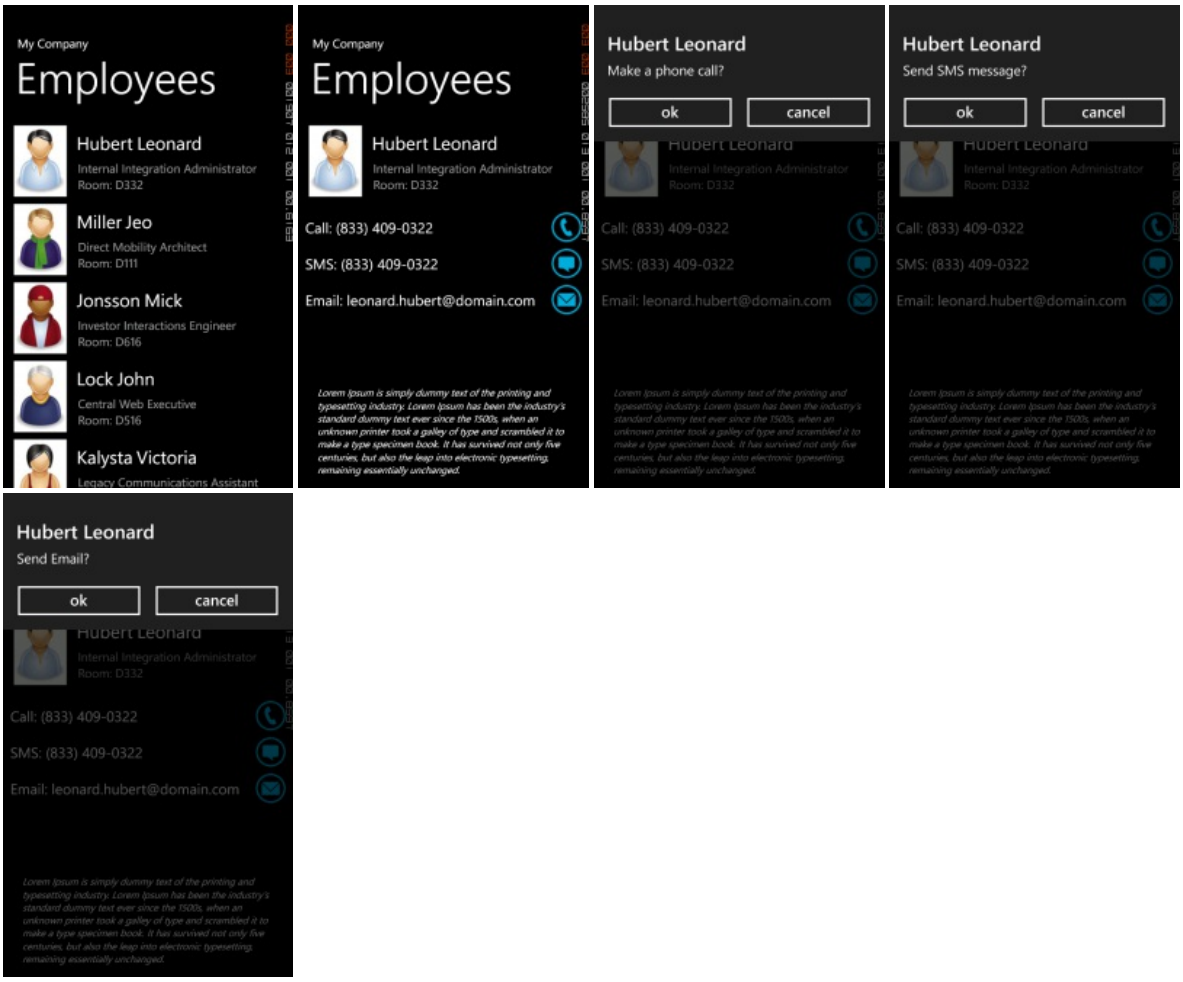
Employees app with XML parsing and messaging in WP7

This code example shows how to load, parse and display (employee) XML data hosted on a server in Windows Phone. The code example also shows how to send the employees an SMS or email, or make a phone call.

22 Jan 2012

Introduction

This article shows how to create a Windows Phone app which loads Employee information from server in XML format. Loaded Employees XML data will be parsed and displayed (image 1) in a `ListBox` Control. Selected Employee's details will be shown in a new Employee Page (image 2). You can call, send SMS or Email messages to selected Employee.



