



Neste artigo veremos como criar uma página de Login usando o Layout Grid no Xamarin Forms.



**Curso de Xamarin Forms Vídeo Aulas**  
Desenvolva para Android, iOS e Windows Phone

A primeira coisa que você não deve fazer é confundir o Layout Grid com as tabelas tradicionais, pois ele não se destina a apresentar dados tabulares. O **Grid** não possui o conceito de formatação de linha, coluna ou célula. Ao contrário das tabelas HTML, o Layout Grid é puramente destinado a exibir conteúdo.

Assim, o Layout Grid permite organizar views em linhas e colunas. As linhas e colunas podem ser configuradas para ter tamanhos proporcionais ou tamanhos absolutos.



**ASP.NET Core 2.0**  
CURSO EM VÍDEO AULAS  
DO BÁSICO AO INTERMEDIÁRIO

PEÇA HOJE MESMO

Então qual a utilidade do Grid ?

Ele pode ser usado para organizar views em uma grade. Isso é útil em vários casos como :

- Organizar botões em um aplicativo de calculadora;
- Organizar botões/opções em uma grade, como as telas do iOS ou Android;
- Organizar views para que elas sejam de igual tamanho em uma dimensão (*como em algumas barras de ferramentas*);

Ao contrário das tabelas tradicionais, o Grid não infere o número e os tamanhos de linhas e colunas do conteúdo.

Em vez disso, ele possui coleções **RowDefinitions** e **ColumnDefinitions** que mantêm as definições de quantas linhas e colunas serão definidas.

As views são adicionadas à grade com índices de linha e coluna especificados que identificam a linha e a coluna em que uma view deve ser colocada.

As informações de linha e coluna são armazenadas nas propriedades **RowDefinitions** e **ColumnDefinitions** da Grade, que são cada coleção de objetos **RowDefinition** e **ColumnDefinition**, respectivamente.

Uma **RowDefinition** tem uma propriedade única, **Height** e uma **ColumnDefinition** tem uma única propriedade, **Width**.

As opções de altura e largura são as seguintes:

1. **Auto** - Redefine o tamanho automaticamente para ajustar o conteúdo na linha ou coluna. Especificado como **GridUnitType.Auto** no código C# ou como **Auto** no código XAML.
2. **Proporcional (\*)** - Dimensiona as colunas e linhas como uma proporção do espaço restante. Especificado como um valor e **GridUnitType.Star** no código C# e como **#\*** no código XAML, com # sendo o valor desejado. (*Especificar uma linha/coluna com \* fará com que ele preencha o espaço disponível.*)
3. **Absolute** - Dimensiona as linhas e colunas com valores específicos de altura e largura fixos. Especificado como um valor e **GridUnitType.Absolute** no código C# e como **#** no código XAML, com # sendo o valor desejado.

A seguir temos um exemplo usando um **Layout Grid** com **3 linhas e 2 colunas** com as seguintes definições:

- A linha da base tem 200px de altura
- A linha do topo é duas vezes maior que a linha do meio
- A coluna da esquerda deve ser larga o suficiente para conter o conteúdo
- A coluna da direita deve preencher o espaço restante.

```
<Grid>
<Grid.RowDefinitions>
  <RowDefinition Height="2*" />
  <RowDefinition Height="*" />
  <RowDefinition Height="200" />
</Grid.RowDefinitions>
```

**Código XAML**

<pre> &lt;Grid.ColumnDefinitions&gt;   &lt;ColumnDefinition Width="Auto" /&gt;   &lt;ColumnDefinition Width="*" /&gt; &lt;/Grid.ColumnDefinitions&gt; &lt;/Grid&gt; </pre>	
<pre> var grid = new Grid();  grid.RowDefinitions.Add (new RowDefinition { Height = new GridLength(2, GridUnitType.Star) }); grid.RowDefinitions.Add (new RowDefinition { Height = new GridLength (1, GridUnitType.Star) }); grid.RowDefinitions.Add (new RowDefinition { Height = new GridLength(200)}); grid.ColumnDefinitions.Add (new ColumnDefinition{ Width = new GridLength (200) }); </pre>	<b>Código C#</b>

Para colocar views em uma grade, você precisará adicioná-las como filhas para a grade, então especifique em qual linha e coluna elas pertencem.

