|  |  |
| --- | --- |
|  | Developer Guide |
| 5/7/2013 | Graduate Capstone |

Table of Contents

[1 Introduction 17](#_Toc359916620)

[1.1 Intended Audience 17](#_Toc359916621)

[1.2 References 17](#_Toc359916622)

[1.3 Revision History 17](#_Toc359916623)

[2 PCL 18](#_Toc359916624)

[3 Windows Phone 18](#_Toc359916625)

[4 Android 18](#_Toc359916626)

[5 Web Admin CMS 18](#_Toc359916627)

# 1 Introduction

@TODO: Rephrase Good explanation

*“A developer’s guide is a walkthrough of how to use the API - it’s like how a teacher would explain the API to their students, but it’s penned down in digital ink (and there’s no talking back!). Some of those students will be new to the API, some will be new to web development entirely, and some will be old pros - the guide should work for all of them.*

*At the minimum, a guide should demonstrate how to get started using the API, bringing the developer from zero knowledge to a working piece of code. If they can get to that minimal working piece of code, then they will feel much more motivated to continue exploring the API using just the reference.*

*If possible, a guide can walk through all of the aspects of using the API, for the developers that learn the best through the instructional format.”*

## 1.1 Intended Audience

This document is intended mainly for developers who will be writing code for the project. The goal is for those individuals to learn about the organizations, conventions and purposes of each file and object and what they do in the project.

## 1.2 References

* <http://developer-support-handbook.appspot.com/documentation.html>
  + Good Explanation of what is supposed to be accomplished in this type of document

## 1.3 Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Date | Reason For Change | Version |
| Andy Bottom | 06/24/2013 | Created the formatting of the document and added references | 0.1 |

# 2 Before You Start

Before you can start, there are several steps that need to be done, such as configuring you’re environment

All software dependencies can be found in the @TODO: link to which document and section , but the following section explain tricky setup steps that were needed in order to prepare the environment and have things running smoothly.

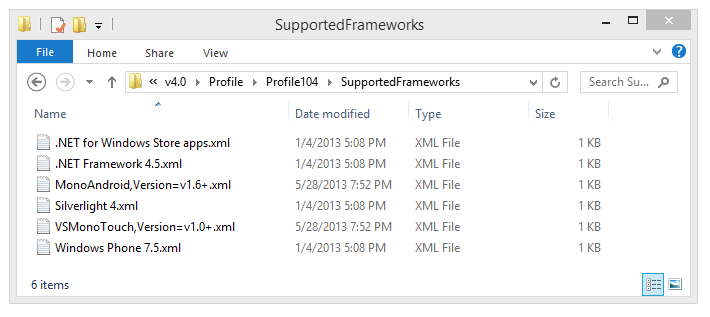
## 2.1.1 Visual Studios 2012

You will need to be running Visual Studios 2012 as it is designed to function properly with

### 2.1.1.1 PCLs for Android and iOS

By default, PCL libraries only work for the Windows Platforms (Windows Phone, Windows Store, X-Box.) However, the project relies on the ability to put all the back-end logic into the PCL to be used by all Phone OS Platforms. Thus additional configuration must be done so that VS can recognize the PCL for all platforms.

The resource comes from <http://slodge.blogspot.com/2012/12/cross-platform-winrt-monodroid.html>.



@TODO: Make a resource folder and copy the files from visual studio into there and document the steps

### 2.1.1.1 Xamarin Plug-In

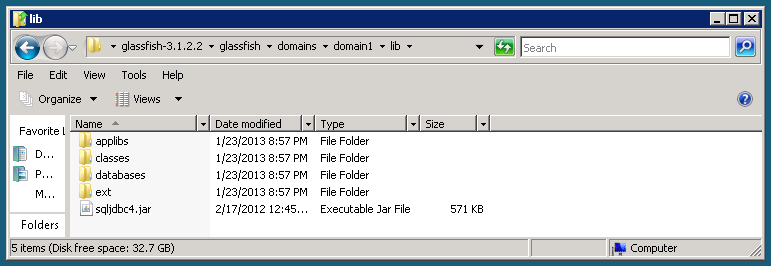
The project utilizes the Xamarin Platform, so the plugins must be installed.

