

Student App Quality Guidelines

Contents

- Introduction 3
- Pillar 1: Unique Concept or Idea 5
 - Idea Generation 5
 - App Research 5
 - Other Considerations 6
- Resources 7
- Action Points 7
- Pillar 2: User Experience and App Design 8
 - User Experience 8
 - App Design 8
 - App Presentation in the Store..... 8
- Resources 9
- Action Points 9
- Pillar 3: Feature Richness 10
 - Windows 8 Features 10
 - Windows Phone Features 11
 - Automation 12
 - Top 20 Features 12
- Resources 13
- Action Points 13
- Pillar 4: Quality Coding..... 14
 - Windows Phone 8 14
- Resources 15
- Action Points 15
- Pillar 5: Designed for multiple devices/contexts 16
 - Multiple Devices..... 16
 - Multiple Contexts..... 16
- Resources 16
- Action Points 17

Pillar 6: Cloud Integration	18
User Settings	18
Data	18
Users	18
Push.....	19
Cloud Services	19
Resources	20
Action Points	20

Introduction

So you want to create an app? Or perhaps a game? We want to help! And we don't want to simply help you to create any old app or game, but want to give you tips, tricks, techniques and tutorials on how to make the best app or game you can.

Fortunately, creating and publishing an app for the Windows platform has become increasingly accessible and straightforward with ever-improving tools that are stable, incredibly functional and provide aid for the developer in a myriad of ways. Microsoft has a variety of languages and technologies a developer can choose from; a wealth of choice that enables developers to create high quality client, mobile, gaming and cloud applications. In addition, Microsoft has a great selection of platforms to develop for, from Windows 8 to Windows Phone to cloud-based solutions with Windows Azure, and even the Xbox 360.

Sound exciting? It should! Making an app has never been easier and the sheer variety of technologies, platforms and tools you have at your disposal gives you the ability to choose the one that suits your needs and your chosen direction.

To make a great, entertaining, feature-rich application, it's not necessarily as simple as having easy to create apps, and so this document provides guidelines on how to make your app shine. They help a developer have the best chance at success in getting an app published to the appropriate store. As a result, developers could end up with a highly regarded entry that receives great reviews, is used by many and could be held up as a fantastic example of what makes an app shine on the Microsoft platform over the competition.

We're conscious that developers run the gamut of experience levels, some with great skill, some approaching app development for the first time. The intention of this guide is to help you recognize the crucial components of a quality app regardless of your experience. For those just starting out, at times it's hard to know if your idea is unique right at the beginning or, at the other end of the process, are unsure how to publish your app effectively.

To help the developer adopt the concept of "quality app" creation instead of simply building a templated app or creating an app in isolation of fundamental processes, we've highlighted some of the top key areas for what makes up a quality app. In this guide, you'll find six pillars of quality – foundational concepts that, if built upon properly, will increase the chance of a successful outcome for you.

And guess what? Having a great quality app, not only provides benefit to your end users but great benefit to you as well. It could be seen as an online resume or a way for people to see your progress historically if you're looking for a career in technology, and just as importantly, the Windows Store and Windows Phone Store provide great revenue opportunities too!

The six pillars are:

- **Pillar 1: *Unique Concept or Idea*** – making sure you can come up with a great idea that is interesting and is not "just another" app amongst a bunch of identical apps is critical.

- **Pillar 2: *User Experience and App Design*** – without great design principles in place, it's easy to have an app have a confusing or illogical experience and/or design and overall provide a poor user experience. This starts with the first place the user interacts with the app – in the store.
- **Pillar 3: *Feature Richness*** – an important concept in differentiating apps built on the Windows platform is implementing the many unique and creative features on Windows 8 or Windows Phone, while keeping a focus on the one specific thing the app is made to do.
- **Pillar 4: *Quality Coding*** – without great code underneath the app, the chances of hitting certification errors in security, technical and content compliance increases through bugs, crashes and freezes.
- **Pillar 5: *Designed for Multiple Devices and Contexts*** – with the wealth of platforms to choose from, building an app that runs across multiple devices, such as a Windows 8 laptop and a Windows Phone 8 mobile device, is a way to differentiate and bring extra value. So if you have a great mobile app, consider how it might work on Windows 8 too. Also, making sure you take advantage of the multiple contextual views that the platform offers, such as snapped and filled views on Windows 8, is also key.
- **Pillar 6: *Cloud Integration*** – the cloud is now an integral part to many quality apps, from roaming settings across devices, to using notification services and live tile updates, to storing historical data or even high score lists for games for comparison between players. Great apps will utilize the cloud in appropriate aspects of the app itself, and using Windows Azure as part of the development process makes this a very straightforward proposition..

