Programação Móvel

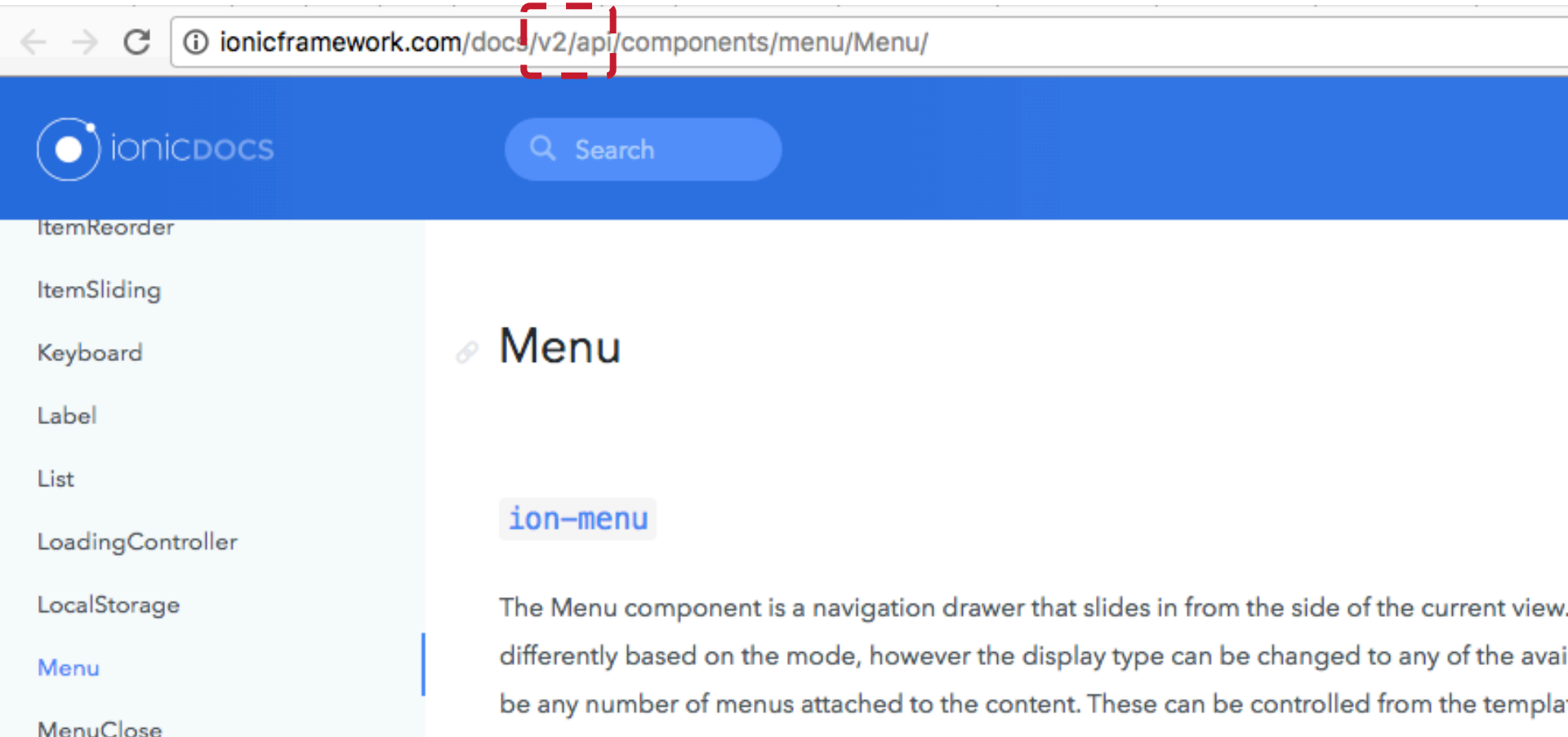
Componentes Ionic

Prof. Dr. Diego R. Antunes
drantunes@utfpr.edu.br

Departamento de Computação
Universidade Tecnológica Federal do Paraná

Documentação Versão 2

Cuidado ao acessar a documentação do Ionic. Na disciplina estamos utilizando a versão 2. Então verifique a URL no navegador para ver se está na versão correta.



Estrutura dos Projetos

Estrutura dos Projetos

Para criarmos um padrão durante a disciplina, vamos criar projetos Ionic para as Listas de Exercícios seguindo a seguinte regra:

Nome do Projeto = Nome_Aluno-Lista-N

N = Número da Lista de Exercícios em Questão

Estrutura dos Projetos

Cada projeto resolve todos os exercícios propostos na lista em questão. Quando for necessário, você também pode gerar uma página para cada exercício.

Estrutura dos Projetos

Assim, para compartilhamento via Ionic View fica mais fácil de identificar o autor do projeto e o que ele está resolvendo.

