|  |
| --- |
| Microsoft SharePoint 2013 - Hands-on Lab |
| Mobile |
| Verified Against Build #15.0.4420.1017 |

|  |
| --- |
| Microsoft  Version 1.0  August 14, 2012 |

Information in this document, including URL and other Internet Web site references, is subject to change without notice. Unless otherwise noted, the example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted herein are fictitious, and no association with any real company, organization, product, domain name, e-mail address, logo, person, place, or event is intended or should be inferred. Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

The names of manufacturers, products, or URLs are provided for informational purposes only and Microsoft makes no representations and warranties, either expressed, implied, or statutory, regarding these manufacturers or the use of the products with any Microsoft technologies. The inclusion of a manufacturer or product does not imply endorsement of Microsoft of the manufacturer or product. Links are provided to third party sites. Such sites are not under the control of Microsoft and Microsoft is not responsible for the contents of any linked site or any link contained in a linked site, or any changes or updates to such sites. Microsoft is not responsible for webcasting or any other form of transmission received from any linked site. Microsoft is providing these links to you only as a convenience, and the inclusion of any link does not imply endorsement of Microsoft of the site or the products contained therein.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

Copyright © 2012 Microsoft Corporation. All rights reserved.

Microsoft and SharePoint are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

# Introduction

## Estimated time to complete this lab

60 minutes

## Objectives

After completing this lab, you will be able to:

* Create Windows Phone 7 applications against SharePoint lists
* Leverage push notifications with a Windows Phone 7 application
* Create a Windows Phone 7 application that provides mapping functionality

## Overview of Lab

In this lab, you will create Windows Phone 7 applications against SharePoint lists.

## Virtual Machine Technology

The computers in this lab are virtual machines that are implemented using Microsoft Hyper-V. Before starting each virtual machine, ensure you apply the **Start-Lab** snapshot. When you have started a virtual machine, log on by pressing **CTRL+ALT+END** and supply the credentials listed in the lab instructions.

## Computers in this lab

This lab uses virtual machines as described in the following table. Before you begin the lab, you must start the virtual machines and then log on to the computers.

|  |  |
| --- | --- |
| **Virtual Machine** | **Role** |
| {Supplied by Instructor} | Domain Controller |
| {Supplied by Instructor} | Actual SharePoint environment with Office client and other required software. |
| {Supplied by Instructor} | Windows 7 with Visual Studio 2010 and phone tools |

All user accounts in this lab use the password {Supplied by Instructor}.

## Prerequisites

This lab requires that you have a SharePoint Web Application created that supports Forms-Based Authentication. Check with your instructor to determine if such a Web Application is available in the lab environment.

# Exercise 1: Creating a Simple List-Based Phone App

In this exercise, you will create a Windows Phone 7 app that uses an External list for a data source.

## Task 1 - Upload BDC Metadata Model

In this task, you will upload a BDC Metadata Model to the BDC Service Application.

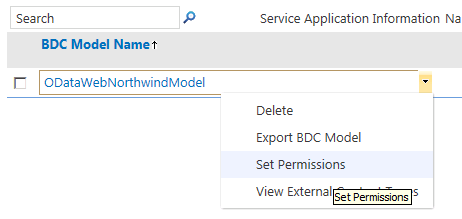
* You will need the file Northwind\_oData.bdcm

1. Open **Central Administration**.
2. Select **Application Management⮚Manage Service Applications**.
3. On the Service Application page, click **Business Data Connectivity Service**.
4. On the Ribbon in the BDC Service Application, click **Import**.
5. On the Import BDC Model page, click **Browse**.
6. In the **Choose a File to Upload** dialog, browse to the **Northwind\_oData.bdcm** file and click **Open**.
7. Click **Import**.
8. After the file is imported, click **OK**.

## Task 2 – Grant Permissions

In this task, you will grant permissions for users to execute the methods of the BDC Metadata Model.

