Universidad Autónoma De Baja California

Facultad de ingeniería

Licenciatura en Sistemas Computacionales

Aplicaciones Web Con Base de Datos

Primer parcial: Examen práctico - Calculadora Angular

Catedrático

Rosendo Rafael Sosa Canales

Alumno

Carlos Francisco Espinoza Rivera

Mexicali, Baja California 9 de abril del 2022

Introducción

Las siguientes capturas de pantalla, representan los pasos que se han llevado a cabo para realizar la calculadora en framework Angular.

Instrucciones:
- **Paso 1:** Instalación de Angular CLI
- **Paso 2:** Inicializando nuestro Proyecto
- **Paso 3:** agregar nuestra plantilla HTML y estilos

Retos:
- **Paso 4:** Escuchar eventos de clic en los botones y obtener sus valores asociados.
- **Paso 7:** Visualización del valor de las variables en la plantilla.

Una vez que tenemos este proyecto los retos para aprobar este examen son los siguientes:

1. Investigar cómo utilizar el enlace de eventos para vincular estos métodos a la plantilla (ver referencias).

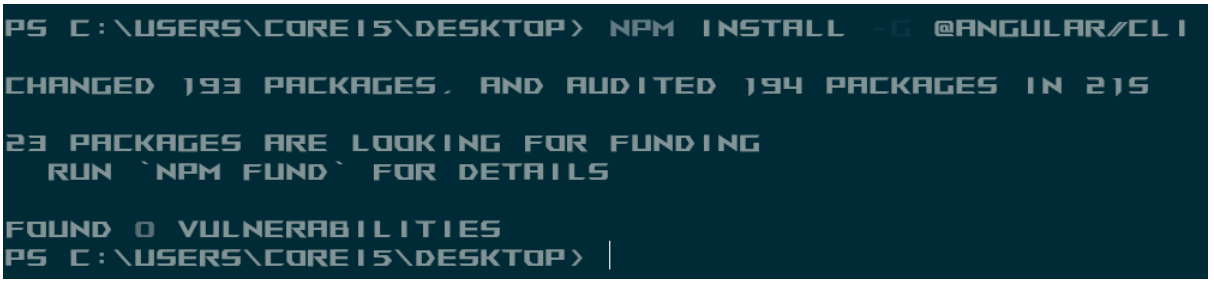
2. Explicar el funcionamiento lógico de los métodos anteriores y cómo funciona el enlace de variables así como los métodos. [Documento word o presentación].

3. Subir a GIT su versión, se toma en cuenta que se personalice.

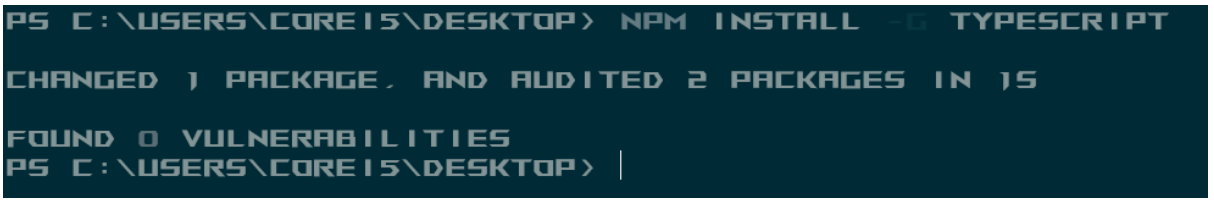
Referencias:
https://angular.io/guide/event-binding
https://angular.io/guide/property-binding
https://angular.io/guide/two-way-binding

