JS | JSON and More

CodeSandBox Link : https://codesandbox.io/p/sandbox/dom-aj-aug-lvqz3v?file=%2Fsrc%2Findex.js%3A17%2C1

Intended For Beginners

Topics

1. JSON and Server Communication
2. JSON Methods and Error Handling (Try-Catch-Finally)
3. JSON and LocalStorage

---

1. JSON and Server Communication

1.1 What is JSON?

JSON (JavaScript Object Notation) is a lightweight data format used for storing and transporting data. It is easy for humans to read and write and easy for machines to parse and generate.

1.2 Fetching Data from a Server

Example 1: Fetching JSON Data

```
<!DOCTYPE html> <html lang="en"> <head> <meta charset="UTF-8"> <meta name
="viewport" content="width=device-width, initial-scale=1.0"> <title>Fetch
JSON Data</title> </head> <body> <h1>User Data</h1> <button id="loadDat
a">Load Users</button> <ul id="userList"></ul> <script src="index.js"></s
cript> </body> </html>
```

index.js:

```
document.getElementById('loadData').addEventListener('click', () => { fet
ch('<https://jsonplaceholder.typicode.com/users>') .then(response => resp
onse.json()) .then(data => { const userList = document.getElementById('us
erList'); userList.innerHTML = ''; // Clear the list before adding new it
ems data.forEach(user => { const li = document.createElement('li'); li.te
xtContent = `${user.name} (${user.email})`; userList.appendChild(li); });
}) .catch(error => console.error('Error fetching data:', error)); });
```

Exercise 1:

- Modify the code to display additional information about each user (e.g., address, company).
- Experiment with fetching data from a different API endpoint.

---

2. JSON Methods and Try-Catch-Finally Blocks

2.1 JSON Methods

- **JSON.stringify()**: Converts a JavaScript object or value to a JSON string.
- **JSON.parse()**: Converts a JSON string back to a JavaScript object.

Example 2: Working with JSON Methods

```
const user = { name: 'John Doe', email: 'john.doe@example.com', age: 30
}; // Convert JavaScript object to JSON string const jsonString = JSON.st
ringify(user); console.log('JSON String:', jsonString); // Convert JSON s
tring back to JavaScript object const jsonObject = JSON.parse(jsonStrin
g); console.log('JavaScript Object:', jsonObject);
```

2.2 Error Handling with Try-Catch-Finally

Example 3: Handling JSON Parsing Errors

```
const faultyJsonString = '{"name": "Jane Doe", "age": 25'; // Missing clo
sing brace try { const parsedData = JSON.parse(faultyJsonString); consol
e.log('Parsed Data:', parsedData); } catch (error) { console.error('Faile
d to parse JSON:', error.message); } finally { console.log('Parsing attem
pt finished.'); }
```

Exercise 2:

- Create an example where `JSON.stringify()` is used to prepare data for sending to a server.
- Experiment with different error scenarios in `try-catch-finally`.

---

3. JSON and LocalStorage

3.1 What is LocalStorage?

LocalStorage is a web storage API that allows you to store data locally in the user's browser. Data stored in LocalStorage persists even after the browser is closed.

3.2 Storing and Retrieving JSON Data

Example 4: Saving and Retrieving User Preferences

```
<!DOCTYPE html> <html lang="en"> <head> <meta charset="UTF-8"> <meta name
="viewport" content="width=device-width, initial-scale=1.0"> <title>Local
Storage Example</title> </head> <body> <h1>User Preferences</h1> <input t
ype="text" id="username" placeholder="Enter your name"> <button id="saveP
references">Save Preferences</button> <button id="loadPreferences">Load P
references</button> <script src="index.js"></script> </body> </html>
```

index.js:

```
document.getElementById('savePreferences').addEventListener('click', () =
> { const username = document.getElementById('username').value; if (usern
ame) { const preferences = { username: username, theme: 'dark' }; // Conv
ert preferences to JSON and save to LocalStorage localStorage.setItem('us
erPreferences', JSON.stringify(preferences)); alert('Preferences save
d!'); } }); document.getElementById('loadPreferences').addEventListener
('click', () => { const storedPreferences = localStorage.getItem('userPre
ferences'); if (storedPreferences) { const preferences = JSON.parse(store
dPreferences); document.getElementById('username').value = preferences.us
ername; alert('Preferences loaded!'); } else { alert('No preferences foun
d.'); } });
```

Exercise 3:

- Modify the example to save and load more user preferences (e.g., language, font size).
- Add a button to clear the stored preferences.

Next Steps: Try building a small web app that uses these techniques, such as a task manager that saves tasks locally or a user settings page that loads preferences from LocalStorage.