Ways to create Object

1. **Objects Literal:**

* Creating blank object first

obj = {}

obj.name = ‘name1’

obj.method = function() {

return obj.name }

* Declaring and assigining value at the same time

obj = {name: ‘name1’ , class = 9 };

1. **Factory Function: function that returns object**
2. **Constructor:**

* Blank constructor first:

obj = new Object()

obj.name = ‘name1’;

obj.class = 9

* Both at the same time:

Function Object(name, class) {

this.name = name;

this.class = class;

}

obj1 = new Obj(name1, 9)

Good thing about constructor method is that we can easlily create as no of obj as possible. But when we assess value of any object pre-define method also comes with it.

How to check Properties exists in Js

1**.Using ‘typeof’ operator:**

Syntax: if(typeof(object\_name.key != ‘undefined’)

Example:

If(typeof(nokia.memory != ‘undefined’) {

document.write(‘Available’)

} else {

document.write(‘doesn’t available);

}

**2. ‘in’ operator:**

Syntax : if(‘key’ in object\_name)

3 **hasOwnProperty():**

Syntax: if(object\_name.hasOwnProperty(‘key’));

For in Loop to get values of js object:

Syntax: for(var variable\_name in object\_name) {

Block of statement;

}

Example:

var person = { name: ‘Rahul’, age: 30 };

for(var property in person) {

document.write(person[property] + ‘<br>’)

}

**NOTE:** **Object.keys(obj\_name)** will only give you members in constructor and not prototype members BUT ‘**for in’** loop will give you all.

#to avoid printing function:

For(var property in person) {

If(typeof(person[propert] != ‘function’) {  
 document.write(person[property] + ‘<br>’)

}

}

Private properties and methods:

To convert any properties and method in constructor private change “**this**” keyword with “var/let/conts”.

To access private property from outside we have to create public **method** inside the constructor which returns that private property/method.

By calling this method we can indirectly access private data.