Macoratti.net Xamarin Android - Exibindo imagens em um componente GridView

Neste artigo vou mostrar como exibir imagens em um componente **GridView** customizado através da implementação da classe base **BaseAdapter**.

Neste artigo vou mostrar como exibir imagens em um **GridView** definindo uma classe que implementa a classe base **BaseAdapter** e fazendo uma vinculação ao **Adapter** (adaptador), que recupera os dados de uma fonte externa e cria uma exibição que representa cada entrada de dados.

Nota: *Veja o* ***artigo anterior*** *onde apresentei o controle GridView.*

Para personalizar o seu GridView você terá que implementar a classe abstrata **BaseAdapter** sobrescrevendo os seguintes métodos:

- **Count** - Informa ao controle quantas linhas estão nos dados.
- **GetView** - Retorna uma View para cada linha, preenchida com dados.
- **GetItemId** - Retorna um identificador de linha (normalmente o número da linha)
- **GetItem**- Retorna o item atual;

Vou exibir imagens de algumas bandeiras e o nome do respectivo país conforme mostra a imagem abaixo:

Recursos usados:

- **Visual Studio Community 2015 ou Xamarin Studio**
- **Xamarin**
- **Emulador Android virtual ou físico (veja como emular usando o Vysor)**

Nota: Baixe e use a versão Community 2015 do VS ela é grátis e é equivalente a versão Professional.

Criando o projeto no Visual Studio 2015 Community

Abra o VS 2015 Community e clique em **New Project;**

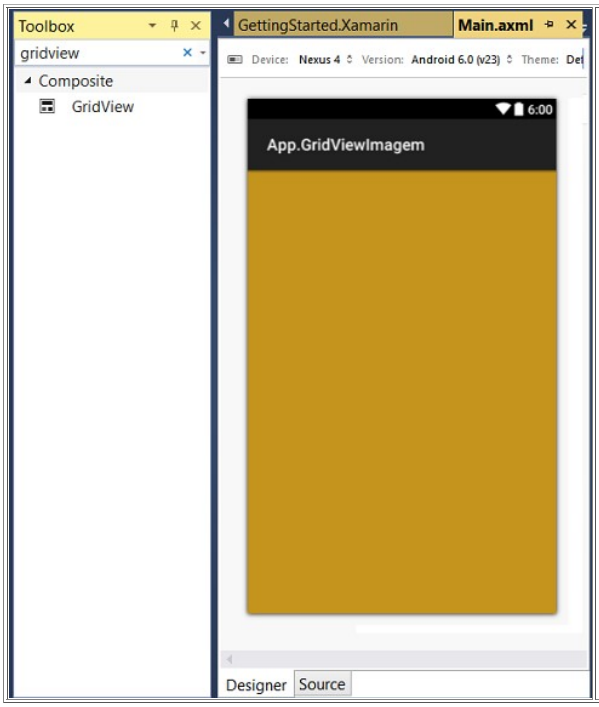
Selecione a linguagem Visual C# e o template **Android -> Blank App(Android)**

Informe o nome **App.GridViewImagem** e clique no botão **OK**;

Abra o arquivo **Main.axml** na pasta **Resources/layout** e no modo **Designer** incluir um novo controle **GridView** a partir da **ToolBox** e definir a seguintes propriedades:

- **id = @+id/gdvDados**

Abaixo vemos o leiaute no emulador do Xamarin exibindo a tela e ao lado o respectivo código XML gerado :




```
<?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"
   android:minWidth="25px"
   android:minHeight="25px">
   <GridView
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:numColumns="3"
      android:background="#c4941c"
      android:id="@+id/gvDados" />
</LinearLayout>
```

Definindo a classe de domínio : Bandeira.cs

No menu **Project** clique em **Add Class** e informe o nome **Bandeira.cs**.

A seguir inclua o código abaixo neste arquivo:

```
namespace App.GridViewImagem
{
   public class Bandeira
   {
      public Bandeira(string _nome, int _imagem)
      {
         this.Nome = _nome;
         this.Imagem = _imagem;
      }

      public string Nome { get; set; }
      public int Imagem { get; set; }
   }
}
```

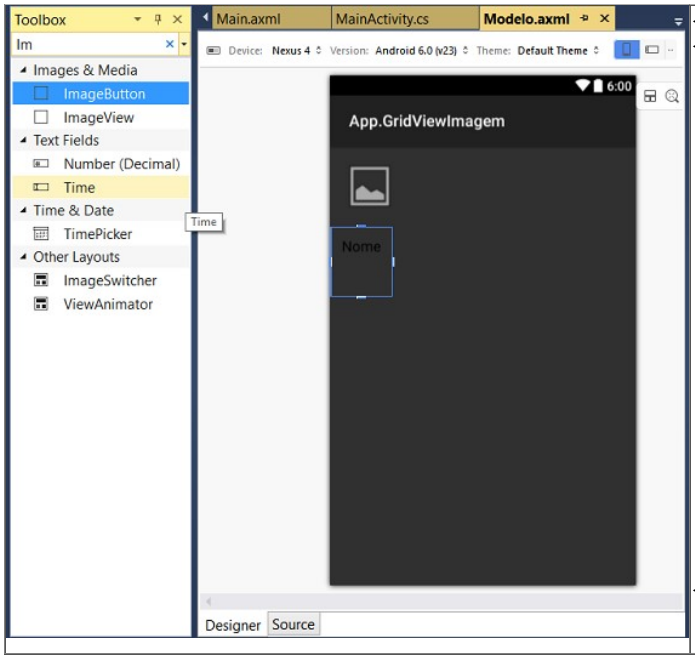
Esta classe representa os dados que vamos exibir no GridView.