1. In the list of BDC Models, hover over **ODataWebNorthwindModel** and select **Set Permissions**.



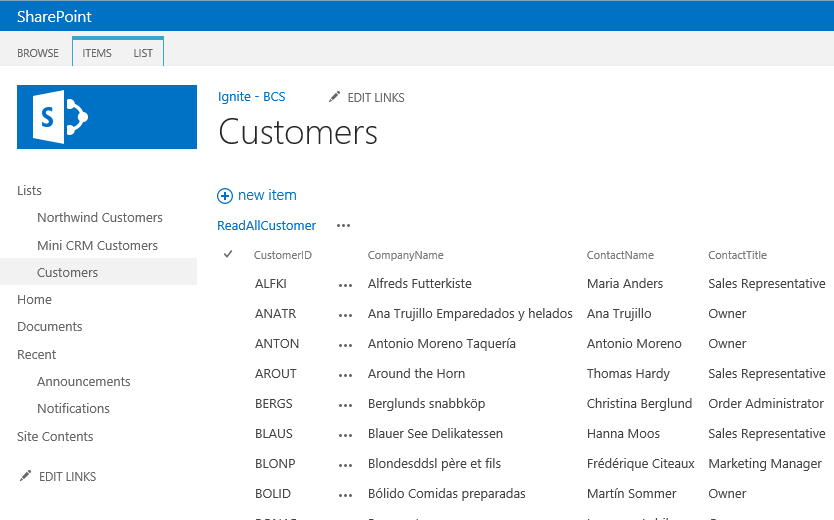
* + - * + Access the Permissions dialog

1. In the **Set Object Permissions** dialog, click the **Browse** button.
2. In the **Select People and Groups** dialog, search for your current account.
3. Click **Add** and **OK**.
4. In the **Set Object Permissions** dialog, click **Add**
5. Check the permissions for **Edit**, **Execute, Selectable In Clients**, and **Set Permissions**.
6. Click **OK**.
7. In the Ribbon, select **External Content Types** from the drop-down list in the View group.
8. In the list of External Content Types, hover over **Customer** and select **Set Permissions**.
9. In the **Set Object Permissions** dialog, click the **Browse** button.
10. In the **Select People and Groups** dialog, search for your current account.
11. Click **Add** and **OK**.
12. In the **Set Object Permissions** dialog, click **Add**
13. Check the permissions for **Edit**, **Execute, Selectable In Clients**, and **Set Permissions**.
14. Ensure the **Propagate Permissions** box is checked.
15. Click **OK**.

## Task 3 – Create an External List

In this task, you will create an External List based on the oData Source.

1. **Navigate** to a SharePoint site where you can create new lists.
2. On the home page of the site, click **More**.
3. On the Apps page, click **Add an App**.
4. On the Add an App page, hover over **External List** and click **Add It**.
5. In the Adding an External List dialog, enter **Customers** in the **Name** field.
6. Click **Select External Content Type**.
7. In the External Content Type Picker, select the source **http://services.odata.org/Northwind/Northwind.svc**.
8. Click **OK**.
9. Click **Create**.
10. On the Apps page, click **Customers** to view the list.

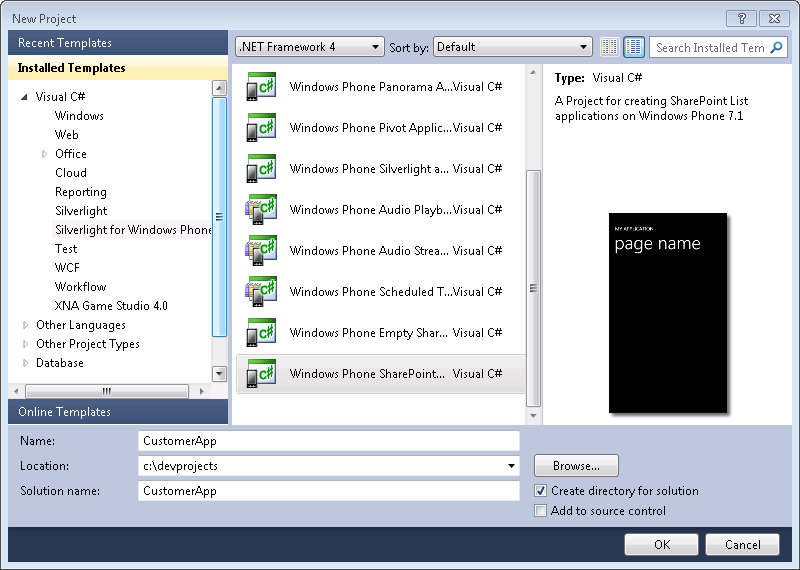


* + - * + External List based on Northwind oData Source

## Task 4 – Create the Phone App

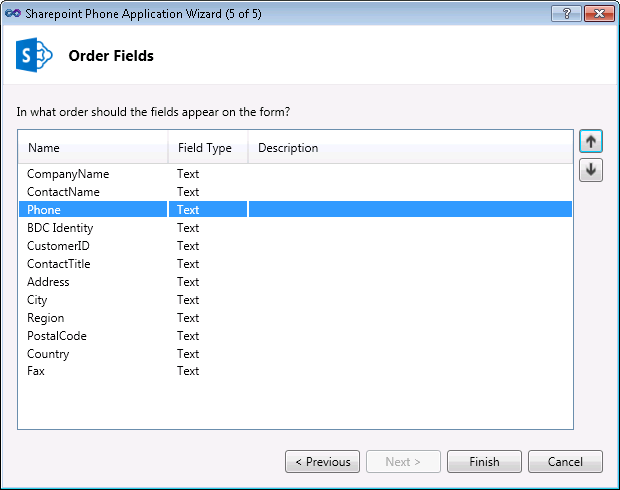
In this task, you will create a simple Windows Phone 7 app against the External List.

1. **Log In** to the Phone Development environment.
2. **Start** Visual Studio 2010.
3. Select **File⮚New⮚Project** from the main menu.
4. In the New Project dialog, select **Visual C#⮚Silverlight for Windows Phone⮚Windows Phone SharePoint List Application**.
5. Name the project **CustomerApp**.
6. Click **OK**.



