


## Aula 04 – Programação de Dispositivos Móveis

Professor: Me. Wellington Tuler Moraes  
Curso: ADS – IFSP – Campus Cubatão

## Navegação

- O Xamarin Forms fornece um diferente número de experiências de navegação dependendo do tipo de página que usamos em nossas aplicações.
- Na figura abaixo vemos os principais tipos de página que podemos usar no Xamarin Forms:



ContentPage   MasterDetailPage   NavigationPage   TabbedPage   CarouselPage

## Navegação


- Temos dois principais tipos de navegação entre páginas usados no Xamarin Forms:
  - Navegação Hierárquica** : Permite ao usuário se mover para baixo em uma pilha de páginas e depois subir novamente pelos níveis;
  - Navegação não Hierárquica ou Modal** : É uma tela de interrupção que requer uma ação do usuário, e pode ser descartada por um botão **Cancel**;

## Navegação Hierárquica


- A classe **NavigationPage** fornece a experiência de navegação hierárquica onde o usuário pode navegar entre as páginas para frente e para trás.
- Esta classe implementa a navegação em uma pilha de páginas (*LIFO- last in, first out*) onde a última página a entrar é a primeira a sair.

## Navegação Hierárquica

Para passar de uma página para outra, uma aplicação irá empurrar uma nova página para a pilha de navegação, onde se tornará a página ativa:

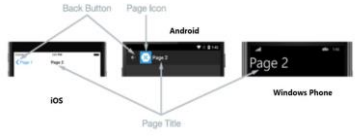


Para retornar à página anterior, o aplicativo puxará a página atual da pilha de navegação e a nova página mais alta se tornará a página ativa:



## Navegação hierárquica

- Na navegação hierárquica a classe **NavigationPage** é usada para navegar através de uma pilha de objetos **ContentPage** onde os principais componentes usados em cada plataforma é mostrado abaixo:



Back Button   Page Icon   Page Title   Page 2

iOS   Android   Windows Phone

## Fluxo de Navegação

- Vamos supor um cenário onde temos duas páginas: **Pagina1.xaml** e **Pagina2. xaml**
- A seguir veremos o fluxo da navegação hierárquica usado para navegar entre as páginas:

1 - A primeira página adicionada a uma pilha de navegação é referida como a página raiz do aplicativo :

```
public App ()
{
    MainPage = new NavigationPage (new Pagina1());
}
```

## Fluxo de Navegação

- Isso faz com que a instância **ContentPage** de Pagina1 seja empurrada para a pilha de navegação, onde ela se torna a página ativa e a página raiz da aplicação.

2 - Para navegar para **Pagina2**, é necessário invocar o método **PushAsync** na propriedade **Navigation** da página atual:

```
Async void OnNextPageButtonClicked (objeto remetente,
EventArgs e)
{
    await Navigation.PushAsync (new Pagina2());
}
```

## Fluxo de Navegação

- Isso faz com que a instância de **Pagina2** seja empurrada para a pilha de navegação, onde ela se torna a página ativa
- A página ativa pode ser extraída da pilha de navegação pressionando o botão **Voltar** no dispositivo, independentemente de se tratar de um botão físico no dispositivo ou de um botão na tela. Para retornar programaticamente à página original, a instância **Pagina2** deve invocar o método **PopAsync**:

```
Async void OnPreviousPageButtonClicked (objeto
remetente, EventArgs e)
{
    await Navigation.PopAsync ();
}
```

## Fluxo de Navegação

- Assim como os métodos **PushAsync** e **PopAsync**, a propriedade de navegação de cada página também fornece um método **PopToRootAsync** : Esse método remove todas, exceto a página raiz da pilha de navegação, tornando a página raiz do aplicativo a página ativa:

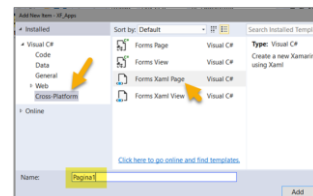
```
Async void OnRootPageButtonClicked (objeto remetente,
EventArgs e)
{
    await Navigation.PopToRootAsync ();
}
```

## Fluxo de Navegação

- Observe que os métodos para navegação são expostos pela propriedade **Navigation**.
- Vejamos a seguir um exemplo prático mostrando a navegação hierárquica.

## Exemplo

- Adicione duas páginas, a **Página1** e **Página2** XAML.



## Exemplo

```

<!-- References -->
public partial class App : Application
{
    <!-- References -->
    public App()
    {
        InitializeComponent();
        MainPage = new NavigationPage(new Pagina1());
    }

    <!-- References -->
    protected override void OnStart()
    {
        // Handle when your app starts
    }
}

<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://schemas.microsoft.com/winfx/2009/xaml"
              xmlns:xf="http://schemas.microsoft.com/winfx/2009/xaml"
              Title="Pagina 1">
    <StackLayout Padding="10">
        <Image Source="http://www.maccoratti.net/imagens/animals/canil.jpg"
              Aspect="AspectFit"
              WidthRequest="300"
              HeightRequest="300"/>
        <Button Text="Ir para Pagina 2" Clicked="btn_Clicked"/>
    </StackLayout>
</ContentPage>

<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://schemas.microsoft.com/winfx/2009/xaml"
              xmlns:xf="http://schemas.microsoft.com/winfx/2009/xaml"
              xmlns:Navigation="http://schemas.microsoft.com/winfx/2009/xaml"
              Title="Pagina 2">
    <StackLayout Padding="10">
        <Image Source="http://www.maccoratti.net/imagens/animals/canil.jpg"
              Aspect="AspectFit"
              WidthRequest="300"
              HeightRequest="300"/>
        <Button Text="Voltar para Pagina 1" Clicked="btn_Clicked"/>
    </StackLayout>
</ContentPage>

```

## Exemplo

No evento **Clicked** do Button inclua o código que vai permitir navegar para a **Pagina2** usando o método **PushAsync**:

```

private async void Button_Clicked(object sender, EventArgs e)
{
    await Navigation.PushAsync(new Pagina2());
}

```

No evento **Clicked** do Button temos o código que permite retornar para a **Pagina1** usando o método **PopAsync**:

```

private async void btn_Clicked(object sender, EventArgs e)
{
    await Navigation.PopAsync();
}

```

## Exemplo

