

Visualização dos dados do usuário

Transcrição

Nós já estamos acessando a tela de perfil, mas os dados estão fixos. Porém os dados estão fixos.



O título também não especifica em qual View estamos. Temos alguns desafios para resolver. Para resolvermos o assunto, sabemos que podemos usar a `. Mas existe a opção de adicionarmos também a diretiva view-title, que terá a mesma finalidade.`

```
<ion-view view-title="Perfil do Usuário">
  <ion-content>
    <div class="list card">
      <div class="item item-avatar">
        Usuário Logado
      </div>
    </div>
  </ion-content>
</ion-view>
```

Precisamos alterar os dados fixos, pelos dados informados no login. No arquivo `controller.js`. Atualmente, usamos o `$rootScope` para pegar os dados do usuário.

```
CarroService.realizarLogin(dadosDoLogin).then(function(dados){
  $rootScope.usuario = dados.usuario;

  $state.go('app.listagem');
}, function(erro){
  $ionicPopup.alert({
    title : 'Opa!',
    template : 'Email ou senha incorretos.'
  })
})
```

```
    })  
  
  }  
})
```

Teremos que pegar os dados e colocar dentro de uma controller , no caso a do perfil que criaremos a seguir.

```
angular.module('starter')  
  .controller('PerfilController', function($rootScope, $scope){  
  
  })
```

Como precisamos pegar o usuário que está no escopo, usaremos o \$rootScope . Também adicionaremos o \$scope , para colocar dentro do \$scope . Vamos criar a variável usuarioLogado , que receberá o \$rootScope.usuario .

```
angular.module('starter')  
  .controller('PerfilController', function($rootScope, $scope){  
    $scope.usuarioLogado = $rootScope.usuario;  
  })
```

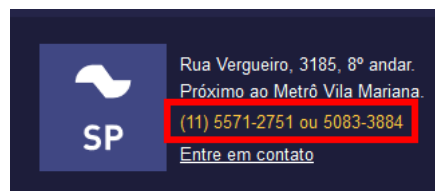
Em seguida, usaremos os dados no cartão. Substituiremos o dado fixo Usuario Logado , por dado dinâmico e darei mais destaque usando a tag <h2> .

```
<ion-view view-title="Perfil do Usuário">  
  <ion-content>  
    <div class="list card">  
      <div class="item item-avatar">  
        <h2> {{usuario.Logado.nome}} </h2>  
      </div>  
      <div class="item item-body">  
        <div class="list">  
  
          <label class="item item-input">  
            <span class="input-label"> {{usuarioLogado.dataNascimento}}  
          </span>  
          </label>  
          <label class="item item-input">  
            <span class="input-label"> {{usuarioLogado.telefone}} </span>  
          </label>  
          <label class="item item-input">  
            <span class="input-label"> {{usuarioLogado.email}} </span>  
          </label>  
          <button class="button button-full button-positive">Editar</button>  
        </div>  
      </div>  
    </div>  
  </ion-content>  
</ion-view>
```

Vamos salvar as alterações feitas no arquivos e ver como ficou a tela.



Veja que agora os dados são dinâmicos. Mas observe que o campo de telefone está fora do padrão - Faltaram os parênteses e o hífen separando os números, como encontramos no site da [Caelum](https://www.caelum.com.br/) (<https://www.caelum.com.br/>):



Grupo Caelum

Esse é um padrão que o usuário está habituado já, então, faremos o mesmo com a nossa aplicação. Como trabalhamos com Angular, para fazer isso, usaremos os filtros. Também é possível usar as bibliotecas de filtros que estão disponíveis na internet. O filtro que usaremos é comumente usado no Brasil: [Angular-br-filters](http://the-darc.github.io/angular-br-filters/) (<http://the-darc.github.io/angular-br-filters/>).

Veja que com o `brPhoneNumber` conseguimos que o telefone fique no formato que desejamos. No [Github](https://github.com/the-darc/angular-br-filters) (<https://github.com/the-darc/angular-br-filters>), encontramos a documentação que explica como instalar. Por exemplo,

vemos as opções de instalar com Bower ou NPM. Nós estamos usando o Ionic, logo já temos o Bower.

Installation

With Bower:

```
bower install --save angular-br-filters
```

With NPM:

```
npm install --save angular-br-filters
```

--

Quando rodarmos o comando, ele rodará dentro do nosso projeto. Depois, teremos que importar o arquivo JS e importar o módulo para a nossa aplicação poder usar.

Começaremos digitando o comando na linha de comando:

```
bower install --save angular-br-filters
```

Em seguida, no `index.html`, vamos carregar o arquivo JS. Adicionaremos outra tag `script` e passaremos a rota.

```
<!-- your app's js -->
<script src="js/app.js"></script>
<script src="js/controllers.js"></script>
<script src="js/routes.js"></script>

<script src="lib/angular-br-filters.min.js"></script>
```

Depois, carregaremos o `idf.bbr-filters` no `app.js`. Atualmente, ele está assim:

```
angular.module('starter', ['ionic'])
```

Já tínhamos adicionado o `ionic`, adicionaremos o `idf.br-filters` recomendado pela documentação.

```
angular.module('starter', ['ionic', idf.br-filters])
```

Agora, usaremos o filtro `brPhoneNumber` no `perfil.html`.

```
<label class="item item-input">
  <span class="input-label"> {{usuarioLogado.telefone | brPhoneNumber}} </span>
</label>
```

Salvaremos as alterações e subiremos para o servidor.



Teremos o mesmo resultado no Android. Uma lib que criada para ser usada Web, está sendo usado no nosso aplicativo. Agora, recebemos os dados do login e colocamos para ser visualizada pelo usuário.

Até a próxima!