No XAML, use **Grid.Row** e **Grid.Column** em cada view individual para especificar o posicionamento. Observe que **Grid.Row** e **Grid.Column** especificam a localização com base nas listas de linhas e colunas baseadas em zero. Isso significa que em uma grade **4x4**, a célula superior esquerda é **(0,0)** e a célula inferior direita é **(3,3)**.

A seguir temos um exemplo básico que mostra como colocar views em uma grade onde definimos uma grade com **duas linhas e duas colunas** e posicionamos 4 views **BoxView** em cada célula da grade:

```

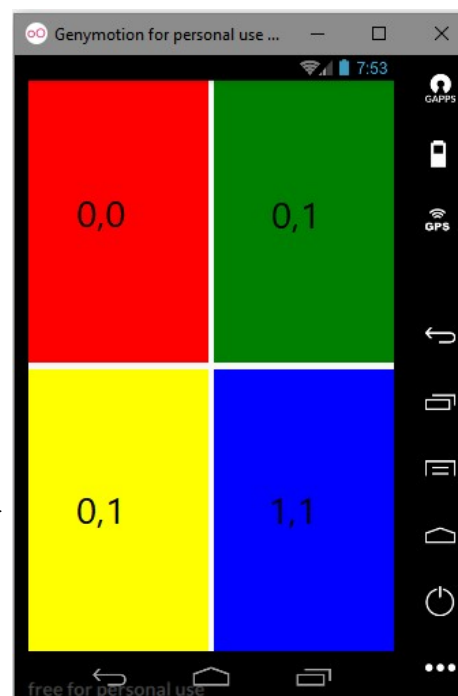
<Grid>
  <Grid.ColumnDefinitions>
    <ColumnDefinition />
    <ColumnDefinition />
  </Grid.ColumnDefinitions>

  <Grid.RowDefinitions>
    <RowDefinition />
    <RowDefinition />
  </Grid.RowDefinitions>

  <BoxView Grid.Column="0" Grid.Row="0" BackgroundColor="Red" />
  <BoxView Grid.Column="1" Grid.Row="1" BackgroundColor="Blue" />

  <BoxView Grid.Column="0" Grid.Row="1" BackgroundColor="Yellow" />
  <BoxView Grid.Column="1" Grid.Row="0" BackgroundColor="Green" />
</Grid>

```



Para controlar o espaçamento entre as linhas e as colunas podemos usar as propriedades :

- **ColumnSpacing** - que define a quantidade de espaços entre as colunas;
- **RowSpacing** - que define a quantidade de espaços entre as linhas;

Exemplo :

```

<Grid ColumnSpacing="5">
  <Grid.ColumnDefinitions>
    <ColumnDefinitions Width="*" />
    <ColumnDefinitions Width="*" />
  </Grid.ColumnDefinitions>
</Grid>

```

Quando um elemento precisar ocupar mais que uma linha ou coluna podemos usar as propriedades:

- **ColumnSpan** - define quantas colunas o elemento poderá ocupar;
- **RowSpan** - define quantas linhas o elemento poderá ocupar;

Exemplo: `<Button Text = "0" Grid.Row="4" Grid.Column="0" Grid.ColumnSpan="2" />`

Neste artigo veremos como usar o Layout Grid para definir uma página de login.

Vamos definir uma grade com 3 linhas onde na segunda linha vamos definir uma grade com 7 linhas e na quinta linha desta grade vamos definir outra grade com 3 colunas.