Desarrollo

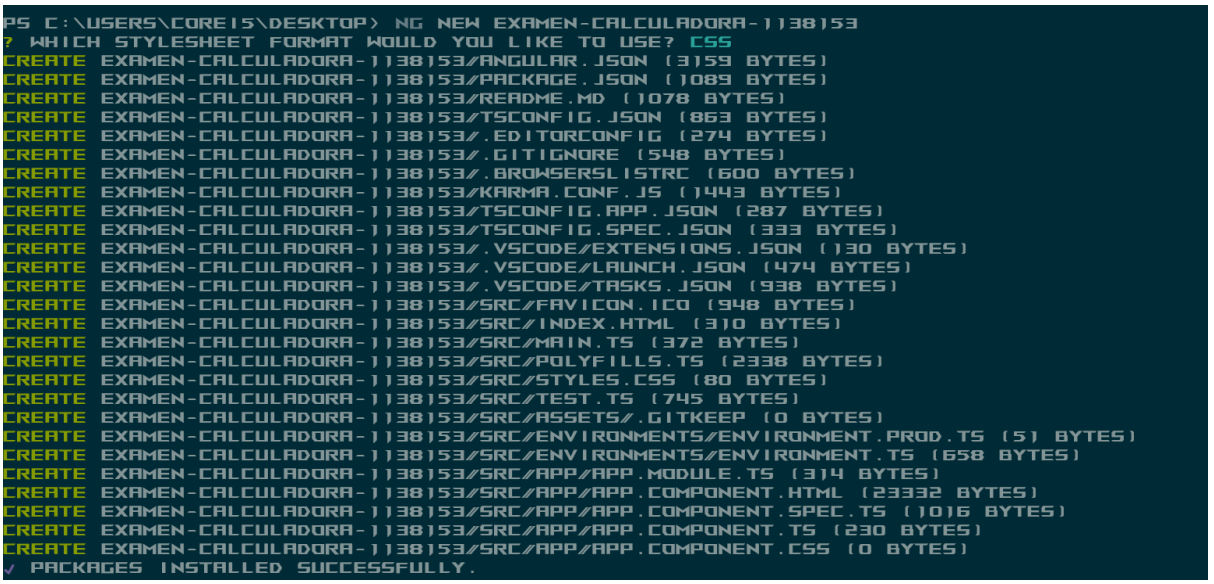
1. Debemos instalar Angular en nuestro equipo.
   comando: npm install -g @angular/cli