XML data on server

In this example Employees data will be loaded from XML file. Prepare XML data and images on server in following format:

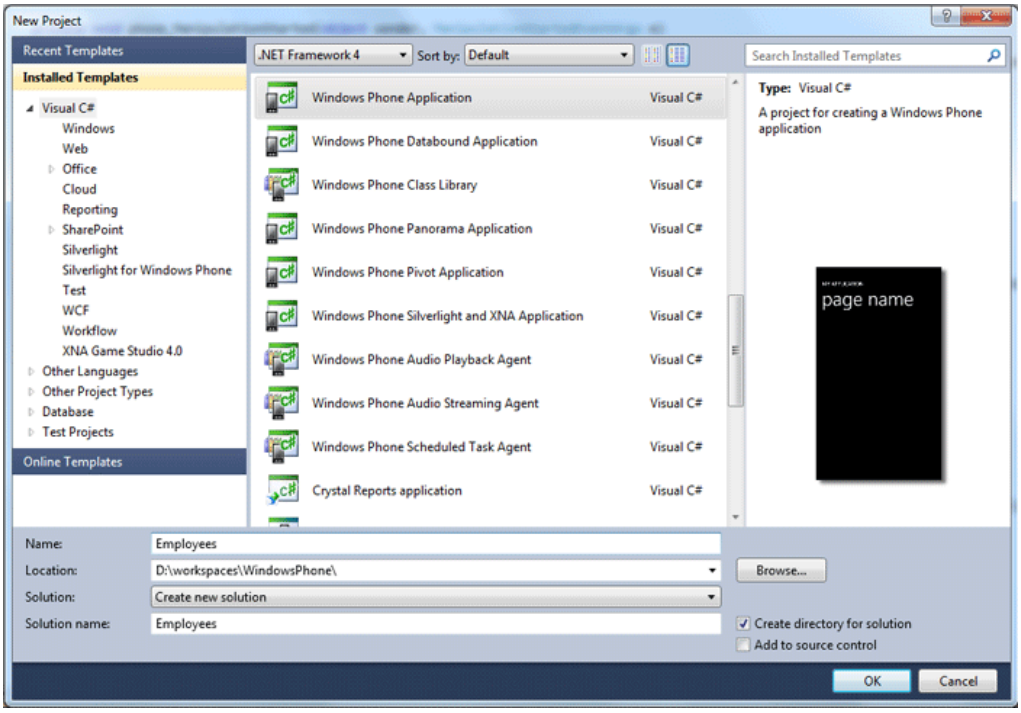
(C) Copyright Nokia 2013. All rights reserved.

```
− <root>
  − <employees>
    − <employee>
        <firstName>Leonard</firstName>
        <lastName>Hubert</lastName>
        <title>Internal Integration Administrator</title>
        <phone>(833) 409-0322</phone>
        <email>leonard.hubert@domain.com</email>
        <room>D332</room>
        <picture>employee01.png</picture>
      − <info>
          Lorem Ipsum is simply dummy text of the printing and
          typesetting industry. Lorem Ipsum has been the industry's
        </info>
      </employee>
    + <employee></employee>
    + <employee></employee>
    </employees>
  </root>
```

Of course you also will need some nice pictures of your Employees. Save those pictures to same folder with XML file.

Windows Phone Application

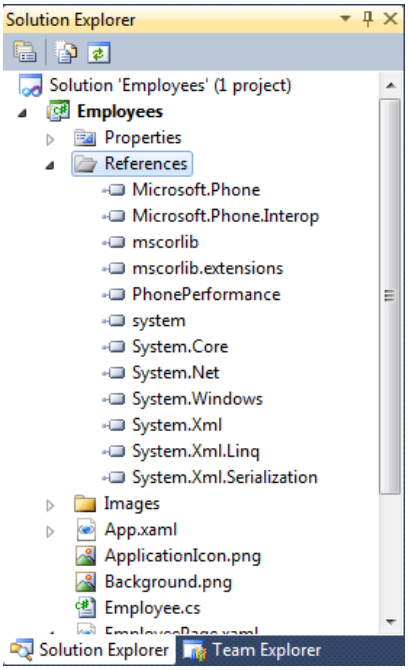
To start creating a new Windows Phone Application using the Windows Phone 7.1 SDK, start Microsoft Visual studio then create a new Project and select Windows Phone Application Template. In this example I have used C# as code behind language.



