Macoratti.net Xamarin Forms - Usando Renderizadores Personalizados

Neste artigo vou mostrar como usar os Custom Renderers para criar controles customizados em aplicações Xamarin Forms usando os no Visual Studio 2017 e a linguagem C#.

No Xamarin Forms (XF) as interfaces de usuário (UI) são processadas usando os controles nativos da plataforma de destino permitindo assim que aplicações Xamarin Forms mantenham a aparência apropriada dos controles para cada plataforma.

Os Renderizadores personalizados ou **Custom Renderers** permitem aos desenvolvedores sobrescrever este processo para customizar a aparência e comportamento de controles Xamarin Forms para cada plataforma. Os Renderizadores personalizados são uma ponte entre o Xamarin Forms e as bibliotecas específicas de plataforma, o Xamarin.iOS e o Xamarin.Android e o Windows Phone SDK.

Os controles do XF são desenhados na tela usando dois componentes principais :

1. **Elementos** - views, páginas, células definidas;
2. **Renderizadores** - obtém um elemento multiplataforma e o desenham na tela usando a biblioteca de UI específica da plataforma;

Cada controle XF possui um renderizador para cada plataforma que cria uma instância do controle nativo. Assim, quando um controle **Entry** é renderizado em uma aplicação XF, ocorre o seguinte:

- No iOS a classe "**EntryRender**" é instanciada e cria uma instância do controle "**UITextField**" nativo;
- No Android a classe "**EntryRender**" instancia um controle "**EditText**";
- No Windows Phone a classe "**EntryRender**" instancia um controle "**UserControl**";

Podemos usar os Custom Renderers para personalizar controles da interface de usuário especificos para cada plataforma como views, páginas e células.

O processo para criar uma classe de **Custom Renderer** é o seguinte:

1. Criar uma subclasse da classe renderer que torna o controle nativo;
2. Sobrescrever o método que renderiza o controle nativo e escrever a lógica para personalizar o controle.
3. Adicionar um atributo **ExportRenderer** na classe renderer personalizada para especificar que ela será usada para processar o controle XF;

Neste artigo vamos criar um controle **Entry** personalizado usando os renderizadores personalizados.

Recursos usados:

- Visual Studio Community 2017 ou Xamarin Studio
- Xamarin

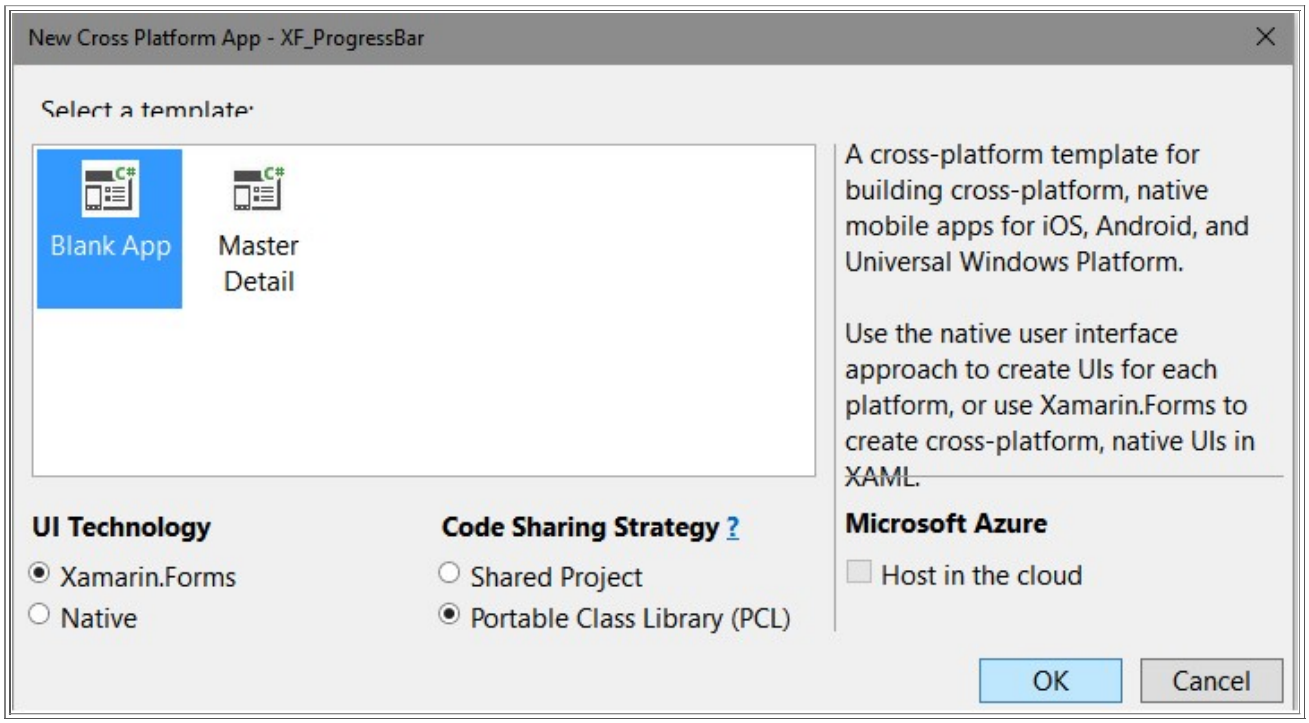
Criando o projeto no Visual Studio 2017 Community