2. Debemos instalar TypeScript con el siguiente comando: npm install - g typescript



3. Cree mi proyecto con el siguiente comando ng new examen-calculadora-1138153.

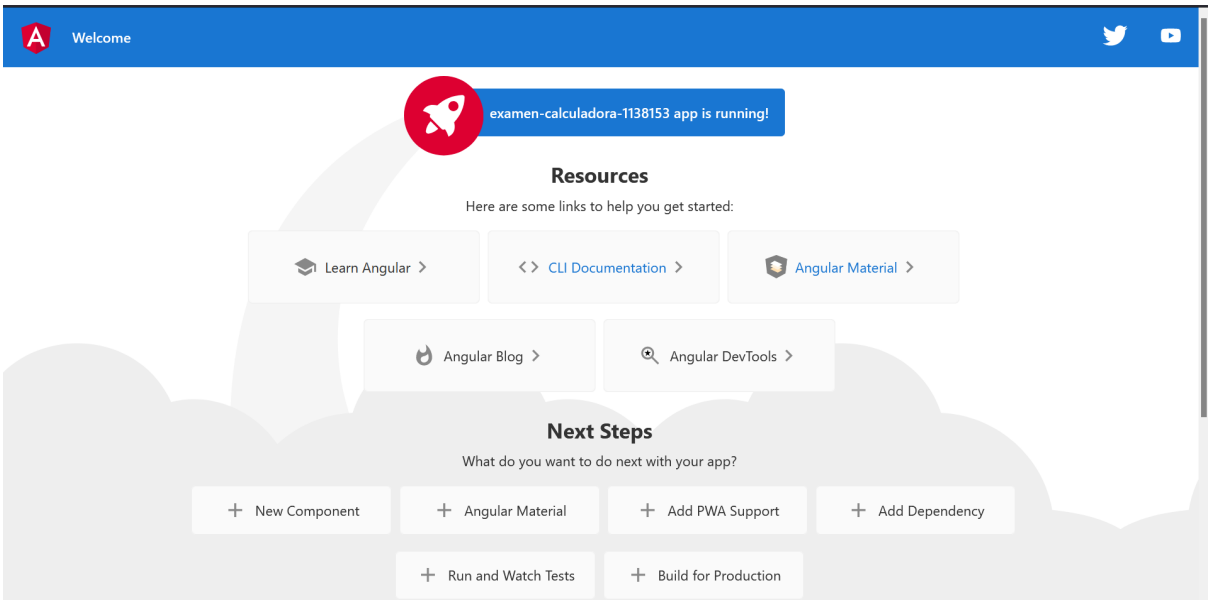


4. Debemos acceder al proyecto que hemos creado.

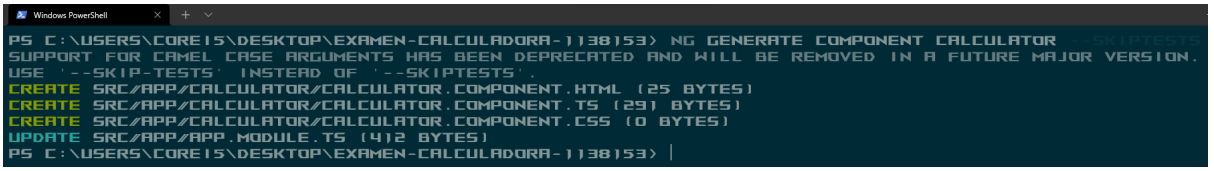
5. Para confirmar que nuestro proyecto se ha creado con todos los paquetes y complementos, debemos levantar nuestro servidor.

Nota: Nuestro puerto local será el 4200.

Debemos de ver una pagina igual a esta:



6. Ahora debemos agregar nuestra plantilla HTML y los estilos css.



7. En en el archivo src/app/calculator/calculator.component.html agregar el contenido que aparece en el siguiente link: https://gist.github.com/Xhendor/98fc2d9210a026e58edb23af8d0ea8cd

8. Nuestra modificación debe verse de la siguiente manera:

calculator.component.html U

src > app > calculator > calculator.component.html > div.calculator

```html
<div class="calculator">

    <input type="text" class="calculator-screen" value="0" disabled />

    <div class="calculator-keys">

      <button type="button" class="operator" value="+">+</button>
      <button type="button" class="operator" value="-">-</button>
      <button type="button" class="operator" value="*">&times;</button>
      <button type="button" class="operator" value="/">&divide;</button>

      <button type="button" value="7">7</button>
      <button type="button" value="8">8</button>
      <button type="button" value="9">9</button>


      <button type="button" value="4">4</button>
      <button type="button" value="5">5</button>
      <button type="button" value="6">6</button>


      <button type="button" value="1">1</button>
      <button type="button" value="2">2</button>
      <button type="button" value="3">3</button>


      <button type="button" value="0">0</button>
      <button type="button" class="decimal" value=".">.</button>
      <button type="button" class="all-clear" value="all-clear">AC</button>

      <button type="button" class="equal-sign" value="=">=</button>

    </div>
</div>
```

9. En en el archivo src/app/calculator/calculator.component.css agregar este contenido:https://gist.github.com/Xhendor/69c1908dd01c4e6b676033a6bff12325

calculator.component.html U | calculator.component.css U

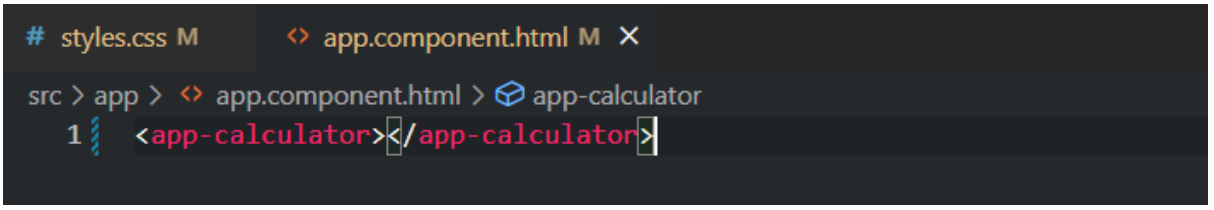
src > app > calculator > calculator.component.css > .calculator-keys

```css
.calculator {
    border: 1px solid #ccc;
    border-radius: 5px;
    position: absolute;
    top: 50%;
    left: 50%;
    transform: translate(-50%, -50%);
    width: 400px;
}

.calculator-screen {
    width: 100%;
    font-size: 5rem;
    height: 80px;
    border: none;
    background-color: #252525;
    color: #fff;
    text-align: right;
    padding-right: 20px;
    padding-left: 10px;
}

button {
    height: 60px;
    background-color: #fff;
    border-radius: 3px;
    border: 1px solid #c4c4c4;
    background-color: transparent;
    font-size: 2rem;
    color: #333;
    background-image: linear-gradient(to bottom,transparent,transparent 50%,rgba(0,0,0,.04));
    box-shadow: inset 0 0 0 1px rgba(255,255,255,.05), inset 0 1px 0 0 rgba(255,255,255,.45), inset 0 -1px 0 0 rgba(255,255,255,.15), 0 1px 0 0 rgb
    text-shadow: 0 1px rgba(255,255,255,.4);
}

button:hover {
    background-color: #eaeaea;
}

.operator {
```

