# The Fast Lane for App Certification: Windows Store Certification Tips

When you sign up for a developer account with the Windows Store, your goal is most certainly to submit an app, get it certified, and get it out to your customers. In this post I’d like to share with you a few of the most important tips to moving this process along as efficiently as possible.

## #1: Make sure your app fulfills basic certification requirements

No matter what stage your apps are in, be sure to regularly review the [App certification requirements for the Windows Store](http://msdn.microsoft.com/en-us/library/windows/apps/hh694083.aspx), which can and do change. We’re now on version 5 of the requirements, which were last revised in January 2014 to relax a number of policies that were previously more restrictive. In other words, by reviewing the requirements you may find that some app ideas you couldn’t implement before are now possible.

When planning apps for the Windows Store, keep in mind the core principles behind certification: apps need to provide value to customers, they need to behave predictably, they need to put customers in control, and they need to be appropriate for a global audience. If you design your apps around these principles and review the specific requirements, you’re already a long way toward certification.

## #2: Create your Store account early

Apps cannot be certified until you have completed [setup of your Store account](http://msdn.microsoft.com/en-us/library/windows/apps/jj863494.aspx). It can take a couple of days to set up an individual account, and longer for a company account. An individual account requires credit card verification as proof of identity, which is done my making a very small charge to your code where a 3-digit code will appear on your statement. You’ll need to enter this code in the Store dashboard to verify your identity.

Company accounts take longer because there are additional verification steps. This is necessary because company accounts can take advantage of additional privileges in apps, such as using shared user certificates, enterprise authentication, and the Documents library.

Also remember to complete the necessary tax forms for your country of origin, which is a requirement for paid apps and in-app purchases. Without completed tax forms, certified apps will not be released and you won’t receive any revenue.

## #3: Reserve appropriate app names early

The name that you’ll be able to use in an app certainly affects many aspects of your design and branding, so reserve those names early in your development process, including any localized names you need in regional markets. You might find, of course, that others have already reserved the name you have in mind, so doing this early gives you time to find new names you can work with.

Verify also that you have the rights to use the name you choose. Be especially careful when using other company names, such as when you’re employing their services within your app. Each company will have their own policy on how their trademarks can be used by third parties, and if you infringe on their trademarks they can ask that your app be removed from the Store altogether.

For more guidance, see [What to name your app](http://msdn.microsoft.com/en-us/library/windows/apps/jj657967.aspx).

## #4: Submit apps that customers want to use

When building apps, think about how each one contributes to the strength of your overall portfolio, and the quality of the overall Windows Store. We’ve found that customers reward creative and useful apps, and have little interest in unfinished apps or the results of your early experimentation. We especially discourage your from submitting variations of the samples we provide in the Windows SDK. Those samples exist to educate *you* on the use of the Windows APIs and provide code snippets that you can use for specific capabilities. However, they are not intended for consumers nor are they intended to provide starting points for a consumer-facing app.

In addition, build apps that are more than just websites. Although it’s possible to simply wrap a website with a Webview control, an app should provide a richer experience than the website itself. For example, apps should integrate with platform capabilities like the Share charm, live tiles, sensors, touch input, and so forth. Your customers should feel that there is definite value in acquiring your app versus just visiting your website in a browser.

## #5: Declare only those capabilities that the app truly needs

A number of Windows Runtime APIs are protected by capabilities that you declare in the app manifest. It’s important that you declare only those that the app needs at present, because you can always add capabilities with future app updates when you’re using their related APIs.

The reason for this is that your declared capabilities are shown directly to consumers on your app’s page in the Windows Store. Those capabilities need to make sense to consumers, especially those that have privacy concerns such as the microphone, camera, and geolocation. If there is any doubt whether your needs for a capability is obvious, explain your use of it in the app’s description.

Declaring capabilities for the user’s Pictures, Videos, and Music libraries is typically necessary only if your app presents a full gallery view of library contents. If your need is to only open or save individual files to those locations, it’s better to use the file pickers for that purpose, as they allow the user to select any location in those libraries or anywhere else on their file system and cloud storage providers.

