Macoratti.net Xamarin Forms - Usando o Local File Storage (PCL Storage) - II

Neste artigo vou mostrar como usar a **API PCL Storage** para acessar informações em uma aplicação Xamarin Forms usando no Visual Studio 2017 e a linguagem C#.

Continuando a **primeira parte do artigo** vamos implementar as páginas **RegistroPage** e **PerfilPage**.

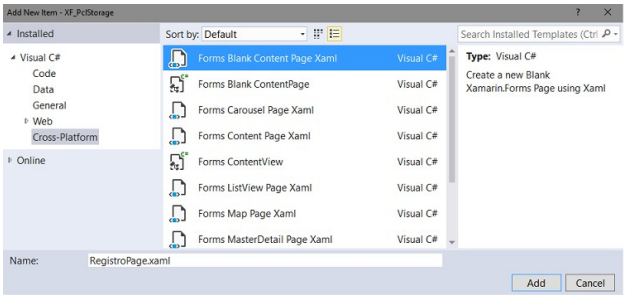
Recursos usados:

- Visual Studio Community 2017 ou Xamarin Studio
- Xamarin

Criando as páginas RegistroPage e PerfilPage

Vamos criar duas páginas no projeto usando o menu **Project -> Add New Item;**

Selecionando **Cross-Platform** e a opção : **Forms Blank Content Page Xaml:**



Informe o nome **RegistroPage** e a seguir repita o procedimento criando a página **PerfilPage**.

Definindo o código da página RegistroPage

No arquivo **RegistroPage.xaml** inclua o código XAML abaixo para definir a interface do usuário para fazer o login ou registro de um novo usuário:

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="XF_PclStorage.RegistroPage"
             Padding="0, 20, 0, 0">
    <Grid>
        <Grid.RowDefinitions>
            <RowDefinition Height="Auto" />
            <RowDefinition Height="Auto" />
            <RowDefinition Height="Auto" />
            <RowDefinition Height="Auto" />
            <RowDefinition Height="Auto" />
        </Grid.RowDefinitions>

        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="Auto" />
            <ColumnDefinition Width="Auto" />
            <ColumnDefinition Width="300" />
        </Grid.ColumnDefinitions>
        <Label Text="Registro" Grid.Row="0" Grid.Column="2" FontSize="50" ></Label>

        <Entry Placeholder="Nome" x:Name="txtNome" Grid.Row="1" Grid.Column="1" Grid.ColumnSpan="2"  />
        <Entry Placeholder="Usuário"  x:Name="txtUsuario" Grid.Row="2" Grid.Column="1" Grid.ColumnSpan="2"  />
        <Entry IsPassword="True"  x:Name="txtSenha" Placeholder="Senha" Grid.Row="3" Grid.Column="1" Grid.ColumnSpan="2" />

        <StackLayout Grid.Row="4" Grid.Column="1" Grid.ColumnSpan="3">
            <Button Text="Registro" Clicked="btnRegistro_Click" />
            <Button Text="Login" Clicked="btnLogin_Click"/>
        </StackLayout>

    </Grid>

</ContentPage>
```

Usamos as views **Entry** para o usuário informar o nome, usuário e senha e definimos os botões **Registro** e **Login** com os respectivos eventos Click.

A seguir temos o código do arquivo **RegistroPage.xaml.cs** onde tratamos os eventos **Click** dos botões:

```
using System;
using System.Text;
using Xamarin.Forms;
using Xamarin.Forms.Xaml;

namespace XF_PclStorage
{
    [XamlCompilation(XamlCompilationOptions.Compile)]
    public partial class RegistroPage : ContentPage
    {
        public RegistroPage()
        {
            InitializeComponent();
        }

        private async void btnRegistro_Click(object sender, EventArgs e)
        {
            bool resultado = false;
            bool arquivoExiste = await txtUsuario.Text.ArquivoExisteAsync();

            if (arquivoExiste != true)
            {
```

```
            if (!string.IsNullOrEmpty(txtNome.Text)  && !string.IsNullOrEmpty(txtSenha.Text) && !string.IsNullOrEmpty(txtUsuario.Text))
                resultado = await txtUsuario.Text.WriteTextAllAsync(ContentBuilder(txtNome.Text, txtUsuario.Text, txtSenha.Text));

