Session 9

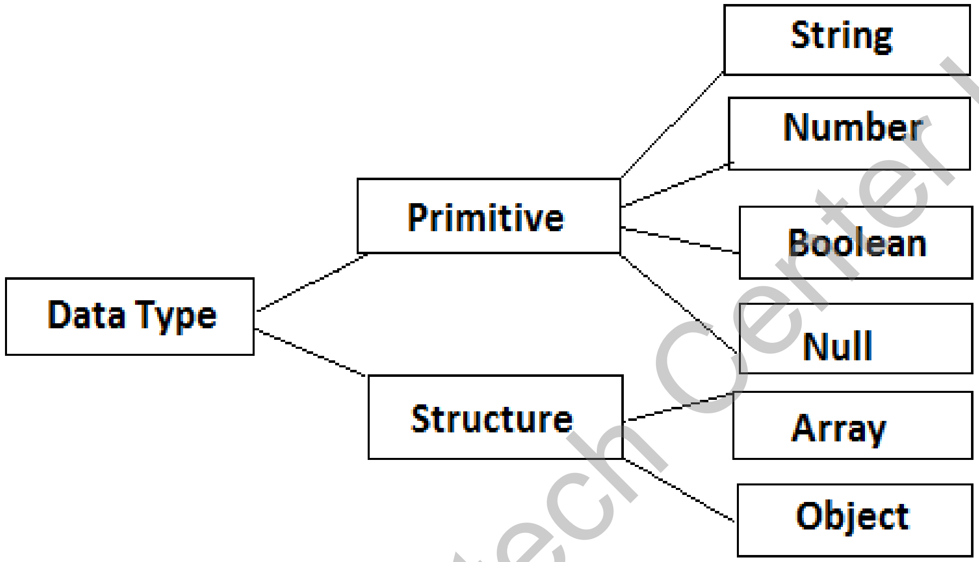
Work with JSON Data

Objectives

- Identify data types supported by JSON
- Explain how to represent complex data with JSON
- Explain how to execute serialization and de-serialization of JSON with JavaScript
- Describe tools and editors that can be used to work with JSON
- Describe the syntax and schema of a JSON document

JSON Data Types 1-4

The two primary data types supported by JSON are primitive and structure.



In JSON, the type of a variable is recognized automatically during the parsing phase.

JSON Data Types 1-4

Number

- It is a floating-point format (double precision).
- Does not accept Octal, hexadecimal, NaN, and Infinity values.
- The types allowed are integer, fraction, and exponent.
- **Syntax** `{"string": number_value, .......}`

String

- A series of zero or other Unicode characters within double quotes and backslash escapes.
- Delimited with double-quotation marks.
- A character denotes a single character string.
- **Syntax** `{"string":"string value", .......}`

JSON Data Types 2-4

Boolean

- Has only two values: true and false.
- Using quotes for Boolean values treats them as String values.
- **Syntax** `{string: true/false, ........}`

Null

- Is an empty type.

White Space

- A white space can appear between the characters in string values to make code more comprehensible.
- **Syntax** `{string:"  ",....}`

JSON Data Types 3-4

Arrays allow storing various values of the same type in one variable.

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

The characteristics of an array in JSON are:

- It is a sequential collection of values, not necessarily of the same type.
- Indexing begins at 0 or 1.
- It is enclosed in square brackets [.].
- Each value is set apart by comma (,).

JSON Data Types 4-4

An object is an independent data type, having its own attributes.

Syntax `{string : value, .......}`

The characteristics of an object in JSON are:

- It is a non-sequential (having no order) set of key/value pairs.
- It is enclosed in curly braces {.}.
- Each key/value pairs are set apart by comma (,) and every key is proceeded by colon (:).
- The keys are only strings, each differing from the other.
- It is used when the key names are random strings.

JSON Value

- In JSON, value can be of any primitive and structure data type.
- A JSON value can be a number, string, Boolean such as true or false, null, object, or an array.
- The following Code Snippet shows some examples of JSON values:

```
var emp-no = 1234;
var name = "Sherill";
var experience = null;
```

Storing Different Values in Arrays

JSON arrays can store elements of different types. Following is a sample code:

```
[ 456, "Dog", 123, "Frog", true ]
```

- Elements 0 and 2 in the array are of the type Number.
- Elements 1 and 3 are of String type.
- Element 4 is a Boolean type.

In JSON, arrays can also contain other arrays. This is called nesting of arrays.