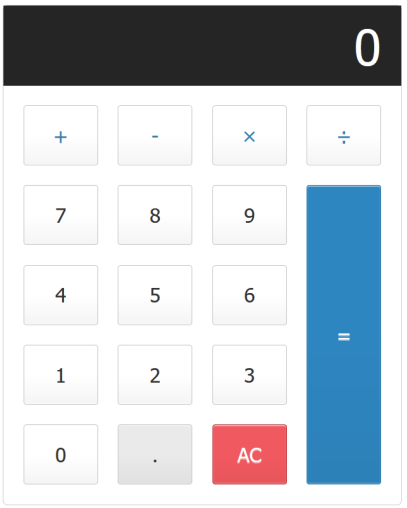
10. En el archivo de estilos global src/styles.css se agrega lo siguiente:

```
html {
    font-size: 62.5%;
    box-sizing: border-box;
  }

*, *::before, *::after {
    margin: 0;
    padding: 0;
    box-sizing: inherit;
}
```

11. Al finalizar de realizar lo anterior en el archivo src/app/app.component.html borrar el contenido y agregar solo lo siguiente:



12. Si recargamos nuestra página podremos ver los cambios que se han realizado.



13. Ahora debemos programar los eventos de escucha para los botones que tenemos en la plantilla, podemos ver que tenemos el siguiente patrón:
dígitos (0-9),
operadores (+, -, *, /, =),
un punto decimal (.)
y una tecla de reinicio.

14. Abriendo el archivo que se encuentra en la siguiente ruta: src/app/calculator/calculator.component.ts debemos definir los siguientes atributos.

```
currentNumber = '0';
firstOperand:any|null = null;
operator:any|null = null;
waitForSecondNumber = false;
```

Estos atributos se traducen de la siguiente manera:

- La variable **currentNumber** contiene la cadena que se mostrará en el elemento de entrada de resultados.
- La variable **firstOperand** contiene el valor del primer operando de la operación.
- La variable del **operator** contiene la operación.
- La variable **waitForSecondNumber** contiene un valor booleano que indica si el usuario ha terminado de escribir el primer operando y está listo para ingresar el segundo operando de la operación.

15. Los metodos para trabajar se encuentran en el siguiente link: https://gist.github.com/Xhendor/b733201693aaaa0158c75907d7a1f835

```
import { Component, OnInit } from '@angular/core';

@Component({
  selector: 'app-calculator',
  templateUrl: './calculator.component.html',
  styleUrls: ['./calculator.component.css']
})
export class CalculatorComponent implements OnInit {

  constructor() { }
  currentNumber = '0';
  firstOperand : any | null = null;
  operator: any|null = null;
  waitForSecondNumber = false;


  ngOnInit(): void {
  }
  public getNumber(v: string){
    console.log(v);
    if(this.waitForSecondNumber)
    {
      this.currentNumber = v;
      this.waitForSecondNumber = false;
    }else{
      this.currentNumber === '0'? this.currentNumber = v: this.currentNumber += v;

    }
  }


  getDecimal(){
    if(!this.currentNumber.includes('.')){
        this.currentNumber += '.';
    }
  }
```

Métodos:
getNumber() que se utilizará para obtener el número actual.
getDecimal() que agrega el punto decimal al número actual.
doCalculation() que realiza el cálculo según el tipo de operador.
getOperation() que se usará para obtener la operación realizada.
clear() que se utilizará para borrar el área de resultados y restablecer los cálculos.

