Objects and its internal representation in Javascript

Objects are important data types in 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 (depending on their types), objects store multiple values of different types. Primitive data types contain one value but Objects can hold many values in form of name: value pair. These names can be variables or functions and are called properties and methods, respectively, in the context of an object.

For Eg. If your object is a Car, it will have properties like Model, Brand, Color, Year, etc. and methods like startEngine, stopEngine etc.

Every object has some property associated with some value. These values can be accessed using these properties associated with them.

*var Car = new Object();*

Car.Brand = 'Audi';

Car.Model = 'R8';

Car.Year = 2015;

Car.Color = ‘Red’;

After creating Car object, the value inside the object can be accessed using names.

i.e.

Car.Model

Output: R8

These values can be accessed using brackets notation also.

Car[Model]

Output: R8

The syntax for adding a property to an object is :

*ObjectName.ObjectProperty = propertyValue;*

The syntax for deleting a property from an object is:

*delete ObjectName.ObjectProperty;*

The syntax to access a property from an object is:

*objectName.property*  //or *objectName["property”]* //or *objectName[expression]*

# Object Methods

An object method is an object property containing a function definition. i.e.,

Let’s assume to start the car there will be a mechanical functionality in our Car object

*function(){return engine.on}*

A simple definition for Java Script Object methods is “Methods are actions that can be performed on objects.”

# Create JavaScript Object with Object Literal

One of easiest way to create a javascript object is object literal, simply define the property and values inside curly braces as shown below

*let Car = {Brand: 'Audi', Model:'R8i', year:'2015', Color:’Red’};*

# Create JavaScript Object with Constructor

Constructor is nothing but a function and with help of new keyword, constructor function allows to create multiple objects of same type

*function Vehicle(brand, model, year, color) {*

*this.brand = brand;*

*this.model= model;*

*this.year= year;*

*this.color= color;*

*}*

*let car1 = new Vehicle(’Audi’, 'R8’, ‘2015’, ‘Red’);*

*let car2 = new Vehicle(’Nissan’, 'GT-R’, ‘2020’ ‘Black’)*

*console.log(car1.model); //Output: R8*

*console.log(car2.model); //Output: GT-R*

# Using the JavaScript Keyword ‘new’

The following example also creates a new JavaScript object with four properties:

*var Car = new Object();*

*Car.Brand = “Audi”;*

*Car.Model = “R8”;*

*Car.Year = 2015;*

*Car.Color = “Red”;*

# Using the Object.create method

Objects can also be created using the Object.create() method. This method can be very useful, because it allows you to choose the prototype object for the object you want to create, without having to define a constructor function.

*// Carl properties and method*

*var Car = {*

*type: 'Gasoline powered', // Default value of properties*

*displayType: function() { // Method which will display type of Animal*

*console.log(this.type);*

*}*

*};*

*// Create new Car type called Audi*

*var Audi = Object.create(Car);*

*Audi.displayType(); // Output :Gasoline powered*

*// Create new Car type called Tesla*

*var Tesla = Object.create(Car);*

*Tesla.type = 'Electric powered';*

*tesla.displayType(); // Output: Electric powered*