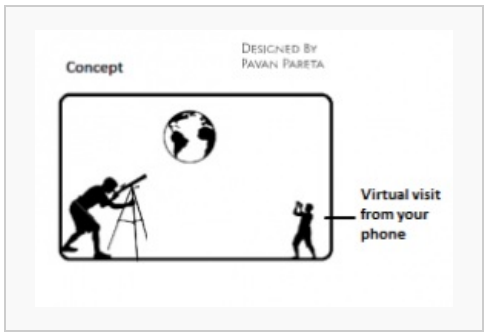
How to use Google - Street View Image API in Windows Phone

This article explains how to use the static Google Street View Image API in a Windows Phone app.

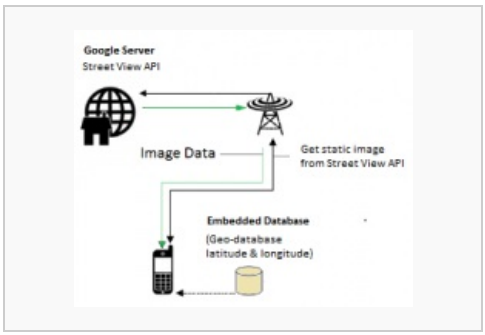
Introduction

Google Street View Image API is a Google service which provides a way to add a Street View image or thumbnail to your phone application.

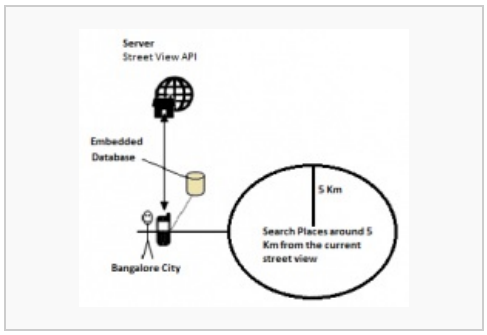
The idea behind to build this application is to give you a basic actual glance of the city from where you are currently so you can search/compare the nearby points of interest. Here is simple conceptual diagram that shows the application objective.



Application Concept



Application Architecture



Application Functionality

The media player is loading...

Prerequisites

For Windows Phone 7.5

- Windows 7 OS or above
- Windows Phone SDK 7.5
- Visual Studio Express Edition 2010 for Windows Phone

For Windows Phone 8

- Windows 8 OS
- Visual Studio 2012 or Microsoft Visual Studio Express 2012 for Windows Phone
- Windows Phone SDK 8.0

API Used

- DispatcherTimer
- WebClient
- MediaLibrary

Get Google API Key

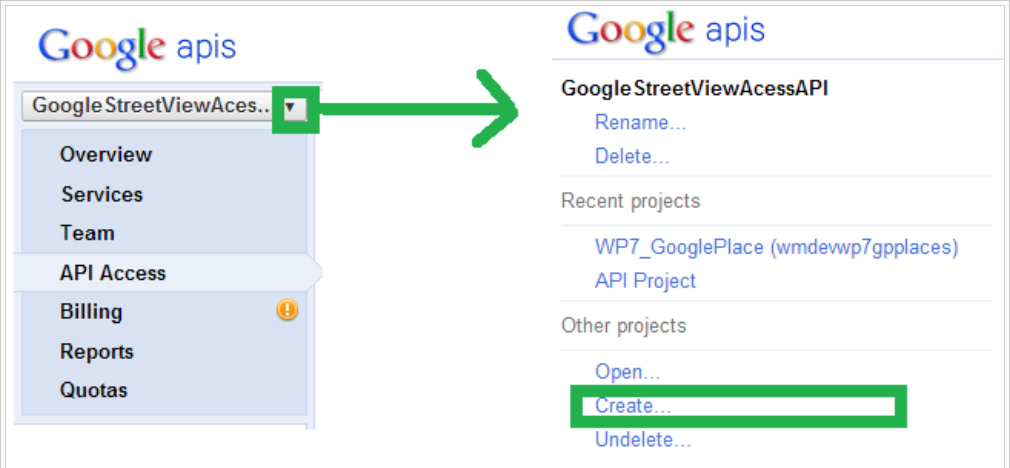
To use the *Street View Image API* in your Windows Phone application you must first register the application and get an **API Key**.

