

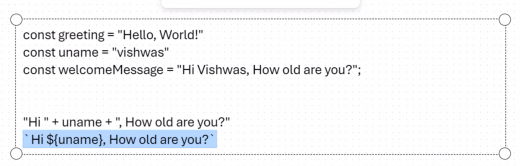
Object person with nested properties like address, city and zipcode, the expression

Console.log(person.address.city) accesses and prints the city property of the address object inside the person object.

1. Person: The main object that contains various properties, such as name, age and address.
2. Address: A property of person that is itself an object. This object contains further properties, like city and zipcode.
3. City: A property of the address object that holds the name of the city.

By dot notation javascipt is accessing the city property inside the address object which is a property of the person object.

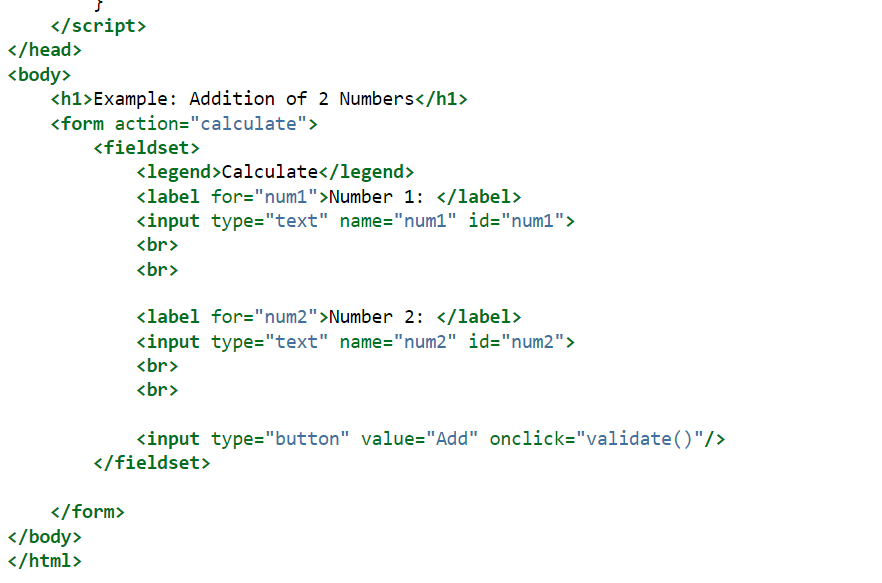
String literal

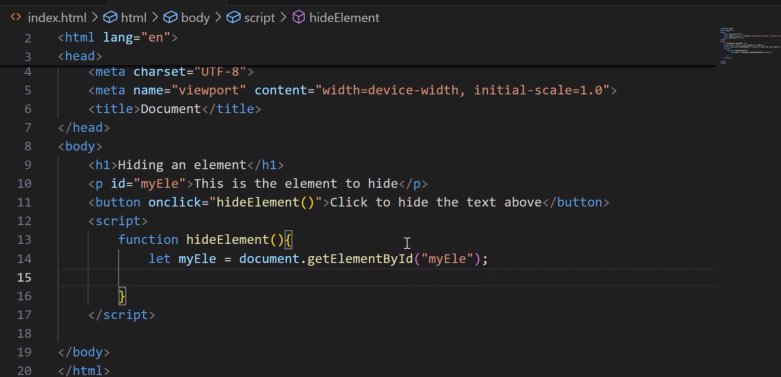
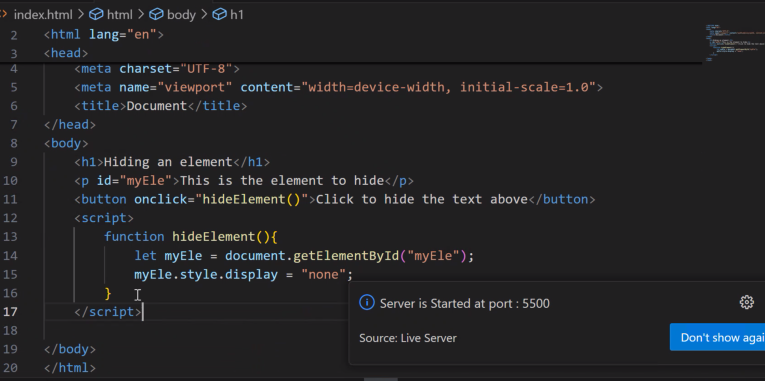


<http://deloitte-foundations-platform-vks.s3-website-us-east-1.amazonaws.com/day8/lab01.html>



1.