* + - * + Start new Phone Application project

1. In the **SharePoint Phone Application Wizard**, enter the URL of the Phone Development Site Collection you created earlier (e. g., http://contososerver:8888) and click **Find Lists**.
2. When prompted, log in using **Forms-Based Authentication**.
3. Select the **Customers** list and click **Next**.
4. On the **Choose Views** screen, check **ReadAllCustomer** and click **Next**.
5. On the **Choose Operations** screen, check **Display** and click **Next**.
6. On the **Choose Fields** screen, check all of the fields and click **Next**.
7. On the **Order Fields** screen, move the CompanyName, ContactName, and Phone fields to the top of the display list as shown and click **Finish**.



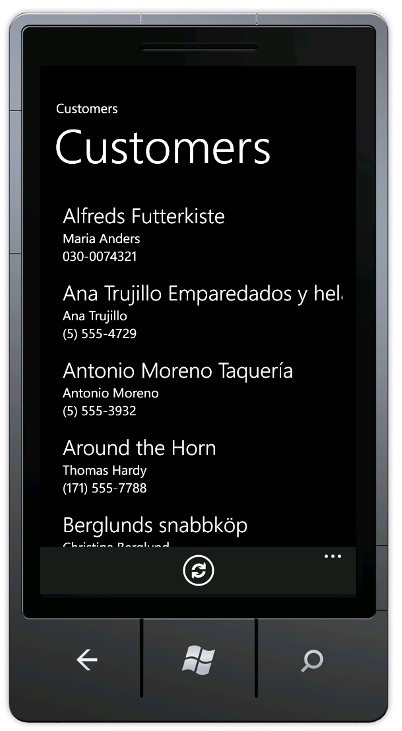
* + - * + Reorder fields

1. In the **List.xaml** file, change the header from **ReadAllCustomer** to **Customers**.
2. Open the **ListDataProvider.cs** file and change the **Context** property to the following:
   1. public override ClientContext Context
   2. {
   3. get
   4. {
   5. if (m\_Context != null)
   6. return m\_Context;
   7. m\_Context = new ClientContext(SiteUrl);
   8. Authenticator at = new Authenticator();
   9. at.AuthenticationMode = ClientAuthenticationMode.FormsAuthentication;
   10. at.PromptOnFailure = true;
   11. m\_Context.Credentials = at;
   12. return m\_Context;
   13. }
   14. }

## Task 5 – Start the App

In this task, you will run the phone app in the emulator.

1. In Visual Studio, select **Debug⮚Start Debugging**.
2. When prompted, log in using Forms-Based Authentication.
3. Scroll through the resulting Customers list.



* + - * + Completed App

# Exercise 2: Push Notifications

In this exercise, you will create a Windows Phone 7 app that receives Push Notifications from SharePoint when items in a list change.

## Task 1 - Install Push Notification Support

In this task, you will install a custom feature to support Push Notifications.

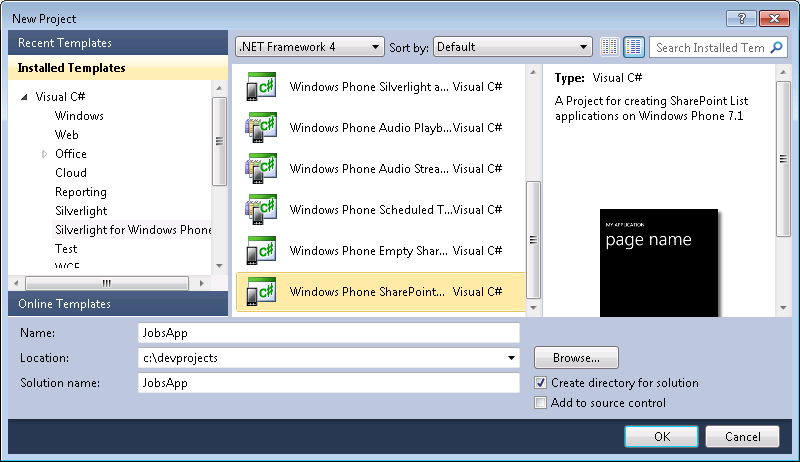
* You will need the file PushNotificationList.wsp

1. **Add** the solution package **PushNotificationList.wsp** to the farm using the following PowerShell command.
   1. Add-SPSolution -LiteralPath [Path to Package]\PushNotificationsList.wsp
2. Open **Central Administration**.
3. On the home page of Central Administration, click **System Settings**.
4. Click **Manage Farm Solutions**.
5. Click **PushNotificationsList.wsp**.
6. Click **Deploy Solution**, then click **OK**.
7. **Wait** until the solution is deployed.
8. **Navigate** to the site collection you are using for phone development.
9. Select **Site⮚Site Settings**.
10. Click **Manage Site Features**.
11. Activate the Feature **Push Notification Support**.
12. Note that the **Push Notifications** feature was automatically activated as well.
13. Select **Site⮚View All Site Content**.
14. Click the **Jobs** list that was just created by the Push Notification Support feature.

## Task 2 - Create the Phone App

In this task, you will build a phone app to receive Push Notifications based on changes to the Jobs list.

