JSON Syntax

JSON syntax is a subset of JavaScript syntax

JSON Syntax Rules

JSON syntax is a subset of the JavaScript object notation syntax:

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

JSON Name/Value Pairs

JSON data is written as name/value pairs.

A name/value pair consists of a field name (in double quotes), followed by a colon, followed by a value:

"firstName" : "John"

This is simple to understand, and equals to the JavaScript statement:

firstName = "John"

JSON Values

JSON values can be:

* A number (integer or floating point)
* A string (in double quotes)
* A Boolean (true or false)
* An array (in square brackets)
* An object (in curly brackets)
* null

JSON Objects

JSON objects are written inside curly brackets,

Objects can contain multiple name/values pairs:

{ "firstName":"John" , "lastName":"Doe" }

This is also simple to understand, and equals to the JavaScript statements:

firstName = "John";  
lastName = "Doe";

JSON Arrays

JSON arrays are written inside square brackets.

An array can contain multiple objects:

{  
"employees": [  
{ "firstName":"John" , "lastName":"Doe" },   
{ "firstName":"Anna" , "lastName":"Smith" },   
{ "firstName":"Peter" , "lastName":"Jones" }  
]  
}

In the example above, the object "employees" is an array containing three objects. Each object is a record of a person (with a first name and a last name).

JSON Uses JavaScript Syntax

Because JSON uses JavaScript syntax, no extra software is needed to work with JSON within JavaScript.

With JavaScript you can create an array of objects and assign data to it like this:

Example

var employees = [  
{ "firstName":"John" , "lastName":"Doe" },   
{ "firstName":"Anna" , "lastName":"Smith" },   
{ "firstName":"Peter" , "lastName": "Jones" }  
];

The first entry in the JavaScript object array can be accessed like this:

employees[0].firstName + " " + employees[0].lastName;

The returned content will be:

John Doe

The data can be modified like this:

employees[0].firstName = "Gilbert";

Source = <http://www.codeproject.com/Articles/773359/Introduction-to-JSON-Using-JSON-in-ASP-NET-Web-App>

JSON is JavaScript Object Notation and was invented by [Douglas Crockford](http://en.wikipedia.org/wiki/Douglas_Crockford). Actually JSON is a lightweight alternative to XML, language and platform independent script.  
JSON is a universal, language-independent format for data. In this way, it's similar to XML. Whereas XML uses tags, JSON is based on the object-literal notation of JavaScript. There is support for use of JSON with many other languages, including C#, ASP.NET, Java, Perl, PHP, Python, and Ruby. More information on JSON can be found at [www.JSON.org](http://www.JSON.org) .

### What is JSON?

* JSON stands for JavaScript Object Notation.
* JSON is lightweight text-data interchange format.
* JSON is syntax for storing and exchanging text information. Much like XML.
* JSON is smaller than XML, and faster and easier to parse.
* JSON is language independent.
* JSON uses JavaScript syntax for describing data objects, but JSON is still language and platform independent.
* JSON parsers and JSON libraries exists for many different programming languages.
* The JSON filename extension is .json.
* JSON Internet Media type is application/js.

### Uses of JSON

* JSON format is used for serializing & transmitting structured data over network connection.
* This is primarily used to transmit data between server and web application.
* Web Services and APIs use JSON format to provide public data.
* It can be used with modern programming languages.

### JSON with JavaScript

JSON is a subset of the object literal notation of JavaScript. Since JSON is a subset of JavaScript, JSON provides for an easy way to create and store data structures within JavaScript.  
The JSON text format is syntactically identical to the code for creating JavaScript objects.  
Because of this similarity, instead of using a parser, a JavaScript program can use the built-in eval() function and execute JSON data to produce native JavaScript objects.  
JSON data can be transported using AJAX.

### Why JSON?

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
* eval() the JSON string

### JSON Syntax

JSON syntax is a subset of the JavaScript object notation syntax:

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

Let’s have a look at the syntax:

http://www.codeproject.com/images/minus.gifCollapse | [Copy Code](http://www.codeproject.com/Articles/773359/Introduction-to-JSON-Using-JSON-in-ASP-NET-Web-App)

<script>

var data={ "Name" : "Arv" };

alert(data.Name);

</script>

First, I create a variable to hold data, and then use JSON to define my object.   
Just an item called Name and its value Arv.

