Turuvashi's Notes

9th February 2023

JavaScript - Class 2

Javascript Basics-2 :-

Multiple linked variable in single entity is 'OBJECT'

let a = {}; ← empty object

Object has a key : value pairs

```
const rectangle = {
    length : 1,
    Breadth : 2;
}
```

key ↑ value ↑

- operater to access properties of object

ex:

```
let rectangle = {
    length : 2,
    breadth : 4,
    draw : function () {
        console.log("function Draw")
    }
}
```

properties ← length, breadth

function ← draw : function ()

rectangle.draw() (to access)

draw : function can also be written as draw()

Function for object Creation

- Factory function
- Construction function

} two types of creation.

Factory Function

1) Function createRectangle () {

(place your object code here)

return rectangle; ← write return at last by writing objectname.

}

OR

2) Function createRectangle () {

return rectangle = {

- - -

}

write return here

Fixed values

Calling Factory function :-

```
let name = createRectangle();
```

name → Object Stored here

createRectangle() → returns object

```
Console.log(name);
```

↑ to show the function which prints object.

Input parameters for Function.

parameter

```
function createRectangle (length, breadth) {
    return rectangle = {
        length,
        breadth,
        draw() {
            alg("draw");
        }
    }
}

let rectangleObj1 = createRectangle(5, 4)
```

length

breadth.

value can change here

(2) Constructor Function

↳ we follow pascal Notation. ✓

Camel → numberOfStudents

✓ pascal → FirstWordOfWordAlwaysCapital

```
function Rectangle() {
    this.length = 2;
    this.breadth = 3;
    this.draw = function() {
        alg("draw"
    }
}
```

(this → current object)

* **Constructer function**
  - defines the properties of methods
  - does not returns

new → keyword that returns empty object.

```
let rectangleObj = new Rectangle();
```

(we can also give parameters here to change value.)

```
function Rectangle(len, bre) {
    this.length = len;
    this.breadth = bre;
}
}
```

```
let rectangleObj = newRectangle(5,6)
```

a.draw()

object — method

```
draw()
{
    cl(this.length)
}
```

this is 'a' so a length will be printed.

Dynamic Nature of object

↳ we can add, or remove property of object.

To add ~~let a = 5;~~

```
rectangleObj.color = "yellow";
```

This will add color property in rectangle object.