Abra o Visual Studio Community 2017 e clique em **New Project;**

Selecione Visual C#, o template **Cross Plataform** e a seguir **Cross Plataform App(Xamarin.Forms or Native);**

Informe o nome **XF_CustomControls** e clique no botão OK;

A seguir selecione **Blank App** e marque as opções - **Xamarin.Forms** e **Portable Class Library (PCL**) e clique em OK;



Será criado um projeto contendo no projeto **Portable** as páginas **App.xaml e MainPage.xaml.**

Criando o controle personalizado Entry

Vamo iniciar criando no projeto uma classe chamada **MacEntry** que herda da classe **Entry**

```
using Xamarin.Forms;

namespace XF_CustomControls
{
    public class MacEntry : Entry
    { }
}
```

Criamos a classe **MacEntry** vazia no projeto PCL sendo ela aapenas representação do controle **Entry**. A customização do controle será feita no renderizador personalizado.

Criando o renderizador personalizado em cada plataforma (Android e iOS)

A seguir vamos realizar as seguintes tarefas para criar a classe do controle personalizado:

1. Criar uma subclasse da classe "**EntryRender**" que representa o controle nativo.
2 .Sobrescrever o método "**OnElementChanged**" que torna o controle nativo e escrever a lógica de controle personalizado. Este método é chamado, quando o controle Xamarin.forms correspondente for criado.
3. Adicionar um atributo "**ExportRender**" para a classe "**MacEntryRender**" para especificar que será utilizado para processar o controle Xamarin.Forms. Este atributo é usado para registrar o controle personalizado com Xamarin.Forms.

a-) Implementação do Custom Renderer no Android

A seguir temos a personalização do controle no projeto Android "**XF_CustomControls.Android**" e a criação de uma classe chamada "**MacEntryRender**" no código é mostrado abaixo:

```
using Xamarin.Forms;
using Xamarin.Forms.Platform.Android;
using XF_CustomControls;
using XF_CustomControls.Droid;

[assembly: ExportRenderer(typeof(MacEntry), typeof(MacEntryRender))]
namespace XF_CustomControls.Droid
{
    public class MacEntryRender : EntryRenderer
    {
```

```
        protected override void OnElementChanged(ElementChangedEventArgs<Entry> e)
        {
            base.OnElementChanged(e);
            if (Control != null)
                Control.SetBackgroundColor(global::Android.Graphics.Color.LightBlue);
        }
    }
}
```

A chamada ao método **OnElementChanged** da classe base instancia um controle **EditText** Android com a referência do controle sendo atribuida a propriedade **Control** do renderizador. A cor de fundo é então definida para **LightBlue** usando o método **Control.SetBackGroundColor().**

b-) Implementação do Custom Renderer no iOS

No projeto iOS "**XF_CustomControls.iOS**" criamos uma classe chamada '**MacEntryRender**' com o seguinte código:

```
using UIKit;
using Xamarin.Forms;
using Xamarin.Forms.Platform.iOS;
using XF_CustomControls;
using XF_CustomControls.iOS;

[assembly: ExportRenderer(typeof(MacEntry), typeof(MacEntryRender))]
namespace XF_CustomControls.iOS
{
    public class MacEntryRender : EntryRenderer
    {
        protected override void OnElementChanged(ElementChangedEventArgs<Entry> e)
        {
            base.OnElementChanged(e);
            if (Control != null)
            {
                Control.BackgroundColor = UIColor.LightGray;
                Control.BorderStyle = UITextBorderStyle.Line;
            }
        }
    }
}
```

A chamada ao método **OnElementChanged** da classe base instancia um controle **UITextField** iOS com a referência do controle sendo atribuida a propriedade **Control** do renderizador. A cor de fundo é então definida para **LightGray.**

