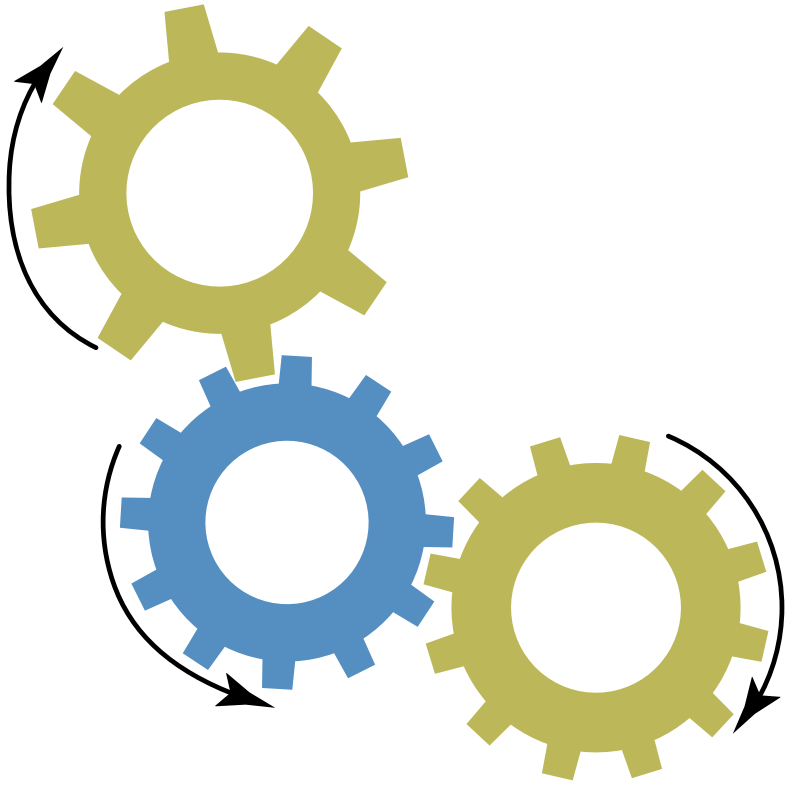
@Mukhi Talibhusain

Do you know how JavaScript code is executed ?



Everything in JavaScript happens inside in "Execution Context"

Whenever a JavaScript program is run an execution context is created.

So What is Execution Context ?

In JavaScript, an execution context is an internal JavaScript engine construct that holds information about the environment in which code is executed. It includes variables, function declarations, the value of the 'this' keyword, and a reference to the outer environment, among other things. The execution context plays a crucial role in managing the execution of JavaScript code

Execution Context is created in two phase

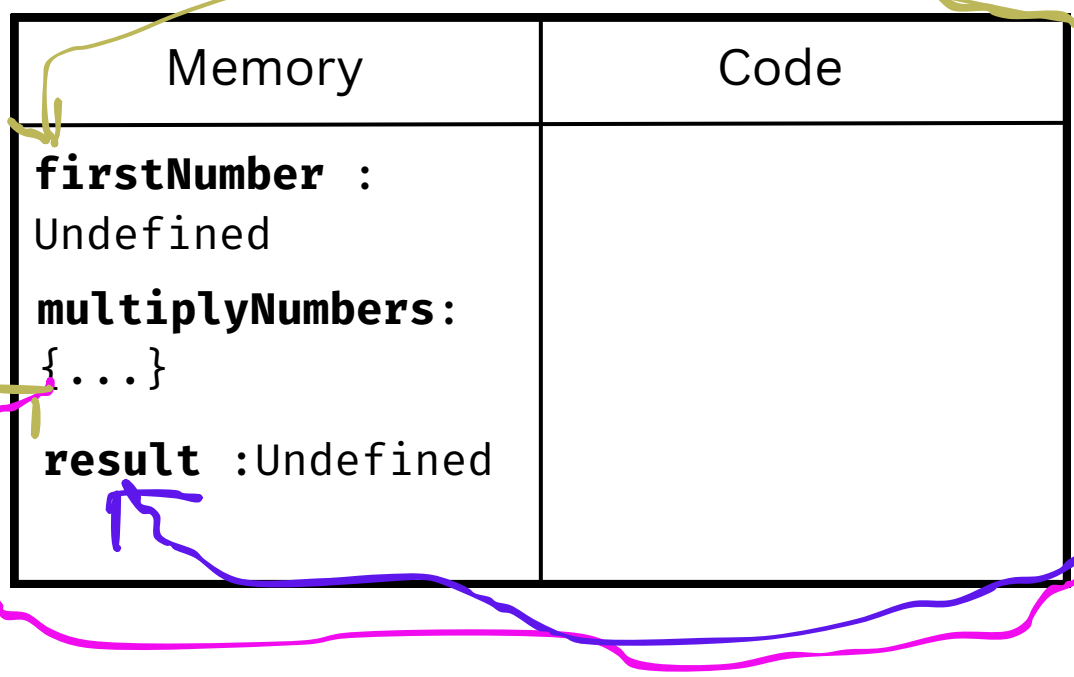
1. **Memory Creation**
2. **Code Execution Phase**

```
var firstNumber=15;

function multiplyNumbers(num){
    return num * 10;
}
var result = multiplyNumbers(7);
```

