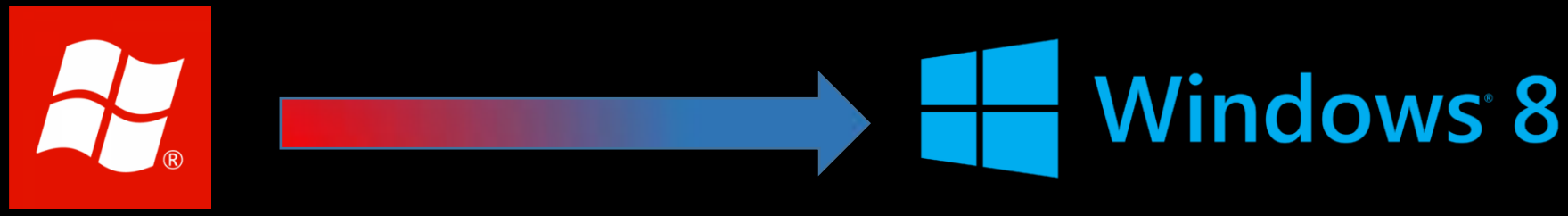
Windows 8 for WP Developers

Hermit Dave



WP Apps published (Invoke IT)

- Scribble
- Alarm Clock
- Learn to write
- Slydr
- Cool Camera
- Nature Sounds
- Wordastic
- Cineworld
- CVC Words

Topics

- Async
- Storage
- Playing Sounds
- Behaviours
- User interface
- Capabilities
- Portable Library

Async

- In Windows Phone dev
  - Async Web service calls
  - Use of thread pool / threads
  - Event driven code
- Windows 8 takes it further
  - System calls are async.
  - Instead of event driven code / callbacks, use await
  - Use of Tasks – no threads directly. Can use Threadpool
  - Sync calls if you do not use await
- Use Async CTP in WP Dev to make porting easier ?


```
private void SetWord(bool bNewWord = true)
{
    this.processedChars.Clear();
    this.SetTitleBackground(true);

    if (bNewWord)
    {
        while (true)
        {
            CurrentWord = LangHelper.GetNextWord();
```

```
private async void SetWord(bool bNewWord = true)
{
    this.processedChars.Clear();
    this.SetTitleBackground(true);

    if (bNewWord)
    {
        while (true)
        {
            CurrentWord = await LangHelper.GetNextWord();
```

Storage

- WP Dev
  - IsolatedStorageFile
  - IsolatedStorageSettings
- Windows 8 dev
  - Storage File / Storage Folders.
  - Get Access to .NET streams using above
  - Package.Current.InstalledLocation.Path – for reading content files
  - ApplicationData.Current provides LocalFolder, RoamingFolder & TempFolder
  - ApplicationData.Current provides LocalSettings & RoamingSettings
  - RoamingFolder / RoamingSettings shared between devices.

```
string file = String.Format("Content/{0}.txt", GetLocale(WordasticConfig.Dictionary));

Uri uriPath = new Uri(file, UriKind.RelativeOrAbsolute);

StreamResourceInfo sriPath = System.Windows.Application.GetResourceStream(uriPath);

using (Stream s = sriPath.Stream)
```


```
string File = Path.Combine(Package.Current.InstalledLocation.Path,
    String.Format("Content\\{0}.txt", GetLocale(WordasticConfig.Dictionary)));

StorageFolder folder = await StorageFolder.GetFolderFromPathAsync(
    Path.GetDirectoryName(File));

using (Stream s = await folder.OpenStreamForReadAsync(Path.GetFileName(File)))
```

```
using (IsolatedStorageFile isf = IsolatedStorageFile.GetUserStoreForApplication())
{
    if (isf.FileExists(GetFileName<T>()))
    {
        using (IsolatedStorageFileStream fs = isf.OpenFile(GetFileName<T>(), System.IO.FileMode.Open))
```


```
StorageFolder storageFolder = Windows.Storage.ApplicationData.Current.LocalFolder;

StorageFile storageFile = await storageFolder.CreateFileAsync(GetFileName<T>(),
    CreationCollisionOption.OpenIfExists);

using (Stream inStream = await storageFile.OpenStreamForReadAsync())
```

```
public static bool IsSettingPersisted(string Key)
{
    return IsolatedStorageSettings.ApplicationSettings.Contains(Key);
}
```


```
public static bool IsSettingPersisted(string Key)
{
    return ApplicationData.Current.LocalSettings.Values.ContainsKey(Key);
}
```

