ProSensia PVT LTD
M Farooq Sajid
JavaScript Objects & Object Methods

1: Introduction to Objects

In JavaScript, an object is a standalone entity, with properties and type. It's similar to real-life objects, like a car, that have properties (color, brand) and behaviors (drive, brake). Objects are one of the most important features in JavaScript, as they allow you to store multiple values and functionality within a single structure.

Objects are created using either object literals, constructors, or the `Object.create()` method.

Example of an object literal:

```
const person = {
 firstName: "John",
 lastName: "Doe",
 age: 30,
 isEmployed: true
};
```

2: Accessing and Modifying Object Properties

Object properties can be accessed using dot notation or bracket notation. You can also add, modify, or delete properties.

Examples:

```
console.log(person.firstName);      // Access using dot notation
console.log(person["lastName"]);    // Access using bracket notation

person.age = 35;                 // Modify property
person.gender = "male";          // Add property
delete person.isEmployed;        // Delete property
```

3: Object Methods

Objects can also have methods, which are functions defined inside the object. Methods allow objects to perform actions using their data.

Example:

```
const person = {
 firstName: "John",
 lastName: "Doe",
 fullName: function() {
  return this.firstName + " " + this.lastName;
 }
};

console.log(person.fullName());
```

The `this` keyword refers to the current object and is used to access its properties.

4: Built-in Object Methods

JavaScript provides built-in methods for objects, which are useful for working with properties and data. Common methods include:

```
Object.keys(obj)      // Returns an array of property names
Object.values(obj)    // Returns an array of property values
Object.entries(obj)   // Returns an array of key/value pairs
Object.hasOwnProperty(key) // Checks if the property exists
```

Example:

```
const car = {
 brand: "Toyota",
 model: "Corolla",
 year: 2020
};

console.log(Object.keys(car));     // ["brand", "model", "year"]
console.log(Object.values(car));   // ["Toyota", "Corolla", 2020]
console.log(Object.entries(car));  // [["brand", "Toyota"], ["model",
"Corolla"], ["year", 2020]]
```

5: Practical Example with Object and Methods

Below is a practical example that defines an object with multiple properties and methods to demonstrate real-world usage.

```
const student = {
  name: "Alice",
  marks: [85, 90, 78],
  calculateAverage: function() {
    let sum = this.marks.reduce((a, b) => a + b, 0);
    return sum / this.marks.length;
  },
  displayInfo: function() {
    return `${this.name}'s average marks are
${this.calculateAverage()}`;
  }
};

console.log(student.displayInfo());
```

This object stores student data, calculates average marks, and displays the result using object methods. It demonstrates how methods can encapsulate behavior tied to object data.