Java**S**cript **O**bject **N**otation

(**JSON**)

JSON Introduction

- JSON stands for JavaScript Object Notation
- JSON is a lightweight data-interchange format
- JSON is plain text written in JavaScript object notation
- JSON is language independent *
- The JSON format was originally specified by Douglas Crockford

JSON Syntax

The JSON syntax is a subset of the JavaScript syntax

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

JSON Data

- JSON data is written as name/value pairs

```
"name" : "John"
```

- JSON keys must be strings, written with double quotes

```
{ "name" : "John" }
```

  - In JavaScript, keys can be strings, numbers, or identifier names

```
{ name : "John" }
```

- In JSON, string values must be written with double quotes:

```
{ "name" : "John" }
```

  - In JavaScript, you can write string values with double or single quotes:

```
{ name : 'John' }
```

JSON Data Types

- In JSON, values must be one of the following data types:
  - a string
  - a number
  - an object (JSON object)
  - an array
  - a boolean
  - null

- In JavaScript values can be all of the above, plus any other valid JavaScript expression, including:
  - a function
  - a date
  - undefined

JSON Files

- The file type for JSON files is ".json"
- The MIME type for JSON text is "application/json"

JSON vs XML

JSON Example

employees object, with an array of 3 employees

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

JSON vs XML

XML Example

```
<employees>
    <employee>
        <firstName>John</firstName>
        <lastName>Doe</lastName>
    </employee>
    <employee>
        <firstName>Anna</firstName>
        <lastName>Smith</lastName>
    </employee>
    <employee>
        <firstName>Peter</firstName>
        <lastName>Jones</lastName>
    </employee>
</employees>
```

Exchanging Data

- When exchanging data between a browser and a server, the data can only be text.
- JSON is text, and we can convert any JavaScript object into JSON, and send JSON to the server
- We can also convert any JSON received from the server into JavaScript objects

JSON.parse()

- A common use of JSON is to exchange data to/from a web server.
- When receiving data from a web server, the data is always a string.
- Parse the data with JSON.parse(), and the data becomes a JavaScript object.
- Example
- When using the JSON.parse() on a JSON derived from an array, the method will return a JavaScript array, instead of a JavaScript object.
- Example

JSON.stringify()

- A common use of JSON is to exchange data to/from a web server.
- When sending data to a web server, the data has to be a string.
- Convert a JavaScript object into a string with JSON.stringify().
- Example
- It is also possible to stringify JavaScript arrays:
- Example

Storing Data

```
//Storing data:
myObj = { "name":"John", "age":31, "city":"New York" };
myJSON = JSON.stringify(myObj);
localStorage.setItem("testJSON", myJSON);

//Retrieving data:
text = localStorage.getItem("testJSON");
obj = JSON.parse(text);
document.getElementById("demo").innerHTML = obj.name;
```

Example