Playing Sounds

- Windows Phone
  - Silverlight or XNA to play sounds
  - MediaElement
  - MediaPlayer
  - SoundEffects
- Windows 8
  - XAML or DirectX
  - MediaElement
  - SharpDX – managed wrapper for DX http://sharpdx.org

Behaviours

- Windows Phone
  - Blend provides excellent set of behaviours though System.Windows.Interactivity.dll that comes with blend
  - FluidMoveBehavior, MouseDragElementBehavior etc
  - Can use both XAML and codebehind to manipulate behaviours
- Windows 8
  - No built in support for behaviours
  - http://winrtbehaviors.codeplex.com provides DragFlickBehavior

User Interface - Layout

- Windows Phone
  - Portrait / Landscape
  - Pivot
  - Panorama
- Windows 8
  - Blank
  - Basic
  - Split
  - Items
  - ItemsDetail
  - GroupedItems
  - GroupedDetail

User Interface - Layout

- Windows Phone
  - Support for auto rotation depending upon SupportedOrientations
- Windows 8
  - Apart from Blank page, all pages derive from LayoutAwarePage
  - Automatically supports change in orientation
  - Explicit orientation support using DisplayProperties.AutoRotationPreferences

User Interface - Navigation

- Windows Phone
  - Page exposes BackKeyPress
  - NaviationService
    - Navigate using Uri to XAML Page
    - Properties CanGoBack & CanGoForward
    - Methods GoBack & GoForward, AddBackEntry & RemoveBackEntry
- Windows 8
  - Frame
    - Navigate using Type not Uri
    - Properties CanGoBack & CanGoForward
    - Methods GoBack & GoForward
  - LayoutAwarePage
    - Methods like GoBack & GoHome
  - No back button – add required buttons to page (layoutaware adds back button)


```
NavigationService.Navigate(
    new Uri("/GamePlay.xaml", UriKind.Relative));
```

```
if (NavigationService.CanGoBack)
    NavigationService.GoBack();
```

```
Frame.Navigate(typeof(Gameplay));
```

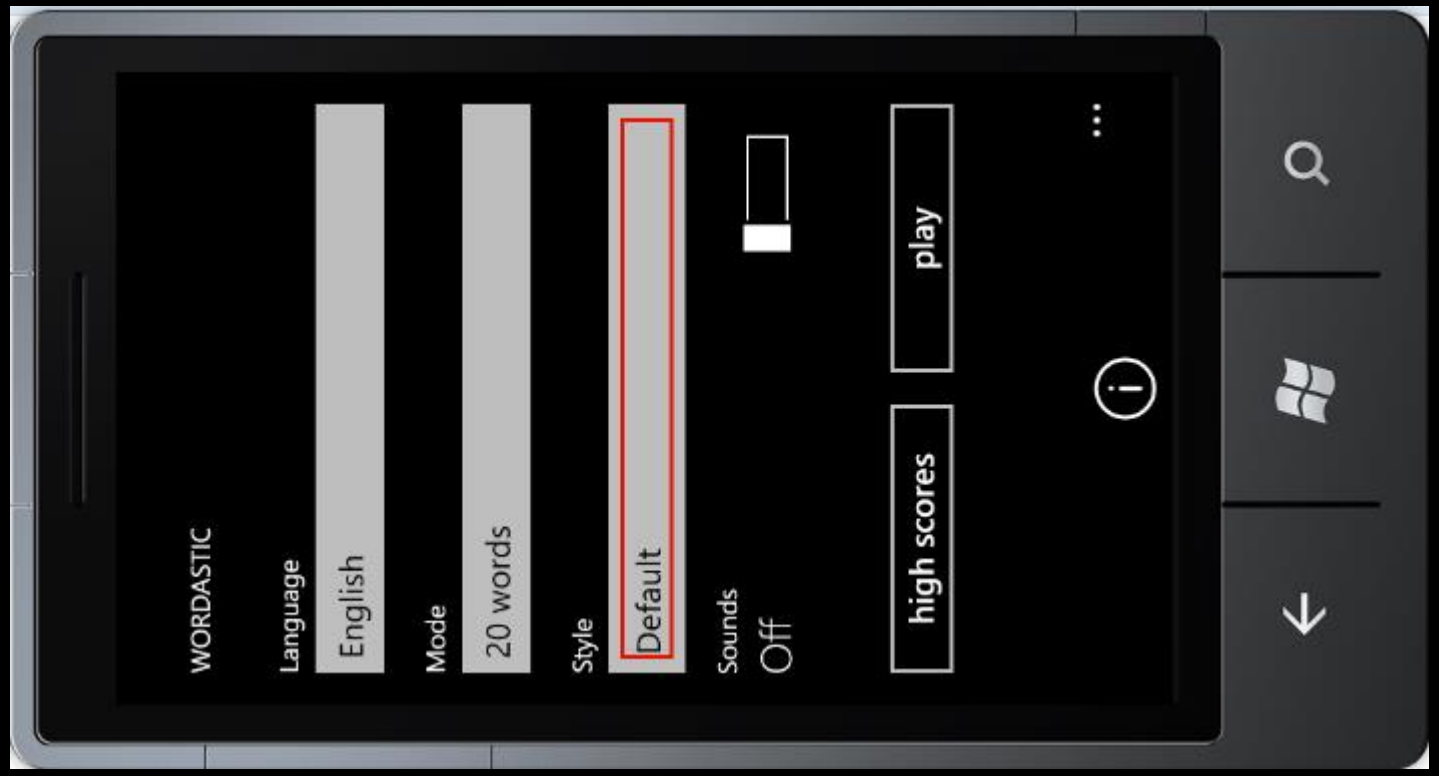
```
if (this.Frame != null && this.Frame.CanGoBack)
    this.Frame.GoBack();
```

