Advanced TypeScript

Goal

In this lab, you will:

- Update a RPN calculator to make the code better
- Use the strict TypeScript settings to do so

Your mission

This is lab you will use TypeScript compiler to detect potential errors in the code and fix them.

A RPN calculator consists of a stack of operands. The mathematical operators work on the last two operands on the stack. After performing the operation, the resulting value is pushed back onto the stack. For example: the formula `2 * (3 + 4)` is entered as `2 Enter 3 Enter 4 + *` and should result in `14`.

The RPN calculator takes input from the browsers DOM and update the DOM after each operation.

Note: You can read more about RPN on Wikipedia.

Type it out by hand?

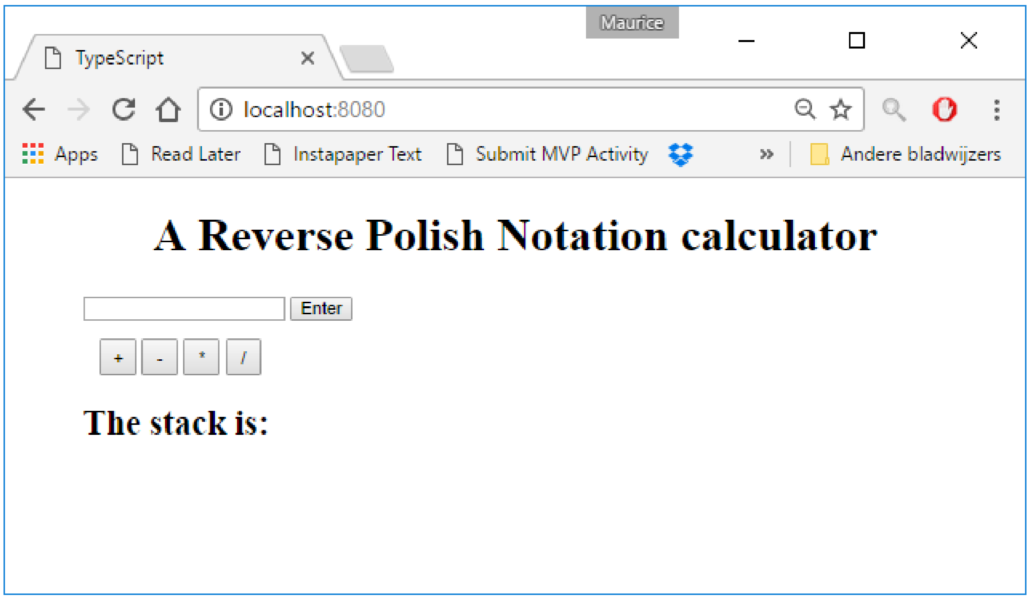
> Typing it drills it into your brain much better than simply copying and pasting it. You're forming new neuron pathways. Those pathways are going to help you in the future. Help them out now.

Compile and run the RPN calculator

In this section you will create the start of a basic RPN calculator class. The RPN calculator will accept inputs from the `<input>` element. It will display the content of the stack using an `<ul>` element.

1. Start the application.
   1. Open the **begin** folder.
   2. Compile the code by running `npm run tsc`. Note: starting the TypeScript compiler in watch using `--watch` makes the following steps easier
   3. Start the web server by running `npm start`.
   4. Try the calculator in the browser. Note that there where no compile errors but there are run-time errors in the console and not all operations work as they should.
2. Prevent any types becoming `any` because the TypeScript compiler can't derive the correct type.

    1. Open **tsconfig.json**
    2. Enable no implicit any mode by setting the compiler option `noImplicitAny` to `true`.
    3. Open **index.ts** and fix all the compile errors



3. Enable strict mode to catch other potential errors.

    1. Open **tsconfig.json** again
    2. Enable strict mode by setting the compiler option `strict` to `true`.
    3. Open **index.ts**.
    4. Change the `numberInput` property to be of type `HTMLInputElement` so the value property is recognized.
    5. Change the `stackElement` property to be of type `HTMLElement | null`.
    6. Fix the compile errors.
    7. Try the calculator in the browser. Note that there where no compile errors and there are no run-time errors in the console. Except for the multiply button all operations should be functional.

4. Fix the last error when multiplying.

    1. Open **index.ts**.
    2. There is a typo in the `btnMult1ply` button. This results in a null value being returned from `this.root.getElementById()` Fix the typo and the multiply button should be fully functional as well.

Solution

The Solution can be found in **complete** folder