# Упражнения: Първи стъпки в програмирането

Задачи за упражнение в клас и за домашно към курса [„Основи на програмирането“ @ СофтУни](https://softuni.bg/courses/programming-basics).

## Празен Visual Studio Code проект

Създайте празен проект във Visual Studio Code. Ще обединяваме решенията на всички задачи, под формата на отделни файлове в този проект. Тази възможност е изключително удобна, когато искаме да работим по няколко проекта и бързо да превключваме между тях или искаме да обединим логически няколко взаимосвързани проекта. Това ни помага да запазваме решенията на задачите отделно и да ги пазим, за да ги използваме за други задачи или преговор.

1. Стартирайте Visual Studio Code
2. Създайте нова папка, която ще държи отделните решения. Ще се отвори диалогов прозорец, в който ще трябва да изберете нейната директория. Препоръчително е да именувате папката според темата на заданието, пример "**First-Steps-in-Calculations**"



След това изберете папката като работна среда, за да добавяте файловете с JavaScript решенията на своите задачи в нея.



Панелът в ляво ще изглежда така:



## Конзолна програма “Hello SoftUni”

Първата задача от тази тема е да се напише **конзолна програма**, която **отпечатва** на конзолата: "Hello, SoftUni! ".

### Примерен вход и изход:

|  |  |
| --- | --- |
| **вход** | **изход** |
| (няма) | Hello, SoftUni! |

1. Създайте **нов JavaScript файл** в съществуващата папка и го именувайте подходящо. Препоръчително е всеки скриптов файл да се казва както името на задачата, чието решение съдържа.

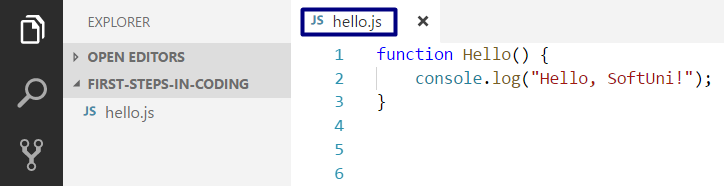




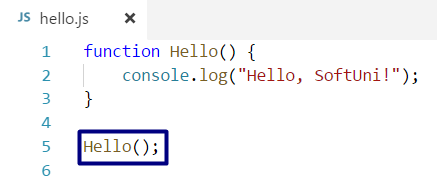
1. Съдържанието на новият файл ще се отвори в прозореца в дясно.



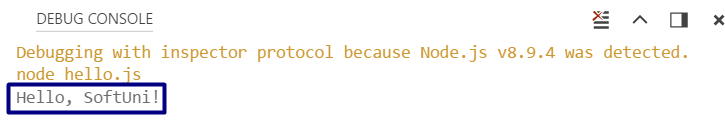
1. Отидете във файла hello.js и създайте функцията Hello(). Можете да си помогнете с картинката по-долу:



1. Стартирайте програмата с Ctrl + [F5]. За да получим резултат трябва да "повикаме" функцията:



Можем да видим резултата в конзолата по-долу:



1. **Тествайте** решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/152#0>. Сменете стратегията на judge с **"JavaScript code(NodeJS)"**. Трябва да получите 100 точки (напълно коректно решение):





## Конзолна програма “Expression”

Напишете **конзолна програма**, която пресмята и отпечатва стойността на следния **числен израз**:

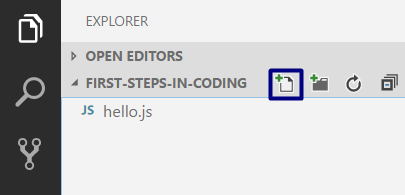
|  |
| --- |
| (3522 + 52353) \* 23 - (2336 \* 501 + 23432 - 6743) \* 3 |

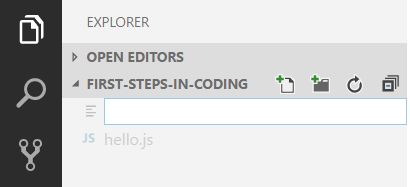
Забележка: не е разрешено да се пресметне стойността предварително (например с Windows Calculator).

### Примерен вход и изход:

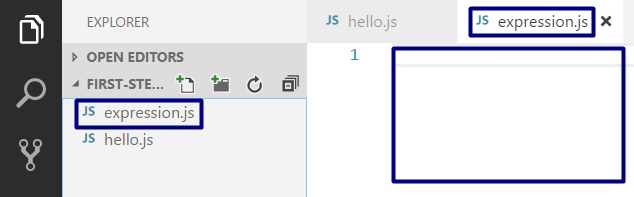
|  |  |
| --- | --- |
| **вход** | **изход** |
| (няма) | -2275950 |

1. Създайте **нов JavaScript файл** в съществуващата папка и го именувайте подходящо.

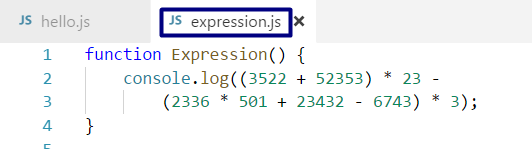




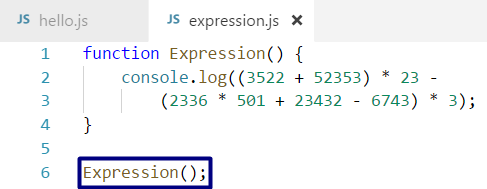
1. Съдържанието на новият файл ще се отвори в прозореца в дясно.