User Interface - Views

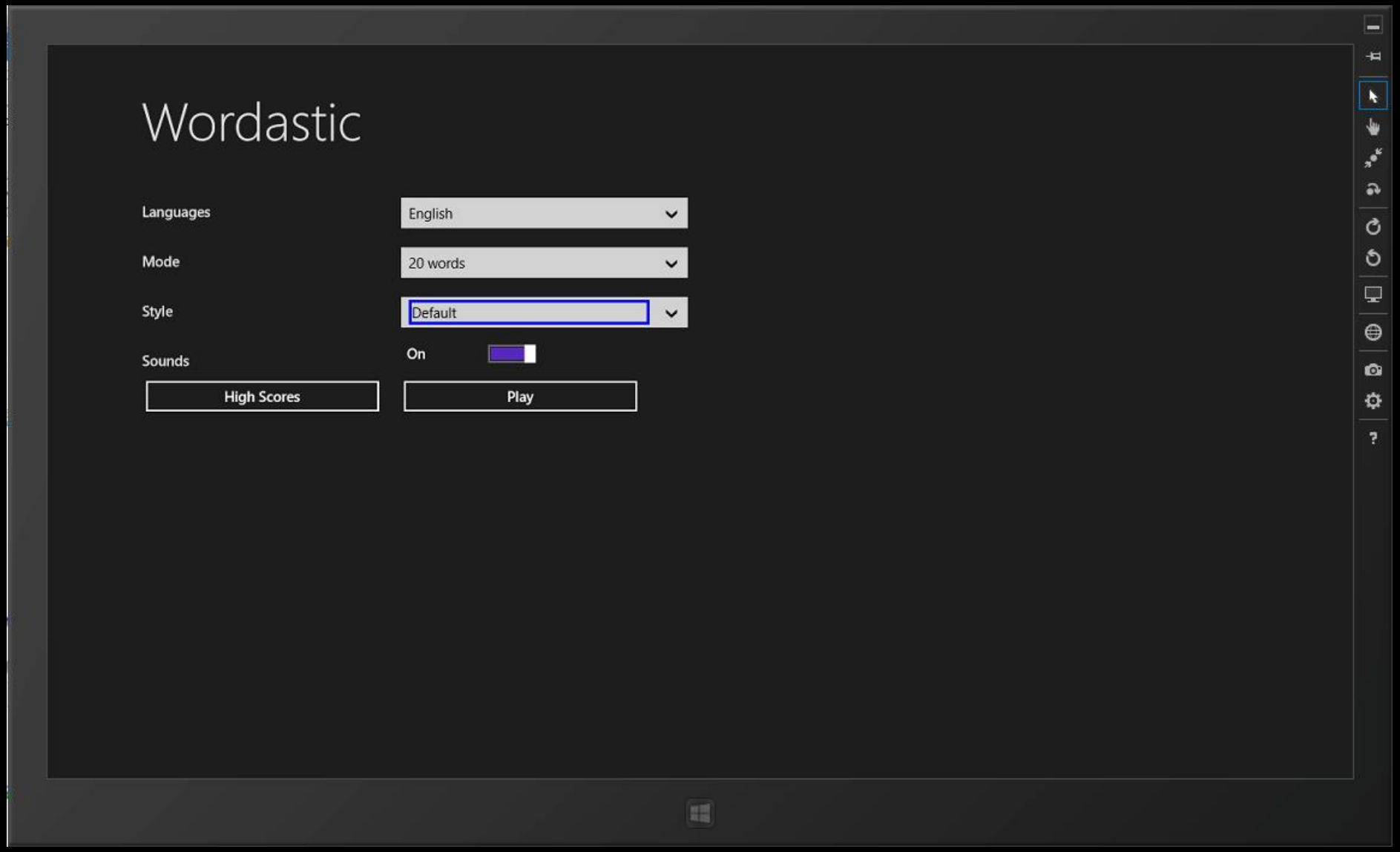
- Windows Phone
  - Landscape
  - Portrait
- Windows 8
  - FullScreenLandscape
  - Filled
  - FullScreenPortrait
  - Snapped