Componentes Visuais

Gesture

São eventos especiais utilizados pelo Ionic, tornando possível mapear gestos realizados pelo usuário e atribuir o evento para uma função em nossa classe.

http://ionicframework.com/docs/v2/components/#gestures

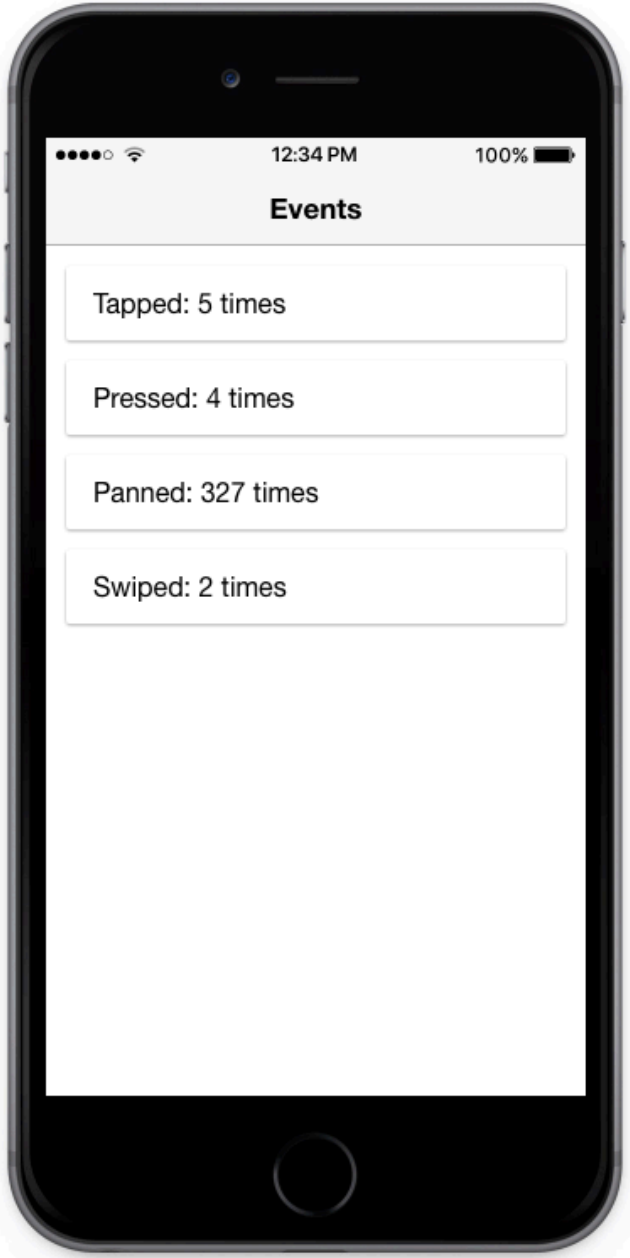
Gesture

Eventos disponíveis no Ionic: tap, press, pan, swipe, rotate e pinch

http://ionicframework.com/docs/v2/components/#gestures

Gesture

```
<ion-card (tap)="tapEvent($event)">
  <ion-item>
    Tapped:  times
  </ion-item>
</ion-card>
```



Gesture

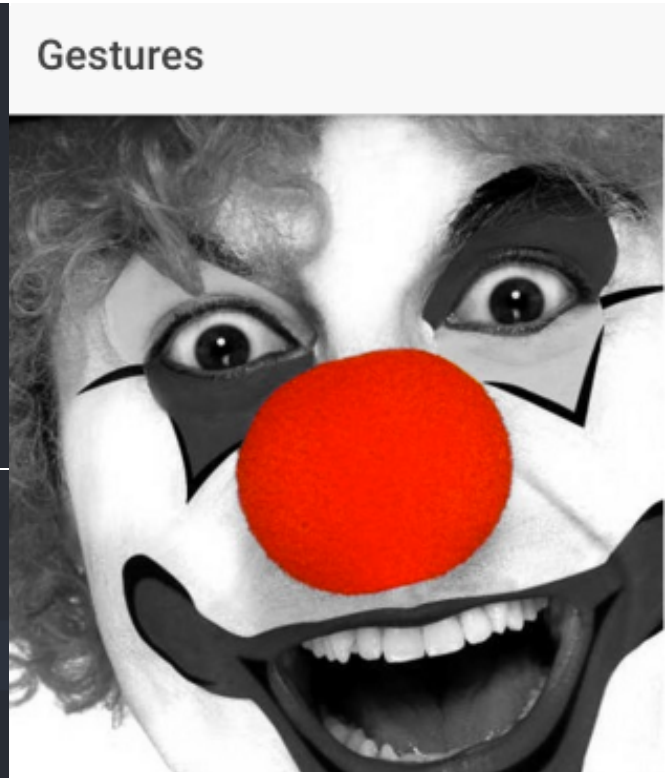
```
9   <ion-content>
10    <div (tap)="colorir(3)">
11      <img src="images/palhaco-{{id}}.jpg" />
12    </div>
13  </ion-content>
14
```

```
15    colorir(id) {
16        this.id = id;
17        setTimeout(() => this.resetImage(), 400);
18    }
19
20    resetImage() {
21        this.id = 1;
22    }
```

Gesture

```
9   <ion-content>
10    <div (tap)="colorir(3)">
11      <img src="images/palhaco-{{id}}.jpg" />
12    </div>
13  </ion-content>
14
```

```
15    colorir(id) {
16        this.id = id;
17        setTimeout(() => this.resetImage(), 400);
18    }
19
20    resetImage() {
21        this.id = 1;
22    }
```



Alert

Consiste em um componente do tipo modal ou janela de informação. O Alert pode ser utilizado para informar algo ao usuário ou mesmo solicitar informações rápidas.

http://ionicframework.com/docs/v2/components/#alert

Alert

Exemplos: *solicitar informações curtas como email ou senha (e.g. Wi-Fi), perguntar ao usuário uma determinada opção, ações do tipo SIM ou NÃO.*

http://ionicframework.com/docs/v2/components/#alert

Alert

No site do Ionic há diversos exemplos do uso do Alert com inputs, botões, funções de retorno, entre outros.