Pillar 1: Unique Concept or Idea

One of the most important starting points in the creation of an app is coming up with a great idea, and experience shows that some developers struggle with this. As a result, their app can be unimaginative and overly simple, or falls into a mass of “same idea, same execution” apps, or simply is not advisable, or violates principles maintained by the Windows Store.

Idea Generation

Here’s a great example: a developer decides to build a calculator. It’s something they use often, and it’s a simple enough concept to be easily coded. So off they go, developing without some simple research. However, with five minutes of searching in the store, it’s easy to see that there are currently at least 615 calculator apps in the Windows Store. This is most probably enough, but what if it was a specific sub-category of calculators, say for Body-Mass Index (BMI)? Again, a simple search reveals that there are over 50 BMI specific calculators in the store. Perhaps they should revisit the idea generation phase for something else.

But how do they even come up with the idea? Experience has shown that a lot of developers find it difficult to generate an idea. Without that initial idea it can be a struggle to get started. Or the developer chooses an existing app to replicate, or they come up with something that is unadvisable, such as using copyrighted content, or creating offensive applications which will ultimately be rejected at publishing time.

Two possible solutions are Ideafests and a guidance on themes. The former are events, or a segment in a bigger event, where you get together with other student developers in one place and brainstorm ideas. Those students who have an idea present it to the group and have the others either come up with their own idea in response or join the presenter as a team to build that particular app in a subsequent hackathon or development effort.

The latter helps idea generation by providing a constraint on the imagination and thought process required in coming up with a great concept. Establishing a theme to work to can aid you in focusing on a particular topic and finding solutions to that area, and allows you to think through:

- Creatively solving an existing problem
- The features you can implement to differentiate your app
- Better analysis of the unique aspects of the Microsoft platform over others in implementing the particular theme, including specific differentiating aspects of Windows, Windows Phone and Windows Azure

App Research

Once you have your idea generation process complete, the next step is research. As mentioned earlier, this is another area where some students, not having truly experienced the technology world from a creator’s standpoint, often fall short. Here’s a quick step-by-step process you can follow:

How to Research (example talking points)

1. First search the store for similar apps to the idea. If there aren’t any, great, move on with the app!

2. If there are similar apps, look at the user submitted reviews of each to look for missing features. This helps identify whether there is indeed scope for another app of this variety.
3. Use the existing apps and find the areas that could be improved upon. Check for differentiated features such as Windows 8 Contracts like Share and Search, synchronizing data to the cloud, etc.
4. If there is sufficient scope for an app of this particular topic with additional features that will have it stand out, then great, move on with the app! If not, think about generating other ideas.

If you are designing a mobile application, you could also research across the various platforms with App Explorer (<http://www.appexplorer.com>). Doing this can often reveal other features you could implement, especially if there is specific user feedback that could easily be implemented on the Microsoft Platform.

Store analysis shows that a significantly high percentage of apps that are considered low quality fall into two areas of concern that are part of this quality pillar:

- Duplicates or Clones – a direct copy of an existing application, often right down to the app tile.
- Spam – multiple apps that are published by the same person but do the same thing with only very minor tweaks that might be better served to be condensed into one app.

Both of these areas have a high chance of rejection at the publishing to the Windows Store stage.

Sketch out your app idea to show how you will differentiate from the others already in the store. Show the navigation path so you can check for logical flow. Make sure you reduce the clutter in your idea and focus on doing one thing well.

Now you can review your research, how your specific app idea might look and behave and consider how it could be viewed by the user. Just because there are similar entries in the store, doesn't mean all is lost for a particular idea.

Thinking back to our example of a BMI calculator, if there had been only a few, the next step of analysis would have been to determine if all of the existing entries had missed important facets of working out BMI. If they all had missed a fundamental feature, such as synchronizing the BMI data to the cloud, or implementing the Share Contract, then despite there being a number of BMI calculators already, consideration for creating new one could continue.

