Map with directions in Windows Phone 8

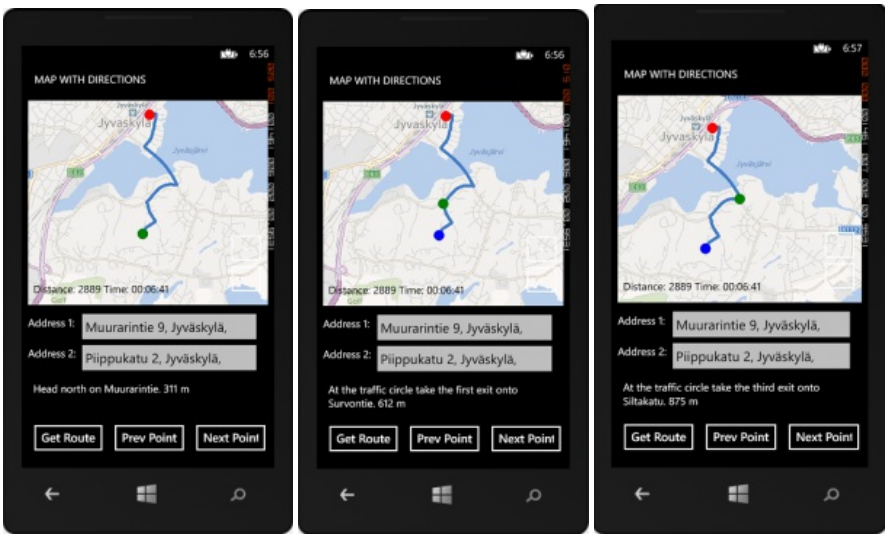
This article explains how to show route in map and get route directions (textual and speech) with Windows Phone 8.

03 Feb 2013

Introduction

This code example demonstrates how to show route with `Map Control` and get direction information with text and text to speech in Windows Phone 8. Start and destination addresses can be added with `TextBlocks`. First both addresses are calculated to geo coordinates and then locations are showed in the Map with map layers. `RouteQuery` is used with these two geo locations and route path is displayed with new map layer. User can simulate route by pressing Prev or Next Point Buttons. Text instructions are show in `TextBlock` and Text-to-speech is used when a new turning point is selected.

This application is useful when you want to test your route beforehand. You can simulate the route with this.



Here is a small demo video, which shows how the application works (sorry there is no audio included in video right now but every time when a new turning point is selected the text will be read with text-to-speech):

The media player is loading...

Layout

This application uses `Map`, `TextBlock` and `Button` Controls in **MainPage.xaml** layout. User can add a new starting and destination locations to `TextBlocks` . A new route is fetched when Get Route Button is pressed. Map is centred to starting location and it can be zoomed with + and - Buttons. Next and previous turning points can be selected using Prev and Next Point Buttons.

```
<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>
```

Programming

This application comes alive when Get Route Button is clicked. First both starting and destination geo coordinates are set to null and a new geo coordinate of starting point is loaded.

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

`getGeoCoordinate` method will be called to start loading geo coordinate of the address. Nothing happens if addresses TextBlocks are empty otherwise a

(C) Copyright Nokia 2013. All rights reserved.

new `GeocodeQuery` object is created and new data is queried.

```
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();
}
```

`geoCoordinateQuery_QueryCompleted` method will be called, after a new geo coordinate is found. If a new geo coordinate is found for the starting point, then a new geo coordinate for the destination point is started to load. After both of these geo coordinates are loaded and found, a new map layers are made to show starting and destination points and `FindRoute` method is called.

```
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();
            }
}
```

Starting and destination points are drawn to map with a new `MapLayer`. Staring point will be Blue and destination point will be Red.

```
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
                    }
                }
            });
}
```

`FindRount` method will use `RouteQuery` with starting and destination geo coordinates as a waypoints.

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

`routeQuery_QueryCompleted` method will be called when `RouteQuery` is finished. Previous route is first removed from the `Map` Control before a new one is added. All the route points are moved to `routePoints` List, so they can be used later with Prev and Next Point Buttons. First (starting point) route info is displayed.

```
void routeQuery_QueryCompleted(object sender, QueryCompletedEventArgs<Route> e)
{
            if (mapRoute != null) map.RemoveRoute(mapRoute);
            routePoint = 0;
            mapRoute = new MapRoute(e.Result);
            map.AddRoute(mapRoute);
            mainInfoTextBlock.Text = "Distance: " + e.Result.LengthInMeters + " Time: " + e.Result.EstimatedDuration;
```


(C) Copyright Nokia 2013. All rights reserved.

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

`ShowRoutePointInfo` method will be called every time when a new route point is selected. Route info is displayed nn `TextBlock` and a new location point is showed in Map with `MapLayer`. Route turning point can be hear with 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());
}
```

The `TTS_info` method will be called every time a new route point is selected.

This method implements a priority based selection from the collection of installed voices. The first priority (`prio` equals 1) is given to a voice in the current UI language. Second priority is given to a Finnish voice if available and third priority is an English voice. The selection loop iterates over the available voices and checks if a voice with higher priority (smaller `prio` value) is found. In this case the actual voice is taken as current selection.

In most cases there will be two voices per language, one male and one female. This method takes the first available voice of a language, but a preference for male or female voices can easily be implemented with the priority scheme, too.

A new `SpeechSynthesizer` object is then created and the route information is spoken with the `SpeakTextAsync` method.

```
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);
}
```

A next or previous route point can be selected with Prev and Next Point Buttons.

```
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();
}
```

Map can be zoomed in and out with + and - Buttons.

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

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

Summary

(C) Copyright Nokia 2013. All rights reserved.

This code example shows how to use Map with directions in Windows Phone 8. Hope you find this article useful and it helps you work with map and route information in your Windows Phone 8 application.

You can download source code from here: File:PTMMapWithDirections.zip

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.