**JS Runtime**

Call stack ->> global function a function b

Synchronous Asynchronous-

Execution Context🡪> variable declaratives ,scope, chain ,this

Scope chain

Variable declaration

TYPES OF SCOPE:

1 Global

let a =5 ;

const name = “hello’’

* Outside any function
* Global scope are available everywhere

2.Local

Function sum(0){

Let a = 1;

Let b =2;

Return a +b;

}

🡪variables are accessible only inside the functions.

3.Block

If(true){  
let a = 5;

B=10;

}

🡪variables are accessible only inside the block

🡪applies to let and const

* If we use strict then the functions are also block scoped.

EXAMPLE:

const a = “hello”

funcA();

function funcA(){

const b = “board”;

funcB();

function funcB(){

const c = “mango”;

funcC();}

}function funcB(){

Const d = “hey”;

Console.log (a+b+c+d);

}

Js lexical scope🡪 depends on code placement

Every scope always has access to all variables from all its outer.

>>when a variable is not a current scope the engine look up in the scope until it finds the variable.