PROGRAMAÇÃO WEBII

1

Arrays, Objetos, for e *ngFor no Angular

Variáveis

- Vamos para o VsCode.
- Abrir o arquivo listar-produtos.component.ts
- Vamos começar criando uma variável para guardar uma lista de strings. Será nosso primeiro array que iremos criar.
- listaStrings: Após os dois pontos, temos que especificar o tipo da variável, que no caso é string e para que o Angular reconheça que é um array, devemos abrir e fechar colchetes []. Agora precisaremos preencher esse array digitando as informações a seguir: Na primeira posição será igual a Primeiro, na segunda posição será igual a Segundo e na terceira posição será Terceiro. Iremos trabalhar somente com 3 posições. Poderíamos ter mais, mas no momento está bom pra entendermos o conceito. Continuando, iremos digitar: = ['Primeiro', 'Segundo', 'Terceiro']. Ficará assim:

Ficará assim:

```
8  export class ListarProdutosComponent implements OnInit {
9
10    listaStrings: string[] = ['Primeiro', 'Segundo', 'Terceiro'];
11
12    constructor() { }
13
14    ngOnInit(): void {
15    }
16
17  }
```

O TypeScript sabe que é um array de string porque declaramos o tipo string com colchetes. Vamos ter um array com 3 elementos: Na posição zero vamos ter o texto Primeiro, na posição um o texto Segundo e na posição 2 o texto Terceiro.

- Agora já iremos conseguir imprimir o conteúdo desse array.
- Vamos ir dentro do método Construtor
- Vamos começar fazendo um for:
- For (). Dentro do parênteses, vamos usar um let que é uma forma para criarmos variáveis dentro de métodos. Ele tem um escopo local. Let permite criar variável dentro das funções, dentro dos nossos métodos e essas variáveis permitem modificações.
- Vamos digitar let item of this.listaString
- Após isso, vamos imprimir : console.log (item);

Criamos aqui, fizemos o construtor fazendo a impressão dos itens na nossa lista.

```
10
        listaStrings: string[] = ['Primeiro', 'Segundo', 'Terceiro'];
11
12
        constructor() {
13
          for (let item of this.listaStrings) {
            console.log(item);
14
15
16
17
18
        ngOnInit(): void {
19
20
21
```

- Vamos salvar, ir na página
- Pressionar <F12> e verificar se está exibindo o conteúdo dos arrays.
- Ir na opção Console



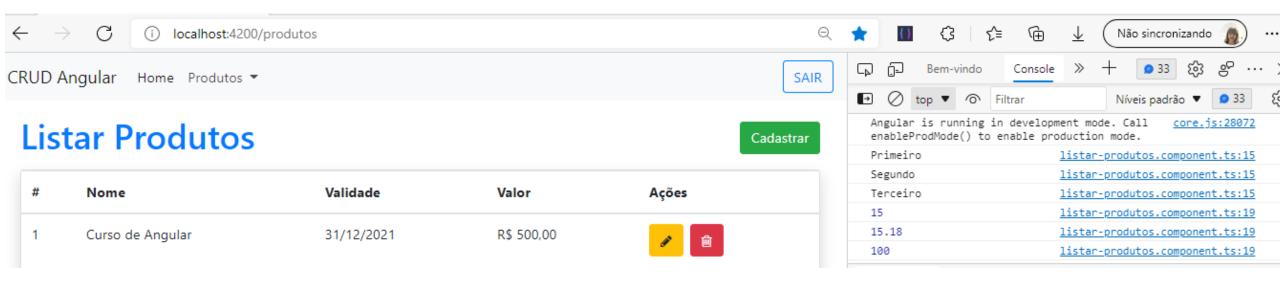
- Dessa maneira, já trabalhamos com o nosso primeiro for, utilizando um array.
- Também podemos criar um array de números
- listaNumeros: number[] = [aqui iremos passar os valores]
- listaNumeros: number[] = [15, 15.18, 100];
- Depois já podemos fazer a impressão dele:
- Vamos utilizar o for() -> Dentro do parênteses, vamos utilizar o const, porque cada item do nosso for será uma constante. Durante a execução da rodada do for os dados do item não poderão ser alterados. Vamos trazer nossa listaNumeros

```
for(const item of this.listaNumeros) {
      console.log(item);
}
```

Vamos percorrer a nossa listaNumeros, para cada item da nossa listaNumeros iremos guardar na variável item e vamos realizar a impressão.

