JS Execution Context

{ } → Global EC
↳ this
↳ Global Execution context
↳ Function " "
↳ Eval " "

{ } → Memory Creation Phase &
→ Execution Phase.

```
let val1 = 10
let val2 = 5
function addNum(num, num2){
    let total = num1 + num2
    return total
}
let result1 = addNum(val1, val2)
let result2 = addNum(10, 2)
```

(1) Global Execution
↓ this

(2) Memory Phase

step 1 val1 → undefined
step 2 val2 → "
3 addNum → definition
4 result1 → undefined
5 " 2 → undefined

(3) Execution Phase
val1 ← 10
val2 ← 5
addNum →
result1 = 15

New EC
new variable environment
+
execution thread

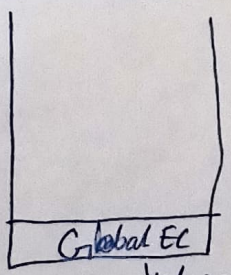
return
1
Memory Phase
val1 → undefined
val2 → "
total

2 EC
num1 → 10
num2 → 5
total → 15

delete

addNum → nve + et
repeat
result2 = 12

Call Stack



↳always there

last in first out

when we call a function, it enters the call stack execution ctx executes, and leaves.
If we call a function inside function, The function that'll enter last will get out first.

Go to Chrome snippets & try