

Macoratti.net Xamarin Android - Localizando a sua aplicação - I



Curso de Xamarin Forms Vídeo Aulas

Desenvolva para Android, iOS e Windows Phone



Neste artigo vou mostrar com podemos **localizar** uma aplicação Android usando o **Xamarin Android**, o Visual Studio 2015 e a linguagem C#.

Curso C# Vídeo Aulas

Do básico ao intermediário

Por um preço justo

A internacionalização é o processo de tornar seu código capaz de exibir diferentes idiomas e de adaptar a sua exibição para diferentes localidades (como a formatação de números e datas). Isso também é conhecido como globalização.

O objetivo principal do esforço de **localização** é a tradução dos recursos na interface do usuário, e, isso é um recurso específico de cada aplicação que devido a grande variedade de culturas e idiomas deve decidir se adota ou não uma implementação genérica.

O mundo é composto de uma infinidade de culturas, cada qual tem uma linguagem e um conjunto de formas definidas para tratar formatos de números, utilizar moedas, realizar classificações, etc.

A plataforma .NET define idiomas e regiões usando a definição padrão do [RFC\(Request for Comments\) 1766](http://www.ietf.org/rfc/rfc1766.txt) (<http://www.ietf.org/rfc/rfc1766.txt>), que especifica um idioma e região utilizando códigos de duas letras separadas por um traço.

A tabela a seguir fornece exemplos de algumas definições de cultura:

Código da Cultura	Descrição
en-US	English Language; United States
en-GB	English Language; United Kingdom
en-AU	English Language; Australia
en-CA	English Language; Canada
fr-CA	French Language; Canada

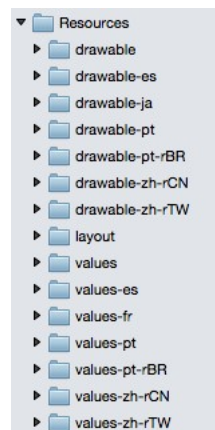
A estratégia de **localização** do Android envolve os seguintes recursos :

- Definir as **pastas de recursos** para conter as strings , imagens e outros recursos que serão localizados;
- A utilização dos métodos **GetText** e/ou **GetString**, que são usados para recuperar as strings localizadas no código;
- A utilização de um identificador único no formato **@String/id** em arquivos **AXML**, para colocar automaticamente strings localizadas em layouts;

Uma das possibilidades de localização em aplicações Android é usar a pasta **Resources** para definir a localização dos recursos a serem localizados. E nessa pasta criar arquivos para cada localização.

Para realizar a tradução para diversos idiomas podemos criar pastas com sufixos que identificam o idioma e a cultura e nessas pastas definir os recursos que vamos localizar.

Assim, para a localização de strings, podemos criar pastas **values** no interior da pasta **Resources** identificando-as com o respectivo sufixo para a cultura desejada da localização: **values-es** , **values-pt**, etc.



Dentro dessas pastas devemos ter um arquivo **Strings.xml** que vai conter o texto traduzido para a respectiva localização.

Cada string traduzível é um elemento XML com o ID do recurso especificado com o atributo **name** e o **valor** da string traduzida:

```
<?xml version="1.0" encoding="utf-8" ?>
<resources>
  <string name="app_nome">Xamarin Android</string>
</resources>
```

Para referenciar as strings localizadas em arquivos de layout, usamos a sintaxe **@string/id** na definição da propriedade **android:text** do respectivo controle:

```
<TextView
  android:id="@+id/NameLabel"
  android:text="@string/app_nome"
... />
```

Para recuperar as strings traduzidas no código, use o método **GetText** e passe o ID do recurso usado :

```
Resources.GetText(Resource.String.app_nome);
```

ou utilize o método **GetString** para obter o valor traduzido do recurso:

```
GetString(Resource.String.app_nome)
```

Vamos então aplicar esses conceitos na prática.

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 VS Community 2015

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

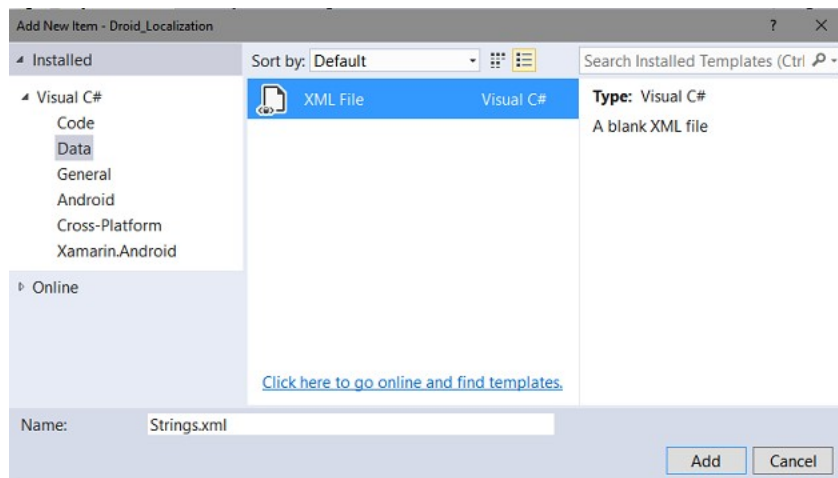
Criando as pastas para localizar os textos dos controles

