

**JAVASCRIPT  
OBJECTS { }**



# Objects in JavaScript

---

A Beginner's Guide

# What Are Objects in JavaScript?

In JavaScript, an **object** is a collection of related data and/or functionality. These usually consist of several variables (called **properties**) and functions (called **methods**) grouped together. Objects can represent real-world entities, such as a car, a person, or more abstract concepts like a user account or an order in an online store.

## Why Use Objects?

Objects help you structure your code by grouping related data and behavior together. Instead of having separate variables and functions scattered around, you can bundle them into an object, making your code more organized and easier to manage.

---

# Creating an Object

## 1. Object Literal


```
const person = {  
  firstName: "John",  
  lastName: "Doe",  
  age: 30,  
  sayHello: function() {  
    console.log("Hello, " + this.firstName + " " +  
    this.lastName);  
  }  
};
```

```
console.log(person.firstName); // Output: John  
console.log(person.age);      // Output: 30  
person.sayHello();            // Output: Hello, John Doe
```

- **firstName, lastName, and age** are properties of the person object.
- **sayHello** is a method of the person object.

# 'new Object()' Syntax

Another way to create an object is by using the new Object() syntax:



```
const person = new Object();
person.firstName = "John";
person.lastName = "Doe";
person.age = 30;
person.sayHello = function() {
  console.log("Hello, " + this.firstName + " " + this.lastName);
};
```

This way is less commonly used compared to object literals, but it achieves the same result.

---

# Accessing and Modifying Object Properties

You can access object properties in two ways:

**1.Dot Notation:** `objectName.propertyName`

**2.Bracket Notation:** `objectName["propertyName"]`



```
console.log(person.firstName);    // Dot notation: Output: John  
console.log(person["lastName"]); // Bracket notation: Output: Doe
```

# Adding and Removing Properties



```
person.job = "Developer";  
console.log(person.job); // Output: Developer
```

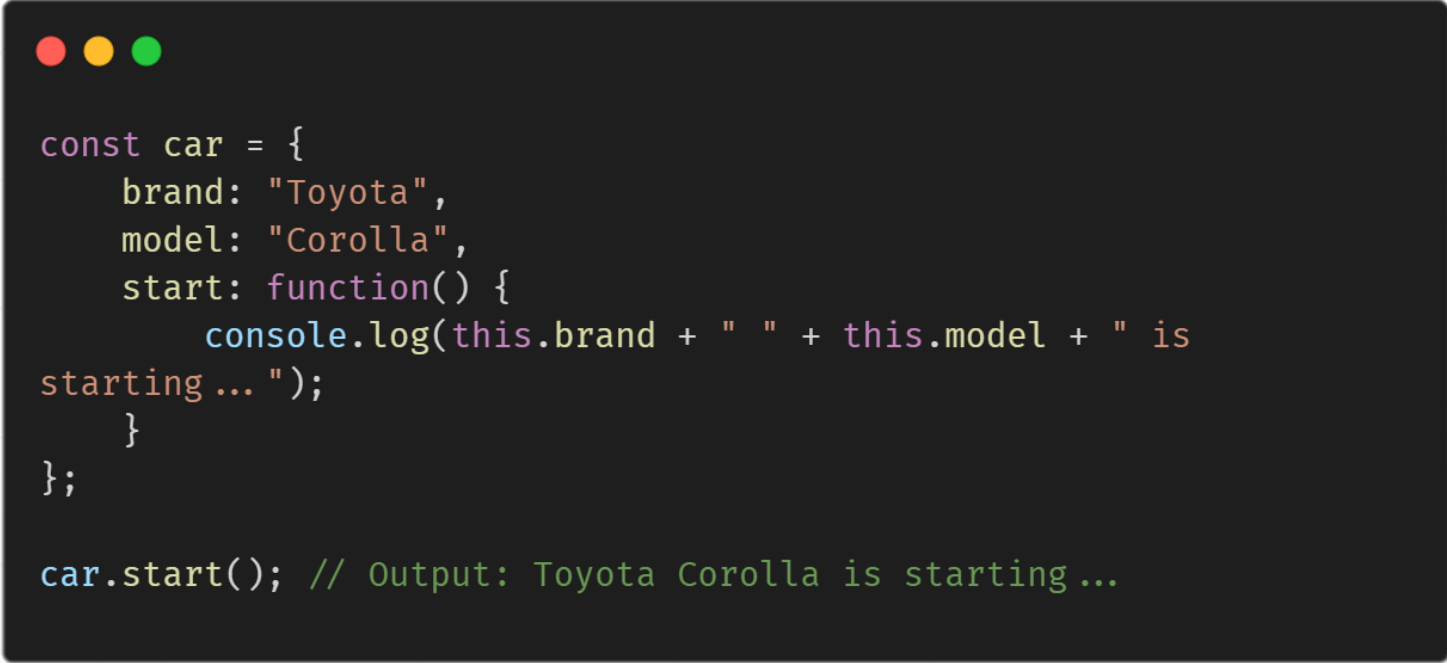


```
delete person.age;  
console.log(person.age); // Output: undefined
```

---

# Methods in Objects

Methods are functions that belong to an object. They can perform actions using the data within the object.



```
const car = {  
  brand: "Toyota",  
  model: "Corolla",  
  start: function() {  
    console.log(this.brand + " " + this.model + " is  
starting... ");  
  }  
};  
  
car.start(); // Output: Toyota Corolla is starting...
```

Inside a method, the `this` keyword refers to the object that is calling the method. In the example above, `this.brand` refers to the `brand` property of the `car` object.

---

# Nested Objects



```
const student = {  
  name: "Alice",  
  grades: {  
    math: 90,  
    science: 85,  
    english: 88  
  }  
};  
  
console.log(student.grades.math); // Output: 90
```



# Summary

- **Objects:** Containers for storing data and functionality.
- **Properties:** Variables that belong to an object.
- **Methods:** Functions that belong to an object.
- **Dot and Bracket Notation:** Ways to access object properties.
- **`this` Keyword:** Refers to the object that calls the method.

Understanding objects is fundamental in JavaScript, as they are used extensively throughout the language. Once you're comfortable with objects, you'll find it easier to work with more advanced concepts like object-oriented programming, JSON, and manipulating the DOM.

---