Now I add some more values:

http://www.codeproject.com/images/minus.gifCollapse | [Copy Code](http://www.codeproject.com/Articles/773359/Introduction-to-JSON-Using-JSON-in-ASP-NET-Web-App)

<script>

var data={ "Name":"Arv", "Designation":"Software Engineer", "ExperienceInYears":4

};

</script>

alert('Name : ' + data.Name + 'Designation : ' + data.Designation + 'Total Experience : ' + data.ExperienceInYears);

### Storing JSON Data in Arrays

To create JSON Array, enclose multiple objects in square brackets.  
Example:

http://www.codeproject.com/images/minus.gifCollapse | [Copy Code](http://www.codeproject.com/Articles/773359/Introduction-to-JSON-Using-JSON-in-ASP-NET-Web-App)

var description= [{"Name":"Arv", "Designation":"Software Engineer", "ExperienceInYears":4},

{"Name":"Rsh", "Designation":"Tester", "ExperienceInYears":2}];

To access this information, we need to access the array index of the description array we wish to access.   
For example:

http://www.codeproject.com/images/minus.gifCollapse | [Copy Code](http://www.codeproject.com/Articles/773359/Introduction-to-JSON-Using-JSON-in-ASP-NET-Web-App)

document.write(description[0].Name); // Output: Arv

document.write(description[0].ExperienceInYears); // Output: 4

document.write(description[1].Name); // Output: Rsh

### Using JSON with ASP.NET Web Application

To use JSON in our application, we need to write JavaScript frunction and call it on our head tag or we can write as follows:

http://www.codeproject.com/images/minus.gifCollapse | [Copy Code](http://www.codeproject.com/Articles/773359/Introduction-to-JSON-Using-JSON-in-ASP-NET-Web-App)

<script type = "text/javascript">

//write JSON Code

</script>

and to communicate with server JSON creates JSONRequest.  
So I am creating a web application named as JSONDemoApplication.

I have attached a demo application that you can find in the attachment. In my application, I create a simple aspx form named JSONTest.aspx.

http://www.codeproject.com/images/minus.gifCollapse | [Copy Code](http://www.codeproject.com/Articles/773359/Introduction-to-JSON-Using-JSON-in-ASP-NET-Web-App)

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="JSONTest.aspx.cs" Inherits="\_JSONTest" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">

<title>JSON Application</title>

</head>

<body style = "font-family:Arial; font-size:10pt">

<form id="form1" runat="server">

<center>

<table>

<tr><td colspan="4" align="center">

Enter your UserName and select Department See SQL script to know default username and department

</td></tr>

<tr>

<td align="right">

UserName : </td>

<td align="left">

<asp:TextBox ID="txtUserName" runat="server"

onkeyup = "OnChange(this)" onblur= "ShowAvailability()"></asp:TextBox>

<span id = "mesg"></span> </td>

<td align="right">

Department : </td>

<td align="left">

<asp:DropDownList ID="ddlDepartment" runat="server">

<asp:ListItem Text="HR" Value="1"></asp:ListItem>

<asp:ListItem Text="Fianace" Value="2"></asp:ListItem>

<asp:ListItem Text="Admin" Value="3"></asp:ListItem>

<asp:ListItem Text="IT" Value="4"></asp:ListItem>

</asp:DropDownList>

<span id = "msg2"></span> </td>

</tr>

</table>

</center>

</form>

</body>

</html>

#### JSONRequest

JSONRequest is a global JavaScript object. It provides three methods: post, get, and cancel.

#### JSONRequest.post

JSONRequest.post does an HTTP POST of the serialization of a JavaScript object or array, gets the response, and parses the response into a JavaScript value. If the parse is successful, it returns the value to the requesting script. In making the request, no HTTP authentication or cookies are sent. Any cookies returned by the server cause the request to fail. The JSONRequest service can only be used to send and receive JSON-encoded values. JSONRequest cannot be used to retrieve other text formats.  
  
