Macoratti.net Xamarin Android - Criando uma calculadora flex : Álcool ou Gasolina ?

Neste artigo vou mostrar como criar uma calculadora simples que verifica se é vantagem usar álcool ou gasolina usando o Visual Studio e o Xamarin.

No artigo de hoje vamos criar uma aplicação Android bem simples para aprender a trabalhar com alguns conceitos importantes existentes em aplicações Android.

Vamos criar uma aplicação que vai usar uma **view** bem simples composta dos controles **TextView , EditText e Button** e que vai receber como entrada do usuário os valores para o litro do álcool e da gasolina; a seguir vai realizar o cálculo e dependendo do resultado irá exibir ao usuário a mensagem de qual combustível compensa usar.

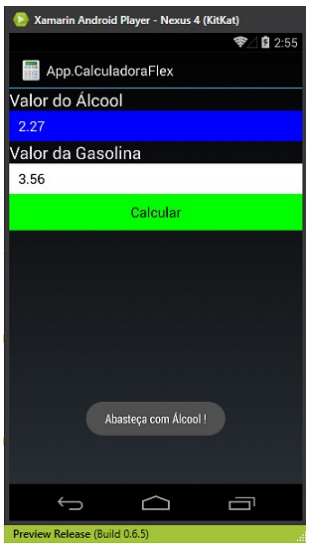
Conforme a ANP só vale a pena abastecer com álcool se o valor for até 70% do valor da gasolina, caso contrário, é melhor abastecer com gasolina.

Basta então dividir o valor do preço do álcool pelo valor do preço da gasolina e verificar se o resultado é menor que 0,7. Neste caso abasteça com com álcool caso contrário abasteça com gasolina.

Nessa aplicação vamos aprender a usar os seguintes recursos:

- **Criar uma interface com os componentes : TextView, EditText e Button**
- **Definir o ícone da aplicação**
- **Receber a entrada do usuário**
- **Definir o código da atividade para tratar os valores informados e realizar o cálculo**
- **Exibir uma mensagem ao usuário informando o resultado usando um Alerta e uma mensagem rápida** (**Toast**)

A aplicação em execução deverá ter a seguinte aparência :



Vamos ao que interessa...

Recursos usados:

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

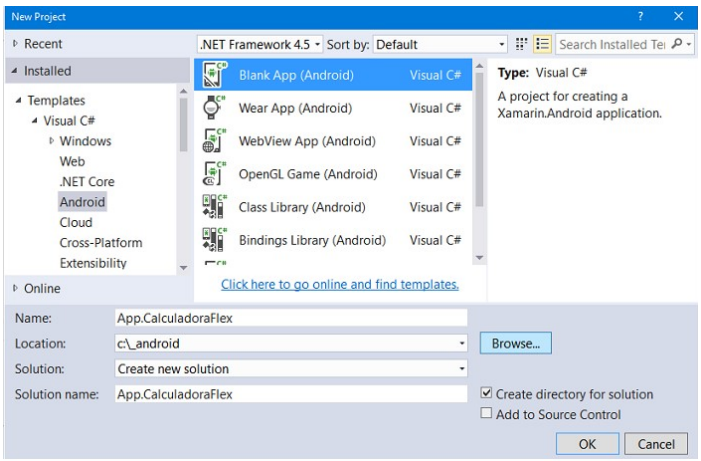
Nota: Baixe e use a versão Community 2015 do VS ela é grátis e é equivalente a versão Professional.

Criando o projeto no Visual Studio 2015 Community

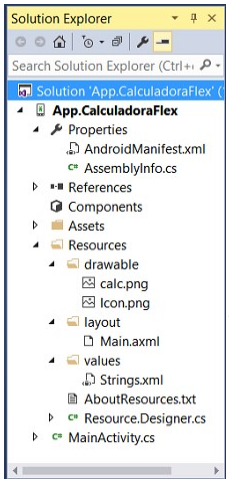
Abra o VS 2015 Community e clique em **New Project;**

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

Informe o nome **App1.Alertas** e clique no botão **OK**;



Será criada uma solução com a seguinte estrutura:



- **Properties** - Contém o arquivo AndroidManifest.xml que descreve as funcionalidades e requisitos da sua aplicação Android, e o arquivo AssemblyInfo.cs contém informação sobre o projeto como número de versão e build.

