Macoratti.net Xamarin Android - Exibindo HTML com o controle WebView

Neste artigo vou mostrar como exibir conteúdo HTML usando o controle **WebView** do Xamarin Android usando o Visual Studio 2015 e a linguagem C#.

O componente **WebView** é uma view usada para a exibição de conteúdos web e HTML em seu aplicativo. O **WebView** é muito versátil e possui os seguintes recursos:

- **Conteúdo** - Suporta várias fontes de conteúdo, incluindo HTML embutido, páginas web, e as strings HTML.
- **Navegação** - Inclui suporte para navegar para uma determinada página e voltar.
- **Eventos** - Escuta e responde às ações tomadas pelo usuário no WebView.
- **Layout** - Possui alguns requisitos específicos de como ele é exibido.

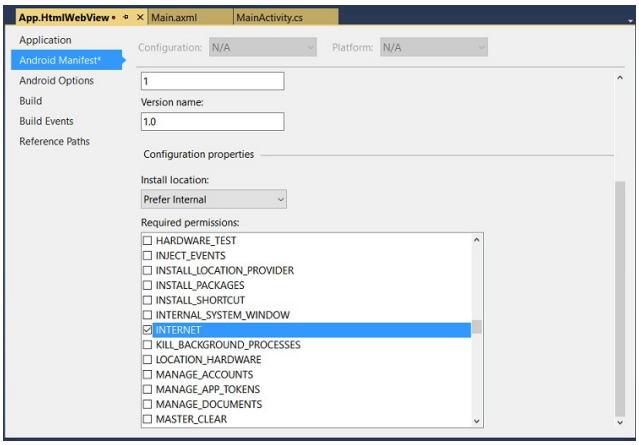
Neste primeiro contato com o controle **WebView** vamos nos ater a exibir conteúdo HTML.

Neste caso o controle vem com suporte para os seguintes tipos de conteúdo:

- **HTML & websites CSS** - O WebView tem suporte completo para sites escritos usando HTML e CSS, incluindo suporte a JavaScript;
- **Documentos** - Como o WebView é implementado usando componentes nativos em cada plataforma, ele é capaz de mostrar os documentos que podem ser visualizados em cada plataforma. Isso significa que os arquivos PDF funcionam no iOS e no Android, mas não o Windows Phone;
- **Strings HMTL**- O WebView pode mostrar strings HTML a partir da memória;
- **Arquivos locais** - O WebView pode apresentar qualquer um dos tipos de conteúdo acima embutido no aplicativo;

Nota: *O WebView no Windows e Windows Phone não suporta Silverlight, Flash ou quaisquer controles ActiveX, mesmo se eles são suportados pelo Internet Explorer nessa plataforma.*

Dependendo do cenário você vai precisar requerer o acesso a Internet abrindo a janela de propriedades do projeto e na opção **Android Manifest** marcar a opção **INTERNET** conforme mostra a figura abaixo:



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)**

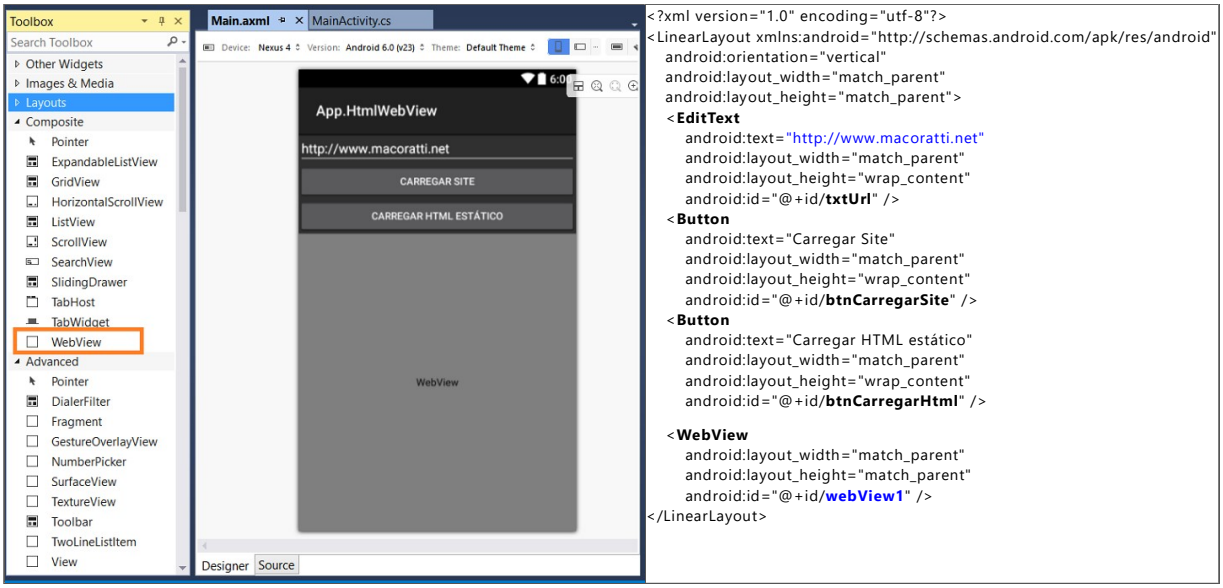
Informe o nome **AppHtmlWebView** e clique no botão **OK**;

