JSON (JavaScript Object Notation)

What is JSON?

JSON (JavaScript Object Notation) *is a lightweight data-interchange format used to store and transfer data between:*

- *Client ↔ Server (Web APIs)*
- *Frontend ↔ Backend*
- *Applications ↔ Databases*

JSON is:

- *easy to read*
- *easy to write*
- *language independent*
- *widely used in APIs*

JSON is used because:

- *It is the standard format for **REST APIs***
- *It is supported directly by JavaScript*
- *It is compact and fast for data transfer*
- *It represents data in a structured way*

```
{
  "name": "Ashwini",
  "course": "Machine Learning",
  "marks": 92
}
```

JSON vs JavaScript Object (Very Important)

Feature	JavaScript Object	JSON
Type	Object	String format
Keys	can be without quotes	must be in double quotes `" "`
Values	can include functions, undefined	only data (no functions)
Used in	Program logic	Data exchange

```
let student = {
  name: "Ashwini",
  age: 25,
  isActive: true
};

console.log(student.name); // Ashwini
```

let *studentJSON =*

'{"name":"Ashwini","age":25,"isActive":true}';

console.log(studentJSON); // JSON is

text/string

JSON Data Types Supported

JSON supports only these data types:

- *String*
- *Number*
- *Boolean*
- *Null*
- *Object*
- *Array*

```
{
  "name": "Rahul",
  "age": 22,
  "isStudent": true,
  "address": null,
  "skills": ["Python", "JavaScript"],
  "marks": { "ml": 88, "ds": 90 }
}
```

JSON does NOT support:

- *functions*
- *undefined*
- *symbols*
- *comments*

JSON Methods in JavaScript

JavaScript provides built-in JSON methods:

1 `JSON.stringify()` → Object to JSON String

Converts JavaScript object into JSON string.

```
let user = {
  name: "Amit",
  city: "Dehradun",
  role: "Faculty"
};

let jsonText = JSON.stringify(user);

console.log(jsonText);

console.log(typeof jsonText);
```

2 JSON.parse() → JSON String to Object

Converts JSON string into JavaScript object.

Example:

```
let jsonData =
'{"name":"Amit","city":"Dehradun","role":"Faculty"}';


let obj = JSON.parse(jsonData);

console.log(obj);

console.log(obj.name);

console.log(typeof obj);
```

Working with JSON Arrays

JSON often contains arrays of objects (common in APIs).

Example JSON Array:

```
let jsonArray = `
[
  {"id": 1, "name": "Python", "level": "Easy"},
  {"id": 2, "name": "JavaScript", "level": "Medium"},
  {"id": 3, "name": "Machine Learning", "level":
"Hard"}
]
`;
```

```
let courses = JSON.parse(jsonArray);

console.log(courses[0].name); // Python
console.log(courses[2].level); // Hard
```

Nested JSON Objects Example

Example:

```
let student1 = {
  name: "Ashwini",
  roll: 101,
  marks: {
    ML: 90,
    DBMS: 85,
    OS: 88
  },
  skills: ["Python", "JavaScript", "Deep Learning"]
};

let jsonText1 = JSON.stringify(student1);
console.log(jsonText1);
```

```
console.log(student.marks.ML);     // 90

console.log(student.skills[1]);    // JavaScript
```

Pretty Printing JSON (Formatting)

By default, JSON.stringify gives a compact string.

We can format it using indentation.

Example:

```
let user = {
  name: "Ashwini",
  age: 26,
  skills: ["ML", "Python", "NLP"]
};

let prettyJSON = JSON.stringify(user, null, 4);
console.log(prettyJSON);
```

JSON with `fetch()` (Real API Example)

JSON is most commonly used in API calls.

Example: Fetching JSON data from API

```
async function getUser() {
  try {
    let res = await
fetch("https://jsonplaceholder.typicode.com/users/1");
    let data = await res.json();  // auto converts JSON → JS object
    console.log(data.name);
    console.log(data.email);
  } catch (error) {
    console.log("Error:", error.message);
  }
}
getUser();
```

Storing JSON in LocalStorage

*LocalStorage can only store **strings**, so we store JSON using stringify.*

Store:

```
let user = { name: "Ashwini", course: "ML" };


localStorage.setItem("userData",
JSON.stringify(user));

console.log("Saved!");
```

Retrieve:

```
let saved =
localStorage.getItem("userData");
let obj = JSON.parse(saved);

console.log(obj.name); // Ashwini
```

Thanks