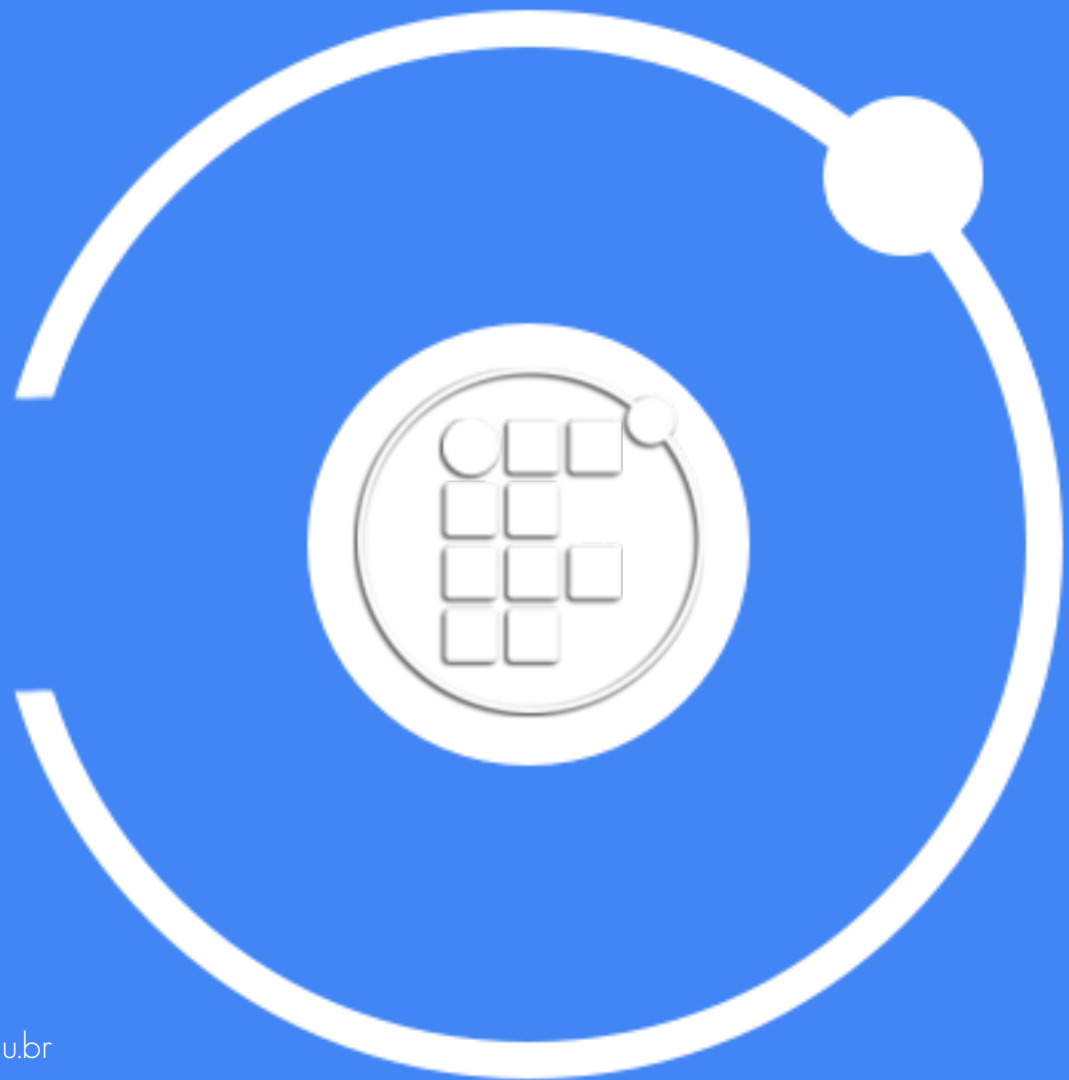


Ionic Framework

Pipes / filtros






- Filtros, ou pipes como é chamado originalmente em inglês, nada mais são do que funções que aplicam alguma transformação em um valor a ser exibido no template / tela.
- Esses valores podem ser um texto simples, uma data, uma número, um valor monetário, ou até mesmo um valor customizado como um número de RG ou CPF.





- Sua utilização é feita através do operador pipe “|”, e funciona de modo muito similar ao comandos utilizados no console de texto de um sistema operacional Unix.
- Nos sistemas baseados em Unix, o operador pipe funciona como um encadeamento de execuções, onde as funções passadas entre os pipes recebem como entrada o valor de saída da operação executada anteriormente.

Filtered Results given to the next command



```
home@VirtualBox:~$ cat sample | grep -v a | sort -r
Hid
First
Dog
Apple
```



- Os filtros mais simples são os de transformação de texto. No exemplo a seguir estamos exibindo um texto em caixa alta (uppercase), e o outro em caixa baixa (lowercase).

```
1 import {Component} from 'angular2/core';
2
3 @Component({
4   selector: 'meu-app',
5   template: `
6     <h1>Angular - Exemplo </h1>
7     <p>Caixa alta: {{texto | uppercase}}</p>
8     <p>Caixa baixa: {{texto | lowercase}}</p>
9   `
10 })
11 export class AppComponent{
12   texto: string = 'Prog. dispositivos móveis'
13 }
```



- Ainda com base no exemplo anterior, repare como o filtro é aplicado, é definido uma variável 'texto' contendo o valor 'Prog. Dispositivos móveis'.
- Ao chamar a variável 'texto' no template, adicionamos o operador **pipe** seguido do nome do filtro, assim o Angular se encarregará de passar o valor a ser exibido na tela para o filtro 'uppercase' ou 'lowercase', que transformará a entrada gerando a saída desejada





- Valores monetários são dados utilizados com muita frequência e variam de país para país dependendo da moeda, assim o Angular fornece um filtro específico para formatação de valores monetários.'
- O filtro 'currency' é o responsável por tal formatação, e ele possui dois parâmetros de configuração, a moeda (país) e se o símbolo monetário deve ser exibido.