(C) Copyright Nokia 2014. All rights reserved.

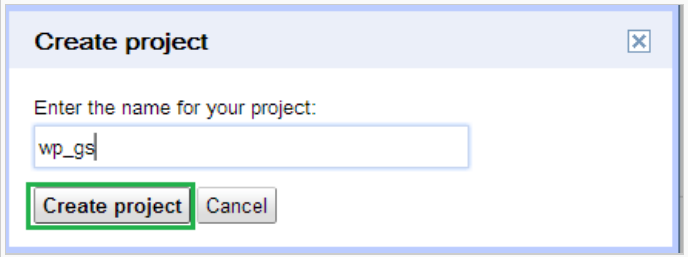
1. Get your valid API Key from here with your valid Google credentials.
2. Must read Google Street View Image API documentation here to understand the service parameters.

Follow the steps below to get access of Google Street View Image API Key.

1. Create new project to access Google API Key. Click on Drop down menu and click on Create in Other Project section -> Create Project popup will show then enter your desired project name and click on **Create project** button, refer the below image screen.

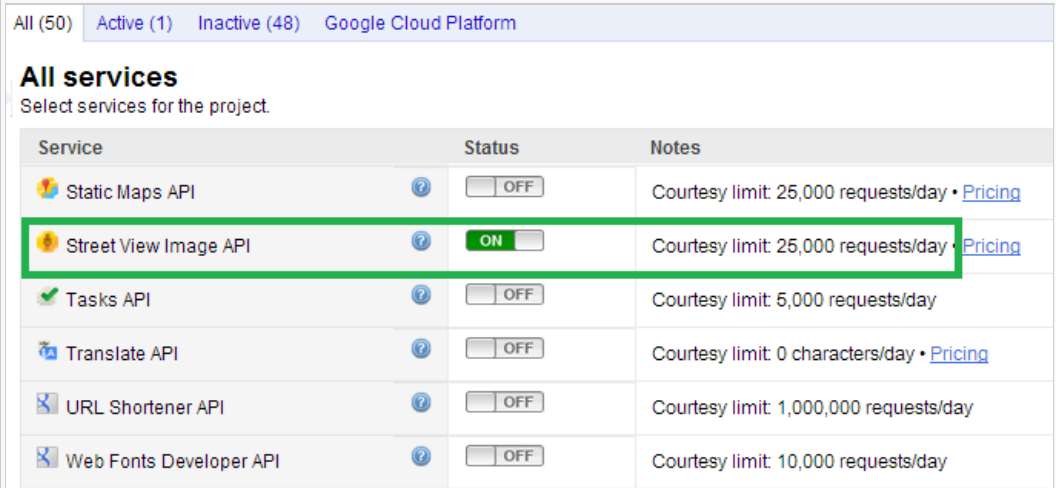


Create new project to access Google API Key



Create new project

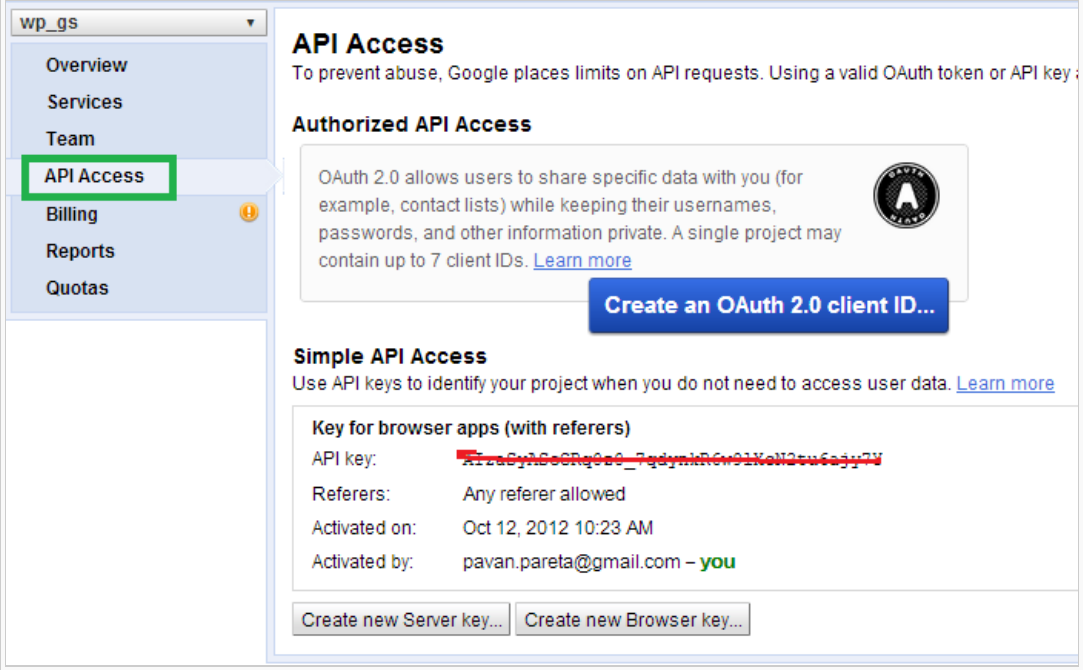
2. Once you click on Create Project Button you will see all the Google services, to active a particular service must have to set ON status of required service such as Street View Image API. This API basically allows developer to make 25000 requests per day for free. Otherwise you can also go for purchase; it has pricing option see below image.



Access Google API Key

3. Now click on API Access and get your API Key.

(C) Copyright Nokia 2014. All rights reserved.



API Access and get your API Key

Coverage Limitations: - Street View Image API supports 48 countries only.

Note: Due to coverage limitations this application uses static Geo location of a city. For the implementation purpose I have used few cities static Geo locations data that is call embedded database.

Implementation

Step 1: Create the Windows Phone application using traditional way.

Do **File -> New Project -> Silverlight for Windows Phone** and choose "Windows Phone Application" Template. Here I have named the application "WmDev_GoogleStreet".

