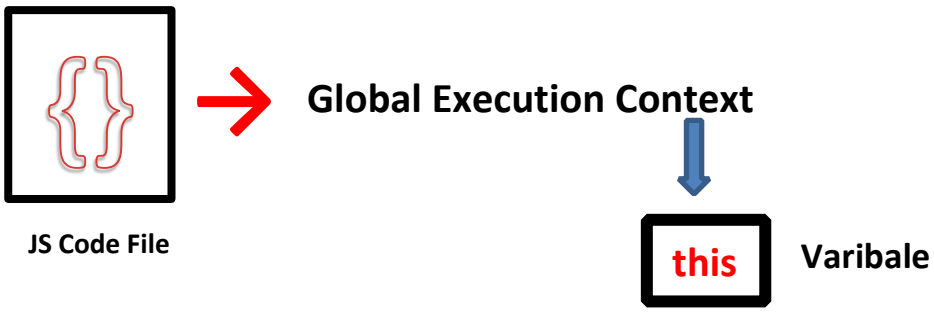
How does javascript works behind the scene

Javascript Execution context

- Javascript Execution Context means how JS runs the file which we have made
- JS runs it in two phases



Browser, Node, Bun, Deno etc all have different Execution Contexts.

Like Value of “this” in Browser is → Window

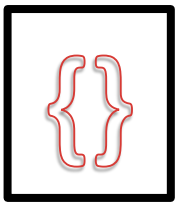
There are 3 Execution Context

1. Global Execution Context
2. Functional Executon Context
3. Eval Execution Context

How the JS Code Executes

JS Code executes in 2-phases

- Memory Creation Phase
- Execution Phase



Memory Creation Phase

During this phase, JavaScript allocates memory for variables and functions. Variable declarations are scanned, and they are set to undefined initially. Functions are stored in memory with their entire code.

Execution Phase

JavaScript operates as a synchronous single-threaded language, meaning it processes one command at a time and in a specific order. This is where the actual code is executed line by line. Calculations are performed, and values are assigned to variables. When a function is called, a new local execution context is created with its own memory and two phases, similar to the global context. After a function finishes executing, its result is stored in memory, and the local execution context is removed.

Steps fro Execution of Below Code

```
let val1 = 10;
let val2 = 5;
function addNum(num1, num2){
let total = numl +num2 3
return total
}
let resultl = addNum(val1, val2);
let result2 = addNun(10, 2);
```

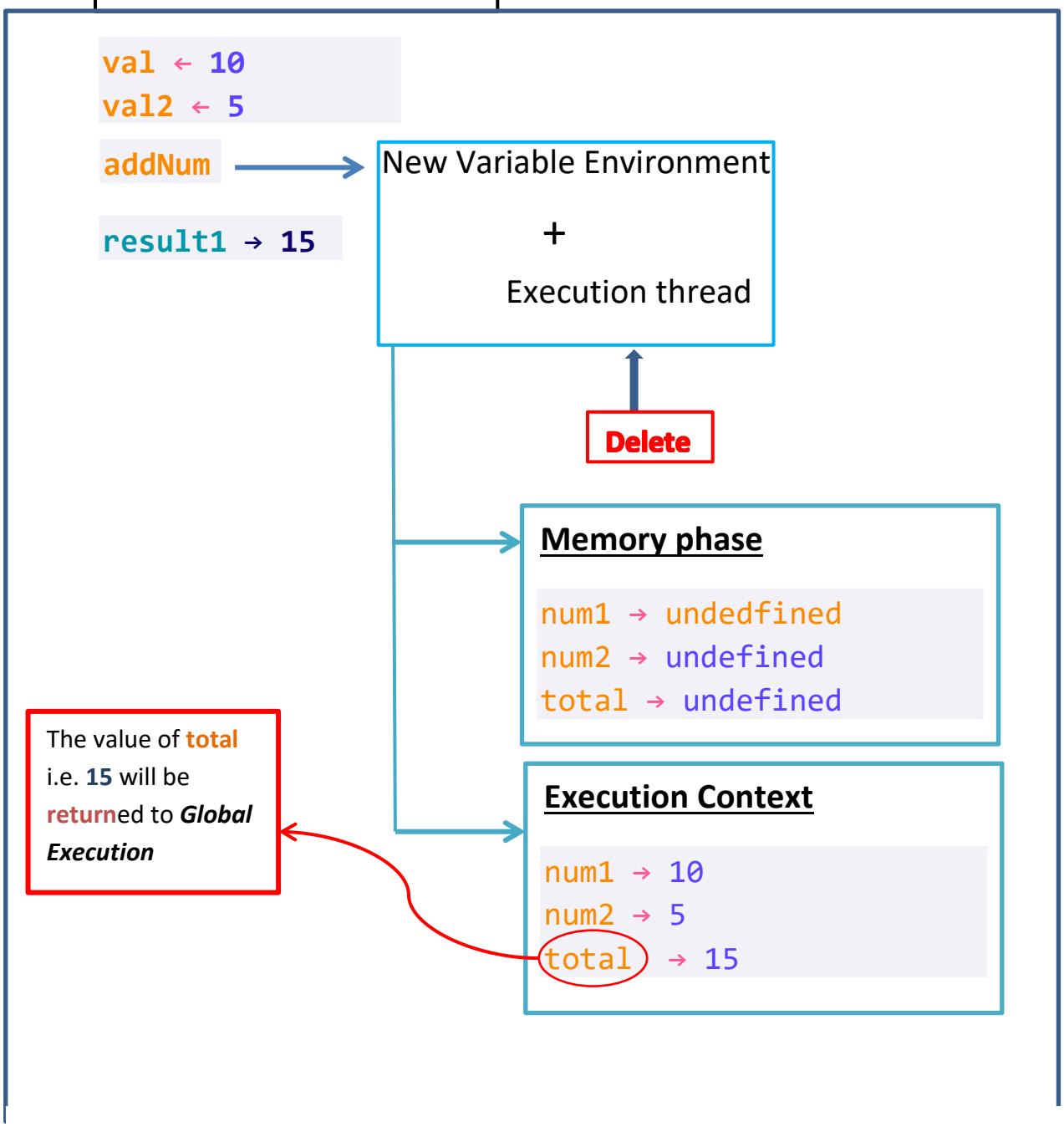
STEP-1 :- Global Execution Phase or Global Environment

