Object Methods:

Objects
An object in JavaScript is a collection of data in key-value pairs where each key is a string (or a Symbol) and each value can be of any data type, including other objects, functions, arrays, and primitive data types like strings, numbers, and booleans. Objects are created using curly braces {}.

Creating Objects:

Literal notation:

```
let person = { name: "John", age: 30 };
```

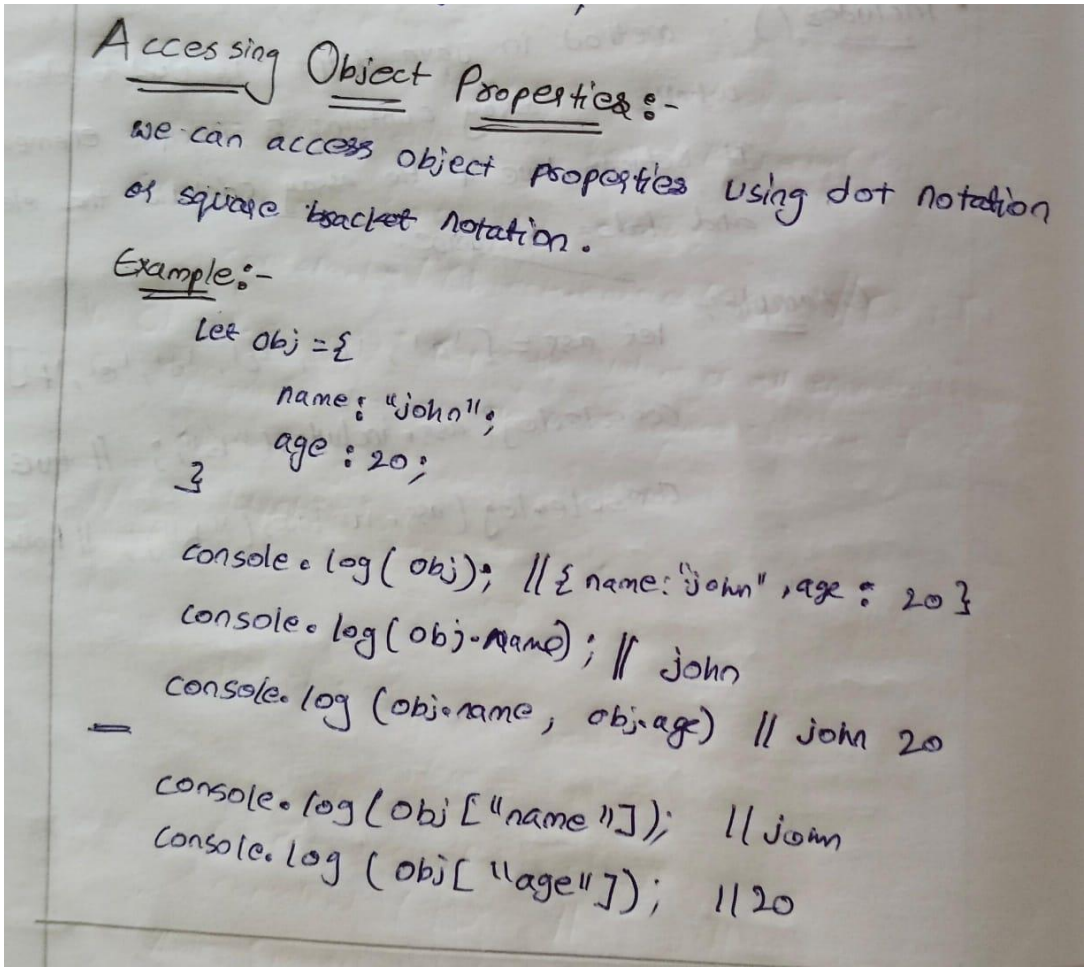
Using the Object constructor:

```
let person = new Object();
person.name = "John";
person.age = 30;
```

Accessing Object Properties:

You can access object properties using dot notation or square bracket notation:

```
console.log(person.name); // Dot notation
 console.log(person['age']); // Square bracket notation
```



Adding & Modifing properties:-

```
obj.gender = "Male";  // Adding New property
obj.age = 31;         // Modifing an existing property

console.log (obj.gender);  // male
console.log (obj.age);     // 31
```

Examples:-

```
let obj = {
        val1: {
            name: "john";
            age : 30;
        },
        val2 : 300
    }
console.log (obj["val1"]["age"]);   // 30
```

Ex:-

```
Let obj = {
        val1 : function () {
            alert ("hello world");
        },
        Val2 : 300
    }
console.log (obj["val1"]());
```

Object Methods:-

Methods are functions stored as object properties.

