the Master Course

{CODENATION}

How JS works. (Intermediate JavaScript)

{CODENATION}

{CODENATION}

Learning Objectives

To develop a **mental model** of what JavaScript is doing when it runs.

To improve **technical communication** and understanding.

Why bother understanding how JS works under the hood?

- It will help you visualise what is ACTUALLY happening when you run JS code.
- It will give you a professional level of technical understanding.
- You may get asked about it in interviews.

{CODENATION}

The most important things for software engineers.

1. Ability to solve problems (break problems down and form solutions).
2. Technical communication (being able to talk about your code/problems/solutions).
3. Your approach to development (code structure, patience, research skills, passion, 'bigger-picture' thinking etc)
4. Non-technical communication (being understanding/empathetic, being a good team member).
5. Programming experience/languages you know.

Moving forward we will begin to level up our technical communication.

That means we will be asking you to explain your code much more, and ensure you use the correct terminology.

{CODENATION}

How JS runs.

{CODENATION}

A JavaScript engine, is a computer program which executes JavaScript code.

A web browser is also a computer program, which has a JavaScript engine built into it.

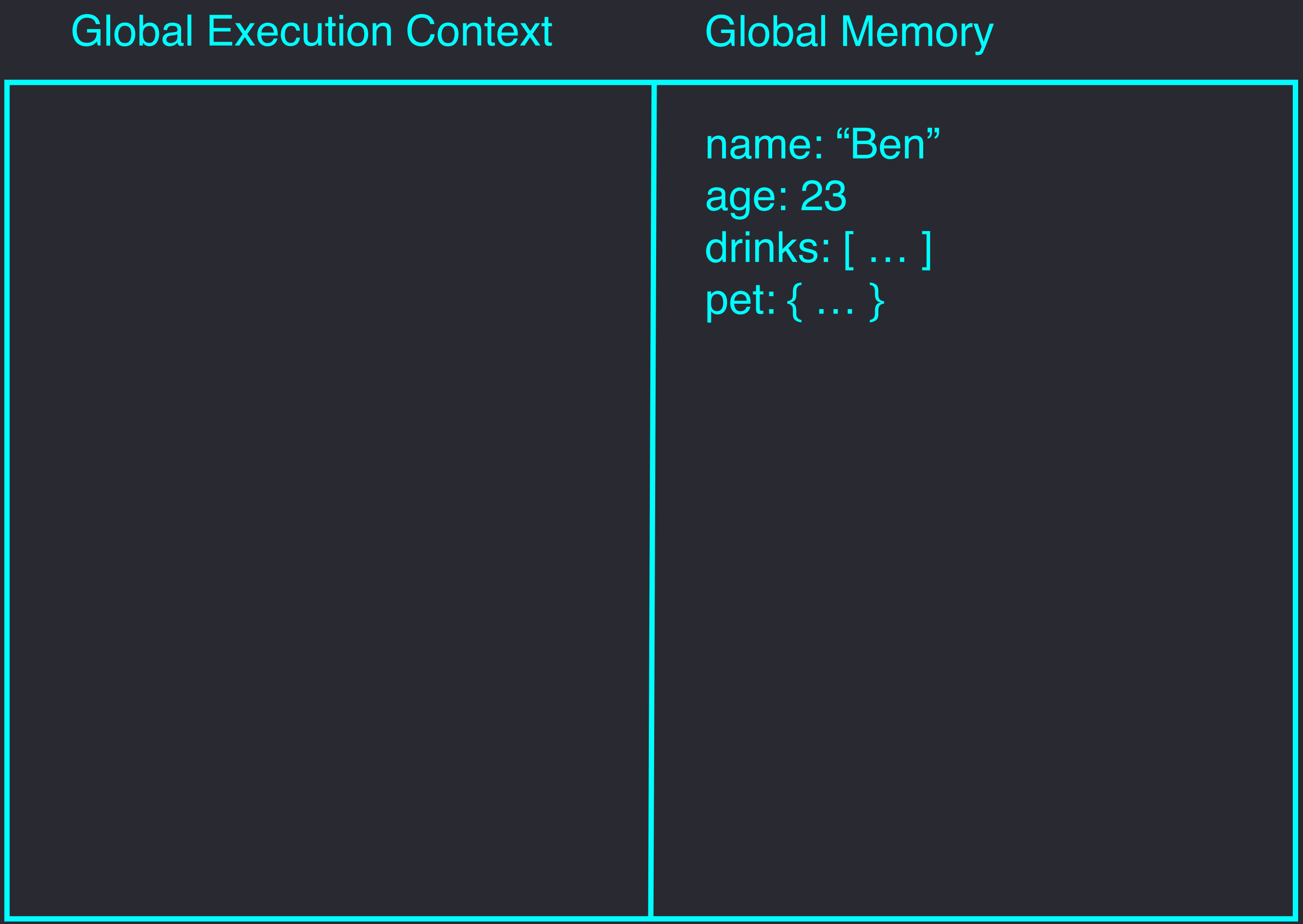
Understanding how (and when) the JavaScript engine runs your code, will help you as a software developer.