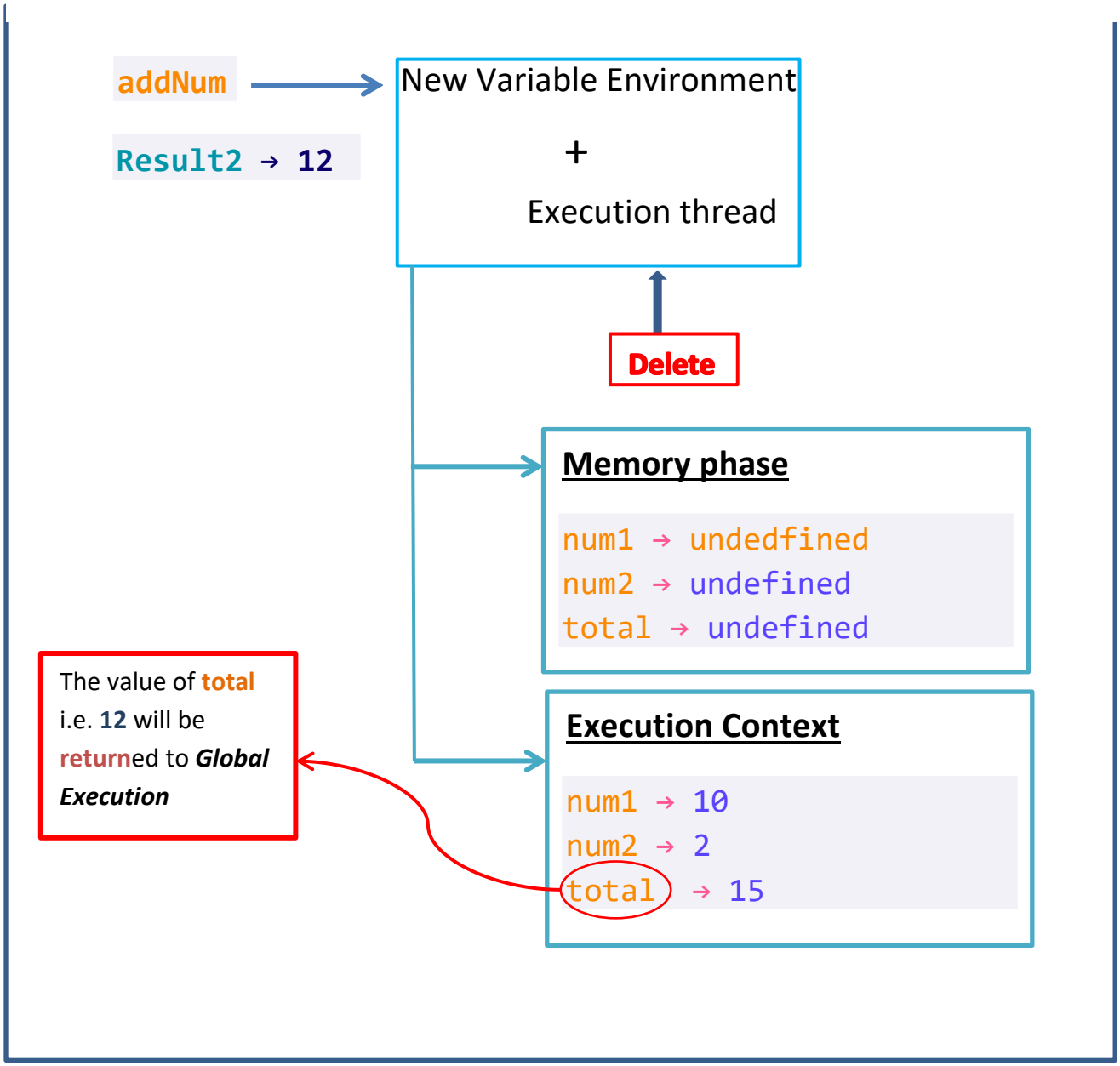
At first any JS code either small or big which we wish to run always starts with Global Execution which is allocated in “this” variable

Step-2:- Memory phase

```
val → undefined
val2 → undefined
addNum → {...} == function defination
result1 → undefined
result2 → undefined
```

Step-3:- Execution Phase




addNum
Result2 → 12
New Variable Environment
+
Execution thread
Delete
Memory phase
num1 → undedfined
num2 → undefined
total → undefined
Execution Context
num1 → 10
num2 → 2
total → 15
The value of total i.e. 12 will be returned to Global Execution