David Bradshaw

Professor Cook

CSCI 4910

16 September 2011

Developing Applications for Windows Phone 7

Recently the world of Information Technology has seen an explosion of new technology, devices, and software all in one field. The field of mobile communication has expanded since the release of cellular phones, and has recently started expanding exponentially since the release of smartphones. Many people do not understand exactly what a smartphone is, verse what a standard cellular phone does; Encyclopædia Britannica defines a smartphone as a "mobile telephone with a display screen, built-in personal information management, and an operating system that allows other computer software to be installed (smartphone)." By following this definition we can see that the main difference between a cellular phone and a smartphone is the ability to install software to broaden what the device can do. As of today, there are four main smartphone operating systems in use; other manufactures have operating systems for their devices, but they are not as widely used as the four main operating systems.

A recent Nielson study showed the growth of cellular network usage and to what device was using it. The study showed that the most network traffic went to Google's operating system, Android OS, this is most likely due to the open source like license that Google uses, allowing multiple smartphone manufactures to use Google's operating system on their hardware. The second most network traffic went to Apple's operating system, iOS, which is used to run the iPhone, iPad, and the iPod Touch. The third most network traffic goes to our topic of discussion, Windows Phone 7 (Whitney).

Windows Phone 7 is a relatively new mobile phone operating system which has taken the market by storm with many experts thinking that it will beat out Apple's iOS by 2015, partly due to Nokia's decision to use Windows Phone 7 on their future devices, and no longer use Symbian OS (Camm-Jones). Windows Phone 7 offers many features that make it a contender to other smartphones. Some of Windows Phone 7's main features include the Live Tiles, on the devices home screen, which keeps you up to data with all of your information and friends. The inclusion of Microsoft Office Mobile, allows for creation and editing of Office Document on the go. The integration to XBOX Live allows current XBOX 360 users to play games with their friends and save scores to their XBOX Live account, as well as new users to download and play games instantly on their smartphone. Like other smartphones Windows Phone 7 also offers a marketplace for Apps, Music, Videos, and games. Windows Phone 7 integrates into the Zune Marketplace so that all of your data integrates easily with your PC copy of Zune. The phone also offers numerous built in apps like, Calendar, Mail, Bing, Internet Explorer Mobile, and many more great features ("Features").

Like other smartphones, Windows Phone 7 allows independent and professional developers to develop applications for their operating system. Microsoft has created a website called App Hub, which is part of the MSDN network on Microsoft's website. App Hub allows developers to create Windows Phone 7 Applications and XBOX 360 Games. App Hub provides developers with all the tools needed to create a full Windows Phone 7 application. In order to create an application for Windows Phone 7 we first need to get a development environment set up. Most people should be able to use their current computer, as long as it meets the system requirements. To set up our development environment we need Windows Vista or Windows 7; all edition or architectures, except Starter Edition. We will also need a computer with at least

4GB of Hard Drive space and 3GB of Memory. You cannot use Windows XP or a Virtual Machine to run Windows Vista or 7; the SDK will not install on this setup. Now that our Development Environment is setup and meets the requirements we need to install the Windows Phone 7 SDK; you can download this from App Hub. If you currently have a copy of Visual Studio 2010 installed, you need to be sure it is updated to Service Pack 1, if you do not currently have Visual Studio 2010, the SDK will install the Express Edition for you. The SDK will install the SDK, Visual Studio Express; if needed; Expression Blend for Windows Phone, XNA Game Studio, Sample Code, and Documentation; this installation will take a bit of time. Once the installation is complete, restart if necessary, and you are ready to create your Windows Phone 7 Application ("Windows Phone Development").

Now that we have the SDK for Windows Phone 7 installed we can create a Windows Phone 7 Application. To get started go to Start >> All Programs >> Microsoft Visual Studio 2010 >> Microsoft Visual Studio 2010. If you have created a shortcut to Visual Studio 2010, you can use it as well. For this paper I am focusing on the development of Windows Phone 7 applications, if you are not familiar with Visual Studio 2010 I recommend browsing the MSDN library to learn more about how Visual Studio 2010 works. Once Visual Studio 2010 is open we can create our application. Go to File >> New >> Project. From this menu you can either choose Visual Basic or Visual C#, whichever language you are more familiar with. For demonstration purposes I will use Visual C#. Once you choose your language go to Silverlight for Windows Phone and then choose Windows Phone Application. If you are prompted to choose a Target Platform, I recommend choosing Windows Phone 7.1; which is the newest update to Windows Phone 7, codename Mango; however, you can create for the older edition, and then upgrade in the future. This option will change as new versions are released, at the time of this paper, Windows Phone 7.0 was current and Windows Phone 7.1 is scheduled for release in Fall of 2011. You will now have a basic Windows Phone 7 application created.