Also be aware that there are three special-user capabilities that guarantee a slower certification process. These are the Documents Library, Shared User Certificates, and Enterprise Authentication. These may be used only by apps submitted with a company account (not an individual account). They are also reserved for specific scenarios and require written justifications. For more, see [App capability declarations](http://msdn.microsoft.com/en-us/library/windows/apps/hh464936.aspx).

Avoid thinking, then, to declare the Documents Library capability just for storing configuration files (user your app data folders instead) or doing anything outside the allowed scenarios, because attempting to do so will cause rejection. And if all you need is to let users save and open files in that library, just use the file pickers instead because through them users can choose any location they want, including Documents.

## #6: Include required privacy statements

A privacy statement is required for *all* network-enabled apps, which includes all apps that declare the *Internet (Client)* capability in the manifest. This capability is turned on by default in the Visual Studio project templates, so remember to uncheck that box in the manifest editor if you don’t need it.

Your privacy statement must be posted on a publically-accessible URI. You then link to that URI in both your app’s description page an in your Settings panel. For your description page, you provide the URI when filling out your app’s details on the Store dashboard. For Settings, you should have a top-level *Privacy Statement* command that launches that URI.

[Requirement 4.1](http://msdn.microsoft.com/en-US/library/windows/apps/hh694083#acr_4_1_1) of the App certification requirements for the Windows Store provides the specific of what your privacy statement must contain. Failure to provide a privacy statement will cause your app to fail certification.

## #7: Be careful when soliciting user ratings

Because user ratings and reviews are so visible on your app’s page in the Windows Store, many developers actively invite their customers to leave their feedback. Obviously, good ratings are important, but apps *cannot* offer rewards for positive ratings or reviews. You must allow consumers to have their own opinions and state them honestly. After all, genuine feedback is how you understand where to improve your app.

In fact, it’s not even possible to programmatically determine what feedback was left, so there isn’t a way to trigger such a reward in the first place. The most you can do is send a user directly to your app’s ratings and review section by launching the URI ms-windows-store:REVIEW?name=[publisher\_name]where the publisher name is what comes back from [*Windows.ApplicationModel.Package.­current.id.publisher*](http://msdn.microsoft.com/en-us/library/windows/apps/windows.applicationmodel.packageid.publisher.aspx). In short, when inviting feedback, avoid using any words that suggest a reward, as that will be grounds for rejection.

## #8: Test your app thoroughly

This one is simple. If your app isn’t functional, it violates [Requirement 1.2](http://msdn.microsoft.com/en-US/library/windows/apps/hh694083#acr_1_2) and will fail certification. Full functionality, however, requires that you test for a wider variety of conditions than you might encounter in your development work. Up-front testing is also important because any updates you make to your app to fix bugs requires that the app goes through the full certification process again.

First, be sure to test on a variety of hardware with different capabilities. This includes touch-only devices, touch-enabled devices, and keyboard+mouse only devices. Test also on x86, x84, and ARM systems—never assume that just because your app works on one that it will work on the others.

Test your app on a variety of screen sizes and orientations. For this purposes you can use real hardware that’s available to you, but also employ the Visual Studio simulator which you can configure for a set of canonical screen sizes and scaling factors. Then test your app when the device is disconnected from the Internet and/or on a metered connection. We take connectivity so much for granted that we can easily forget about all the customer scenarios where connectivity isn’t available!

Remember also to switch from using the *CurrentAppSimulator* object, which you use during development to work with the Store APIs, to the *CurrentApp* object. If you forget, you’ll fail certification right away but will have to resubmit anew. Also be aware that some Store APIs will throw exceptions when the device is offline and licensing information isn’t available. Be sure to catch those exceptions, especially for in-app purchases, because you’ll typically never see such exceptions when using the *CurrentAppSimulator*.

<TODO: code snippet?>

## #9: WACK your app (run the Windows App Certification Kit)

When you submit it for certification, the first thing that happens is that the Store runs it through the [Windows App Certification Kit](http://msdn.microsoft.com/en-US/windows/apps/bg127575) or WACK. This tool catches many common problems and issues, any one of which will cause your app to be rejected right away. You can save yourself that trouble by running the tool yourself, which is included with the Windows SDK. Just search for “Cert Kit” from the Start screen and you’ll find it.

The WACK subjects the app to a number of tests, including loading time, suspending time, responsiveness, and so forth, and provides you with a full report at the end of the process. You must address any issues that the report identifies before you’ll pass certification.