Vejam que cada variável que criamos aqui, a primeira item está restrita somente ao escopo do for e depois a variável item novamente está restrita ao escopo do segundo for. Essas duas variáveis estão restritas a escopo, por isso é que nós podemos utilizar let e const.

- Vamos salvar, ir na página
- Pressionar <F12> e verificar se está exibindo o conteúdo dos arrays.
- Ir na opção Console



- Qual a diferença de let e const?
 - Let permite alteração de conteúdo do item
 - Const cria uma constante e essa constante não pode ser alterada durante a programação. Uma vez criada essa variável item, ela não pode ser alterada até o final da execução. Como estamos dentro do for, o final da execução é o final das chaves. Só imprime o item, já limpa a memória e já coloca um novo item no lugar.

```
for(const item of this.listaNumeros) {
    console.log(item);
}
```

Qual a forma correta de fazer aqui? Utilizar let ou const? As duas estão corretas. Por boas práticas, iremos utilizar o const item que forma aqui uma constante.

- Como fazemos agora para criarmos objetos?
- Após a criação das variáveis, vamos criar um objeto. Para criar objetos, após o nome do objeto, vamos digitar chaves {}. As chaves criam objetos dentro do javaScript, consequentemente dentro do TypeScript. Um objeto é sempre composto por propriedade e valor, sendo propriedade: valor. Vamos criar alguns exemplos abaixo, como string, number, boleano.

```
objetoModelo = {
nome: 'Fatima',
idade: 18,
altura: 1.56,
graduado: true
```

12

Ficaria assim:

```
10
        listaStrings: string[] = ['Primeiro', 'Segundo', 'Terceiro'];
        listaNumeros: number[] = [15, 15.18, 100];
12
13
        objetoModelo = {
14
          nome: 'Fatima',
15
          idade: 18,
16
          altura: 1.56,
17
          graduado: true
18
        };
```

- Como devemos fazer para realizar a impressão desse objeto?
- Podemos imprimir o objeto inteiro utilizando o console. Log. Lembrando que só podemos fazer a impressão do objeto inteiro dentro do console.log, se estiver dentro do HTML não é possível fazer isso. Ficará assim:

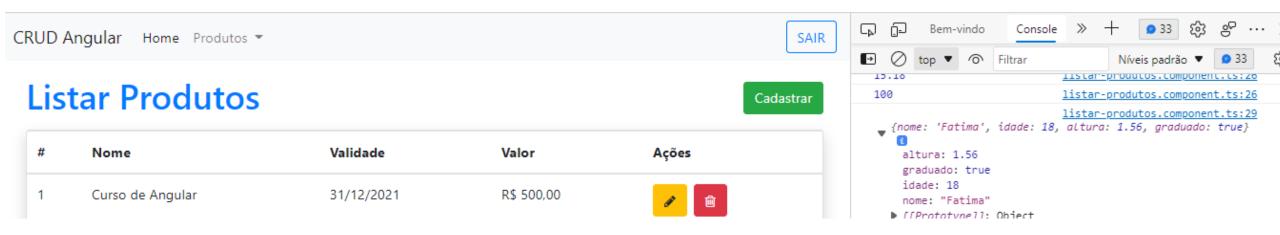
```
constructor() {
    for (let item of this.listaStrings) {
        console.log(item);
    }

for(const item of this.listaNumeros) {
        console.log(item);
    }

console.log(item);
}

console.log(this.objetoModelo);
}
```

- Vamos salvar, ir na página
- Pressionar <F12> e verificar se está exibindo o conteúdo dos arrays.
- Ir na opção Console e vamos verificar que temos agora o objeto embaixo dos numeros.
 Vamos ter o nome, altura, peso e graduado:

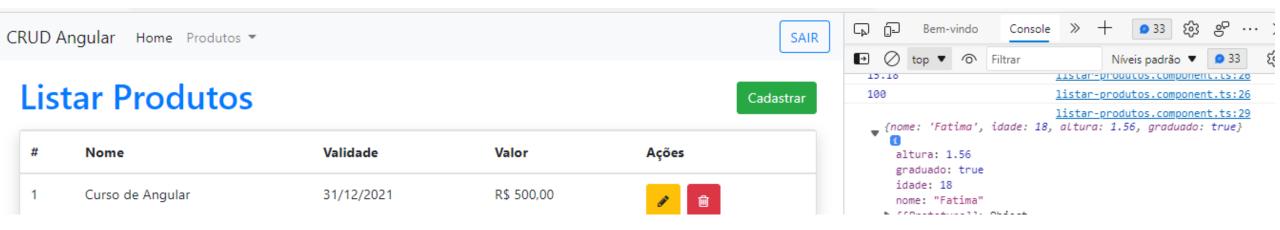


- Como faria para imprimir somente um item do objeto?
- Após o objeto, digitaria um ponto. Através do ponto, conseguimos acesso a qualquer uma das propriedades do meu objeto

console.log(this.objetoModelo.nome)

```
constructor() {
          for (let item of this.listaStrings) {
21
22
            console.log(item);
23
24
          for(const item of this.listaNumeros) {
25
26
            console.log(item);
27
28
29
          console.log(this.objetoModelo);
30
          console.log(this.objetoModelo.nome);
31
```

- Vamos salvar, ir na página
- Pressionar <F12> e verificar se está exibindo o conteúdo dos arrays.
- Ir na opção Console e vamos verificar tanto o objeto sendo impresso, quanto o nome.



- Poderíamos criar também um array de objetos. Vamos criar uma lista de produtos. Essa lista de produtos será igual a colchetes [] e, para que não tenhamos problemas, vamos conhecer um novo tipo que é o any. O any aceita qualquer coisa que iremos colocar aqui dentro. Então, any[] para que possamos criar um array.
- Vamos colocar o nosso objeto dentro do array {}. Dentro do objeto iremos passar o nome do produto, o preço do produto, validade e por fim o id que será 1. Ficaria assim:
 - nome: Curso de Angular
 - precoProduto: 35.36
 - validade: 2021-12-31
 - Id: 1

```
listaProdutos: any[] = [
{nome: 'Curso de Angular', precoProduto: 35.56, validade: '2021-12-31', id: 1},
```

Após a chaves inserir a vírgula

```
objetoModelo = {
14
          nome: 'Fatima',
15
          idade: 18,
16
          altura: 1.56,
          graduado: true
18
        };
19
20
        listaProdutos: any[] = [
21
          {nome: 'Curso de Angular', precoProduto: 35.56, validade: '2021-12-31', id: 1},
22
        ];
```

Não esquecer do ponto e vírgula no final

- Vamos inserir mais um objeto, copiando e colando a linha anterior e alterando o conteúdo:
 - nome: Curso de Ionic
 - precoProduto: 50
 - Validade : a mesma da anterior
 - Id: 2

- Agora temos aqui uma lista de produtos que temos aqui o tipo any. O tipo any aceita qualquer coisa, inclusive vai permitir que troque os tipos de conteúdos que estão dentro da minha variável durante a execução do meu código, diferente do string que se eu passar string ele sempre terá que receber um texto. Se eu colocar number, sempre terá que receber um número. Com o any iremos aceitar qualquer tipo de coisa que iremos passar por aqui.
- Estamos criando uma lista de produtos onde teremos 2 produtos. O primeiro produto é o curso de Angular, com preço de 35,56, com validade de 31/12/2021 e o id é igual a 1.
- Depois temos o curso de lonic, com preço de 50, validade também o final do ano e o id é igual a 2.

```
listaProdutos: any[] = [
  {nome: 'Curso de Angular', precoProduto: 35.56, validade: '2021-12-31', id: 1},
  {nome: 'Curso de Ionic', precoProduto: 50, validade: '2021-12-31', id: 2},
1:
```