The first page you will see should have an image of a generic Windows Phone 7; this is a live preview of how your application currently looks. The first page will be called MainPage.xaml. This is a markup based document, like XML or HTML, which you can use to create the layout of elements on your page. If you look in the Solution Explorer you will see a list of the files in your current project. Let me take a moment to explain what the different files are for. The first one, called Properties, is a collection of the document properties; including Application information, Developer Information, and Project Information; the next two, called References and App.xaml, are files that ensure that your app follow and can access the features built into Windows Phone 7 API. The image files includes in this collection, called ApplicationIcon.png, Background.png, and SplashScreenImage.jpg; are files that can be edited in your preferred photo editor and are used to show your app in the End User's application list of home screen. The final file, called MainPage.xaml is the design layout of the application, and if you expand this file you can see the MainPage.xaml.cs[vb for Visual Basic]. In the end, the only two files that you need to modify are the MainPage.xaml and MainPage.xaml.cs. The use of these two standard files allow for easy programming, you can use the XAML file to edit how the application looks and then the C# or Visual Basic file to edit what the program does; like what to do on load, button press, shake, or any event (Petzold). The next two sections will show you how to edit the XAML file to create the layout of your application, and then the next section will show you how to create handlers to perform actions when events occur.

Now that you have your MainPage.xaml file open, let us take a look at the Toolbox section. The Toolbox section has a list of the different controls available to us, this includes;

AdControl, Border, Button, Canvas, Checkbox, Ellipse, Grid, Hyperlink Button, Image, List Box, Map, Media Element, Password Box, Progress Bar, Radio Button, Rectangle, Scroll View, Slider, Stack Panel, Text Box, Text Block, Web Browser. It may seem hard to believe, but with the combination of these items, you can create nearly an application you want. For the most part each of these items are self-explanatory, however if you hover over them with your mouse, Visual Studio 2010 will provide a brief description. With the blank canvas on the generic phone, you can drag and drop any item you want over to the Grid; you have to have a Grid in your Phone, before you can add stuff. If you would like to add additional windows, that will contain items, that are not seen outside of the window, Hierarchy Layout, you can create Canvases for each Window, and drop the items into the Canvas. You can resize any item you drag and drop into the Grid, by simply grabbing a side or corner; you can also change text by making the element active and then find the Properties. From this section you can change the element name by changing the box by the element type [note: if you have Events created, changing a name of an item can break the Event, it is best to set names before coding Events]. From the Properties box you can also change what is displayed in the item [if text is displayed in this element, like a button, Text Box, Radio Button…] by changing the Content option, you can also change other values in the property window to manipulate the item into exactly what you want. You can repeat this process for every element you need for your application. I highly recommend creating your application's layout before writing Event handlers or code.

Now that we have our application looking how we like, we need to make it do something. To do this we will add Event Handlers. For example, we can create an event handler to activate when the End User presses a Button. To do this double click the button on the generic Windows Phone view; this will create an event for the most used action of that element [button is Click], if you want to specify a different event, highlight the element and go to the Properties page, then click the Event tab, double click the Event you want to use. This will take you to your MainPage.xaml.cs file which is where we store all of our code and event handlers. You will see a section created called ‘private void [element]_[event]’ you can then add any code in here, when the event takes place the code will be executed. Now, you can apply that idea into any task that you are trying to accomplish. For example, if you wanted a button to be pressed and show the text “Hello, Welcome to Windows Phone 7!” you could use this simple code inside your button_Click action (Petzold).

```
private void button1_Click(object sender, RoutedEventArgs e)
 {
     textBlock1.Text = "Hello, Welcome to Windows Phone 7!";

 }
```

To accomplish this you need to have a Button and a Text Block on your program. This is just an example of how to code Windows Phone 7. Depending on your task at hand, you can use the numerous tools available to create a full application that can do whatever you can think of. When coding, be sure to have given your elements easy to remember names. Failure, to name your elements properly can lead to simple errors that could have been avoided with a better naming scheme. When coding, make sure that you take into consideration Error Handling, if your application performs an operation that could generate an Error, consider using a Try/Catch to ensure that your application does not crash. If you want your application to be able to save its state when closed you will need to implement the proper backend code structures to ensure that your application saves its state and can retrieve it when the application is closed by the End User. The implementation for this is available through MSDN and requires backend files to be altered to allow this feature to occur (Windows Phone Silverlight Application Life Cycle…).