Step 2: Design UI (User interface) for displaying cities in list which contains the static data.

```
<Grid x:Name="ContentPanel" Grid.Row="1" Margin="12,0,12,0">
            <ListBox Margin="12,75,12,0" Name="lstCities"
SelectionChanged="lstCities_SelectionChanged_1" Foreground="#FF4AFFFF">
                <ListBoxItem Tag="60.1656,024.9334" FontSize="30">Åland
Islands(Finland)</ListBoxItem>
                <ListBoxItem Tag="33.4672,-112.0410"
FontSize="30">Phoneix</ListBoxItem>
                <ListBoxItem Tag="44.9790,-093.2831"
FontSize="30">Minneapolis</ListBoxItem>
                <ListBoxItem Tag="30.2660,-097.7538"
FontSize="30">Austin</ListBoxItem>
                <ListBoxItem Tag="45.0864,-093.2634"
FontSize="30">Fridley</ListBoxItem>
            </ListBox>
</Grid>
```

Design View: **CitiesList.xaml** It shows the cities name in list and the `ListBoxItem` contains a `Tag` property and it hold the Geo-locations.

(C) Copyright Nokia 2014. All rights reserved.

Design View

CitiesList.xaml.cs contains an event of list control called `lstCities_SelectionChanged`, it allows user to navigate on street view screen.

```
private void lstCities_SelectionChanged_1(object sender, SelectionChangedEventArgs e)
        {
            var value = (lstCities.SelectedItem as ListBoxItem).Tag;
            var name = (lstCities.SelectedItem as ListBoxItem).Content;
            string url = string.Format("/MainPage.xaml?value={0}&city={1}", value,
name);
            NavigationService.Navigate(new Uri(url, UriKind.Relative));
        }
```

Step 3: Layout designing using XAML code; to build the required user interface using standard controls such as `Grid`, `StackPanel`, `Image`, `Buttons` and `TextBlocks` in **MainPage.xaml**.

