Day-7

Learned about JSON

JSON (JavaScript Object Notation) is a popular data format used for exchanging information between applications and servers. It's known for being lightweight, human-readable, and easy for machines to parse. Here's a closer look at JSON:

Structure and Syntax:

- JSON data is built using key-value pairs, similar to dictionaries in some programming languages.
- Keys are always strings enclosed in quotes, and values can be strings, numbers, booleans, arrays (ordered lists of values), or even nested objects (collections of key-value pairs within curly braces).
- Everything is separated by commas, making the structure clear and easy to follow.

Here's an example of a simple JSON object:

JSON

```
{
  "name": "Alice",
  "age": 30,
  "city": "New York",
  "hasPet": true,
  "friends": ["Bob", "Charlie"]
}
```

In this example:

- "name", "age", "city", and "hasPet" are keys.
- "Alice", 30, "New York", and true are values.
- "friends" is a key with an array of values ("Bob" and "Charlie").

Why Use JSON?

- **Readability:** JSON is designed to be easily understood by both humans and machines. The clear structure and use of plain text make it easy to read and write.
- **Lightweight:** JSON files are compact, making them efficient for data transmission over networks.
- **Language Independence:** Unlike XML, JSON doesn't rely on specific tags. This makes it language-independent and usable across various programming languages.
- **Simplicity:** Compared to XML, JSON has a simpler syntax with fewer rules, making it easier to learn and use.

Common Uses of JSON:

- **APIs (Application Programming Interfaces):** JSON is a popular format for transmitting data between web applications and servers. APIs often use JSON to send and receive data requests and responses.

- **Data Interchange:** JSON is widely used for exchanging data between different systems and applications due to its simplicity and flexibility.
- **Configuration Files:** Some applications use JSON files to store configuration settings in a human-readable format.

In conclusion, JSON's readability, lightweight nature, and language independence make it a go-to format for data exchange on the web. It offers a simpler alternative to XML for many data management tasks.

JSON v/s XML

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

The following JSON and XML examples both define an employees object, with an array of 3 employees:

JSON Example

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

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>
```

JSON is Like XML Because

- Both JSON and XML are "self describing" (human readable)

- Both JSON and XML are hierarchical (values within values)
- Both JSON and XML can be parsed and used by lots of programming languages
- Both JSON and XML can be fetched with an XMLHttpRequest

JSON is Unlike XML Because

- JSON doesn't use end tag
- JSON is shorter
- JSON is quicker to read and write
- JSON can use arrays

The biggest difference is:

XML has to be parsed with an XML parser. JSON can be parsed by a standard JavaScript function.

Why JSON is Better Than XML

XML is much more difficult to parse than JSON.
JSON is parsed into a ready-to-use JavaScript object.

For AJAX applications, JSON is faster and easier than XML:

Using XML

- Fetch an XML document
- Use the XML DOM to loop through the document
- Extract values and store in variables

Using JSON

- Fetch a JSON string
- JSON.Parse the JSON string