Consumindo o controle criado

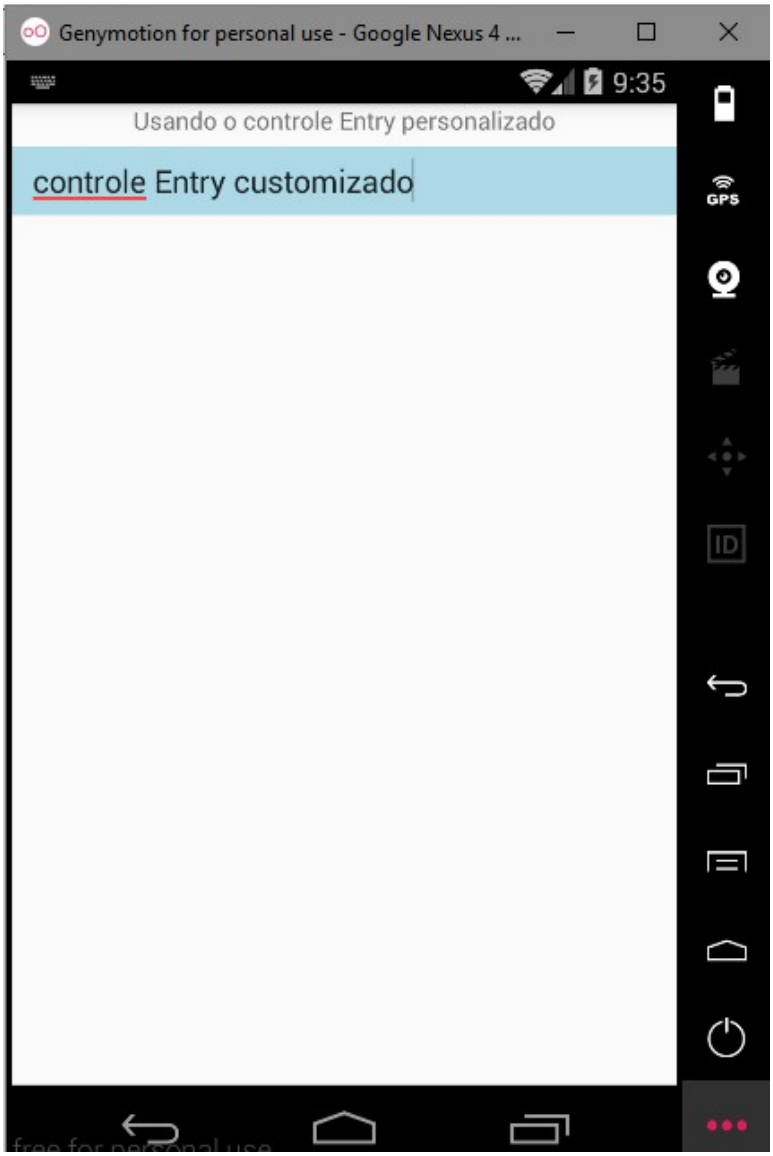
Vamos agora fazer a prova dos 9 e consumir o nosso controle personalizado criado.

Abra o arquivo **MainPage.xaml** criado no projeto e a seguir defina o código abaixo:

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             xmlns:local="clr-namespace:XF_CustomControls"
             x:Class="XF_CustomControls.MainPage">

    <StackLayout>
        <Label Text="Usando o controle Entry personalizado"
          VerticalOptions="Center"
          HorizontalOptions="Center" />

            <local:MacEntry Placeholder="Inserir Texto"></local:MacEntry>


    </StackLayout>

</ContentPage>
```

No código XAML estamos declarando o nosso controle **MacEntry**.

Executando o projeto iremos obter o seguinte resultado: *(Nosso projeto padrão é o projeto Android)*



Conforme esperado obtemos o controle **MacEntry** com cor de fundo **LightBlue** definida no Custom Renderer do Android. Da mesma forma podemos customizar outros controles Xamarin Forms.

Os renderizadores personalizados fornecem uma poderoda abordagem para personalizar a aparência dos controles XF podendo ser usados para pequenas alterações de estilo, como vimos neste artigo, ou uma customização mais sofisticada.

Pegue o código do projeto aqui: **XF_CustomControls.zip** (sem as referências)

"Se dissermos que temos comunhão com ele (Deus), e andarmos em trevas, mentimos, e não praticamos a verdade."
1 João 1:6

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**
- VB.NET 2005 - Controles - Macoratti.net
- Seção de Jogos do site Macoratti .net
- 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/
- 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/
- https://developer.xamarin.com/guides/xamarin-forms/application-fundamentals/custom-renderer/entry/
- https://developer.xamarin.com/guides/xamarin-forms/application-fundamentals/custom-renderer/

José Carlos Macoratti