XAML Code Snippet:

```
<Grid x:Name="LayoutRoot" Background="Transparent">
        <Grid x:Name="ContentPanel" Margin="12,0" Grid.ColumnSpan="2">
            <Grid.RowDefinitions>
                <RowDefinition Height="73*" />
                <RowDefinition Height="407*" />
            </Grid.RowDefinitions>

            <StackPanel Orientation="Horizontal" Grid.Row="0">
                <Button Content="Download" Height="72" HorizontalAlignment="Left"
Name="btnDownload" VerticalAlignment="Top" Width="auto" Click="btnDownload_Click" />
                <Button Content="Offline" Height="72" Name="btnOffline"
VerticalAlignment="Top" Width="auto" Click="btnOffline_Click" />

                <Grid x:Name="infoGrid" Width="482" ShowGridLines="True">
                    <Grid.RowDefinitions>
                        <RowDefinition Height="Auto" />
                        <RowDefinition Height="*" />
                    </Grid.RowDefinitions>
```

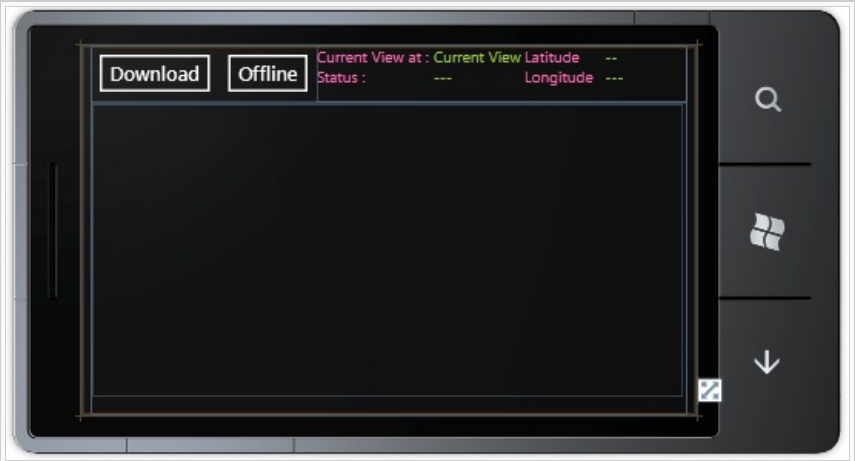

(C) Copyright Nokia 2014. All rights reserved.

```
                    <Grid.ColumnDefinitions>
                        <ColumnDefinition Width="265"/>
                        <ColumnDefinition Width="110"/>
                        <ColumnDefinition Width="*"/>
                    </Grid.ColumnDefinitions>

                    <TextBlock Foreground="HotPink" Height="auto" Width="auto"
Name="txtView" Text="Current View at : " Margin="0,0,113,0" />
                    <TextBlock Foreground="YellowGreen" Height="auto" Width="auto"
Name="txtViewValue" Text="Current View" Margin="152,0,95,0" Grid.ColumnSpan="2" />
                    <TextBlock Foreground="HotPink" Grid.Row="1" Height="auto"
Width="auto" Name="txtStatus" Text="Status :" Margin="0,0,113,0" />
                    <TextBlock Foreground="YellowGreen" Grid.Row="1" Height="auto"
Name="txtStatusValue" Text="---" Margin="152,0,95,0" Grid.ColumnSpan="2" />

                    <TextBlock Foreground="HotPink" Grid.Column="1" Name="txtLat"
Text="Latitude" Margin="5,0,0,0" />
                    <TextBlock Foreground="YellowGreen" Grid.Column="2" Height="auto"
Width="auto" Name="txtLatValue" Text="--" />
                    <TextBlock Foreground="HotPink" Grid.Column="1" Grid.Row="1"
Name="txtLong" Text="Longitude" Margin="5,0,0,0" />
                    <TextBlock Foreground="YellowGreen" Grid.Column="2" Grid.Row="1"
Height="auto" Width="auto" Name="txtLongValue" Text="---" />

                </Grid>
            </StackPanel>

            <StackPanel Orientation="Horizontal" Grid.Row="1">
                <Image Grid.Row="1" Height="380" HorizontalAlignment="Left"
Margin="2,2,0,0" Name="image1" Stretch="Fill" VerticalAlignment="Top" Width="768"
Tap="image1_Tap" />
            </StackPanel>

            <StackPanel x:Name="stListBox" Orientation="Horizontal" Grid.Row="1"
Visibility="Collapsed">
                <ListBox x:Name="PlacesList">
                    <ListBox.ItemTemplate>
                        <DataTemplate>
                            <Grid>
                                <TextBlock Margin="20,5,5,5" Text="{Binding PlaceName}"
FontSize="30" Foreground="White"  />
                            </Grid>
                        </DataTemplate>
                    </ListBox.ItemTemplate>
                </ListBox>
            </StackPanel>

            <StackPanel x:Name="dowloadDetailBoard" Visibility="Collapsed"
Orientation="Vertical" Grid.Row="1" Background="Black" Opacity=".8"
Margin="10,247,10,35">
                <StackPanel Orientation="Vertical">
                    <TextBlock x:Name="TextBlock2" Text="Download Status"
Margin="10,5,10,10" />
                    <TextBlock x:Name="txtCityName" Text="City Name:"
Margin="10,0,10,10" />
                </StackPanel>
                <ProgressBar Background="White" x:Name="ProgressBar1"
```


