode and decode JSON object in java using json.simple library.

The org.json.simple package contains important classes for JSON API.

- JSONValue
- JSONObject
- JSONArray
- JsonString
- JsonNumber

Java JSON Encode

Let's see a simple example to encode JSON object in java.

```
import org.json.simple.JSONObject;
public class JsonExample1{
public static void main(String args[]){
JSONObject obj=new JSONObject();
  obj.put("name","sonoo");
  obj.put("age",new Integer(27));
  obj.put("salary",new Double(600000));
   System.out.print(obj);
}}
```

Output:

```
{"name":"sonoo","salary":600000.0,"age":27}
```

9.JSON Example using AJAX

AJAX provides facility to get response asynchronously. It doesn't reload the page and saves bandwidth.Any data that is updated using AJAX can be stored using the JSON format on the web server. AJAX is used so that javascript can retrieve these JSON files when necessary, parse them, and perform one of the following operations −

- Store the parsed values in the variables for further processing before displaying them on the webpage.
- It directly assigns the data to the DOM elements in the webpage, so that they are displayed on the website.

Advantages of using JSON instead of XML in AJAX:

- The Code of JSON will be short in comparison to XML that's why transferring data will be smooth
- JSON is easier to understand in comparison with XML
- In JSON we can easily represent a NULL Value.

AJAX works on Request and Response means AJAX will Request anything from the server and the server will give the Response back to AJAX. We have a built-in Object of Javascript known as "XMLHttpRequest" to send Responses and get Requests. There are some of the properties and methods of XMLHttpRequest, are described below:

- **new XMLHttpRequest:** It will create a new object by which we can send requests and receive responses.
- **open():** It will specify any request. It takes various parameters like types of requests(GET or POST), location of server file, etc.
- send(): It is used to send a request to the server. It contains a string as a parameter if it is used for POST requests.
- **onload**: It is the property of XMLHttpRequest object which is used to define a function that will be called on complete loading of data.
- **onreadystatechange**: It is the property of XMLHttpRequest which is used to define a function that will be called on change of ready state. readystate is also a property of the XMLHttpRequest object.
- **readystate**: It is used to represent the status of the request. It can contain 5 values and every value has a different meaning.

- 0 means the request is not initialized
- 1 means Connection with the server is established
- 2 means the request is received
- 3 means processing the request
- 4 means the request is finished

- **status**: It is the property of XMLHttpRequest which is used to represent the status number of any request
- **responseText:** It is the property of XMLHttpRequest which is used to return the data of the response as a string

Steps for Sending the request with AJAX:

- Create a new XMLHttpRequest.
- Use the open() method of XMLHttpRequest to specify the request
- Use send() method of XMLHttpRequest to send request to the server

PROGRAM

```
<html>
<head>
<meta content="text/html; charset=utf-8">
<title>AJAX JSON by Javatpoint</title>
<script type="application/javascript">
function load()
{
  var url = "http://date.jsontest.com/";//use any url that have json data
  var request;

  if(window.XMLHttpRequest){
   request=new XMLHttpRequest();//for Chrome, mozilla etc
  }
  else if(window.ActiveXObject){
   request=new ActiveXObject("Microsoft.XMLHTTP");//for IE only
  }
  request.onreadystatechange  = function(){
    if (request.readyState == 4  )
```

```
        {
          var jsonObj = JSON.parse(request.responseText);//JSON.parse() returns JSON objec
    t
          document.getElementById("date").innerHTML =  jsonObj.date;
          document.getElementById("time").innerHTML = jsonObj.time;
        }
      }
      request.open("GET", url, true);
      request.send();
    }
    </script>
    </head>
    <body>

    Date: <span id="date"></span><br/>
    Time: <span id="time"></span><br/>

    <button type="button" onclick="load()">Load Information</button>
    </body>
    </html>
```

Output:

```
Date:
Time:
Load Information
```