Mapa com direções no Windows Phone 8

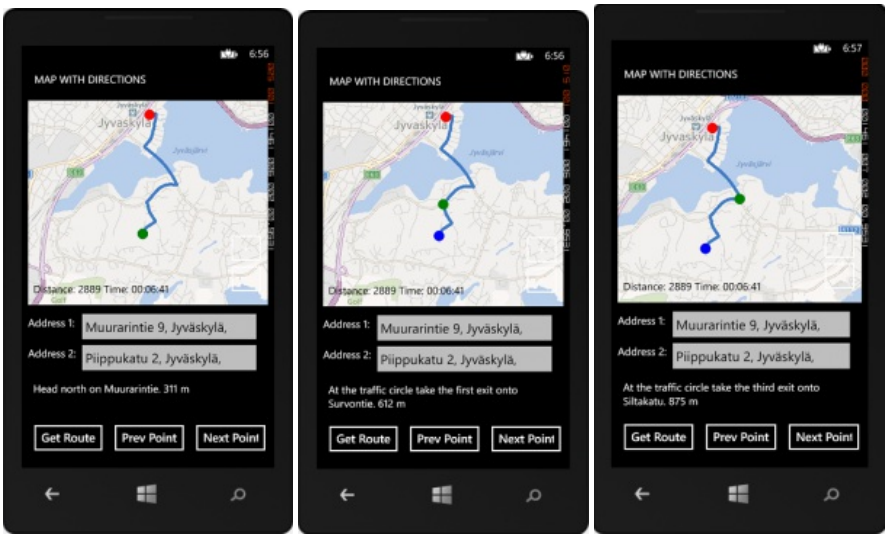
Este artigo explica como mostrar uma rota no mapa e obter as direções (texto e voz) com o Windows Phone 8.

03 Feb 2013

Introdução

Esta demonstração de código demonstra como mostrar uma rota com o `Map Control` e obter informações de direções em texto e voz (text to speech) com o Windows Phone 8. O endereço inicial e de destino podem ser inseridos com `TextBlocks`. Em primeiro lugar ambos os endereços são convertidos em coordenadas geográficas e são mostradas as suas localizações no mapa utilizando uma camada (layer) no mapa. É utilizada a `RouteQuery` com as duas localizações geográficas e a rota é visualizada em uma nova camada (layer) no mapa. O utilizador pode simular rotas pressionando os botões próximo ponto (Next Point) e ponto anterior (Prev Point). As instruções são visualizadas em `TextBlock` e a voz (Text-to-speech) é utilizada quando um ponto da rota é selecionado.

Esta aplicação é util para testar previamente a sua rota. Pode simular a rota utilizando a aplicação.



Um pequeno vídeo de demonstração, que mostra como a aplicação funciona (desculpem por neste momento o vídeo não ter som mas sempre que um ponto da rota é selecionado o texto irá ser lido com o text-to-speech):

The media player is loading...

Layout

Esta aplicação utiliza os controlos `Map`, `TextBlock` e `Button` no layout `MainPage.xaml`. O utilizador pode adicionar novos pontos de partida e chegada nas `TextBlocks` . Uma nova rota é obtida quando o botão "Get Route" é pressionado. O mapa é centrado no ponto inicial e pode-se utilizar o zoom através dos botões "+" e "-". O ponto seguintes e anterior na rota podem ser selecionados ao usar os botões "Next Point" e "Prev Point", respetivamente.

```
<Grid x:Name="ContentPanel" Grid.Row="1" Margin="12,0,12,0">
            <maps:Map x:Name="map" HorizontalAlignment="Left" VerticalAlignment="Top" Height="388" Width="456"/>
            <TextBlock HorizontalAlignment="Left" Margin="0,407,0,0" TextWrapping="Wrap" Text="Address 1:" VerticalAlignment="Top"/>
            <TextBlock HorizontalAlignment="Left" Margin="0,465,0,0" TextWrapping="Wrap" Text="Address 2:" VerticalAlignment="Top"/>
            <TextBox x:Name="address1TextBox" HorizontalAlignment="Left" Height="72" Margin="91,393,0,0" TextWrapping="Wrap" Vertica
            <TextBox x:Name="address2TextBox" HorizontalAlignment="Left" Height="72" Margin="91,452,0,0" TextWrapping="Wrap" Vertica
            <Button x:Name="getRouteButton" Content="Get Route" HorizontalAlignment="Left" Margin="0,604,0,0" VerticalAlignment="Top
            <Button x:Name="prevPointButton" Content="Prev Point" HorizontalAlignment="Left" Margin="153,604,0,0" VerticalAlignment=
            <Button x:Name="nextPointButton" Content="Next Point" HorizontalAlignment="Left" Margin="308,604,-7,0" VerticalAlignment
            <TextBlock Foreground="Black" x:Name="mainInfoTextBlock" HorizontalAlignment="Left" Margin="10,343,0,0" TextWrapping="Wr
            <TextBlock x:Name="directionTextBlock" HorizontalAlignment="Left" Margin="10,532,0,0" TextWrapping="Wrap" VerticalAlignm
            <Button Content="-" HorizontalAlignment="Left" Margin="375,302,0,0" VerticalAlignment="Top" Width="80" Click="mapZoomMin
            <Button Content="+" HorizontalAlignment="Left" Margin="375,244,0,0" VerticalAlignment="Top" Width="80" Click="mapZoomPlu
</Grid>
```