(C) Copyright Nokia 2014. All rights reserved.

```
HorizontalAlignment="Right" Width="756" Height="13"  />
            </StackPanel>
            <StackPanel x:Name="nearByHit" Visibility="Collapsed" Orientation="Vertical"
Grid.Row="1" Background="Black" Opacity=".8" Margin="10,247,10,35">
                <StackPanel Orientation="Vertical">
                    <TextBlock x:Name="Places" Text="Near by places" Margin="10,5,10,10"
/>
                    <StackPanel Orientation="Horizontal">
                        <Image Margin="90,10,10,20" Tap="bankClick"
Source="/img/bank.png" Height="50" Width="48" />
                        <Image Margin="70,10,10,20" Tap="hospitalClick"
Source="/img/hospital.png" Height="50" Width="48" />
                        <Image Margin="70,10,10,20" Tap="shoppingClick"
Source="/img/shopping.png" Height="50" Width="48" />
                        <Image Margin="70,10,10,20" Tap="hotelClick"
Source="/img/hotel.png" Height="50" Width="48" />
                        <Image Margin="70,10,10,20" Tap="cinemaClick"
Source="/img/cinema.png" Height="50" Width="48" />
                    </StackPanel>
                </StackPanel>
            </StackPanel>
        </Grid>
    </Grid>
```

Design View:



Design View

Here are controls and their functionality details.

Download Button: - It allows you to invoke Street View Image API with required parameter.
Offline Button: - It allows you to view images at 360 degree (after the images downloaded on your device)
Current View at: - It shows the current view in degree.
Status: - It shows the current application working status like (Online/Offline).
Latitude: - It shows the current geo-location's Latitude.
Longitude: - It shows the current geo-location's Longitude.

Step 4: Now moving to the core coding part (**MainPage.xaml.cs**) as per application required this solution use the following APIs.

- DispatcherTimer
- WebClient
- MediaLibrary
- BitmapImage
- WriteableBitmap
- ProgressBar

(C) Copyright Nokia 2014. All rights reserved.

Here I have divided task which we need to achieve such kind of functionality.

1. Download images for street view using street view API by Pressing Download
2. Save images in Media Library.
3. Offline button basically a used to show all saved images from Media Library.
4. Search places nearby, specific city which is in “street view” by using Google Places API
5. Show a list of nearby places of the current street view in specific radius

Class files are used in application

- AppInfo.cs
- GoogleLandMarkItem.cs
- Util_ISO.cs

  `AppInfo` class is a static class that contents the Google API_KEY.

  `GoogleLandMarkItem` class is property or entity class that contents the nearby places information

  `Util_ISO` class is a static class that basically handles the `ApplicationSettings` using `IsolatedStorage` operations.

MainPage.cs : Required objects are initialized in the Constructor and Load event.