Data Structures Supported by JSON

In JSON, the built-in data structures can be used to develop other data structures.

The two data structures supported by JSON are:

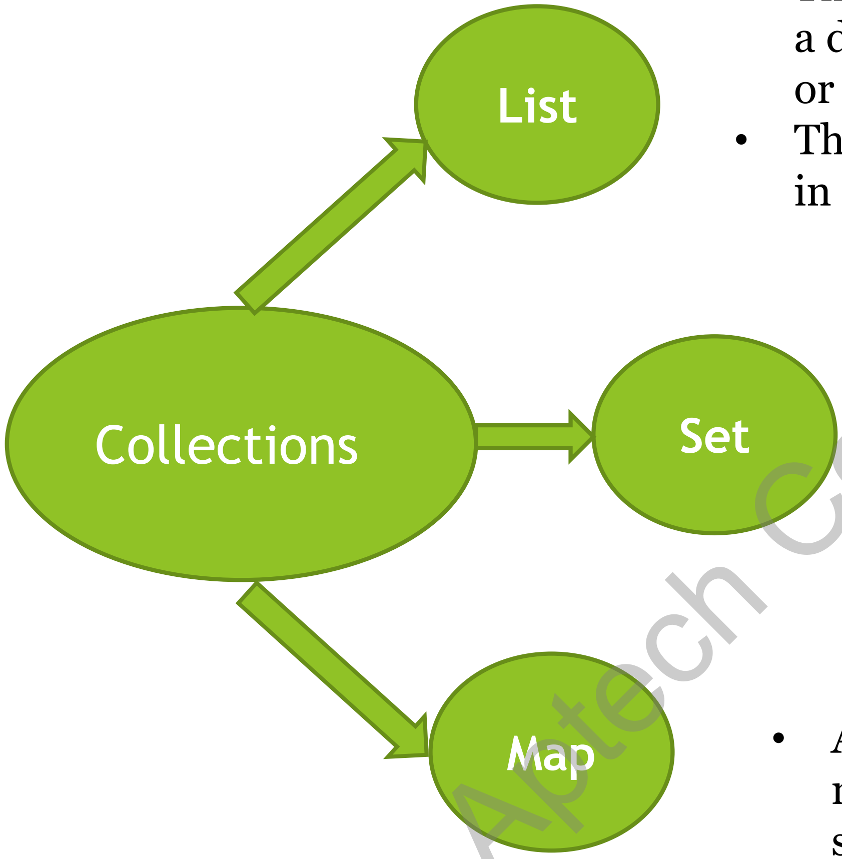
Collection of Name/Value Pairs

- Supported by different programming languages, in the form of objects.

Ordered List of Values

- Includes arrays, lists, and vectors.

Collections: Maps, Sets, and Lists



- The elements in a list follow a defined order and copies or duplicates are allowed.
- The elements are positioned in a specific location.

- Collection without duplicate elements.
- In JSON, it is not possible to avoid replicas, however, the parser removes the duplicates while serializing the data.

- A type of data structure that mainly aids in quick searching of data.
- Accepts data in the form of key and value pairs and each key is distinct.

JSON Schema

- JSON schema specifies the rules that defines the structure of a JSON document.

- The two main aspects that define why a JSON schema language is required are:

To specify JSON data structures

- Handy when the JSON-based Web services need to be made accessible to a large audience

To validate JSON data structures

- Handy when validating JSON documents from other applications
- Helps to avoid parsing an invalid data structure in valid JSON documents

Schema Overview

- `Application/schema+json` is the media type described by JSON schema and prescribes the design of JSON documents.
- It offers provision for defining the structure of the documents with respect to the permitted values, descriptions, and decoding connections to other resources.
- The JSON schema format is arranged in the following individual definitions:

Core Schema Specification	• Explains a JSON structure and states valid elements in it.
Hyper Schema Specification	• Explains the elements in a JSON structure that can be considered as hyperlinks.

JSON Comments

- JSON does not have any provision for documentation or comments.

- However, comments are still supported by a few JSON parsers and must be provided within /* ... */.

```
{
 "id":2,
 "title":"The fallen Hero", /* This books
is about Harvey Dent */
 "noOfCopies":14,
 "tags":[
 "BatMan",
 "Gowtam"
 ],
}
```

Creating and Parsing JSON Messages with JavaScript

The JSON.stringify() method allows converting a JavaScript object into a JSON String.

The JSON.parse() method allows parsing a JSON object using JavaScript.

