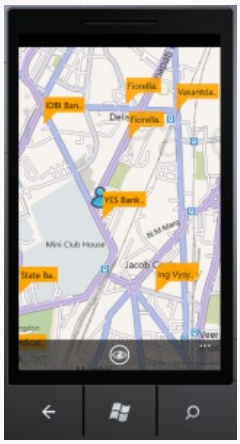
# Google Landmarks on a Bing Map for WP7

This article demonstrates how to plot a Google landmark on Bing Map on Windows Phone.

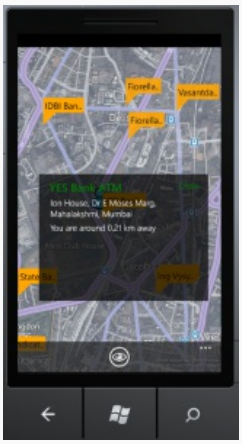
## Introduction

This example uses Google Places API to get landmark information that we then plot on a Bing Map. When pressed, the Pushpins display a popup with the landmark information and the distance of the landmark from user's location.

Land Mark Category



Pushpin on Landmarks



Land Mark information detail

## Preconditions

First we need to have a Bing Map account from Bing Maps Account Center with a Windows Live ID. Once the account is being created, get the **KEY** from the account to use in our application.

## Implementation

Lets create a WP7 project using *Windows Phone Application* template. We have created a list of landmark category on `MainPage.xaml` using WP7 ListBox control. When user clicks on any of the list item, the `Menu_SelectionChanged` event handler is called in `MainPage.xaml.cs` class.

When the Map loads, first we check the ability to provide location update with GeoCoordinateWatcher:: StatusChanged event and then check the user position changed with GeoCoordinateWatcher:: PositionChanged.

```
void myStatusChanged(GeoPositionStatusChangedEventArgs e)
        {
            switch (e.Status)
            {
                case GeoPositionStatus.Disabled:

                    progressBar.IsIndeterminate = false;
                    progressBar.Visibility = Visibility.Collapsed;
                    MessageBox.Show("location is unsupported on this device");
                    break;
                case GeoPositionStatus.Initializing:

                    break;
                case GeoPositionStatus.NoData:

                    progressBar.IsIndeterminate = false;
                    progressBar.Visibility = Visibility.Collapsed;
                    MessageBox.Show("data unavailable");
                    break;
                case GeoPositionStatus.Ready:
                    break;
            }
        }//end of funt
```

```
void myPositionChanged(GeoPositionChangedEventArgs<GeoCoordinate> e)
        {
            latitude = e.Position.Location.Latitude;
            longitude = e.Position.Location.Longitude;
            LoadLandMarkList();
        }//end of funt
```

The `LoadLandMarkList()` method download and parse the XML data from Google Places API. Where we get *name*, *address*, *latitude* and *longitude* of each landmark. To access the Google Places API please get the key from here and replace **<Your Google API Key>** with the key in the below code.


(C) Copyright Nokia 2013. All rights reserved.

```
void LoadLandMarkList()
        {
            string url = "https://maps.googleapis.com/maps/api/place/search/xml?location=" + latitude + "," + longitude + "&radius=1

            WebClient xmlClient = new WebClient();
            xmlClient.DownloadStringCompleted += new DownloadStringCompletedEventHandler(LoadLandMarkList_Completed);
            xmlClient.DownloadStringAsync(new Uri(url, UriKind.RelativeOrAbsolute));
        }//end of funct

        void LoadLandMarkList_Completed(object sender, DownloadStringCompletedEventArgs e)
        {
            if (e.Error == null)
            {
                LandMarkCount = 0;

                XElement xmlDataStill = XElement.Parse(e.Result);
                foreach (var item in xmlDataStill.Descendants("result"))
                {
                   string name = (string)item.Element("name").Value;
                   string address = (string)item.Element("vicinity").Value;
                   string latitude = (string)item.Element("geometry").Element("location").Element("lat").Value;
                   string longitude = (string)item.Element("geometry").Element("location").Element("lng").Value;

                   LandMarkPositionArray.SetValue(new RssLandMark() { name = name, address = address, longitude = longitude, latitud
                   LandMarkCount++;
                 }


                if (LandMarkCount > 0)
                {
                    PlotUserAndLandMarks();
                }
                else
                {
                    MessageBox.Show("There is no information available for your location");
                    progressBar.IsIndeterminate = false;
                    progressBar.Visibility = Visibility.Collapsed;
                }
            }
        }//end of funt
```