In the demo application, I am using post method.

JSONRequest.post takes some parameters:

url: The URL to POST to. The URL does not need to be related to the page's URL.  
send- object: The JavaScript object or array to send as the POST data. It will be serialized as JSON text. Cyclical structures will fail.  
done- function (requestNumber, value, exception): The function to be called when the request is completed. If the request was successful, the function will receive the request number and the returned value. If it is not successful, it will receive the request number and an exception object. The done function will not be called until after the call to JSONRequest returns a serial number.

So I have created a ShowAvailability function and create JSONRequest:

http://www.codeproject.com/images/minus.gifCollapse | [Copy Code](http://www.codeproject.com/Articles/773359/Introduction-to-JSON-Using-JSON-in-ASP-NET-Web-App)

<script type = "text/javascript">

function ShowAvailability() {

$.ajax({

type: "POST",//using post method to send data

url: "CS.aspx/CheckUserName", //path to find web method CheckUserName

data: '{userName: "' + $("#<%=txtUserName.ClientID%>")[0].value + '" }',//assign value to username from textbox

contentType: "application/json; charset=utf-8",

dataType: "json",//datatype json is compulsory

success: OnSuccess,

failure: function(response) {

alert(response);

}

});

}

function OnSuccess(response) {

var mesg = $("#mesg")[0];

switch (response.d) {

case "0":

mesg.style.color = "red";

mesg.innerHTML = "Username not exist";

break;

case "1":

mesg.style.color = "green";

mesg.innerHTML = "Available";

break;

case "error":

mesg.style.color = "red";

mesg.innerHTML = "Error occurred";

break;

}

}

function OnChange(txt) {

$("#mesg")[0].innerHTML = "";

}

</script>

Put this code into the head section of JSONTest.aspx form:

http://www.codeproject.com/images/minus.gifCollapse | [Copy Code](http://www.codeproject.com/Articles/773359/Introduction-to-JSON-Using-JSON-in-ASP-NET-Web-App)

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="JSONTest.aspx.cs" Inherits="\_JSONTest" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">

<title></title>

<script src="scripts/jquery-1.9.1.js" type="text/javascript"></script>

<script type = "text/javascript">

function ShowAvailability() {

$.ajax({

type: "POST",

url: "CS.aspx/CheckUserName",

data: '{userName: "' + $("#<%=txtUserName.ClientID%>")[0].value + '" }',

contentType: "application/json; charset=utf-8",

dataType: "json",

success: OnSuccess,

failure: function(response) {

alert(response);

}

});

}

function OnSuccess(response) {

var mesg = $("#mesg")[0];

switch (response.d) {

case "0":

mesg.style.color = "red";

mesg.innerHTML = "Username not exist";

break;

case "1":

mesg.style.color = "green";

mesg.innerHTML = "Available";

break;

case "error":

mesg.style.color = "red";

mesg.innerHTML = "Error occurred";

break;

}

}

function OnChange(txt) {

$("#mesg")[0].innerHTML = "";

}

</script>

</head>

<body style = "font-family:Arial; font-size:10pt">

<form id="form1" runat="server">

<center>

<table>

<tr><td colspan="4" align="center">

Enter your UserName and select Department See SQL script to know default username and department

</td></tr>

<tr>

<td align="right">UserName : </td>

<td align="left"><asp:TextBox ID="txtUserName" runat="server"

onkeyup = "OnChange(this)" onblur= "ShowAvailability()">

</asp:TextBox><span id = "mesg"></span> </td>

<td align="right">Department : </td>

<td align="left">

<asp:DropDownList ID="ddlDepartment" runat="server">

<asp:ListItem Text="HR" Value="1"></asp:ListItem>

<asp:ListItem Text="Fianace" Value="2"></asp:ListItem>

<asp:ListItem Text="Admin" Value="3"></asp:ListItem>

<asp:ListItem Text="IT" Value="4"></asp:ListItem>

</asp:DropDownList> <span id = "msg2"></span> </td>

</tr>

</table>

</center>

</form>

</body>

</html>

Now we need to create web method as we are calling from JSON similar name as CheckUserName with one parameter at the location as we assigned in url JSONTest.aspx.

