**JavaScript Execution Context – How JS Works Behind the Scenes**

**Everyting happens in js happens inside an execution context**

|  |  |
| --- | --- |
| **Memory** | **code** |
| **Key : value** | **Here code executed by line by line** |
| **Fn {………}** | **----** |
|  | **----** |

**So in memory component all variables and functions are stored in key value pair**

Memory component also known as variable environment

Code component also known as thread of execution

Javascript is synchronous , single-threaded language

This means javascript can only execute one command at a time in specific order

Var n = 2;

Function square (num){

Var ans = num \* num;

Return ans;

}

Var square2 = square(n)

Var square4 = square(4)

|  |  |
| --- | --- |
| memory | code |
| n = undefined later 2 in second phase |  |
| Square:{ ….} |  |
| Square2:undefined / 4 after code executed 2 \*2 and store in ans |  |
| Square4:undefined / after store ans 16 then allocated memory here 16 |  |
|  |  |

First phase allocated memory to variable and function and usedefined

In second phase key or data provide allocated to identifiers

First phase javascript goes through all the program line by line and allocated memory to ALL var and functions

Now in second phase that is code execution phase all 2 value assign to identifiner that is n now 2 allocated to n in second phase

Later after function square2 and square4 function invoked and brand new execution context created

|  |  |
| --- | --- |
| memory | code |
| Num: undefined first phase / 2 | num \* num 2\* 2 |
|  |  |
| ans : undefined first phase /4 | Num \* num |
|  |  |

Call stack is for managing execution context

After all execution call stack gets empty that is whole code inside in java script engine executed

Call stack maintain the order of execution of execution context.