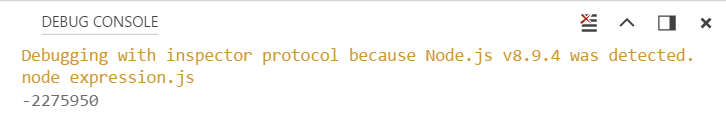
1. Отидете във файла expression.js и създайте функцията Expression(). Можете да си помогнете с картинката по-долу:



1. Стартирайте програмата с Ctrl + [F5]. За да получим резултат трябва да "повикаме" функцията:



Можем да видим резултата в конзолата по-долу:



1. Тествайте вашата програма в judge системата: <https://judge.softuni.bg/Contests/Practice/Index/150#1>.

## Числата от 1 до 20

Напишете конзолна програма, която отпечатва числата от 1 до 20 на отделни редове на конзолата.

### Примерен вход и изход:

|  |  |
| --- | --- |
| **вход** | **изход** |
| (няма) | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 |

1. Създайте **нов JavaScript файл** в съществуващата папка и го именувайте "Nums-1-To-20". Използвайте тялото на функцията "nums1To20()":

function nums1To20() {

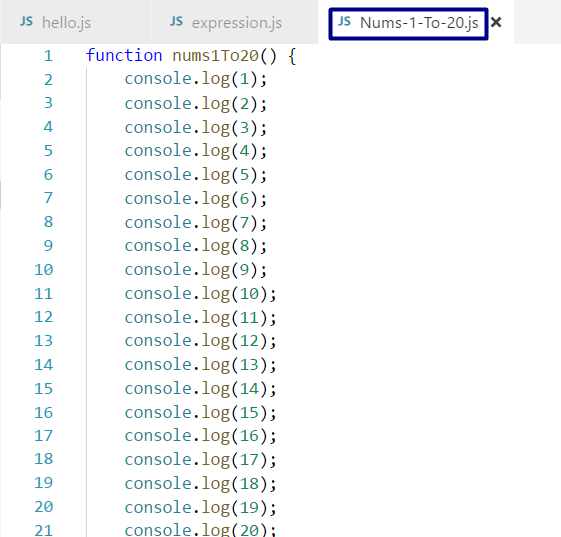
// Your code goes here

}

// Call the function to execute the code inside

nums1To20();

1. Напишете 20 команди console.log(), една след друга, за да отпечатате числата от 1 до 20.



1. Стартирайте програмата с Ctrl + [F5]. За да получите резултат трябва да "повикате" функцията.
2. Тествайте вашата програма в judge системата: [https://judge.softuni.bg/Contests/Practice/Index/150#2](https://judge.softuni.bg/Contests/Practice/Index/150%232).
3. Можете ли да напишете програмата по **по-умен начин**, така че да не повтаряте 20 пъти една и съща команда? Потърсете в Интернет информация за „[**for loop JavaScript**](https://www.google.bg/search?q=for+loop+javascript&oq=for+loop+jav&aqs=chrome.2.69i57j0l5.3355j0j7&sourceid=chrome&ie=UTF-8)“.

## Триъгълник от 55 звездички

Напишете конзолна програма, която отпечатва **триъгълник от 55 звездички**, разположени на 10 реда:

|  |
| --- |
| \*  \*\*  \*\*\*  \*\*\*\*  \*\*\*\*\*  \*\*\*\*\*\*  \*\*\*\*\*\*\*  \*\*\*\*\*\*\*\*  \*\*\*\*\*\*\*\*\*  \*\*\*\*\*\*\*\*\*\* |

### Примерен вход и изход:

|  |  |
| --- | --- |
| **вход** | **изход** |
| (няма) | \*  \*\*  \*\*\*  \*\*\*\*  \*\*\*\*\*  \*\*\*\*\*\*  \*\*\*\*\*\*\*  \*\*\*\*\*\*\*\*  \*\*\*\*\*\*\*\*\*  \*\*\*\*\*\*\*\*\*\* |

1. Създайте **нов JavaScript файл** в съществуващата папка и го именувайте “Triangle-Of-55-Stars”.
2. Напишете код, който печата триъгълника от звездички, например чрез 10 команди, подобни на console.log("\*"). Използвайте тялото на функцията "triangleOfStars()**"**:

function triangleOfStars() {

// Your code goes here

}

// Call the function to execute the code inside

triangleOfStars();

1. **Тествайте** кода си в judge системата: <https://judge.softuni.bg/Contests/Practice/Index/150#3>.
2. Опитайте да подобрите решението си, така че да няма много повтарящи се команди. Може ли това да стане с **for цикъл**?

## Лице на правоъгълник

Напишете програма, която прочита от конзолата две числа, **a** и **b,** въведени от потребителя, пресмята и отпечатва **лицето на правоъгълник** със страни **a** и **b**.

### Примерен вход и изход:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| [2,7] | 14 | [7,8] | 56 | [12,5] | 60 |

1. Създайте **нов JavaScript файл** в съществуващата папка и го именувайте подходящо.
2. За да прочетете двете числа, използвайте следния код:

function rectangleArea([arg1, arg2]) {

let a = Number(arg1);

let b = Number(arg2);

// Print the area

}

// Call the function to execute the code inside

rectangleArea(["2", "7"]);

rectangleArea(["7", "8"]);

1. Допишете програмата по-горе, за да пресмята лицето на правоъгълника и да го проверява.
2. Тествайте решението си в judge системата: <https://judge.softuni.bg/Contests/Practice/Index/150#4>.

## \* Квадрат от звездички

Напишете конзолна програма, която прочита от конзолата цяло положително число **N**, въведено от потребителя и отпечатва на конзолата **квадрат от N звездички**.