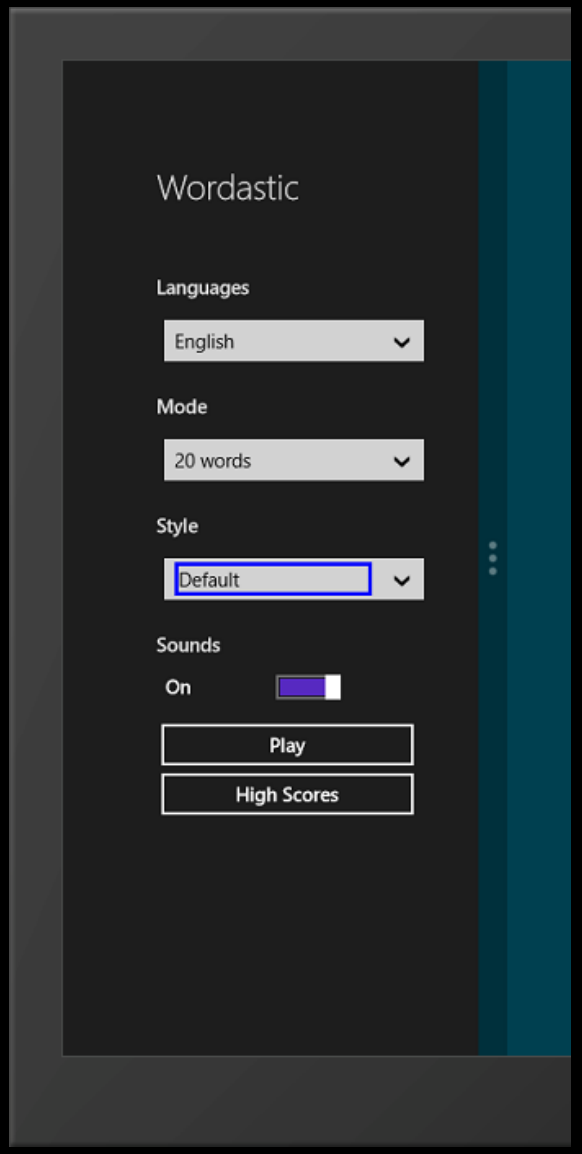
  - LayoutAware automatically adds these. Customise them as needed.
  - Snapped View is required – Can choose to support it fully if wanted


WORDASTIC
Language
English
Mode
20 words
Style
Default
Sounds
Off
high scores
play


WORDASTIC
Language
English
Mode
20 words
Style
Default
Sounds
Off
high scores
play

Windows 8


Wordastic
Languages
English
Mode
20 words
Style
Default
Sounds
On
High Scores
Play