16. Los métodos que se han colocado dependen de los atributos, es necesario corregir los errores que aparecen:

```
private doCalculation(op:any,secondOp:any){
 switch (op){
   case '+':
   return this.firstOperand += secondOp;
   case '-':
   return this.firstOperand -= secondOp;
   case '*':
   return this.firstOperand *= secondOp;
   case '/':
   return this.firstOperand /= secondOp;
   case '=':
   return secondOp;
  }
}
```

17. Ahora debemos de poder llamar eventos al momento de presionar los botones.

```
<div class="calculator">

  <input type="text" class="calculator-screen" value="0" disabled />

  <div class="calculator-keys">

    <button type="button" class="operator" value="+">+</button>
    <button type="button" class="operator" value="-">-</button>
    <button type="button" class="operator" value="*">&times;</button>
    <button type="button" class="operator" value="/">&divide;</button>

    <button type="button" value="7">7</button>
    <button type="button" value="8">8</button>
    <button type="button" value="9">9</button>


    <button type="button" value="4">4</button>
    <button type="button" value="5">5</button>
    <button type="button" value="6">6</button>


    <button type="button" value="1">1</
    <button type="button" value="2">2</
    <button type="button" value="3">3</button>


    <button type="button" value="0">0</button>
    <button type="button" class="decimal" value=".">.</button>
    <button type="button" class="all-clear" value="all-clear">AC</button>

    <button type="button" class="equal-sign" value="=">=</button>

  </div>
</div>
```

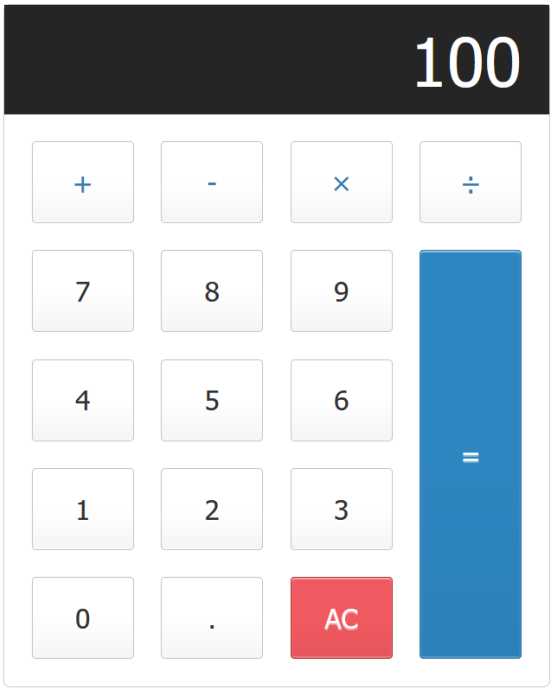
The button element represents a button labeled by its contents.
MDN Reference

Nota: Este componente se encuentra en src/app/calculator/calculatorComponent0.html

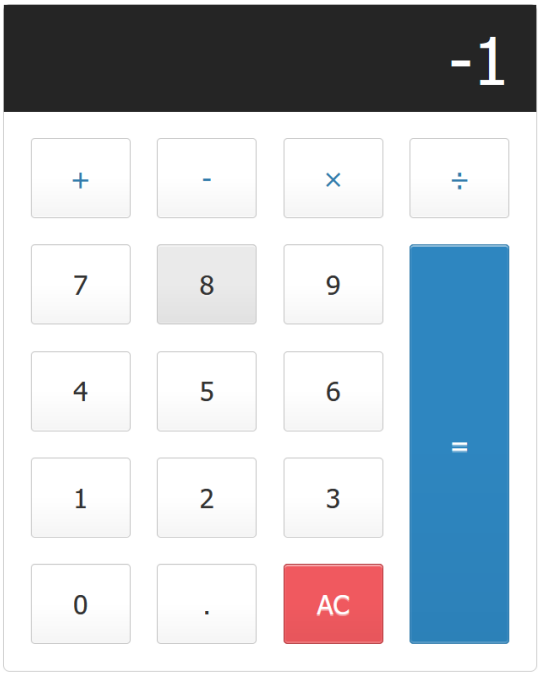
```
<div class="calculator">


  <input type="text" class="calculator-screen" [value]="currentNumber" disabled />

  <div class="calculator-keys">

    <button type="button" (click)="getOperation('+')" class="operator" value="+">+</button>
    <button type="button" (click)="getOperation('-')" class="operator" value="-">-</button>
    <button type="button" (click)="getOperation('*')" class="operator" value="*">&times;</button>
    <button type="button" (click)="getOperation('/')" class="operator" value="/">&divide;</button>

    <button type="button" (click)="getNumber('7')" value="7">7</button>
    <button type="button" (click)="getNumber('8')" value="8">8</button>
    <button type="button" (click)="getNumber('9')" value="9">9</button>


    <button type="button" (click)="getNumber('4')" value="4">4</button>
    <button type="button" (click)="getNumber('5')" value="5">5</button>
    <button type="button" (click)="getNumber('6')" value="6">6</button>


    <button type="button" (click)="getNumber('1')" value="1">1</button>
    <button type="button" (click)="getNumber('2')" value="2">2</button>
    <button type="button" (click)="getNumber('3')" value="3">3</button>


    <button type="button" (click)="getNumber('0')" value="0">0</button>
    <button type="button" (click)="getDecimal()" class="decimal" value=".">.</button>
    <button type="button" (click)="clear()" class="all-clear" value="all-clear">AC</button>

    <button type="button" (click) = "getOperation('=')" class="equal-sign" value="=">=</button>

  </div>
```

