Como achar lugares em torno de você com Nokia Maps e GeoNames API, usando Windows Phone

Este artigo mostra-lhe o caminho para obter lugares famosos ao seu redor usando GeoNames API. Depois de obter as informações iremos mostrar-lhes como marcadores / pushpins nos mapas usando um controle Nokia Maps.

Note: A versão deste artigo para Windows Phone 7 pode ser encontrada aqui.

Introdução

GeoNames é uma grande coleção de base de dados contendo informação de todos os países. Esta base de dados contém mais de oito milhões de nomes de lugares cuja informação pode ser obtido gratuitamente.

Usando GeoNames API, podemos obter diversos tipos de informações, mas neste artigo vamos apenas obter um tipo de informação; *Informação de lugares*. Para obter essas informações, vamos usar o Wikipedia API.

Para desenhar pushpins sobre o controle Maps, vamos usar o contolo `Pushpin` fornecido pelo Windows Phone Toolkit.

Pré-requisitos

- Ambiente de desenvolvimento do Windows Phone
- Registo no site GeoName
- Referência ao Windows Phone Toolkit

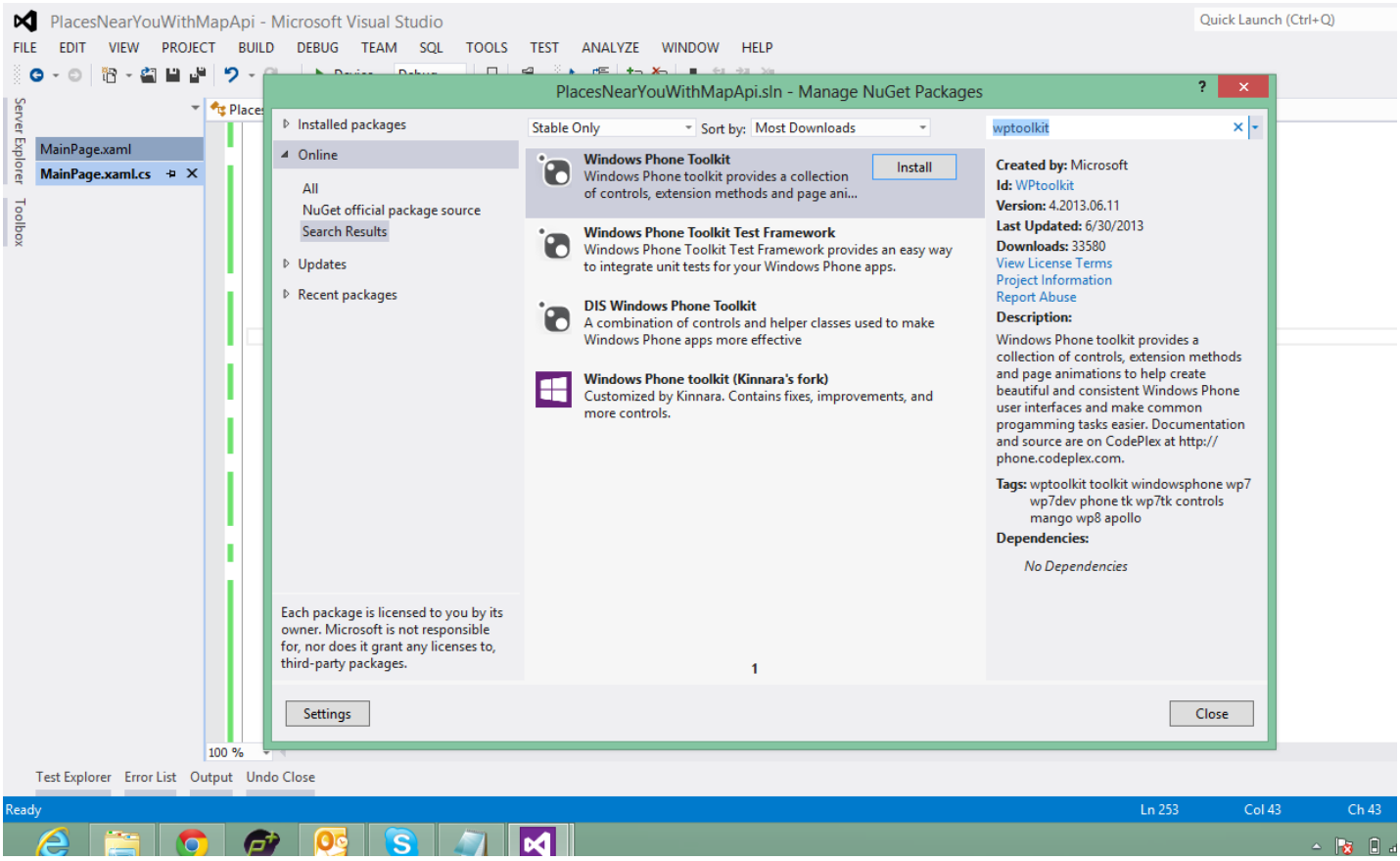
Referências Obrigatórias

As seguintes dlls são obrigatórias:

- System.Collections.ObjectModel;
- System.IO.IsolatedStorage;
- Microsoft.Phone.Maps.Controls;
- Microsoft.Phone.Maps.Toolkit;

Estes podem ser adicionados por **Project | References | Add Reference..**

Note: Para adicionar o Windows Phone Toolkit, procure a palavra-chave "wptoolkit" a partir de NuGet, como mostrado na imagem a seguir. NuGet é uma ferramenta embutida no VS2012.