http://ionicframework.com/docs/v2/components/#alert

Importante

A maioria dos componentes do Ionic possuem uma descrição mais técnica relacionado à API. O Link é http://ionicframework.com/docs/v2/api/

Alert

Primeiro, importamos o controller do componente

```
1   import { Component } from '@angular/core';
2
3   import { AlertController } from 'ionic-angular';
4
5   @Component({
6       templateUrl: 'build/pages/alert-example/alert-example.html',
7   })
8   export class AlertExamplePage {
9
10      constructor(public alert: AlertController) { }
11
```

Alert

Depois, inserimos o componente no construtor, para o Angular realizar a instanciação para nós.

```
1   import { Component } from '@angular/core';
2
3   import { AlertController } from 'ionic-angular';
4
5   @Component({
6       templateUrl: 'build/pages/alert-example/alert-example.html',
7   })
8   export class AlertExamplePage {
9
10      constructor(public alert: AlertController) { }
11
```

Alert

Em seguida, vamos definir uma função para exibir nosso alerta.

```
12      abrirAviso() {
13          // Criar a instancia do Alert
14          let info = this.alert.create();
15          // Incluir as informações
16          info.setTitle('Aviso Urgente');
17          info.setMessage('Este componente é importante');
18          info.addButton('Entendi');
19          // Apresentar o Alert
20          info.present();
21      }
```

Alert

Nesta função nós criamos um alerta e atribuímos a uma variável local.

```
12    abrirAviso() {
13        // Criar a instancia do Alert
14        let info = this.alert.create();
15        // Incluir as informações
16        info.setTitle('Aviso Urgente');
17        info.setMessage('Este componente é importante');
18        info.addButton('Entendi');
19        // Apresentar o Alert
20        info.present();
21    }
```

Alert

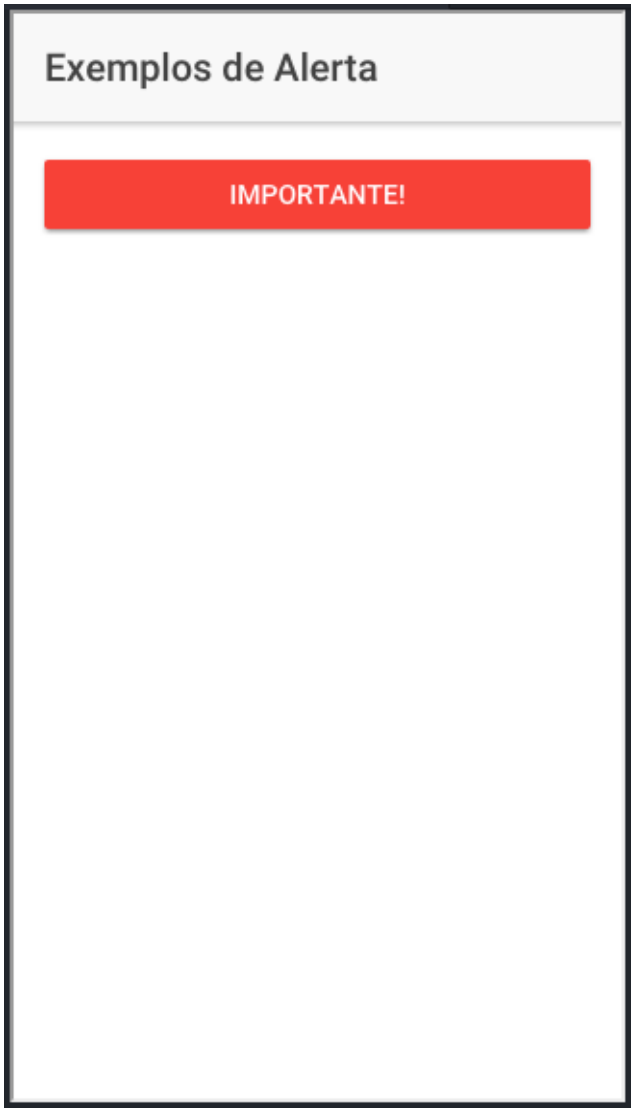
Então adicionamos o conteúdo ao Alert de acordo com a API do Ionic