Abra a pasta **Resources** e no seu interior crie outra pasta chamada **values-pt** onde vamos definir os textos para o idioma português.

Clique com o botão direito do mouse sobre a pasta **Resources** e a seguir em **Add -> New Folder** e informe o nome **values-pt**.

A seguir clique com o botão direito do mouse sobre a pasta **values-pt** e a seguir em **Add->New Item**;

A seguir selecione o template **XML File** e informe o nome **Strings.xml** :



A seguir defina o seguinte conteúdo para o arquivo **Strings.xml** da pasta **values-pt** :

```
<?xml version="1.0" encoding="utf-8" ?>
<resources>
  <string name="app">Localizacao Portugues</string>
  <string name="add">Incluir Tarefa</string>
  <string name="notes">Notas sobre a tarefa</string>
  <string name="done">Tarefa Concluída</string>
  <string name="cancel">Cancelar Tarefa</string>
</resources>
```

Definimos as propriedades names para **app**, **add**, **notes**, **done** e **cancel** e seu respectivo valor que será exibido no idioma português.

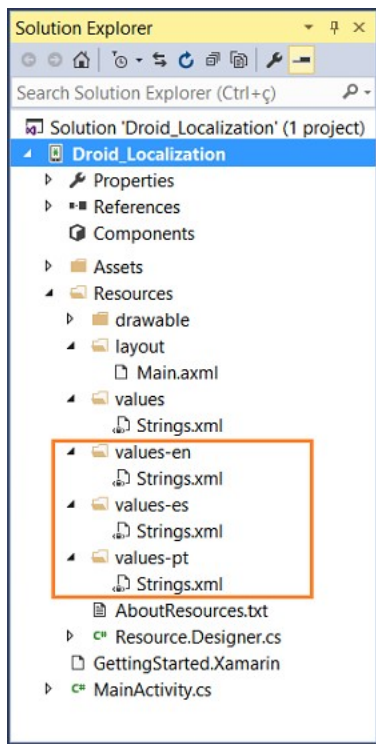
Repita os procedimentos acima e crie outra pasta chamada **values-es** que deverá conter o texto em **Espanhol**. Crie o arquivo **Strings.xml** e defina nele o seguinte conteúdo:

```
<?xml version="1.0" encoding="utf-8" ?>
<resources>
  <string name="app">Localizacion Espanhol</string>
  <string name="add">Agregar tarea</string>
  <string name="notes">Notas tarea</string>
  <string name="done">Tarea Completa</string>
  <string name="cancel">Cancelar la Tarea</string>
</resources>
```

Repita os procedimentos novamente e crie a pasta **values-en** para o texto em **Inglês** e defina o conteúdo do arquivo **Strings.xml** conforme abaixo:

```
<?xml version="1.0" encoding="utf-8" ?>
<resources>
  <string name="app">Localization English</string>
  <string name="add">Add Task</string>
  <string name="notes">Notes about Task</string>
  <string name="done">Task Done</string>
  <string name="cancel">Cancel Task</string>
</resources>
```

Ao final nosso projeto deverá apresentar os seguintes arquivos incluídos na pasta **Resources**:



Dessa forma já temos os textos traduzidos para os idiomas [português](#), [espanhol](#) e [inglês](#) e podemos usá-los em nossa aplicação.