            if (resultado != true)
            {
                await DisplayAlert("Registro", "Registro Falhou .. \nTente novamente", "OK");
            }
            else
            {
                await DisplayAlert("Registro", "Registro feito com Sucesso... \nFaça o Login e Edite o Perfil", "OK");
                await Navigation.PushModalAsync(new MainPage());
            }
        }
        else
        {
            await DisplayAlert("Registro Falhou", "Nome do usuário já existe... \nTente um nome diferente", "OK");
            txtUsuario.Text = "";
            txtUsuario.Focus();
        }
    }

    public string ContentBuilder(params string[] content)
    {
        StringBuilder contentbuilder = new StringBuilder();
        foreach (var item in content)
        {
            contentbuilder.AppendLine(item.ToString());
        }
        return contentbuilder.ToString();
    }

    private async void btnLogin_Click(object sender, EventArgs e)
    {
        await Navigation.PushModalAsync(new MainPage());
    }
  }
}
```

Neste código verificamos se o arquivo existe usando o método **ArquivoExisteAsync()** e caso os dados informados forem válidos usamos o método **WriteTextAllAsync()** para gravar as informações do novo usuário no arquivo texto local.

Definindo o código da página PerfilPage

No arquivo **PerfilPage.xaml** inclua o código XAML abaixo para definir a interface do usuário para editar o perfil do usuário.

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="XF_PclStorage.PerfilPage"
             Padding="0,20,0,0">

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

        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="Auto" />
            <ColumnDefinition Width="Auto" />
            <ColumnDefinition Width="300" />
        </Grid.ColumnDefinitions>

        <Label Text="Registro" Grid.Row="0" Grid.Column="2" FontSize="50" />
        <Entry Placeholder="Nome" x:Name="txtNome" Grid.Row="1" Grid.Column="1" Grid.ColumnSpan="2"  />
        <Entry Placeholder="Usuário"  x:Name="txtUsuario" IsEnabled="false" Grid.Row="2" Grid.Column="1" Grid.ColumnSpan="2"  />
        <Entry IsPassword="True"  x:Name="txtSenha" Placeholder="Senha" Grid.Row="3" Grid.Column="1" Grid.ColumnSpan="2" />

        <StackLayout Grid.Row="4" Grid.Column="1" Grid.ColumnSpan="3">
            <Button Text="Atualizar Perfil" Clicked="btnAtualizaPerfil_Click" />
            <Button Text="LogOut" Clicked="btnLogin_Click"/>
        </StackLayout>
    </Grid>

</ContentPage>
```

Neste código usamos as Views **Entry** para permitir o usuário alterar o nome e a senha e definimos os eventos Click dos botões **Atualizar Perfil e Logout**.