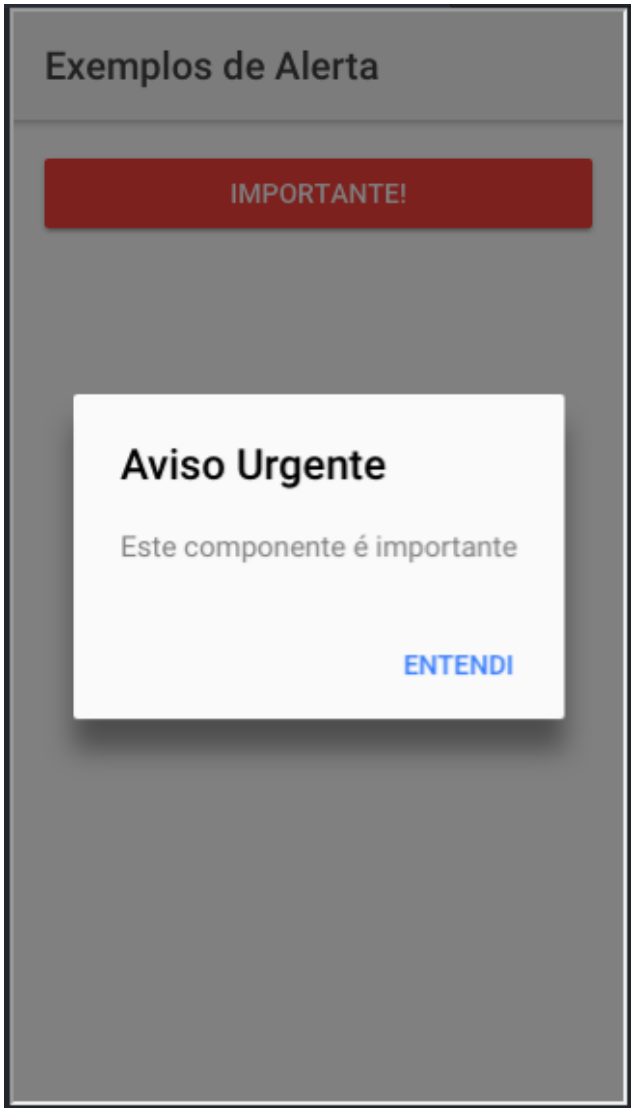
```
12    abrirAviso() {
13        // Criar a instancia do Alert
14        let info = this.alert.create();
15        // Incluir as informações
16        info.setTitle('Aviso Urgente');
17        info.setMessage('Este componente é importante');
18        info.addButton('Entendi');
19        // Apresentar o Alert
20        info.present();
21    }
```

Alert

Então, podemos apresentar o Alert. Você pode, por exemplo, criar o Alert em uma função e apresentá-lo em outra.

```
12      abrirAviso() {
13          // Criar a instancia do Alert
14          let info = this.alert.create();
15          // Incluir as informações
16          info.setTitle('Aviso Urgente');
17          info.setMessage('Este componente é importante');
18          info.addButton('Entendi');
19          // Apresentar o Alert
20          info.present();
21      }
```


Exemplos de Alerta
IMPORTANTE!


Exemplos de Alerta
IMPORTANTE!
Aviso Urgente
Este componente é importante
ENTENDI

Alert

Neste exemplo vamos inserir um input e uma ação ao botão

```
26    abrirAlertInput() {
27        let info = this.alert.create();
28        info.setTitle('Informe a Rede!');
29        // Adicionando um input
30        info.addInput({
31            type: 'text',
32            placeholder: 'Nome do Wifi'
33        });
34        // adicionar ação ao botão
35        info.addButton({
36            text: 'Salvar',
37            handler: (dados) => {
38                this.wifi = dados[0];
39            }
40        });
41
42        info.present();
43    }
```

```
26  abrirAlertInput() {
27      let info = this.alert.create();
28      info.setTitle('Informe a Rede!');
29      // Adicionando um input
30      info.addInput({
31          type: 'text',
32          placeholder: 'Nome do Wifi'
33      });
34      // adicionar ação ao botão
35      info.addButton({
36          text: 'Salvar',
37          handler: (dados) => {
38              this.wifi = dados[0];
39          }
40      });
41
42      info.present();
43  }
```

```
26    abrirAlertInput() {
27        let info = this.alert.create();
28        info.setTitle('Informe a Rede!');
29        // Adicionando um input
30        info.addInput({
31            type: 'text',
32            placeholder: 'Nome do Wifi'
33        });
34        // adicionar ação ao botão
35        info.addButton({
36            text: 'Salvar',
37            handler: (dados) => {
38                this.wifi = dados[0];
39            }
40        });
41
42        info.present();
43    }
```

Handler

Esta propriedade recebe uma função anônima. Com ela você pode receber os dados do Alert e usá-los em sua classe. As informações são injetadas no nome da variável que você definiu como parâmetro, em nosso caso, dados.

```
37          handler: (dados) => {
38              this.wifi = dados[0];
39          }
```

Modal

Funciona de maneira similar ao Alert. A diferença é que o modal carrega mais informações, geralmente em tela cheia.

http://ionicframework.com/docs/v2/components/#modals

Modal

Exemplos: escrever uma mensagem ou email, exibir informações detalhadas sobre um item, entre outros.

http://ionicframework.com/docs/v2/components/#modals

Vamos definir duas páginas: a página de origem (ModalExample) e a página destino (ModalDestino).

No ModalExample vamos criar listas ou botões que irão acionar e criar uma nova Modal, passando como parâmetro qual Modal será aberta e parâmetros adicionais.

Precisamos importar o ModalController para poder criar uma Modal, semelhante ao Alert. Em seguida, devemos injetar este componente no construtor.

```
1   import { Component } from '@angular/core';
2   import { ModalController } from 'ionic-angular';
3
4   import { ModalDestino } from '../modal-destino/modal-destino';
5
6   @Component({
7       templateUrl: 'build/pages/modal-example/modal-example.html',
8   })
9   export class ModalExamplePage {
10
11      constructor(private modalCtrl: ModalController) { }
```

Também precisamos importar o componente que define o modelo da ModalDestino.

```
1   import { Component } from '@angular/core';
2   import { ModalController } from 'ionic-angular';
3
4   import { ModalDestino } from '../modal-destino/modal-destino';
5
6   @Component({
7       templateUrl: 'build/pages/modal-example/modal-example.html',
8   })
9   export class ModalExamplePage {
10
11      constructor(private modalCtrl: ModalController) { }
```

