# Execution content

- In Javascript everything happens inside the execution content.

- It is like a sealed-off container where javascript runs inside it.

Container

↙ ↘

Memory Component (variable environment)
↓
Has all the variables and functions in key value pairs.

Code Component (Thread of execution)
↓
The place where code is executed one line at a time.

| Memory component | code component |
|---|---|
| key : value | o ____ |
| a : 10 | o ____ |
| function : {--} | o ____ |

- when a javascript program is run, a Global Execution content (GEC) is created, even if we don't write a single line of JS program.

JS execution created in 2 ways
↙ ↘

creation phase
or
Memory creation phase

code execution phase

eg
```
var n = 2;
function square (num) {
        var ans = num * num;
        return ans;
}
var square2 = square (n);
var square4 = square (4);
```

# Creation phase

↳ All the variables get the memory in memory space with a special value "undefined".

↳ They are partially hoisted

↳ All the functions (classical) get the memory in memory space with its exact value in it.

↳ They are completely hoisted.

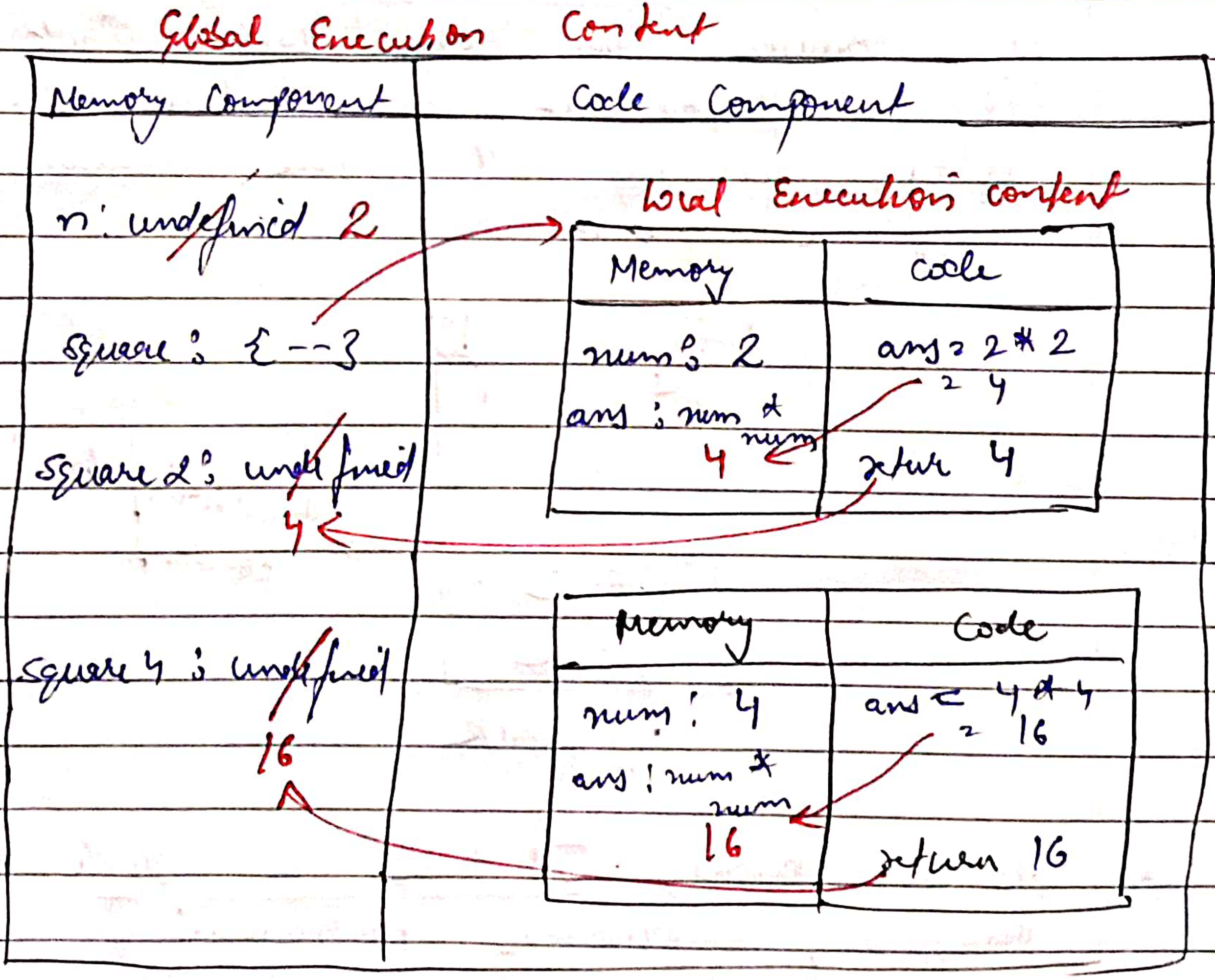
NOTE → We learn about hoisting in later upcoming classes.