when we run the above code, a global execution context (GEC) is created.

Memory	Code

(Global Execution Context)



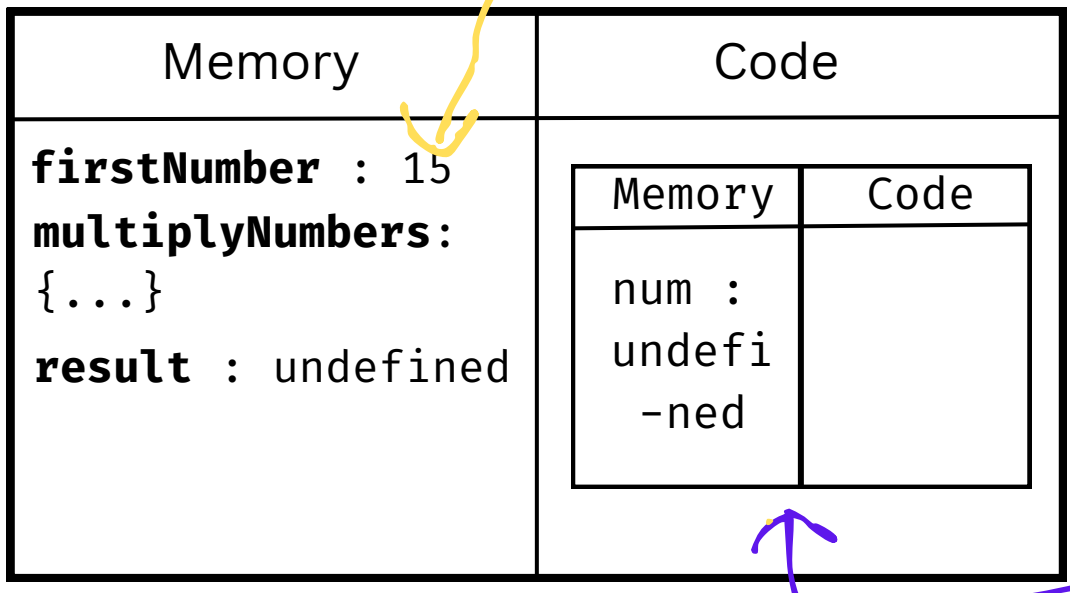

```
var firstNumber=15;

function multiplyNumbers(num){
    return num * 10;
}
var result = multiplyNumbers(7);
```

Memory Creation :

In this phase, javascript allocates the memory to all the variables and functions present in the program. The variables are stored with the value undefined and the function is stored with all the code present in that particular function.

For the above code, variable **firstNumber** is stored with the value undefined and the function **multiplyNumbers** is stored with value between the {...} curly braces. The **result** is also a variable so it is stored with the value undefined.

You can see above Design.



