|| Hare Krishna ||

Miracle
Pg No :
Date: / /20

Namaste - Javascript (Akshay Saini)

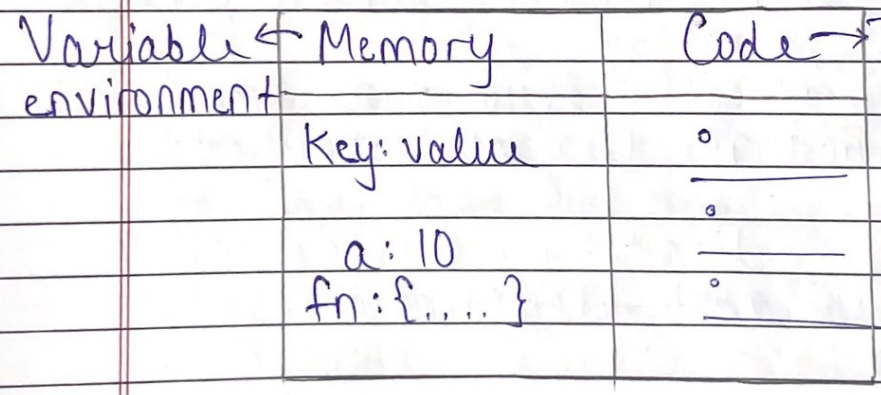
Episode - 1

How Javascript Works & Execution Context

→ Everything in Javascript happens inside an "Execution Context".

→ Execution Context:

It is a big box or a container in which whole Javascript code is executed..



Two components:

(1) Memory Component (Variable Env)

→ This is the place where all variables and functions are stored as key-value pairs

e.g. a=2 → a:2 (Key : value)

(2) Code Component (Thread of Execution)

→ This is the place where code is executed one line at a time.

✱ Javascript is a synchronous single-threaded language?

→ Single Threaded : JS can execute only one command at a time.

→ Synchronous Single Threaded : Means that Javascript can only execute one command at a time and in a specific order.

Can only go on the next line once the current line has been finished executing.

Episode-2

✱ What happens when you run Javascript code?

→ When we run a code, a global execution context is created.

Pg No :
Date: / /20

→ Code:

```
var n = 2

function square (num) {
    var ans = num * num
    return ans
}

var square2 = square(n)
var square4 = square(4)
```

→ Execution Context is created in 2 phases.

1) MEMORY CREATION PHASE

→ Javascript will allocate memory to all the variables and functions.

→ Variables: undefined
functions: whole code is stored.

Memory	Code
n: undefined square: {...} square2: undefined square4: undefined	

2) CODE EXECUTION PHASE

→ Once again runs through the whole JS program line by line and it executes the code now.

→ 2 value of n is assigned in memory.
n: 2

→ function square : Nothing to do, now to
var square2 = square(n)

→ Now in var square2 = square(n)
func. invocation
When a new function is invoked, a new execution context is created.

Memory	Code
n: 2	Memory | Code → for square(n)

And again the cycle repeats.

Whenever a function is invoked, new execution context is created in 2 phases and so on.

num :- parameter M
n :- argument

C

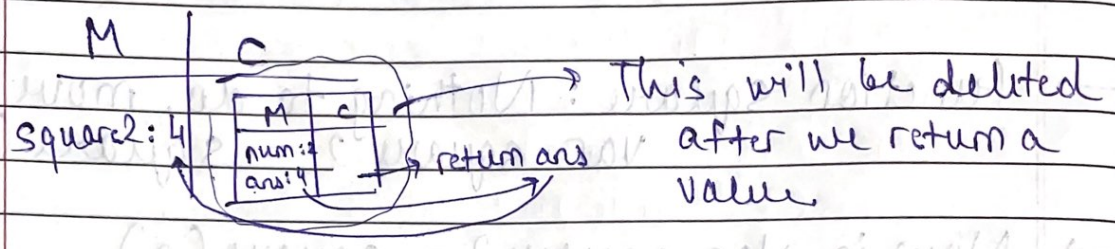
M	C
num: undefined	num * num
ans: undefined ←	

In phase 2, value of n = 2 is passed to num. num * num value is stored in ans variable ans : val

Miracle
Pg No :
Date: / /20

return keyword states that return the control of the program, to the place where this func. is invoked.

i.e. var square2 = square(n) ← it will return here



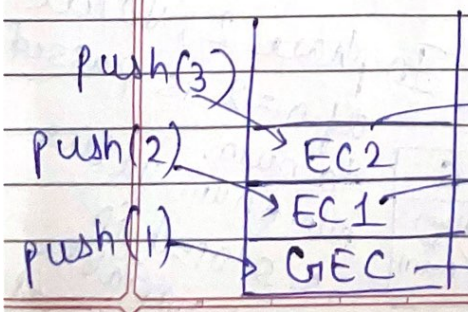
As soon as we return the value, whole instance of the execution context (func. ne will be deleted.

→ Same for var square4 = square(4).

New execution context will be created and so on.

→ Once JS is done with all its work, now the program is finished, the global execution context also deletes, it goes off.

✱ Call Stack



First, global exe. context is pushed, when new EC is created. It is pushed. After its work is completed it is poped out, then EC2 is pushed and so on.

→ Call Stack maintains the order of execution of execution contexts.

→ It handles everything to manage this execution context creation, deletion, and the control.

→ Other names of Call Stack

1. Execution Context Stack
2. Program Stack
3. Control "
4. Runtime "
5. Machine "