1. **Start** Visual Studio 2010.
2. Select **File⮚New⮚Project** from the main menu.
3. In the New Project dialog, select **Visual C#⮚Silverlight for Windows Phone⮚Windows Phone SharePoint List Application**.
4. Name the project **JobsApp**.
5. Click **OK**.



* + - * + Start new Phone Application project

1. In the **SharePoint Phone Application Wizard**, enter the URL of the Phone Development Site Collection you created earlier (e. g., http://contososerver:8888) and click **Find Lists**.
2. When prompted, log in using **Forms-Based Authentication**.
3. Select the **Jobs** list and click **Next**.
4. On the **Choose Views** screen, check **All Items** and click **Next**.
5. On the **Choose Operations** screen, check **New**, **Display, Edit,** and **Delete.**
6. Click **Next**.
7. On the **Choose Fields** screen, check all of the fields and click **Next**.
8. On the **Order Fields** screen, click **Finish**.
9. Edit the **ListDataProvider.cs** file and change the **Context** property to the following:
   1. public override ClientContext Context
   2. {
   3. get
   4. {
   5. if (m\_Context != null)
   6. return m\_Context;
   7. m\_Context = new ClientContext(SiteUrl);
   8. Authenticator at = new Authenticator();
   9. at.AuthenticationMode = ClientAuthenticationMode.FormsAuthentication;
   10. at.PromptOnFailure = true;
   11. m\_Context.Credentials = at;
   12. return m\_Context;
   13. }
   14. }

## Task 3 - Build the Subscription Support

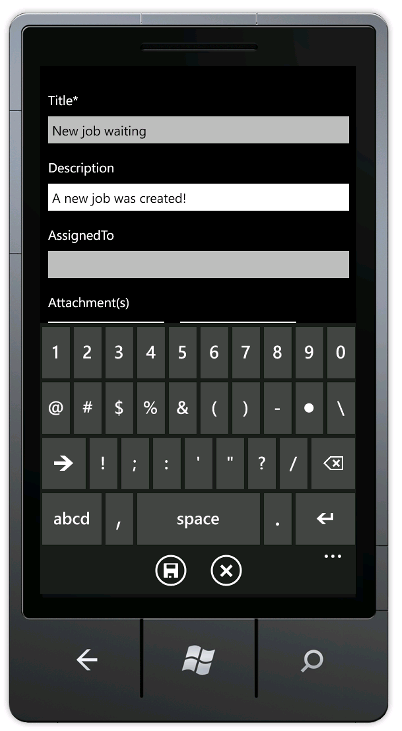
In this task, you will add subscription support to the phone app.

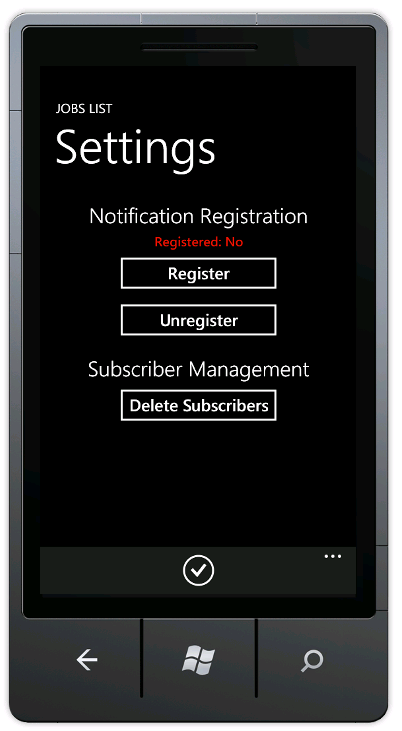
* You will need the file Notifications.cs

