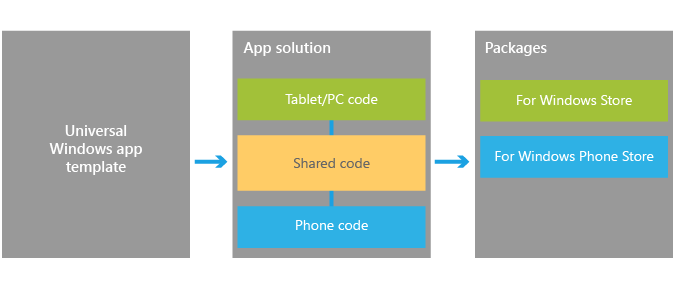
# About me

I am an IT professional with current experience in writing business applications in C#, C, Xaml, Silverlight on Windows desktop, Windows store, and Windows phone. I have also had previous experience writing applications in Lawson 4GL, Visual Basic, Cobol, PL/SQL, Sybase T/SQL, C, Perl, Java, and C# .Net on the Windows, Unix/Linux, Web, and Mainframe platforms. Many of my systems have used Sybase, SQL Server, Oracle, or DB2 as the back end database and various front end and batch languages. I have 15+ years of experience designing, coding, and testing systems and started writing code for fun long before it became my career. I have been exposed to the entire lifecycle of systems development from the initial idea, gathering requirements, developing a solution, writing code, testing, deployment, and post go live support.

# Universal app. What is it



Develop once for all Windows devices using a unified Windows runtime and VS tools that allow you to both support experiences unique to a device in XAML, HTML, and DirectX, and share the code that supports those experiences across all devices using C++, C#, or JavaScript. When your work is finished you can you can produce the app packages that you will submit to the Windows Store and Windows Phone Store with a single action to get your app out to customers on any Windows device.

In terms of the Store experience, customers will benefit from an app identity shared across the Windows Store and the Windows Phone Store. Shared identity means that if they purchase your app from the Windows Store, they are capable of installing it on a device from the Windows Phone Store using the same Microsoft account without having to purchase the app again. Optionally, this can also include things like in-app purchases.

To get you started, the following sections will identify key upgrade paths, migration paths, and other critical resources that will help you bring your ideas to the converged Windows 8.1 platform. For more information on universal Windows apps and context on Microsoft’s vision for developer opportunity on Windows devices, please see our blog post [Introducing universal Windows apps](https://go.microsoft.com/fwlink/p/?LinkId=394342).

## How to get setup to do store and phone development

Windows store and phone apps have gone through a number of changes over the last couple of years. The stores started out separate but Microsoft has been working to bring both stores together. Getting a developer license started out at about $100 then 10 and now free. The newest iteration adds a few levels depending on what your app qualifies for. I have not yet spent much time looking at what this means.

<https://dev.windows.com/en-us>

<https://devcenterbenefits.windows.com/>

## Where are we with feature parity

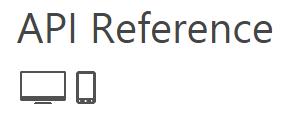
What I have found is that we are really close on feature parity between universals 8.1 store and phone and then Silverlight 8.1 apps but not completely there.

## Documentation

<http://msdn.microsoft.com/library/>

## **Requirements**

|  |  |
| --- | --- |
| **Minimum supported client** | Windows 8 |
| **Minimum supported server** | Windows Server 2012 |
| **Minimum supported phone** | Windows Phone 8.1 [Windows Phone Silverlight 8.1 and Windows Runtime apps] |
| **Namespace** | Windows.Media.MediaProperties  Windows::Media::MediaProperties [C++] |
| **Metadata** | Windows.winmd |



Packages Class

<http://msdn.microsoft.com/en-us/library/windows/apps/xaml/windows.applicationmodel.package.aspx>

Phone support does not necessarily mean that it works in Universal app yet.

