**(2) Write a blog about objects and its internal representation in Javascript**

**Objects and its internal representation in Javascript**

**Introduction:**

The Object type represents one of JavaScript's data types. It is used to store various

keyed collections and more complex entities. Objects can be created using the Object ()

the constructor or the object initializer / literal syntax.

JavaScript is designed on a simple object-based paradigm. An object is a collection of

properties and a property is an association between a name (or key) and a value.

A property’s value can be a function, in which case the property is known as a method.

**\*\*Objects and its internal representation in Javascript\*\***

Objects, in JavaScript, are its most important data type and form 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 that

they all store a single value each (depending on their types).

A JavaScript object has properties associated with it. A property of an object can be

explained as a variable attached to the object. Object properties are

the same as ordinary JavaScript variables except for object attachment.

The properties of an object define its characteristics. We can access the

properties of an object with a simple dot notation: "objectName.propertyName"

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 assigned a reference value are

given a reference or a pointer to that value. That reference or information points to the

location in memory where the object is stored. The variables don’t store the value.

Loosely speaking, 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.

These keys 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 student, it will have properties like name, age, address,

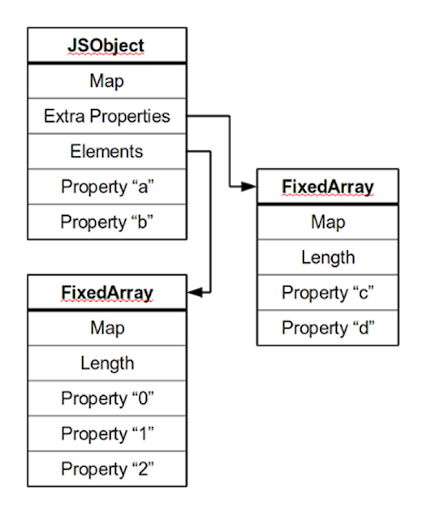
id, etc, and methods like updateAddress, updateName, etc.

**JavaScript’s internal representation of Objects:**

A simple diagram (Refer to the figure "overview of the object representation.png" ) is

probably the best way to give a quick overview of the object representation in Javascript.

Refer to figure "overview of the object representation.png."



Most objects contain all their properties in a single memory block (‘a’ and ‘b’).

All blocks of memory have a pointer to a map, which describes their structure.

Named properties that don’t fit an object are stored in an overflow array.

(‘c’ and ‘d’).

Numbered properties are stored separately, usually in a contiguous array.

The JavaScript standard allows developers to define objects flexibly,

and developing an efficient representation that works for everything is hard.

A thing is essentially a collection of properties, basically key-value pairs.

We can access properties using two different kinds of expressions:

**obj.prop**

**Obj[“prop”]**

According to the spec, property names are always strings. If we use a name that is not a

string, it is implicitly converted to a string. This may be a little surprising: using

a number as a property name also gets converted to a string.

So, a JavaScript object is a map from strings to values.