Other Considerations

You should also think through your app idea and make sure it provides great value to the user. Typically, if your app falls into one of the following categories, it is not considered great quality or value:

- Only implements an RSS feed without significant additional functionality.
- Single page of content with very limited functionality.
- Displays a single feed from a social media channel such as Twitter or YouTube, such as a single channel or an individual Twitter user or search query result.
- Uses a web browser for its core content.

- Uses the web for its core functionality by simply providing buttons to go to the internet for that function.
- Based on a WindowsSDK sample with no or little modification and little extra value added.
- Unit converters of only one type of unit.

Resources

- Windows Phone app quality guidance document highlights key areas of concern for student created apps when it comes to unique ideas and avoiding “spam” apps – <http://aka.ms/AQ016>
- App Category Opportunity Map (forthcoming) provides insight into what areas of the store are ideal for student apps and which ones need more apps in general.

Action Points

- **DO** think about your app idea carefully.
- **DO** research once you’ve come up with your idea. Look for apps similar to yours, and determine if there’s scope to make something with sufficient differentiating features.
- **DO** review the app category opportunity map to identify areas of the store that would benefit from more apps.
- **DON’T** submit the app you learn to build at an app camp. Instead, come up with a new idea for your first app to be published to the store. Submitting an app based on a lesson or tutorial will most likely be rejected during the publishing process.

Pillar 2: User Experience and App Design

Good User Experience and Design is crucial to the creation of a quality app. Without a compelling and easy to use interface, users will simply not use the app no matter how functional it is under the hood. With thousands of apps in the store, they'll more likely just find another one that does a similar thing but better.

User Experience

In general, apps:

- Need to be easy to use
- Have a logical application flow
- Should provide a compelling experience that could be interesting or exciting to the user
- Need to look professional, fun, visually stimulating from a graphics perspective, and/or fall into line with the Windows design language.

Unfortunately a lot of developers do not have experience in visual or user experience design. As a result, apps built by a single developer can sometimes fall (far) short of the mark on the user experience side of the app creation process. However, Microsoft has made available a significant amount of content available to help designers, and those developers wanting to understand how to design better, approach the Microsoft platforms with an appropriate direction for their app.

The Windows Store app design guidelines come in a variety of forms. For example, <http://design.windows.com> has a collection of great resources to leverage. And we run one and three day camps specifically aimed at designers for approaching Windows app design properly.

App Design

The design process isn't just about the user interface, but about approaching the overall design of the application effectively. This includes data design, infrastructure design and functional design of the application. The above guidelines include coverage of these topics too but often students aren't aware of the importance of them.

The developer app camps (<http://devcamps.ms>) that were developed have a number of sessions that walk through the core fundamentals of what a Windows application is and the presentation on the 8 traits of a great app serves as a great introduction to approaching the app design process (also see Pillar 3: Feature Richness for more information on quality app characteristics).

During the functional design of the app, flag the capabilities the developer (even if that's you) will need to implement in the manifest file. If any of the capabilities selected require a privacy policy (such as Internet), include the privacy policy definition in the design process.

In addition to upfront design of the application, as well as the user experience needing to be implemented in a great way, this pillar of quality also includes making sure the user sees the application in the best possible light, from the very first interaction.

App Presentation in the Store

This means the publishing of the application needs to be approached with the same level of care and diligence as creating the app itself. The Windows Phone app quality guidance document highlights

aspects of the publishing process that are crucial to presenting an app as a high quality app. These traits are also pertinent for the Windows Store. These include:

- Use a unique name that describes what the app does
- Avoid “techspeak” such as underscores or not using spaces
- Create a great icon to invite the potential user into the store entry and subsequently download and use the app
- Write a compelling and information description for the app
- Take great screenshots to highlight the app’s characteristics.

Microsoft is running a series of Design Camps (designcamps.ms) – look for local camps to attend so you can learn and build high quality apps from the ground up.