1. Right click the **JobsApp** project node in the Solution Explorer and select **Add⮚Existing Item** from the context menu.
   1. **Add** the **Notifications.cs** class to the project.
2. Right click the **JobsApp** project node in the Solution Explorer and select **Add⮚New Item** from the context menu.
   1. Select **Windows Phone Portrait Page**.
   2. Name the file **Settings.xaml** and click **Add**.
   3. **Replace** the file with the following code.
   4. <phone:PhoneApplicationPage
   5. x:Class="JobsApp.Settings"
   6. xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
   7. xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
   8. xmlns:phone="clr-namespace:Microsoft.Phone.Controls;assembly=Microsoft.Phone"
   9. xmlns:shell="clr-namespace:Microsoft.Phone.Shell;assembly=Microsoft.Phone"
   10. xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
   11. xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
   12. FontFamily="{StaticResource PhoneFontFamilyNormal}"
   13. FontSize="{StaticResource PhoneFontSizeNormal}"
   14. Foreground="{StaticResource PhoneForegroundBrush}"
   15. SupportedOrientations="Portrait" Orientation="Portrait"
   16. mc:Ignorable="d" d:DesignHeight="768" d:DesignWidth="480"
   17. shell:SystemTray.IsVisible="True">
   18. <Grid x:Name="LayoutRoot" Background="Transparent">
   19. <Grid.RowDefinitions>
   20. <RowDefinition Height="Auto"/>
   21. <RowDefinition Height="\*"/>
   22. </Grid.RowDefinitions>
   23. <!--TitlePanel contains the name of the application and page title-->
   24. <StackPanel x:Name="TitlePanel" Grid.Row="0" Margin="12,17,0,28">
   25. <TextBlock x:Name="ApplicationTitle" Text="JOBS LIST" Style="{StaticResource PhoneTextNormalStyle}"/>
   26. <TextBlock x:Name="PageTitle" Text="Settings" Margin="9,-7,0,0" Style="{StaticResource PhoneTextTitle1Style}"/>
   27. </StackPanel>
   28. <!--ContentPanel - place additional content here-->
   29. <Grid x:Name="ContentPanel" Grid.Row="1" Margin="12,0,12,0">
   30. <StackPanel Margin="0,5,0,5">
   31. <StackPanel Orientation="Vertical" Margin="0,5,0,5">
   32. <TextBlock TextWrapping="Wrap" HorizontalAlignment="Center" Style="{StaticResource PhoneTextTitle2Style}">Notification Registration</TextBlock>
   33. <StackPanel Orientation="Vertical" Margin="0,5,0,5">
   34. <TextBlock x:Name="txtRegistrationStatus" TextWrapping="Wrap" HorizontalAlignment="Center" Text="Registered: No" Style="{StaticResource PhoneTextAccentStyle}" Foreground="{StaticResource PhoneAccentBrush}" />
   35. <Button x:Name="btnRegister" Content="Register" Height="71" Width="260" Click="OnRegisterButtonClick" />
   36. <Button x:Name="btnUnregister" Content="Unregister" Height="71" Width="260" Click="OnUnregisterButtonClick" />
   37. </StackPanel>
   38. </StackPanel>
   39. <StackPanel Orientation="Vertical" Margin="0,5,0,5">
   40. <TextBlock TextWrapping="Wrap" HorizontalAlignment="Center" Style="{StaticResource PhoneTextTitle2Style}">Subscriber Management</TextBlock>
   41. <Button x:Name="btnDeleteSubscribers" Content="Delete Subscribers" Height="71" Width="260" Click="OnDeleteSubscribersButtonClick" />
   42. </StackPanel>
   43. </StackPanel>
   44. </Grid>
   45. </Grid>
   46. <!--Sample code showing usage of ApplicationBar-->
   47. <phone:PhoneApplicationPage.ApplicationBar>
   48. <shell:ApplicationBar IsVisible="True" IsMenuEnabled="False">
   49. <shell:ApplicationBarIconButton x:Name="btnOK" IconUri="/Images/appbar.check.rest.png" Text="OK" Click="OnOKButtonClick" />
   50. </shell:ApplicationBar>
   51. </phone:PhoneApplicationPage.ApplicationBar>
   52. </phone:PhoneApplicationPage>
3. Open **Settings.xaml.cs** for editing.
   1. **Replace** the contents of the file with the following code.
   2. using System;
   3. using System.Collections.Generic;
   4. using System.Linq;
   5. using System.Net;
   6. using System.Windows;
   7. using Microsoft.Phone.Controls;
   8. using Microsoft.SharePoint.Client;
   9. namespace JobsApp
   10. {
   11. public partial class Settings : PhoneApplicationPage
   12. {
   13. private const string RegisteredYesText = "Registered: Yes";
   14. private const string RegisteredNoText = "Registered: No";
   15. public Settings()
   16. {
   17. InitializeComponent();
   18. }
   19. protected override void OnNavigatedTo(System.Windows.Navigation.NavigationEventArgs e)
   20. {
   21. this.txtRegistrationStatus.Text = (Notifications.GetRegistrationStatus()) ? RegisteredYesText : RegisteredNoText;
   22. }
   23. private void OnOKButtonClick(object sender, EventArgs e)
   24. {
   25. NavigationService.Navigate(new Uri("/Views/List.xaml", UriKind.Relative));
   26. }
   27. private void OnRegisterButtonClick(object sender, RoutedEventArgs e)
   28. {
   29. Notifications.OpenNotificationChannel(true);
   30. // Navigating back to List form. User will be notified when process is complete.
   31. NavigationService.Navigate(new Uri("/Views/List.xaml", UriKind.Relative));
   32. }
   33. private void OnUnregisterButtonClick(object sender, RoutedEventArgs e)
   34. {
   35. Notifications.UnSubscribe();
   36. // Navigating back to List form. User will be notified when process is complete.
   37. NavigationService.Navigate(new Uri("/Views/List.xaml", UriKind.Relative));
   38. }
   39. private void OnDeleteSubscribersButtonClick(object sender, RoutedEventArgs e)
   40. {
   41. Notifications.ClearSubscriptionStore();
   42. // Navigating back to List form. User will be notified when process is complete.
   43. NavigationService.Navigate(new Uri("/Views/List.xaml", UriKind.Relative));
   44. }
   45. }
   46. }
4. Right click the **Images** folder and select **Add⮚Existing File** from the context menu.
   1. Navigate to **%PROGRAMFILES%\Microsoft SDKs\Windows Phone\v7.1\Icons\dark**
   2. Select the file **appbar.check.rest.png** and click **Add**.
   3. Click the file **appbar.check.rest.png** in the Solution Explorer.
   4. In the **Properties** window, set **Build Action** to **Content**.
   5. Set **Copy to Output Directory** to **Copy if Newer**.
5. Open **List.xaml** for editing.
   1. Add the following markup directly beneath the btnRefresh button declaration.
   2. <shell:ApplicationBarIconButton x:Name="btnSettings" IconUri="/Images/appbar.feature.settings.rest.png" Text="Settings" IsEnabled="True" Click="OnSettingsButtonClick" />
6. Open **List.xaml.cs** for editing.
   1. **Add** the following code to the class.
   2. private void OnSettingsButtonClick(object sender, EventArgs e)
   3. {
   4. NavigationService.Navigate(new Uri("/Settings.xaml", UriKind.Relative));
   5. }
   6. **Replace** the **OnViewModelInitialization** method with the following code.
   7. private void OnViewModelInitialization(object sender, InitializationCompletedEventArgs e)
   8. {
   9. this.Dispatcher.BeginInvoke(() =>
   10. {
   11. //If initialization has failed, show error message and return
   12. if (e.Error != null)
   13. {
   14. MessageBox.Show(e.Error.Message, e.Error.GetType().Name, MessageBoxButton.OK);
   15. return;
   16. }
   17. App.MainViewModel.LoadData(((PivotItem)Views.SelectedItem).Name);
   18. this.DataContext = (sender as ListViewModel);
   19. });
   20. // Open notification channel here if user has chosen to subscribe to notifications.
   21. if (Notifications.GetRegistrationStatus() == true)
   22. Notifications.OpenNotificationChannel(false);
   23. }
7. Open **App.xaml.cs** for editing.
8. **Replace** the **Application\_launching** method with the following code.
   1. private void Application\_Launching(object sender, LaunchingEventArgs e)
   2. {
   3. // Get set up for notifications.
   4. Notifications.Context = App.DataProvider.Context;
   5. Notifications.SaveDeviceAppIdToStorage();
   6. }