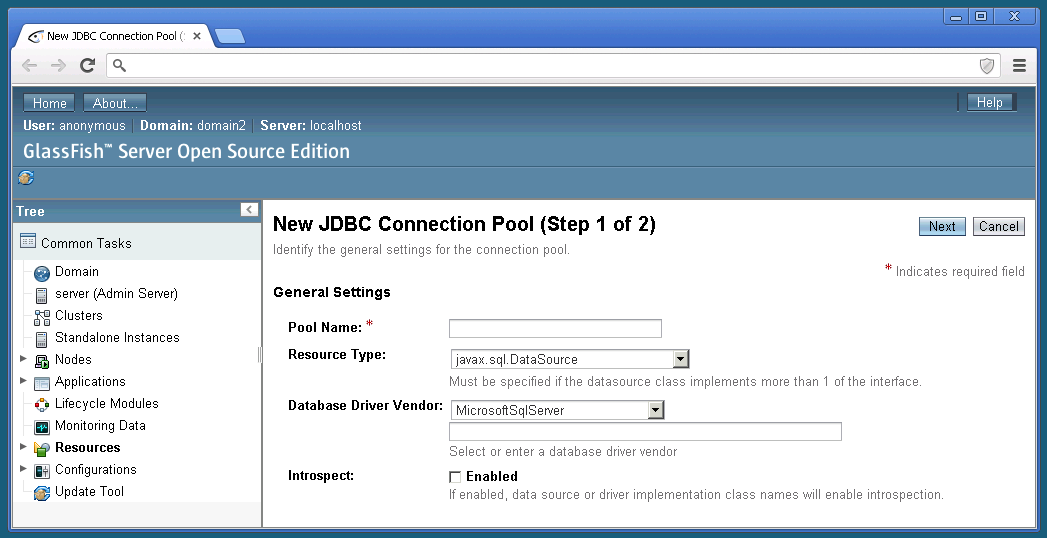
@TODO: Get screenshots and directions for the Xamarin Platform.

## 2.2 Netbeans IDE

The Netbeans IDE is used to create the Web Services. Other Java IDEs could be used, but I use this one, thus instructions about how to set it up will be given.

### 2.2.1 Glassfish with Sql Server Support

Since the Database is on a Microsoft SQL Server, the glassfish server must contain the jar resource so that it can successfully make this connection. 



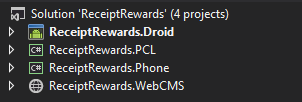
@TODO: Put jar in resource file and then document the steps and screenshots

### 2.3 Virtual Machines and Emulators

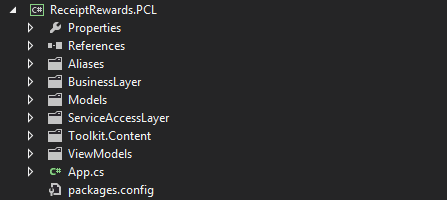
During the testing of the phone app, you will be using the Phone Emulators. However, you may run into a problem where you receive an error message saying “TODO: Error Message” and the Emulators fail to start.

@TODO: Steps for the computer startup settings to allow virtual machine.

### 2.4 Project Solutions



# 3 PCL

The portable class library is the common functionality needed for all version of the phone application. Think of the PCL as containing everything needed for the phone app, just without the user interface and handlers. The phone applications will utilize this special library to make the app work and function.

The PCL contains certain packages which will be described in greater detail below.

## C:\Users\Ernie\Desktop\DeveloperGuideScreenshots\SolutionExplorer_PCL_02.PNG3.1 References Package

The references folder contains all the libraries that the PCL code is dependent on. As you can see in the structure below, the MCCM Cross Platform is used mainly for the ability to perform the Web Services JSON Requests, and the Newtonsoft is used to easily parse the JSON into objects.

## C:\Users\Ernie\Desktop\DeveloperGuideScreenshots\SolutionExplorer_PCL_03.PNG3.2 Aliases Package

The alias classes are the objects used to receive and transfer the standard objects to and from the Restful Services via an anonymous structure. This allows the request to be substantially smaller, thus improving the performance.

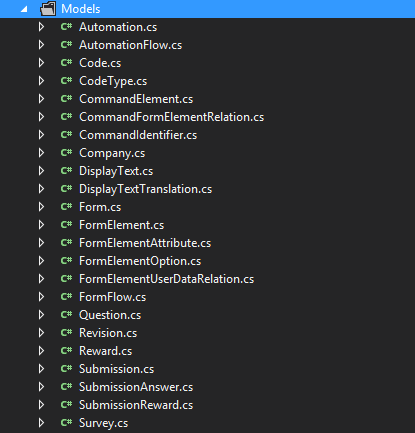
## C:\Users\Ernie\Desktop\DeveloperGuideScreenshots\SolutionExplorer_PCL_04.PNG3.3 Business Layer Package

The business layer of the PCL contains static classes of methods that are commonly used throughout the rest of the PCL classes, and even for other code bases to use. By having this package, instills an emphasis on code reuse.

* *Alias Helper:* Contains the logic to convert all the alias objects to and from the standard objects.
* *Simple Rest Service:* Contains the logic to perform an asynchronous call to the Web Service Layer. The requests are based on JSON.
* *Common Utils:* Contains very miscellaneous functions that were used. Simply a code reuse of most common functions.