De esta manera utilizamos button(click) = “método” para poder llamar métodos que se encuentran .calculator.component.ts.

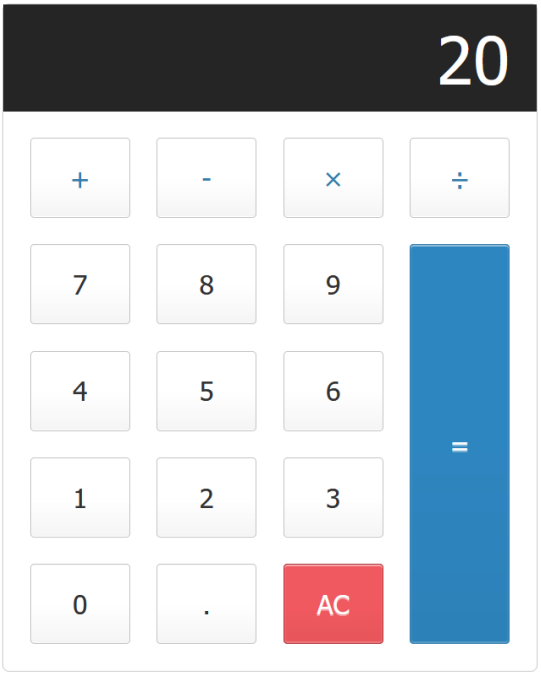
Realizaremos pruebas para verificar el funcionamiento.Al realizar la suma de 50+50 nos dará el siguiente resultado:



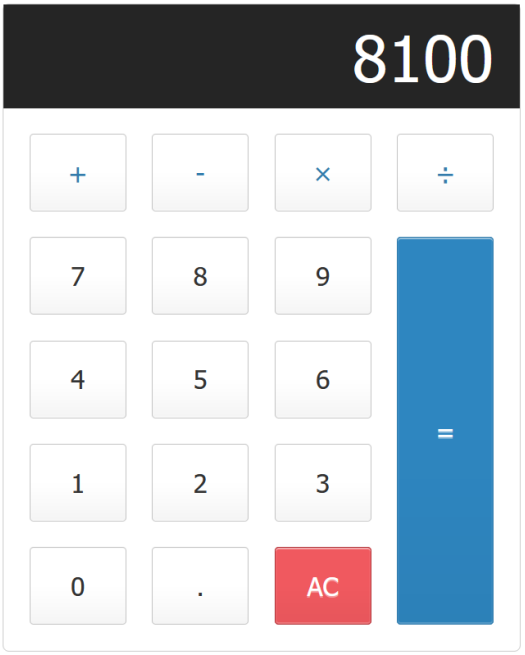
Al realizar una resta de 40-41 nos dará el siguiente resultado:



Para la división utilizaremos los siguientes datos 100/5



Para la multiplicación usaremos los siguientes datos 90 x 90



También podemos usar decimales:

Investigar cómo utilizar el enlace de eventos para vincular estos métodos a la plantilla (ver referencias).

De la siguiente manera podemos mandar a llamar eventos, los cuales en este caso se encuentran en nuestro archivo ts.
<button class="btn btn-primary" (click)="clickAddTodo()">Add Todo</button> al ser presionado mandar a llamar nuestros eventos, para ello debemos colocar nuestro método en el click. Además en estos metodos tambien podemos mandar parametros de la siguiente forma: <button type="button" (click)="getOperation('+')" class="operator" value="+">+</button> en donde el símbolo '+' es nuestro parámetro para el método getOperation.

