Macoratti.net Xamarin Android - Criando Action Bar Tabs - II

Este artigo mostrar como podemos criar **Actionbar Tabs** em uma aplicação Xamarin Android usando o Visual Studio com Xamarin e a linguagem C#.

Continuando o **artigo anterior**, hoje vou mostrar mais uma abordagem para criar **ActionBar tabs** em uma aplicação Xamarin Android.

Nesta abordagem vamos usar **Fragments** em combinação com a barra de ação(*Action Bar)* para a guia de navegação*(tab Navigation*). Podemos adicionar uma nova guia à **Action Bar** chamando o método **newTab().**

Para saber mais sobre **Fragments** leia o artigo : Xamarin Android - Apresentando e usando Fragments - Macoratti

O exemplo que vamos usar mostra uma **Activity** (atividade) com duas guias **"Áudio"** e **"Vídeo"**. Estamos usando **Fragments** e a barra de ações inclui suporte para adicionar interfaces com guias no Android 4.0.

Recursos usados:

- **Visual Studio Community 2017 ou Xamarin Studio**
- **Xamarin**
- **Emulador Android virtual ou físico (veja como emular usando o Vysor)**

Nota: Baixe e use a versão Community 2017 do VS ela é grátis e é equivalente a versão Professional.

Criando o projeto no VS Community 2017

Abra o VS 2017 Community e clique em **New Project;**

Selecione a linguagem Visual C# e o template **Android -> Blank App(Android)**

Informe o nome um nome adequado ao seu projeto, eu vou usar o nome **Droid_Tabs2**, e clique no botão **OK**;

Abra o arquivo **Main.axml** na pasta **Resources/layout** no modo **Source** e a seguir inclua o código abaixo:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:orientation="vertical"
  android:layout_width="match_parent"
  android:layout_height="match_parent">

 <FrameLayout
    android:id="@+id/fragmentContainer"
    android:layout_width="match_parent"
    android:layout_height="0dip"
    android:layout_weight="1" />

</LinearLayout>
```

Este código apenas inclui um **FrameLayout** usando o layout **LinearLayout**.

Agora podemos iniciar a implementação do código para exibir criar as tabs o arquivo **MainActivity.cs**.

Vamos começar definindo os namespaces usados no projeto:

```
using Android.App;
using Android.Widget;
using Android.OS;
using System;
```

A seguir a declaração da **Activity** como sendo a principal e a definição do ícone da aplicação:

```
[Activity(Label = "Droid_Tabs2", MainLauncher = true, Icon = "@drawable/icon")]
```

No método **OnCreate**() vamos incluir o código abaixo:

```
    protected override void OnCreate(Bundle
bundle)
    {
        base.OnCreate(bundle);
        SetContentView (Resource.Layout.Main);

        // habilita o modo de navegação para
suporar a tab layout
        this.ActionBar.NavigationMode =
ActionBarNavigationMode.Tabs;

        //adiciona a tab Audio
        AddTab("Audio", Resource.Drawable.Icon,
new AudioFragment());

        //adiciona a tab Video
        AddTab("Video", Resource.Drawable.Icon,
new VideoFragment());
    }
```

Para criar guias na Barra de Ação, primeiro precisamos definir sua propriedade **NavigationMode** para suportar guias :

```
ActionBar.NavigationMode = ActionBarNavigationMode.Tabs;
```

Depois chamamos o método **AddTab**() para criar as guias e também criamos a instância das classes **AudioFragment**() e **VideoFragment**() que iremos definir a seguir.

O código do método **AddTab**() pode ser visto abaixo:

```
    void AddTab(string tabText, int iconResourceId, Fragment fragment)
    {
        var tab = this.ActionBar.NewTab();
        tab.SetText(tabText);
        tab.SetIcon(iconResourceId);

        // define o tratamento de eventos para substituir as tabs
        tab.TabSelected += delegate (object sender, ActionBar.TabEventArgs e) {
            e.FragmentTransaction.Replace(Resource.Id.fragmentContainer, fragment);
        };