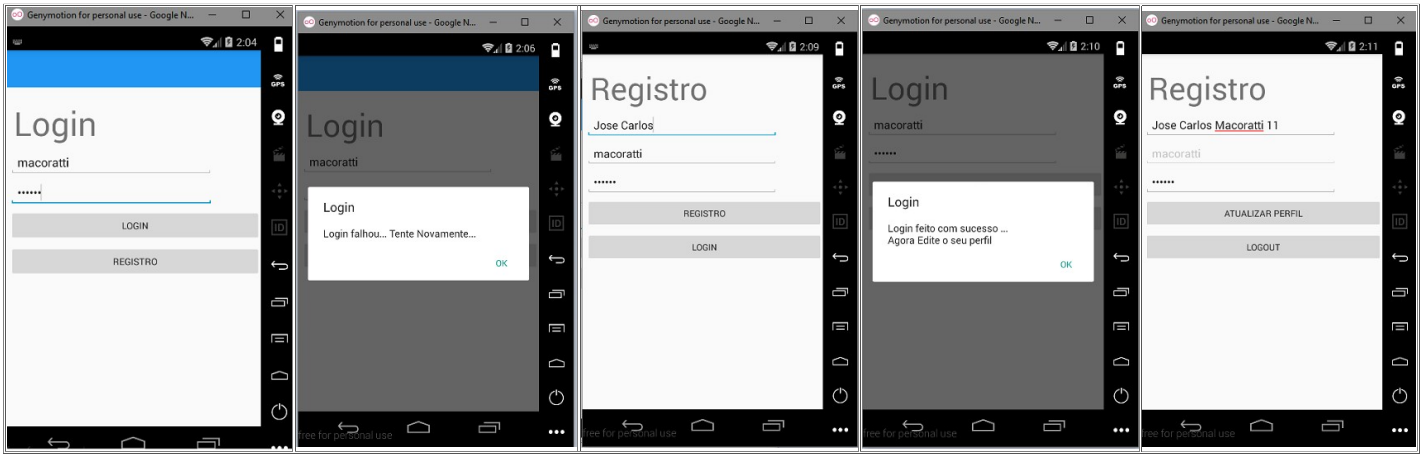
A seguir temos o código do arquivo **PerfilPage.xaml.cs** onde tratamos os eventos **Click** dos botões:

```
using System;
using System.Text;
using Xamarin.Forms;
using Xamarin.Forms.Xaml;

namespace XF_PclStorage
{
    [XamlCompilation(XamlCompilationOptions.Compile)]
    public partial class PerfilPage : ContentPage
    {
        public PerfilPage(string usuario)
        {
            InitializeComponent();
            LeInfoUsuario(usuario);
        }

        public async void LeInfoUsuario(string usuario)
        {
```

```
            string conteudo = await PCLHelper.ReadAllTextAsync(usuario);
            string[] valor = conteudo.Split('\n');
            txtNome.Text = valor[0].ToString();
            txtUsuario.Text = valor[1].ToString();
            txtSenha.Text = valor[2].ToString();
        }

        private async void btnLogin_Click(object sender, EventArgs e)
        {
            await Navigation.PushModalAsync(new MainPage());
        }

        public string ContentBuilder(params string[] content)
        {
            StringBuilder contentbuilder = new StringBuilder();
            foreach (var item in content)
            {
                contentbuilder.AppendLine(item.ToString());
            }
            return contentbuilder.ToString();
        }

        private async void btnAtualizaPerfil_Click(object sender, EventArgs e)
        {
            bool resultado = false;

            bool arquivoExiste = await txtUsuario.Text.ArquivoExisteAsync();
            if (arquivoExiste == true)
            {
                if (!string.IsNullOrEmpty(txtNome.Text) && !string.IsNullOrEmpty(txtSenha.Text) && !string.IsNullOrEmpty(txtUsuario.Text))
                    resultado = await txtUsuario.Text.WriteTextAllAsync(ContentBuilder(txtNome.Text, txtUsuario.Text, txtSenha.Text));

                if (resultado != true)
                {
                    await DisplayAlert("Atualiza Perfil", "Atualização falhou...\nTente novamente", "OK");
                }
                else
                {
                    await DisplayAlert("Atualiza Perfil", "Perfil atualizado com sucesso.", "OK");

                }
            }
            else
            {
                await DisplayAlert("Atualiza Perfil", "Tente novamente.", "OK");
            }
        }
    }
}
```

Neste código verificamos se o arquivo existe usando o método **ArquivoExisteAsync()** e caso os dados informados forem válidos usamos o método **WriteTextAllAsync()** para gravar as alterações do perfil do usuário no arquivo texto local.

Executando o projeto obteremos o seguinte resultado:



Na sequência das figuras vemos a página de Login onde inicialmente não temos nenhum usuário registrado.

A seguir registramos um usuárioi e efetuamos o login acessando a página de perfil.

As informações do usuário como nome, usuário e senha estão sendo armazenadas no sistema local de arquivos usando a PCL Storage.

Pegue o projeto aqui : **XF_PclStorage.zip**

Eu sou a videira, vós as varas; quem está em mim, e eu nele, esse dá muito fruto; porque sem mim nada podeis fazer.
João 15:5

- **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 (@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/

José Carlos Macoratti