It converts a JSON String to an object in JavaScript.

Start by defining a string in JSON format, using the method for conversion, and then looping through the attributes for printing its values.

JSON with Developer Tools on Browser

- There are many plugins or extensions that help in validating and formatting a JSON document or a JSON HTTP response.

- One such extension in Firefox is **`JSONView`**, which allows viewing a JSON document.

- With **`JSONView`** the JSON document is displayed in the browser.

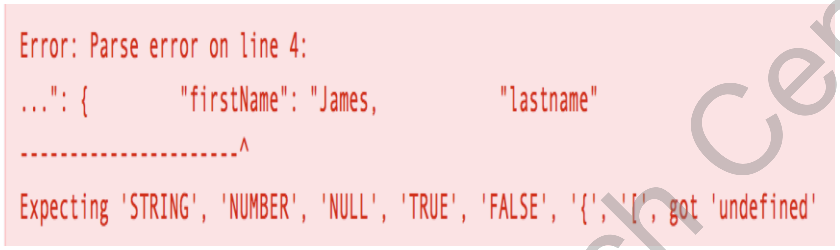
- **`JSONView`** displays the raw text even though the JSON document has errors.

- Chrome also offers **`JSONView`** for validating a JSON document.

- For modifying the JSON values during runtime, it is recommended converting the JSON document to a JavaScript object before using the built-in browser developer tools.

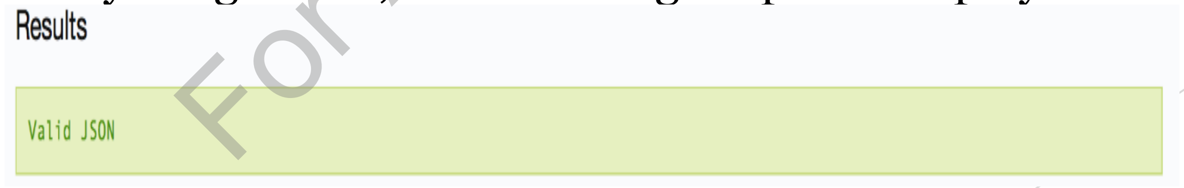
Online Tools and Editors

- `JSONLint` is an open source project that helps in validating JSON data.
- Using `JSONLint` or any other online tool is easy, as it only requires copying the JSON data to its online editor.
- In case of an error, the output similar to the following is displayed:



```
Error: Parse error on line 4:
...": {        "firstName": "James,           "lastname"
---------------------^
Expecting 'STRING', 'NUMBER', 'NULL', 'TRUE', 'FALSE', '{', '[', got 'undefined'
```

- Output shows the line number and the type of error that occurred.
- If everything is fine, the following output is displayed:

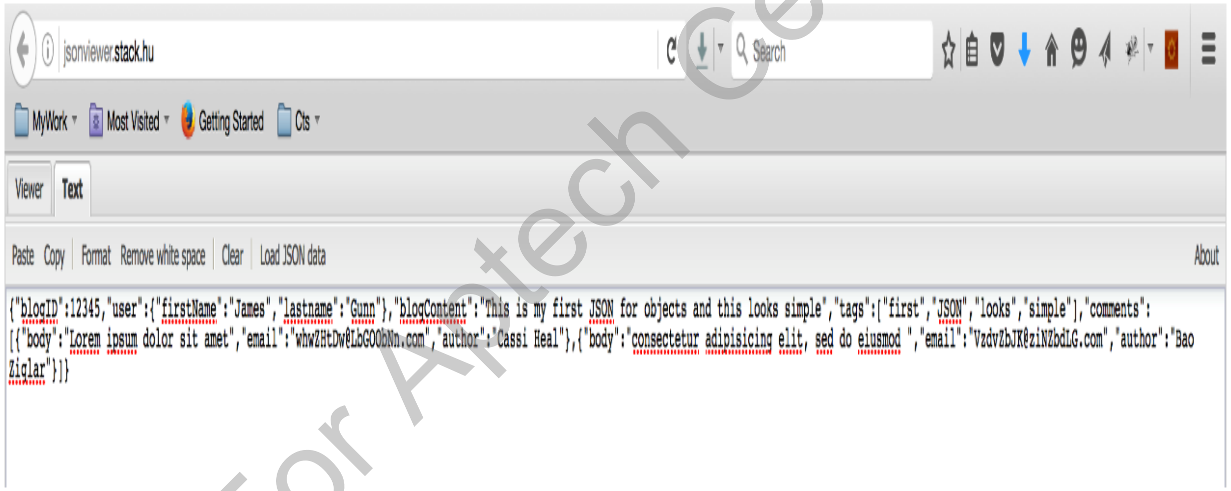


```
Results

Valid JSON
```

Online JSON Viewers 1-2

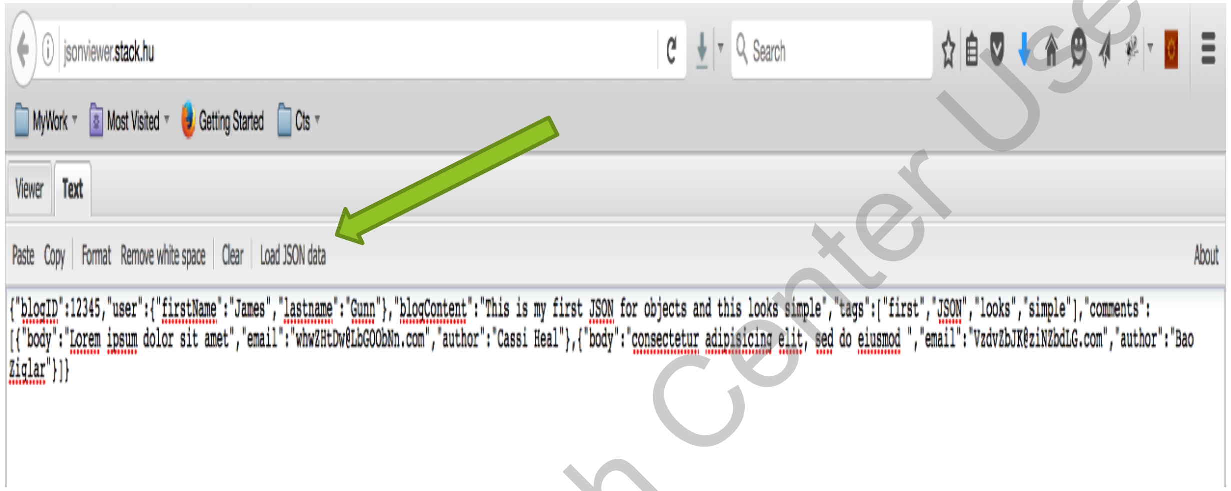
- While working with JSON, it is often required to use a JSON viewer to see how the document will look in the browser.
- One of the most widely used is jsonviewer.stack.hu and JSON text can be directly copied to this online viewer.
- The following screenshot shows the `jsonviewer.stack.hu`:



```
{"blogID":12345,"user":{"firstName":"James","lastname":"Gunn"},"blogContent":"This is my first JSON for objects and this looks simple","tags":["first","JSON","looks","simple"],"comments":
[{"body":"Lorem ipsum dolor sit amet","email":"whwZHtDw@LbGOObNn.com","author":"Cassi Heal"},{"body":"consectetur adipisicing elit, sed do eiusmod ","email":"VzdvZbJK@ziNZbdLG.com","author":"Bao
Ziglar"}]}
```

Online JSON Viewers 2-2

- Another way of supplying data to the viewer is by clicking **Load JSON data**.



```
{"blogID":12345,"user":{"firstName":"James","lastname":"Gunn"},"blogContent":"This is my first JSON for objects and this looks simple","tags":["first","JSON","looks","simple"],"comments":
[{"body":"Lorem ipsum dolor sit amet","email":"whwZHtDw@LbGOObNn.com","author":"Cassi Heal"},{"body":"consectetur adipisicing elit, sed do eiusmod ","email":"VzdvZbJK@ziNZbdLG.com","author":"Bao
Ziglar"}]}
```

- The viewer takes data from the URL you provide and loads data from that source.

- Once the JSON code is visible in the viewer, clicking **Format** formats the code.

Summary

- The two primary data types supported by JSON are primitive and structure.
- The primitive types are String, Number, Null, and Boolean. The structure types are Array and Object.
- Arrays allow storing various values of the same type in one variable.
- An object is an independent data type, having its own attributes.
- In JSON, arrays can also contain other arrays. This is called nesting of arrays.
- The two data structures supported by JSON are Collection of Name/Value Pairs and Ordered List of Values.
- JSON schema specifies the rules that defines the structure of a JSON document.
- JSON does not have any provision for documentation or comments.
- `JSONLint` is an open source project that helps in validating JSON data.