Pipes valores monetários



```
1 @Component({
2   selector: 'meu-app',
3   template: `
4     ...
5     <p>Valor: {{valor | currency:'BRL':true}}
6   `
7 })
8 export class AppComponent{
9   ...
10   valor: number = 1046.99;
11 }
```

- No exemplo acima, utilizamos o operador de dois pontos para separar os parâmetros de configuração, e os parâmetros aceitos indicam a moeda em questão, e se o símbolo monetário deve ser exibido.

Pipes para datas



```
1 @Component({
2   selector: 'meu-app',
3   template: `
4     <h1>Angular - pipes </h1>
5     ...
6     <p>Data atual: {{dataAtual | date:'dd/MM/yyyy'}}</p>
7   `
8 })
9 export class AppComponent{
10   ...
11   dataAtual: Date = new Date();
12   formato: boolean = true;
13 }
14 }
```

- filtro utilizado para a formatação de datas é o 'date', e ele possui como parâmetro o formato da data

Exemplos práticos

Pipe personalizado





O Ionic CLI fornece um mecanismo para gerar filtros dentro da nossa aplicação

```
$ ionic g pipe path/nome_filtro
```



- Por vezes precisamos aplicar transformações que não são suportadas pelo Angular, como por exemplo. formatar um CEP, CPF, entre outros.
- Um pipe personalizado é uma classe (typescript) anotada com a anotação '@Pipe' que implementar a interface 'PipeTransform', ambas do pacote core do Angular..



Exemplo



- Ao implementar a interface 'PipeTransform', você deverá sobrescrever o método 'transform', que receberá o texto padrão como entrada e retornará o texto de modo a ser exibido na tela.
- Abaixo segue um filtro personalizado que formata um cep..

```
1  import { Pipe, PipeTransform } from '@angular/core';
2
3  @Pipe({
4    | name: 'cep' // <-- este será o nome utilizado no template
5  })
6  export class CepPipe implements PipeTransform {
7    | transform(value: string, ...args: any[]): string {
8    | |   return value.substr(0, 5) + '-' + value.substr(5, 3);
9    | }
10 }
```

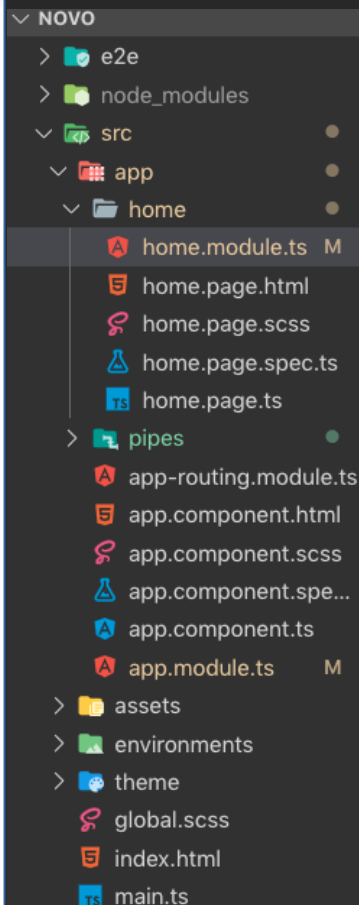
Exemplo



- Para utilizar o pipe personalizado será necessário carregá-lo nos módulos nos quais ele será aplicado (para cada view)
- Obs: A entrada automática gerada pelo CLI no `app.module.ts` deve ser removida!



Exemplo



```
src > app > home > home.module.ts > ...  
5 import { RouterModule } from '@angular/router';  
6  
7 import { HomePage } from './home.page';  
8 import { CepPipe } from '../pipes/cep.pipe';  
9  
10 @NgModule({  
11   imports: [  
12     CommonModule,  
13     FormsModule,  
14     IonicModule,  
15     RouterModule.forChild([  
16       {  
17         path: '',  
18         component: HomePage  
19       }  
20     ])  
21   ],  
22   declarations: [HomePage, CepPipe]  
23 })  
24 export class HomePageModule {}
```

Declaração no module
específico da página

```
src > app > home > home.page.html > ...
```

```
1 <ion-header>
2   <ion-toolbar>
3     <ion-title>
4       Ionic app
5     </ion-title>
6   </ion-toolbar>
7 </ion-header>
8
9 <ion-content>
10   {{ cepNormal | cep }}
11 </ion-content>
```

Visão

```
1 import { Component } from '@angular/core';
2
3 @Component({
4   selector: 'app-home',
5   templateUrl: 'home.page.html',
6   styleUrls: ['home.page.scss']
7 })
8 export class HomePage {
9   cepNormal = '97670000';
10   constructor() {}
11 }
```

Controle



- A API padrão do Angular possui alguns outros filtros implementados, e mais ainda deverão ser adicionados com o passar do tempo.
- Para se manter atualizado sobre os filtros existentes acesse a documentação oficial

<https://angular.io/guide/pipes>

