**Object Oriented Programming in JavaScript**

**1: Classes**

Classes are **blueprints** of an Object. A class can have many Objects because the class is a **template** while Objects are **instances** of the class or the concrete implementation.

Before we move further into implementation, we should know unlike other Object Oriented languages there are **no classes in JavaScript** we have only Object.

To be more precise, JavaScript is a prototype-based Object Oriented Language, which means it doesn’t have classes, rather it defines behaviors using a constructor function and then reuses it using the prototype.

**Example:**

Let’s use ES6 classes then we will look at the traditional way of defining an Object and simulate them as classes.

// Defining class using es6

class Vehicle {

    constructor(name, maker, engine) {

        this.name = name;

        this.maker = maker;

        this.engine = engine;

    }

    getDetails() {

        return (`The name of the bike is ${this.name}.`)

    }

}

// Making object with the help of the constructor

let bike1 = new Vehicle('Hayabusa', 'Suzuki', '1340cc');

let bike2 = new Vehicle('Ninja', 'Kawasaki', '998cc');

console.log(bike1.name); // Hayabusa

console.log(bike2.maker); // Kawasaki

console.log(bike1.getDetails());

**2.** [**Object**](https://www.geeksforgeeks.org/objects-in-javascript/)**:**

An Object is a **unique** entity that contains **properties** and **methods**.

For example “a car” is a real-life Object, which has some characteristics like color, type, model, and horsepower and performs certain actions like driving.

The characteristics of an Object are called Properties in Object-Oriented Programming and the actions are called methods.

An Object is an **instance** of a class. Objects are everywhere in JavaScript, almost every element is an Object whether it is a function, array, or string

.

**Note:** A Method in javascript is a property of an object whose value is a function.

The object can be created in two ways in JavaScript:

* **Object Literal**
* **Object Constructor**

**Example:**Using an Object Literal.

// Defining object

let person = {

    first\_name: 'Mukul',

    last\_name: 'Latiyan',

    //method

    getFunction: function () {

        return (`The name of the person is

        ${person.first\_name} ${person.last\_name}`)

    },

    //object within object

    phone\_number: {

        mobile: '12345',

        landline: '6789'

    }

}

console.log(person.getFunction());

console.log(person.phone\_number.landline);

**Example:**Using an Object Constructor.

// Using a constructor

function person(first\_name, last\_name) {

    this.first\_name = first\_name;

    this.last\_name = last\_name;

}

// Creating new instances of person object

let person1 = new person('Mukul', 'Latiyan');

let person2 = new person('Rahul', 'Avasthi');

console.log(person1.first\_name);

console.log(`${person2.first\_name} ${person2.last\_name}`);