## **Requirements**

|  |  |
| --- | --- |
| **Minimum supported client** | Windows 8 [Windows Store apps, desktop apps] |
| **Minimum supported server** | Windows Server 2012 [Windows Store apps, desktop apps] |
| **Minimum supported phone** | Windows Phone 8 |
| **Namespace** | Windows.ApplicationModel  Windows::ApplicationModel [C++] |
| **Metadata** | Windows.winmd |

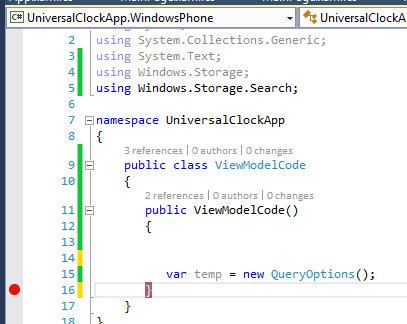
## Not implemented issues

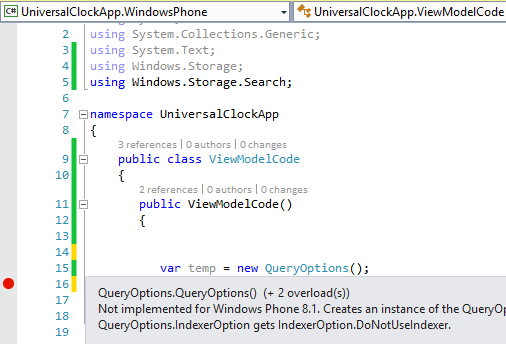
Strongly agree here, it's confusing because you never know for sure when an API you use will blow up on either platform, unless you try it out (or read the docs, TBH). One of such examples is in Windows.Storage namespace:

KnownFolders.MediaServerDevices exists on both platforms, but using it in Windows Phone would blow up "unexpectedly".

On the other hand, ApplicationData.Current.LocalCacheFolder exists only on Windows Phone and you can't compile a Store project until you #ifdef it out.

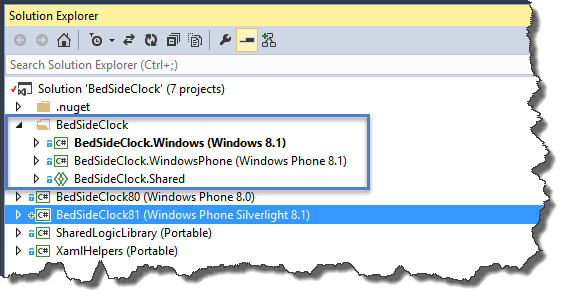
Both APIs return a StorageFolder so it's not like they are completely different things. Hiding unsupported APIs would make this work much more transparently for developers...



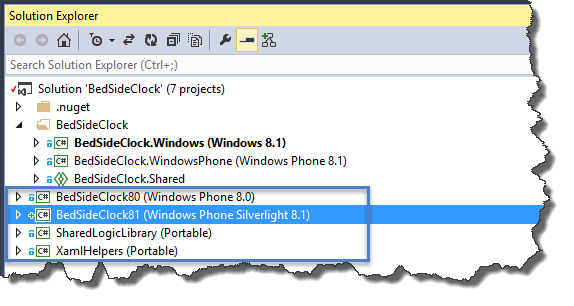


## Parts of the universal app

Universal app is made up of 3 parts, the Windows Store project, the Windows Phone project, and the shared project.

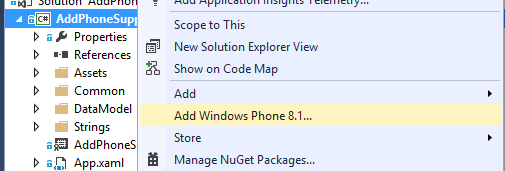


You can also have Silverlight phone projects in the same solution that can share code but not as straight forward as a Universal app.



## Adding support for phone to existing apps

Because of the documentation and differences between store and phone, this can be a more difficult approach to adding phone support.



## Tools

### Windows phone Power tools

<http://wptools.codeplex.com/>

### Iso Store Spy

<https://isostorespy.codeplex.com/>

### Xaml Spy

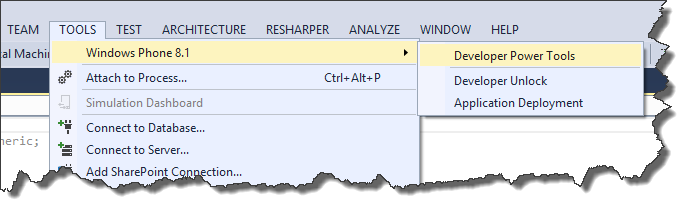
<http://xamlspy.com/>

I have not done much with this tool yet but having access to how the xaml tree is created can help understand what is happening.