| Memory component | Code Component |
|---|---|
| n : undefined | |
| square : {-- } | |
| square2 : undefined | |
| square 4 : undefined | |

# Execution phase

↳ It starts going through the program line by line and execute it.

Global Execution Context



| Memory Component | Code Component |
|---|---|
| n: ~~undefined~~ 2 | |
| square : {--} | |
| square2 : ~~undefined~~ 4 | |
| square 4 : ~~undefined~~ 16 | |

Local Execution context

| Memory | code |
|---|---|
| num : 2 | ans = 2 * 2 = 4 |
| ans : num * num 4 | return 4 |

| Memory | Code |
|---|---|
| num : 4 | ans = 4 * 4 = 16 |
| ans : num * num 16 | return 16 |

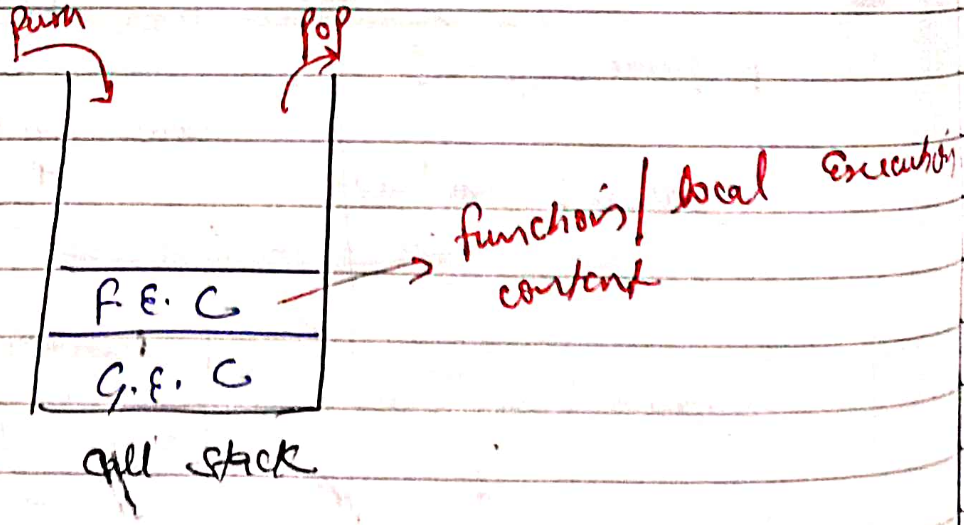
Call stack

↳ A mechanism to keep track of its place in the script.

Some other names of call stack is

↳ Program stack
↳ Control stack
↳ Runtime stack.
↳ Machine stack
↳ Execution context stack

- So whenever we run JS file, before executing a single line of code, a Global Execution context (G.E.C) is created in the call stack. and it is popped out from the stack after executing all the commands of script.



- As we know, JS is a single ~~thr~~ threaded, synchronous programming language because it has only one call stack, so JS engine execute the command one by one at a time.

~~So / if / we / block~~ So how asynchronous operations are performed in JS (because these operation take sometime to generate response).

↳ So how we handle asynchronous operation (learn this later).

- So we get 2 things from the Global Execution Content

↙ Global object (Browser → window object)

↘ 'this' keyword in JS

console

```
> this
→ window. object
> this === window
→ true
```

→ In case of Browser.