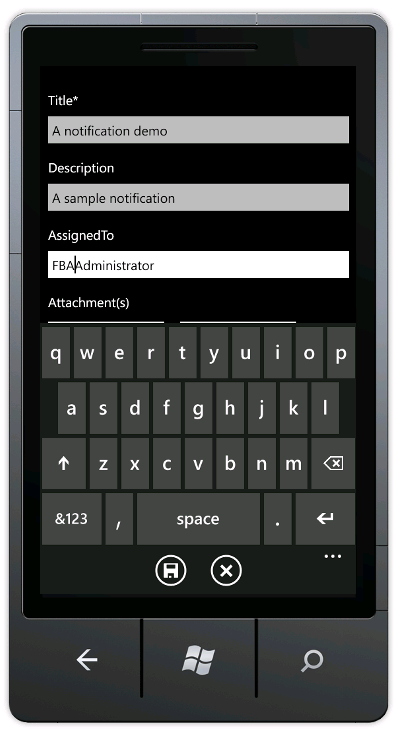
## Task 4 – Test the App

In this task, you will run the phone app in the emulator.

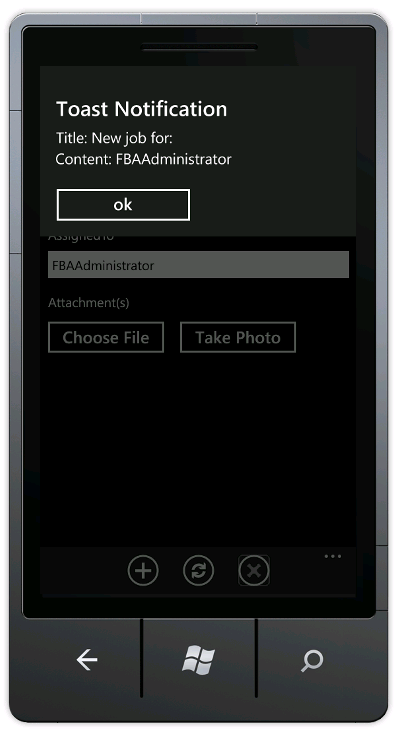
1. In Visual Studio, select **Debug⮚Start Debugging**.
2. When prompted, log in using Forms-Based Authentication.
3. When the list loads click the **Add** icon.
4. Add a new item to the list.
5. 
6. After the item is created, click the **close** button (the circle with an X inside it).
7. On the main screen of the application, click the **Register** button.