Criando o Layout Modelo.axml

Vamos definir um layout que será usado para exibir as imagens no controle GridView.

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

Selecione o template **Layout** e informe o nome **Modelo.axml :**




```
<?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">
  <ImageView
    android:src="@android:drawable/ic_menu_gallery"
    android:layout_width="100dp"
    android:layout_height="100dp"
    android:layout_marginRight="0.0dp"
    android:padding="15dp"
    android:id="@+id/bandeiraImg" />
  <TextView
    android:text="Nome"
    android:textAppearance="?android:attr/textAppearanceMedium"
    android:layout_width="wrap_content"
    android:layout_height="90dp"
    android:padding="15dp"
    android:layout_toRightOf="@+id/bandeiraImg"
    android:layout_alignParentRight="true"
    android:textColor="#000"
    android:id="@+id/txtNome" />
</LinearLayout>
```

Este layout será usado para personalizar a exibição dos dados no GridView na implementação da classe **BaseAdapter**.

Criando a classe BandeiraAdapter que implementa BaseAdatper

No menu **Project** clique em **Add Class** e informe o nome **BandeiraAdapter**.cs e a inclua o código abaixo nesta classe:

```
using Android.Content;
using Android.Views;
using Android.Widget;
using System.Collections.Generic;

namespace App.GridViewImagem
{
   public class BandeiraAdapter : BaseAdapter
   {
      private Context ctx;
      private List<Bandeira> bandeiras;
      private LayoutInflater layinfla;

      public BandeiraAdapter(List<Bandeira> _bandeiras, Context c)
      {
         this.bandeiras = _bandeiras;
         this.ctx = c;
      }

      public override int Count
      {
         get
         {
            return bandeiras.Count;
         }
      }

      public override Java.Lang.Object GetItem(int position)
      {
         return bandeiras[position].Nome;
      }

      public override long GetItemId(int position)
      {
         return position;
      }

      public override View GetView(int position, View convertView, ViewGroup parent)
      {
         if (layinfla == null)
         {
            layinfla = (LayoutInflater)ctx.GetSystemService(Context.LayoutInflaterService);
         }
```

```
            if (convertView == null)
            {
               convertView = layinfla.Inflate(Resource.Layout.Modelo, parent, false);
            }

            //vincula dados
            TextView txtNome = convertView.FindViewById<TextView>(Resource.Id.txtNome);
            ImageView Imagem = convertView.FindViewById<ImageView>(Resource.Id.bandeiraImg);

            txtNome.Text = bandeiras[position].Nome;
            Imagem.SetImageResource(bandeiras[position].Imagem);

            return convertView;
        }
    }
}
```

A classe **BaseAdapter** é uma classe abstrata usada para implementação de um **Adapter** que pode ser usado em um ListView, Spinner e Gridview.

A classe **BandeiraAdapter** herda de **BaseAdapter** e implementa os métodos :

- **GetItem()**
- **GetItemId()**
- **Count()**
- **GetView(int position, View convertView, ViewGroup parent)**
  - **position** - é o índice do item da view;
  - **convertView** - a view a ser usada;
  - **parent** - o pai da view;

Desses métodos o mais importante é o método **GetView**() que retorna uma view correspondendo aos dados a serem exibidos;

É dentro do método **GetView**() que vamos transformar o arquivo de layout **Modelo.axml** em uma view contendo o leiaute do item da lista usando o método **inflate** da classe **LayoutInflater**.

O código é muito usado :

```
if (layinfla == null)
{
   layinfla = (LayoutInflater)ctx.GetSystemService(Context.LayoutInflaterService);
}

if (convertView == null)
{
   convertView = layinfla.Inflate(Resource.Layout.Modelo, parent, false);
}
```

Este método cria uma nova view para cada bandeira adicionada ao **BandeiraAdapter**.

Quando ele for chamado, a **View** é passada, o que normalmente é um objeto reciclado, então temos uma verificação para ver se o objeto é nulo.

Se o objeto for nulo, uma **view** é instanciada e configurada com as propriedades desejadas para a apresentação dos itens. Após isso estamos prontos para usar a view na **Activity** principal.

Agora podemos iniciar a implementação do código para exibir itens no **GridView** no arquivo **MainActivity.cs**.

