Experiment No. 07

Title: To implement data handling with JSON

Batch:A2 **Roll No.:16010421063** **Experiment No.:7**

Aim: To Implement data handling with JSON.

Resources needed: Notepad++, Web Browser

Theory:

JSON stands for **J**ava**S**cript **O**bject **N**otation. JSON is a **text format** for storing and transporting data. JSON is "self-describing" and easy to understand

- JSON stands for **J**ava**S**cript **O**bject **N**otation
- JSON is a lightweight data-interchange format
- JSON is plain text written in JavaScript object notation
- JSON is used to send data between computers
- JSON is language independent *****

Why Use JSON?

- The JSON format is syntactically similar to the code for creating JavaScript objects. Because of this, a JavaScript program can easily convert JSON data into JavaScript objects.
- Since the format is text only, JSON data can easily be sent between computers, and used by any programming language.
- JavaScript has a built in function for converting JSON strings into JavaScript objects: **JSON.parse()**
- JavaScript also has a built in function for converting an object into a JSON string: **JSON.stringify()**

Both JSON and XML can be used to receive data from a web server.

JSON Example

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

JSON.stringify()

- When sending data to a web server, the data has to be a string.
- Convert a JavaScript object into a string with JSON.stringify().
- Stringify a JavaScript Object

Imagine we have this object in JavaScript:

```
const obj = {name: "John", age: 30, city: "New York"};
```

Use the JavaScript function JSON.stringify() to convert it into a string.

```
const myJSON = JSON.stringify(obj);
```

The result will be a string following the JSON notation.

myJSON is now a string, and ready to be sent to a server:

Example

```
const obj = {name: "John", age: 30, city: "New York"};
const myJSON = JSON.stringify(obj);
```

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 - Parsing JSON

Imagine we received this text from a web server:

```
'{"name":"John", "age":30, "city":"New York"}'
```

Use the JavaScript function JSON.parse() to convert text into a JavaScript object:

```
const obj = JSON.parse('{"name":"John", "age":30, "city":"New York"}');
```

Make sure the text is in JSON format, or else you will get a syntax error.

Use the JavaScript object in your page:

Example

```
<p id="demo"></p>

<script>
```

```
document.getElementById("demo").innerHTML = obj.name;
</script>
```

Date objects are not allowed in JSON. If you need to include a date, write it as a string.

You can convert it back into a date object later:

Example

Convert a String into date

```
const text = '{"name":"John", "birth":"1986-12-14", "city":"New York"}';
const obj = JSON.parse(text);
obj.birth = new Date(obj.birth);

document.getElementById("demo").innerHTML = obj.name + ", " + obj.birth;
```

Storing Data

When storing data, the data has to be a certain format, and regardless of where you choose to store it, *text* is always one of the legal formats.

JSON makes it possible to store JavaScript objects as text.

Example

Storing data

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

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

JSON Server

Sending Data

If you have data stored in a JavaScript object, you can convert the object into JSON, and send it to a server:

Example

```
const myObj = {name: "John", age: 31, city: "New York"};
const myJSON = JSON.stringify(myObj); window.location
= "demo_json.php?x=" + myJSON;
```

Receiving Data

If you receive data in JSON format, you can easily convert it into a JavaScript object:

Example

```
const myJSON = '{"name":"John", "age":31, "city":"New York"}';
const myObj = JSON.parse(myJSON);
document.getElementById("demo").innerHTML = myObj.name;
```

JSON HTML

HTML Table

Make an HTML table with data received as JSON:

Example

```
const dbParam = JSON.stringify({table:"customers",limit:20});
const xmlhttp = new XMLHttpRequest();
xmlhttp.onload = function() {
  myObj = JSON.parse(this.responseText);
  let text = "<table border='1'>"
  for (let x in myObj) {
    text += "<tr><td>" + myObj[x].name + "</td></tr>";
  }
  text += "</table>"
  document.getElementById("demo").innerHTML = text;
}
xmlhttp.open("POST", "json_demo_html_table.php");
xmlhttp.setRequestHeader("Content-type", "application/x-www-form-urlencoded");
xmlhttp.send("x=" + dbParam);
```

HTML Drop Down List

Make an HTML drop down list with data received as JSON:

Example

```
const dbParam = JSON.stringify({table:"customers",limit:20});
const xmlhttp = new XMLHttpRequest();
xmlhttp.onload = function() {
  const myObj = JSON.parse(this.responseText);
  let text = "<select>"
  for (let x in myObj) {
    text += "<option>" + myObj[x].name + "</option>";
  }
  text += "</select>"
  document.getElementById("demo").innerHTML = text;
  }
}
xmlhttp.open("POST", "json_demo_html_table.php", true);
xmlhttp.setRequestHeader("Content-type", "application/x-www-form-urlencoded");
xmlhttp.send("x=" + dbParam);
```

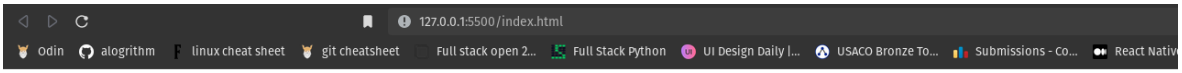
Activity:

1. **Convert JSON objects into string using JSON.sringify().**
2. **Replace any data in JSON object JSON.repalce()**
3. **Valid JSON sting into JSON using JSON.parse()**

Results: (Program printout with output)

```
<!DOCTYPE html>
<html>
   <head>
       <title>JSON Example</title>
   </head>
   <body>
       <p id="demo"></p>

       <script>
           var myObj = { name: "Arya", age:
19, city: "Mumbai" };

           var myJSON =
JSON.stringify(myObj);

           myJSON = myJSON.replace("Arya",
"Nair");
           var myNewObj =
```

```
JSON.parse(myJSON);

document.getElementById("demo").innerHTML =
                "Name: " +
                myNewObj.name +
                "<br>Age: " +
                myNewObj.age +
                "<br>City: " +
                myNewObj.city;
        </script>
    </body>
</html>
```



Name: Nair
Age: 19
City: Mumbai

Questions:

1. Why Jason is better than xml?

JSON (JavaScript Object Notation) and XML (eXtensible Markup Language) are both data interchange formats that are commonly used in web development. Here are some reasons why JSON is often considered better than XML:

1. Simpler and more lightweight: JSON is a simpler and more lightweight format than XML, with fewer syntax rules and less markup overhead. This makes it easier to read and write for both humans and machines.
2. Better performance: JSON is more efficient and faster to parse than XML, making it a better choice for data-intensive applications or APIs.
3. Native JavaScript support: JSON is a native format in JavaScript, which means that it can be easily parsed and used without any additional libraries or tools.
4. More widely used: JSON has become more widely used than XML, particularly in web development and APIs, which means that there are more tools and resources available for working with it.

That being said, XML is still a valid choice in some scenarios, particularly where more complex data structures or more advanced validation rules are required. However, for most web development purposes, JSON is often considered the better choice.

2. Write difference between JSON and Javascript

JSON (JavaScript Object Notation) and JavaScript are related concepts, but they are not the same thing. Here are some key differences between the two:

- Syntax: JSON is a lightweight data interchange format that uses a simpler syntax than JavaScript. While JSON is based on a subset of the JavaScript syntax, it is a distinct format with its own rules and conventions.
- Purpose: JavaScript is a programming language used to create dynamic and interactive web pages and applications, while JSON is a data format used to exchange data between different systems and languages.
- Execution: JavaScript code is executed in a web browser or server environment, while JSON data is typically transmitted between different systems or applications as plain text.
- Structure: In JavaScript, objects and arrays can be nested and manipulated dynamically, while JSON data is a static representation of data that must be parsed before it can be used.
- Standardization: JSON is a standardized format that can be used with any programming language, while JavaScript is a proprietary language developed and maintained by the Mozilla Foundation.

In summary, JavaScript is a programming language used for creating dynamic web pages, while JSON is a lightweight data interchange format used for exchanging data between different systems and languages.

Outcomes:

CO 4 Implement web application using React JS, Angular JS, JSON and CBOR

Conclusion: (Conclusion to be based on the outcomes achieved)

Implemented various JSON methods in html

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of faculty in-charge with date

References:

Books/ Journals/ Websites:

- "Web technologies: Black Book", Dreamtech Publications
- http://www.w3schools.com