{CODENATION}

To start this story, we first need to understand three fundamental parts of the JavaScript engine.

1. Execution Context
2. Memory/variable environment
3. Thread of execution.

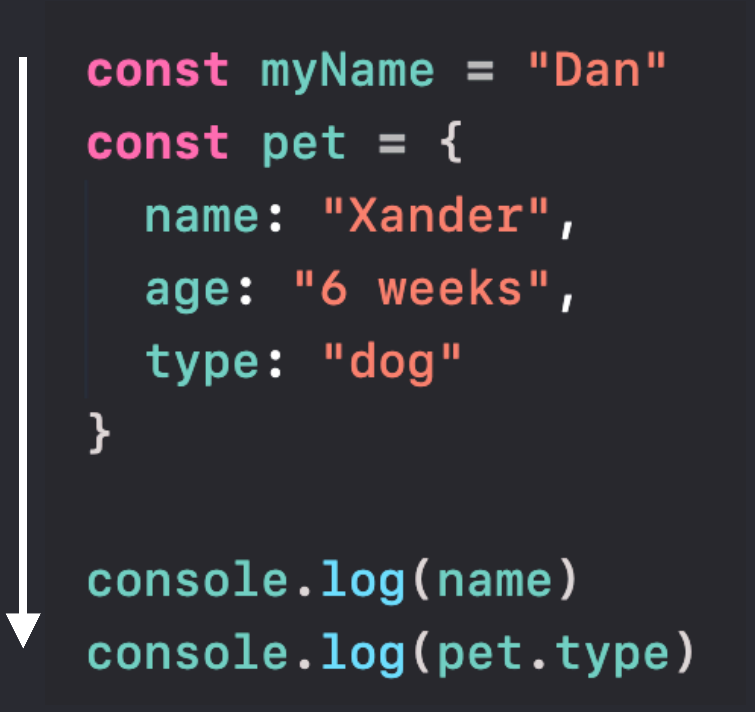
```
const name = "Ben"
const age = 22
const drinks = ["coffee", "tea", "coke"]
const pet = {
  name: "Xander",
  age: "6 weeks",
  type: "dog"
}
```


