JavaScript objects are somewhat different in the way they are created. There is no requirement for a class as such.

**Object Creation👇**

We can create objects in many ways in javascript, let’s look at each one of them.

1. Object literal (Direct way) — Object literals are a comma-separated list of key-value pairs wrapped in curly braces. Object literal property values can be of any data type, including array literals, functions, nested object literals or primitive data type.

var student = {  
id: 1,  
name: “deepak”,  
age: “27”,  
updateAddress: () => {  
// logic to update address  
},  
grade: [‘A’, ‘A+’, ‘A’]  
}

2. Object.create()— the method creates a new object with the specified prototype and properties of the old object.

*Note: Every JavaScript function has a prototype property which .is empty by default. We may attached methods or properties to prototype.*

// syntax — Object.create(prototype[, propertiesObject])  
var newStudent = Object.create(student);  
// this create a new object with old object added in its prototype // chain

Object Instance— The use of Objectconstructor in conjunction with the “new” keyword allows us to initialize new objects.

Let’s take a look by an example:

const newObj = new Object();  
newObj.name = ‘Deepak’;  
newObj.location = ‘Delhi, India’;

4. Object Constructor —

However, the above method using new Object()is not well suited to programs that require the creation of multiple objects of the same kind, as it would involve repeatedly writing the above lines of code for each such object.

Constructors can be useful when we need a way to create an object “type” that can be used multiple times without having to redefine the object every time and this could be achieved using the Object Constructor function.

Let’s take a look by an example:

function Vehicle(name, model) {  
this.name = name;  
this.model = model;  
}  
  
let car1 = new Vehicle(‘Fiesta’, ‘2019’);  
let car2 = new Vehicle(‘DC avanti’, ‘2018’);

We created two objects with the same property but with different value.

6. Object.fromEntries() — method transforms a list of key-value pairs into an object. Let’s take a look by an example

const entries = new Map([  
[‘foo’, ‘bar’],  
[‘baz’, 42]  
]);

const obj = Object.fromEntries(entries);

console.log(obj);  
// expected output: Object { foo: “bar”, baz: 42 }