        this.ActionBar.AddTab(tab);
    }
```

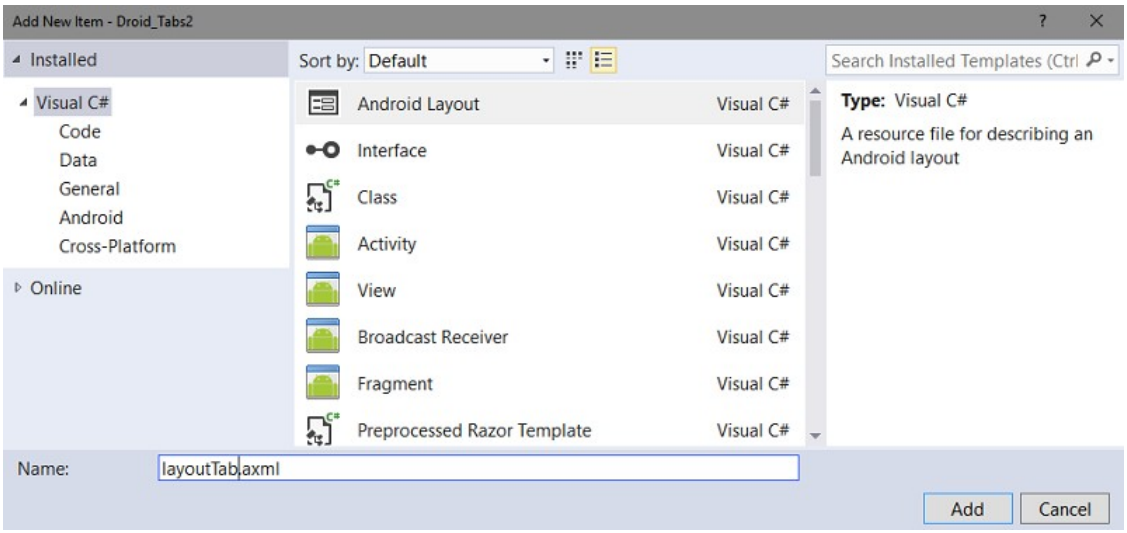
Para criar as guias chamamos o método **NewTab** na **Action Bar** e depois definimos o texto e o evento **tabSelected** que vai permitir alternar as guias conforme a seleção do usuário.

Definindo o arquivo de layout comun aos Fragments : layoutTab.axml

Vamos criar um novo arquivo de layout na pasta **Resources/layout** do projeto que vai atuar como um layout comum aos dois **Fragments** que iremos criar.

Clique com o botão direito do mouse sobre a pasta **Resources/layout** e a seguir clique em **Add -> New Item;**

A seguir clique no template **Android Layout** e informe o nome l**ayoutTab.axml** e clique no botão **Add**;



Inclua o código abaixo neste arquivo:

```
<?xml version="1.0" encoding="utf-8"?>
<TextView xmlns:android="http://schemas.android.com/apk/res/android"
   android:id="@+id/textView"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:gravity="center"
   android:textStyle="bold"
   android:textSize="16dp"
   android:background="#ffe3e1e1"
   android:textColor="#ff393939" />
```

Neste arquivo de leiaute apenas definimos um TextView com algumas propriedades.

Implementando os Fragment : AudioFragment e VideoFragment

Precisamos agorar definir o código das classes **AudioFragment** e **VideoFragment** que herdam de Fragment.

No menu **Project** clique em **Add Class** e informe o nome **AudioFragment** e digite o código abaixo nesta classe:

```
using Android.App;
using Android.OS;
using Android.Views;
using Android.Widget;

namespace Droid_Tabs2
{
   public class AudioFragment : Fragment
   {
      public override View OnCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)
      {
         base.OnCreateView(inflater, container, savedInstanceState);

         var view = inflater.Inflate(Resource.Layout.layoutTab, container, false);
         var meuTextView = view.FindViewById<TextView>(Resource.Id.textView);
         meuTextView.Text = "Audio Fragment";
         return view;
      }
   }
```

```
}
```

Repita o procedimento acima para criar a classe **VideoFragment** com o seguinte código :

```
using Android.App;
using Android.OS;
using Android.Views;
using Android.Widget;

namespace Droid_Tabs2
{
   public class VideoFragment : Fragment
   {
      public override View OnCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)
      {
         base.OnCreateView(inflater, container, savedInstanceState);