Example:-

```
let obj = {
        val1 : [1,2, {
                name : "john",
                val2 : [40,50,60, {
                    name : "john2",
                    val3 : [1,2,3,4,5]
                }]
            }]
    }
console.log (obj["val1"][2]["val2"][3]["val3"][4]);
```

access :

Deleating an object:-

Ex:- delete obj.name;

Methods in JS:-

Object.key()
object.values()
object.entries()
object.assign()
object.create()
Object.freeze()
Object.seal()
Object.hasown property()

1. Object.keys():- Returns an assay of a given objects property name.

Example:- var obj = {
a:1,
b:2,
c:3 };
console.log(Object.key(obj)); //["a", "b", "c"]

2. Object.values():- Returns an assay of a given object's own enumerable property values.

Ex:- var obj = { a:1, b:2, c:3};
console.log (Object.values (obj)); // [1,2,3]

3. Object.entries():- Returns an assay of a given object's own enumerable string-keyed property [key, value] pairs.

Ex:- var obj = {a:1, b:2, c:3};
console.log (Object.entries (obj)); // ["a",1],["b",2],["c",3]

4. Object.assign():- Copies the values of all enumerable own properties from one or more source objects to a target object.

syntax:- Object.assign(target, new);

Ex:- let obj = {
name = "john";
age = 25;
}

let obj1 = {
gender = "Male";
place = "hyd";
}

Object.assign(obj, obj1);
console.log(obg);

// o/p:- {name: 'john', age = 25, gender: 'male', place: 'hyd'}

5. Object.create():- creates a new object with the specified prototype object and properties.

Ex:
```
let obj = Object.create(null);
                        ↳ object or null is mandatory.
console.log(obj) // { }
obj.name = "jenny";
obj.age = 22;
console.log(obj); // { name: "jenny", age: 22 }
```

```
let obj1 = Object.create({city: "vizag"});
obj1.name = "jenny"
console.log(obj1) // { name: "jenny" }
console.log(obj1.city) // vizag.
```

6. Object.freeze():- Freezes an Object, preventing new properties from being added to it, existing properties from being removed, and values from being changed.

Ex:-
```
let obj = {
    name: "john",
    age: 25;
}
Object.freeze(obj);
obj.name = "jenny"
obj.gender = "female";
delete obj.age;
console.log(obj);

// o/p:- { name: "john", age: 25 }
```

7. Object.seal():-

Seal an object, preventing new properties from being added to it and marking all existing properties as non-configurable.

Ex:- var obj = { name: john, age: 25 };

Object.seal(obj)

obj.name = "jenny";

add:- obj.gender = "female";

del:- delete obj.age;

console.log(obj);

// O/P:- { name = "jenny", age: 25 };

- it will just modify an object ~~not add~~ Unable to add (or) delete.

8. Object.hasOwnProperty():-

Returns a boolean indicating whether the object has the specified property as its own property.

Example:-

```
let obj = {
    name: "john",
    age: 25
}
console.log(Objects.hasOwnProperty("name"));
// true.
```

How to iterate Objects:-

for in loop:-

```
var obj = {
    name: "john", age: 25
};
for(i in obj){
    console.log(i); // name age (keys)
    console.log(obj[i]); // John 25 (values)
```

Iteration using for of method:-

Object.values(obj):-

```
for (i of Objects.values(obj)){
    console.log(i);        // john  25
}
```

Objects.keys(obj):-

```
for (i of Objects.keys(obj)) {
    console.log(i);        // name  age
}
```

Objects.entries(obj):-

```
for (i of Object.entries(obj)) {
    console.log(i);        // ['name', 'john']
}                          // ['age', 25]
```

```
for ([i,j] of Objects.entries(obj)){
    console.log(i,j);  // name john
}                      // age 25
```

Api object Iteration:-

```
var obj = [
    // fake store API
]

var id = [];

for (let i in obj){
    id.push(obj[i]["id"]);
}

console.log(id);
```