Em seguida, criamos uma função (abrir) para criar a Modal, de forma semelhante ao Alert.

```
11      constructor(private modalCtrl: ModalController) { }
12
13      abrir() {
14          // Cria a modal e como parâmetros:
15          // A pagina de Destino (Componente Importado)
16          // Array com Dados que serão enviados (Opcional)
17          let modal = this.modalCtrl.create(ModalDestino, {
18              nome: 'Diego',
19              profissao: 'Professor',
20              disciplina: 'Arquitetura'
21          });
22          modal.present();
23      }
```

Então passamos como parâmetros o nome da modal (ModalDestino) e um objeto com parâmetros opcionais.

```
11      constructor(private modalCtrl: ModalController) { }
12
13      abrir() {
14          // Cria a modal e como parâmetros:
15          // A pagina de Destino (Componente Importado)
16          // Array com Dados que serão enviados (Opcional)
17          let modal = this.modalCtrl.create(ModalDestino, {
18              nome: 'Diego',
19              profissao: 'Professor',
20              disciplina: 'Arquitetura'
21          });
22          modal.present();
23      }
```

Já em nosso componente ModalDestino, precisamos importar dois módulos principais: NavParams (parâmetros) e ViewController (fechar o modal)

```
1  import { Component } from '@angular/core';
2  import { NavParams, ViewController } from 'ionic-angular';
3
4  @Component({
5      templateUrl: 'build/pages/modal-destino/modal-destino.html',
6  })
7  export class ModalDestino {
8      protected nome;
9      protected profissao;
10     protected disciplina;
11
12     constructor(private params: NavParams, private view: ViewController) {
```

Também é necessário criar algumas variáveis para atribuirmos os valores enviados por parâmetro para esta modal.

```
1   import { Component } from '@angular/core';
2   import { NavParams, ViewController } from 'ionic-angular';
3
4   @Component({
5       templateUrl: 'build/pages/modal-destino/modal-destino.html',
6   })
7   export class ModalDestino {
8     protected nome;
9     protected profissao;
10    protected disciplina;
11
12      constructor(private params: NavParams, private view: ViewController) {
```

No construtor, podemos realizar a atribuição de todos os valores enviados por parâmetro. Para isso, usamos this.params.get(nome_do_parametro);

```
12    constructor(private params: NavParams, private view: ViewController) {
13        this.nome = this.params.get('nome');
14        this.disciplina = this.params.get('disciplina');
15        this.profissao = this.params.get('profissao');
16    }
17
18    fechar() {
19        this.view.dismiss();
20    }
```

E também definimos uma função para fechar a modal, para que o usuário possa retornar a tela principal. Para isso, usamos o ViewController definido no construtor.

```
12    constructor(private params: NavParams, private view: ViewController) {
13        this.nome = this.params.get('nome');
14        this.disciplina = this.params.get('disciplina');
15        this.profissao = this.params.get('profissao');
16    }
17
18    fechar() {
19        this.view.dismiss();
20    }
```

A view do ModalPage deve exibir as informações definidas no contrutor. Mas, precisamos criar um botão para o usuário poder retornar (fechar a modal). Para isso, usaremos o ion-toolbar.

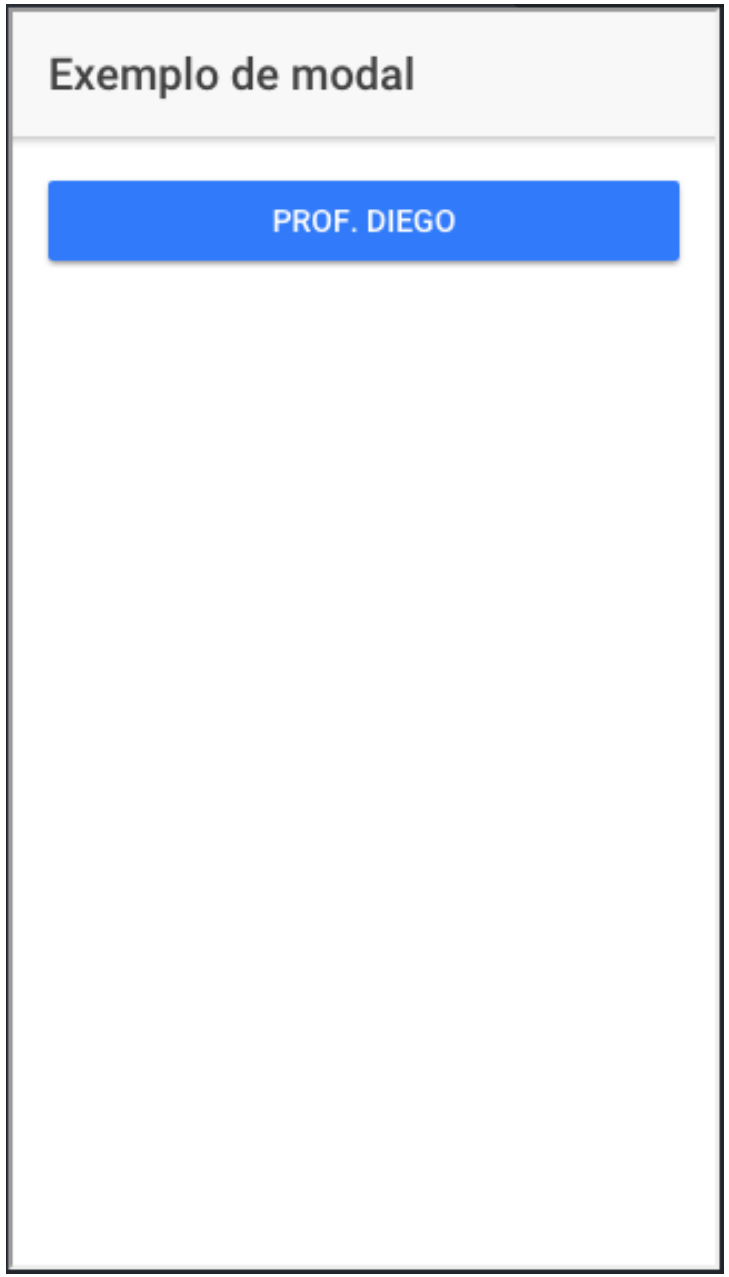
```
<ion-header>
  <ion-toolbar primary>
    <ion-title>
      {{ nome }}
    </ion-title>
    <ion-buttons start>
      <button (click)="fechar()">
        <ion-icon name="md-close"></ion-icon>
      </button>
    </ion-buttons>
  </ion-toolbar>
</ion-header>

<ion-content padding>
  <h2>Profissão: {{ profissao }}</h2>
  <p>Disciplina: {{ disciplina }}</p>
</ion-content>
```