Global Execution Context

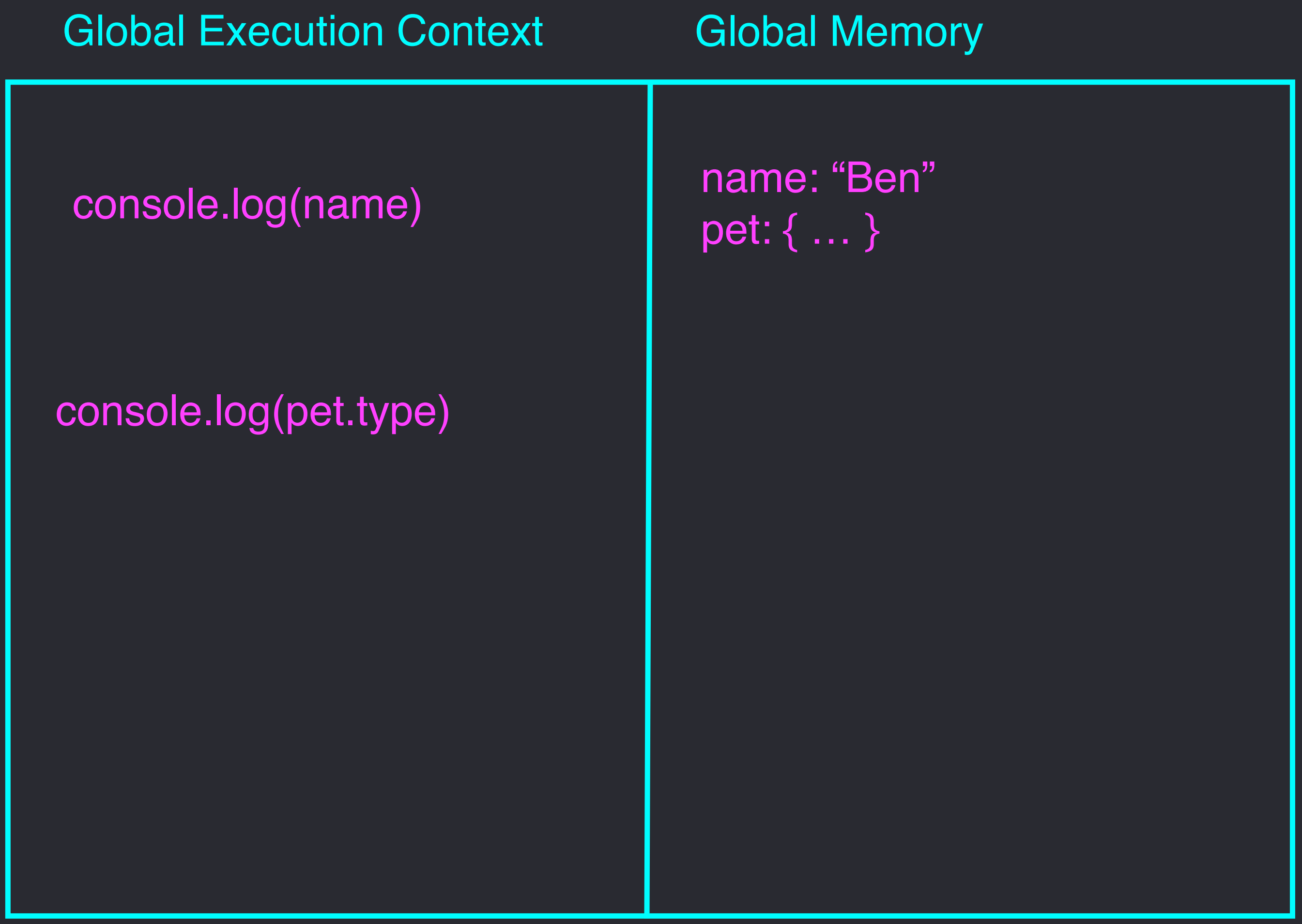
Global Memory

name: “Ben”
age: 23
drinks: […]
pet: { … }

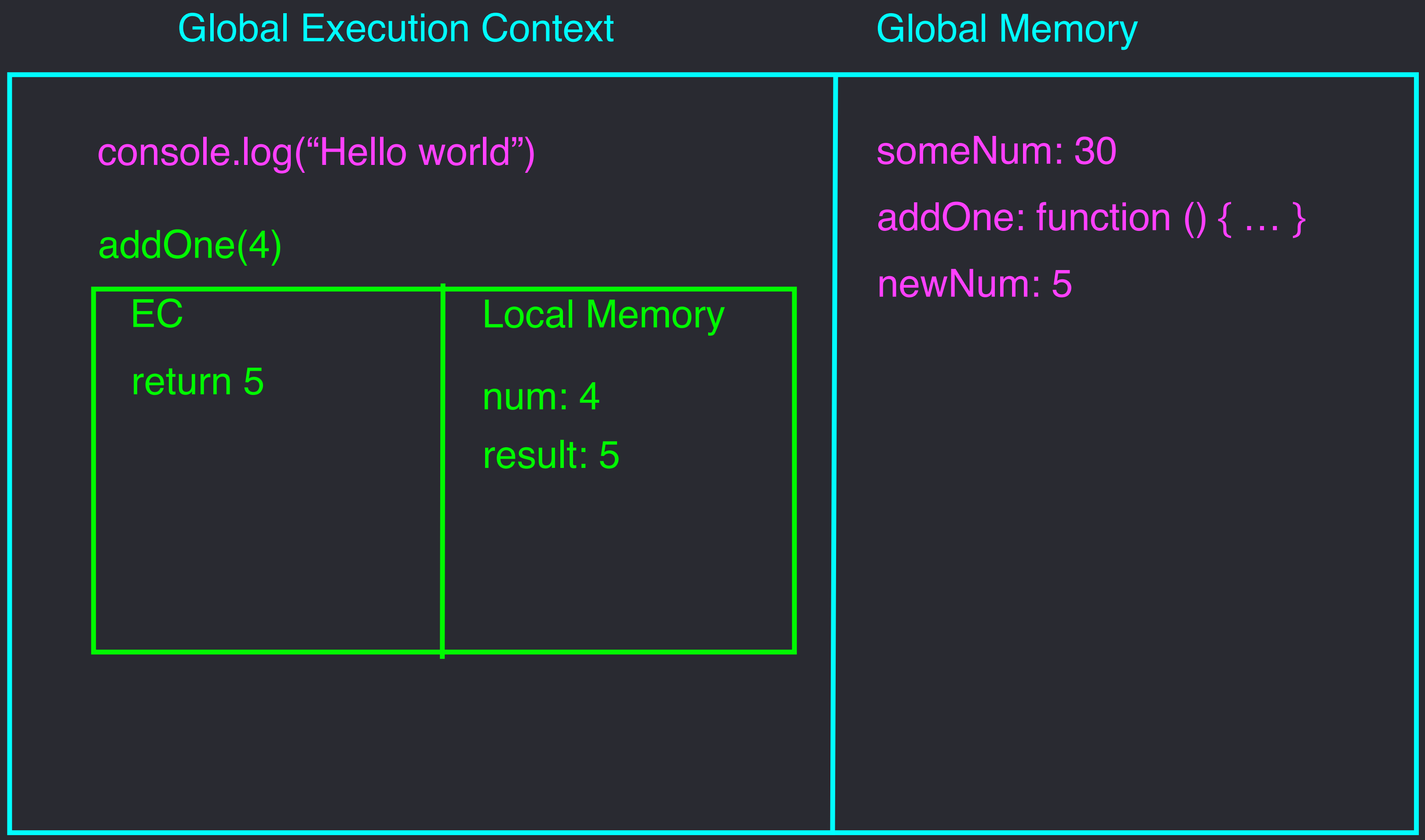
{CODENATION}