### Примерен вход и изход:

|  |  |
| --- | --- |
| **вход** | **изход** |
| 3 | \*\*\*  \* \*  \*\*\* |
| 4 | \*\*\*\*  \* \*  \* \*  \*\*\*\* |
| 5 | \*\*\*\*\*  \* \*  \* \*  \* \*  \*\*\*\*\* |

1. Направете конзолна Node.js програма. За да прочетете числото **N** (2 ≤ N ≤100), използвайте следния код:

function squareOfStars([arg1]) {

let n = Number(arg1);

// Print the square

}

// Call the function to execute the code inside

squareOfStars(["3"]);

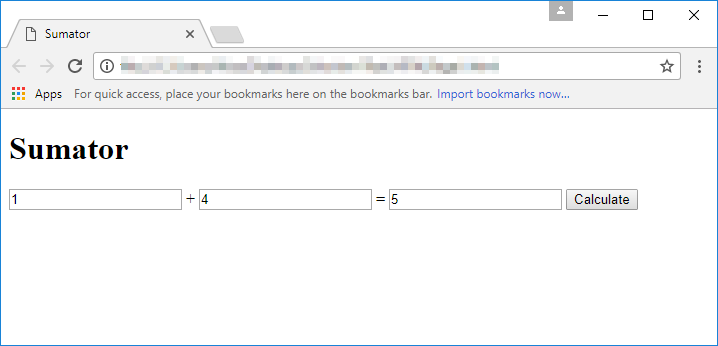
squareOfStars(["4"]);

1. Допишете програмата по-горе, за да отпечатва квадрат, съставен от звездички. Може да се наложи да използвате **for-цикли**. Потърсете информация в интернет.
2. Тествайте решението си в judge системата: <https://judge.softuni.bg/Contests/Practice/Index/150#5>.

# Упражнения: Десктоп и Web приложения

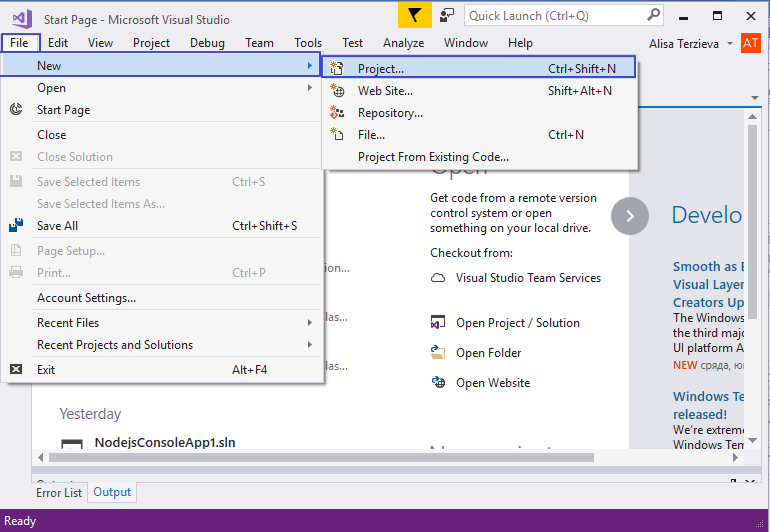
## Уеб приложение „Суматор за числа“

Напишете **уеб приложение**, което изчислява **сумата на две числа**. При въвеждане на две числа в първите две текстови полета и натискане на бутона [Calculate] се изчислява тяхната сума и резултатът се показва в третото текстово поле. Уеб приложението би могло да изглежда по следния начин:

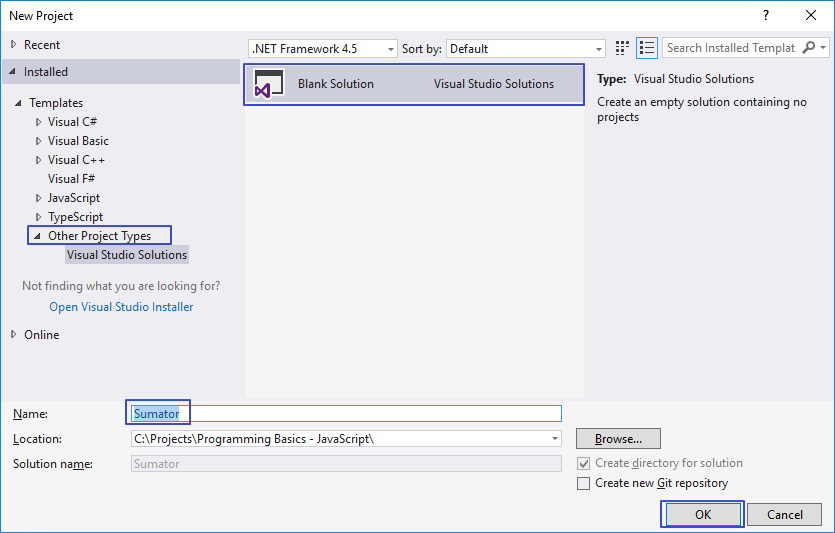


За разлика от конзолните приложения, които четат и пишат данните си във вид на текст на конзолата, **уеб приложения** имат **уеб базиран потребителски интерфейс**. Уеб приложенията се зареждат от някакъв Интернет адрес (URL) чрез стандартен **уеб браузър**. Потребителите пишат входните данни в страница, визуализирана от уеб приложението, данните се обработват и резултатите се показват отново в страницата в уеб браузъра.