1. 
2. 

Hiding an element body code:

<h1>Hiding an element</h1>

    <p id="myEle">This is the element to hide</p>

    <button onclick="hideElement()">Click to hide the text above</button>

    <script>

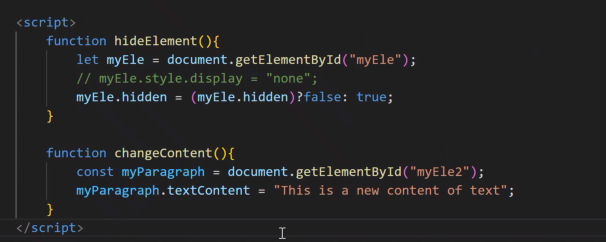
        function hideElement(){

            let myEle = document.getElementById("myEle");

            myEle.style.display = "none";

        }

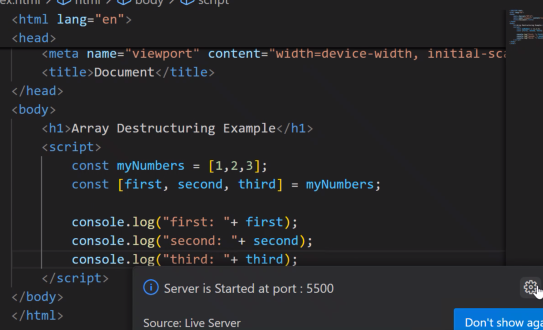
    </script>







ARRAY DESTRUCTURING



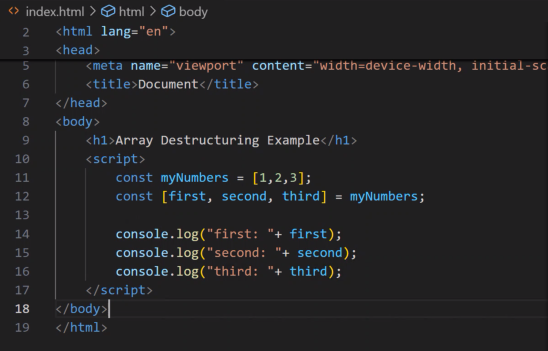
const myNumbers = [1,2,3];

        const [first, second, third] = myNumbers;

        console.log("first: "+ first);

        console.log("second: "+ second);

        console.log("third: "+ third);

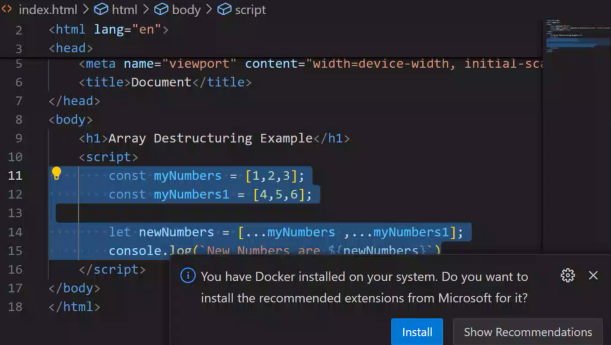


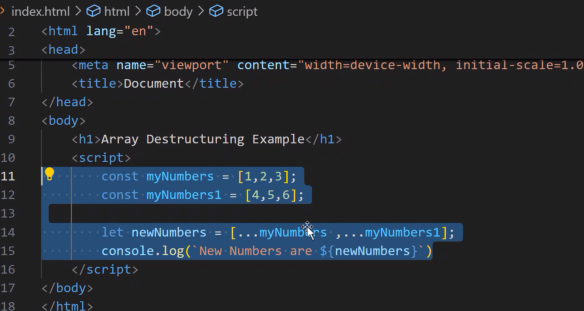
const myNumbers = [1,2,3];