Programação

Esta aplicação ganha vida quando se pressiona o botão "Get Route". Inicialmente ambas as coordenadas geográficas, a inicial e a de destino, são colocadas a null e uma nova coordenada geográfica inicial é carregada.

```
private void getRouteButton_Click(object sender, RoutedEventArgs e)
{
            geo1 = null;
            geo2 = null;
            getGeoCoordinate(address1TextBox.Text);
}
```


(C) Copyright Nokia 2013. All rights reserved.

O método `getGeoCoordinate` irá ser executado para começar a carregar as coordenadas geográficas da morada. Não acontece nada se as `TextBlocks` das moradas encontrarem-se vazias caso contrario um novo objeto `GeocodeQuery` é criado e nova informação é consultada.

```
private void getGeoCoordinate(string address)
{
            if (address == "")
            {
                MessageBox.Show("Address cannot be empty!");
                return;
            }
            GeocodeQuery query = new GeocodeQuery()
            {
                GeoCoordinate = new GeoCoordinate(0, 0),
                SearchTerm = address
            };
            query.QueryCompleted += geoCoordinateQuery_QueryCompleted;
            query.QueryAsync();
}
```

O método `geoCoordinateQuery_QueryCompleted` irá ser executado, após uma nova coordenada geográfica ser encontrada. Se uma nova coordenada geográfica for encontrada para o ponto de partida, então uma nova coordenada geográfica para o ponto de destino começa a ser carregada. Depois das duas coordenadas geográficas terem sido carregadas e encontradas, é criada uma nova camada no mapa com os pontos de inicio e destino e o método `FindRoute` é executado.

```
void geoCoordinateQuery_QueryCompleted(object sender, QueryCompletedEventArgs<IList<MapLocation>> e)
{
            var item = e.Result;
            if (item.Count() == 0)
            {
                MessageBox.Show("No GeoCoordinate found!");
                return;
            }
            if (item.ElementAt(0).GeoCoordinate == null)
            {
                MessageBox.Show("No GeoCoordinate found!");
                return;
            }
            if (this.geo1 == null) {
                this.geo1 = item.ElementAt(0).GeoCoordinate;
                getGeoCoordinate(address2TextBox.Text);
            }
            else if (this.geo2 == null)
            {
                this.geo2 = item.ElementAt(0).GeoCoordinate;
            }
            // is both geos there
            if (this.geo1 != null && this.geo2 != null)
            {
                map.Center = geo1;
                map.ZoomLevel = 13;
                // remove possible previous one layer
                if (map.Layers.Count() > 0) map.Layers.Clear();
                AddMapLayer(geo1, Colors.Blue);
                AddMapLayer(geo2, Colors.Red);
                FindRoute();
            }
}
```

O ponto inicial e de destino são desenhados no mapa com um novo `MapLayer`. O ponto inicial será Azul e o de destino será Vermelho.

```
private void AddMapLayer(GeoCoordinate geo, Color color)
{
            map.Layers.Add(new MapLayer()
            {
                new MapOverlay()
                {
                    GeoCoordinate = geo,
                    PositionOrigin = new Point(0.5,0.5),
                    Content = new Ellipse
                    {
                        Fill = new SolidColorBrush(color),
                        Width = 20,
                        Height = 20
                    }
                }
            });
}
```

O método `FindRount` irá utilizar o `RouteQuery` com as coordenadas geográficas do ponto inicial e de destino como waypoints.

```
private void FindRoute()
{
            RouteQuery query = new RouteQuery()
            {
                TravelMode = TravelMode.Driving,
                Waypoints = new List<GeoCoordinate>()
                {
                    geo1,
                    geo2
                }
            };
            query.QueryCompleted += routeQuery_QueryCompleted;
            query.QueryAsync();
}
```

O método `routeQuery_QueryCompleted` irá ser executado quando o `RouteQuery` terminar. A rota anterior é removida inicialmente do controlo `Map` antes de uma nova rota ser inserida. Todos os pontos da rota são movidos para a lista `routePoints`, para que possam ser utilizados posteriormente com os botões "Prev Point" e "Next Point". Primeiro a informação da rota (ponto inicial) é visualizada.

```
void routeQuery_QueryCompleted(object sender, QueryCompletedEventArgs<Route> e)
{
            if (mapRoute != null) map.RemoveRoute(mapRoute);
```