## Recursos Usados

- [Visual Studio 2017 Community \(update 15.5\)](#)

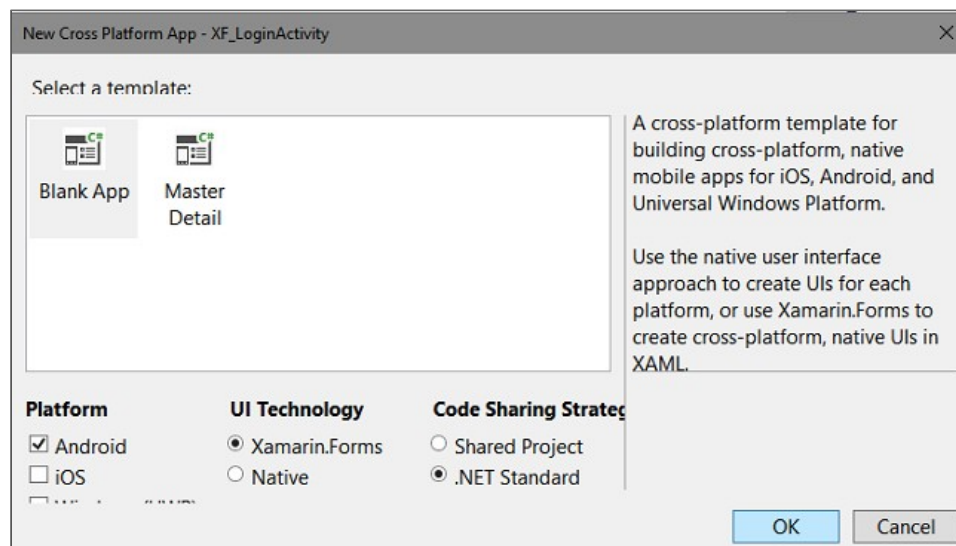
## Criando o projeto Xamarin Forms

Abra o VS 2017 Community [update 15.5](#) e clique em **New Project** e a seguir escolha **Cross Platform -> Cross Platform App (Xamarin.Forms)** e informe o nome **XF\_Login1**;

Ao criar um projeto Xamarin Forms em uma versão anterior à atualização 15.5, você tinha duas opções para compartilhar o código entre as plataformas:

1. **Shared Project**
2. **Portable Class Library (PCL)**

Pois a partir da versão [15.5 do Visual Studio](#)(lançada em dezembro/2017) a opção **Portable Class Library (PCL)** foi substituída pela **.NET Standard**:



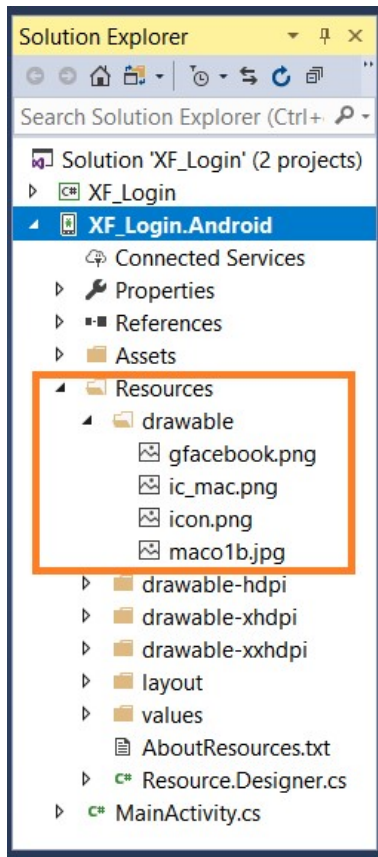
Marque as opções **Android e/ou iOS**, marque **Xamarin Forms** e a seguir marque **.NET Standard** e clique no botão **OK**.

***Nota: Eu estou marcando somente o projeto Android pois vou testar apenas no emulador do Android.***

Pronto, nosso projeto já está criado.

## Definindo as imagens usadas na página de Login

Eu vou exibir imagens locais na página de Login e para isso no projeto **Android** vamos incluir na pasta **Resources/drawable** as imagens conforme mostra a figura abaixo:



Você pode usar o site <https://romannurik.github.io/AndroidAssetStudio/> para definir ícones a partir de imagens ou usar ícones prontos para Android em suas aplicações Xamarin Forms.

As imagens para o projeto iOS deve ser colocadas na pasta **Resources** respeitando a nomenclatura adotada.

## Definindo a página de login no arquivo MainPage

Vamos aproveitar o arquivo **MainPage** criado no projeto para criar a nossa página de login.