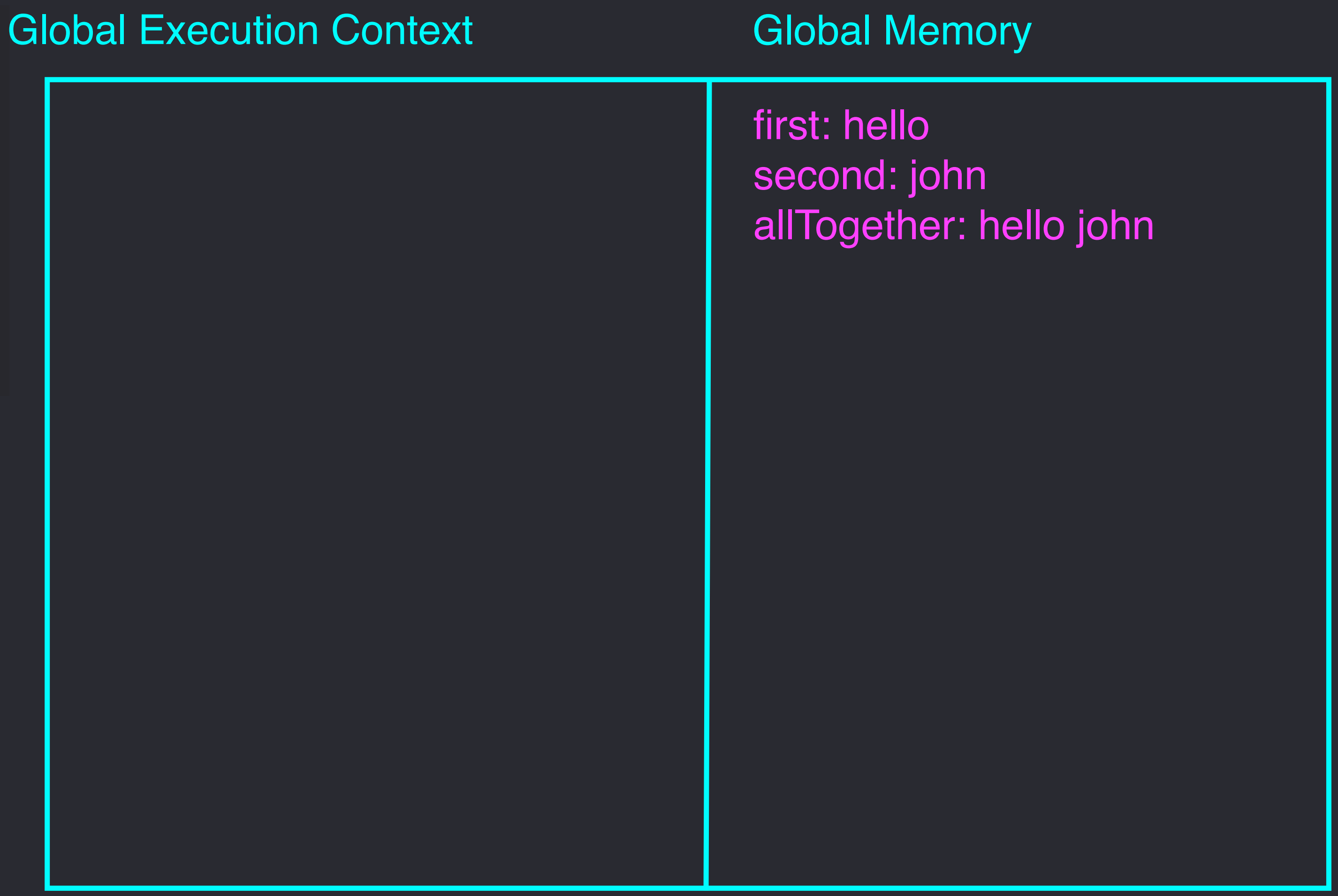
```
const myName = "Dan"
const pet = {
   name: "Xander",
   age: "6 weeks",
   type: "dog"
}

console.log(name)
console.log(pet.type)
```