Explicar el funcionamiento lógico de los métodos anteriores y cómo funciona el enlace de variables así como los métodos. [Documento word o presentación].

Nuestros métodos son llamados directamente desde el archivo calculator.components.ts y de esta forma se pueden ejecutar al presionar un botón.

Por ejemplo, el método **getNumber()** nos proporciona el primer número que el usuario presiona, para luego esperar el segundo número. De esta forma podemos seguir ingresando números sin que estos se sustituyan pues comenzamos a concatenar una vez nuestro **waitForSecondNumber** sea verdadero.

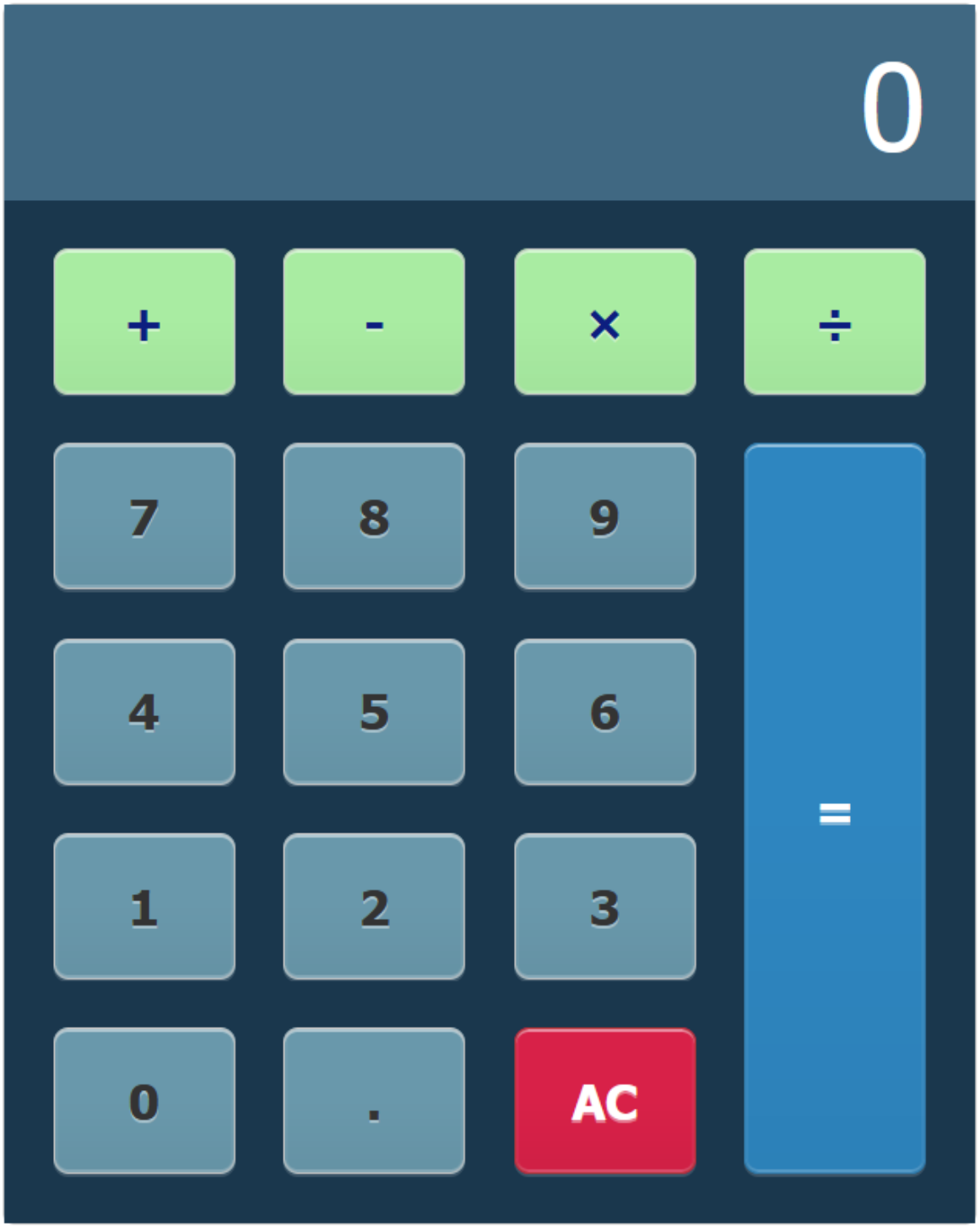
getDecimal() nos retorna un punto decimal cuando este es invocado desde la parte visual por medio de un botón, ya que cuando nosotros presionamos este botón enviamos por parámetro un punto decimal. De este modo, si nuestro parámetro sea diferente de un punto decimal, este se concatenan con la cadena que ya hemos enviado.

