# Write a blog about objects and its internal representation in Java script?

* JavaScript is an object-oriented programming language, so everything in JavaScript is an object.
* A JavaScript object is like a real-world entity having state and behavior, for example, a car. We can take a car as an object. It will have a state like color and model. It will also have behaviors like accelerating and brake.
* JavaScript is template based and we can create objects without the need of having a class. Generally, an object is accompanied by a class that defines an object’s blueprints but JavaScript doesn’t require any class to be defined for objects.

**Creating a JavaScript Object**

There are three ways in which we can create a JavaScript object. Let’s go through each method:

* We can use the object literal to create and define a JavaScript object. In this method, an object can be created in a way very similar to that of defining a dictionary, with keys and associated values.
  + In this case, one of the keys can be age and its value is 21.

<script>

      var student = {

        name: "Chris Hemsworth",

        age: 21,

        branch: "Computer Science",

      };

      document.getElementById("demo").innerHTML = student.name + " of the age " + student.age + " studies " + student.branch + ".";

    </script>

* We can use the new [keyword](https://www.simplilearn.com/tutorials/javascript-tutorial/javascript-this-keyword) to create and define an object. This method is similar to how objects are created in the [Java programming language](https://www.simplilearn.com/tutorials/asp-dot-net-tutorial/dotnet-vs-java). We can easily add properties to JavaScript objects too like illustrated below.

<script>

      var student = new Object();

      student.name = "Chris Hemsworth";

      student.age = 21;

      student.branch = "Computer Science";

      document.getElementById("demo").innerHTML = student.name + " of the age " + student.age + " studies " + student.branch + ".";

    </script>

* We can also use an object constructor to initialize a JavaScript object. This method is also commonly known as object prototyping. The constructor takes in a few parameters and using those parameters, we define the value for each property in an object.

    <script>

      function stud(name, age, branch) {

        this.name = name;

        this.age = age;

        this.branch = branch;

      }

      var student = stud("Chris Hemsworth", 21, "Computer Science");

      document.getElementById("demo").innerHTML = student.name + " of the age " + student.age + " studies " + student.branch + ".";

    </script>

## **JavaScript Object Properties**

A JavaScript object is basically a collection of unordered properties. Values associated with a JavaScript object are called its properties. Properties can usually be added, updated, and deleted, excluding read-only properties.

Let’s now look at the few ways for accessing object properties:

<!--ways to access properties of objects-->

    <script>

      var student = {

        name: "Chris Hemsworth",

        age: 21,

        branch: "Computer Science",

      };

      //first method

      student.age;

      //second method

      student[age];

      //third method

      x = "age";

      student[x];

    </script>

* We first define an object and name it student and add a few relevant properties.
* The first method is to access the property by using the dot(.) notation - object.property
* The second method is by using square brackets - object[property]
* Lastly, we can store a property name, in the form of a string, in a variable and then use that variable to access the associated property.

## **JavaScript Object Methods**

Actions that can be performed on a JavaScript object are called methods.

<script>

      let user = {

        name: "Chris",

        age: 24,

      };

      // create a new function that we will use as an object method

      function sayHi() {

        alert("Hello!");

      }

      // then add the previously created method

      user.sayHi = sayHi;

      // this will print username on the screen

      document.getElementById("demo").innerHTML = "Hi " + user.name;

      //user.sayHi(); // this will create an alert, Hello!

      document.getElementById("click me").onclick = user.sayHi;

    </script>

* We first define an object, user; and add a couple of properties to it, namely, name and age.
* Then we create a new function that throws an alert saying “Hello!”.
* We can add this method to the object the same way we add properties to objects.
* We then use [JavaScript DOM](https://www.simplilearn.com/tutorials/javascript-tutorial/javascript-dom) to attach an event listener to an HTML button that in turn triggers the alert when clicked on.

## **JavaScript Object Accessors**

Getters and setters allow the defining of object accessors.

<!--JavaScript getter-->

    <script>

      // Create an object:

      var car = {

        model: "BMW 320d",

        color: "Navy Blue",

        fuel\_type: "Diesel",

        get fuel() {

          return this.fuel\_type;

        },

      };

      // Display data from the object using a getter:

      document.getElementById("demo").innerHTML = car.fuel;

    </script>

    <script>

      var car = {

        model: "Audi A4",

        color: "Bright Red",

        fuel\_type: "",

        set fuel(fuel) {

          this.fuel\_type = fuel;

        },

      };

      // Set an object property using a setter:

      car.fuel = "Petrol";

      // Display data from the object:

      document.getElementById("demo").innerHTML = car.fuel\_type;

    </script>

* We first create an object, car, and add a few properties to that object.
* Then, we can use a getter() function to access the properties of an object, this helps in the implementation of abstraction in a JavaScript application.
* We can also set the value of a property of an object using setter() function, this provides more control as to what values can be assigned to that particular property.