XML Parsing in Windows Phone 7

There are many different ways to load and parse XML data in Windows Phone 7. In this example I will use XML Deserialization to load XML document and pass the data to a deserializer. You will need to add references to `System.Xml.Serialization` and `System.Xml.Linq` to your project. Right click "References" in your project Solutions Explorer and select "Add new Reference..."

(C) Copyright Nokia 2013. All rights reserved.



Employees Class

Employees information will be load from XML data. All information will be stored to Employees and Employee Classes.

To create a `Employees` Class in to your project:

1. Right click your project in Solutions Explorer
2. Select Add and then Class...

Name: Employees.cs and click Add.

```
using System;
using System.Xml.Serialization;
using System.Collections.ObjectModel;

namespace Employees
{
    [XmlRoot("root")]
    public class Employees
    {
        [XmlArray("employees")]
        [XmlArrayItem("employee")]
        public ObservableCollection<Employee> Collection { get; set; }
    }
}
```

Like you can see in XML file, there is root element and here in `Employees` Class we use it same `root` element which contains all the Employees (you have to add using statement for `Serialization`). After that we have array of `employees` and it contains a `employee` items. All the `employee` items will be stored to an `ObservableCollection` (you have to add using statement for `ObjectModel`).

Employee Class

In XML data each employee had elements for firstname, lastname, title and so on. You have to define them with using with `[XMLElement]`. Remember add using statement for `Serialization`.

To create a `Employee` Class in to your project: 1. Right click your project in Solutions Explorer 2. Select Add and then Class...

Name: Employee.cs and click Add.

```
using System;
using System.Xml.Serialization;

namespace Employees
{
    public class Employee
    {
        [XmlElement("firstName")]
        public string firstName { get; set; }

        [XmlElement("lastName")]
        public string lastName { get; set; }

        [XmlElement("title")]
        public string title { get; set; }

        [XmlElement("phone")]
        public string phone { get; set; }

        [XmlElement("email")]
        public string email { get; set; }
```


(C) Copyright Nokia 2013. All rights reserved.

```
        [XmlElement("room")]
        public string room { get; set; }

        [XmlElement("picture")]
        public string picture { get; set; }

        [XmlElement("info")]
        public string info { get; set; }
    }
}
```

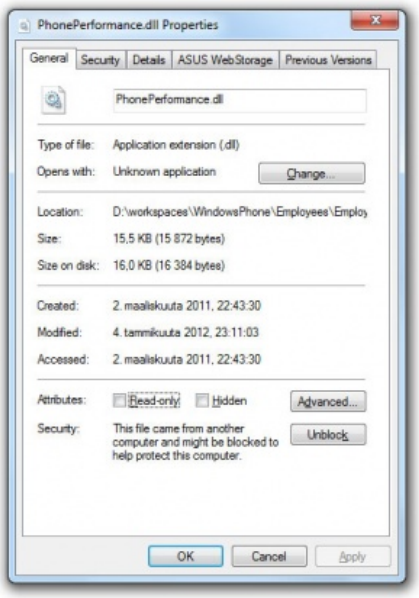
MainPage.xaml

All the employees will be shown in a ListBox in an Application Main Page.

Load a images

There can be a small issues when you try to load lot of images from the web at the same time in Windows Phone. You can find a few assemblies from the net to fix this. One good one is PhonePerformance assembly (you can download compiled PhonePerformance assembly with source codes here). You have to download and unzip it. Copy **PhonePerformance.dll** to your project and add a new reference to it with Solution Explorer.

You might get warning with to unblock **PhonePerformance.dll** assembly (because it is downloaded from web). Close your Visual Studio. Open Windows Exlorer and browse your **PhonePerformance.dll** file. Open file properties and Unblock.



Application and Page Title

Modify your application and page title to suit your purpose

```
<StackPanel x:Name="TitlePanel" Grid.Row="0" Margin="12,17,0,28">
 <TextBlock x:Name="ApplicationTitle" Text="My Company" Style="{StaticResource PhoneTextNormalStyle}"/>
 <TextBlock x:Name="PageTitle" Text="Employees" Margin="9,-7,0,0" Style="{StaticResource PhoneTextTitle1Style}"/>
</StackPanel>
```

Use ListBox to show Employees

(C) Copyright Nokia 2013. All rights reserved.