doCalculation() realiza las operaciones por medio de los dos parámetros y concatena los números que vayamos ingresando.

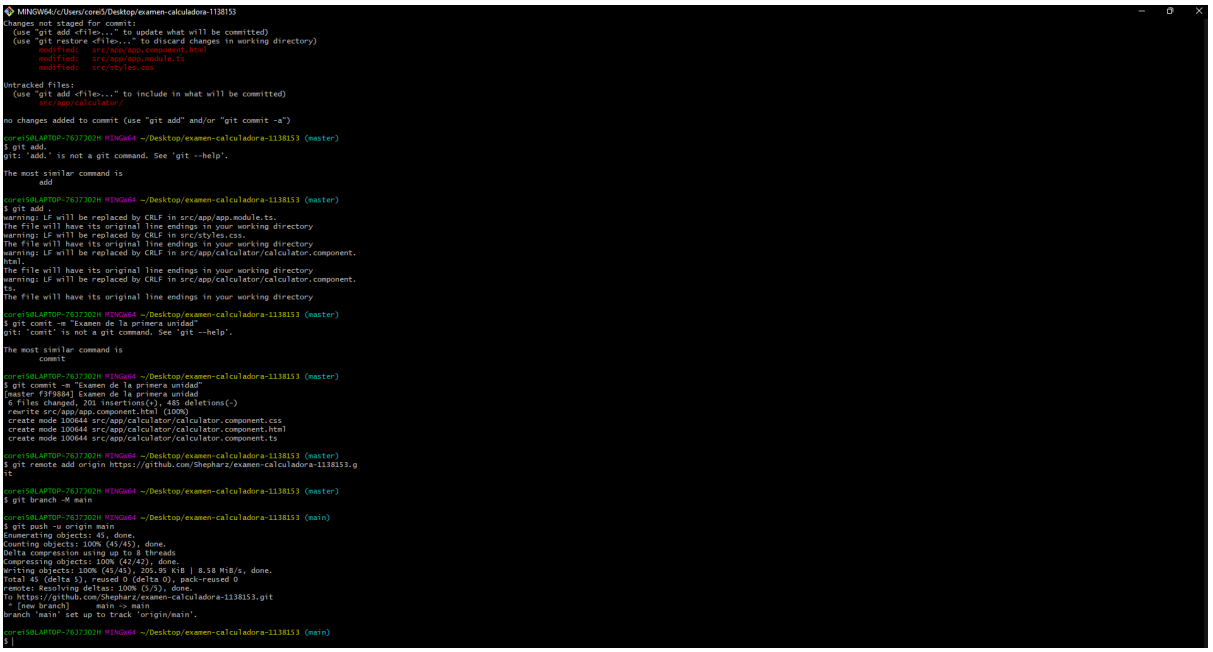
getOperation realiza las operaciones por medio de un switch que nos ayuda a entrar la las diferentes operaciones, de nuevo este método se activa al presionar un botón y al enviar un parametro, ya sea como suma +, resta -, division /, o multiplicacion *, etc.

El método **clear()** nos ayuda a limpiar nuestra pantalla al momento de presionar nuestro botón AC, reinicia nuestras variables al valor por defecto.

Le di un estilo diferente a mi calculadora, y quedó de la siguiente manera:



Utilizando gitHub para subir nuestro repositorio:



Link: https://github.com/Shepharz/examen-calculadora-1138153

Conclusiones:

Aprendí el uso de los eventos por medio de botones, y reforcé lo que he aprendido en clase sobre TypeScript, sin embargo considero que aun me hace falta mucho para poder manejarlo desde cero. Este examen proporciona todos los elementos básicos de cómo realizar métodos con parámetros y con base a ello, entrenar con Angular el cual es un framework que me interesa mucho aprender.