Code Snippet:

```
public MainPage()
        {
            InitializeComponent();

            // ----------- Init Timer --------------
            downloadTimer = new DispatcherTimer();
            downloadTimer.Interval = TimeSpan.FromMilliseconds(1500); //delayed for 1.5
seconds
            downloadTimer.Tick += new EventHandler(downloadTimer_Tick);

            // ----------- Init offline Timer --------------
            offlineTimer = new DispatcherTimer();
            offlineTimer.Interval = TimeSpan.FromMilliseconds(1500); //delayed for 1.5
seconds
            offlineTimer.Tick += new EventHandler(offlineTimer_Tick);

            Loaded += MainPage_Loaded;
        }

void MainPage_Loaded(object sender, RoutedEventArgs e)
        {
            string vlu = NavigationContext.QueryString["value"];
            strCityName = NavigationContext.QueryString["city"];
            strFileName = strCityName + vlu.Replace('.', '-').Replace(',', '_');
            string[] geoData = vlu.Split(',');

            Latitude = double.Parse(geoData[0]);
            Longitude = double.Parse(geoData[1]);
        }
```

Download button's event allows user to initiate the image download process using download timer. Download timer tick event called every 1.5 second and download an image using `WebClient` API, here `WebClient` object has two events as required `OpenReadCompletedEventHandler` and `DownloadProgressChangedEventArgs`.

 (C) Copyright Nokia 2014. All rights reserved.

`DownloadProgressChangedEventArgs`: show the current progress of the download image.

`OpenReadCompletedEventHandler`: it raised when an image data completely download.

Once the download timer stopped, then `Util_ISO.Save()` method saved the file information with respect to specific city in key value pair in `ApplicationSetting` using `IsolatedStorage` API.

Code Snippet:

```
private void btnDownload_Click(object sender, RoutedEventArgs e)
        {
            downloadTimer.Start();
            dowloadDetailBoard.Visibility = System.Windows.Visibility.Visible;
            btnDownload.IsEnabled = false;
            btnOffline.IsEnabled = false;
        }

private void downloadTimer_Tick(object sender, EventArgs e)
        {
            if (degree >= 360)
            {
                degree = degreeValue;
                downloadTimer.Stop();
                dowloadDetailBoard.Visibility = System.Windows.Visibility.Collapsed;
                btnDownload.IsEnabled = true;
                btnOffline.IsEnabled = true;
                Util_ISO.Save(strCityName, strKeepStoreFileName);
            }
            else
            {
                downloadTimer.Stop();
                GetImage(degree.ToString());
            }
        }
private void GetImage(string heading)
{
            string lat = "";
            string lng = "";



            lat = Latitude.ToString();
            lng = Longitude.ToString();


            string url = string.Empty;
            if (true)
            {
                url = string.Format("http://maps.googleapis.com/maps/api/streetview?
size=600x300&location={0},{1}&heading={2}&fov={3}&pitch={4}&sensor=false&key={5}", lat,
lng, heading, fov, pitch, AppInfo.API_KEY);
            }
            else
            {
                url = string.Format("http://maps.googleapis.com/maps/api/streetview?
size=600x300&location={0},{1}&sensor=false&key={2}", lat, lng, AppInfo.API_KEY);
            }
```


(C) Copyright Nokia 2014. All rights reserved.

```
            WebClient wc = new WebClient();
            wc.DownloadProgressChanged += wc_DownloadProgressChanged;
            wc.OpenReadCompleted += new
OpenReadCompletedEventHandler(wc_OpenReadCompleted);
            wc.AllowReadStreamBuffering = true;
            wc.OpenReadAsync(new Uri(url), wc);
}

void wc_DownloadProgressChanged(object sender, DownloadProgressChangedEventArgs e)
        {
            ProgressBar1.Value = e.ProgressPercentage;
            TextBlock2.Text = e.ProgressPercentage + "% done " + e.BytesReceived + "
bytes received of " + e.TotalBytesToReceive + " Total bytes.";
            txtCityName.Text = strCityName;
        }

void wc_OpenReadCompleted(object sender, OpenReadCompletedEventArgs e)
        {
            if (e.Error == null && !e.Cancelled)
            {
                try
                {
                    ProgressBar1.Value = 0;
                    SaveImage(e.Result, strFileName);
                }
                catch (Exception ex)
                {
                    MessageBox.Show("Error:-" + ex.Message);
                }
            }
            else
            {
                MessageBox.Show("Image Stream Error");
            }

            UpdateUI();
            downloadTimer.Start();
        }
```

Save an image in MediaLibrary and update the UI data.