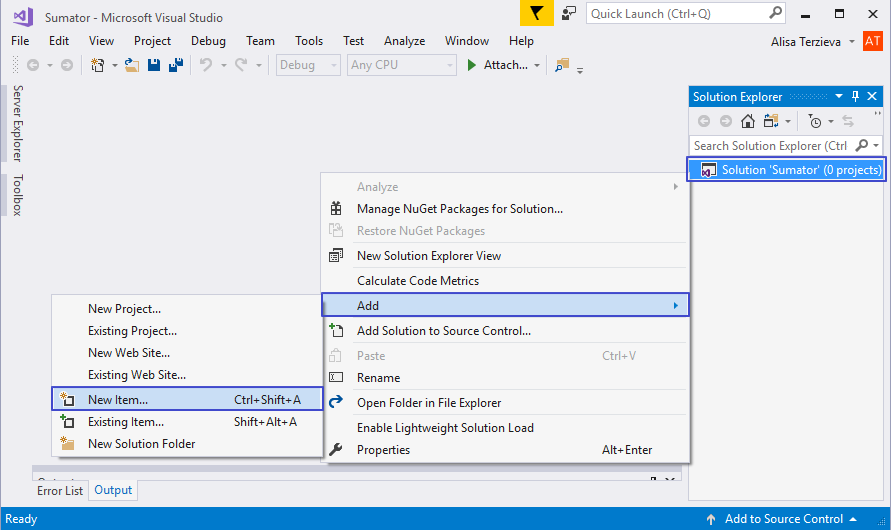
1. Създайте **нов проект** във Visual Studio 2017:



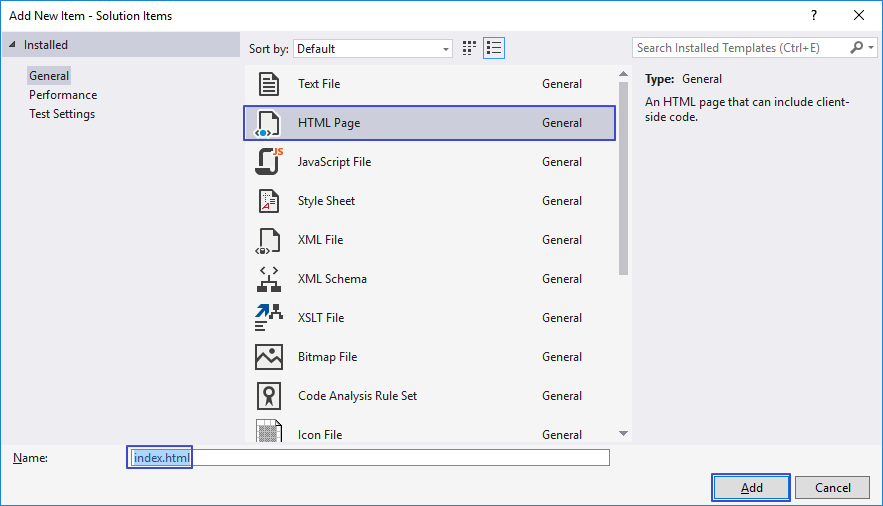
1. Изберете "Blank Solution" и въведете подходящо име на приложението, например "Sumator":



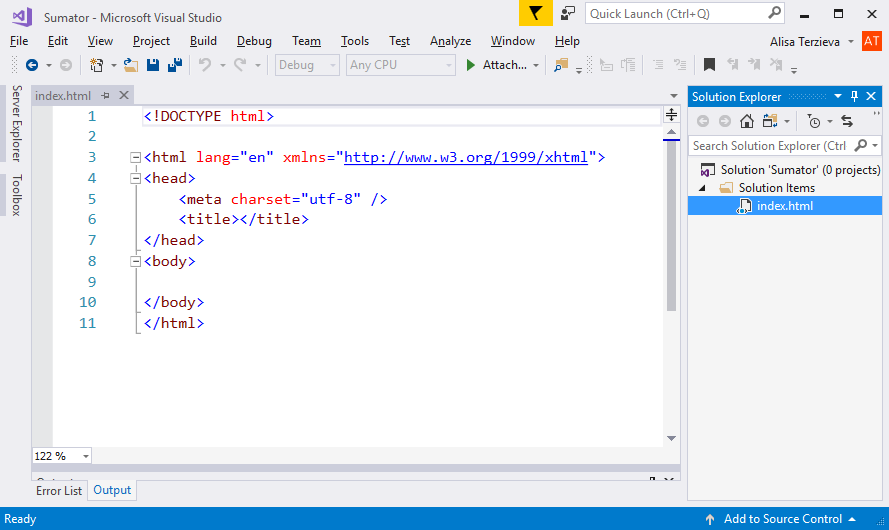
1. Създайте нов файл с клик на десния бутон на мишката върху името на проекта, както е показано на изображението:



1. Изберете "**HTML Page**" и въведете **подходящо име**:



1. До този момент проектът трябва да изглежда по следния начин:



Файлът с разширение "**\*.html**" съдържа елементите, които ще се визуализират в браузъра. Въведете име "**Sumator**" в тага "**<title>**". Напишете кода, който се съдържа в тага "**<body>**":

