

Instruções para a entrega: mostrar a aplicação para o professor na aula do dia [12/set](#). A entrega pode ser em dupla.

Considere a seguinte aplicação que possui uma calculadora e um log de todas as teclas clicadas.



O projeto é formado pelo componente log (mostra as teclas clicadas) e serviço serviço. Considere os seguintes códigos:

Arquivo: src/app/app.component.ts

```
@Component({
  selector: 'my-app',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  private teclas: string[] = [
    '7', '8', '9', '+',
    '4', '5', '6', '-',
    '1', '2', '3', '*',
    '.', '0', '%', '/'
  ];
  private display: string;

  constructor(
    private servico: ServicoService) {
    this.ce();
  }
}
```

```
addDigito(digito) {
  try{
    /* evitar Octal literals */
    if(Number(this.display) != 0){
      this.display += digito;
    }
    else{
      this.display = digito;
    }
  }catch(e){
    this.display = digito;
    console.log(e.message);
  }
  this.servico.append(
    "<" + digito + ">");
}

ce(): void {
  this.display = "";
  this.servico.append("<CE>");
}

calcular(): void {
  try {
    let r = eval(this.display);
    if (!isNaN(r)) {
      this.display = r;
    }
  } catch (e) {
    console.log(e.message);
  }
  this.servico.append("<=>");
}

isNumber(digito): boolean {
  return !isNaN(Number(digito)) ||
    digito == '.';
}
}
```

Arquivo: src/app/log.component.ts

```
@Component({
  selector: 'app-log',
  templateUrl: './log.component.html',
  styleUrls: ['./log.component.css']
})
export class LogComponent
  implements OnInit {
  constructor(
    private servico: ServicoService) { }

  ngOnInit() {
    this.servico.reset();
  }
}
```

Arquivo: src/app/servico.service.ts

```
@Injectable()
export class ServicoService {
  private logs: string[];
  constructor() {
    this.reset();
  }

  append(tecla): void {
```

```
        this.logs.push(tecla);  
    }  
  
    reset(): void {  
        this.logs = [];  
    }  
}
```

1 – Toda a view do aplicativo está no componente AppComponent. Desta forma, o componente LogComponent é um componente filho. Codificar na view do AppComponent os seguintes itens.

- a) O display da calculadora está vinculado com o atributo `display` do modelo do componente.
- b) O display da calculadora não pode aceitar entradas do teclado, ou seja, deverá estar desabilitado.
- c) Utilize o array `teclas` para criar os botões na view.
- d) Vincular os botões 7, 8, 9, +, 4, 5, 6, -, 1, 2, 3, *, ., 0, % e / com o método `addDigito`.
- e) Os botões 7, 8, 9, +, 4, 5, 6, -, 1, 2, 3, *, ., 0, % e / possuem as classes `btn-secondary` e `btn-danger`, porém os botões que são dígitos devem ter somente a classe `btn-secondary` e aqueles que são operações matemáticas devem ter a classe `btn-danger`. Utilize `class binding` para remover a classe que não se aplica ao botão. Use o método `isNumber` para identificar os botões.
- f) Vincular os botões CE e = (igual) aos métodos `ce` e `calcular`, respectivamente.
- g) Adicionar o LogComponent como componente filho.

2 – Codificar a view do componente log.

- a) Quando o array de logs estiver vazio o componente log não deverá estar visível, isto é, nenhuma marcação poderá estar disponível dentro do componente.
- b) Carregar cada elemento do array logs em uma marcação do tipo `<div>` da HTML.