Fitting content into the view with Windows Phone maps API

Note: This article was a winner in the Windows Phone 8 Wiki Competition 2012Q4.

The full example for this code can be found from Windows Phone 8 Maps examples project, the code used here is implemented for example in SimpleContent example inside that project.

In general there are no direct ways on telling the map to zoom the map area in a way that specified content would be fully visible, however there are several ways which you could get this simple use case done, and luckily the coding is rather simple.

With polygon and polyline objects you would have the option on using Path defined for them. The path is `GeoCoordinateCollection` type object, and thus it is simply array of GeoCoordinates, which could be used with `LocationRectangle.CreateBoundingRectangle()` function to create a `LocationRectangle` instance, which then could be used to set the view with map like this:

```
LocationRectangle setRect = null;
setRect = LocationRectangle.CreateBoundingRectangle(myPolyGon.Path);
map1.SetView(setRect);
```

With Markers, which with WP8 are actually just MapOverlay objects inside MapLayer, we do not have the path or any other function which would give us the array of GeoCoorfinate's, so instead of having a simple function call, we need to implement simply for loop:

```
LocationRectangle setRect = null;
GeoCoordinate[] geoArr = new GeoCoordinate[markerLayer.Count()];
for (var p = 0; p < markerLayer.Count(); p++)
{
    geoArr[p] = markerLayer[p].GeoCoordinate;
}
setRect = LocationRectangle.CreateBoundingRectangle(geoArr);
map1.SetView(setRect);
```

And if you would want to do both of these object types with one way, you could basically construct the `LocationRectangle` with north, west, south and east arguments, and have a logic which loops through each `GeoCoorfinate` and checks the north, west, south and east bounds, and then uses these values for the `LocationRectangle`.

Here's example on how you could do it for `polyline`:

```
double north = 0;
double west = 0;
double south = 0;
double east = 0;

north = south = polyline.Path[0].Latitude;
west = east = polyline.Path[0].Longitude;

foreach (var p in polyline.Path.Skip(1))
{
    if (north < p.Latitude) north = p.Latitude;
    if (west > p.Longitude) west = p.Longitude;
    if (south > p.Latitude) south = p.Latitude;
    if (east < p.Longitude) east = p.Longitude;
}

setRect = new LocationRectangle(north, west, south, east);
```

```
map1.SetView(setRect);
```