        const myNumbers1 = [4,5,6];

        let newNumbers = [...myNumbers ,...myNumbers1];

        console.log(`New Numbers are ${newNumbers}`





**Explanation:**

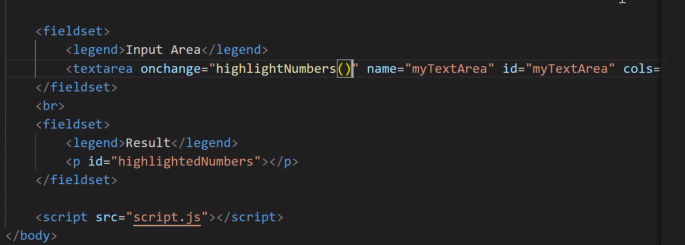
1. **Array Declaration**:
   * const myNumbers = [1, 2, 3];: This creates a constant array named myNumbers that contains the numbers 1, 2, and 3.
   * const myNumbers1 = [4, 5, 6];: This creates another constant array named myNumbers1 that contains the numbers 4, 5, and 6.
2. **Spread Operator**:
   * let newNumbers = [...myNumbers, ...myNumbers1];

 The spread operator (...) is used here to expand (or "spread") the elements of both myNumbers and myNumbers1 into a new array.

 The expression [...myNumbers, ...myNumbers1] effectively combines the two arrays into one. So, newNumbers will be [1, 2, 3, 4, 5, 6].

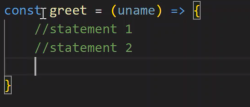
HIGHLIGHTING

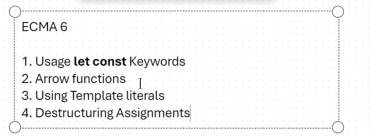




[lab08 (deloitte-foundations-platform-vks.s3-website-us-east-1.amazonaws.com)](http://deloitte-foundations-platform-vks.s3-website-us-east-1.amazonaws.com/day8/lab07.html)

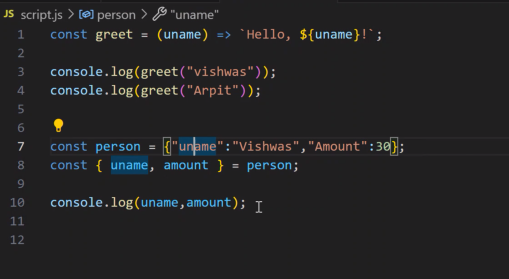
<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Regular_expressions>

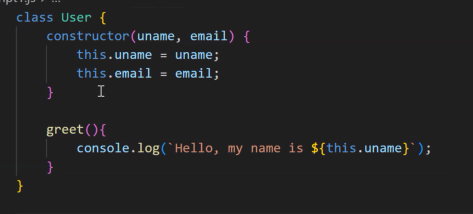




And oop







lass User {

Class User{

constructor(uname, email) {

        this.uname = uname;

        this.email = email;

    }

    greet(){

        console.log(`Hello, my name is ${this.uname}`);

    }

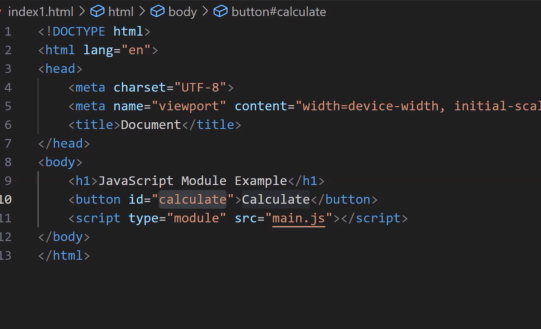
}

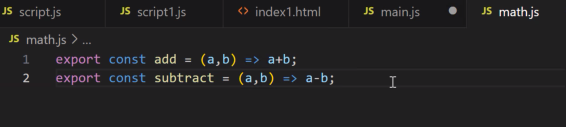
const u1 = new User("Vishwas", "vishwas@cloudthat.com");