Each Employee is a one row in a ListBox. Each row is 130 px height. Image will take 100 px and employees name, title and room will be shown at the right side of image. Data Binding from the code to ListBox will be used (look more detailed information about Data Binding in later from MainPage.xaml.cs). PhonePerformance.dll is used to load images (look `Image` element: Image delay:LowProfileImageLoader.UriSource="{Binding picture}"). Remember add a new XML xmlns:delay="clr-namespace:Delay;assembly=PhonePerformance" `namespace` to your phone:PhoneApplicationPage element.

When user clicks a row in ListBox, a new Employee Page is displayed (it calls `employeesList_SelectionChanged` method). EmployeePage.xaml is described later in this article.

```
<phone:PhoneApplicationPage
    x:Class="Employees.MainPage"
    ....
    xmlns:delay="clr-namespace:Delay;assembly=PhonePerformance"
```

```
            <ListBox x:Name="employeesList"
                     SelectionChanged="employeesList_SelectionChanged">
                <ListBox.ItemTemplate>
                    <DataTemplate>
                        <Grid Height="130">
                            <Grid.ColumnDefinitions>
                                <ColumnDefinition Width="100"/>
                                <ColumnDefinition Width="*"/>
                            </Grid.ColumnDefinitions>
                            <Image delay:LowProfileImageLoader.UriSource="{Binding picture}"
                                   Grid.Column="0"
                                   Width="97"
                                   Height="125"
                                   VerticalAlignment="Center"
                                   HorizontalAlignment="Center"
                                   Margin="6"/>
                            <StackPanel Margin="10,15,0,0"
                                        Grid.Column="1"
                                        Height="60"
                                        Orientation="Horizontal"
                                        VerticalAlignment="Top">
                                <TextBlock Text="{Binding lastName}"
                                           FontSize="30" />
                                <TextBlock Text=" " />
                                <TextBlock Text="{Binding firstName}"
                                           FontSize="30"/>
                            </StackPanel>
                            <StackPanel Margin="0,50,0,0"
                                        Grid.Column="1"
                                        VerticalAlignment="Center">
                                <TextBlock Grid.Column="1"
                                       Text="{Binding title}"
                                       Style='{StaticResource PhoneTextSubtleStyle}'
                                       />
                                <TextBlock Grid.Column="1"
                                       Text="{Binding room, StringFormat='Room: \{0\}'}"
                                       Style='{StaticResource PhoneTextSubtleStyle}'
                                       />
                            </StackPanel>
                        </Grid>
                    </DataTemplate>
                </ListBox.ItemTemplate>
            </ListBox>
```

MainPage.xaml.cs

Code behind MainPage loads employees XML data from server, parses it to Employees and Employee Classes and finally Binds the data to UI.

Load XML data from Server

In MainPage constructor after InitializeComponent() is done, we first check if there is a network connection available and then start loading XML data

```
public MainPage()
        {
            InitializeComponent();
            // is there network connection available
            if (!System.Net.NetworkInformation.NetworkInterface.GetIsNetworkAvailable())
            {
                MessageBox.Show("No network connection available!");
                return;
            }
            // start loading XML-data
            WebClient downloader = new WebClient();
            Uri uri = new Uri("http://your.domain.url.here/employees.xml", UriKind.Absolute);
            downloader.DownloadStringCompleted += new DownloadStringCompletedEventHandler(EmployeesDownloaded);
            downloader.DownloadStringAsync(uri);
        }
```

All the XML data is loaded, it is time to parse it.

```
void EmployeesDownloaded(object sender, DownloadStringCompletedEventArgs e)
        {
            if (e.Result == null || e.Error != null)
            {
                MessageBox.Show("There was an error downloading the XML-file!");
            }
            else
            {
                // Deserialize if download succeeds
                XmlSerializer serializer = new XmlSerializer(typeof(Employees));
                XDocument document = XDocument.Parse(e.Result);
                // get all the employees
                Employees employees = (Employees) serializer.Deserialize(document.CreateReader());
                // bind data to ListBox
                employeesList.ItemsSource = employees.Collection;
```


