Google Weather for Qt and Windows Phone

This article demonstrates how to create a Weather app using data from the Google Weather service in Qt and Windows Phone 7. It also illustrates how to display **GIF** images in Windows Phone 7.5.

20 Nov 2011

Introduction

The example app uses the Google Weather API to get (and display) the *Forecast Conditions* of a specified city over the next three days. City. User can enter a city name and will get the weather of current and next three days in an XML format. We parse the XML and display the content.

The implementation for Qt is straightforward. On WP7 its bit tricky as the icons in Google Weather API is of **GIF** format, and WP7 doesn't support **GIF** format "out of the box". So for this case WP7 we use ImageTools to display the **GIF** icons.

Google Weather Application for Qt

Google Weather Application for WP7

Implementation

Create an empty project for both Qt and WP7.

Qt Project (main.qml)

To take user input we create a TextField and a Button.

```
// text box
    TextField {
        id:textField
        width: 260
        x:20
        y:100
        text: "Enter City"
    }
    Button {
                id: button
                y:100
                x:parent.width -65
                text: "Go"
                onClicked: {
```


(C) Copyright Nokia 2014. All rights reserved.

```
                    feedModel.city = textField.text;
                    weatherInfo.text = textField.text + " weather forecast";
                }
            }
```

When user enters a valid city name in the TextField and click on the Button we take the text property of the TextField and pass as a parameter to the Google Weather API

```
http://www.google.com/ig/api?weather= <TextField.text>
```

This returns an XML data with the weather condition of the city.

Qt Project (RssModel.qml)

Once we get the XML data we need to parse the data and store it for further use. We use XmlListModel to store the data. Here property `source` is the Google Weather API and city is the city name entered in the TextField .

```
XmlListModel {
    id: feedModel
    property string city : ""
    property string imageConstPath: "http://www.google.com"
    source: "http://www.google.com/ig/api?weather=" +city
    query: "/xml_api_reply/weather/forecast_conditions"
    XmlRole { name: "day_of_week"; query: "day_of_week/@data/string()" }
    XmlRole { name: "low"; query: "low/@data/string()" }
    XmlRole { name: "high"; query: "high/@data/string()" }
    XmlRole { name: "icon"; query: "icon/@data/string()" }
    XmlRole { name: "condition"; query: "condition/@data/string()" }
```

Qt Project (Delegate.qml)

The **Delegate.qml** defines how the data will be displayed. We will follow a list structure to display the content.

```
Component {
    Item {
        id: wrapper; width: wrapper.ListView.view.width; height: 86
        Item {
            id: moveMe
           Rectangle {
                x: 5;
                y: 5;
                width: 40;
                height:40;
                color: "black";
                smooth: true
                Image {
                    source: feedModel.imageConstPath + icon;
                    x: 0;
                    y: 0;
                    height:40
                    width:40
                }
            }
            Column {
                x: 80; width: wrapper.ListView.view.width - 95; y: 0; spacing: 2
```


(C) Copyright Nokia 2014. All rights reserved.

```
                Text { text: day_of_week; color: "white"; width: parent.width;
font.pixelSize: 14; font.bold: true; elide: Text.ElideRight; style: Text.Raised;
styleColor: "black" }
                Text { text: "MinF : " + low; width: parent.width; font.pixelSize: 14;
elide: Text.ElideLeft; color: "white"; style: Text.Raised; styleColor: "black" }
                Text { text: "MaxF : " + high; width: parent.width; font.pixelSize: 14;
elide: Text.ElideRight; color: "white"; style: Text.Raised; styleColor: "black" }
                Text { text: condition; width: parent.width; font.pixelSize: 14; elide:
Text.ElideRight; color: "white"; style: Text.Raised; styleColor: "black" }
            }
        }
    }
}
```

Qt Project (main.qml)

Once the delegate is defined we can call the *Model* to parse the data and to display the data according to the *Delegate* in ListView.

```
//weather list ViewSection
    RssModel {id: feedModel}
    ListView {
     id: list
     y:200
     x:25
     width: parent.width ; height: parent.height
     model: feedModel
     delegate: Delegate {}
             }
```

Now the application is ready to run on Qt enabled device.

