// what is a callback function in js

// the fucntion which we pass to another function , that function is known as callback function.

//this call back functions are very powerful in js world.it gives us access to the whole asynchonousworld in a synchornous single threaded language.

//js is a single threaded lang- it will do everything in a certain order, but due to callback functions  u can do async things in js

//

// eg:

// function x(y) {}

// x(function y() {});

//here function y is passing as a parameter to another function.

//here, function y is known as callback function.

//so passed y as an argument to x.

//so we are giving access to fucntion y to x

//now its x responsibility to decide when to call function y .

//now lets see how it can be used in asynchronous tasks.

//for that lets take eg of settimeout

//as we know settimeout func will take a callback func and lets prints something in it to visualize

setTimeout(function () {

  console.log("timer");

}, 5000);

function x(y) {

  console.log("x");

  y();

}

x(function y() {

  console.log("y");

});

//now see how js execute it.

//as js is single threaded lang,it means code execute one after another and in order sequence

//so settimeout will take this callback func and store it in separate space.and it will attach a timer to it and store it.

//js wont wait for settimeout to finish . so callback func give us the power of asynchronity

//it doesnot wait for 5000 ms to expire and wt ever needs to be done after 5000 ms ,we are passing that piece of code as a callback function

//it will be later exeucted

//js waits for none . so it moves to next line see funciton def of x     and it will try to call this x functin,it wil pass y as a callback function and do console.log(x);

//till this time this 5000 ms is not expired.

//so it will print x ,then print y then after 5000 ms go to settimeout func

//so output will be

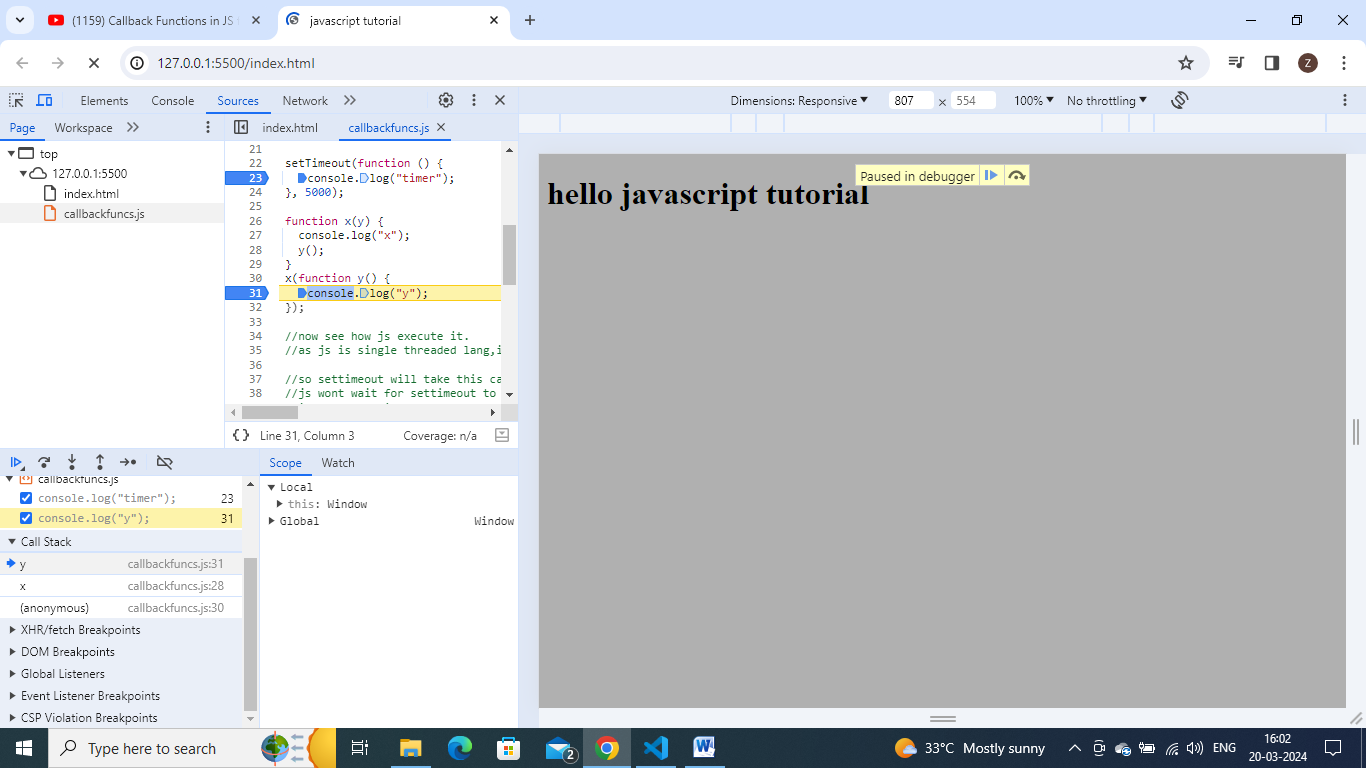
//x y then after 5000 ms gap we get timer