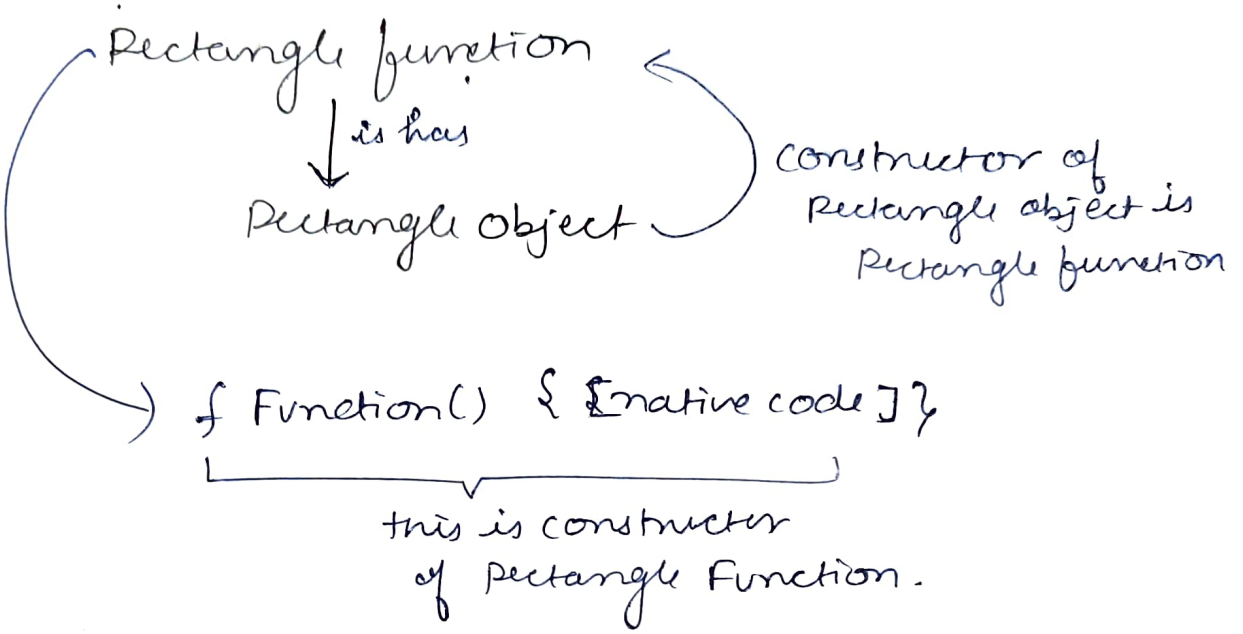
To remove

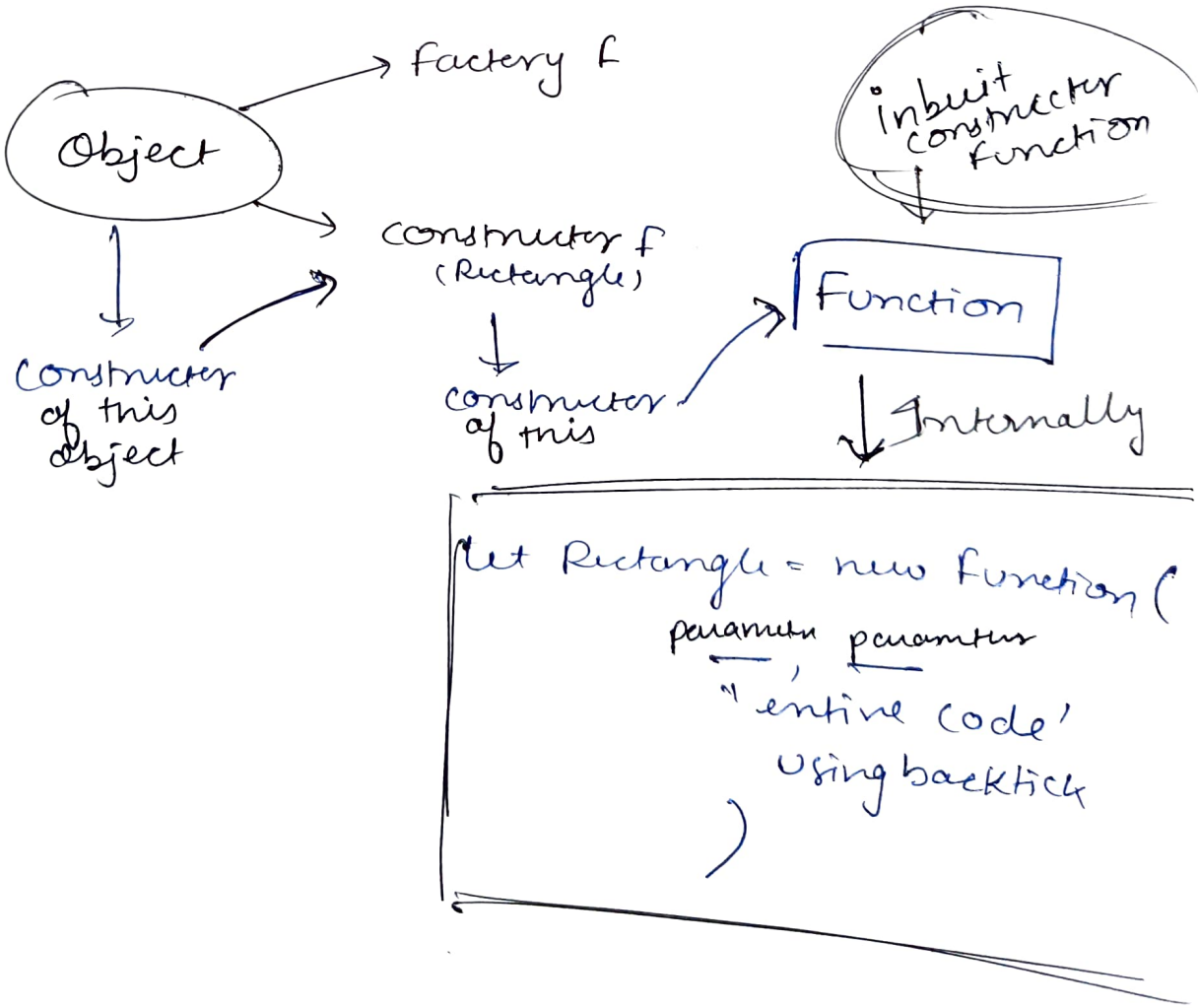
```
delete rectangleObj.color;
```

CONSTRUCTOR

Function is also an object

all object has a constructer





✓ Functions are Objects

as it have properties & entity ✓

Difference primitive type & Reference type:

primitive

let a = 10 → a [10]
let b = a → b [10]

a++;