Code Snippet:

```
private void SaveImage(Stream sr, string imageName)
        {
            intFileCounter++;
            string strFileNameTemp = string.Format(strFileName + "_ID_{0}.jpg",
intFileCounter);

            var streamResourceInfo = new StreamResourceInfo(sr, null);

            var userStoreForApplication =
IsolatedStorageFile.GetUserStoreForApplication();
            if (userStoreForApplication.FileExists(strFileNameTemp))
            {
                userStoreForApplication.DeleteFile(strFileNameTemp);
```

 (C) Copyright Nokia 2014. All rights reserved.

```
            }

            var isolatedStorageFileStream =
userStoreForApplication.CreateFile(strFileNameTemp);

            var bitmapImage = new BitmapImage { CreateOptions = BitmapCreateOptions.None
};
            bitmapImage.SetSource(streamResourceInfo.Stream);

            var writeableBitmap = new WriteableBitmap(bitmapImage);
            writeableBitmap.SaveJpeg(isolatedStorageFileStream,
writeableBitmap.PixelWidth, writeableBitmap.PixelHeight, 0, 85);

            isolatedStorageFileStream.Close();
            isolatedStorageFileStream =
userStoreForApplication.OpenFile(strFileNameTemp, FileMode.Open, FileAccess.Read);

            // Save the image to the camera roll or saved pictures album.
            var mediaLibrary = new MediaLibrary();

            // Save the image to the saved pictures album.
            mediaLibrary.SavePicture(strFileNameTemp, isolatedStorageFileStream);

            isolatedStorageFileStream.Close();
            GetMediaLibPhoto(strFileNameTemp, false);

        }

private void UpdateUI()
        {
            txtViewValue.Text = degree.ToString();
            txtStatusValue.Text = strStatus;

            txtLatValue.Text = Latitude.ToString();
            txtLongValue.Text = Longitude.ToString();
        }
```

After saved image, need to update street view image over the street view screen. To a get image from the Media Library.

```
private void GetMediaLibPhoto(string photoName, bool callOffline)
        {
            if (!callOffline)
            {
                if (string.IsNullOrEmpty(strKeepStoreFileName))
                {
                    strKeepStoreFileName = photoName;
                }
                else
                {
                    strKeepStoreFileName += "," + photoName;
                }
            }

            MediaLibrary myMediaLibrary = new MediaLibrary();
            foreach (var imgData in myMediaLibrary.SavedPictures)
            {
```


(C) Copyright Nokia 2014. All rights reserved.

```
                if (string.Compare(imgData.Name, photoName) == 0)
                {
                    Stream imageStream = imgData.GetImage();
                    BitmapImage bmp = new BitmapImage();
                    bmp.SetSource(imageStream);
                    image1.Source = bmp;
                    this.UpdateLayout();
                    break;
                }
            }
            degree = degree + degreeValue;
        }
```

Step 5: Once the all images downloaded you can view them in 360 degree view by using Offline button.

Code snippet:

```
private void btnOffline_Click(object sender, RoutedEventArgs e)
        {
            degree = degreeValue;
            intFileCounter = 0;
            strStatus = "Offline";

            if (strKeepStoreFileName == "")
            {
                strKeepStoreFileName = Util_ISO.Get(strCityName);
                files = strKeepStoreFileName.Split(',');
            }
            else
            {
                files = strKeepStoreFileName.Split(',');
            }
            offlineTimer.Start();
            UpdateUI();
        }

private void offlineTimer_Tick(object sender, EventArgs e)
        {
            if (files.Length > intFileCounter)
            {
                GetMediaLibPhoto(files[intFileCounter], true);
                intFileCounter++;
                UpdateUI();
            }
            else
            {
                intFileCounter = 0;
                offlineTimer.Stop();
            }
        }
```

Step 6: User can navigate the nearby places menu by tap on the street view screen. It contains five buttons that basically has a specific category to search nearby places such as (bank, hospitals, shopping malls, hotel and cinema). Here is a menu design in xaml code as well as design view.