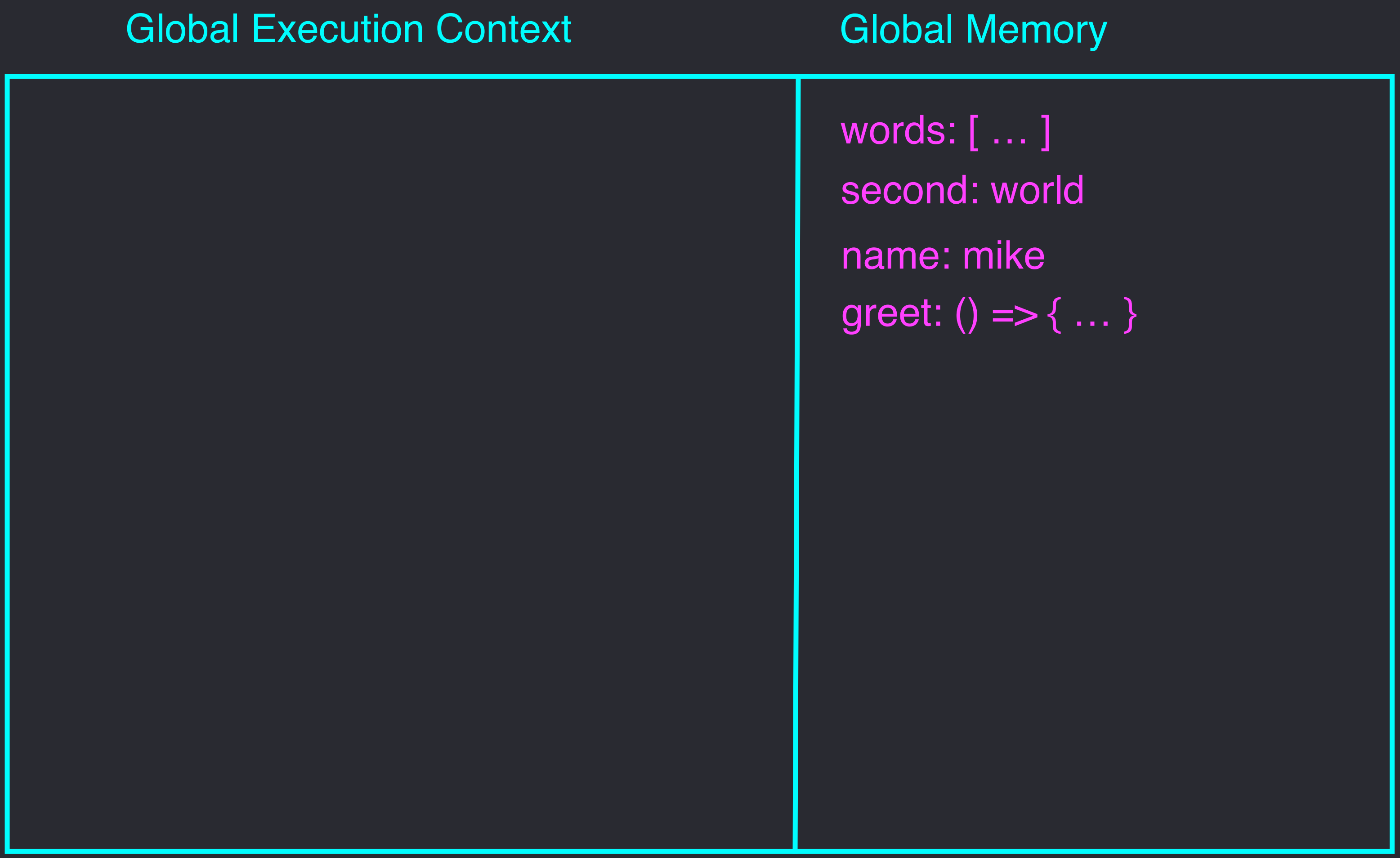
```javascript
const someNum = 30

const addOne = (num) => {
  const result = num + 1
  return result
}

console.log("Hello World")
const newNum = addOne(4)
```



```
const first = "Hello"
const second = "John"
const allTogether = `${first} ${second}`

console.log(allTogether)
```