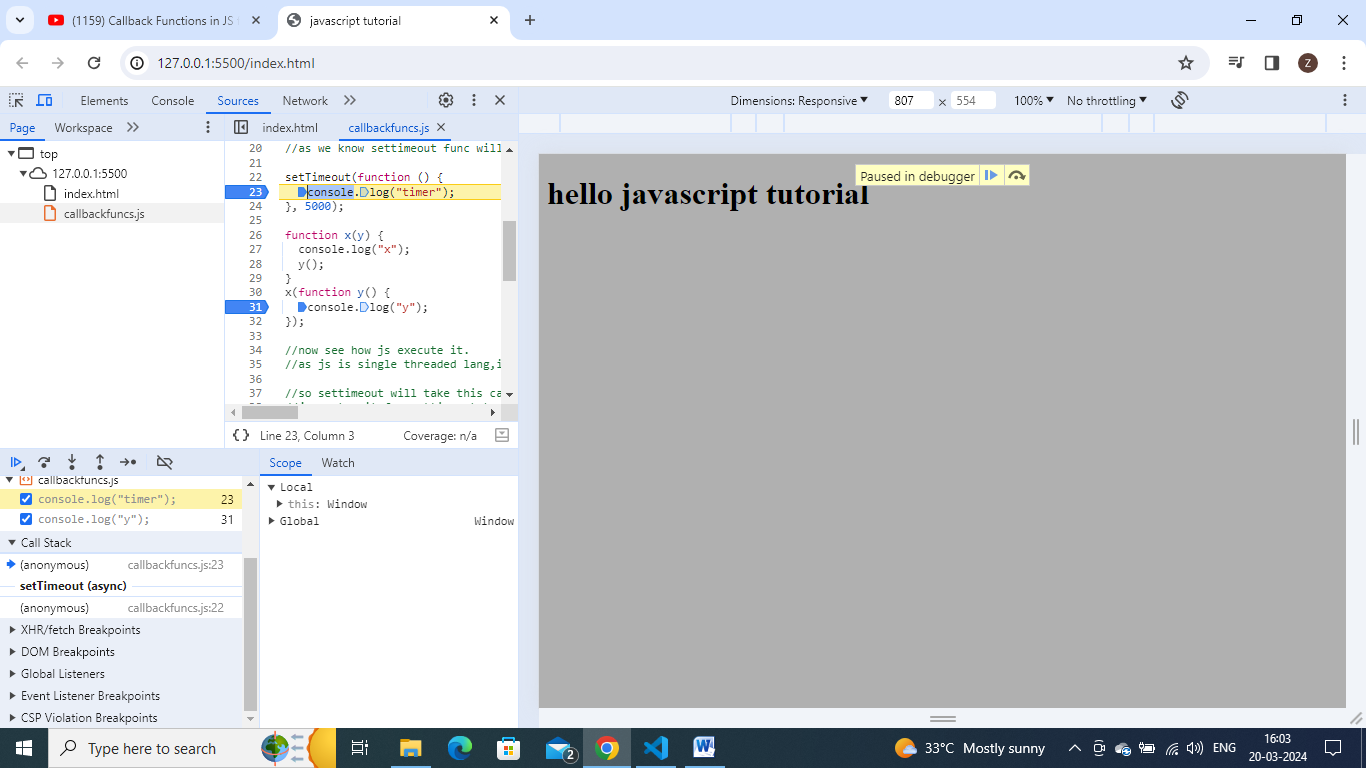
//so settimeout asynchronous operations are not possible , without these callbacks

//we have given this callback func to settimeout to execute after some time

//keep debugger at console.log("y");  and keep another debugger at console.log("timer")  and observe callstack

//so observe callstack -- it will have x,y then  callstack becomes empty, then after 5000  ms , then anonymous fun will get into callstack





Before getting anonymous func into callstack,first becomes empty then aftr 5000 ms , get anonymous fun into callstack.anonymous got into callstack automatically after 5000 ms