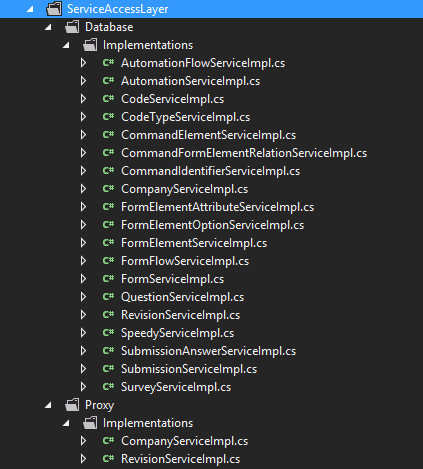
## 3.4 Models Package

The models contain all the objects used in the system for the C# code base.



## 3.5 Service Access Layer Package

The Service Access Layer contains all the definitions to both the Web Services and the Reverse Proxy URL Locations.



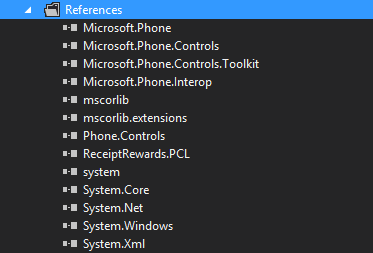
## C:\Users\Ernie\Desktop\DeveloperGuideScreenshots\SolutionExplorer_PCL_08.PNG3.6 View Models

The View Models is the most important aspect of the PCL in regards to the phone applications. These classes are logical representations of all the screens that will be in all the phone applications. In other words, these are only the business logic and functionality of a page. These are everything but the UI and event handlers. The actual applications will utilize a view model and hook up the UI and handlers to the logic located in the view models.

# C:\Users\Ernie\Desktop\DeveloperGuideScreenshots\SolutionExplorer_Phone_07.PNG4 Windows Phone

The windows phone project is the source code of creating the phone application. The structure to the project is follows the standard structure of the windows phone applications.

## C:\Users\Ernie\Desktop\DeveloperGuideScreenshots\SolutionExplorer_Phone_02.PNG4.1 Properties Folder

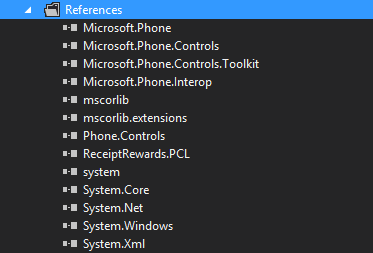
The properties folder contains the configuration files involved with how the app functions on the operating system.

## 4.2 References Folder

The references folder contains all the libraries and jar files that gets used in the phone application.

Specifically the libraries are the default jar files that are be default loaded into phone applications. Also, there are some open source controls that are installed too.

## C:\Users\Ernie\Desktop\DeveloperGuideScreenshots\SolutionExplorer_Phone_04.PNG4.3 Panorama Model Package

The panorama package contains all the view objects that are the actual pages. These are the objects that get bind to the view models located inside the PCL library.

## 4.4 Resources Folder

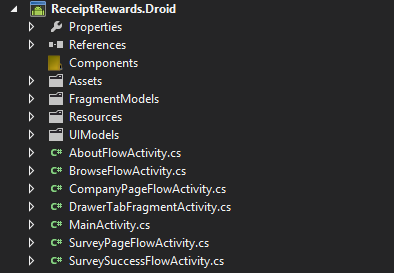
The resources folder contains all of the extra files that the phone will need. Specifically this contains images that are displayed to the user.

