## What is JSON?

JSON is used to store information in an organized, and easy-to-access manner. Its full form is JavaScript Object Notation. It offers a human-readable collection of data which can be accessed logically.

Its filename extension for written programming code is `.json`. The Internet Media type for JSON is application/json and the public.json is its Uniform Type Identifier. The file name extension is .json.

## Why use JSON?

Here are the important benefits/ pros of using JSON:

- Provide support for all browsers
- Easy to read and write
- Straightforward syntax
- You can natively parse in JavaScript using eval() function
- Easy to create and manipulate
- Supported by all major JavaScript frameworks
- Supported by most backend technologies
- JSON is recognized natively by JavaScript
- It allows you to transmit and serialize structured data using a network connection.
- You can use it with modern programming languages.
- JSON is text which can be converted to any object of JavaScript into JSON and send this JSON to the server.

## Features of JSON

**Easy to use** - JSON API offers high-level facade, which helps you to simplify commonly used use-cases.

**Performance** - JSON is quite fast as it consumes very less memory space, which is especially suitable for large object graphs or systems.

**Free tool** - JSON library is open source and free to use.

**Doesn't require to create mapping** - Jackson API provides default mapping for many objects to be serialized.

**Clean JSON** - Creates clean, and compatible JSON result that is easy to read.

**Dependency** - JSON library does not require any other library for processing.

### Rules for JSON Syntax

Rules for JSON syntax are:

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

## Data Types in JSON

Important data type used in JSON are:

| Data Type | Description |
|---|---|
| Number | It includes real number, integer or a floating number |
| String | It consists of any text or Unicode double-quoted with backslash escapement |
| Boolean | The Boolean data type represents either True or False values |
| Null | The Null value denotes that the associated variable doesn't have any value |
| Object | It is a collection of key-value pairs and always separated by a comma and enclosed in curly brackets. |
| Array | It is an ordered sequence of values separated. |

### Number:

- The number is a double-precision floating-point format which depends on its implementation method.
- In JSON you can't use Hexadecimal and Octal formats.

Following table displays number types:

| Type | Description |
|---|---|
| Integer | Number 1-9, and 0. Both positive and negative numbers. |
| Fraction | Fractions like 3 |
| Exponent | Exponent like e, e+ |

Syntax:

```
var json-object-name = { string : number_value,.....}
```

Example:

```
var obj = {salary: 2600}
```

### String:

It is a series of double-quoted Unicode characters and having backslash escaping.

The following table shows various string types:

| Type | Description |
|---|---|
| * | Use for double quotation typing |
| / | Use for solidus |
| \ | Use for reverse solidus |
| B | Use to add backspace |
| F | From feed |
| N | To create a new line |
| R | Use for carriage return |
| T | To show horizontal tab |
| U | Hexadecimal digits |

Syntax:

```
var json-object-name = { string : "string value",…..}
```

Example:

```
var obj= {name: 'Andy'}
```

### Boolean

It stores only true or false values.

Syntax:

```
var json-object-name = {string : true/false, …..}
```

Example:

```
var obj = {active: 'true'}
```

### Array

- It is an ordered collection of values.
- You should use an array when the key names are sequential integers.
- It should be enclosed inside square brackets which should be separated by ',' (comma)

Syntax:

```
[value, .......]
```

Example:

Showing an array storing multiple objects:

```
{
   "eBooks":[
      {
         "language":"Pascal",
         "edition":"third"
      },
      {
         "language":"Python",
         "edition":"four"
      },
      {
         "language":"SQL",
         "edition":"second"
      }
   ]
}
```

### Object

- An object should be enclosed in curly braces,
- It should be an unordered set of name or value pairs.
- Name should be followed by ":" (colon)and the name/value pairs need to be separated using "," (comma).
- You can use it when key names are arbitrary strings.

Syntax:

```
{ string :  value, …. }
```

Example:

```
{
"id": 110,
"language": "Python",
"price": 1900,
}
```

### Whitespace

You can insert whitespace between a pair of tokens.

Example:

Syntax:

```
{string:"    ",….}
```

Example:

```
var a = " Alex"; var b = "Steve";
```

## Example of JSON

The given code example defines how to use JSON to store information related to programming books along with edition and author name.

```
{
   "book":[
      {
         "id":"444",
         "language":"C",
         "edition":"First",
         "author":"Dennis Ritchie "
      },
      {
         "id":"555",
         "language":"C++",
         "edition":"second",
         "author":" Bjarne Stroustrup "
      }
   ]
}
```

Lets understand JSON format with another example. Here, JSON defines the first name, last name and id of a student.

```
{
  "student": [

     {
        "id":"01",
        "name": "Tom",
        "lastname": "Price"
     },

     {
        "id":"02",
        "name": "Nick",
        "lastname": "Thameson"
     }
  ]
}
```

## Application of JSON

Here are some common applications of JSON:

- Helps you to transfer data from a server
- JSON format helps transmit and serialize all types of structured data.
- Allows you to perform asynchronous data calls without the need to do a page refresh
- Helps you to transmit data between a server and web applications.
- It is widely used for JavaScript-based application, which includes browser extension and websites.
- You can transmit data between the server and web application using JSON.
- We can use JSON with modern programming languages.
- It is used for writing JavaScript-based applications that include browser add-ons.
- Web services and Restful APIs use the JSON format to get public data.

### JSON vs. XML

Here is the prime difference between JSON vs. XML

| JSON | XML |
|---|---|
| JSON object has a type | XML data is typeless |
| JSON types: string, number, array, Boolean | All XML data should be string |
| Data is readily accessible as JSON objects | XML data needs to be parsed. |
| JSON files are more human-readable. | XML files are less human-readable. |
| JSON is supported by most browsers. | Cross-browser XML parsing can be tricky |
| JSON has no display capabilities. | XML provides a capability to display data because it is a markup language. |
| Retrieving value is easy | Retrieving value is difficult |
| Supported by many Ajax toolkit | Not fully supported by Ajax toolkit |
| A fully automated way of deserializing/serializing JavaScript. | Developers have to write JavaScript code to serialize/de-serialize from XML |
| Native support for object. | The object has to be express by conventions - mostly missed use of attributes and elements. |

### JSON Example

```
{
  "student": [

    {
       "id":"01",
       "name": "Tom",
       "lastname": "Price"
    },

    {
       "id":"02",
       "name": "Nick",
       "lastname": "Thameson"
    }
  ]
}
```

### XML Example

```
<?xml version="1.0" encoding="UTF-8" ?>
<root>
        <student>
                <id>01</id>
                <name>Tom</name>
                <lastname>Price</lastname>
        </student>
        <student>
                <id>02</id>
                <name>Nick</name>
                <lastname>Thameson</lastname>
        </student>
</root>
```

## Summary:

- JSON method is used to store information in an organized, and easy-to-access manner.
- JSON Provides support for all browsers offers by many languages.
- Douglas Crockford specified the JSON format in the early 2000s
- JSON API offers high-level facade, which helps you to simplify commonly used use-cases
- The important rules for writing JSON system is that data should be written in name/value pairs.
- Number, String, Boolean, Null, Object, and Array are important Data types used in JSON.
- It helps you to transfer data from a server.
- JSON object has a type whereas XML data is typeless
- JSON is not a document format
- No namespace support, hence poor extensibility
- JSONLint is an open-source project that is used as a validator and reformatter for JSON.