Lesson 2: Complex Data Types (Arrays & Objects)

Lesson Overview

Duration: ~2 hours

Goal: Teach students how to use JavaScript arrays and objects effectively, including key methods and operations.

Format: Explanation + Interactive Coding Exercises + Mini Project

---

1. Introduction to Arrays (50 min)

What is an Array?

An **array** is a data structure used to store multiple values in a single variable. Arrays maintain the order of elements and are accessed using **zero-based indexing**.

Declaring and Initializing Arrays

Arrays can be created using **array literals** or the `Array` constructor.

```
// Using array literal
const fruits = ["Apple", "Banana", "Cherry"];

// Using Array constructor
const numbers = new Array(1, 2, 3, 4, 5);
```

Accessing Array Elements

Elements in an array are accessed using **indices**, starting from `0`.

```
console.log(fruits[0]); // Apple
console.log(fruits[1]); // Banana
```

```
console.log(fruits[fruits.length - 1]); // Cherry
```

Checking if a Variable is an Array

```
console.log(Array.isArray(fruits)); // true
```

Array Methods for Adding & Removing Elements

1️⃣ Adding Elements

```
fruits.push("Mango");  // Adds to the end
fruits.unshift("Grapes"); // Adds to the beginning
console.log(fruits);
```

2️⃣ Removing Elements

```
fruits.pop();  // Removes the last element
fruits.shift(); // Removes the first element
console.log(fruits);
```

3️⃣ Removing/Adding Elements at a Specific Index

```
fruits.splice(1, 1); // Removes 1 element at index 1 (Banana)
fruits.splice(2, 0, "Peach"); // Inserts "Peach" at index 2
console.log(fruits);
```

---

2. Iterating Over Arrays (30 min)

Iterating over arrays allows us to perform operations on each element.

Using a `for` Loop

```
for (let i = 0; i < fruits.length; i++) {
    console.log(fruits[i]);
}
```

Using `forEach()`

```
fruits.forEach((fruit, index) => {
    console.log(`${index}: ${fruit}`);
});
```

Using `map()` (Creates a New Array)

```
const upperCaseFruits = fruits.map(fruit => fruit.toUpperCase());
console.log(upperCaseFruits);
```

Using `filter()` (Filters Elements Based on Condition)

```
const longNamedFruits = fruits.filter(fruit => fruit.length > 5);
console.log(longNamedFruits);
```

---

3. Introduction to Objects (30 min)

What is an Object?

An **object** is a collection of **key-value pairs**, where each key is a string and the values can be of any type.

```
const student = {
    name: "Alice",
    age: 14,
    score: 90
};
```

```
console.log(student.name); // Alice
console.log(student["score"]); // 90
```

Modifying Objects

```
student.grade = "A";  // Adding a new property
student.score = 95;    // Modifying an existing property
delete student.age;    // Removing a property
console.log(student);
```

Looping Through Object Properties

```
for (let key in student) {
    console.log(`${key}: ${student[key]}`);
}
```

---

4. Mini Project: Student Score Manager (40 min)

Goal:

- Allow users to dynamically add and remove student scores.
- Display the list of students and their scores.
- Filter students based on their scores.

Start Code (Students Complete the Missing Parts)

```
let students = [];

function addStudent() {
    // TODO: Student need to complete this function to add a student object to t
he array
}
```

```
function removeStudent() {
    // TODO: Student need to complete this function to remove a student object from the array
}

function showHighScorers() {
    // TODO：Student need to complete this function to show high scorers
}

function updateStudentList() {
    //TODO: Student need to complete this function to update the student list
}

function getName() {
    return document.getElementById("studentName").value.trim();
}

function getScore() {
    return parseInt(document.getElementById("studentScore").value);
}

function putOnScreen(String) {
    document.getElementById("studentList").textContent = String;
}

document.getElementById("addStudent").addEventListener("click", addStudent);
document.getElementById("removeStudent").addEventListener("click", removeStudent);
document.getElementById("showHighScorers").addEventListener("click", showHighScorers);
```

Extensions:

- Allow the user to input student names and scores dynamically.

- Sort students by score in descending order.
- Allow updating a student’s score dynamically.

---

5. Recap & Q&A (10 min)

- Quick review of the main concepts.
- Ask students what they found easy/hard.
- Provide hints for the next lesson (working with structured data and API handling).