→ print(a) → 11
→ print(b) → 10

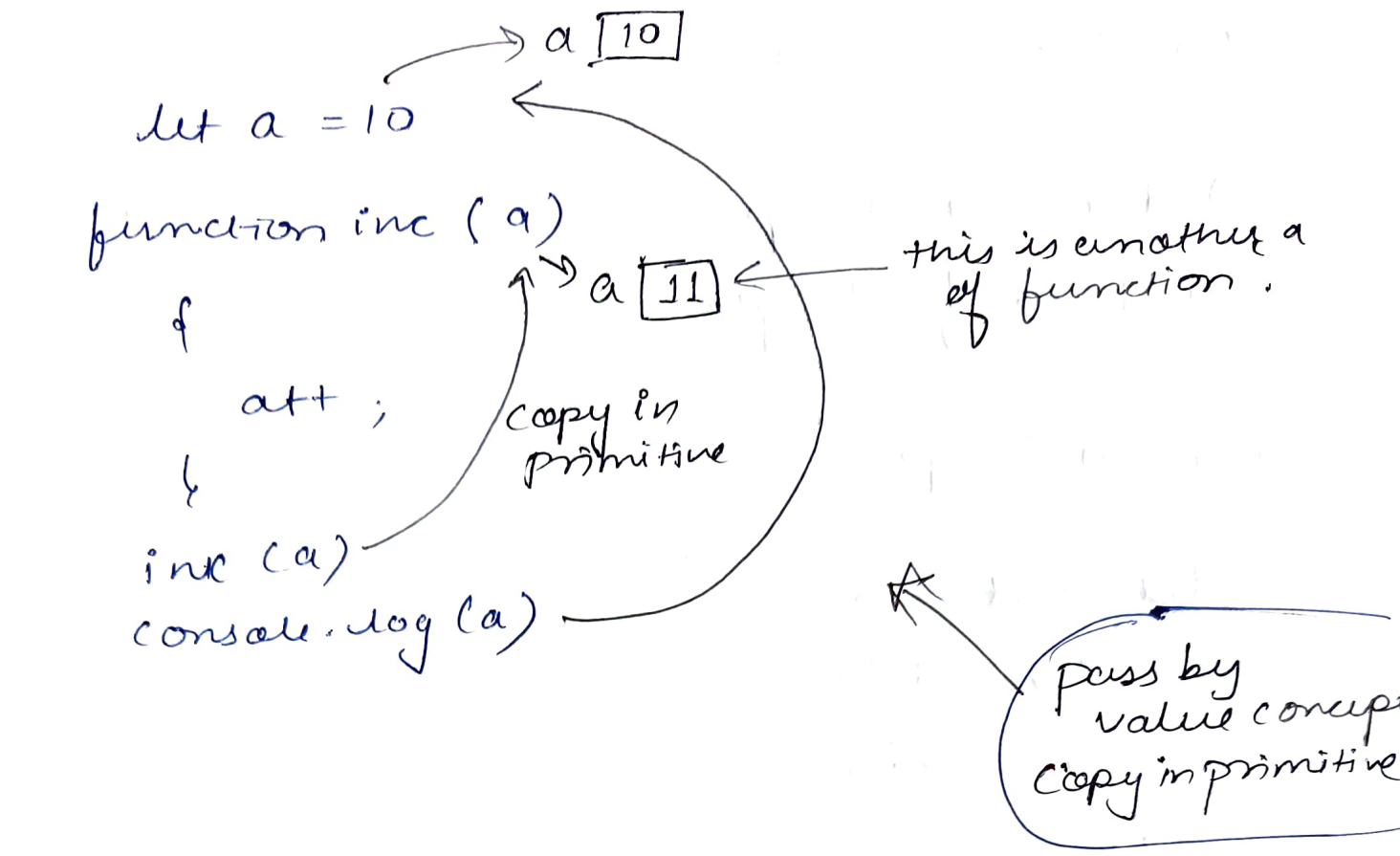
here copy is created ✓

Reference

let a = { value : 10 };
let b = a;
a.value ++
console.log(a); → 11
console.log(b); → 11

In objects, address is passed so, both will represent the memory location.

NOTE :- primitives are copied by their value
References are copied by their address.



```
let a = {value: 10};
  inc (a); // fuctioncall
  console.log (a)
```

11

not copied here

```
Function inc (a)
  {
    a.value ++;
  }
```

pass by refrence in same address

Iterating through Objects

1) For - in loop
2) For - of loop.

