How to define objects in Javascript. Explain with Example.

The most common and simplest way to create an object is using the **object**, which is a comma-separated list of key-value pairs enclosed in curly braces {}.

Example:

```
const person = {
  firstName: "John",
  lastName: "Doe",
  age: 30,
  isEmployed: true,
  greet: function() {
    console.log("Hello, my name is " + this.firstName + " " + this.lastName);
  }
};

// Accessing object properties
console.log(person.firstName); // Output: John
console.log(person.age); // Output: 30

// Calling object method
person.greet(); // Output: Hello, my name is John Doe
```

Explanation:

Properties: The person object has four properties: firstName, lastName, age, and isEmployed. These are key-value pairs where the keys are strings (e.g., "firstName") and the values are data (strings, numbers, booleans, etc.).

Method: The greet property is a method (a function defined inside the object). You can call this method using the dot notation like person.greet().

How do you access properties and methods of Javascript objects. Explain with example.

Dot Notation (.): The easiest and most commonly used method to access properties and methods, suitable for most use cases.

```
  const car = {
  brand: "Toyota",
  model: "Corolla",
  year: 2020,
  start: function() {
    console.log("The car has started.");
  }
};

// Accessing properties
console.log(car.brand);  // Output: Toyota
console.log(car.year);   // Output: 2020
```

```
// Modifying a property
car.year = 2021;
console.log(car.year);   // Output: 2021

// Accessing and calling a method
car.start();  // Output: The car has started.
```

Explanation:

- Properties: You can access properties like brand, model, and year directly using car.brand, car.year, etc.
- Methods: The start method can be invoked by using car.start().

Bracket Notation ([]): Used when the property name is dynamic, contains spaces, or includes characters not allowed in dot notation.

```
const car = {
  brand: "Toyota",
  model: "Corolla",
  year: 2020,
  "fuel type": "Petrol", // Property with space
  start: function() {
    console.log("The car has started.");
  }
};
```

```
// Accessing properties using bracket notation
console.log(car["brand"]);  // Output: Toyota
console.log(car["fuel type"]);  // Output: Petrol

// Modifying a property
car["year"] = 2022;
console.log(car["year"]);   // Output: 2022

// Accessing and calling a method
car["start"]();  // Output: The car has started.
```

Explanation:

- Properties: With bracket notation, you can access properties like brand and year using car["brand"], and properties with spaces like "fuel type" using car["fuel type"].
- Methods: The start method can be accessed using car["start"]().

Both notations allow you to access and modify properties and methods, as well as work with nested objects.

what is the use of this keyword in javascript?

The this keyword in JavaScript refers to the **context** in which a function or method is being executed. Its value depends on how and where the function is called. It provides a way to refer to the object that is currently executing the code, and it helps in accessing the properties and methods of that object.

Global context: this refers to the global object (window in browsers, global in Node.js).

Object method: this refers to the object that owns the method.

Regular function: this refers to the global object (non-strict mode) or undefined (strict mode).

Constructor function: this refers to the new object being created by the constructor.

Arrow function: this inherits from the surrounding (lexical) context.

Event handler: this refers to the DOM element that fired the event.