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head>

<meta charset="utf-8" />

<title>Sumator</title>

</head>

<body>

<h1>Sumator</h1>

<form>

<input type="number" id="num1"/>

<span>+</span>

<input type="number" id="num2" />

<span>=</span>

<input type="number" readonly="readonly" id="sum"/>

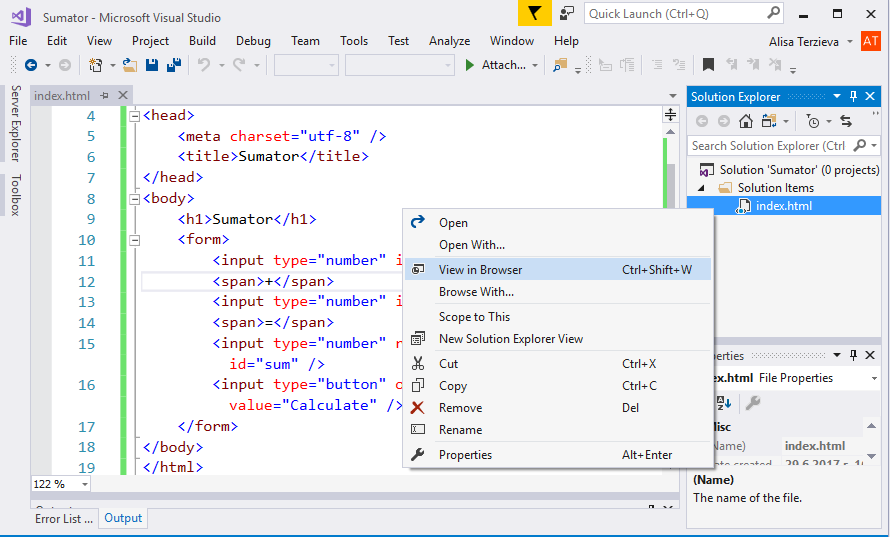
<input type="button" onclick="calc()" value="Calculate"/>

</form>

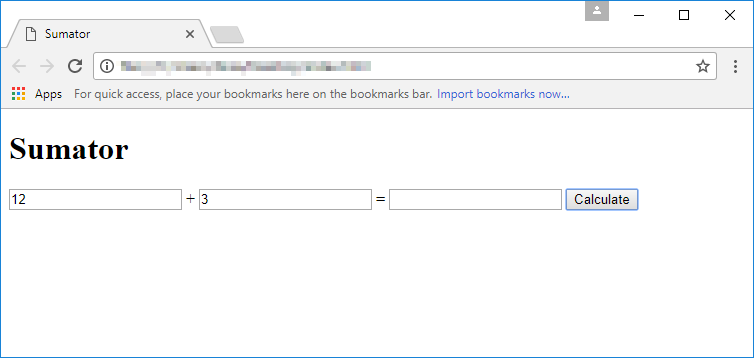
</body>

</html>

1. Стартирайте файла "**index.html**" с клавишната комбинация **Ctrl+Shift+W** или с десен бутон върху файла 🡪 "**View in Browser**":



1. Визуализира се съдържанието на файла, но при натискане на бутона "Calculate" все още не се пресмята сумата на числата. За целта, трябва да напишем функцията calc().



1. Следващата стъпка е създаването на функцията calc(), която извършва сумирането на числата и записва получения резултат в полето "sum":

function calc() {

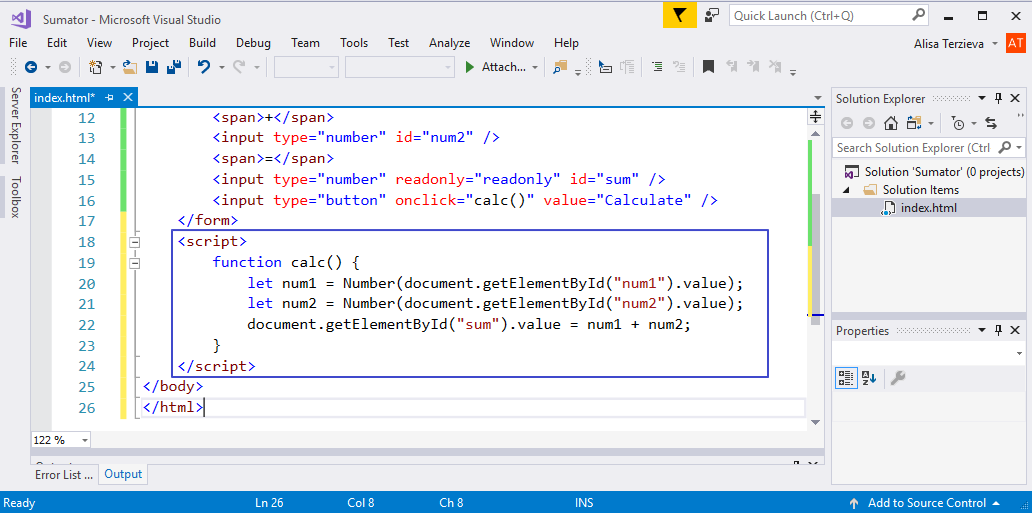
let num1 = Number(document.getElementById("num1").value);

let num2 = Number(document.getElementById("num2").value);

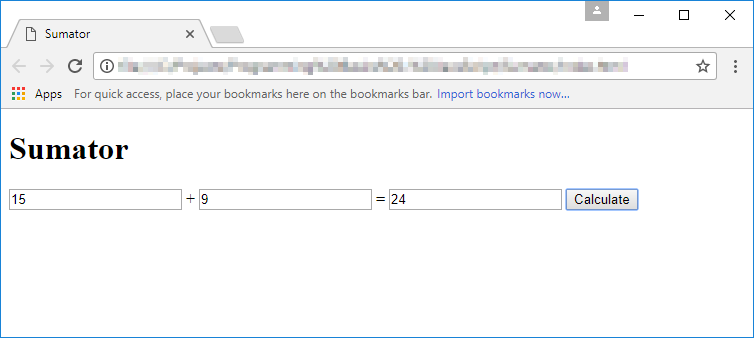
document.getElementById("sum").value = num1 + num2;