Resources

- Windows Phone app quality guidance document highlights a number of aspects in publishing an app to the store including advice on app titles, tiles, descriptions and supporting materials - <http://aka.ms/AQ016>
- <http://design.windows.com> is the main launching point for app design, from the initial planning phase of what the app should do, to how to implement features well.
- Training materials:
 - Windows 8 Dev Camp content – in particular, check out the first two presentations introduce concepts important to approaching app design in a high quality fashion including detailing the 8 traits of a Windows Store application - <http://aka.ms/AQ001>.
 - Windows Designer Camp content – includes 1 and 3 day camp layouts with learning content for designers and developers wanting to understand design. <http://designcamps.ms>
 - Windows 8 HTML5 Jump Start content on Channel 9 – <http://aka.ms/AQ018>
 - Windows Phone 8 Jump Start content on Channel 9 - <http://aka.ms/AQ017>
 - Windows Phone design boot camp - <http://aka.ms/AQ002>
 - Windows Phone Patterns and Practices - <http://aka.ms/AQ003>

Action Points

- **DO** find your local Design Camps and attend!
- **DO** check out the Windows 8 Developer Camp content
- **DO** look for online training sessions for User Experience and Design

Pillar 3: Feature Richness

Essential to a great quality app on the Microsoft platform is implementation of a rich set of features, including those that differentiate Windows from other platforms. Feature richness doesn't mean to make an app that does many things, like a Swiss Army Knife. Having an app that is rich with features means an app that serves its intended purpose but does it with style, and implements operating system features such as exploiting touch, automation and contracts, live tiles and notifications and more.

Another good way of looking for features that make an app great is to consider whether it provides benefit above and beyond what could be done on a web site. If the app is really just a web site with no additional or unique features, reconsider whether it should be created as is.

Windows 8 Features

Quality apps implement key features of Windows 8. A number of these are covered in the 8 traits of a Windows Store application presentation in the developer camp material and in more depth in the design camp content.

It's required to implement the touch interface, but for an app to be considered high quality in the store, it must also implement mouse and keyboard interfaces in a logical way. So, while the commonly heard mantra goes "implement touch first, get mouse and keyboard for free", you should analyze your app to make sure the way the mouse and keyboard interact with the app features makes sense, and designing to include mouse support during the initial planning phases should not be ignored

Apps should take advantage of the App Bar, in both Windows and Windows Phone. This means implementing commands in the correct app bar and appropriate app bar location. For instance, global commands are normally placed in the left side of the bottom app bar, while contextual commands go on the right. On Windows, navigation commands are typically in the top app bar, as shown in both Internet Explorer and the Windows Store app. Keep in mind that the real estate in the Windows Phone app bar is limited and decisions need to be made about which commands should appear in the visible bar.

Implement the appropriate contracts for the app:

- The Search charm/contract allows the app data to be searched, even from outside of the app itself. This brings part of the app's value to the user, even when they're using another app.
- The Share charm/contract can be implemented in both directions – the app can be a Share Source, sharing its data out through the operating system so that other apps can consume it; or it can be a Share Target, which can receive data through the operating system from apps that share their information.
- The File Picker gives the app the ability to use a system-provided interface to select a file or files to consume in the app. It also allows an app to be the source or destination for files -- the SkyDrive app, for example.
- The Settings charm should be utilized to provide settings for the app, as well as information about the app itself, including privacy policy links if the app requires it.
- Please note that not all capabilities are enabled by default when you create an app project. Visual Studio 2012 provides a great interface for enabling and disabling capabilities. Make sure you only select those capabilities which you intend your app to use.

Windows Phone Features

The Microsoft Windows Phone design team has taken great care in the look and feel of the phone. The new style design language is intended to be clear, concise and provide appropriate and relevant information for the user at all times. Your app should aim to adopt the same philosophy so that users will appreciate the consistency with the underlying operating system and familiar, expected interfaces will reassure them that your app can be used effectively.

All high quality apps on any smart device exist and strive to excel on two areas:

1. Present information
2. Collect input

However, implementing these two fundamental features can be done with unique features of Windows Phone. Consider the various controls, sensors, visual identity and the wealth of information around guidelines for all of these as well as motion, lock screen and the overall app design process.

Windows Phone hardware comes with a consistent chassis design with multiple resolutions, capacitive touch screens, a number of sensors including accelerometer, GPS, compass, light and proximity sensors, digital camera, NFC, Bluetooth, DirectX acceleration, and in some cases, Wi-Fi Direct.