Wikipedia API

Para fazer uso da Wikipedia API, vamos ter que fazer uma solicitação do servidor HTTP no link abaixo: http://api.geonames.org/findNearbyWikipediaJSON?

Vamos passar alguns parâmetros necessários como a nossa localização, nome de utilizador, etc Em resposta, receberá uma lista de lugares perto de si em formato de JSON

Note: Um dos parâmetros, ou seja, *'username'*, é obrigatório enviar a cada solicitação do servidor. Para saber mais sobre o

(C) Copyright Nokia 2014. All rights reserved.

usarname verifique este link.

Criando interface

Na nossa interface, iremos ter três textblocks e o contolo de mapa da Nokia. Primeiro `TextBlock` irá In our UI, we will make use of three text blocks and a Nokia maps control. First `TextBlock` terá a mensagem de status depois de obter a nossa localização do dispositivo. Segundo irá ter a nossa latitude. E por último `TextBlock` irá ter a nossa longitude. No nosso controlo do mapa, vamos mostrar os lugares resultantes com a ajuda de pushpins.

O código em 'MainPage.xaml **é como mostrado de seguida:**

```
    <!--ContentPanel - place additional content here-->
        <Grid x:Name="ContentPanel" Grid.Row="0" Margin="12,0,12,0">
            <Grid.RowDefinitions>
                <RowDefinition Height=".05*"/>
                <RowDefinition Height=".05*"/>
                <RowDefinition Height=".05*"/>
                <RowDefinition Height="*"/>
            </Grid.RowDefinitions>

            <TextBlock Grid.Row="0" HorizontalAlignment="Left"  x:Name="statusTextBlock"
Text="TextBlock" Visibility="Collapsed"/>
            <TextBlock Grid.Row="1" HorizontalAlignment="Left"
x:Name="latitudeTextBlock" Text="TextBlock" Visibility="Collapsed"/>
            <TextBlock Grid.Row="2" HorizontalAlignment="Left"
x:Name="longitudeTextBlock" Text="TextBlock" Visibility="Collapsed"/>
            <maps:Map Grid.Row="3" Name="mapControl" Visibility="Collapsed"
CartographicMode="Road">
                <toolkit:MapExtensions.Children>
                    <toolkit:MapItemsControl>
                        <toolkit:MapItemsControl.ItemTemplate>
                            <DataTemplate>
                                <toolkit:Pushpin GeoCoordinate="{Binding Coordinate}"
Content="{Binding Info}" />
                            </DataTemplate>
                        </toolkit:MapItemsControl.ItemTemplate>
                    </toolkit:MapItemsControl>
                </toolkit:MapExtensions.Children>
            </maps:Map>
        </Grid>
```

Inicialmente a visibilidade do controlo mapa é definido como `Collapsed` sendo mostrado quando é requerido. `MapExtensions` é uma extensão fornecedia pelo `Windows Phone Toolkit` para desenhar controlos no mapa.. `MapExtensions` tem uma propriedade `Children` que irá definir os itens a serem mostrados no mapa. Estamos criando uma `MapItemsControl` para a propriedade `Children` que eventualmente terá muitos itens a serem exibidos.

Code Behind

JSON Parsing

O seguinte ficheiro (**Place.cs**) tem sido utilizado para JSON parsing.

```
namespace PlacesNearYou.Data
{
    [DataContract]
    public class Place
    {
        [DataMember(Name = "summary")]
        public string Summary { get; set; }

        [DataMember(Name = "distance")]
        public string Distance { get; set; }

        [DataMember(Name = "rank")]
        public string Rank { get; set; }

        [DataMember(Name = "title")]
        public string Title { get; set; }

        [DataMember(Name = "wikipediaUrl")]
        public string WikipediaUrl { get; set; }

        [DataMember(Name = "elevation")]
        public string Elevation { get; set; }

        [DataMember(Name = "lng")]
```


(cc) Copyright Nokia 2014. All rights reserved.

```
        public string Longitude { get; set; }

        [DataMember(Name = "feature")]
        public string Feature { get; set; }

        [DataMember(Name = "lang")]
        public string Langauge { get; set; }

        [DataMember(Name = "lat")]
        public string Latitude { get; set; }
    }
}
```

Outro ficheiro (**PlacesList.cs**) usado para JSON parsing tem a seguinte declaração:

```
namespace PlacesNearYou.Data
{
    [DataContract]
    public class PlacesList
    {
        [DataMember(Name = "geonames")]
        public List<Place> PlaceList { get; set; }
    }
}
```

Criação de estrutura para Binding

O seguinte ficheiro (**PlaceToMap.cs**) foi usado:

```
namespace PlaceToMap.Data
{
   public class PlaceToMap
   {
     public GeoCoordinate Coordinate { get; set; }
     public string Info { get; set; }
   }
}
```

