**Synchronous and Asynchronous programming**

**Sync in java-script**

Synchronous means the code runs in a particular sequence of instructions given in the program.

each instruction waits for previous instruction to complete its execution.

In other words suppose you write a multiple line of instruction in your program in a sequence . in that case all lines of code run one by one and second lines of code waits for the previous line when previous line of code is executed now control of execution jump to the next line.

Note : JavaScript is also synchronous programming language or single threaded

// 1. SYNCHRONOUS  PROGRAMMING////////////////////////////////////////////////

console.log("Start the execution from first line of code") // Execute

console.log("Second Line.................................") // Execute

console.log("Third Line.................................")  // Execute

console.log("Fourth Line.................................") // Execute

console.log("Fifth Line.................................")  // Execute

console.log("Sixth Line.................................")  // Execute

for(let  i=0; i<2000; i++){ // Execute

    console.log("hello");

}

console.log("Eight Line.................................")// Wait for previous line of execution when complete the execution then it start the executing

console.log("Nineth Line.................................")

console.log("End the execution of code Line.................................")

**Async in java-script**

Due to synchronous programming sometimes important instruction get blocked due to some previous instruction , which causes a delay in the UI.

Asynchronous programming allow us to execute next instruction immediately and dose not block the flow of code

In other words suppose we write a 5 line of instruction in program. And unfortunately 3rd  line instruction take some time for execution. In that case  Asynchronous programming allow us to execute next instruction immediately so all line of instructions run one by one and also 3rd line of instruction execute parley . when it execute print whatever written in 3rd instruction.

// 1. ASYNCHRONOUS  PROGRAMMING///////////////////////////////////////////////

console.log("Start the execution from first line of code") // Execute

console.log("Second Line.................................") // Execute

console.log("Third Line.................................")  // Execute

console.log("Fourth Line.................................") // Execute

setTimeout(() => {*// Line 5 takes some time for executing in that case*

*Execution of control goes to next line but line 3 execute parallel*

     console.log("Hello")

},5000);

console.log("Fifth Line.................................")  // Not Wait for previous line of execution.

console.log("Sixth Line.................................")  // Execute

console.log("Seventh Line.................................") // Execute

**Callback**

A function which is passed inside the another function as an argument. that is called callback.

  functionSum(a,b){

            console.log(a+b)

        }

         functionCal(x,y,Sumcallback){

            Sumcallback(x,y)

         }

         Cal(10,20,Sum);

**Callback hell (it is problem in javaScript .)**

Nested callbacks stacked below one another forming a pyramid structure

Nesting of callback . means it create a pyramid structure. That is very difficult to understand

function getData(dataId, getNextData){

        setTimeout(()=>{

            console.log("data",dataId);

            if(getNextData){

                getNextData();

            }

        } , 2000);

    }

*// Callback hell*

  getData(1,()=>{

          console.log("Getting data 2...");

          getData(2, ()=>{

            console.log("Getting data 3...");

            getData(3, ()=>{

                console.log("Getting data 4...");

               getData(4);

            });

          });

    });