1. Go to the Jobs list in SharePoint and add a new item. Make sure to fill in the **AssignedTo** field.



1. Click **Save**.
2. Verify that you receive a notification.



* + - * + A new notification

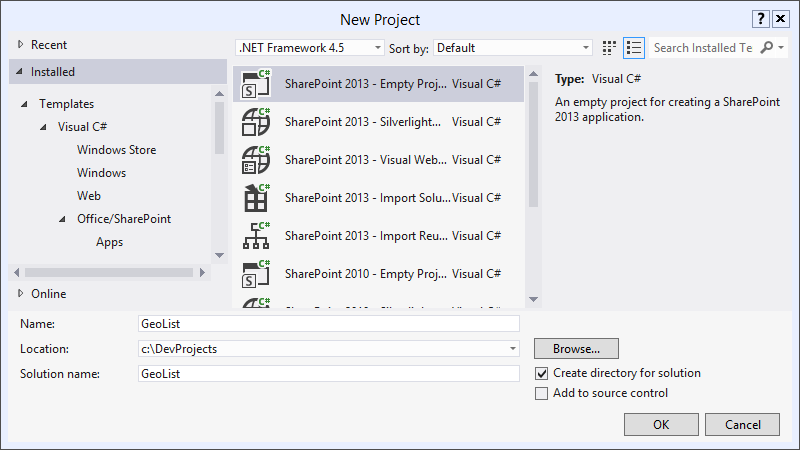
# Exercise 3: Creating a Map-Based Phone App

In this exercise, you will create a Windows Phone 7 app that uses a Location field to demonstrate mapping capabilities.

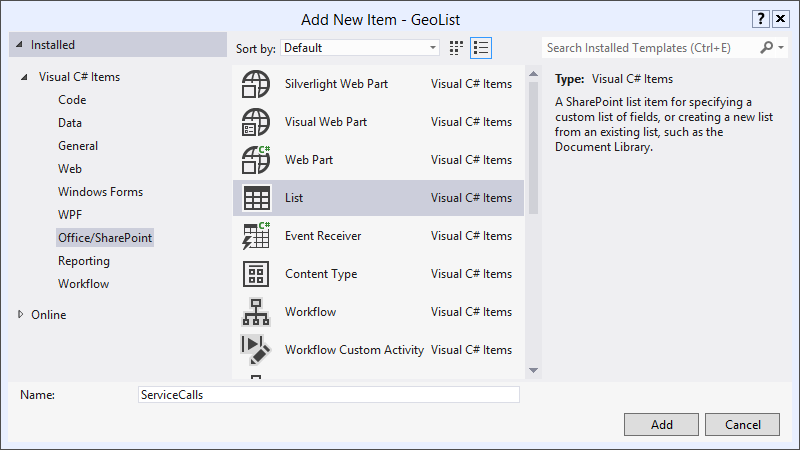
## Task 1 - Create the SharePoint List

In this task, you will create a SharePoint list containing a location field. Because this field must be added through code, you will create and deploy the list as a feature.

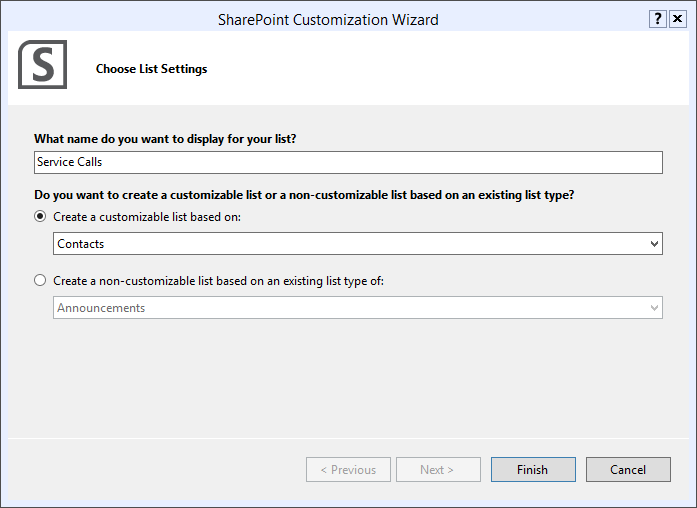
1. Log into the SharePoint server as an administrator.
2. Start **Visual Studio**.
   1. Select **File⮚New Project** from the main menu.
   2. In the New Project window, select **Visual C#⮚Office/SharePoint⮚SharePoint Solutions** and choose the **SharePoint 2013 – Empty Project** Project type.
   3. Name the new project **GeoList** and click **OK**.



1. In the **SharePoint Customization Wizard**, enter the URL for the site collection that uses Forms-Based Authentication that you set up for phone development, and choose **Deploy as a farm solution**.
2. In the Solution Explorer, right click the **GeoList** project and select **Add⮚New Item** from the context menu.
   1. In the **Add New Item** dialog, select **List**.
   2. Name the new List **ServiceCalls** and click **Add.**



* 1. In the **Choose List Settings** dialog, enter **Service Calls** as the display name (note the space in the name, this is important as the code in the following steps depends on this).
  2. Choose to customize the list based on the **Contacts** list type.