Definindo o código da classe MainActivity

Vamos começar definindo os namespaces usados no projeto:

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

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

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

Antes de implementar o método **OnCreate**() vamos definir as variáveis que iremos usar :

```
  GridView gv;
```

```
private BandeiraAdapter adapter;
private List<Bandeira> bandeiras;
```

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

```
protected override void OnCreate(Bundle bundle)
{
   base.OnCreate(bundle);

   // Set our view from the "main" layout resource
   SetContentView(Resource.Layout.Main);

   gv = FindViewById<GridView>(Resource.Id.gvDados);
   adapter = new BandeiraAdapter(GetBandeiras(), this);
   gv.Adapter = adapter;

   gv.ItemClick += Gv_ItemClick;

}
```

Por padrão, o **BandeiraAdapter** passa uma coleção de bandeiras obtida pelo método **GetBandeiras**() e o contexto da Activity atual (**this**).

Abaixo vemos o código do método **GetBandeiras**():

```
private List<Bandeira> GetBandeiras()
{
   bandeiras = new List<Bandeira>();

   Bandeira band;

   band = new Bandeira("Brasil", Resource.Drawable.brasil);
   bandeiras.Add(band);

   band = new Bandeira("França", Resource.Drawable.franca);
   bandeiras.Add(band);

   band = new Bandeira("India", Resource.Drawable.India);
   bandeiras.Add(band);

   band = new Bandeira("Itália", Resource.Drawable.italian);
   bandeiras.Add(band);

   band = new Bandeira("Japão", Resource.Drawable.japao);
   bandeiras.Add(band);

   band = new Bandeira("México", Resource.Drawable.mexico);
   bandeiras.Add(band);

   band = new Bandeira("USA", Resource.Drawable.usa1);
   bandeiras.Add(band);

   return bandeiras;
}
```

Neste código criamos uma lista de objetos bandeira com imagem e nome. As imagens estão sendo obtidas da pasta **Resources/drawable**.

Temos pois que incluir nesta pasta as imagens que vamos usar. Assim clique com o botão direito do mouse sobre a pasta **drawable** e a seguir em **Add Existing Item** e selecione as imagens que deseja exibir.

Concluindo usamos a propriedade **Adapter** que retorna o adaptador atualmente em uso nesta **GridView** e exibe os dados na view:

```
gv.Adapter = adpt;
```

Concluindo definindo o evento **ItemClick** do GridView para exibir o item selecionado:

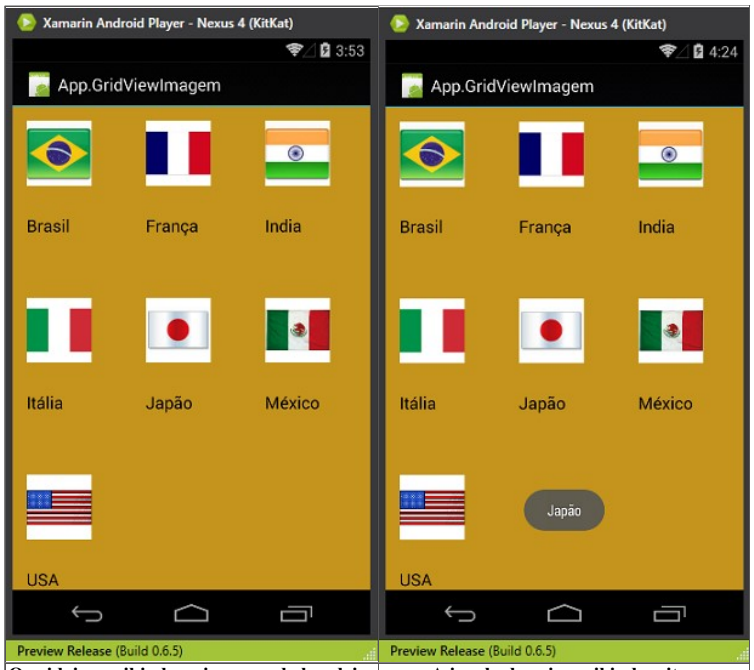
```
gv.ItemClick += Gv_ItemClick;
```

No evento **ItemClick** do **GridView** definimos uma janela de aviso usando a classe **Toast** para exibir qual o item foi selecionado :

```
private void Gv_ItemClick(object sender, AdapterView.ItemClickEventArgs e)
{
   Toast.MakeText(this, lista[e.Position].ToString(), ToastLength.Short).Show();
}
```

Para saber mais sobre diálogos de alerta e avisos veja o artigo : Xamarin Android - Exibindo uma janela de Alerta com AlertDialog.Builder

Executando o projeto iremos obter o seguinte resultado:



O gridview exibindo as imagens da bandeira e o nome de cada país	A janela de aviso exibindo o item selecionado

Aguarde em breve mais artigos sobre o componente **GridView** em aplicações Xamarin Android.

Pegue o projeto completo aqui : **App.GridViewImagem.zip** (*sem as referências*)

E os que são de Cristo crucificaram a carne com as suas paixões e concupiscências.
Gálatas 5:24

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