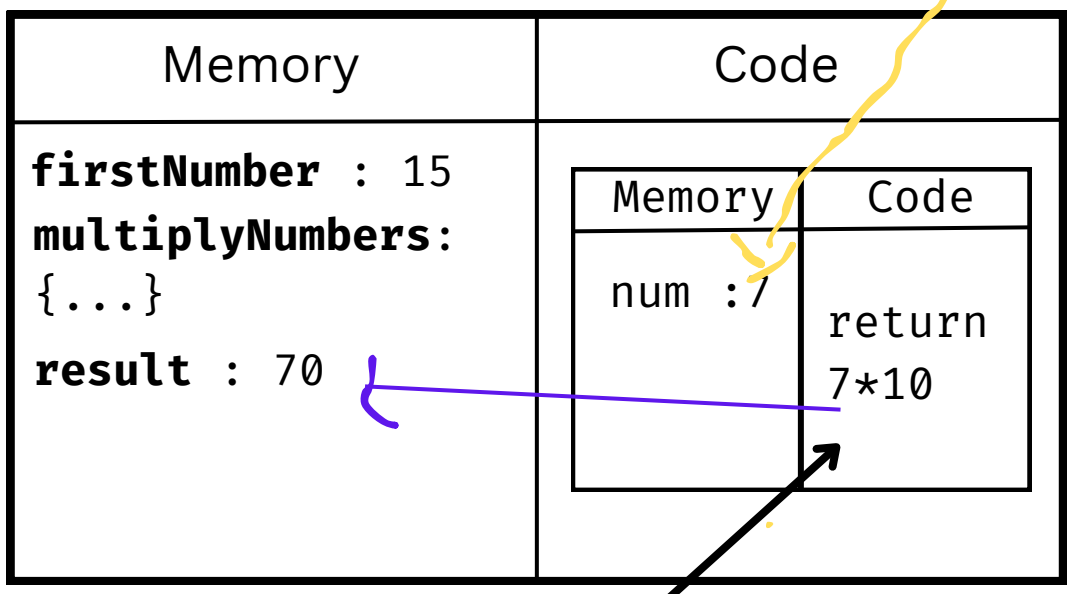

```
var firstNumber=15;

function multiplyNumbers(num){
   return num * 10;
}
var result = multiplyNumbers(7);
```

Code Execution Phase :

In this phase, the main execution takes place and the **javascript code runs through the code line by line.**

So for the above code, variable **firstNumber** value is changed from undefined to 15. Then it moves to the next line as there is nothing to execute it moves to line result and multiply- -Numbers function invocation takes place. When a new function is invoked a new execution context is created within the GEC. and repeat the whole process in new execution context

You can see above Design.

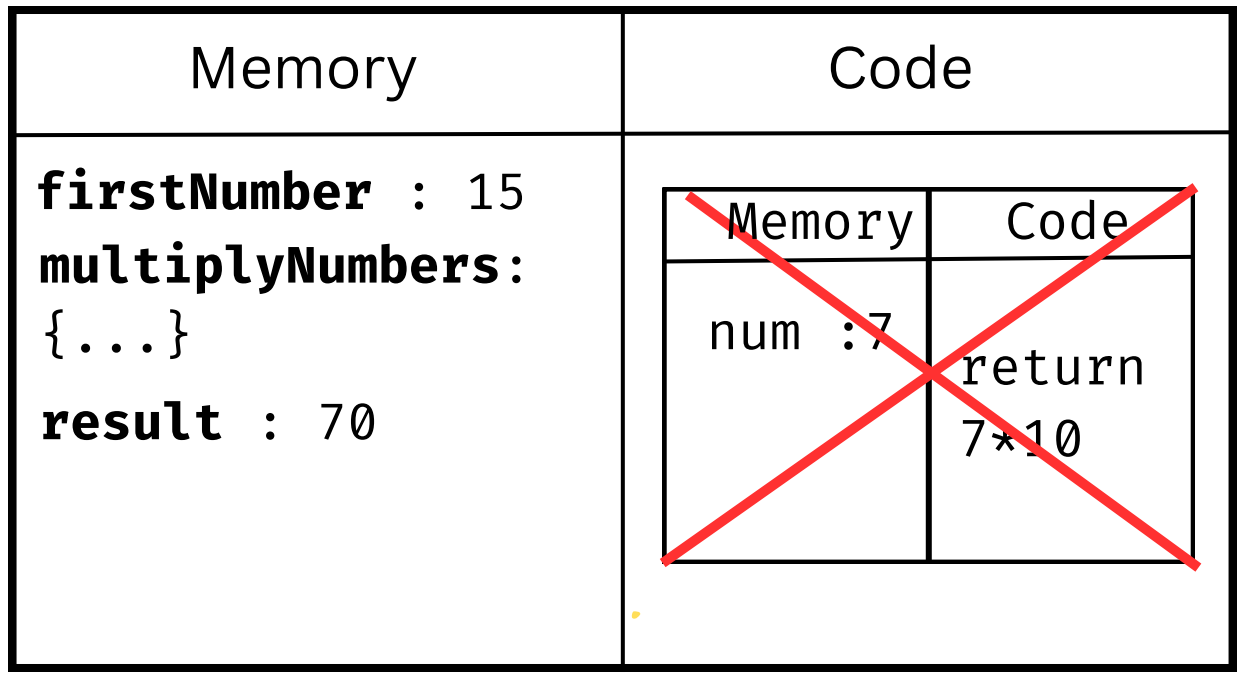



```
var firstNumber=15;

function multiplyNumbers(num){
   return num * 10;
}
var result = multiplyNumbers(7);
```