WP7 Project (MainPage.xaml)

As that of Qt we add a TextBox and a Button to take user input.

```
<TextBox Height="72" HorizontalAlignment="Left" Margin="6,15,0,0" Name="locationTextBox"
Text="Enter City" VerticalAlignment="Top" Width="342" />
            <Button Content="Go" Height="72" HorizontalAlignment="Left"
Margin="354,15,0,0" Name="button1" VerticalAlignment="Top" Width="83"
Click="button1_Click" />
```

Similar to Qt when user enters a valid city name in the TextBox and clicks on the Button, we take the text property of the TextBox and pass as a parameter in the Google Weather API

```
http://www.google.com/ig/api?weather= <TextBox.Text>
```

WP7 Project(MainPage.xaml.cs)

This returns an XML data, which we then parse and store it.

```
XElement XmlWeather = XElement.Parse(e.Result);

            foreach (var item in XmlWeather.Descendants("forecast_conditions"))
```


(C) Copyright Nokia 2014. All rights reserved.

```
            {
                WeatherInfo rssMain = new WeatherInfo();
                rssMain.day_of_week =
(string)item.Element("day_of_week").Attribute("data").Value;
                rssMain.high = (string)item.Element("high").Attribute("data").Value;
                rssMain.low = (string)item.Element("low").Attribute("data").Value;
                rssMain.condition =
(string)item.Element("condition").Attribute("data").Value;

                listBox1.Items.Add(rssMain);
            }
```

We use data binding to display the data in a ListBox.

```
     <ListBox Height="450" HorizontalAlignment="Left" Margin="12,141,0,0"
Name="listBox1" VerticalAlignment="Top" Width="425" >
                <ListBox.ItemTemplate>
                    <DataTemplate>
                        <StackPanel Orientation="Horizontal" Height="132" >
                            <Grid Height="100">
                                <Grid.ColumnDefinitions>
                                    <ColumnDefinition Width="100"></ColumnDefinition>
                                    <ColumnDefinition Width="*"></ColumnDefinition>
                                </Grid.ColumnDefinitions>




                                <TextBlock Text="{Binding Path=day_of_week}"
Foreground="Orange"  Grid.Column="1" Margin="10,10,10,60"></TextBlock>
                                <TextBlock Grid.Column="1" Margin="10,40,300,30"
Text="MinF: "></TextBlock>
                                <TextBlock Text="{Binding Path=low}" Grid.Column="1"
Margin="70,40,250,30" FontWeight="Bold"></TextBlock>
                                <TextBlock Grid.Column="1" Margin="150,40,160,30"
Text="MaxF: "></TextBlock>
                                <TextBlock Text="{Binding Path=high}" Grid.Column="1"
Margin="215,40,90,30" FontWeight="Bold"></TextBlock>
                                <TextBlock Text="{Binding Path=condition}"
Grid.Column="1" Margin="10,75,10,0"></TextBlock>

                            </Grid>
                        </StackPanel>
                    </DataTemplate>

                </ListBox.ItemTemplate>
            </ListBox>
```

Till now if we run the code this brings you the weather condition of the city except the thumbnail images. Let's see how to display the *GIF* thumbnail images and add it to the ListBox. We use ImageTools project from Codeplex to decode **GIF** image and to display it. First download the project and add the reference `ImageTools`, `ImageToolsControls` and `ImageTools.IO.Gif` to the project.

Now let's open the **MainPage.xaml** and add the ImageTools xaml, namespace references

(C) Copyright Nokia 2014. All rights reserved.

```
xmlns:imageTools="clr-namespace:ImageTools.Controls;assembly=ImageTools.Controls"
```

Add the resource

```
<phone:PhoneApplicationPage.Resources>
        <imageTools:ImageConverter x:Key="ImageConverter" />
    </phone:PhoneApplicationPage.Resources>
```

Add the image in the ListBox.

```
<imageTools:AnimatedImage Stretch="Fill" x:Name="gifImage"  Grid.Column="0" Height="100"
Width="100" Source="{Binding icon}" />
```

Now let's add code to decode the image after parsing

```
// gif conversion
ImageTools.IO.Decoders.AddDecoder<GifDecoder>();
ImageTools.ExtendedImage image = new ImageTools.ExtendedImage();
gifUrl = gifImageConstPath + (string)item.Element("icon").Attribute("data").Value;
image.UriSource = new Uri(gifUrl, UriKind.Absolute);
rssMain.icon = image;
```

At this time if we run the project it will display the weather condition of the city along with the **GIF** thumbnail images.

Source Code

- The full source code of Qt example is available here: File:GoogleWeatherQt.zip
- The full source code of WP7 example is available here: File:GoogleWeatherWP7.zip

(C) Copyright Nokia 2014. All rights reserved.