From a software perspective, Windows Phone brings some great features to the end-user including start and lock screens, hubs such as People, Pictures, Media, Games, Wallet integration, In-App purchases. Combining these two sets of features in a creative way, with unique value and an immersive user experience is the aim for a quality app.

The customizable live tile is one standout feature of Windows Phone, and now with Windows Phone 8, this ability to present information to your user without even opening the app has been extended to a customizable lock screen too.

Live tile designs are created from a collection of text, images and counter or icon elements. Make sure you use only those elements that make sense for your application. When designing your live tiles, consider what information you may want to share on the different sized tiles.

Use notifications intelligently. The most efficient (and battery friendly!) way of sending updates from a server to your app is through push notifications.

Here are some other simple and easy to follow tips for helping your app be of great quality:

- Screen real estate is always at a premium on smart phones. You can hide the System Tray when not in use by implementing the `changeSystemTrayVisibility`.
- Images in the app need should be made available to cater to various display resolutions: WVGA, WXGA, 720P, 1080P. For example if the device resolution is 720p make sure that the Resolution Helper class returns the appropriate image path (in this case, `ImagePath720p`), if it is WVGA it should be WVGA image path and it should be WXGA in all other cases.
- When an app uses Windows Phone device features like geolocation the App should first make decisions on orientation, accuracy and movement sensitivity for slow activities such as walking or the opposite for fast activities such as driving. For example:
 - The app can set the accuracy of the location tracking to 50 meters, and configure it to raise a `PositionChanged` event if the user moves more than 10 meters.

- When retrieving the location, the app can set a timeout of 10 seconds, and a maximum acceptable age of cached location data of 5 seconds to ensure you get data in an appropriate timeframe.
- Windows Phone 8 Lenses are apps that take advantages of the high resolution camera present on Windows Phone devices. The platform further enriches the user experience by allowing captured image to be processed in real time (for example, adding a watermark).
- Enabling voice commands is a great way to engage with the user. However, voice command can fail if the user declines your app access, so build the appropriate exception handling mechanism.
- Windows Phone devices can only show one active app at a time. Apps that are not currently executing go into suspended state and resume back to the original state once the user activates them. Windows Phone does take care of this by itself, but sometimes the app needs to resume by itself in such cases make sure to use the state the dictionary to achieve this. Managing application state provides a consistent and smooth experience to app use even when the app resumes or suspends.
- Web services provide the standard gateway to facilitate access to data over the Internet and the Windows Phone 8 SDK provides classes to work with remote web services. The SDK also has classes that allow an app to get information about the network status of the device it is running on so you can properly handle the situation when the network state is problematic.
- The Windows Phone 8 background transfer API provides the ability to queue, execute and monitor multiple uploads and downloads over HTTP while users continue to use the app features in the foreground.
- Fast app resume allows users to re-launch apps more quickly, because instead of launching a new instance, the system resumes the suspended instance if available.

Automation

Apps should automate as much of the tasks and processes as possible to avoid duplicate effort by the user. Settings can be synchronized easily through the cloud, and user account information should be stored and made accessible through the Settings charm without getting in the way of the normal use of the app so the user can focus on what's important – the content.

Apps should have a great live tile to provide information to the user even without having to open the app itself. Live tiles are more than just a static icon – they can include data such as the next appointment on the calendar, or a preview of the latest new article from a website. Apps can provide the ability to produce secondary tiles that deep link directly to specific parts of the app itself, providing even more value to the user.

Notifications are another good way for an app to provide information to the user without the app needing to be open, but should be used intelligently – not overused, and with the understanding that a notification is transient and easily dismissed.

Both notifications and live tiles can be automatically updated through the cloud, with Windows Azure and Windows Notification Services.

Top 20 Features

If you're thinking through your app's design and want to implement features that will differentiate it from others, or take advantage of some of the unique aspects of Windows 8 or Windows Phone, you might wonder what components or services you could implement Here's a

summary of 20 great features that can be implemented in an app to make it exceptional. **In no particular order:**

1. A great live tile.
2. Implementing secondary tiles to deep link to parts of the app.
3. Notifications updated through Windows Notification Service (WNS) and Windows Azure.
4. Search Source Contract.
5. Search Target Contract.
6. Share Contract.
7. App Bar.
8. Settings.
9. Using Roaming Settings to synchronize user settings across devices.
10. Storage of user data in the cloud.
11. Maintaining and retrieving communal data (such as a high score list in a game).
12. Gracefully handle suspend, resume and termination of the app.
13. Solid touch, mouse and keyboard support.
14. Implementing snap, fill and portrait modes.
15. Semantic zoom.
16. In-app purchase.
17. Trial version of a paid app.
18. Rate and review pop up after a number of times the app has been opened to encourage users to submit feedback.
19. Utilize File Picker contract.
20. Take advantage of the hardware features of the device: camera, microphone, accelerometer, compass, etc.

