5-Arrays and Objects

Chapter 5: Arrays and Objects

In JavaScript, arrays and objects are foundational for organizing and manipulating data efficiently. This chapter explores their syntax, usage, and practical applications in detail.

5.1 Understanding Arrays

An array is a collection of items stored in a single variable. Arrays are particularly useful for working with lists of data.

5.1.1 Declaring and Initializing Arrays

You can create arrays using square brackets `[]`.

Example:

```javascript
let fruits = ["apple", "banana", "cherry"];
```

Arrays can contain different data types:

```javascript
let mixedArray = [42, "hello", true];
```

5.1.2 Accessing Array Elements

Access elements using their index (starting from 0).

Example:

```javascript
console.log(fruits[0]); // Output: "apple"
```

Modify elements by assigning new values:

```javascript
fruits[1] = "blueberry";
console.log(fruits); // Output: ["apple", "blueberry", "cherry"]
```

5.1.3 Array Methods

JavaScript provides built-in methods to work with arrays. Here are some commonly used ones:

1. **Adding Items:**
   - `push()` adds an item to the end.

     ```
     fruits.push("grape");
     console.log(fruits); // Output: ["apple", "blueberry", "cherry",
     "grape"]
     ```

   - `unshift()` adds an item to the beginning.

     ```
     fruits.unshift("kiwi");
     console.log(fruits); // Output: ["kiwi", "apple", "blueberry",
     "cherry", "grape"]
     ```

2. **Removing Items:**
   - `pop()` removes the last item.

     ```
     fruits.pop();
     console.log(fruits); // Output: ["kiwi", "apple", "blueberry",
     "cherry"]
     ```

   - `shift()` removes the first item.

     ```
     fruits.shift();
     console.log(fruits); // Output: ["apple", "blueberry", "cherry"]
     ```

3. **Other Useful Methods:**
   - `length`: Gets the number of items.

     ```
     console.log(fruits.length); // Output: 3
     ```

   - `indexOf()`: Finds the index of a specific item.

     ```
     console.log(fruits.indexOf("blueberry")); // Output: 1
     ```

   - `splice()`: Adds or removes items at a specific position.

     ```
     fruits.splice(1, 0, "orange");
     ```

```
console.log(fruits); // Output: ["apple", "orange", "blueberry",
"cherry"]
```

- `slice()`: Extracts a portion of the array.

  ```
  let newFruits = fruits.slice(1, 3);
  console.log(newFruits); // Output: ["orange", "blueberry"]
  ```

5.1.4 Iterating Over Arrays

Use loops to process array elements.

Example:

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

Use the `forEach()` method for cleaner iteration:

```
fruits.forEach((fruit) => console.log(fruit));
```

5.2 Understanding Objects

Objects are collections of key-value pairs and are ideal for storing related data. Each key is a `string`, and its value can be of any data type.

5.2.1 Declaring and Initializing Objects

Create objects using curly braces `{}`.

Example:

```
let person = {
  name: "Rick",
  age: 30,
  profession: "Developer"
};
```

Access properties using dot notation or bracket notation:

```javascript
console.log(person.name); // Output: "Rick"
console.log(person["age"]); // Output: 30
```

Modify properties:

```javascript
person.age = 31;
console.log(person.age); // Output: 31
```

Add new properties:

```javascript
person.hobby = "coding";
console.log(person.hobby); // Output: "coding"
```

5.2.2 Nested Objects

Objects can contain other objects, arrays, or even functions.

Example:

```javascript
let person = {
  name: "Rick",
  age: 30,
  skills: ["JavaScript", "HTML", "CSS"],
  address: {
    city: "Centerville",
    state: "Utah"
  }
};

console.log(person.address.city); // Output: "Centerville"
```

5.2.3 Iterating Over Object Properties

Use a `for...in` loop to iterate over keys.

Example:

```javascript
for (let key in person) {
  console.log(key + ": " + person[key]);
}
```

5.3 Combining Arrays and Objects

Arrays and objects often work together. For example, you can create an array of objects:

Example:

```
let people = [
  { name: "Rick", age: 30 },
  { name: "Sarah", age: 25 }
];

people.forEach((person) => console.log(person.name));
```

5.4 Practical Examples

1. **Task Manager:**

```
let tasks = [
  { title: "Learn JavaScript", completed: true },
  { title: "Build a project", completed: false }
];

tasks.forEach((task) => {
  console.log(`${task.title} - ${task.completed ? "Done" : "Pending"}`);
});
```

2. **Inventory Tracker:**

```
let inventory = [
  { item: "Laptop", quantity: 10 },
  { item: "Phone", quantity: 5 }
];

inventory.push({ item: "Tablet", quantity: 3 });
inventory.forEach((entry) =>
  console.log(`${entry.item}: ${entry.quantity} in stock`)
);
```

3. **Interactive Example:**
   Create a simple shopping cart with JavaScript:

```javascript
let cart = [];

function addToCart(product, price) {
  cart.push({ product, price });
}

function showCart() {
  cart.forEach((item) => console.log(`${item.product}: $${item.price}`));
}

addToCart("Apple", 1.5);
addToCart("Banana", 0.75);
showCart();
```