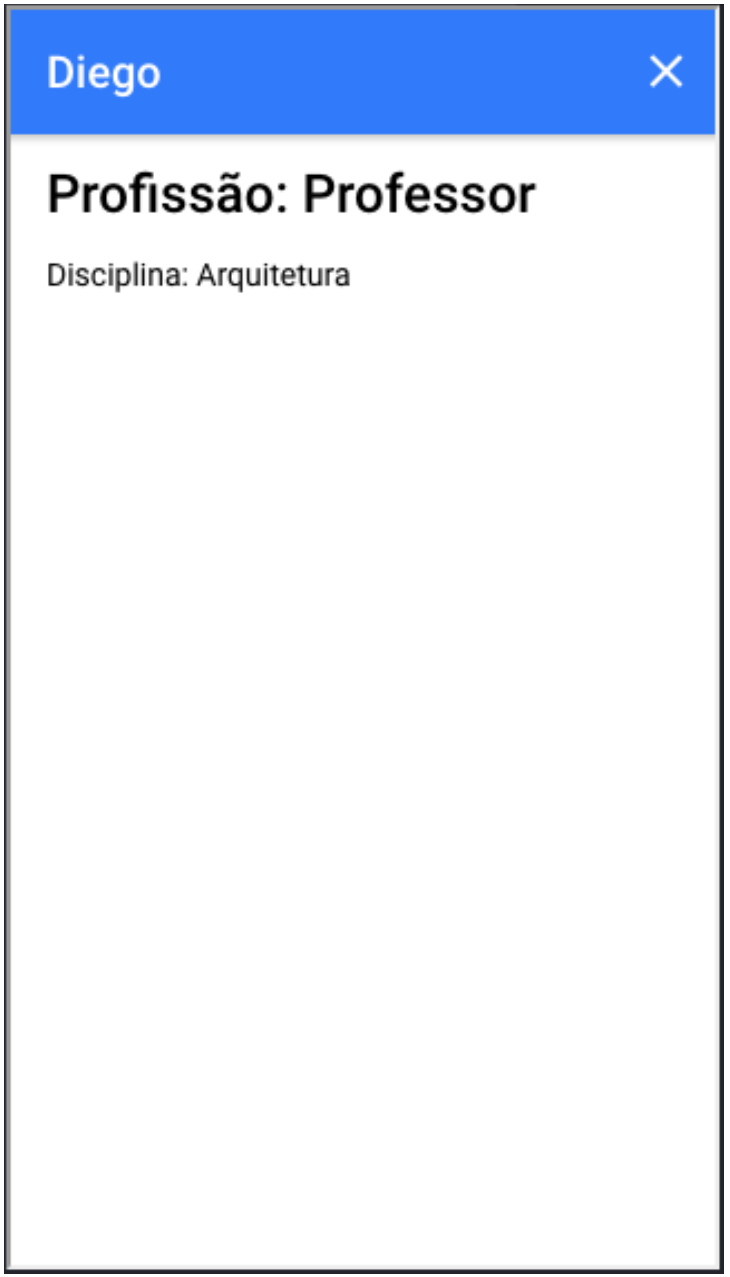
```
<ion-header>
  <ion-toolbar primary>
    <ion-title>
      {{ nome }}
    </ion-title>
    <ion-buttons start>
      <button (click)="fechar()">
        <ion-icon name="md-close"></ion-icon>
      </button>
    </ion-buttons>
  </ion-toolbar>
</ion-header>

<ion-content padding>
  <h2>Profissão: {{ profissao }}</h2>
  <p>Disciplina: {{ disciplina }}</p>
</ion-content>
```

```
<ion-header>
  <ion-toolbar primary>
    <ion-title>
      {{ nome }}
    </ion-title>
    <ion-buttons start>
      <button (click)="fechar()">
        <ion-icon name="md-close"></ion-icon>
      </button>
    </ion-buttons>
  </ion-toolbar>
</ion-header>

<ion-content padding>
  <h2>Profissão: {{ profissao }}</h2>
  <p>Disciplina: {{ disciplina }}</p>
</ion-content>
```

```
<ion-header>
  <ion-toolbar primary>
    <ion-title>
      {{ nome }}
    </ion-title>
    <ion-buttons start>
      <button (click)="fechar()">
        <ion-icon name="md-close"></ion-icon>
      </button>
    </ion-buttons>
  </ion-toolbar>
</ion-header>

<ion-content padding>
  <h2>Profissão: {{ profissao }}</h2>
  <p>Disciplina: {{ disciplina }}</p>
</ion-content>
```

```
<ion-header>
  <ion-toolbar primary>
    <ion-title>
      {{ nome }}
    </ion-title>
    <ion-buttons start>
      <button (click)="fechar()">
        <ion-icon name="md-close"></ion-icon>
      </button>
    </ion-buttons>
  </ion-toolbar>
</ion-header>

<ion-content padding>
  <h2>Profissão: {{ profissao }}</h2>
  <p>Disciplina: {{ disciplina }}</p>
</ion-content>
```