Once we get the landmarks data, we then count the total number of landmark and plot Pushpin on Bing Map for each landmark.

```
for (int i = 0; i < LandMarkCount; i++)
            {
                RssLandMark st = LandMarkPositionArray.GetValue(i) as RssLandMark;
                Pushpin landMark = new Pushpin();
                landMark.Location = new GeoCoordinate(Convert.ToDouble(st.latitude), Convert.ToDouble(st.longitude));

                if (st.name.Length > 8)
                {

                    LandMarkTitle = st.name.Substring(0, 8) + "..";
                }

                landMark.Foreground = new SolidColorBrush(Colors.Black);
                landMark.Background = new SolidColorBrush(Colors.Orange);
                landMark.Content = LandMarkTitle;
                landMark.FontSize = 21;
                landMark.Opacity = .9;
                landMark.Tag = i + "";
                landMark.Tap += new EventHandler<System.Windows.Input.GestureEventArgs>(showandMarkDetails_Completed);

                map1.Children.Add(landMark);
            }
```

When users clicks on any of the Pushpin we call `EventHandler` to display a message with the landmark address and distance of the landmark from user's location.

```
   void showandMarkDetails_Completed(object sender, System.Windows.Input.GestureEventArgs e)
        {
            int id = Convert.ToInt32((sender as Pushpin).Tag);
            RssLandMark st = LandMarkPositionArray.GetValue(id) as RssLandMark;

            title_txt.Text = st.name;
            address_txt.Text = st.address;
           dist_txt.Text = "You are around " + calcutaleDistance(latitude, longitude, Convert.ToDouble(st.latitude), Convert.ToDoubl

            popup.Visibility = Visibility.Visible;
            bg.Visibility = Visibility.Visible;

            }//end of funt
```

The `calcutaleDistance()` method is called to calculate the distance between landmark and user's location.

```
public static string calcutaleDistance(double ulat, double ulon, double lat, double lon)
        {
            var sCoord = new GeoCoordinate(ulat, ulon);
            var eCoord = new GeoCoordinate(lat, lon);

            var R = 6371;
            double lat1 = (sCoord.Latitude) / 180 * Math.PI;
            double lat2 = (eCoord.Latitude) / 180 * Math.PI;

            double lng1 = (sCoord.Longitude) / 180 * Math.PI;
            double lng2 = (eCoord.Longitude) / 180 * Math.PI;

            double dlng = lng2 - lng1;
            double dlat = lat2 - lat1;

            var a = Math.Pow(Math.Sin(dlat / 2), 2) + Math.Cos(lat1) *
                            Math.Cos(lat2) * Math.Pow(Math.Sin(dlng / 2), 2);
            var c = 2 * Math.Asin(Math.Min(1, Math.Sqrt(a)));

            var d = R * c;

            return (String.Format("{0:0.00}", d) + " km");
        }//end of funt
```


(C) Copyright Nokia 2013. All rights reserved.

We have also added an application bar to change the Bing Map view mode.

```
private void ApplicationBarIconButton_Click(object sender, EventArgs e)
        {
            // TODO : Change map view mode.
            if (map1.Mode is AerialMode)
            {
                map1.Mode = new RoadMode();
            }
            else
            {
                map1.Mode = new AerialMode(true);
            }
        }//end of funt
```

## Summary

This is a basic Bing Map operation, there are huge resources on Bing Map in MSDN. I found one here. But this article shows how simple and flexible to operate Bing Map Controls.

## Source Code

The full source code of the example is available here: File:GoogleLandmarksWP7.zip


(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.