- **References** - Contém as bibliotecas Mono.Android, System.Core e todas as bibliotecas usadas no seu projeto;

- **Components** - Contém componentes de terceiros ou desenvolvidos por você usados no seu projeto.

A maioria dos componentes está disponíveis diretamente do **Xamarin Component Store** e são **free** *(não todos)* e prontos para serem usados; (Para incluir um componente clique com o botão direito sobre **Components** e a seguir em Get More Components);

- **Assets e Resources** - Contém arquivos que não são código, como imagens, sons, arquivos XML e qualquer outro recurso que sua aplicação for usar. Os arquivos externos colocados na pasta **Assets** são facilmente acessíveis em tempo de execução através do Asset Manager.

Já os arquivos colocados na pasta **Resources** precisam ser declarados e mantidos em uma lista com os IDs dos recursos que você desejar usar em tempo de execução.

De forma geral, todas a imagens, ícones, sons e outros arquivos externos são colocados na pasta **Resources** enquanto que dicionários e arquivos XML são postos na pasta **Assets**;

Na subpasta **layout** temos os arquivos .axml que definem as views usadas no projeto;

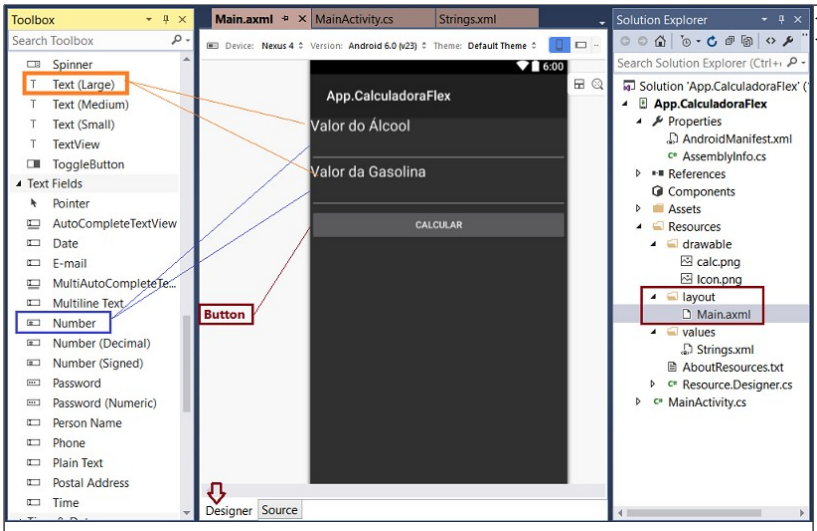
Na subpasta **values** temos o arquivo Strings.xml onde definimos as strings usadas no projeto;

Nota : A pasta **Drawable** contém recursos como imagens png, jpg, etc., usadas no aplicativo. Ela contém múltiplas pastas especificas para cada resolução possível em uma aplicação Android. Numa aplicação típica Android você vai acabar encontrando as pastas: Drawable-LDPI, Drawable-mdpi, Drawable-hdpi, Drawable-xhdpi, Drawable-xxhdpi, etc.

1- Vamos Abrir o arquivo **Main.axml** na pasta **Resources/layout** e no modo **Designer** incluir a partir da **ToolBox** os seguintes componentes:

• **TextView** • **id = @+id/txtValorAlcool** • **text = 'Valor do Álcool'** • **TextView** • **id = @+id/txtValorGasolina** • **text = 'Valor da Gasolina'**	• **EditText** • **id = @+id/valor_alcool** • **inputType="numberDecimal"** • **EditText** • **id = @+id/valor_gasolina** • **inputType='numberDecimal'**	• **Button** • **id = @+id/btnCalcular** • **text = @string/btnCalcular**

