Lab: JavaScript Syntax and Basic Web

Problems for exercises and homework for the "Software Technologies" course @ SoftUni.

You can submit your solutions here https://judge.softuni.bg/Contests/223/.

1. Sum Numbers with HTML and JS

Create an HTML form holding two text fields and a result field and write a JavaScript function to **sum** them.

Examples

Input	Output
1 2	3

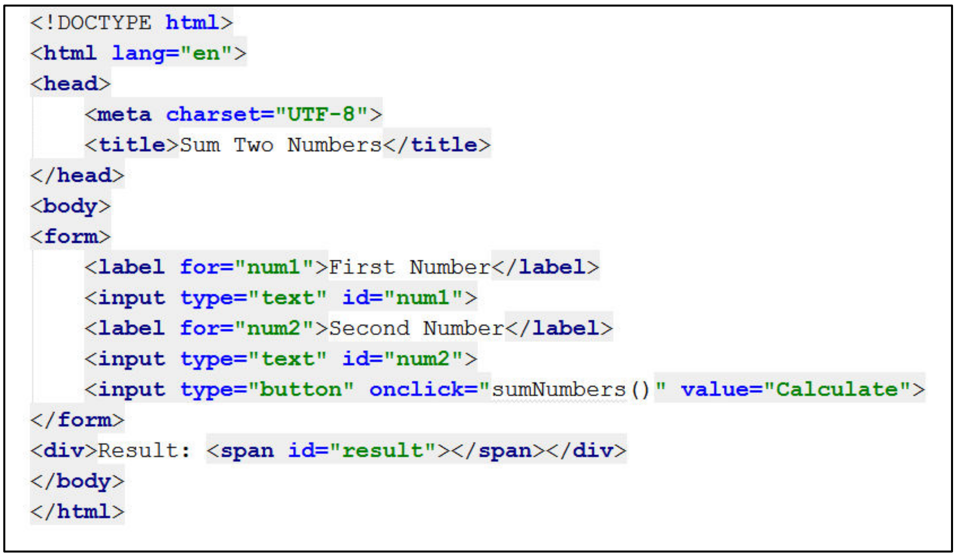
Input	Output
5 5	10

Input	Output
10 15	25

Input	Output
1 5	6

1. Create the Form

After you create the file, it's time to make the form. Create a **`<form>`** tag with some **`<input>`** fields inside:



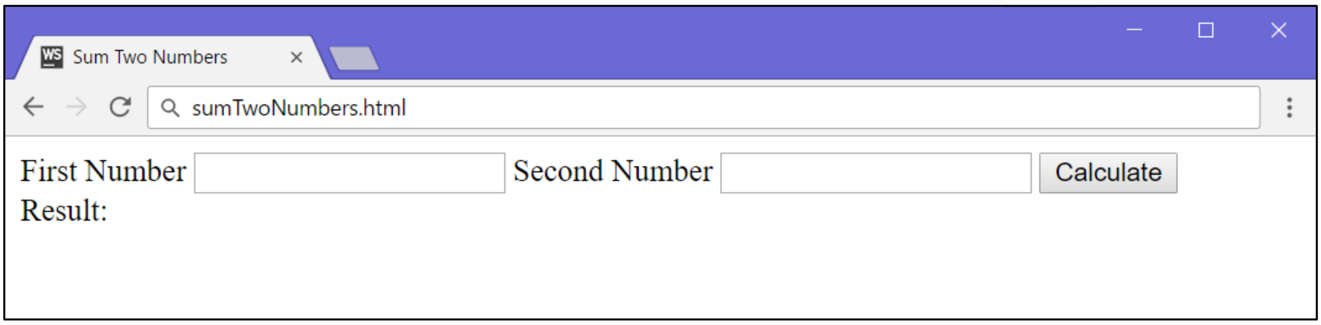
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Sum Two Numbers</title>
</head>
<body>
<form>
    <label for="num1">First Number</label>
    <input type="text" id="num1">
    <label for="num2">Second Number</label>
    <input type="text" id="num2">
    <input type="button" onclick="sumNumbers()" value="Calculate">
</form>
<div>Result: <span id="result"></span></div>
</body>
</html>
```

The input fields are as follows:

- A **text** input field, called **num1**
- A **text** input field, called **num2**
- A **button** input, which calls the "**`sumNumbers()`**" function upon being clicked, and has the text "Calculate" inside it.

Apart from the form, we've also created a **div**, which will hold the **result** of the calculation.

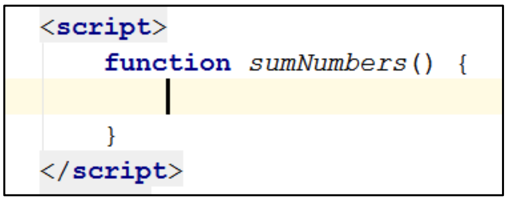
When we're done writing the code, our little html page should look something like this:



2. Write the JavaScript Code

Let's create a **`<script>`** tag after the form, which will hold our logic.

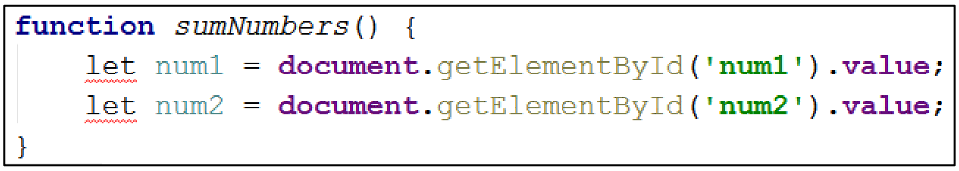
In order to perform the calculation, we need to **access** the **num1** and **num2** fields' **values** and then sum them. Let's first create a function **`calculateResult()`**, which will be called on click:



```
<script>
    function sumNumbers() {

    }
</script>
```

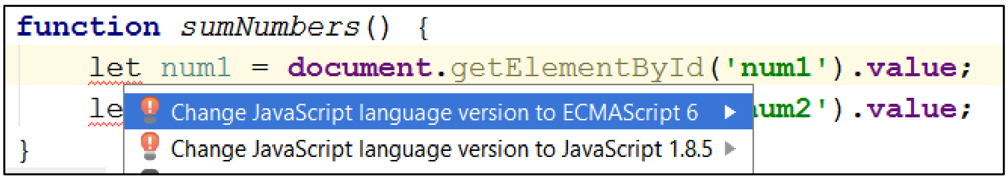
After we create the script tag, we can get both the elements **by id**, and sum them:



```
function sumNumbers() {
    let num1 = document.getElementById('num1').value;
    let num2 = document.getElementById('num2').value;
}
```

We've written the code, but WebStorm is showing us some errors. The reason it's showing errors with the **let** expression is because it **didn't exist yet** in **ECMAScript 5.1**, which WebStorm uses by **default**. We need to **fix** that.

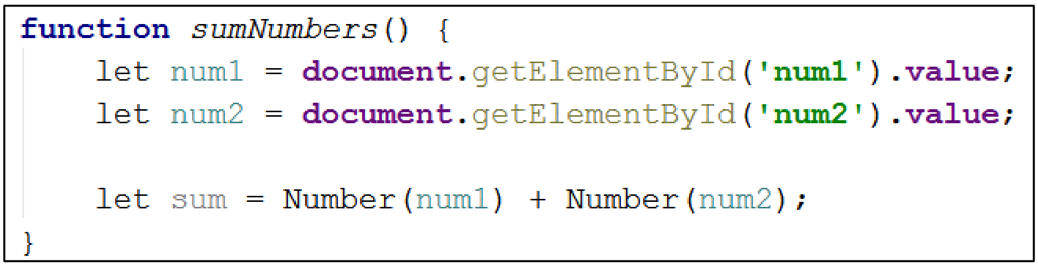
Place the cursor over the **let** expression and press **`[Alt+Enter]`** to open the **quick fix** menu and **change** the **JavaScript version** to **ECMAScript 6**:



```
function sumNumbers() {
    let num1 = document.getElementById('num1').value;
    le                                        um2').value;
}
```

Change JavaScript language version to ECMAScript 6
Change JavaScript language version to JavaScript 1.8.5

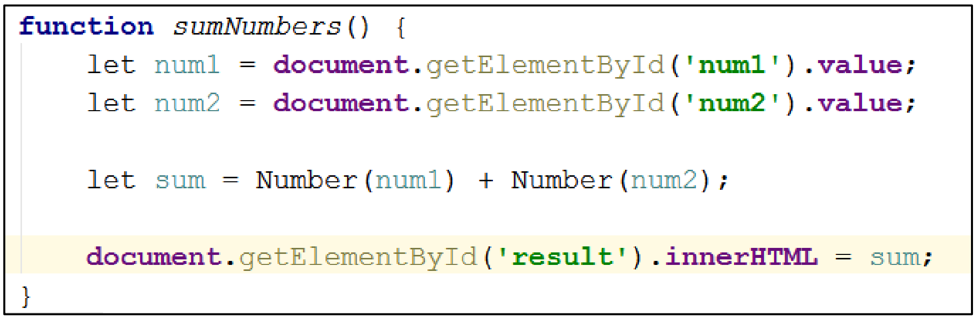
Afterwards, the errors should disappear. Let's continue writing the code. Next, we need to sum the numbers. We'll do that by creating a variable called **sum**, which will hold the result of num1 and num2, converted to the **number** data type:



```
function sumNumbers() {
    let num1 = document.getElementById('num1').value;
    let num2 = document.getElementById('num2').value;

    let sum = Number(num1) + Number(num2);
}
```

Finally, we need to set the text of the **`#result`** element, which we'll access by its **id**, and set its **inner HTML** value to the sum:



```
function sumNumbers() {
    let num1 = document.getElementById('num1').value;
    let num2 = document.getElementById('num2').value;

    let sum = Number(num1) + Number(num2);

    document.getElementById('result').innerHTML = sum;
}
```

2. Calculate Expression

Write a JavaScript program to print the value of the following expression:

- **((30 + 25) * 1/3 * (35 - 14 - 12))**^{2}

Examples

Input	Output
(no input)	27225

3. Sum Two Numbers

Write a JavaScript program to sum **two numbers**, which are received as a **string array**.

Examples

Input	Output
['10', '20']	30

Input	Output
['66', '4']	70

4. Three Integers Sum

Write a JavaScript program, which receives **three numbers**, as a **string array**. Your task is to check whether there exists a number in the sequence, which is equal to the **sum** of the other two.

- If they are, print the numbers and their sum in the following format: “**`${num1} + ${num2} = ${sum}`**”. Print the elements, in such way, that **num1 <= num2**
- If there’s no such element, print “**No**”.

Examples

Input	Output
8 15 7	7 + 8 = 15

Input	Output
-5 -3 -2	-3 + -2 = -5

Input	Output
3 8 12	No

5. Symmetric Numbers

Write a JavaScript program, which receives **a number n**, as a **string array** with a single element, and print all symmetrical numbers in the range **[1…n]**.

Examples

Input	Output
100	1 2 3 4 5 6 7 8 9 11 22 33 44 55 66 77 88 99

Input	Output

750	1 2 3 4 5 6 7 8 9 11 22 33 44 55 66 77 88 99
	101 111 121 131 141 151 161 171 181 191 202
	212 222 232 242 252 262 272 282 292 303 313
	323 333 343 353 363 373 383 393 404 414 424
	434 444 454 464 474 484 494 505 515 525 535
	545 555 565 575 585 595 606 616 626 636 646
	656 666 676 686 696 707 717 727 737 747

6. Sums by Town

You are given a sequence of **JSON strings** holding **town** + **income**. Write a JS function to **sum** and print the **incomes** for **each town**. Towns can appear **multiple times**. In the output, **order** the towns by **name**.

Examples

Input	Output

{"town":"Sofia","income":200}
{"town":"Varna","income":120}
{"town":"Pleven","income":60}
{"town":"Varna","income":70}
	Pleven -> 60
	Sofia -> 200
	Varna -> 190

7. Largest 3 Numbers

Write a program to read an **array** of **numbers** and find and print the **largest 3** of them, sorted in **descending order**.

Examples

Input	Output
10 30 15 20 50 5	50 30 20

Input	Output
20 30	30 20

Input	Output
10 5 20 3 20	20 20 10

8. Extract Capital-Case Words

Write a **JavaScript** function to **extract** from array of strings all **capital-case** words. All **non-letter chars** are considered **separators**.

Examples

Input	Output
We start by HTML, CSS, JavaScript, JSON and REST. Later we touch some PHP, MySQL and SQL. Later we play with C#, EF, SQL Server and ASP.NET MVC. Finally, we touch some Java, Hibernate and Spring.MVC.	HTML, CSS, JSON, REST, PHP, SQL, C, EF, SQL, ASP, NET, MVC, MVC