Ideally, run the WACK periodically during your development process rather than waiting until the very end. It will be much easier to fix issues early on, and running the WACK will give you a certain level of regression testing as well.

Note that you can also choose to run only specific tests in the WACK. A full test can take 10-15 minutes during which time you should not interact with the app or the machine as a whole. If you’re working on a specific issue that a previous WACK session identified, then, you can shortcut the process by rerunning only that one test.

## #10: Maintain truth in advertising

When filling out all the descriptive information for your app’s page in the Store—including your description, feature list, and screen shots—be honest about the real customer experience of your app. Make sure that whoever is writing this marketing copy is fully attuned to what the app actually does—don’t list features, for instance, that might be implemented in the future, and don’t show unimplemented features in screen shots.

Be sure to also disclose geographical restrictions, that is, identify features that might not be available in some markets. For example, legal requirements might limit certain data that you can supply from your backend services, thereby limiting the functionality of the app itself.

Similarly, disclose any hardware requirements such as required peripherals, that are necessary for the user to experience all of your features and capabilities.

You must also disclose the extent of localization. Ideally, you should provide localized versions of your descriptive text, features, search keywords, screenshots, privacy statements, and so forth for each language you declare in your app manifest. If you have any partial localization, however, be sure to disclose it. This includes disclaiming that content the app retrieve from its services might not be localized for the user’s language.

## #11: Select an appropriate age rating for your app

Your age rating represents the kinds of content that users can expect to see in you app, and the suitability of that content for children, especially. Your age rating also determines whether your app appears in the Windows Store when age filters are in effect.

The [Age ratings and rating boards](http://msdn.microsoft.com/en-us/library/windows/apps/hh694080.aspx) topic in the documentation fully describes what kinds of content are appropriate for which age rating. Some countries and regions also require a rating from a ratings board, most typically for games. Typically this means going through an online self-rating process, but some countries require that you submit your app for review. The topic above has links to all the ratings boards.

Be especially careful with apps that provide uncontrolled sharing any kind of personal information or access social networks, as this requires that the app is rated at 12+ or higher. “Controlled” sharing means that the app includes parental controls or requires parental permission to use sharing features, which must be clearly identified and explained in your Notes to Testers (see #12 below).

If you app acquires unfiltered content from an online service, especially one that allows customers to upload their own, your app’s age rating must match whatever content controls are enforced by that service. Otherwise it’s possible that content that’s inappropriate for your age rating can appear in the app and cause it to fail certification.

Finally, regardless of your app’s age rating, your descriptive content for the Windows Store must be suitable for the 12+ rating. For example, a game with a 16+ rating due to graphic violence cannot include screenshots that depict that violence because they’re not allowable with the 12+ rating.

## #12: Use the Notes to Testers field

The final step in submitting your app is providing any notes to the testers that they need to properly exercise your app. Manual testing by a human being is the last phase of certification, and that person needs to know how to best use and experience your app, how to access any hidden features, and what test accounts they can use where logins are required. For app updates, also explain what has changed since your last submission. This is especially helpful if you’ve not actually changed any code—perhaps you’ve just updated listing details or added a new set of localized resources.

And remember that you’re talking to a real person through these notes, so be courteous, polite, and kind.

For more, see [Providing info to certification testers](http://msdn.microsoft.com/en-US/library/windows/apps/hh694071).

## Conclusion

Developers often make the mistake of concentrating too much on the internal workings of their app’s code and too little on the larger process of making that app available to customers. The Windows Store certification process is designed to make sure that the apps that get listed in the Store are ready for those customers. By understanding the process ahead of time, you’ll be able to avoid a number of mistakes that will cause the app to be rejected from certification or otherwise delayed.

For more help on certification issues, see [Avoiding common certification failures](http://msdn.microsoft.com/en-us/library/windows/apps/jj657968.aspx) and [Resolving certification errors](http://msdn.microsoft.com/en-us/library/windows/apps/hh921583.aspx) in the documentation, or visit the [Publishing Windows Store Apps forum](http://social.msdn.microsoft.com/Forums/windowsapps/en-us/home?forum=windowsstore) on MSDN.

Gus Salloum, Program Manager, Windows Store