Code Execution Phase :

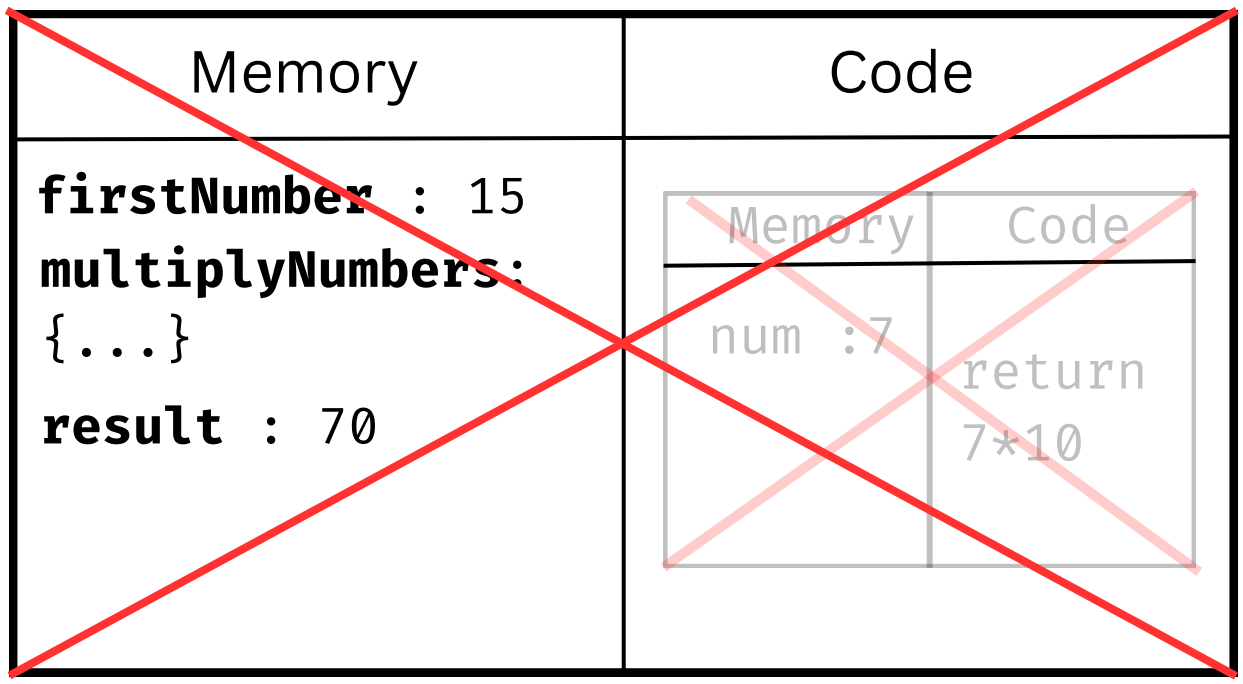
So for the above code, variable num value is changed from undefined to 7. Then code is executed in code section and return 70 and result value changed undefined to 70. after return value 70 execution context is deleted.

You can see above and below Design.



So JavaScript is done the All the work because program is finished. Then Global context is also Deleted.

You can see the below Design.