Resources

- Developer camp material, particularly the 8 traits of a Windows Store application - <http://aka.ms/AQ001>.
- Camp in a Box hands on labs walk through the creation of a quality app that's rich with differentiating features - <http://aka.ms/AQ001>.
- <http://interop.windows.com> provides Resources for iOS developers wanting to take advantage of the Windows 8 platform and features – <http://aka.ms/AQ004>.
- Windows Phone control, sensor and other design process guidelines – <http://aka.ms/AQ005>.
- How to implement push notifications on Windows Phone – <http://aka.ms/AQ006>.
- Windows Phone launchers and choosers – <http://aka.ms/AQ007>.

Action Points

- **DO** review the 8 traits of a great app content
- **DO** use the Camp in a Box hands on labs so you understand how to implement the core features to differentiate their app.

Pillar 4: Quality Coding

Behind every great app is great code. While an app can look great if its user interface has been designed well, and can have a set of great features to help differentiate it, without solid code under the hood, the quality of the app is almost guaranteed to be lower than it otherwise could be.

Code should be:

- clean and well tested (even in situations where there are network issues or outages)
- designed with scale in mind
- implemented for stored versus shared data usage
- written to avoid app crashing, freezing and blackouts
- created to provide fluid navigation, clarity in text and images
- without any hidden or background activity that's unnecessary and unknown to the user

Above all, you should optimize your code for performance as much as possible. Seamless transitions, fluidity in the navigation, highly responsive feedback and activity based on user actions, and information and data surfaced to the user so they know what's going on if you truly do need to kick off a slower task.

If you want to create a successful Windows or Windows Phone app that users will enjoy, then you have to spend time to improve its quality and performance. Even the best idea for an app can be unsuccessful if it is slow, unresponsive or crashes periodically. Users will stop using it and may even give it negative reviews. Track your app quality from two equally important perspectives:

- The quality of the app itself – features, user interface, navigation flow, etc.
- The quality of the code behind your app.

The development tools that come as part of Visual Studio are helpful here. Not only can you do your own testing, issue tracking and other fundamental analysis of your app, but you can use tools such as the Windows Application Certification Kit (WACK) for Windows Store applications, or the App Monitoring tools to analyze important behavioral aspects of your code such as responsiveness or start up time.

There are reports on network latency, bandwidth consumption and even potential battery drain that you can assess before you even get your app on a physical device.

App profiling highlights the quality of your app's execution model including memory usage. This can expose quality issues or even design flaws that you can address before your customers even have the chance of giving you feedback.

Windows Phone 8

Windows Phone 8 introduces some new APIs which are different from the earlier version of the Windows Phone SDK. Make sure you look at these changes, particularly those that will help you optimize your app's performance. Review the CLR and .NET Framework class library, look at background file transfer optimizations and even look at quirks mode.

Follow these general ground rules when upgrading your app to Windows Phone 8

- Create a backup of your existing WP project especially if you are targeting both WP7.x and WP8. Ensure that the AppManifest.XAML is not in read only mode.
- To upgrade your WP7.x app to WP8 use the Upgrade command of Visual Studio Solution explorer.
- If you have an upgraded project that has dependencies on other WP7.x projects please ensure to upgrade those projects too.
If you are using any APIs new to WP8 OS make sure to add references.

Resources

- The MCSD for Windows Store Apps is available – <http://aka.ms/AQ008>
- Windows Store apps Support for developers – <http://aka.ms/AQ009>
- Resolving certification errors – <http://aka.ms/AQ010>
- Windows 8 app Certification Requirements – <http://aka.ms/AQ011>
- Windows Phone 8 API considerations – <http://aka.ms/AQ012>
- Upgrading to Windows Phone 8 – <http://aka.ms/AQ013>

Action Points

- **DO** train for and sit the MCSD certification exams for Windows Store Apps in either HTML5 or C#.
- **DO** review the app certification requirements and ensure you test your app against all requirements before submitting to the store.
- **DO** use the Windows Application Certification Kit (WACK) to run automated tests on your application before submitting to store.
- **DO** use the resources for apps that fail certification.
- **DO** attend dev camps to hone your development skills.

