Pipes Customizados com Parâmetros

Seguindo o exemplo anterior, de nosso Pipe que soma os valores de um array, suponhamos que, ao usar o Pipe, o usuário dele possa informar quantos valores do array ele quer somar ao invés de somar todos os valores. Nós podemos fazer isso permitindo que o usuário do Pipe passe um parâmetro para ele. Para isso, temos que fazer com que o nosso Pipe espere um parâmetro. Para que ele espere parâmetros, basta nós adicionarmos mais parâmetros ao método *transform* que nós implementamos.

```
import { Pipe, PipeTransform } from '@angular/core';

@Pipe({
  name: 'arraySum'
})
export class ArraySumPipe implements PipeTransform {

  transform(value: number[], limit: number = 0): number {
    // nosso código...
  }
}
```

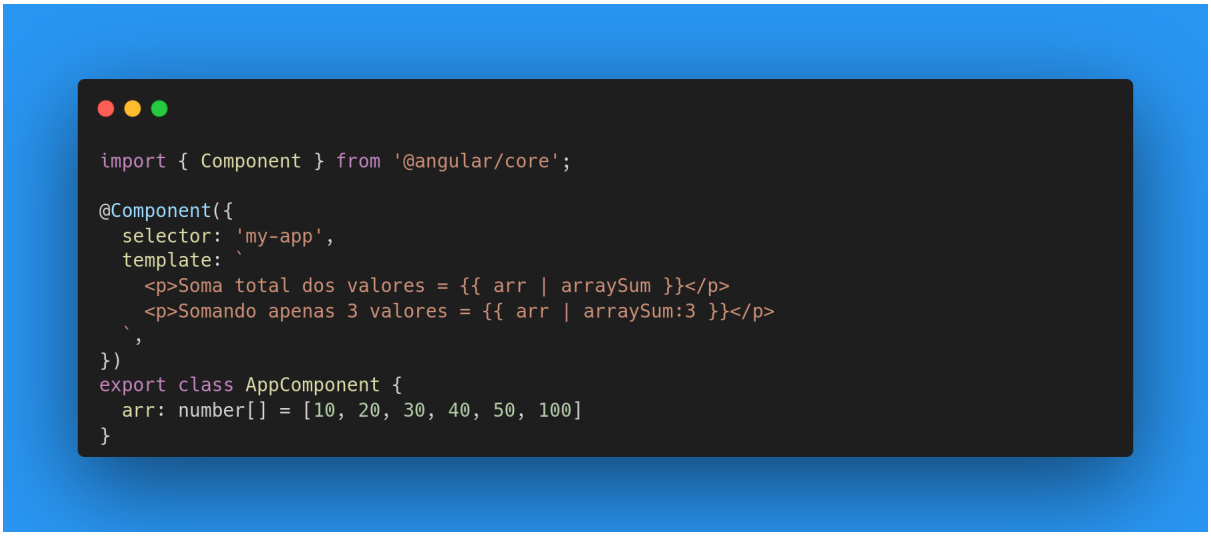
O nome do novo parâmetro é **limit**, pois será ele quem vai nos informar o quantidade limite de valores que devemos somar. Demos à variável um valor inicial de **0**. É uma boa prática nos Pipes fazer com que os parâmetros tenham um valor inicial, justamente para que, caso o usuário não informe um valor, você já tenha um valor padrão.

Agora, já que estamos recebendo um novo parâmetro, vamos alterar o nosso código para que ele possa trabalhar com esse novo parâmetro.

```
import { Pipe, PipeTransform } from '@angular/core';

@Pipe({
  name: 'arraySum'
})
export class ArraySumPipe implements PipeTransform {

  transform(value: number[], limit: number = 0): number {
    let sum: number = 0

    if (limit <= -1 || limit > value.length) {
      for (let v of value) {
        sum += v
      }
    } else {
      for (let i = 0; i < limit; i++) {
        sum += value[i]
      }
    }

    return sum
  }
}
```

A alteração que foi feita foi fazer um teste. Caso o valor que esteja dentro do parâmetro limite seja menor ou igual a zero ou seja maior que a quantidade de valores dentro do Array, ele vai fazer a soma total do Array. Caso contrário, ele vai somar conforme o limite informado.

Agora vamos testar ele no nosso componente



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

@Component({
  selector: 'my-app',
  template: `
    <p>Soma total dos valores = {{ arr | arraySum }}</p>
    <p>Somando apenas 3 valores = {{ arr | arraySum:3 }}</p>
  `,
})
export class AppComponent {
  arr: number[] = [10, 20, 30, 40, 50, 100]
}
```

Como vimos anteriormente, para passarmos valores ao parâmetros de um Pipe, nós fazemos a sintaxe **nomeDoPipe:parametro1:parametro2:parametro3…**. Cada parâmetro que passamos fica separados por dois pontos (**:**). Nesse caso, o número **3** é o valor que passei ao parâmetro **limit**, declarado no Pipe.