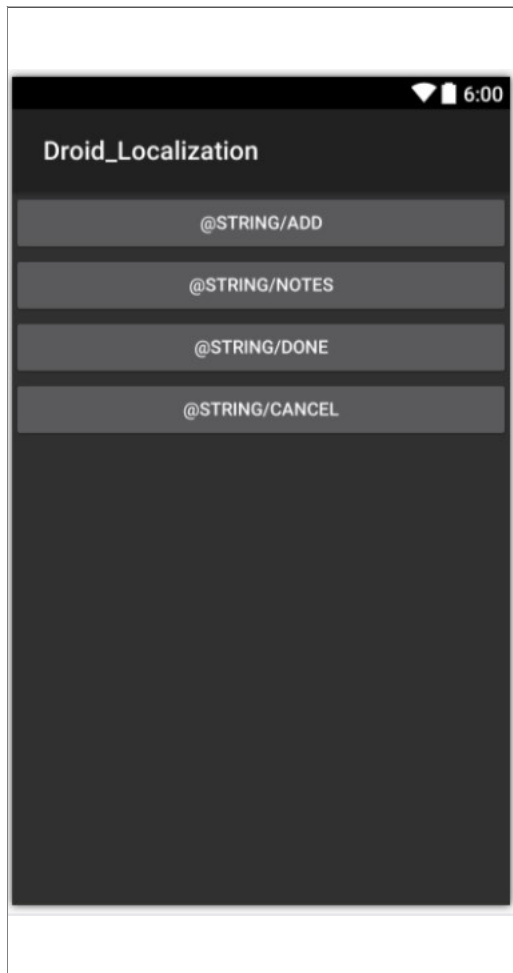
Observe que a propriedade **name** é a mesma em todos os arquivos somente o seu valor muda correspondendo à tradução do respectivo idioma.

Definindo o arquivo de layout da aplicação

Abra o arquivo **Main.xml** na pasta **Resources/layout** no modo **Designer** e a seguir inclua 4 controles [Button](#):

- 1 Button - id => **btnAdd**, **btnNotes**, **btnDone** e **btnCancel**

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

	<pre><?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"> <Button android:text="@string/add" android:layout_width="match_parent" android:layout_height="wrap_content" android:id="@+id/btnAdd" /> <Button android:text="@string/notes" android:layout_width="match_parent" android:layout_height="wrap_content" android:id="@+id/btnNotes" /> <Button android:text="@string/done" android:layout_width="match_parent" android:layout_height="wrap_content" android:id="@+id/btnDone" /> <Button android:text="@string/cancel" android:layout_width="match_parent" android:layout_height="wrap_content" android:id="@+id/btnCancel" /> </LinearLayout></pre>
---	--

No arquivo de layout definimos cada propriedade **android:text** dos Buttons usando o id correspondente à propriedade **name** do arquivo de recursos definido nas pastas de recursos.

Definindo o código da MainActivity

Abra o arquivo **MainActivity** e inclua o código abaixo :

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

namespace Droid_Localization
{
    [Activity(Label = "@string/app", MainLauncher = true, Icon = "@drawable/icon")]
    public class MainActivity : Activity
    {
        protected override void OnCreate(Bundle bundle)
        {
            base.OnCreate(bundle);

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

            var btnCancel = FindViewById<Button>(Resource.Id.btnCancel);
            var btnAdd = FindViewById<Button>(Resource.Id.btnAdd);
            var btnNotes = FindViewById<Button>(Resource.Id.btnNotes);
            var btnDone = FindViewById<Button>(Resource.Id.btnDone);

            if (btnCancel != null)
            {
                btnCancel.Text = GetString(Resource.String.cancel);
            }
        }
    }
}
```

```

    }
    if (btnAdd != null)
    {
        btnAdd.Text = Resources.GetText(Resource.String.add);
    }
    if (btnNotes != null)
    {
        btnNotes.Text = Resources.GetText(Resource.String.notes);
    }
    if (btnDone != null)
    {
        btnDone.Text = Resources.GetText(Resource.String.done);
    }
}
}
}

```

Este código apenas referencia o arquivo de Layout **Main.axml** e cria instâncias de cada um dos Buttons.

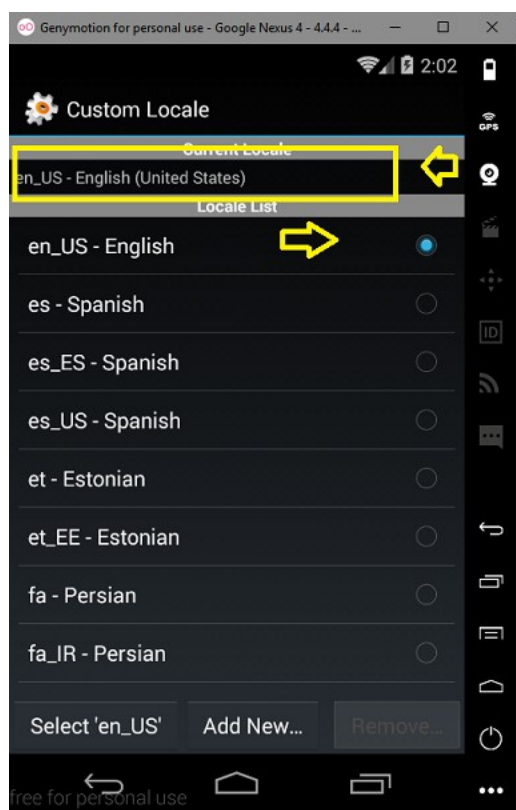
A seguir verificamos se a instância de cada Button não é **null** e atribuímos à sua propriedade **Text** o valor obtido do arquivo de recurso que criamos na pasta **Resources** usando o método **GetString()**.