(C) Copyright Nokia 2013. All rights reserved.

```
            }
        }
```

When user clicks a row in a ListBox, `employeesList_SelectionChanged`) method will be called. Here we first find our Application and store selected employee there. This is one way to share data between views in Windows Phone Applications. After that a new page will be displayed. This EmployeePage.xaml will be created later in this article.

```
// selection in EmployeeList is changed
private void employeesList_SelectionChanged(object sender, SelectionChangedEventArgs e)
{
            var app = App.Current as App;
            app.selectedEmployee = (Employee) employeesList.SelectedItem;
            this.NavigationService.Navigate(new Uri("/EmployeePage.xaml", UriKind.Relative));
}
```

Now all the Employees should be visible in MainPage.

-Remember to add the following two namespaces in mainpage.xaml.cs

```
using System.Xml.Serialization;
using System.Xml.Linq;
```

App.xaml.cs

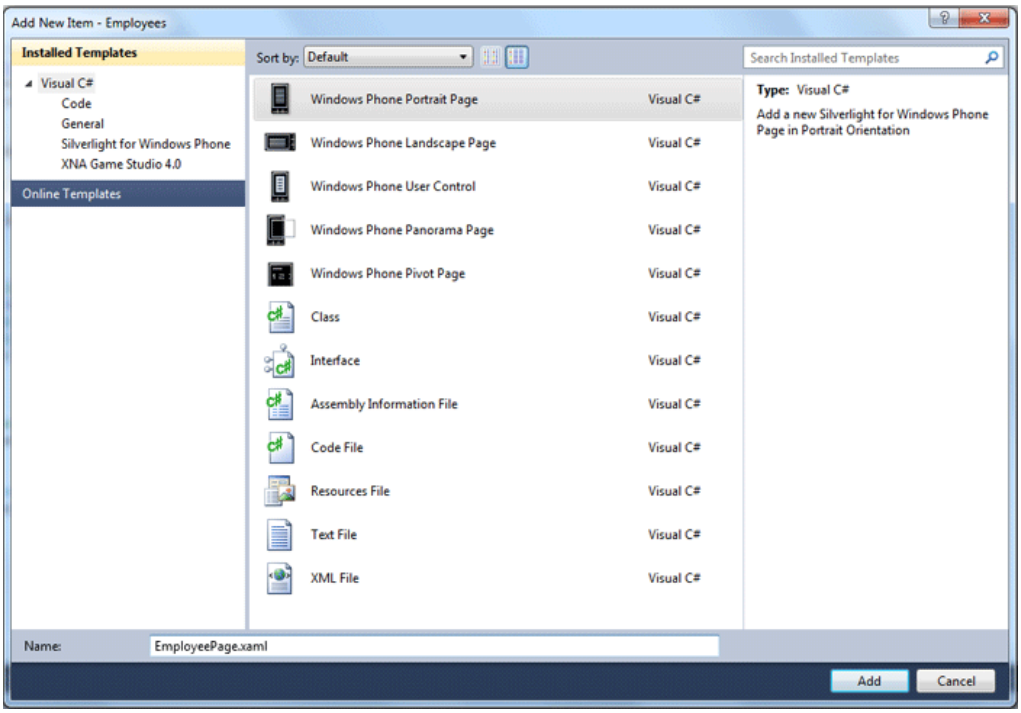
In previous chapter we stored selected employee to App Class. So we need to change App.xaml.cs file a little (add selectedEmployee):

```
        // selected employee from EmployeeList
        public Employee selectedEmployee { get; set; }
```

EmployeePage.xaml

Create a new Page to your Project

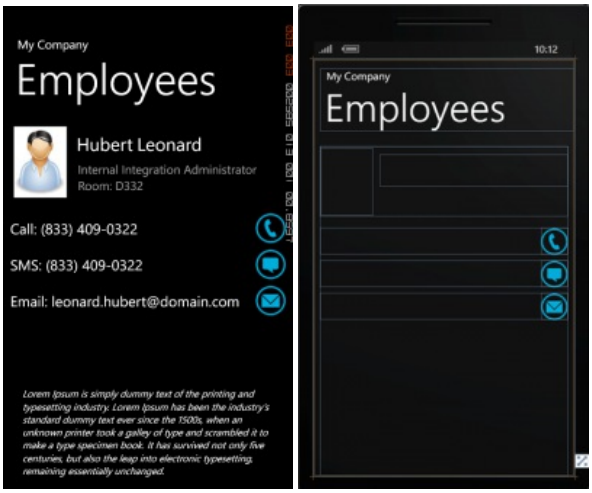
Employee's detailed information is shown a new Employee Page. To create a new Page to your project, right click your project in Solution Exloperer and select Add, New Item... Select Windows Phone Portrait Page and name it to EmployeePage.xaml



Employee Page Design

In right you can see Employee Detail Page in Design Mode.