Fazer Pedido Servidor

Nós vamos começar a encontrar a localização atual do telefone. Para isso, vamos fazer uso de `GeoCoordinateWatcher` API como é explicado neste artigo. Depois de obter com sucesso a localização atual do dispositivo (em termos de latitude e longitude), vamos começar uma solicitação HTTP para o API GeoNames. O seguinte código é mostrado abaixo:

```
    private void startGeoNamesAPICall()
        {
            try
            {
                HttpWebRequest httpReq = (HttpWebRequest)HttpWebRequest.Create(new
Uri(AppConstants.baseUri + "&lat=" + currentLatitude + "&lng=" + currentLongitude +
"&username=" + AppConstants.strUserName + "&radius=" +
AppConstants.strDefaultRadiusForWikiAPI + "&maxRows=" +
AppConstants.strDefaultResultRowsForWikiAPI));
                httpReq.BeginGetResponse(HTTPWebRequestCallBack, httpReq);
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }
```

Como mostrado acima, estamos passando alguns parâmetros, juntamente com a URL base. Estes parâmetros são:

- Nossa própria latitude
- Nossa longitude
- O nome de utilizador(em nome do qual o pedido é feito, é obrigatório fornecer um nome de utilizador registrado). O nome de utilizador pode ser registrado aqui. Para mais informações sobre nome de utilizador, você pode verificar isso aqui.
- Raio - para determinar em quanto raio pesquisa será realizada
- Máximo de Linhas - quantas linhas são exigidas como máximo no resultado

Analisar a resposta

O seguinte código para manipular a resposta é mostrada de seguinda:

(cc) Copyright Nokia 2014. All rights reserved.

```
 private void HTTPWebRequestCallBack(IAsyncResult result)
        {
            string strResponse = "";

            try
            {
                Dispatcher.BeginInvoke(() =>
                {
                    try
                    {
                        HttpWebRequest httpRequest = (HttpWebRequest)result.AsyncState;
                        WebResponse response = httpRequest.EndGetResponse(result);
                        Stream stream = response.GetResponseStream();
                        StreamReader reader = new StreamReader(stream);
                        strResponse = reader.ReadToEnd();

                        parseResponseData(strResponse);
                    }
                    catch (Exception ex)
                    {
                        MessageBox.Show(ex.Message);
                    }
                });
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }
```

Depois de converter o `WebResponse` tipo de dados em string, precisamos analisá-lo agora, pois ainda está no formato JSON. o código para análise é a seguinte:

```
 private void parseResponseData(String aResponse)
        {
            placesListObj = new PlacesList();

            MemoryStream ms = new MemoryStream(Encoding.UTF8.GetBytes(aResponse));
            DataContractJsonSerializer ser = new
DataContractJsonSerializer(placesListObj.GetType());
            placesListObj = ser.ReadObject(ms) as PlacesList;
            ms.Close();

            // updating UI
            if (placesListObj != null)
            {
                updateMap(placesListObj);
            }
        }
```

Pushpins de desenho no mapa

Em caso do parsing ser feito com sucesso e obtermos pelo menos um simples resultado na resposta, iremos mostrar a informação usando pushpin(s).

```
 private void updateMap(PlacesList aWiKIAPIResponse)
        {
            int totalRecords = aWiKIAPIResponse.PlaceList.Count();
            mapControl.Visibility = System.Windows.Visibility.Visible;

            try
            {
                ObservableCollection<PlaceToMap> placeToMapObjs = new
ObservableCollection<PlaceToMap>();
                for (int index = 0; index < totalRecords; index++)
                {
                    placeToMapObjs.Add(new PlaceToMap()
                    {
                        Coordinate = new
GeoCoordinate(Convert.ToDouble(aWiKIAPIResponse.PlaceList.ElementAt(index).Latitude),

Convert.ToDouble(aWiKIAPIResponse.PlaceList.ElementAt(index).Longitude)),
                        Info = aWiKIAPIResponse.PlaceList.ElementAt(index).Title +
Environment.NewLine + aWiKIAPIResponse.PlaceList.ElementAt(index).Feature
```


(C) Copyright Nokia 2014. All rights reserved.

```
                    });
                }

                ObservableCollection<DependencyObject> children =
MapExtensions.GetChildren(mapControl);
                var obj = children.FirstOrDefault(x => x.GetType() ==
typeof(MapItemsControl)) as MapItemsControl;

                obj.ItemsSource = placeToMapObjs;
                mapControl.SetView(new GeoCoordinate(Convert.ToDouble(currentLatitude),
Convert.ToDouble(currentLongitude)), 14);
            }
            catch (Exception)
            {

            }
        }
```

Criar e executar

Agora você pode construir o aplicativo e tentar executá-lo.

Referências

- geonames.org
- geonames.org Wikipedia Webservice
- Using Location Service APIs in WP7
- How to add UIElements to a Map control in Windows Phone 8
- Maps in Windows Phone 8 and Phone toolkit: a winning team – Part 2 (qmatteoq.com blog)

(cc) Copyright Nokia 2014. All rights reserved.