```

if (btnCancel != null)
{
    btnCancel.Text = GetString(Resource.String.cancel);
}

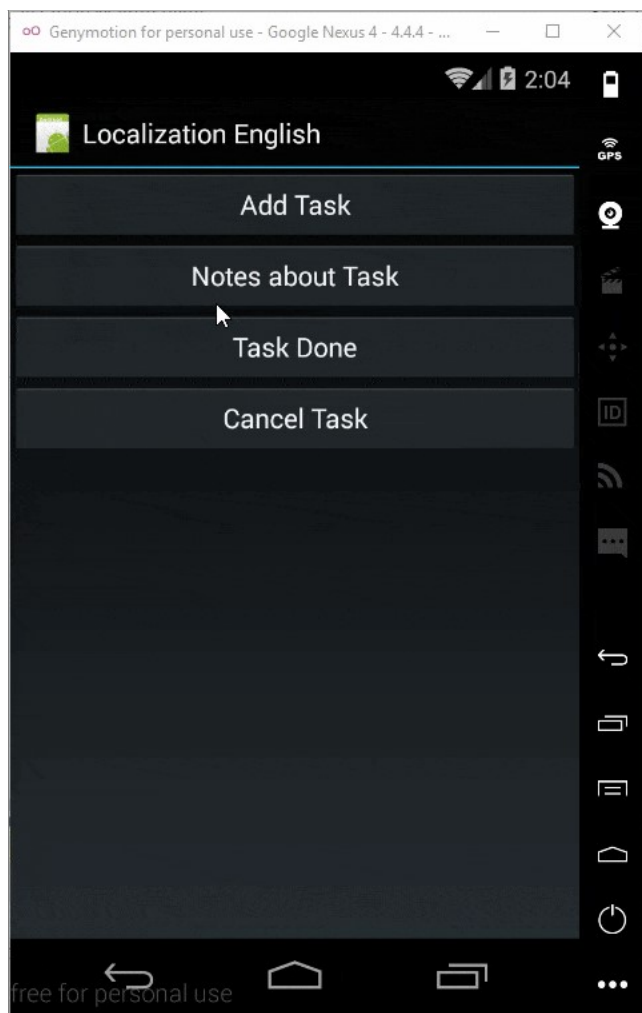
```

Quando o a aplicação for executada o dispositivo possui um idioma padrão que esta definido em **Custom Locale** conforme podemos ver abaixo:



Com base nessa informação, que podemos modificar, o respectivo arquivo do idioma local será usado para exibir os textos traduzidos.

Executando o projeto usando o emulador **Genymotion** iremos obter o seguinte resultado:



Essa abordagem pode ser usada para realizar a localização em pequenas aplicações. Na [segunda parte do artigo](#) vou mostrar outra maneira de obter o mesmo resultado.

Pegue o projeto aqui : [Droid Localization.zip](#) (sem as referências)

Não se turbe o vosso coração; credes em Deus, crede também em mim.

Na casa de meu Pai há muitas moradas; se não fosse assim, eu vo-lo teria dito. Vou preparar-vos lugar.

João 14:1,2

[Veja os Destaques e novidades do SUPER DVD Visual Basic \(sempre atualizado\) : clique e confira !](#)

Quer migrar para o VB .NET ?

- Veja mais sistemas completos para a plataforma .NET no [Super DVD .NET](#) , confira...
- [Curso Básico VB .NET - Vídeo Aulas](#)

Quer aprender C# ??

- Chegou o [Super DVD C#](#) com exclusivo material de suporte e vídeo aulas com curso básico sobre C#.
- [Curso C# Basico - Video Aulas](#)

Quer aprender os conceitos da Programação Orientada a objetos ?

- [Curso Fundamentos da Programação Orientada a Objetos com VB .NET](#) NEW

Quer aprender o gerar relatórios com o 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 \(@macoratti\) | Twitter](#)
- [VB.NET 2005 - Controles - Macoratti.net](#)
- [Seção de Jogos do site Macoratti .net](#)
- [Xamarin - 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/>
- [Xamarin Android - Tratando eventos de forma declarativa](#)
- [Seção Mobile/Xamarin do site Macoratti .net](#)
- <https://developer.android.com/reference/android/app/Activity.html>
- <https://developer.xamarin.com/api/type/Android.Widget.ProgressBar/>

[José Carlos Macoratti](#)