What is JSON

- JSON stands for JavaScript Object Notation.
- JSON is lightweight data-interchange format.
- JSON is easy to read and write than XML.
- JSON is language independent.
- JSON supports array, object, string, number and values.

JSON Example

In this tutorial, you will get a lot of JSON examples to understand the topic well. The JSON file must be save with .json extension. Let's see a simple JSON example.

File: first.json

```
{"employees":[
   {"name":"Raj", "email":"raj1987@gmail.com"},
   {"name":"Rahul", "email":"rahul32@gmail.com"},
   {"name":"John", "email":"john32bob@gmail.com"}
]}
```

What is JSON

JSON is an open standard for exchanging data on the web. It supports data structures like object and array. So it is easy to write and read data from JSON.

What is JSON

- JSON stands for JavaScript Object Notation.
- JSON is an open standard data-interchange format.
- JSON is lightweight and self describing.
- JSON is originated from JavaScript.
- JSON is easy to read and write.
- JSON is language independent.
- JSON supports data structures such as array and objects.

Features of JSON

1. Simplicity
2. Openness
3. Self Describing
4. Internationalization
5. Extensibility
6. Interoperability

No.	JSON	XML
1)	JSON stands for JavaScript Object Notation.	XML stands for eXtensible Markup Language.
2)	JSON is simple to read and write.	XML is less simple than JSON.
3)	JSON is easy to learn.	XML is less easy than JSON.
4)	JSON is data-oriented.	XML is document-oriented.
5)	JSON doesn't provide display capabilities.	XML provides the capability to display data because it is a markup language.
6)	JSON supports array.	XML doesn't support array.
7)	JSON is less secured than XML.	XML is more secured.
8)	JSON files are more human readable than XML.	XML files are less human readable.
9)	JSON supports only text and number data type.	

XML support many data types such as text, number,

images, charts, graphs etc. Moreover, XML offeres
options for transferring the format or structure of the data with actual data.

JSON Example

```
{"employees":[
    {"name":"Vimal", "email":"vjaiswal1987@gmail.com"},
    {"name":"Rahul", "email":"rahul12@gmail.com"},
    {"name":"Jai", "email":"jai87@gmail.com"}
]}
```

XML Example

```
<employees>
    <employee>
        <name>Vimal</name>
        <email>vjaiswal1987@gmail.com</email>
    </employee>
    <employee>
        <name>Rahul</name>
        <email>rahul12@gmail.com</email>
    </employee>
    <employee>
        <name>Jai</name>
        <email>jai87@gmail.com</email>
    </employee>
</employees>
```

Similarities between JSON and XML

- Both are simple and open.
- Both supports unicode. So internationalization is supported by JSON and XML both.
- Both represents self describing data.
- Both are interoperable or language-independent.

JSON Example

JSON example can be created by object and array. Each object can have different data such as text, number, boolean etc. Let's see different JSON examples using object and array.

JSON Object Example

A JSON object contains data in the form of key/value pair. The keys are strings and the values are the JSON types. Keys and values are separated by colon. Each entry (key/value pair) is separated by comma.

The **{** (curly brace) represents the JSON object.

```
{
    "employee": {
        "name":       "sonoo",
        "salary":      56000,
        "married":    true
    }
}
```

JSON Array example

The **[** (square bracket) represents the JSON array. A JSON array can have values and objects.

Let's see the example of JSON array having values.

```
["Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"]
```

Let's see the example of JSON array having objects.

```
[
    {"name":"Ram", "email":"Ram@gmail.com"},
    {"name":"Bob", "email":"bob32@gmail.com"}
]
```

JSON Example 1

```
{"employees":[
    {"name":"Shyam", "email":"shyamjaiswal@gmail.com"},
    {"name":"Bob", "email":"bob32@gmail.com"},
    {"name":"Jai", "email":"jai87@gmail.com"}
]}
```

The XML representation of above JSON example is given below.

```
<employees>
    <employee>
        <name>Shyam</name>
        <email>shyamjaiswal@gmail.com</email>
    </employee>
    <employee>
        <name>Bob</name>
        <email>bob32@gmail.com</email>
    </employee>
    <employee>
        <name>Jai</name>
        <email>jai87@gmail.com</email>
    </employee>
</employees>
```

JSON Example 2

```
{"menu": {
  "id": "file",
  "value": "File",
  "popup": {
    "menuitem": [
      {"value": "New", "onclick": "CreateDoc()"},
      {"value": "Open", "onclick": "OpenDoc()"},
      {"value": "Save", "onclick": "SaveDoc()"}
    ]
  }
}}
```

The XML representation of above JSON example is given below.

```
<menu id="file" value="File">
  <popup>
    <menuitem value="New" onclick="CreateDoc()" />
    <menuitem value="Open" onclick="OpenDoc()" />
    <menuitem value="Save" onclick="SaveDoc()" />
  </popup>
</menu>
```

JSON Object

JSON object holds key/value pair. Each key is represented as a string in JSON and value can be of any type. The keys and values are separated by colon. Each key/value pair is separated by comma.

The curly brace **{** represents JSON object.

Let's see an example of JSON object.

```
{
    "employee": {
        "name":       "sonoo",
        "salary":      56000,
        "married":    true
    }
}
```

In the above example, employee is an object in which "name", "salary" and "married" are the key. In this example, there are string, number and boolean value for the keys.

JSON Object with Strings

The string value must be enclosed within double quote.

```
{
        "name":       "sonoo",
        "email":      "sonoojaiswal1987@gmail.com"
}
```

JSON Object with Numbers

JSON supports numbers in double precision floating-point format. The number can be digits (0-9), fractions (.33, .532 etc) and exponents (e, e+, e-,E, E+, E-).

```
{
"integer": 34,
"fraction": .2145,
"exponent": 6.61789e+0
}
```

JSON Object with Booleans

JSON also supports boolean values *true* or *false*.

```
{
"first": true,
"second": false
}
```

JSON Nested Object Example

A JSON object can have another object also. Let's see a simple example of JSON object having another object.

```
{
    "firstName": "Sonoo",
    "lastName": "Jaiswal",
    "age": 27,
    "address" : {
        "streetAddress": "Plot-6, Mohan Nagar",
        "city": "Ghaziabad",
        "state": "UP",
        "postalCode": "201007"
    }
}
```

JSON Array

JSON array represents ordered list of values. JSON array can store multiple values. It can store string, number, boolean or object in JSON array.

In JSON array, values must be separated by comma.

The **[** (square bracket) represents JSON array.

JSON Array of Strings

Let's see an example of JSON arrays storing string values.

```
["Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"]
```

JSON Array of Numbers

Let's see an example of JSON arrays storing number values.

```
[12, 34, 56, 43, 95]
```

JSON Array of Booleans

Let's see an example of JSON arrays storing boolean values.

```
[true, true, false, false, true]
```

JSON Array of Objects

Let's see a simple JSON array example having 4 objects.

```
{"employees":[
    {"name":"Ram", "email":"ram@gmail.com", "age":23},
    {"name":"Shyam", "email":"shyam23@gmail.com", "age":28},
    {"name":"John", "email":"john@gmail.com", "age":33},
    {"name":"Bob", "email":"bob32@gmail.com", "age":41}
]}
```

JSON Multidimensional Array

We can store array inside JSON array, it is known as array of arrays or multidimensional array.

```
[
 [ "a", "b", "c" ],
 [ "m", "n", "o" ],
 [ "x", "y", "z" ]
]
```