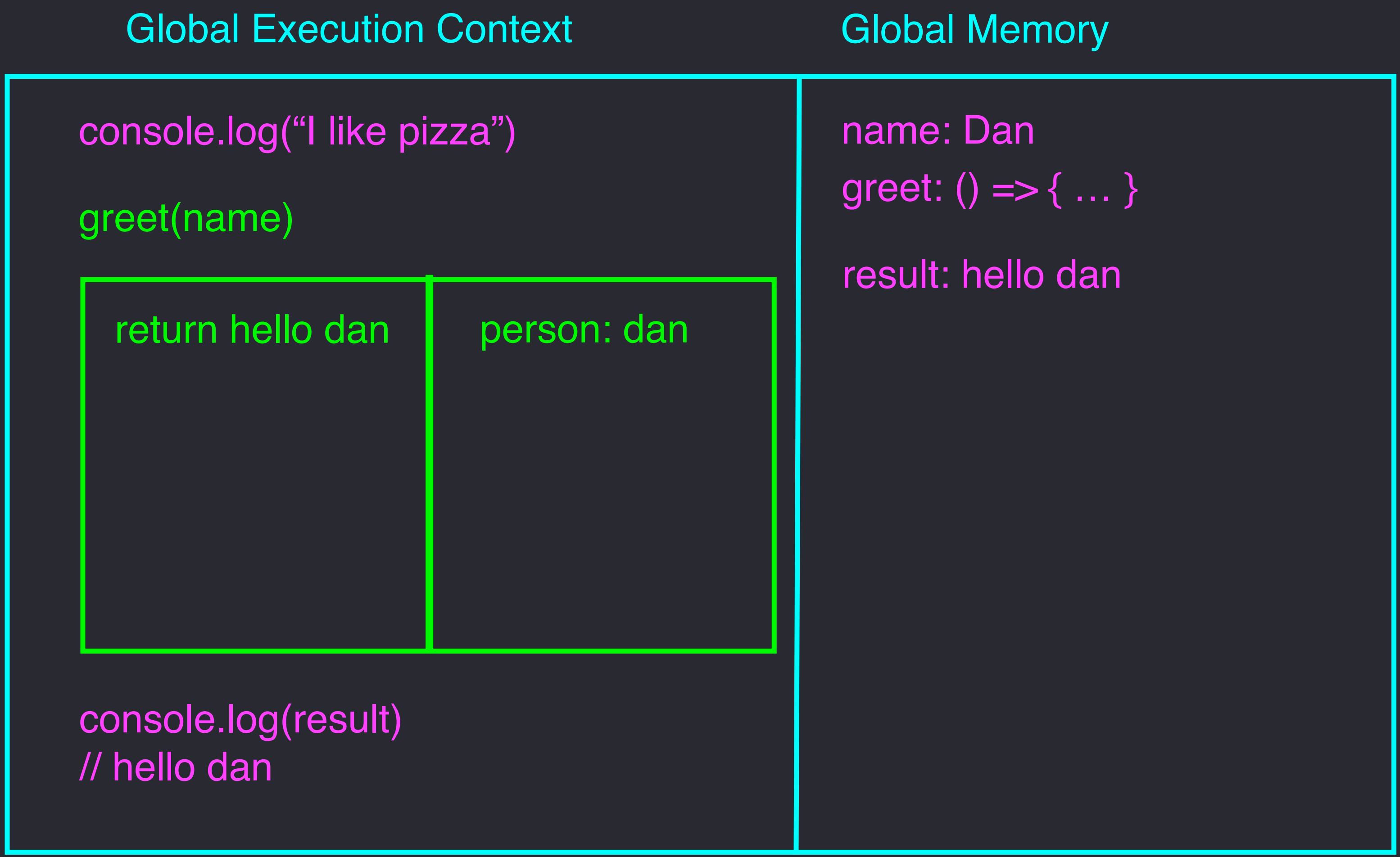


```
const words = ["hello", "world"]

const second = words[1]

let name = "Dan"
name = "Mike"

const greet = () => {
  return "hello"
}
```