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">
  <TextView
    android:text="Valor do Álcool"
    android:textAppearance="?android:attr/textAppearanceLarge"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/txtValorAlcool" />
  <EditText
    android:inputType="numberDecimal"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/valor_alcool" />
  <TextView
    android:text="Valor da Gasolina"
    android:textAppearance="?android:attr/textAppearanceLarge"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/txtValorGasolina" />
  <EditText
    android:inputType="numberDecimal"
```

A partir da **ToolBox** arrastamos e soltamos os controles:

- **Text(Large)** - Valor do Álcool e Valor da Gasolina - **Number** - para receber a entrada do usuário - **Button** - para interagir com o usuário via evento **Click** cujo código iremos definir no arquivo MainActivity	android:layout_width="match_parent" android:layout_height="wrap_content" android:textColor="#000" android:id="@+id/**valor_gasolina**" /> <Button android:text="@string/btnCalcular" android:layout_width="match_parent" android:layout_height="wrap_content" android:id="@+id/**btnCalcular**" /> </**LinearLayout**>

Vamos agora abrir o arquivo **Strings.xml** na pasta **Resources/values** e definir o texto que será exibido no botão de comando:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
   <string name="btnCalcular">Calcular</string>
   <string name="ApplicationName">App.CalculadoraFlex</string>
</resources>
```

Agora podemos iniciar a implementação do código para criar as janelas de alertas no arquivo **MainActivity.cs**.

Veja como deve ficar o código do arquivo **MainActivity.cs**: (*O código em azul foi o que incluímos)*

1- Primeiro a declaração dos namespaces usados

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

2- A declaração das variáveis dos tipos dos objetos usados e definidos na View **Main.axml:**

```
EditText vlrAlcool;
EditText vlrGasolina;
TextView txtAlcool;
TextView txtGasolina;
Button btnCalcular;
```

Na declaração do atributo **Activity** do método **OnCreate**() definimos o ícone da aplicação conforme mostrado abaixo:

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

Naturalmente incluímos o arquivo **calc.png** na pasta **Resources/drawable.**

3- No método **OnCreate**() criamos instâncias de cada um dos componentes usados e alteramos a cor de texto e a cor de fundo dos componentes usando os métodos **SetTextColor**() e **SetBackGroundColor**().

Também criamos uma instância do controle Button e definimos o seu evento **Click**:

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

   SetContentView(Resource.Layout.Main);

   txtAlcool = FindViewById<TextView>(Resource.Id.txtValorAlcool);
   txtAlcool.SetTextColor(Android.Graphics.Color.White);

   txtGasolina = FindViewById<TextView>(Resource.Id.txtValorGasolina);
   txtGasolina.SetTextColor(Android.Graphics.Color.White);

   vlrAlcool = FindViewById<EditText>(Resource.Id.valor_alcool);
   vlrAlcool.SetBackgroundColor(Android.Graphics.Color.Blue);
   vlrAlcool.SetCursorVisible(true);

   vlrGasolina = FindViewById<EditText>(Resource.Id.valor_gasolina);
   vlrGasolina.SetBackgroundColor(Android.Graphics.Color.White);
   vlrGasolina.SetCursorVisible(true);

   btnCalcular = FindViewById<Button>(Resource.Id.btnCalcular);
   btnCalcular.SetBackgroundColor(Android.Graphics.Color.Green);
   btnCalcular.SetTextColor(Android.Graphics.Color.Black);

   btnCalcular.Click += BtnCalcular_Click;
}
```

4- No evento **Click** do botão de comando estamos obtendo o valor informando pelo usuário nos campos **vlrAlcool** e **vlrGasolina** e realizando o cálculo para verificar qual é mais indicado usar :

```
double resultado = (valor_Alcool / valor_Gasolina);
```

A seguir com base no resultado chamamos o método **ExibeMensagem**() passando uma string que vai ser a mensagem exibida ao usuário:

```
private void BtnCalcular_Click(object sender, System.EventArgs e)
{
   double valor_Alcool = double.Parse(vlrAlcool.Text);
   double valor_Gasolina = double.Parse(vlrGasolina.Text);
```

```
        double resultado = (valor_Alcool / valor_Gasolina);

        if (resultado > 0.70)
        {
            ExibeMensagem("Abasteça com Gasolina !");
        }
        else
        {
            ExibeMensagem("Abasteça com Álcool !");
        }
    }
```

5- O método **ExibeMensagem**() exibe um alerta usando a classe classe **AlertDialog.Builder** e exibe também uma pequena mensagem que se desvanece usando a classe **Toast.**