## C:\Users\Ernie\Desktop\DeveloperGuideScreenshots\SolutionExplorer_Phone_08.PNG4.5 Pages

These are the XAML pages that implement what

# 5 Android

@TODO



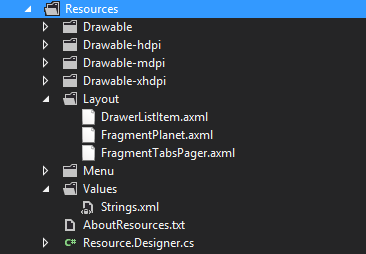
## 5.1 Properties

@TODO

C:\Users\Ernie\Desktop\DeveloperGuideScreenshots\SolutionExplorer_Android_07.PNG

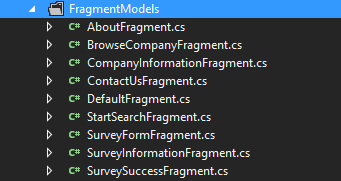
## 5.2 References

@TODO



## 5.3 Fragment Models Page

@TODO



## 5.4 Resources

@TODO

## C:\Users\Ernie\Desktop\DeveloperGuideScreenshots\SolutionExplorer_Android_06.PNG

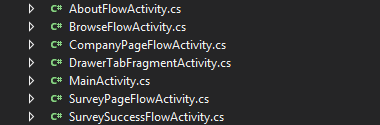
## 5.5 UI Models

@TODO

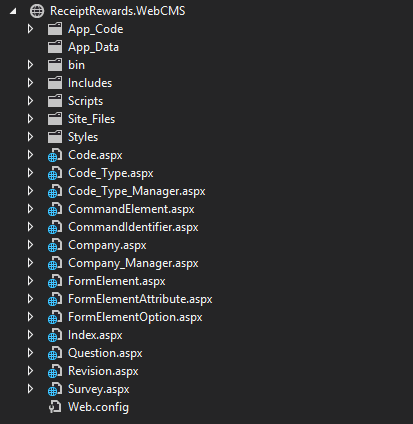
## C:\Users\Ernie\Desktop\DeveloperGuideScreenshots\SolutionExplorer_Android_09.PNG

## 5.6 Pages

@TODO

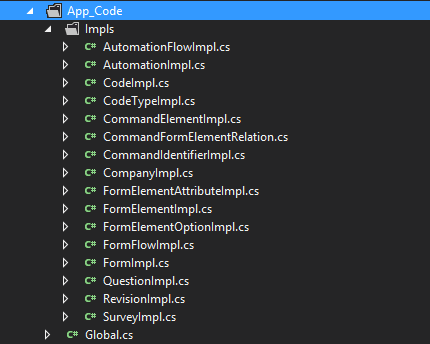


# 6 Web Admin CMS

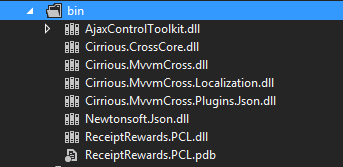


## 6.1 App Code

@TODO



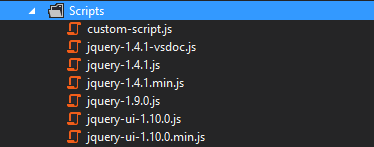
## 6.2 Bin



## 6.3 Includes

C:\Users\Ernie\Desktop\DeveloperGuideScreenshots\SolutionExplorer_Web_04.PNG

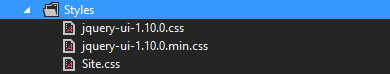
## 6.4 Scripts



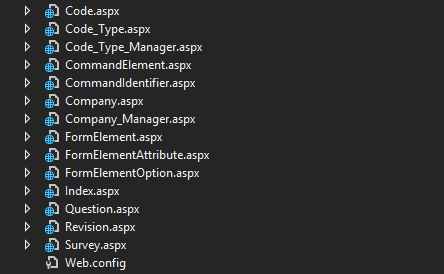
# 6.5 Site Files

C:\Users\Ernie\Desktop\DeveloperGuideScreenshots\SolutionExplorer_Web_06.PNG

# 6.6 Styles



## 6.7 Pages

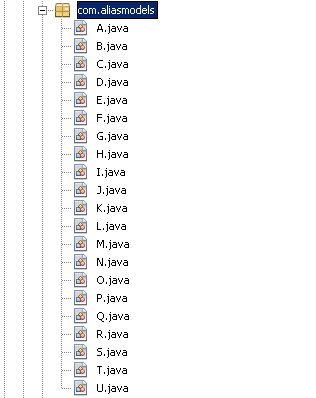


# 7 Web Services

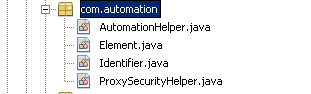
# C:\Users\Ernie\Desktop\DeveloperGuideScreenshots\SolutionExplorer_WebServices_01.PNG

# C:\Users\Ernie\Desktop\DeveloperGuideScreenshots\SolutionExplorer_WebServices_02.PNG

## 7.1 Alias Models Package



## 7.2 Automation Package



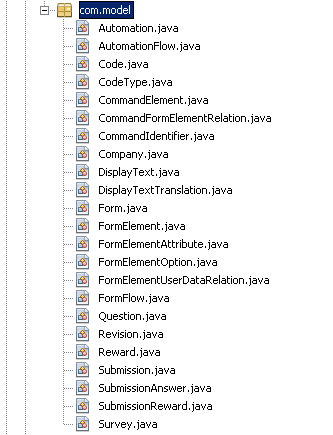
## 7.3 Common Package

C:\Users\Ernie\Desktop\DeveloperGuideScreenshots\SolutionExplorer_WebServices_05.PNG

## 7.4 Controller Package

## C:\Users\Ernie\Desktop\DeveloperGuideScreenshots\SolutionExplorer_WebServices_07.PNG

## 7.5 Models Package



## 7.6 Libraries