* 1. Click **Finish**.
     + - 1. Add a list

1. In the **Solution Explorer**, expand the **Features** node.
2. Right click the **Feature1** node and select add select **Add Event Receiver** from the context menu.
3. Add the following code to the feature event receiver.
   1. public override void FeatureActivated(SPFeatureReceiverProperties properties)
   2. {
   3. SPWeb site = properties.Feature.Parent as SPWeb;
   4. SPList list = site.Lists.TryGetList("Service Calls");
   5. if (list != null)
   6. {
   7. list.Fields.AddFieldAsXml(
   8. "<Field Type='Geolocation' DisplayName='Location'/>",
   9. true,
   10. SPAddFieldOptions.Default);
   11. list.Update();
   12. }
   13. }
   14. public override void FeatureDeactivating(
   15. SPFeatureReceiverProperties properties)
   16. {
   17. SPWeb site = properties.Feature.Parent as SPWeb;
   18. SPList list = site.Lists.TryGetList("Service Calls");
   19. if (list != null)
   20. {
   21. list.Delete();
   22. }
   23. }

## Task 2 – Deploy and Use the List

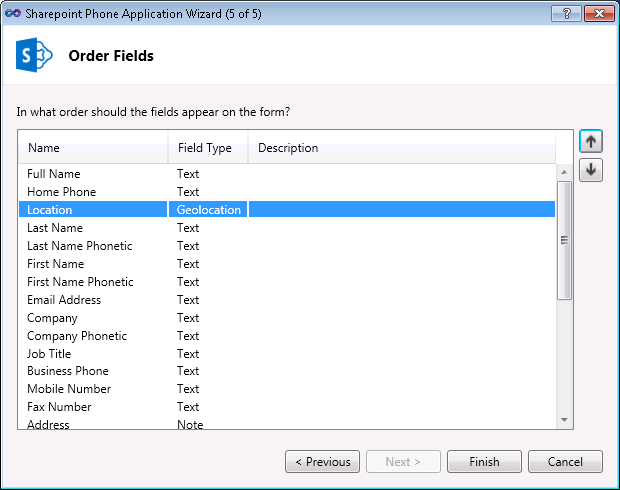
In this task, you will utilize the new Location field in SharePoint.

1. In the **Solution Explorer**, right click the **GeoList** project and select **Deploy** from the context menu.
2. After the list successfully deploys, open the site you are using for phone development.
   1. Click **More**.
   2. Click **Service Calls**.
   3. Click **Add New Item**
   4. Enter **Cox** in the **Last Name** field.
   5. Enter **Brian** in the **First Name** field.
   6. Enter **Brian Cox** in the **Full Name** field.
   7. Enter **555-555-5555** in the **Home Phone** Field.
   8. Click **Use Current Location** in the **Location** field.
   9. Click **Save**.

## Task 3 – Build the Phone App

In this task, you will create a phone app that uses the SharePoint list.

1. **Log In** to the Phone Development environment.
2. **Start** Visual Studio 2010.
3. Select **File⮚New⮚Project** from the main menu.
4. In the New Project dialog, select **Visual C#⮚Silverlight for Windows Phone⮚Windows Phone SharePoint List Application**.
5. Name the project **GeoApp**.
6. Click **OK**.
7. In the **SharePoint Phone Application Wizard**, enter the URL of the Phone Development Site Collection you created earlier (e. g., http://contososerver:8888) and click **Find Lists**.
8. When prompted, log in using **Forms-Based Authentication**.
9. Select the **Service Calls** list and click **Next**.
10. On the **Choose Views** screen, check **All Contacts** and click **Next**.
11. On the **Choose Operations** screen, check **Display** and click **Next**.
12. On the **Choose Fields** screen, check all of the fields and click **Next**.
13. On the **Order Fields** screen, reorder the fields so that **Full Name**, **Home Phone**, and **Location** appear at the top of the list.

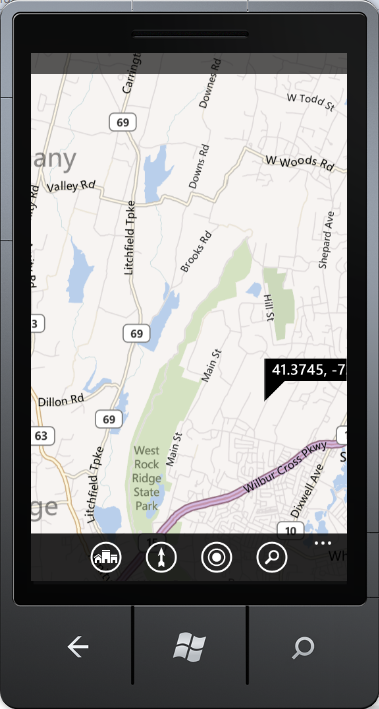


1. In the **List.xaml** file, change the header from **All Contacts** to **Service Calls**.
2. Open the **ListDataProvider.cs** file and change the **Context** property to the following:
   1. public override ClientContext Context
   2. {
   3. get
   4. {
   5. if (m\_Context != null)
   6. return m\_Context;
   7. m\_Context = new ClientContext(SiteUrl);
   8. Authenticator at = new Authenticator();
   9. at.AuthenticationMode = ClientAuthenticationMode.FormsAuthentication;
   10. at.PromptOnFailure = true;
   11. m\_Context.Credentials = at;
   12. return m\_Context;
   13. }
   14. }

## Task 5 – Start the App

In this task, you will run the phone app in the emulator.

1. In Visual Studio, select **Debug⮚Start Debugging**.
2. When prompted, log in using Forms-Based Authentication.
3. Click on **Brian Cox**.
4. Click the **Map It** link found in the **Location** field.



* + - * + Completed App