For- in loop

```
let rect = {
    length: 2;
    breadth: 4,
};
for (let key in rect) {
    console.log( key , rectangle [key])
}
```

key → to access key name

rectangle [key] → to access value of key

For of

→ doesn't work in object

→ iterables
  → Arrays
  → maps.

for of in object :- (HACK)

```
for (let key of object.keys(rect)) {
    console.log (key)
}
```

object.keys(rect) → gives key name

use Object.entries(rect) for values too

We can use if else to know the property is present or not-

```
if ('length' in rect) {
    console.log("yes")
}
else {
    console.log("No")
}
```

Object Cloning (Same to same one more)

- iteration
- Assign
- Spread

} rules.

1) iteration.

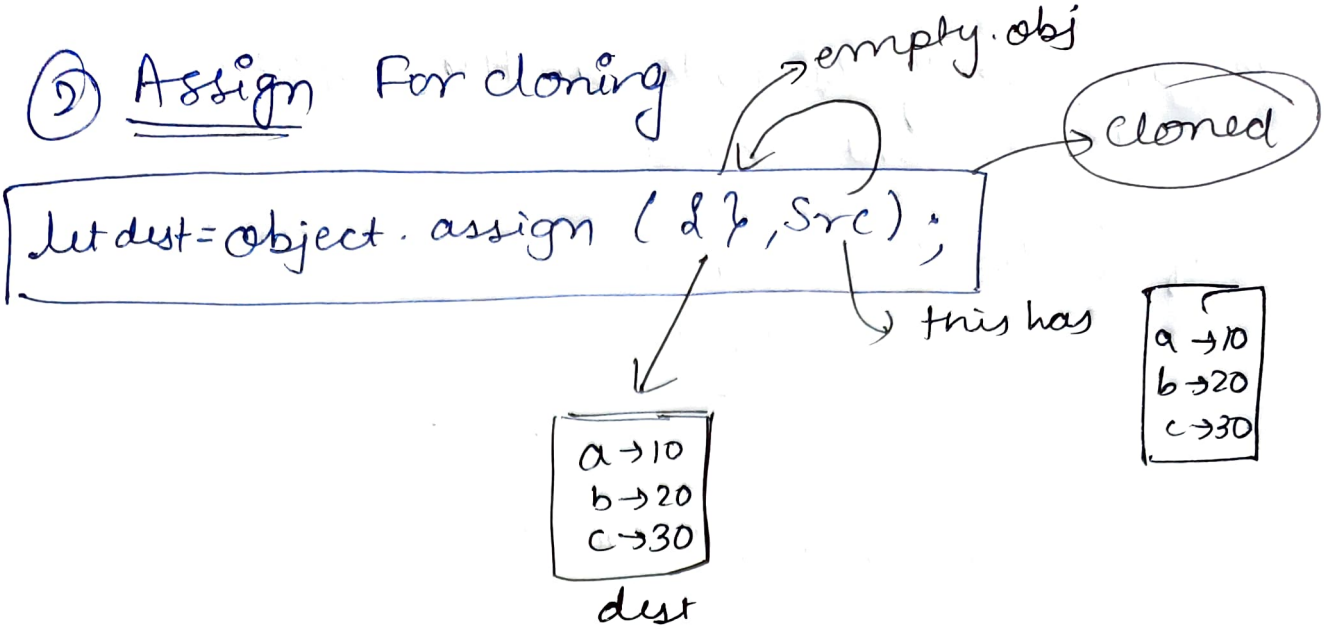
```
for (let key in Rectangle)
{
  console.log(key, Rectangle[key]);
}

let obj2 = { };
```

↑ we will copy all key & value of Rectangle in obj2 one by one. At first it will be empty.

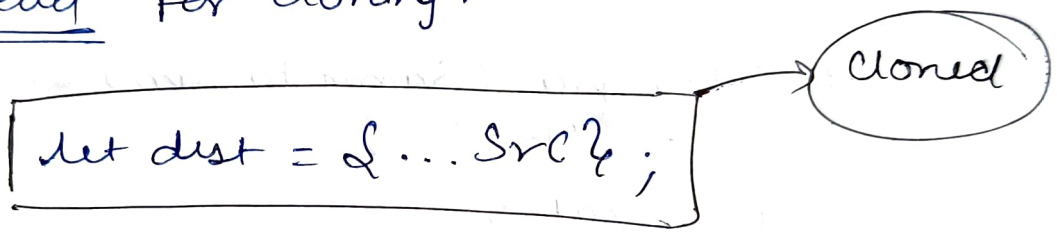
Cloned

```
for (let key in src)
{ dest[key] = src[key];
}
```

② Assign For cloning



③ Spread For cloning.



Garbage Collection :-

↳ Find all variables / constants which are not in use and automatically deallocates their value

↓

done by

(Garbage collector)

↑

tool in JS

✓ we have no control over Garbage Collector, runs in Background itself.