Abra o arquivo **Main.axml** na pasta **Resources/layout** e no modo **Designer.**

Primeiro remova o controle **Button** que vem definido por padrão e a seguir inclua o seguinte controle a partir da **ToolBox**:

- **1 EditText - id = txtUrl**
- **1 Button - id - btnCarregarSite**
- **1 Button - id - btnCarregarHtml**
- **1 WebView - id = webView1**

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



A seguir abra o arquivo **MainActivity.cs** e altere o código desse arquivo conforme abaixo:

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

namespace App.HtmlWebView
{
    [Activity(Label = "App.HtmlWebView", MainLauncher = true, Icon = "@drawable/icon")]
    public class MainActivity : Activity
    {
        EditText txtUrl;
        WebView webview1;
        Button btnCarregarSite;
        Button btnCarregarHtml;

        protected override void OnCreate(Bundle bundle)
        {
            base.OnCreate(bundle);
            SetContentView(Resource.Layout.Main);

            //cria instância dos controles usados na view principal
            txtUrl = FindViewById<EditText>(Resource.Id.txtUrl);
            btnCarregarSite = FindViewById<Button>(Resource.Id.btnCarregarSite);
            btnCarregarHtml = FindViewById<Button>(Resource.Id.btnCarregarHtml);
            webview1 = FindViewById<WebView>(Resource.Id.webView1);
            webview1.Settings.JavaScriptEnabled = true;

            //define os eventos dos botões
            btnCarregarSite.Click += BtnCarregarSite_Click;
            btnCarregarHtml.Click += BtnCarregarHtml_Click;

        }

        private void BtnCarregarHtml_Click(object sender, System.EventArgs e)
        {
            //define um conteúdo html estático
            string html = "<html><body><h1>Ola, WebView</h1>" +
            "<hr/>" +
            "<h2>Macoratti.net</h2>" +
            "<h3>Quase Tudo para a plataforma .NET</h3>" +
            "<h4>http://www.macoratti.net</h4>" +
            "<p>Xamarin Android - WebView</p>" +
            "</body></html>";
            webview1.LoadData(html, "text/html", "UTF-8");
        }

        private void BtnCarregarSite_Click(object sender, System.EventArgs e)
        {
            //cria uma instância de WebViewClient
            webview1.SetWebViewClient(new MeuWebViewClient());

            //carrega a url a ser renderizada no WebView
            //txtUrl.Text obtém o texto digitado
            webview1.LoadUrl(txtUrl.Text);
        }

        public class MeuWebViewClient : WebViewClient
        {
            public override bool ShouldOverrideUrlLoading(WebView view, string url)
            {
                view.LoadUrl(url);
                return true;
            }
        }
    }
}
```

Vou explicar o código a seguir.

Executando o projeto usando o emulador do **Xamarin Android Player** e emulando o **KitKat(API 19)** iremos obter o seguinte resultado:

1- Exibindo o conteúdo de um site - O usuário informa uma URL completa e clica no botão **Carregar Site**:

O comportamento padrão do Android é abrir um navegador quando um link for clicado, mas o nosso WebView não funciona como um navegador embutido.

Por isso estamos usando a classe **WebViewClient** que ajuda a monitorar os eventos em um WebView. No exemplo sobrescrevemos o método shouldOverrideUrlLoding() que permite realizar nossa própria ação quando uma URL for selecionada, carregando a URL. (**LoadUrl(url)**)

Nota: *A classe WebViewClient possui outros métodos como OnPageStarted(), OnPageFinished() e OnReceivedError() que ajudam a exibir o progresso da exibição do conteúdo no WebView ou tratar erros.*

```
private void BtnCarregarSite_Click(object sender, System.EventArgs e)
{
   //cria uma instância de WebViewClient
   webview1.SetWebViewClient(new MeuWebViewClient());

   //carrega a url a ser renderizada no WebView
   //txtUrl.Text obtém o texto digitado
   webview1.LoadUrl(txtUrl.Text);
}

public class MeuWebViewClient : WebViewClient
{
   public override bool ShouldOverrideUrlLoading(WebView view, string url)
   {
      view.LoadUrl(url);
      return true;
   }
}
```

2- Exibindo HTML estático

Aqui estamos definindo o código HTML de forma estática e usando o método **LoadData** para carregar e exibir o resultado no WebView :

```
private void BtnCarregarHtml_Click(object sender, System.EventArgs e)
{
   //define um conteúdo html estático
   string html = "<html><body><h1>Ola, WebView</h1>" +
   "<hr/>" +
   "<h2>Macoratti.net</h2>" +
   "<h3>Quase Tudo para a plataforma .NET</h3>" +
   "<h4>http://www.macoratti.net</h4>" +
   "<p>Xamarin Android - WebView</p>" +
   "</body></html>";
   webview1.LoadData(html, "text/html", "UTF-8");
}
```

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

"Aquele que diz: Eu conheço-o (a Jesus), e não guarda os seus mandamentos, é mentiroso, e nele não está a verdade."
1 João 2:4

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**

José Carlos Macoratti