}

1. Последната стъпка е закачането на функцията към нашия "**index.html**" файл. Това става като използваме тага "**<script>**", съдържащ функцията. Добавя се точно преди затварящия таг "**</body>**":

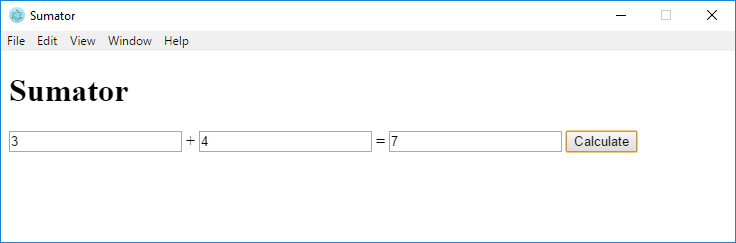


1. Отново стартирайте файла с **Ctrl+Shift+W** и тествайте суматора:



## Десктоп приложение „Суматор за числа“

Напишете **графично (GUI) приложение**, което изчислява **сумата на две числа**:

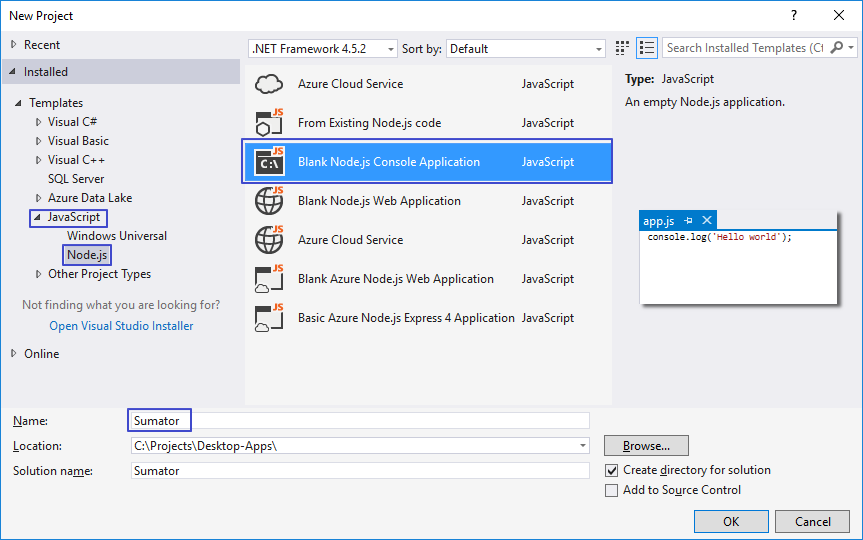


При въвеждане на две числа в първите две текстови полета и натискане на бутона [Calculate] се изчислява тяхната сума и резултатът се показва в третото текстово поле.

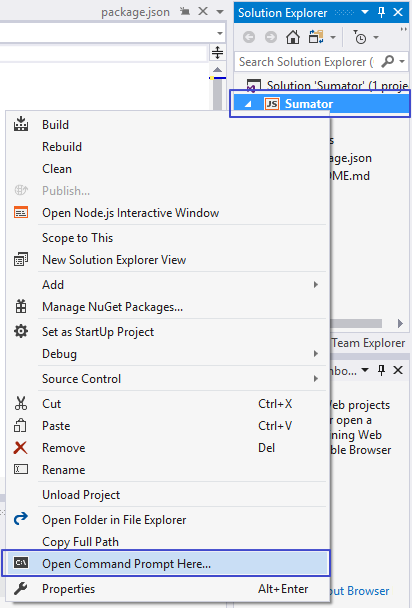
За разлика от конзолните приложения, които четат и пишат данните си във вид на текст на конзолата, **графичните (GUI) приложения** имат визуален потребителски интерфейс. Графичните приложения (настолни приложения, desktop apps) се състоят от един от няколко графични прозореца, в които има контроли: текстови полета, бутони, картинки, таблици и други.

За нашето приложение ще използваме технологията **Electron.js**, която позволява създаване на графични приложения за всички операционни системи с езика за програмиране **JavaScript, HTML и CSS**.

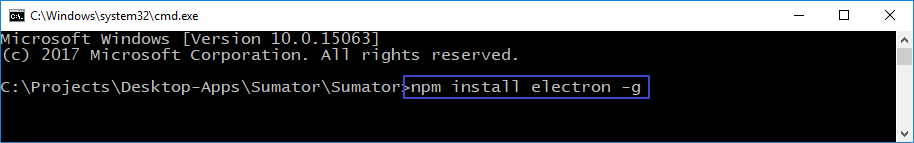
1. Във Visual Studio създайте нов Node.js проект от тип „**Blank Node.js Console Application**“:



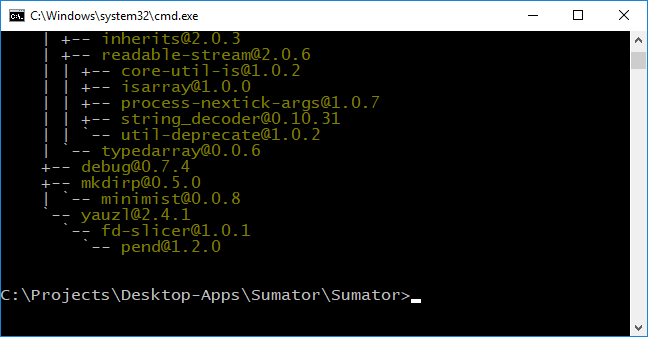
1. Следващата стъпка е инсталиране на платформата Electron.js. За целта стартирайте команден ред (**cmd**) в папката на проекта:



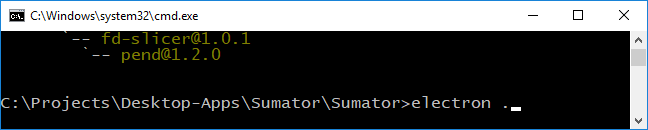
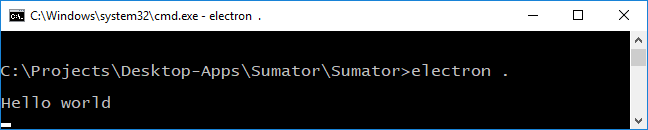
Изпълнете следната команда: "npm install electron -g**"**



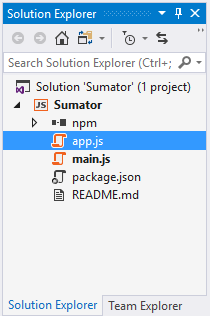
След приключване на инсталацията, прозорецът трябва да изглежда по подобен начин:



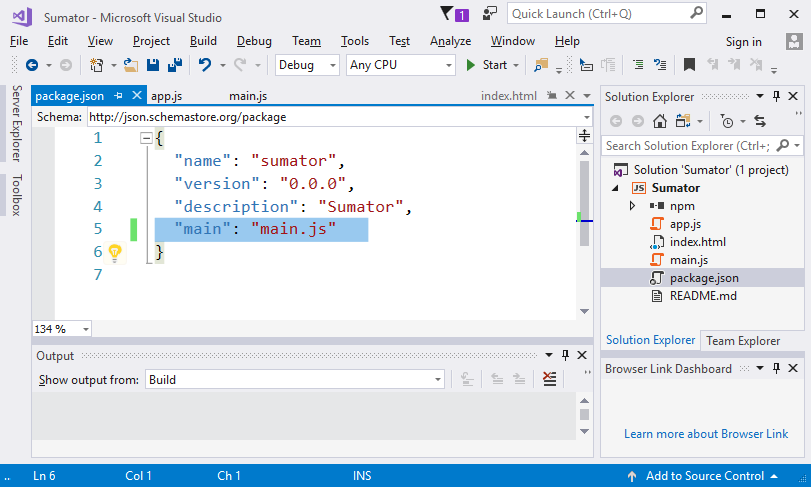
1. Платформата се стартира с командата "electron .**"**.В "**package.json**"за стартов файл на проекта е обозначен скриптът "app.js", който печата в конзолата ред с текст "Hello world". Това означава, че платформата е правилно инсталирана и следващата стъпка е създаването на файл "main.js", който ще бъде отправна точка за стартиране на десктоп приложението.

Структурата на проекта трябва да изглежда по следния начин:



1. В проекта намерете файла "package.json" и променете main скрипта с "main.js":



1. Electron.js стартира от файла "main.js", а в случай, че не е указан стартов файл, стартира от "index.js". Кодът, описан в този файл, управлява събитията и създава нови прозорци в приложението. Трябва да изглежда по следния начин:

const {app, BrowserWindow} = require('electron');

const path = require('path');

const url = require('url');

let win;

function createWindow () {

win = new BrowserWindow({width: 750, height: 250, resizable: false});

win.loadURL(url.format({

pathname: path.join(\_\_dirname, 'index.html'),

protocol: 'file:',

slashes: true

}));

win.on('closed', ()=>{

win = null;

});

}

app.on('ready', createWindow);

app.on('window-all-closed', () => {

app.quit()

})

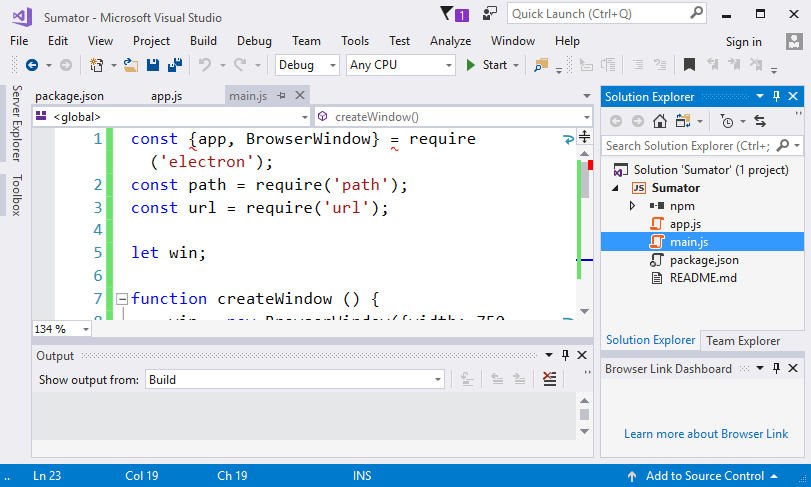
app.on('activate', () => {

if (win === null) {

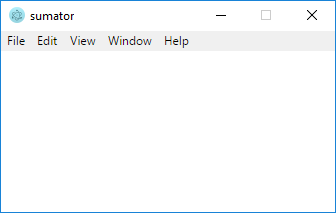
createWindow()