http://www.codeproject.com/images/minus.gifCollapse | [Copy Code](http://www.codeproject.com/Articles/773359/Introduction-to-JSON-Using-JSON-in-ASP-NET-Web-App)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Data.SqlClient;

using System.Configuration;

using System.Data;

public partial class \_JSONTest : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

}

[System.Web.Services.WebMethod]

public static string CheckUserName(string userName)//parameter send from JSON call

{

string returnValue = string.Empty;

try

{

string consString = ConfigurationManager.ConnectionStrings["conString"].ConnectionString;

SqlConnection conn = new SqlConnection(consString);

SqlCommand cmd = new SqlCommand("Sp\_CheckAvailability", conn);//SP to check username available in database

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@UserName", userName.Trim());

conn.Open();

returnValue = cmd.ExecuteScalar().ToString();

conn.Close();

}

catch(SqlException ex)

{

returnValue = "error" + ex.ToString();

}

return returnValue;

}

}

I have created a table and inserted some records:

http://www.codeproject.com/images/minus.gifCollapse | [Copy Code](http://www.codeproject.com/Articles/773359/Introduction-to-JSON-Using-JSON-in-ASP-NET-Web-App)

CREATE DATABASE MyJSONDB ----create database

GO

USE MyJSONDB

GO

CREATE TABLE tblLogin ---create table to check data

(

EmpUserName NVARCHAR(100),

EmpDepartment NVARCHAR(100)

)

GO

----Insert dummy data

INSERT INTO tblLogin VALUES('hruser','1')

INSERT INTO tblLogin VALUES('Fianaceuser','2')

INSERT INTO tblLogin VALUES('adminuser','3')

INSERT INTO tblLogin VALUES('ituser','dd','4')

INSERT INTO tblLogin VALUES('productionuser','5')

INSERT INTO tblLogin VALUES('testinguser','6')

GO

and created Stored Procedure to check username in database table:

http://www.codeproject.com/images/minus.gifCollapse | [Copy Code](http://www.codeproject.com/Articles/773359/Introduction-to-JSON-Using-JSON-in-ASP-NET-Web-App)

CREATE PROCEDURE Sp\_CheckAvailability ( @UserName NVARCHAR(100)='')

AS

BEGIN

SELECT COUNT(\*) FROM tblLogin WHERE EmpUserName=@UserName

END

Points of Interest

You see, in an Ajax environment where we make calls to web services, we expect to get some data back in some form. Well, if we receive XML back as a direct result of an Ajax call, we have to send that data through an XML parser before we can even begin to manipulate the data to be useful to JavaScript. If we receive the data in JSON... we do not have to do anything but assign the results to a variable because JSON is already JavaScript. From there, we can manipulate the data as normal.

**JSON Return from Web Service**

Return the list instead, and use [ScriptMethod(ResponseFormat = ResponseFormat.Json)] attribute - it will create JSON object as return automatically:

[WebMethod]

[ScriptMethod(ResponseFormat = ResponseFormat.Json)]

public List<RetUsers> GetSomething()

{

// avoid circual reference(parent child)

List<RetUsers> res = repo.GetAllUser().Select(c => new RetUsers {User\_ID = c.User\_ID,User\_Name = c.User\_Name,Date\_Expire = c.Date\_Expire }).ToList();

return res;

}

And on JS side:

$.ajax(

{

type: "POST",

async: true,

url:http://example.com/yourwebservice.asmx/GetSomething,

data: {some data},

contentType: "application/json; charset=utf-8",

dataType: "json",

success: function(msg)

{

var resultAsJson = msg.d // your return result is JS array

// Now you can loop over the array to get each object

for(var i in resultAsJson)

{

var user = resultAsJson[i]

var user\_name = user.User\_Name

// Here you append that value to your label

}

}

})