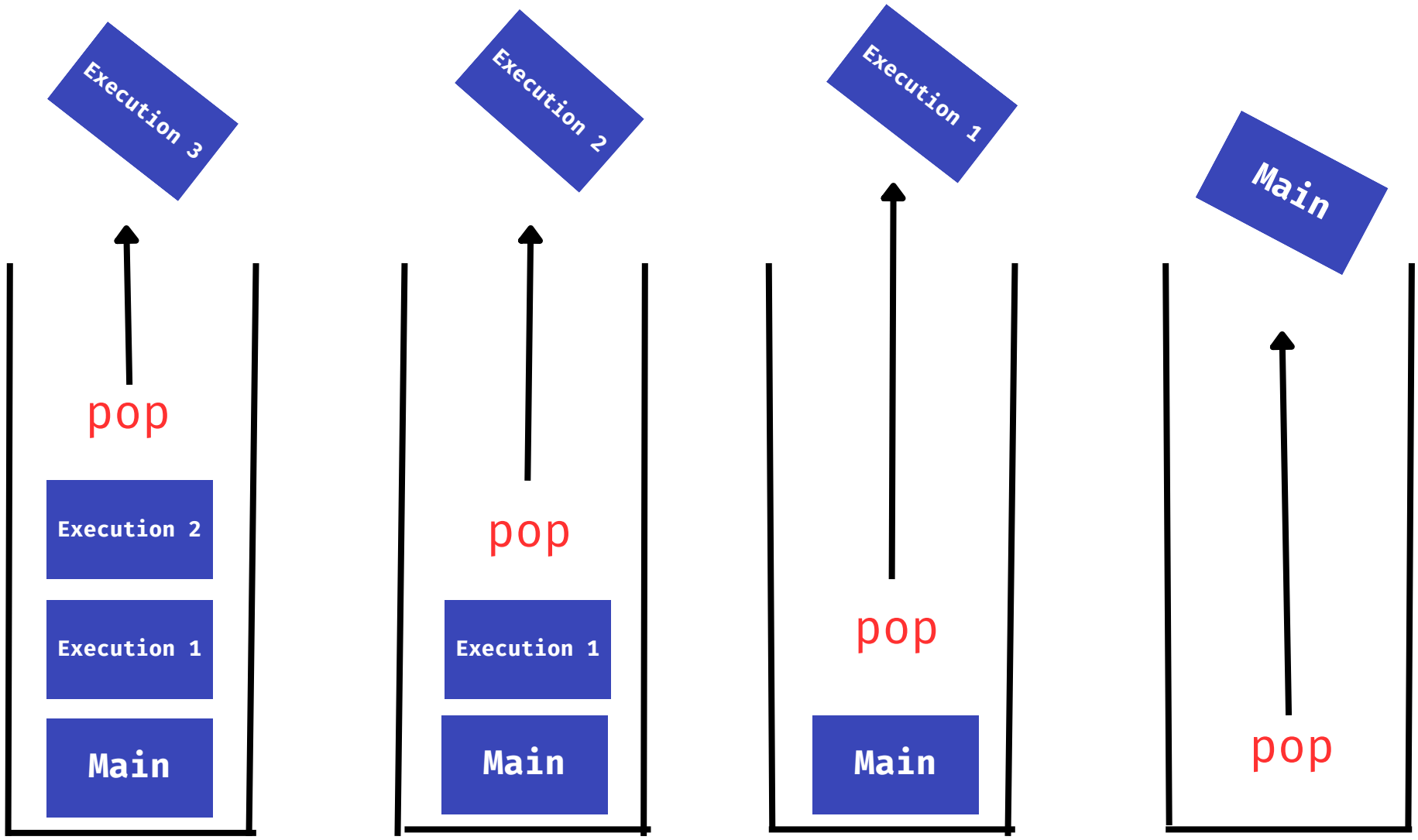
So that is all about how JavaScript code is executed.

But whole things are how JavaScript engine handle ?

A Call stack is also maintained by javascript. Call stack maintains the "Order of execution of execution contexts". It works similarly as a stack whenever a new function invoked its execution context is pushed into the call stack.

So Call Stack is Work ?

First of All, **Stack** is a linear data structure that follows the principle of Last In First Out (LIFO). This means the last element inserted inside the stack is removed first.



So You can see above how execution completed and deleted from call stack, Now i hope you can understand How whole JavaScript is executed.

FOLLOW ME & LEARN
WEB DEVELOPMENT

@Mukhi Talibhusain

Thanks For Your Attention

Mukhi Talibhusain

LIKE AND SAVE IT FOR LATER
BY LIKING IT.