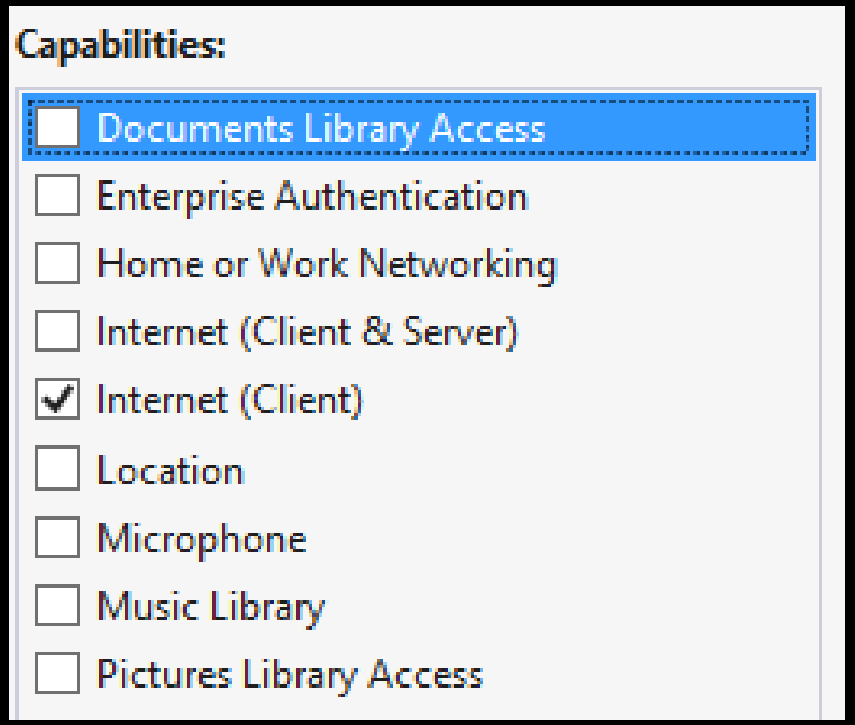

Wordastic
Languages
English
Mode
20 words
Style
Default
Sounds
On
Play
High Scores

Capabilities

- WMAppManifest.xml

```
<Capabilities>
  <Capability Name="ID_CAP_GAMERSERVICES" />
  <Capability Name="ID_CAP_IDENTITY_DEVICE" />
  <Capability Name="ID_CAP_IDENTITY_USER" />
  <Capability Name="ID_CAP_LOCATION" />
  <Capability Name="ID_CAP_MEDIALIB" />
  <Capability Name="ID_CAP_MICROPHONE" />
  <Capability Name="ID_CAP_NETWORKING" />
  <Capability Name="ID_CAP_PHONEDIALER" />
  <Capability Name="ID_CAP_PUSH_NOTIFICATION"
  <Capability Name="ID_CAP_SENSORS" />
  <Capability Name="ID_CAP_WEBBROWSERCOMPONENT
  <Capability Name="ID_CAP_ISV_CAMERA" />
  <Capability Name="ID_CAP_CONTACTS" />
```