```
    private void ExibeMensagem(string texto)
    {
        //define o alerta para executar a tarefa
        AlertDialog.Builder builder = new AlertDialog.Builder(this);
        AlertDialog alerta = builder.Create();
        //Define o Titulo
        alerta.SetTitle("Calculadora Flex");
        //define o icone
        alerta.SetIcon(Android.Resource.Drawable.IcDialogAlert);
        //define o texto
        alerta.SetMessage(texto);
        //define o button
        alerta.SetButton("OK", (s, ev) =>
        {
            //define uma mensagem que se desvanece
            Toast.MakeText(this, texto, ToastLength.Short).Show();

        });
        alerta.Show();
    }
```

1- Para criar uma instância da classe **AlertDialog.Builder**

AlertDialog.Builder builder = new AlertDialog.Builder(this); e AlertDialog.Builder alerta = new AlertDialog.Builder(this);

3- Definimos o título usando o método **SetTittle**() e os ícones : **IcDialogAlert e IcInputAdd,** usando o método **SetIcon()**;

4- Definimos a mensagem usando o método **SetMessage**()

5- Para o segundo botão definimos o primeiro botão usando - **SetPositiveButton**() e o segundo botão usando - **SetNegativeButton**()

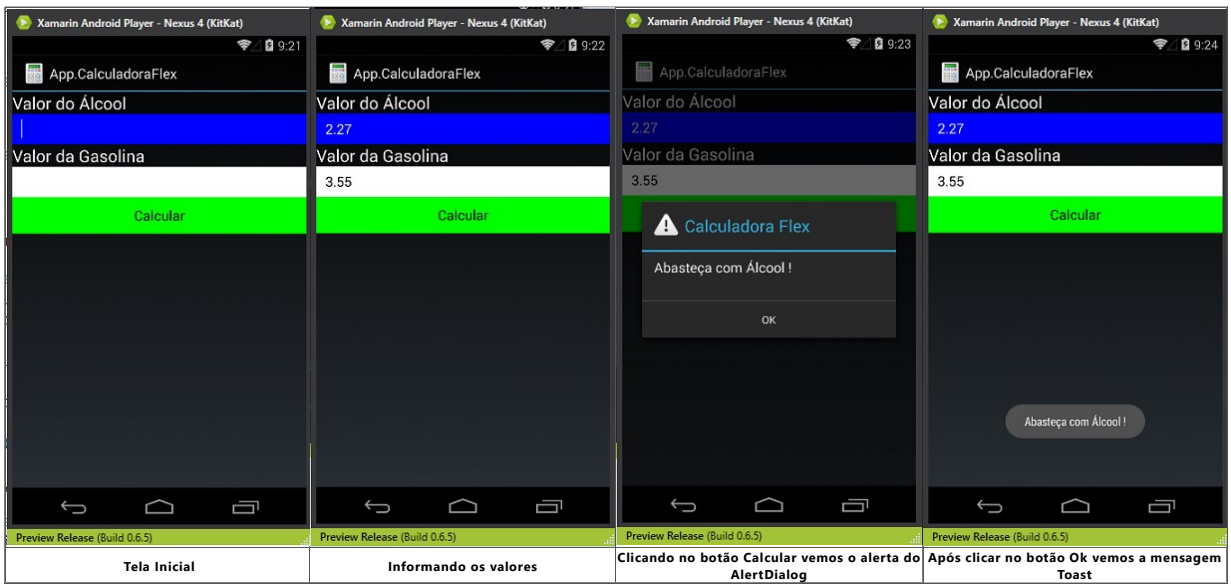
6- Exibimos o alerta : **alerta.Show**

Estamos usando também o recurso do método **Toast.MakeText** que cria uma pequena mensagem de retorno da operação exibindo um texto em um pequeno *pop up* durante um certo tempo.

Ex: **Toast.MakeText(this, "Cancelado !", ToastLength.Short).Show();**

- **this** : representa o contexto
- A seguir vem o texto a ser exibido : **'Macoratti .net'**
- A enumeração **ToastLength** define o tempo de exibição que pode ser : Long ou Short.

Executando o projeto e debugando em um emulador Android Xamarin Player - Nexus 4 (KitKat) - iremos obter o seguinte resultado:



Tela Inicial	Informando os valores	Clicando no botão Calcular vemos o alerta do AlertDialog	Após clicar no botão Ok vemos a mensagem Toast

Aguarde mais artigos sobre a criação de aplicações com Xamarin.Android.

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

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.App.AlertDialog/
- https://developer.xamarin.com/api/type/Android.Widget.Toast/

José Carlos Macoratti