XAML Code Snippet:

(C) Copyright Nokia 2014. All rights reserved.

```
<StackPanel x:Name="nearByHit" Visibility="Visible" Orientation="Vertical" Grid.Row="1"
Background="Black" Opacity=".8" Margin="10,247,10,35">
                <StackPanel Orientation="Vertical">
                    <TextBlock x:Name="Places" Text="Near by places" Margin="10,5,10,10"
/>
                    <StackPanel Orientation="Horizontal">
                        <Image Margin="90,10,10,20" Tap="bankClick"
Source="/img/bank.png" Height="50" Width="48" />
                        <Image Margin="70,10,10,20" Tap="hospitalClick"
Source="/img/hospital.png" Height="50" Width="48" />
                        <Image Margin="70,10,10,20" Tap="shoppingClick"
Source="/img/shopping.png" Height="50" Width="48" />
                        <Image Margin="70,10,10,20" Tap="hotelClick"
Source="/img/hotel.png" Height="50" Width="48" />
                        <Image Margin="70,10,10,20" Tap="cinemaClick"
Source="/img/cinema.png" Height="50" Width="48" />
                    </StackPanel>
                </StackPanel>
</StackPanel>
```

Design View:

Menu Bar-Design View

Step 7: Each button's tap event handler called specific category of places, using Google Places API, here I have written a `GetLocation()` generic method that is called by all the places buttons with specific parameter value. This value is a places name. The `GetLocation()` method is responsible to invoke the Google Places API by using C# managed `WebClient` API and it's response would be handled in the `GoogleLocationAPIService_DownloadStringCompleted` event, and xml response is parsed by the `ParseData()` method and build a listbox control which will appearing on the street view screen.

Code Snippet:

```
private void GetLocation(string placeName)
        {
            WebClient GoogleLocationAPIService = new WebClient();
            string url = "https://maps.googleapis.com/maps/api/place/search/xml?
location=" + Latitude + "," + Longitude + "&radius=" + intRadius + "&name=" + placeName
+ "&sensor=false&key=AIzaSyBL-9iS6XdNFRB1m_FFMjEG-x1x8YOuhPE";
            GoogleLocationAPIService.DownloadStringAsync(new Uri(url));
            GoogleLocationAPIService.DownloadStringCompleted += new
DownloadStringCompletedEventHandler(GoogleLocationAPIService_DownloadStringCompleted);
        }


private void GoogleLocationAPIService_DownloadStringCompleted(object sender,
DownloadStringCompletedEventArgs e)
        {
            if (e.Error == null)
            {
                PlacesList.ItemsSource = ParseData(e.Result);
```


(C) Copyright Nokia 2014. All rights reserved.

```
                stListBox.Visibility = System.Windows.Visibility.Visible;
            }
        }
        private static List<GoogleLandMarkItem> ParseData(string resultXml)
        {
            List<GoogleLandMarkItem> oList = new List<GoogleLandMarkItem>();
            GoogleLandMarkItem.CurrentLocation = new GeoCoordinate(Latitude, Longitude);

            XElement xmlDataStill = XElement.Parse(resultXml);
            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;
                oList.Add(new GoogleLandMarkItem { PlaceName = name, Location = new
GeoCoordinate(System.Convert.ToDouble(latitude), System.Convert.ToDouble(longitude)) });
            }
            return oList;
        }
```

Step 8: Now it's time to run the application, to play with Google Street and Google Places API

Downloads

- To get the complete source code of this example here: File:WmDevGoogleStreetAPI.zip

References

- Google Street View
- Google Street View Image API
- Google Places API
- DispatcherTimer
- WebClient Class
- MediaLibrary Class
- BitmapImage Class
- WriteableBitmap Class

(C) Copyright Nokia 2014. All rights reserved.