# OBJECT AND ITS INTERNAL REPRESENTATION IN JAVASCRIPT

Objects, in JavaScript, is its most important data-type and forms the building blocks for modern JavaScript. These objects are quite different from JavaScript’s primitive data-types (Number, String, Boolean, null, undefined and symbol) in the sense that while these primitive data-types all store a single value each.

Objects are more complex and each object may contain any combination of these primitive data-types as well as reference data-types. An object, is a reference data type. Variables that are assigned a reference value are given a reference or a pointer to that value. That reference or pointer points to the location in memory where the object is stored. The variables don’t actually store the value. Objects in JavaScript may be defined as an unordered collection of related data, of primitive or reference types, in the form of “key: value” pairs.

## OBJECT CREATION:

1. **Direct Method**: Object literals are a comma-separated list of key-value pairs wrapped in curly braces.

**Ex**: var student = {  
id: 1,  
name: “deepak”,  
age: “27”,  
}

1. **Object.create( ):** the method creates a new object with the specified prototype and properties of the old object.

**Syntax** — Object.create(prototype[, propertiesObject])

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

**Ex:**

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

1. **Object Constructor** — 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.

**Ex:**

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 values.

1. **Object.assign()** —this is another method to create a new object from other objects.

It copies the values of all enumerable own properties from one or more source objects to a target object. It will return the target object.

**Ex:**

const target = { a: 1, b: 2 };  
const source = { b: 4, c: 5 };

const returnedTarget = Object.assign(target, source);

console.log(target);  
// expected output: Object { a: 1, b: 4, c: 5 }

console.log(returnedTarget);  
// expected output: Object { a: 1, b: 4, c: 5 }

1. **Object.fromEntries()** — method transforms a list of key-value pairs into an object

**Ex**:

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

const obj = Object.fromEntries(entries);

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