Abra o arquivo **MainPage.xaml** e inclua no 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_Login"
    x:Class="XF_Login.MainPage">

    <Grid BackgroundColor="White">
        <Grid.RowDefinitions>
            <RowDefinition Height="50"/>
            <RowDefinition Height="*/>
            <RowDefinition Height="50"/>
        </Grid.RowDefinitions>

        <StackLayout Orientation="Horizontal" HorizontalOptions="Center" Margin="0,10,0,0" Grid.Row="0">
            <Image Source="maco1b.jpg" Opacity="0.6" VerticalOptions="Start" Margin="0,3,0,0"/>
        </StackLayout>

        <Grid Grid.Row="1" Margin="20,0,20,0">
            <Grid.RowDefinitions>
                <RowDefinition Height="*/>
                <RowDefinition Height="50"/>
                <RowDefinition Height="50"/>
                <RowDefinition Height="Auto"/>
                <RowDefinition Height="40"/>
                <RowDefinition Height="40"/>
            </Grid.RowDefinitions>
        </Grid>
    </Grid>
```

```

        <RowDefinition Height="*" />
    </Grid.RowDefinitions>

    <Image Source="ic_mac.png" HeightRequest="70" VerticalOptions="EndAndExpand" />
    <Entry Grid.Row="1" Placeholder="Email ou Telefone" PlaceholderColor="#bababa" FontSize="16" />
    <Entry Grid.Row="2" Placeholder="Senha" PlaceholderColor="#bababa" FontSize="16" />
    <Button Text="LogIn" BackgroundColor="#3897f0" TextColor="White" HeightRequest="50"
VerticalOptions="Start" Grid.Row="3" />
    <Label Text="Problemas no Login ? " HorizontalOptions="Center" Grid.Row="4" Margin="0,10,0,0"
FontSize="12" />

    <Grid Grid.Row="5">
        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="*" />
            <ColumnDefinition Width="Auto" />
            <ColumnDefinition Width="*" />
        </Grid.ColumnDefinitions>

        <Label BackgroundColor="#bababa" HeightRequest="1" HorizontalOptions="FillAndExpand"
VerticalOptions="Center" />
        <Image Source="gfacebook.png" Grid.Column="1" VerticalOptions="Center" Margin="10,0,10,0" />
        <Label BackgroundColor="#bababa" Grid.Column="2" HeightRequest="1"
HorizontalOptions="FillAndExpand" VerticalOptions="Center" />
    </Grid>

    <StackLayout Orientation="Horizontal" HorizontalOptions="CenterAndExpand" Grid.Row="6">
        <Label Text="Login com Facebook" TextColor="#485992" />
    </StackLayout>
</Grid>

<StackLayout Grid.Row="2" BackgroundColor="#ffffff">
    <Label HeightRequest="1" BackgroundColor="#e3e3e3" />
    <Label Text="Não possui uma Conta ? Registre-se." VerticalOptions="FillAndExpand"
VerticalTextAlignment="Center"
        HorizontalTextAlignment="Center" />
</StackLayout>

</Grid>
</ContentPage>

```

Neste código temos as seguintes definições:

- Uma **Grade com 3 linhas** :

- Na primeira linha (row=0) colocamos a imagem do logo Macoratti.net
- Na segunda linha (row=1) definimos uma **grade com 7 linhas** e exibimos a imagem do usuário do login, o email, a senha, o botão Login e a Label com o texto '*Problemas com Login ?*'
  - Na linha 5 desta grade definimos uma **grade com 3 colunas** onde exibimos o logo do facebook
  - Na linha 6 desta grade exibimos o texto 'Login com Facebook'
- Na terceira linha exibimos a Label com o texto '*Não possui conta ? Registre-se*'

Usamos assim 3 grades para definir o leiaute da página de login.

## Definindo a página de login como a página principal

Lembrando que a página principal da aplicação, **MainPage** , esta definida no arquivo **App.xaml.cs**.