Now that you have an application, it is time to test it out. Microsoft makes this super easy with the built in Emulator available right from Visual Studio 2010. Once your app is ready, and contains no errors, you can hit the play button, or go to Debug >> Start Debugging [or Start Without Debugging, just to run the program]. Before running the app make sure your application Properties page is accurate, and a default language and Build status is selected. Once you have ran it inside the emulator, and it worked you can get ready to package it for distribution. The first step is to package your application for distribution, in Visual Studio 2010 go to Build >> Build Solution. The Output Console on the bottom will tell you where you can find your XAP file and if any errors prevented building your application. A XAP file will be generate in your Solution folder, a XAP file is the packaged content of your application. It contains all the files and code you created, XAP uses the standard ZIP compression, so if you want to see what is inside, change the extension to ZIP, and you can view the items of your application – Be sure to change the file back to ZAP before attempting to submit it (Petzold). Once you have your XAP file you are ready to upload it to App Hub. To do this you need an App Hub account. You can visit http://create.msdn.com, to create an account; at the time of this paper there was a $99.00 USD Annual Fee to publish on App Hub. Once you have an App Hub account you can log in and submit your application. To submit your application go to My Dashboard >> Windows Phone. From here click Submit a New App, follow the Wizard, and give your application a name, description, price, and select to what markets to distribute to. While you are submitting your application, App Hub will test it for basic required features; if you pass you will be able to submit your application. Once you submit your application, it will take about a week for it to be added to the Zune Marketplace, it will go through testing to ensure that it works properly and is not malicious or contains inappropriate content. Once your application is approved you will be able to see it in the Zune marketplace, and from App Hub be able to see how many downloads and reviews it has received.

We have seen many amazing features available in the new smartphone operating system available from Microsoft, and how, as developers we can use them to create amazing applications. App Hub has provided easy to use tools, resources, and an amazing platform to create next generation applications. Windows Phone 7's implementation of already existing technology makes it easy to learn and debug for professional and novice developers. Windows Phone 7 uses Silverlight to write applications in the Visual Basic and Visual C# languages. With the use of Silverlight you can create engaging and interactive applications, using Visual Studio 2010 or Expression Blend 4. Due to the integration of Silverlight, application porting from Web Servers to Mobile Phones is also a great way to move existing application to Windows Phone 7 with little changes needed for the new platform. For developers wanting to create games for Windows Phone 7 they can easily accomplish this with the existing tools used in XBOX 360 Arcade games, the XNA Game Development SDK can be used to create games for Windows Phone 7 as well. In the end, Windows Phone 7 is a great platform for developers of all experience, with easy to use tools, and freely available information makes it a contender for other smartphones. With future updates and features planned for Windows Phone 7, it will provide a rapidly growing market for application developers to target their applications to.

Works Cited

Camm-Jones, Ben. "Windows Phone to Beat iOS by 2015, Analysts Say." PCWorld. 30 March 2011. Web. 16 September 2011. http://www.pcworld.com/article/223693/windows_phone_to_beat_ios_by_2015_analysts_say.html.

Microsoft Corporation. "Features." Discover Windows Phone 7. Microsoft Corporation, 2011. Web. 16 September 2011. http://www.microsoft.com/windowsphone/en-us/features/default.aspx.

Microsoft Corporation. “Windows Phone Development.” Microsoft Developer Network (MSDN): Windows Phone. Version 1.0 (2011): Web. 16 September 2011. <http://msdn.microsoft.com/library/ff402535(v=VS.92).aspx>.

Microsoft Corporation. “Windows Phone Silverlight Application Life Cycle for Application Launch, Activate, and Deactivate Events.” App Hub Educational Resources. Version 1.0. (2010): Web. 16 September 2011. http://create.msdn.com/en-US/education/catalog/article/silverlight_application_life_cycle.

Petzold, Charles. Programming Windows Phone 7: *Microsoft Silverlight Edition*. 1st Edition. Redmond, Washington: Microsoft Press, 2010. Print.

"Smartphone." Encyclopædia Britannica. Encyclopædia Britannica Online. Encyclopædia Britannica, 2011. Web. 16 September 2011. <http://www.britannica.com/EBchecked/topic/1498102/smartphone>.

Whitney, Lance. "Nielsen: Smartphone data usage soars 89 percent." CNET News. 17 June 2011. Web. 16 September 2011. http://news.cnet.com/8301-1023_3-20071941-93/nielsen-smartphone-data-usage-soars-89-percent.