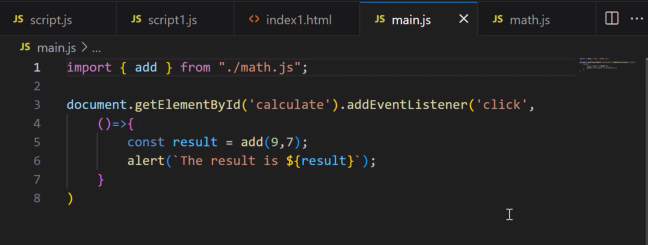
u1.greet();

const u2 = new User("Neelesh","neelesh@deloitte.com");

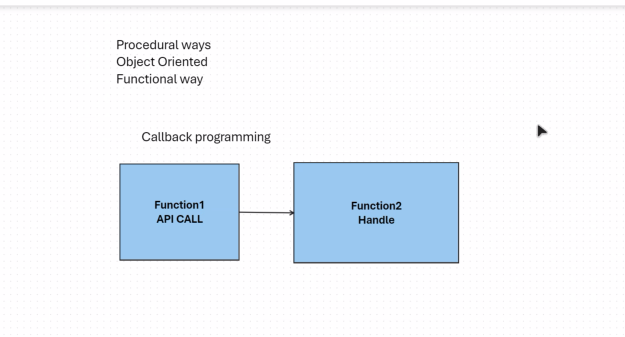
u2.greet();







CALLBACK FUNCTIOn



One function calls another function which in turn calls another function and it keeps on repeating like a chain of callbacks until the results comeback.

Asynchronous programming?

In an asynchronous operation – network request, file reading

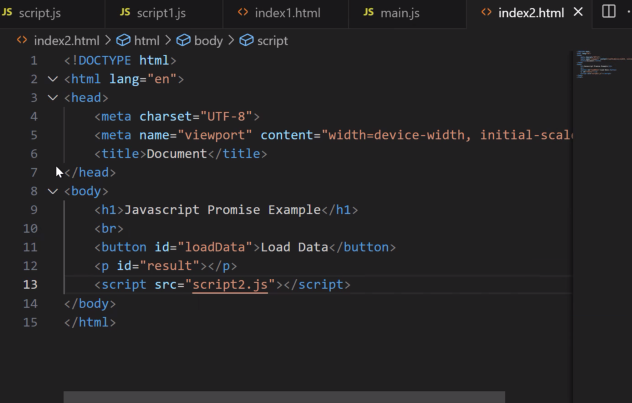
When call a function a promise is created, it takes two parameters – either it can resolve or reject.

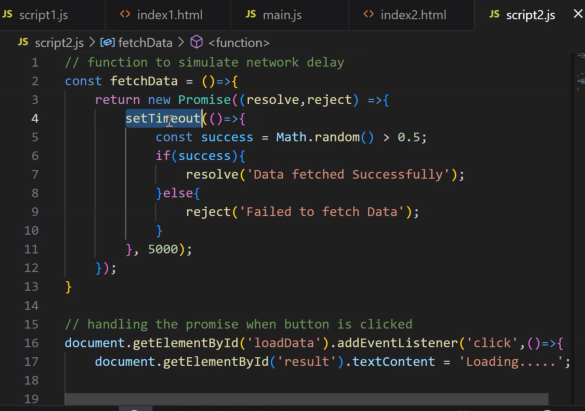
resolve() – if the answer is got

and

reject() – if there is some error and the answer is not satisfied.

two methods





Network fetch

<https://codeshare.io/0b8Npg>

Extracting numbers

[lab08 (deloitte-foundations-platform-vks.s3-website-us-east-1.amazonaws.com)](http://deloitte-foundations-platform-vks.s3-website-us-east-1.amazonaws.com/day8/lab07.html)