```

using Xamarin.Forms;

namespace XF_ApresentacaoAnimada
{
    public partial class App : Application
    {

```

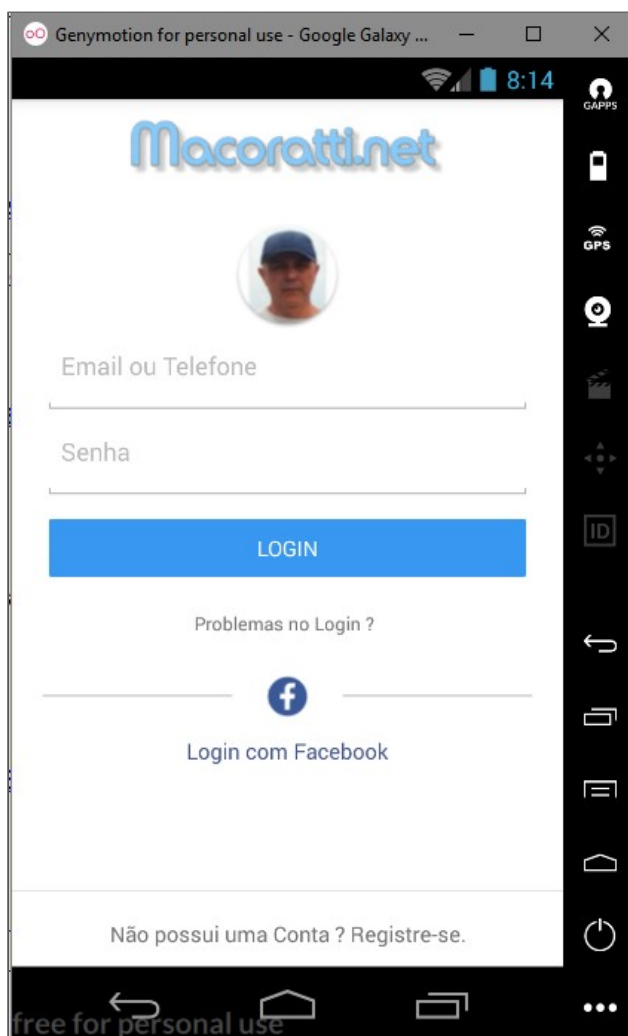
```
public App ()
{
    InitializeComponent();
    MainPage = new XF_Login.MainPage();
}

protected override void OnStart ()
{
    // Handle when your app starts
}


protected override void OnSleep ()
{
    // Handle when your app sleeps
}

protected override void OnResume ()
{
    // Handle when your app resumes
}
}
```

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



Criamos assim uma página de Login estática que você pode usar como um modelo para a sua página de Login.

Pegue o código do projeto compartilhado aqui :  [XF\\_Login.zip](#) (sem as referências)

**"Portanto, agora nenhuma condenação há para os que estão em Cristo Jesus, que não andam segundo a carne, mas segundo o Espírito."**

[Romanos 8:1](#)

[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 - Vídeo 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](#)

Gostou ?  [Compartilhe no Facebook](#)  [Compartilhe no Twitter](#)

#### 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#](#)
- [Seção C# do site Macoratti.net](#)
- [Super DVD C#](#)
- [Super DVD Visual Basic](#)
- [Curso Básico VB .NET - Vídeo Aulas](#)
- [Curso C# Básico - Vídeo Aulas](#)
- [Xamarin Android - Apresentando o controle ProgressBar - Macoratti.net](#)
- [Xamarin Android - Usando o serviço de Alarme - Macoratti.net](#)
- [Xamarin.Forms - Principais Recursos - Macoratti.net](#)
- [Xamarin Forms - Previsão do Tempo - Macoratti.net](#)
- [Xamarin Android - Aplicando animações - Macoratti.net](#)
- [Xamarin Android - Usando a classe Timer - Macoratti.net](#)
- [Xamarin Android - Epelhando o dispositivo físico no ... - Macoratti.net](#)
- [Xamarin.Forms - Trabalhando com ListView - Macoratti.net](#)
- [Xamarin Forms - Dicas de desempenho - Macoratti.net](#)
- [Xamarin.Forms - Usando a view ActivityIndicator - Macoratti.net](#)
- [Curso de Xamarin.Forms - Macoratti.net](#)

---

José Carlos Macoratti