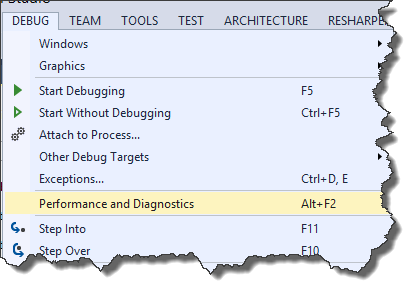
### Diagnostic and Profiling

#### Windows Phone Developer Power Tools (8.1)

This is part of the Windows Phone 8.1 sdk but requires an install on the device. If you install these tools with the device connected to your pc you will get a message about having to install on the device. You will then get a message on the device that an update is available on the device.



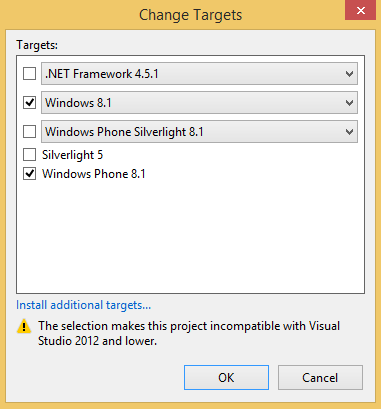
#### Visual Studio Performance and Diagnostics



# Portable class library’s

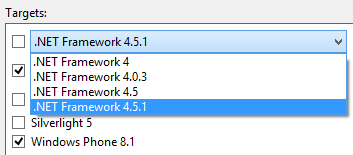
## What can be shared

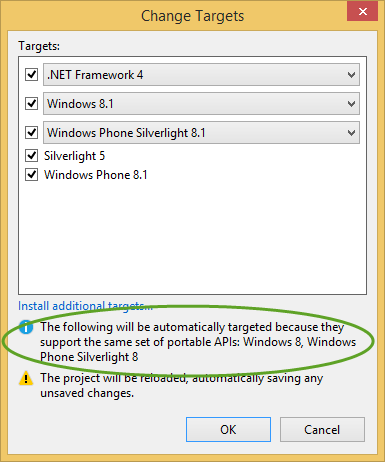
Portable class libraries can be used and can share some classes but you have to be careful since 8.0 level of support is not the same as 8.1 and the actual support will be the lowest common denominator.



### .net versions and how they affect what can be shared

So if you select .Net 4.0 then you can create a pcl that will work in an 8.1 application but only supports feature available for .Net 4.0 and Phone 8.0.





### Sharing combinations

WPF, Silverlight,

<http://msdn.microsoft.com/en-us/library/system.windows.data.ivalueconverter(v=vs.110).aspx>

Windows Store, phone documentation

<http://msdn.microsoft.com/en-US/library/windows/apps/windows.ui.xaml.data.ivalueconverter.aspx>

IValueConverter – Xaml is a mess between versions of .net and platforms.

Windows phone and store use an IValueConverter defined below

object Convert(object value, Type targetType, object parameter, string language);

object ConvertBack(object value, Type targetType, object parameter, string language);

But WPF and Silverlight phone is defined this way

object Convert(object value, Type targetType, object parameter, CultureInfo culture);

object ConvertBack(object value, Type targetType, object parameter, CultureInfo culture);

Obviously, the CultureInfo has changed in future versions of Xaml. This does make sense from the stand point of originally, I am sure the designers were thinking that you would only use a converter to change text around. But with the way binding works in Xaml you can use converters for all kinds of things like changing bool to Visibility or maybe the color of the text. In these cases, language really has no relevance to it is a useless parameter.

Also, CultureInfo was added in .Net 4.5 so what if we have conversion logic that is already being used in .Net 4.0?

And should you be able to create a portable class library or a normal class library that is common between WPF and Silverlight Phone?

<http://msdn.microsoft.com/en-us/library/system.globalization.cultureinfo(v=vs.110).aspx>

## How to share when that doesn’t work

# Async and Await hell

# Hardware performance

# Differences between platforms

## Openfilepicker on both platforms

## Tomestone

## Suspension manager

## Continuation manager

## Phone platform

### Suspend/Resume debugging

# Trust the xaml

## Binding is your friend

## viewStates

### Transitions and animations

# Debugging

# Helpful frameworks