CODENATION

```
let name = "Dan"

const greet = (person) => {
  return `Hello ${person}`
}

console.log("I like pizza")
const result = greet(name)

console.log(result)
```

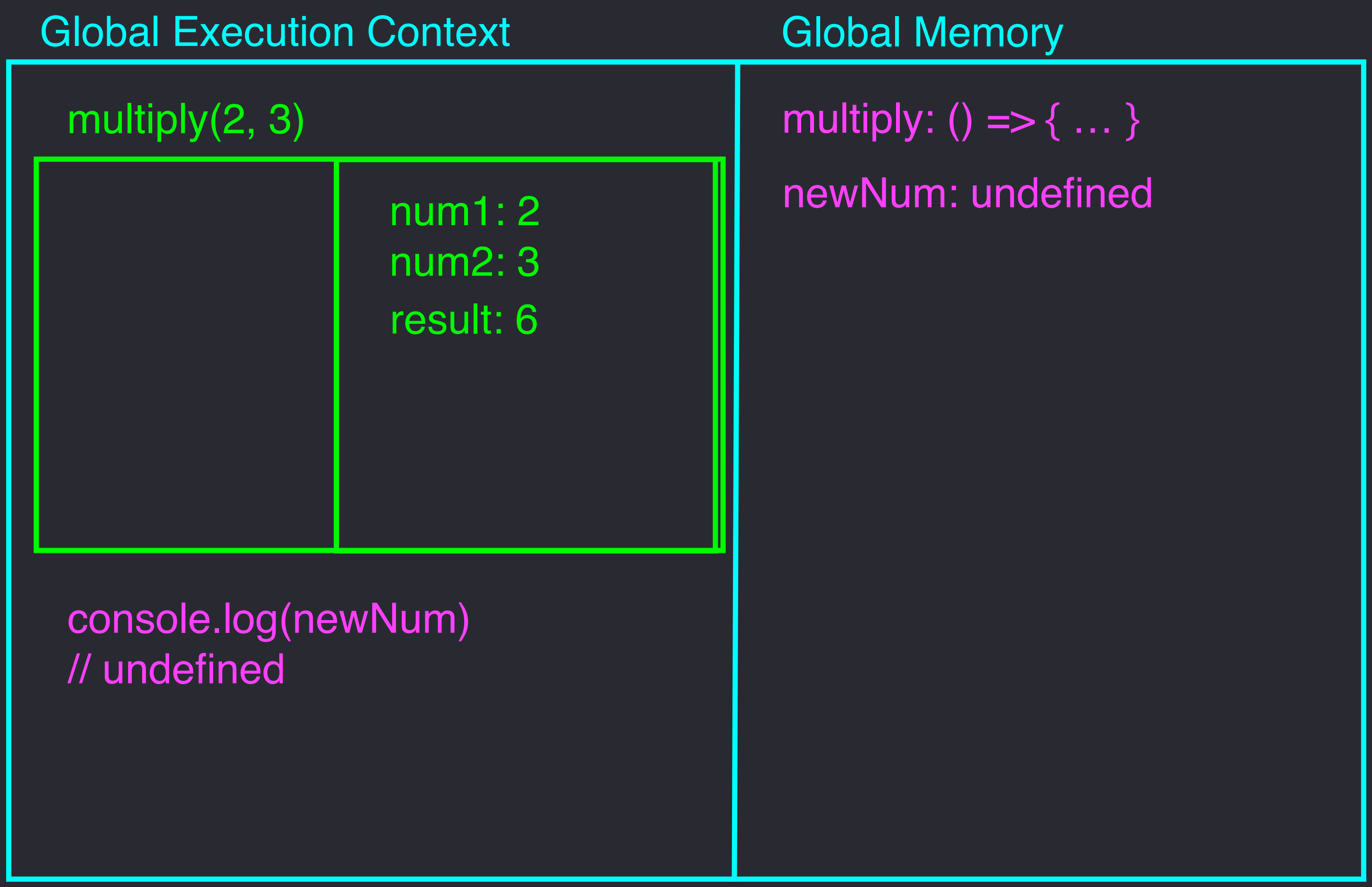


```
console.log("I like pizza")

greet(name)

return hello dan

person: dan

console.log(result)
// hello dan
```