(C) Copyright Nokia 2013. All rights reserved.



Company name and Page title is shown as they are in Main Page. You can modify those to your own needs.

```
<StackPanel x:Name="TitlePanel" Grid.Row="0" Margin="12,17,0,28">
            <TextBlock x:Name="ApplicationTitle" Text="My Company" Style="{StaticResource PhoneTextNormalStyle}"/>
            <TextBlock x:Name="PageTitle" Text="Employees" Margin="9,-7,0,0" Style="{StaticResource PhoneTextTitle1Style}"/>
        </StackPanel>
```

Next there are employee's image, last and first name and title and room. `StackPanel` is used to organize controls.

```
<Grid Height="130" VerticalAlignment="Top">
                <Grid.ColumnDefinitions>
                    <ColumnDefinition Width="100"/>
                    <ColumnDefinition Width="*"/>
                </Grid.ColumnDefinitions>
                <Image x:Name="image"
                                  Grid.Column="0"
                                  Width="97"
                                  Height="125"
                                  VerticalAlignment="Center"
                                  HorizontalAlignment="Center"
                                  Margin="6"/>
                <StackPanel Margin="10,15,0,0"
                                       Grid.Column="1"
                                       Height="60"
                                       Orientation="Horizontal"
                                       VerticalAlignment="Top">
                    <TextBlock x:Name="lastName" FontSize="30" />
                    <TextBlock Text=" " />
                    <TextBlock x:Name="firstName" FontSize="30"/>
                </StackPanel>
                <StackPanel Margin="0,50,0,0"
                            Grid.Column="1"
                            VerticalAlignment="Center">
                    <TextBlock Grid.Column="1"
                               x:Name="title"
                               Style='{StaticResource PhoneTextSubtleStyle}'
                               />
                    <TextBlock Grid.Column="1"
                               x:Name="room"
                               Style='{StaticResource PhoneTextSubtleStyle}'
                               />
                </StackPanel>
            </Grid>
```

User can call or send SMS and Email messages to selected employee by pressing visible phone numbers. When end user is pressing the numbers, `ManipulationStarted` will be launched. Event handling will be programmend in to EmployeePage.xaml.cs file.

```
<Grid Height="50" VerticalAlignment="Top" Margin="0,150"
                  ManipulationStarted="phone_ManipulationStarted">
                <Grid.ColumnDefinitions>
                    <ColumnDefinition Width="*"/>
                </Grid.ColumnDefinitions>
                <TextBlock x:Name="phone" VerticalAlignment="Center" FontSize="24"/>
                <Image Source="Images/call.png" HorizontalAlignment="Right"/>
            </Grid>

            <Grid Height="50" VerticalAlignment="Top" Margin="0,210"
                  ManipulationStarted="sms_ManipulationStarted">
                <Grid.ColumnDefinitions>
                    <ColumnDefinition Width="*"/>
                </Grid.ColumnDefinitions>
                <TextBlock x:Name="sms" VerticalAlignment="Center" FontSize="24"/>
                <Image Source="Images/msg.png" HorizontalAlignment="Right"/>
            </Grid>

            <Grid Height="50" VerticalAlignment="Top" Margin="0,270"
                  ManipulationStarted="mail_ManipulationStarted">
                <Grid.ColumnDefinitions>
                    <ColumnDefinition Width="*"/>
                </Grid.ColumnDefinitions>
                <TextBlock x:Name="mail" VerticalAlignment="Center" FontSize="24"/>
                <Image Source="Images/mail.png" HorizontalAlignment="Right"/>
            </Grid>
```