}

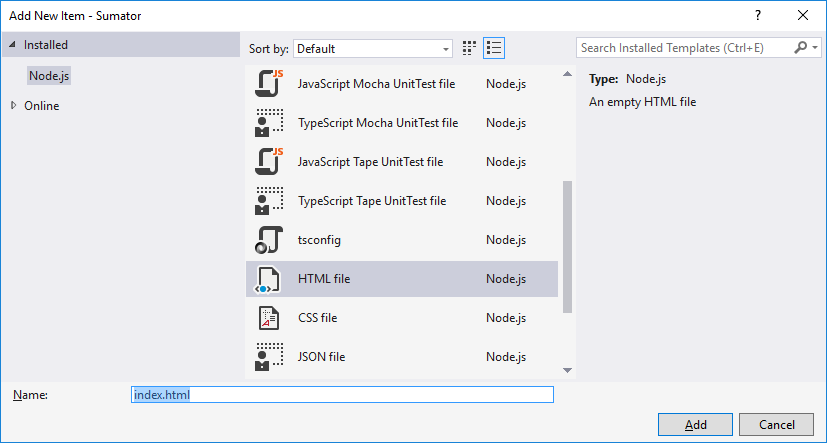
})



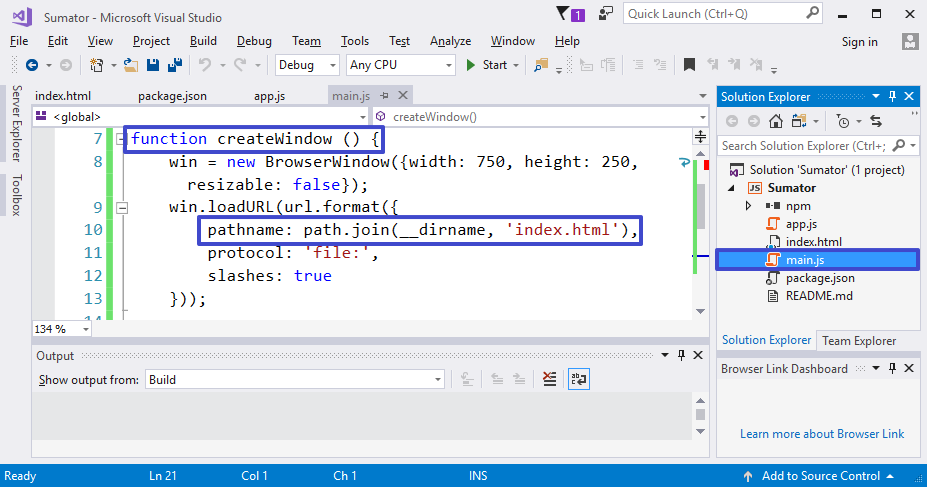
Стартирайте приложението с командата "**electron .**" в командния ред. Тъй като проектът все още не съдържа html файл с елементи за визуализиране, кодът генерира празен прозорец:



1. Създайте нов файл с разшрение \*.html и име "index.html":



Това е файлът, който платформата визуализира в прозореца при стартиране на приложението:



Кодът в "index.html" трябва да изглежда по подобен начин:

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head>

<meta charset="utf-8" />

<title>Sumator</title>

<script src="app.js" type="text/javascript"></script>

</head>

<body>

<h1>Sumator</h1>

<form>

<input type="number" id="num1"/>

<span>+</span>

<input type="number" id="num2" />

<span>=</span>

<input type="number" readonly="readonly" id="sum"/>

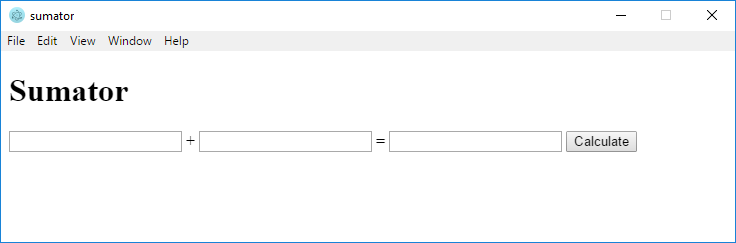
<input type="button" onclick="calc()" value="Calculate"/>

</form>

</body>

</html>

При повторно стартиране на приложението, трябва да се визуализират полетата за въвеждане на числата и бутона за изчисляване на сумата им:



1. Последната стъпка е създаването на функцията calc(), която извършва сумирането на числата и записва получения резултат в полето "sum". Във файла "app.js" напишете следния код:

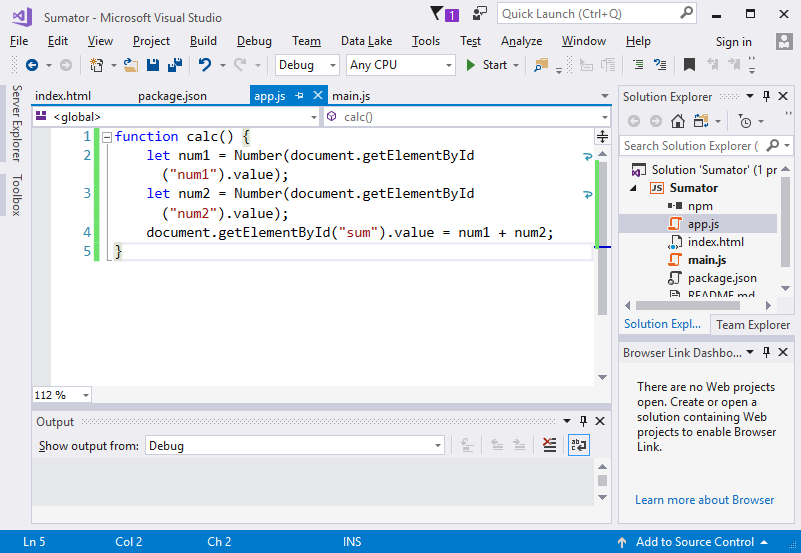
function calc() {

let num1 = Number(document.getElementById("num1").value);

let num2 = Number(document.getElementById("num2").value);

document.getElementById("sum").value = num1 + num2;

}



1. Стартирайте приложението и го тествайте :