Pillar 5: Designed for multiple devices/contexts

One of the benefits of building for the Microsoft platform is that a student can target multiple devices, whether they be laptops, tablets or even mobile devices.

Multiple Devices

There is a lot of common ground that can be covered when developing an app for Windows and Windows Phone. A significant portion of the XAML control set is common across both, and using a Model-View-View Model (known as MVVM) application structure separates the data model and business logic from the user view, allowing for a large amount of potential code reuse. Review the Build for Both series to help understand how to implement and program with MVVM.

Visual Studio 2012 also provides a Portable Class Library project that allows the developer to work with a subset of the .NET Framework and API set that works across all platform options. This provides the ability to have a single codebase that works across Windows, Silverlight, Windows Phone and Xbox.

Developers can also use the classic conditional compilation options to include code that is targeting a specific platform so that the majority of their code can be used as common across their chosen targets.

Using the inheritance features of the languages supported for Windows Store applications allows the developer to create a base class with common code that can be inherited by platform-specific subclasses, again allowing the developer to target different platforms with a minimum of duplication.

Microsoft has developed a four part series entitled Build for Both that covers critical topics for approaching multiple device development, including an easy introduction to MVVM.

When developing for multiple devices, you can leverage Windows Azure and Windows Azure Mobile Services (WAMS) to share data across the different platforms, and using the one set of data model code to maximize reuse in the data design.

Multiple Contexts

Within the Windows Store application model, you should ensure you are designing your app for all of the different contexts it can be in, including:

- Different screen resolutions, from 1366x768 through to 2048x1536
- Snapped view
- Filled view (minimum resolution of 1024x768)
- Portrait view

Visual Studio 2013 comes with a Visual Simulator that allows you to run your app in multiple resolutions and orientations. This is a great tool for most programmers who only have limited access to different resolutions, or have a device that doesn't automatically transition between landscape and portrait.

Resources

- The Build for Both series for multiple device development and learning MVVM. Available on Channel 9 (<http://aka.ms/AQ019>) or Microsoft Virtual Academy (<http://aka.ms/AQ020>).
- Camp in a Box Hands on Labs walk through the implementation of Snap, Fill and Portrait views in both HTML5/JS and XAML/C# - <http://aka.ms/AQ001>.

- App Camp content includes presentations on different views, and how to handle high definition screens - <http://aka.ms/AQ001>.

Action Points

- **DO** investigate how to build for both Windows and Windows Phone with common code, portable class libraries, inheritance and subclassing.
- **DO** learn MVVM and how to use it for Windows applications.
- **DO** understand different view states in Windows 8.
- **DO** understand to use Windows Azure and Windows Azure Mobile Services (WAMS) to store data online to help share across devices.

Pillar 6: Cloud Integration

Great Modern apps invariably leverage the always online nature of the world today. When developing a great quality app for Windows 8 or Windows Phone 8, you should consider integrating cloud services. Adding cloud services to your app can enable much more compelling features for your users just like the professionally developed apps you've seen in the app stores.

A cloud service can provide fast access to centrally stored data if you want to enable data sharing between users like a high score board for example. A cloud services can make authenticating users much, much easier if you want to store data associated with your users like their personal best scores. Even better cloud services let your users not have to remember yet another password for your app and login with an existing authentication service like Microsoft Account, Facebook or Twitter. A cloud service can also let your app keep in touch with your users with what we call push messages, even when they aren't using it at that time to say let a player know it's their turn next. That's just a few benefits of integrating a cloud service to your app.

User Settings

User settings can be stored locally or roamed to the cloud. The latter is as easy to implement as the former, but has the advantage of having the settings for the application saved to the cloud and then to any device the user runs the app on. The roamed settings are attached to the user's Microsoft Account. The developer doesn't need to do anything else in addition other than use the appropriate APIs to save the roamed settings.