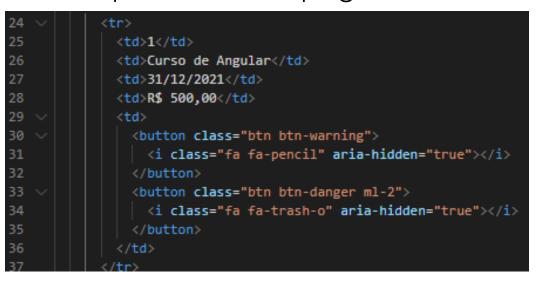
- Vamos criar um terceiro objeto. Uma coisa legal, os objetos não precisam seguir a mesma ordem, podemos mudar.
 - id: 3
 - nome: Curso de Ionic Avançado
 - precoProduto: 50
 - validade: 31/12/2021

```
listaProdutos: any[] = [
    {nome: 'Curso de Angular', precoProduto: 35.56, validade: '2021-12-31', id: 1},
    {nome: 'Curso de Ionic', precoProduto: 50, validade: '2021-12-31', id: 2},
    {id: 3, nome: 'Curso de Ionic Avançado', precoProduto: 50, validade: '2021-12-31'},
];
```

- Podemos ter uma propriedade a mais em um dos objetos e em outro não?]
 - Sim, podemos.
 - Por exemplo, no Curso de Ionic iremos acrescentar a propriedade promocao.

```
listaProdutos: any[] = [
   {nome: 'Curso de Angular', precoProduto: 35.56, validade: '2021-12-31', id: 1},
   {nome: 'Curso de Ionic', precoProduto: 50, validade: '2021-12-31', id: 2, promocao: true},
   {id: 3, nome: 'Curso de Ionic Avançado', precoProduto: 50, validade: '2021-12-31'}
];
```

- Agora podemos fazer a utilização dessa lista de produtos, mas não iremos mais utilizar o construtor. Iremos aprender como fazer for dentro da nossa lista de produtos.
- No arquivo listar-produtos.componente.html, nós temos aqui uma linha que é a que criamos como modelo para a nossa página.
- Após verificar a página, vamos fechar o módulo desenvolvedor.



#	Nome	Validade	Valor	Ações
1	Curso de Angular	31/12/2021	R\$ 500,00	

- Vamos vir agora na tag porque é o elemento onde queremos que se repita, pois iremos querer ter várias linhas de curso agora e vamos digitar, após o <tr,
- *ngFor = "". Dentro das aspas, vamos criar da mesma maneira que fizemos no for. Ficaria assim:

- Vamos salvar e verificar como ficou nossa página.
- Exibiu na tela o Curso de Angular 3x. Agora vamos alterar as informações de cada linha.

Listar Produtos

Cadastrar

#	Nome	Validade	Valor	Ações
1	Curso de Angular	31/12/2021	R\$ 500,00	
1	Curso de Angular	31/12/2021	R\$ 500,00	
1	Curso de Angular	31/12/2021	R\$ 500,00	

- Para exibir o conteúdo, vamos utilizar a interpolação. Para utilizar a interpolação, temos que digitar chaves {{ }}.
- Para exibir o código, como devemos fazer?
 - { prod.id }}
- No nome do curso, como ficaria?
 - { prod.nome }}
- Verifiquem que estamos colocando as mesmas propriedades que temos na nossa lista.
 Por que estou utilizando prod? Porque foi o nome da variável que criamos para receber temporariamente cada item da nossa lista de produtos.
- Para a validade, após a propriedade vamos utilizar o pipe | date: "shortDate"
 - { prod.validade | date: "shortDate" }}
- Para o preço do produto
 - { prod.precoProduto | currency: 'BRL' }}

- Vamos formatar nosso código: <ALT> <SHIFT> <F>
- Vamos salvar
- Vamos verificar como ficou nossa página

Visualizando a página

- Itens 1, 2 e 3, curso de Angular, curso de Ionic e Curso de Ionic Avançado
- Datas já foram formatadas para o português
- Os valores já aparecem com a formatação em português

Listar Produtos



#	Nome	Validade	Valor	Ações
1	Curso de Angular	31/12/2021	R\$ 35,56	
2	Curso de Ionic	31/12/2021	R\$ 50,00	
3	Curso de Ionic Avançado	31/12/2021	R\$ 50,00	

O que aprendemos...

- Agora vocês já sabem como fazer for utilizando o ngFor, a parte do html como fazer for dentro do html
- Como imprimir as propriedades de um objeto utilizando a interpolação, inclusive junto com pipes
- Conseguimos fazer isso também dentro do typeScript, fazendo os for aqui dentro e criando as nossas variáveis de arrays e de objeto, inclusive a nossa variável que é um array de objetos.