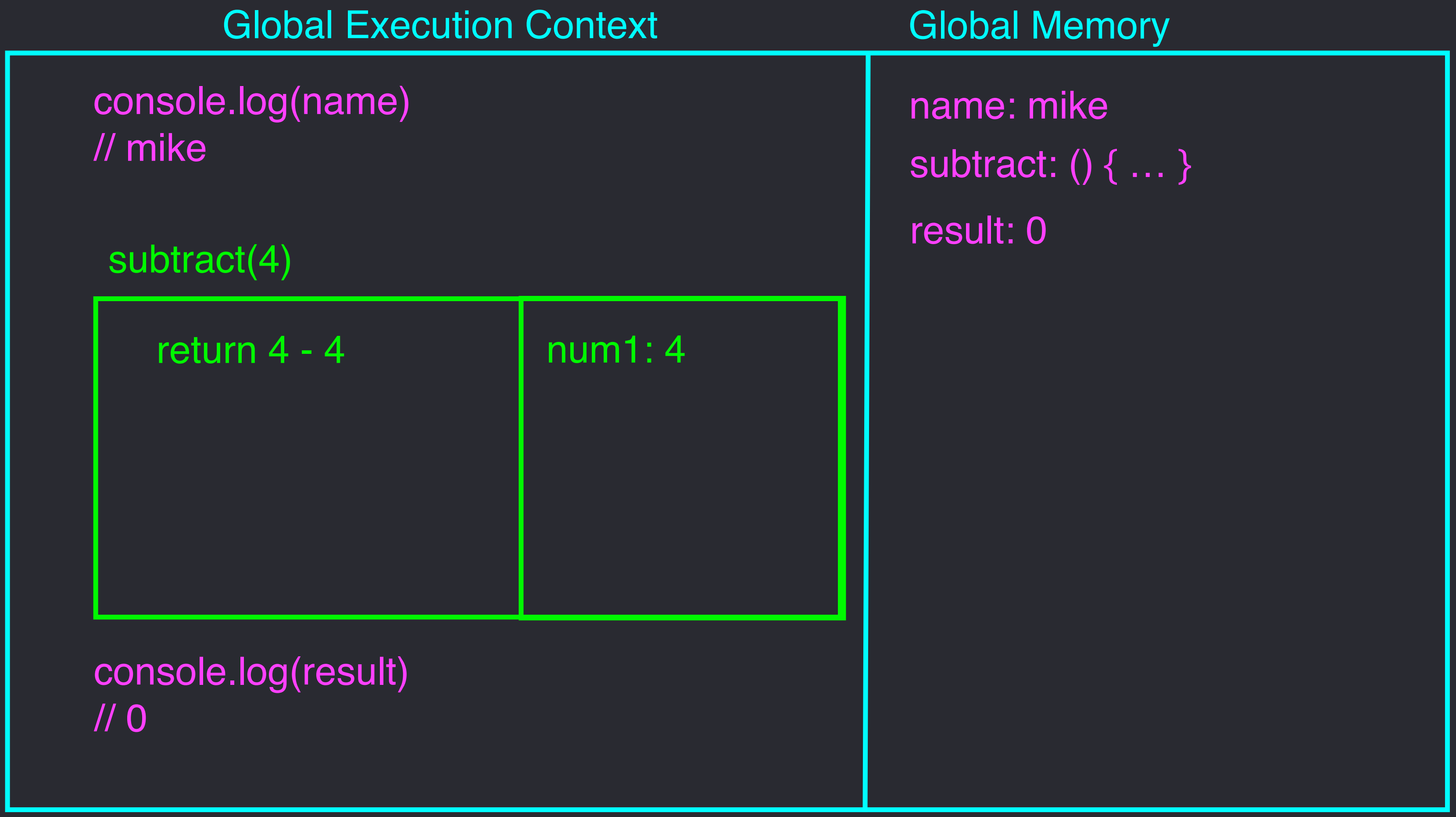
```
name: Dan
greet: () => { … }

result: hello dan
```

{CODENATION}

```
const multiply = (num1, num2) => {
  const result = num1 * num2
}

const newNum = multiply(2,3)

console.log(newNum)
```



```javascript
let name = "Mike"

function subtract(num1) {
  return num1 - 4
}

console.log(name)
const result = subtract(4)

console.log(result)
```



```javascript
const addUp = (num1, num2) => {
  return num1 + num2
}

const multiplyByTwo = (num1) => {
  return num1 * 2
}

const anotherFunc = () => {
  let myName = "dan"
  const innerFunc = (name) => {
    return `Hello ${name}`
  }
  return innerFunc(myName)
}

const value = anotherFunc()
const result = addUp(2,5)
const final = multiplyByTwo(result)
```

Revisiting Learning Objectives

To develop a **mental model** of what JavaScript is doing when it runs.

To improve **technical communication** and understanding.

{CODENATION}