User settings go hand in hand with Process Lifecycle Management, allowing an app to properly suspend, resume and re-launch after termination. Imagine how nice that is when you are switching computers all the time especially in a class room setting. Your app's roamed settings allows a user to terminate an app on one device and start it on another, and resume right where they left off.

Data

Going another step down the path of online data, developers can also save and retrieve data that's shared across all instances of the application. For example, a game might have a high score list that can be stored in a Windows Azure SQL Database, and retrieved whenever any user plays the game. Or the user experience of the app could greatly benefit from data that is contributed (what they call crowdsourcing) by all users like the most popular or most read article for example.

With Windows Azure Mobile Services storing an app's data centrally in the cloud is very simple. The app developer simply creates tables in a Mobile Service and then with one line of code in the app creates, reads, updates or deletes objects (also known as CRUD) without having to write the REST APIs (calling a cloud service using HTTP, for example - GET <cloudapp-url>/player?id=1) or SQL code if you don't want to. Great for getting started with cloud integration and allows you the app developer to concentrate on the real goal, writing a great quality Windows 8 app or Windows Phone app.

Users

Developers can also utilize Windows Azure to store user profile data or data that only that user should be able to read or update.

To implement this the app will need to authenticate users to access the data it's storing for them. Windows Azure Mobile Services provides a very easy way to allow users to login into an app using the

Microsoft Account (formerly Live ID) provider or even use Facebook, Twitter or Google identity providers. That way the user doesn't have to remember yet another password and you're not reinventing the wheel with creating your own user profile and authentication code. Once user authentication is in place it's a very simple step to attach the user's identity to data items that are deemed part of their profile.

Push

Another quality feature that cloud service integration can bring to an app is notifying the user when some data has changed or an event occurs on the back end. Like when a user beats another user's high score or it's a user's turn next, for example.

With the advent of modern mobile operating systems, including Windows 8, apps aren't given endless processing time to keep a connection open to a remote service or constantly poll for updates when they aren't the foreground app. The O/S does this mainly to save battery. These apps are frozen and not executing at all when not in use, however the user still expects to be notified when new data arrives. Consider email apps: you'd want to know when new mail arrives without having to constantly check the app. Windows 8 and Windows Phone have really unique ways to keep the user up to date. One of those ways, as mentioned previously, is the app's live tile.

To implement this Windows Azure Mobile Services provides a really simple way to send user push notifications and update the app's live tile using trigger scripts when data is inserted, updated or deleted.

Cloud Services

We've been talking a lot about Windows Azure Mobile Services in the section. That's because it's an excellent way to get started in implementing features that only come from cloud services for building a good quality modern app. Having said that Windows Azure has a lot more to offer with more advanced and more flexible services.

Windows Azure is a very powerful platform for building really, really big scale systems. Some organizations use Windows Azure to host virtual server machines (or VMs) that help them migrate IT infrastructure to the cloud. For example, an organization might migrate email servers, file share servers or legacy web applications to the cloud with Windows Azure VMs.

However, when building cloud services for your application, using a VM shouldn't always be the default choice. Instead, think of the Windows Azure platform as building a system from individual components or building blocks like cloud services, queues, SQL databases, no-SQL data, blob storage and service buses to name a few. Doing this enables you to build a distributed and scalable system that abstracts the layers and services away from a web server or operating system. Basically, focus on designing your cloud system rather than how to set up or manage IT infrastructure.

Also designing a system with these individual components means scale is designed in from the beginning and all parts of the system can be scaled independently from one another after you've deployed them. Consider a video compression cloud service and some video file storage separated by a queue. This way our building blocks are decoupled from each other and easily can scale up and down with demand for uploading videos from a video app. This change can be set up to be automatic and elastic or manual, by simply changing a configuration setting.

Resources

- Windows Azure Getting Started Guide <http://aka.ms/AQ021>
- Windows Azure Cloud Services – <http://aka.ms/AQ015>
- Windows Azure Mobile Services Dev Center (Tutorials and Videos) – <http://aka.ms/AQ014>

Action Points

- **DO** look for events that teach how to use Windows Azure Mobile Services + Windows 8 and Windows Phone.
- **DO** consider using Windows Azure Mobile Services when building Windows 8 and Windows Phone apps
- **DO** look at the Windows Azure Mobile Services support for iOS and Android developers if that's your background.