Exemplo de modal
PROF. DIEGO


Diego
Profissão: Professor
Disciplina: Arquitetura

Menu

Componente para posibilitar a navegação lateral, ou seja, ao clicar em um ícone o menu surge da lateral da tela. Ao clicar em uma opção, uma nova página é aberta.

http://ionicframework.com/docs/v2/components/#menus

Menu

O Componente também funciona por meio do ”Drag and Drop”, a partir do canto da tela.

http://ionicframework.com/docs/v2/components/#menus

Exemplo

Se você não criar um projeto Ionic por meio do template sidemenu, você pode adicionar um menu manualmente.

Exemplo

Primeiro, você deve gerar uma página nova em seu projeto, chamada menu. Você pode criar a página usando o comando ionic g page menu

Exemplo

Em seguida, você deve corrigir o root do arquivo app.ts. Precisamos informar o app, que ele deve iniciar na página menu! Não esqueça de importar o menu!

```
5   import { MenuPage } from './pages/menu/menu';
6
7
8   @Component({
9       template: '<ion-nav [root]="rootPage"></ion-nav>'
10  })
11  export class MyApp {
12      rootPage: any = MenuPage;
13
```

Exemplo

Então, no componente menu, precisamos importar todas as páginas que queremos disponibilizar no menu.

```
1  import { Component } from '@angular/core';
2  import { NavController } from 'ionic-angular';
3
4  import { HomePage } from '../home/home';
5  import { ModalExamplePage } from '../modal-example/modal-example';
6
7  @Component({
8    templateUrl: 'build/pages/menu/menu.html',
9  })
10 export class MenuPage {
11
```

Depois de importar, precisamos atribuir cada página a uma variável ou uma lista (se você preferir).

```
12  protected paginaAtual = HomePage;
13  protected home = HomePage;
14  protected modal = ModalExamplePage;
15
16      constructor(private navCtrl: NavController) {
17      }
18
19      abrir(page){
20          this.paginaAtual = page;
21      }
```

Também criamos a variável chamada paginaAtual, que recebe a primeira página que desejamos que o menu abra na primeira vez.

```
12    protected paginaAtual = HomePage;
13    protected home = HomePage;
14    protected modal = ModalExamplePage;
15
16      constructor(private navCtrl: NavController) {
17      }
18
19      abrir(page){
20          this.paginaAtual = page;
21      }
```

Depois, definimos um método chamado abrir, que recebe a página que queremos abrir. Então, podemos alterar a página atual.

```
12    protected paginaAtual = HomePage;
13    protected home = HomePage;
14    protected modal = ModalExamplePage;
15
16      constructor(private navCtrl: NavController) {
17      }
18
19      abrir(page){
20          this.paginaAtual = page;
21      }
```

No HTML do Menu precisamos sinalizar qual o conteúdo o Ionic deve olhar. Ou seja, para abrir o conteúdo externo e não o conteúdo do menu. Para isso vamos usar o elemento ion-menu e ion-nav

```
1    <ion-menu [content]="paginas">
2      <ion-header>
3        <ion-navbar>
```

```
14   </ion-menu>
15
16   <ion-nav #paginas [root]="paginaAtual"></ion-nav>
```

menu.html

```
<ion-menu [content]="paginas">
  <ion-header>
    <ion-navbar>
      <ion-title>Opções</ion-title>
    </ion-navbar>
  </ion-header>

  <ion-content>
    <ion-list>
      <button menuClose ion-item (click)="abrir(home)">Home</button>
      <button menuClose ion-item (click)="abrir(modal)">Exemplo</button>
    </ion-list>
  </ion-content>
</ion-menu>

<ion-nav #paginas [root]="paginaAtual"></ion-nav>
```

menu.html

```
<ion-menu [content]="paginas">
  <ion-header>
    <ion-navbar>
      <ion-title>Opções</ion-title>
    </ion-navbar>
  </ion-header>

  <ion-content>
    <ion-list>
      <button menuClose ion-item (click)="abrir(home)">Home</button>
      <button menuClose ion-item (click)="abrir(modal)">Exemplo</button>
    </ion-list>
  </ion-content>
</ion-menu>

<ion-nav #paginas [root]="paginaAtual"></ion-nav>
```

menu.html

```
<ion-menu [content]="paginas">
  <ion-header>
    <ion-navbar>
      <ion-title>Opções</ion-title>
    </ion-navbar>
  </ion-header>

  <ion-content>
    <ion-list>
      <button menuClose ion-item (click)="abrir(home)">Home</button>
      <button menuClose ion-item (click)="abrir(modal)">Exemplo</button>
    </ion-list>
  </ion-content>
</ion-menu>

<ion-nav #paginas [root]="paginaAtual"></ion-nav>
```