- Package.appxmanifest

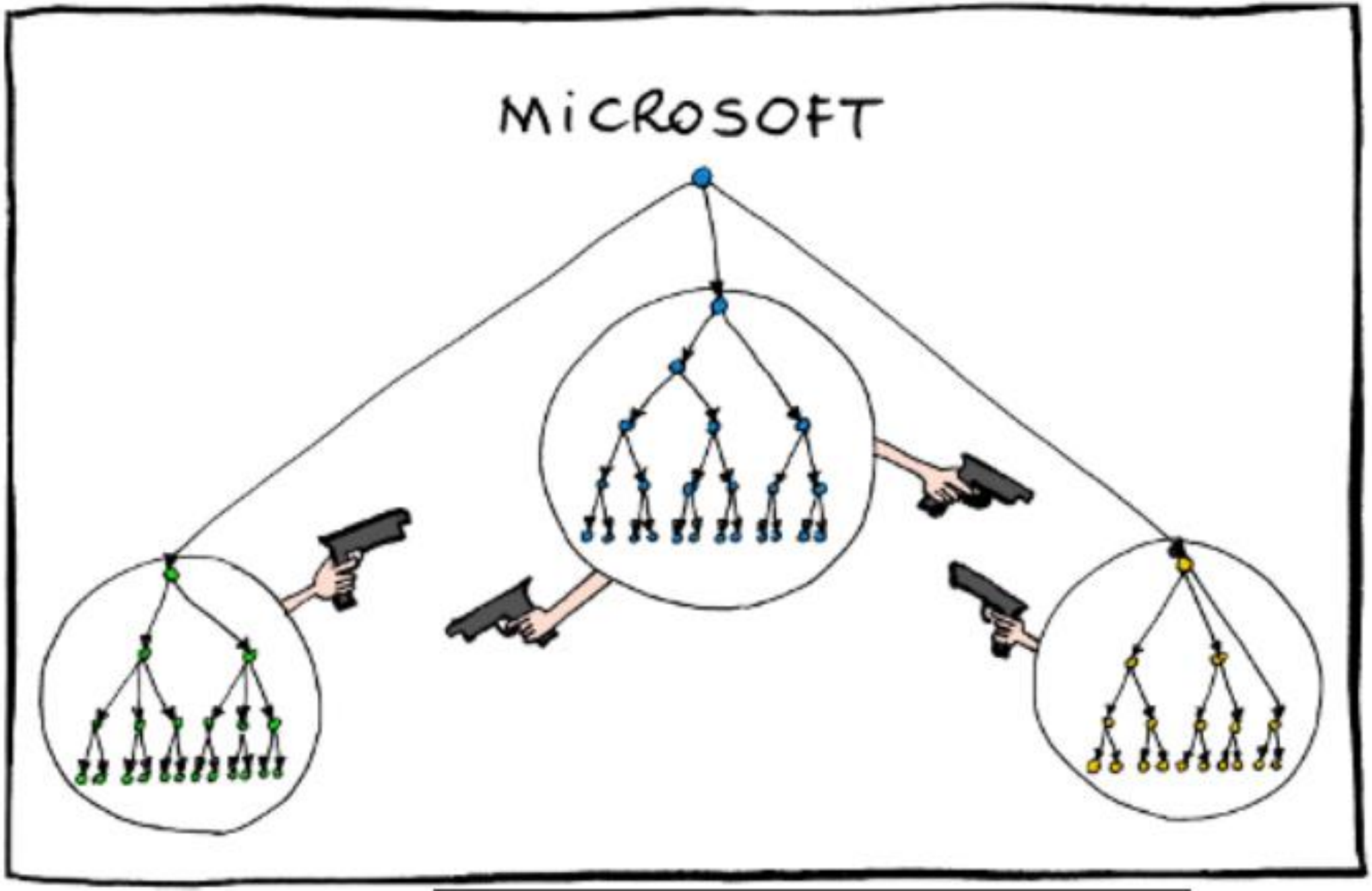


Portable Library

- In addition to use Async CTP, you can put common logic in portable library.
- Portable libraries require you to choose target platforms and the output can be directly referenced on other platforms.
- Create with VS 2012 for broadest support
- http://bit.ly/PortableLib for more info
- Alternative to porting code to individual platforms

Portable Library

Feature	Assemblies	.NET Framework	Metro style	Silverlight	Windows Phone	Xbox 360
Core	mscorlib.dll, System.dll, System.Core.dll, System.Xml.dll, System.Xml.Serialization.dll	√	√	√	√	√
Managed Extensibility Framework (MEF)	System.ComponentModel.Composition.dll	√	√	√		
Network Class Library (NCL)	System.Net.dll	√	√	√	√	
Serialization	System.Runtime.Serialization.dll	√	√	√	√	
Windows Communication Foundation (WCF)	System.ServiceModel.dll, System.ServiceModel.Web.dll	√	√	√	√	
Model-View-View Model (MVVM)	System.Windows.dll	Only 4.5	√	√	√	
Data annotations	System.ComponentModel.DataAnnotations.dll	Only 4.0.3 and 4.5	√	√		
LINQ to XML	System.Xml.Linq.dll	Only 4.0.3 and 4.5	√	√	√	
.NET for Metro styles apps	See .NET APIs for Metro style apps.	Only 4.5	√			


MICROSOFT

XNA

- As you know XNA for some strange un-explained reason did not make it to Windows 8 Metro apps.
- Xamarin to the rescue
- http://monogame.codeplex.com/
- Allows you to create XNA game useable on Metro and available in Marketplace as Metro apps.
- Consume your favourite XNA api
- Still use C++ and DX if you like it like that.