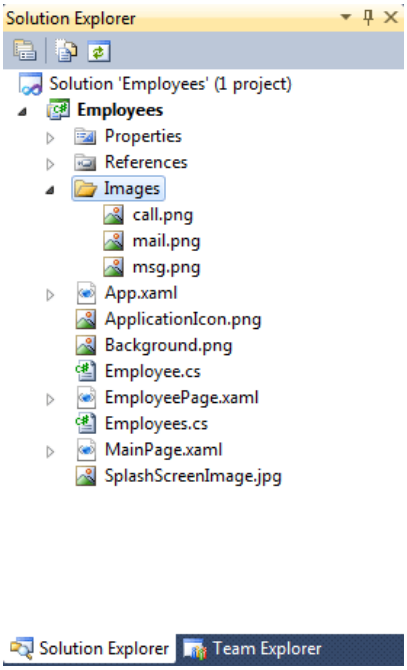
Detailed information of selected employee is displayed at the bottom of the screen.

```
<TextBlock x:Name="info" TextWrapping="Wrap"
                      FontSize="16" FontStyle="Italic"
                      VerticalAlignment="Bottom"
```


(C) Copyright Nokia 2013. All rights reserved.

```
                        Margin="20,20,20,20"/>
```

In this example I have used a few images to display Call, SMS and Email messages. You can add a new images to your project also. First create a new Folder for images to your Solution Explorer (name it to Images for example). Copy your images to that folder in Windows. Right click your Images folder in Solution Explorer and select Add, Existing Item... Browse your images and add to project.



EmployeePage.xaml.cs

Code behind Employee Page first gets the selected employee from App class and then displays employee's informations to UI.

```
// selected Employee
Employee employee;

public EmployeePage()
        {
            InitializeComponent();

            // get selected employee from App Class
            var app = App.Current as App;
            employee = app.selectedEmployee;

            // show employee details in page
            lastName.Text = employee.lastName;
            firstName.Text = employee.firstName;
            title.Text = employee.title;
            room.Text = "Room: " + employee.room;
            image.Source = new BitmapImage(new Uri(employee.picture, UriKind.RelativeOrAbsolute));
            mail.Text = "Email: " + employee.email;
            sms.Text = "SMS: " + employee.phone;
            phone.Text = "Call: " + employee.phone;
            info.Text = employee.info;
        }
```

When user want's to call employee, `phone_ManipulationStarted` method will be called. (Remember add `using Microsoft.Phone.Tasks;` in your code)

```
        private void phone_ManipulationStarted(object sender, ManipulationStartedEventArgs e)
        {
            MessageBoxResult result = MessageBox.Show("Make a phone call?", employee.lastName + " " + employee.firstName, MessageBox
            if (result == MessageBoxResult.OK)
            {
                // make a phone call
                PhoneCallTask phoneTask = new PhoneCallTask();
                phoneTask.DisplayName = employee.lastName + " " + employee.firstName;
                phoneTask.PhoneNumber = employee.phone;
                phoneTask.Show();
            }
        }
```

When user want's to Send SMS message to employee, `sms_ManipulationStarted` method will be called.

```
        private void sms_ManipulationStarted(object sender, ManipulationStartedEventArgs e)
        {
            MessageBoxResult result = MessageBox.Show("Send SMS message?", employee.lastName + " " + employee.firstName, MessageBoxB
            if (result == MessageBoxResult.OK)
            {
                // sms
                SmsComposeTask composeSMS = new SmsComposeTask();
                composeSMS.Body = "Edit your message";
                composeSMS.To = employee.phone;
                composeSMS.Show();
            }
        }
```


(C) Copyright Nokia 2013. All rights reserved.

When user want's to Send Email to employee, `mail_ManipulationStarted` method will be called.

```
        private void mail_ManipulationStarted(object sender, ManipulationStartedEventArgs e)
        {
            MessageBoxResult result = MessageBox.Show("Send Email?", employee.lastName + " " + employee.firstName, MessageBoxButton.
            if (result == MessageBoxResult.OK)
            {
                // sähköposti
                EmailComposeTask emailcomposer = new EmailComposeTask();
                emailcomposer.To = "<a href=\"mailto:" + employee.email + "\">" + employee.email + "</a>";
                emailcomposer.Subject = "Edit your Subject";
                emailcomposer.Body = "Edit your Message";
                emailcomposer.Show();
            }
        }
```

Summary

Working with XML data in Windows Phone Applications is quite a easy. Hope you find this article usefull and it helps you work with XML in WP7. And finally there are sources codes for this article: File:PTM Employees.zip

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.