

Lab 2

الاسم/ محمود علي أحمد علي غلاب

الرقم الجامعي/20011811

Problem Statement:

Build a web-based calculator similar to that of windows. The buttons should be web buttons.(Design a Calculator with Angular and Spring).

- 1- No need for fancy styling.
- 2- Calculation should be done on the server side.
- 3- For simplicity, you can ignore the difference between the C and CE buttons.
- 4- Repeating pressing the = button does not issue new calculations
Handle exceptions such as dividing by 0, by displaying an E.

Frameworks & technology Used:

Angular & SpringBoot.

Explanation about how to use the app:

1) Frontend part, in angular we can run the project on visual studio code editor by these steps:

- 1- select new terminal from Terminal bar.
- 2- reaching the project directory by few “cd” commands
- 3- then ng serve to run the project into the server.

2) Backend part, I have made 3 java classes:

- 1) Lab2CalculatorApplication : The main class is here (i.e. start running from it) and this is the .
- 2) Lab2CalculatorService: methods implementation is here.
- 3) Lab2CalculatorController: All the requests to the expression we need to evaluate from frontend is here.

So, I used the MVC pattern in backend for this project.


1-Model: The service class.

2-View: The frontend class.

3-Controller: Controller class.

Then, Copy this link ("http://localhost:4200/") on chrome and start using the app.

FrontEnd design By CSS&HTML:

0			
%	CE	C	
$1/x$	x^2	$\sqrt{}$	\div
7	8	9	\times
4	5	6	$-$
1	2	3	$+$
$\pm/_$	0	.	$=$

FrontEnd some Typescript functions:

1- Clear():

```

clear() {
  this.prev_oper = '';
  this.oper = '';
  this.sing_opr = '';
  this.curr_oper = '0';
  this.result = '';
  this.equalpressed = false;
  this.single_oper = false;
  this.operand_press = false;
  this.eq_sign = '';
  this.num_pressed = false;
}

```

2- append_num():

```

}
append_num(val: string) {
  this.num_pressed = true;
  if (this.curr_oper.toString().includes('.') && val === '.') {
    return;
  }
  if (this.equalpressed) {
    this.clear();
  }
  if (this.curr_oper == '0' && val == '0') {
    return;
  }
  if (
    this.curr_oper == '0' ||
    this.curr_oper == 'E' ||
    this.operand_press ||
    this.single_oper
  ) {
    this.curr_oper = val;
    this.operand_press = false;
    this.single_oper = false;
    this.equalpressed = false;
    //this.prev_oper = "";
  } else {
    this.curr_oper += val;
    this.equalpressed = false;
  }
}
}

```

3- del():

```

del() {
  if (this.equalpressed /*&& this.operand_press && this.single_oper*/) {
    this.equalpressed = false;
    this.eq_sign = '';
    this.prev_oper = '';
    this.oper = '';
    this.sing_opr = '';
    this.result = '';
    this.operand_press = true;
  }
  if (this.curr_oper === '0' || !this.num_pressed) {
    return;
  } else {
    if (this.curr_oper.length === 1) {
      this.curr_oper = '0';
    } else if (this.equalpressed) {
      this.result = '';
    } else {
      this.curr_oper = this.curr_oper.slice(0, -1);
    }
  }
}
}

```

4- calculate():

```

    calculate() {
        if (this.curr_oper == 'E') {
            if(this.oper)
            {
                this.result=this.prev_oper+" "+this.oper;
            }
            return;
        }
        else if(this.oper=='&&this.prev_oper=='')
        {
            if(this.curr_oper!='E')
            {
                this.result=this.curr_oper;
            }
            this.eq_sign='';
        }
        if (!this.equalpressed && this.prev_oper) {
            this.sendeqn(this.oper, this.prev_oper, this.curr_oper);
            this.equalpressed = true;
            this.result = this.prev_oper + ' ' + this.oper + ' ' + this.curr_oper;
            this.eq_sign = '=';
        } else if (!this.prev_oper) {
            this.eq_sign = '=';
            this.equalpressed = true;
        }
    }
}
}

```

BackEnd Controller Code:

```

CalculatorApplication.java  application.properties  Lab2CalculatorController.java  Lab2CalculatorService.java
package com.example.lab2.calculator;

import ...

@RestController
@CrossOrigin(origins = "http://localhost:4200")
public class Lab2CalculatorController {
    @PostMapping(value = "/operate/{oper}/{prev}/{curr}")
    public String calc(@PathVariable("oper") String operator, @PathVariable("prev") String prev_op, @PathVariable("curr") String curr_op) {
        Lab2CalculatorService calculator = new Lab2CalculatorService();
        return calculator.handle(operator, prev_op, curr_op);
    }
}

```

Sample runs:

1) Add:

2 + 3 =

5

%	CE	C	
$1/x$	x^2	$\sqrt{}$	\div
7	8	9	\times
4	5	6	$-$
1	2	3	$+$
$\pm/_$	0	.	=

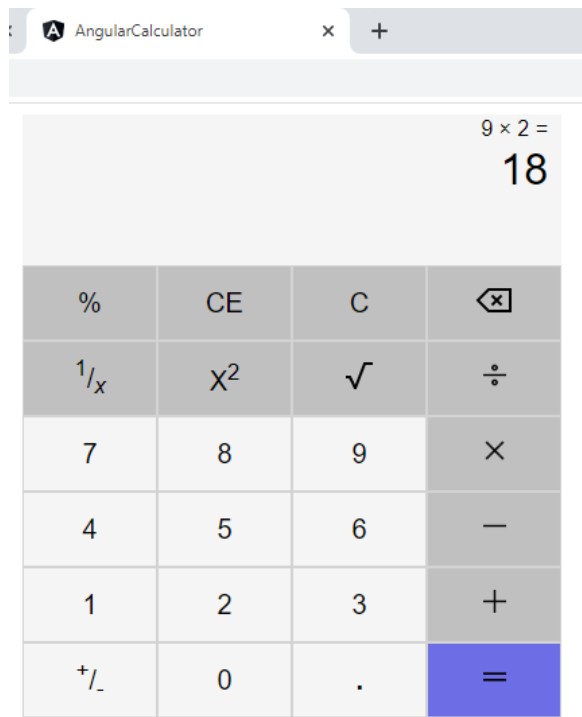
2) Sub:

9 - 3 =

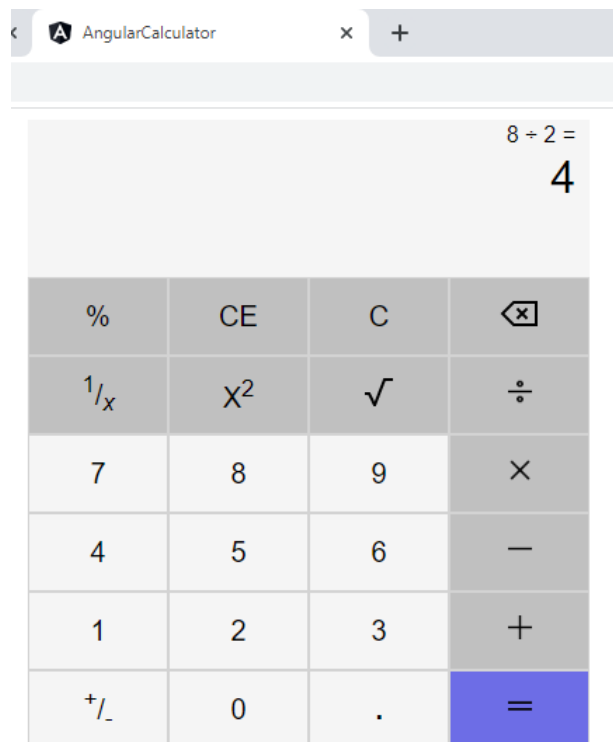
6

%	CE	C	
$1/x$	x^2	$\sqrt{}$	\div
7	8	9	\times
4	5	6	$-$
1	2	3	$+$
$\pm/_$	0	.	=

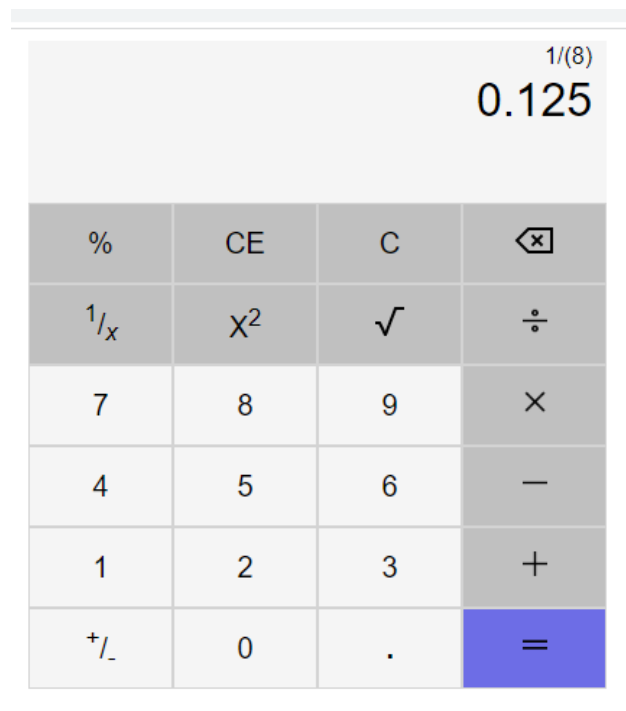
3)Mul:



4)Division:



5)Single operation added to Single one:



0.125 +sqrt(2)			
1.4142135623730951			
%	CE	C	
$1/x$	x^2	$\sqrt{}$	\div
7	8	9	\times
4	5	6	$-$
1	2	3	$+$
$\pm/_$	0	.	=

0.125 + 1.4142135623730951 =			
1.5392135623730951			
%	CE	C	
$1/x$	x^2	$\sqrt{}$	\div
7	8	9	\times
4	5	6	$-$
1	2	3	$+$
$\pm/_$	0	.	=

Handling Corner cases:


1)Division by 0:

6 div 0 =			
E			
%	CE	C	
$1/x$	x^2	$\sqrt{}$	\div
7	8	9	\times
4	5	6	$-$
1	2	3	$+$
$\pm/_$	0	.	=

2)fraction of 0:

$1/(0)$			
E			
%	CE	C	
$1/x$	x^2	$\sqrt{}$	\div
7	8	9	\times
4	5	6	$-$
1	2	3	$+$
$\pm/_$	0	.	=

3)negative roots:

sqrt(-3) E			
%	CE	C	
$1/x$	x^2	$\sqrt{}$	\div
7	8	9	\times
4	5	6	$-$
1	2	3	$+$
$\pm/_$	0	.	=

Assumptions:


1-Assume propagated calculations performed only after pressing on equal button.

2-The result printed on the screen shows the previous operand and the operator and the current one but when he click on any operator button the screen shows only the value of that operation.

For example:

The expression $(1/(8)+\text{sqrt}(2))$

Will be on multiple steps:

0.125 + 1.4142135623730951 =			
1.5392135623730951			
%	CE	C	
$1/x$	x^2	$\sqrt{}$	\div
7	8	9	\times
4	5	6	$-$
1	2	3	$+$
$\pm/_$	0	.	=

3-multiple clicks on equal doesn't issue new calculations.