menu.html

```
<ion-menu [content]="paginas">
  <ion-header>
    <ion-navbar>
      <ion-title>Opções</ion-title>
    </ion-navbar>
  </ion-header>

  <ion-content>
    <ion-list>
      <button menuClose ion-item (click)="abrir(home)">Home</button>
      <button menuClose ion-item (click)="abrir(modal)">Exemplo</button>
    </ion-list>
  </ion-content>
</ion-menu>

<ion-nav #paginas [root]="paginaAtual"></ion-nav>
```

menu.html

```
<ion-menu [content]="paginas">
  <ion-header>
    <ion-navbar>
      <ion-title>Opções</ion-title>
    </ion-navbar>
  </ion-header>

  <ion-content>
    <ion-list>
      <button menuClose ion-item (click)="abrir(home)">Home</button>
      <button menuClose ion-item (click)="abrir(modal)">Exemplo</button>
    </ion-list>
  </ion-content>
</ion-menu>

<ion-nav #paginas [root]="paginaAtual"></ion-nav>
```

menu.html

```
<ion-menu [content]="paginas">
  <ion-header>
    <ion-navbar>
      <ion-title>Opções</ion-title>
    </ion-navbar>
  </ion-header>

  <ion-content>
    <ion-list>
      <button menuClose ion-item (click)="abrir(home)">Home</button>
      <button menuClose ion-item (click)="abrir(modal)">Exemplo</button>
    </ion-list>
  </ion-content>
</ion-menu>

<ion-nav #paginas [root]="paginaAtual"></ion-nav>
```

menu.html

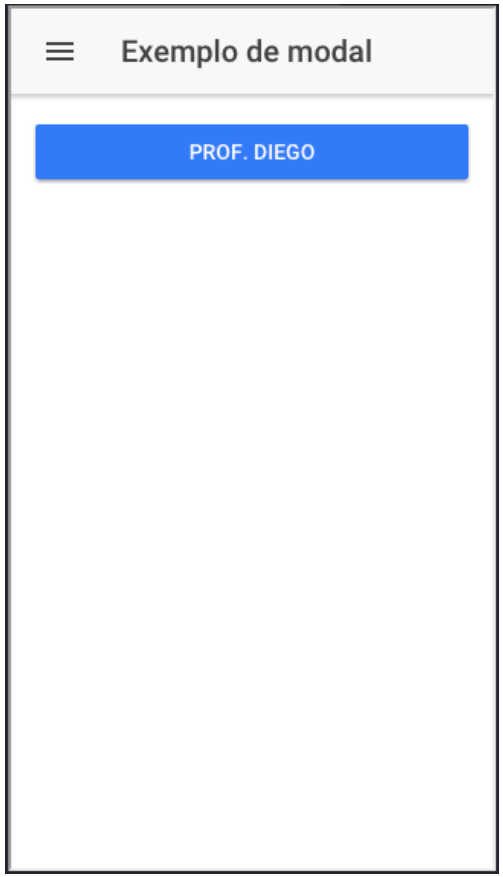
```
<ion-menu [content]="paginas">
  <ion-header>
    <ion-navbar>
      <ion-title>Opções</ion-title>
    </ion-navbar>
  </ion-header>

  <ion-content>
    <ion-list>
      <button menuClose ion-item (click)="abrir(home)">Home</button>
      <button menuClose ion-item (click)="abrir(modal)">Exemplo</button>
    </ion-list>
  </ion-content>
</ion-menu>

<ion-nav #paginas [root]="paginaAtual"></ion-nav>
```

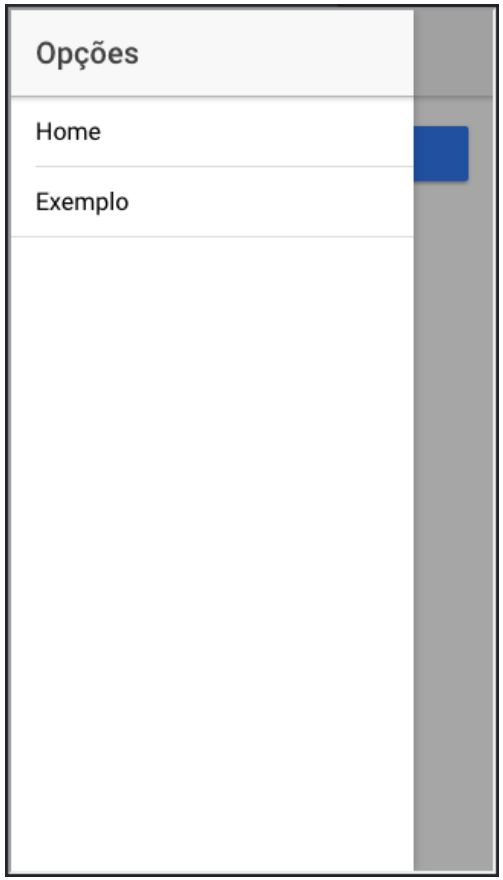
Agora, precisamos apenas adicionar um ícone ou botão para abrir o menu no cabeçalho das páginas de destino, no HTML. Por exemplo, a pagina home.html:

```
<ion-header>
  <ion-navbar>
    <button menuToggle>
      <ion-icon name="menu"></ion-icon>
    </button>
    <ion-title>
      Gestures
    </ion-title>
  </ion-navbar>
</ion-header>
```

Então, app irá iniciar e carregar a página menu primeiro.

Menu é um componente de navegação, que irá chamar no ion-nav a página inicial que definimos (Home).





Lista 2: Moodle

Os exercícios da Lista 1 e 2 podem ser apresentados até 02/09