(C) Copyright Nokia 2013. All rights reserved.

```
            routePoint = 0;
            mapRoute = new MapRoute(e.Result);
            map.AddRoute(mapRoute);
            mainInfoTextBlock.Text = "Distance: " + e.Result.LengthInMeters + " Time: " + e.Result.EstimatedDuration;

            routePoints = e.Result.Legs.SelectMany(l => l.Maneuvers).ToList();
            ShowRoutePointInfo();
}
```

O método `ShowRoutePointInfo` irá ser executado sempre que um novo ponto da rota ser selecionado. A informação da rota é visualizada no `TextBlock` e um novo ponto de localização é visualizado no mapa com o `MapLayer`. Um ponto na rota pode ser ouvido com o text-to-speech.

```
private void ShowRoutePointInfo() {
            // location
            GeoCoordinate geoCoordinate = routePoints.ElementAt(routePoint).StartGeoCoordinate;
            StringBuilder sb = new StringBuilder();
            // route info
            sb.AppendLine(routePoints.ElementAt(routePoint).InstructionText + " " +
                          routePoints.ElementAt(routePoint).LengthInMeters + " m");
            directionTextBlock.Text = sb.ToString();
            // delete previous one
            if (map.Layers.Count() > 2) map.Layers.RemoveAt(2);
            // show a new one
            AddMapLayer(geoCoordinate, Colors.Green);
            // center to this location
            map.Center = geoCoordinate;
            // play audio
            TTS_info(sb.ToString());
}
```

O método `TTS_info` irá ser executado sempre que um novo ponto na rota ser selecionado. Este método implementa uma seleção baseada na prioridade da coleção de vozes instaladas. A primeira prioridade (`prio` igual a `1`) é atribuída à voz correspondente ao UI atual. A segunda prioridade é atribuída ao idioma finlandês se disponível e a terceira prioridade é atribuída à língua inglesa. O ciclo de seleção itera por todas as vozes disponíveis e verifica se uma voz com mais prioridade (menor `prio` valor) é encontrada. Neste caso a voz atual é selecionada como voz corrente. Em muitos casos irá haver duas vozes por língua, uma masculina e uma feminina. Este método obtém a primeira voz disponível da língua, mas uma preferência entre uma voz masculina ou feminina é facilmente implementada com um schema de prioridades. Um novo objeto `SpeechSynthesizer` é então criado e a informação da rota é fornecida pela voz utilizando o método `SpeakTextAsync`.

```
private async void TTS_info(string text)
{
            VoiceInformation voiceToUse = null;
            string uiLanguage = System.Globalization.CultureInfo.CurrentUICulture.Name;
            int prio = 99;

            // prioritized voice selection
            foreach (var voice in InstalledVoices.All)
            {
                Debug.WriteLine(voice.Language);
                if (voice.Language.IndexOf(uiLanguage) != -1 && prio > 1)
                {// UI language first prio
                    voiceToUse = voice;
                    prio = 1;
                }
                else if (voice.Language.IndexOf("fi-FI") != -1 && prio > 2)
                {// finnish second prio
                    voiceToUse = voice;
                    prio = 2;
                }
                else if (voice.Language.IndexOf("en-US") != -1 && prio > 3)
                { // then english
                    voiceToUse = voice;
                    prio = 3;
                }
            }

            var text2speech = new SpeechSynthesizer();
            if (voiceToUse != null) text2speech.SetVoice(voiceToUse);
            await text2speech.SpeakTextAsync(text);
}
```

O ponto seguinte e anterior da rota podem ser selecionados com os botões "Next Point" e "Prev Point".

```
private void prevPointButton_Click(object sender, RoutedEventArgs e)
{
            routePoint--;
            if (routePoint < 0) routePoint = 0;
            else ShowRoutePointInfo();
}

private void nextPointButton_Click(object sender, RoutedEventArgs e)
{
            routePoint++;
            if (routePoint > routePoints.Count() - 1) routePoint = routePoints.Count()-1;
            else ShowRoutePointInfo();
}
```

O mapa pode ser ampliado ou reduzido utilizando os botões + e -.

```
private void mapZoomPlus(object sender, RoutedEventArgs e)
{
            map.ZoomLevel++;
}

private void mapZoomMinus(object sender, RoutedEventArgs e)
{
            map.ZoomLevel--;
}
```

Resumo

(C) Copyright Nokia 2013. All rights reserved.

Esta demonstração de código mostra como utilizar o mapa com direções no Windows Phone 8. Espero que este artigo lhe seja útil e o ajude a trabalhar com mapas e informações de rotas em aplicações para Windows Phone 8.

Pode fazer o download do código fonte aqui: File:PTMMapWithDirections.zip

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.