         var view = inflater.Inflate(Resource.Layout.layoutTab, container, false);
         var meuTextView = view.FindViewById<TextView>(Resource.Id.textView);
         meuTextView.Text = "Video Fragment";
         return view;
      }
   }
}
```

Ambas as classes herdam da classe Fragment.

Para retornar um layout de onCreateView(), é possível inflá-lo a partir de um recurso de layout(**layoutTab**) definido no XML. Para ajudar a fazer isto, o onCreateView() fornece um objeto **LayoutInflater**.

O parâmetro container passado para onCreateView() é o pai de ViewGroup *(do layout da atividade*) em que o layout do fragmento será inserido.

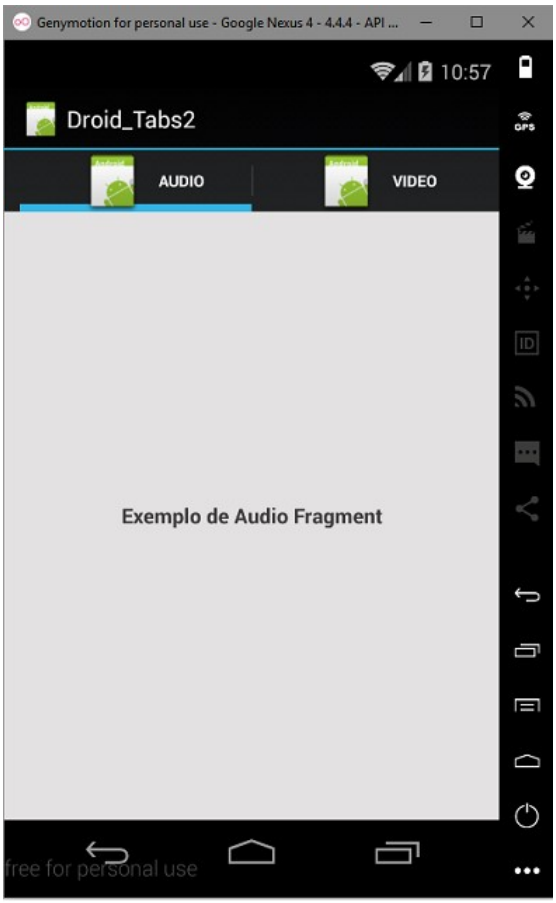
O parâmetro **savedInstanceState** é um Bundle que fornece dados sobre a instância anterior do fragmento, se o fragmento estiver sendo retomado.

O método **inflate()** usa três argumentos:

- O **ID** do recurso do layout que você quer inflar;(**layoutTab**)
- O ViewGroup que será o pai do layout inflado. Passar o container é importante para que o sistema aplique os parâmetros de layout à view raiz do layout inflado, especificado pela view pai em que está ocorrendo;
- Um **booleano** que indica se o layout inflado deve ser anexado à ViewGroup *(o segundo parâmetro)* durante a inflação *(neste caso, isto é falso, pois o sistema já está inserindo o layout inflado no container, retornar como verdadeiro criaria um grupo de vistas redundante no layout final)*.

Este é o modo de criar um fragmento que fornece um layout.

Executando o projeto iremos obter o seguinte resultado:



Pegue o projeto aqui : **Droid_Tabs2.zip** (*sem as referências*)

Jesus dizia, pois, aos judeus que criam nele: Se vós permanecerdes na minha palavra, verdadeiramente sereis meus discípulos;
E conhecereis a verdade, e a verdade vos libertará.
João 8:31,32

ReportViewer no VS 2013 ?

- **Curso - Gerando Relatórios com o ReportViewer no VS 2013 - Vídeo Aulas** NEW

Referências:

- Seção VB .NET do Site Macoratti.net
- Super DVD .NET - A sua porta de entrada na plataforma .NET
- Super DVD Vídeo Aulas - Vídeo Aula sobre VB .NET, ASP .NET e C#
- Super DVD C# - Recursos de aprendizagens e vídeo aulas para C#
- Seção C# do site Macoratti.net
- Seção ASP .NET do site Macoratti .net
- Curso Básico VB .NET - Vídeo Aulas
- Curso C# Básico - Vídeo Aulas
- Curso Fundamentos da Programação Orientada a Objetos com VB .NET NEW
- **Macoratti .net | Facebook**
- **macoratti - YouTube**
- **Jose C Macoratti (@macorati) | Twitter**
- Xamarim - Desenvolvimento Multiplataforma com C# ... - Macoratti.net
- Xamarin - Apresentando Xamarin.Forms - Macoratti.net
- Xamarin.Forms - Olá Mundo - Criando sua primeira ... - Macoratti.net
- Xamarin.Forms - Olá Mundo - Anatomia da aplicação - Macoratti.net
- https://developer.xamarin.com/api/type/Android.Widget.ListView/
- https://developer.xamarin.com/api/property/Android.Widget.ListView.Adapter/

José Carlos Macoratti