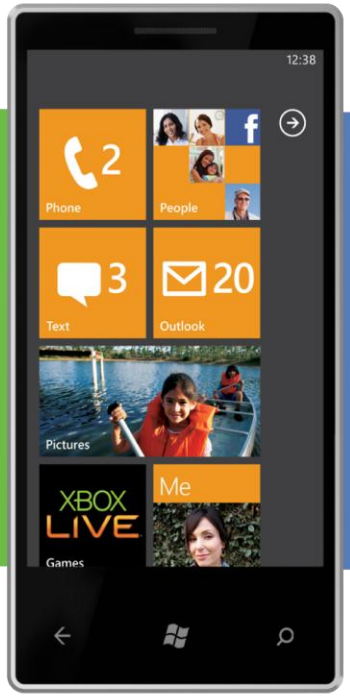
Windows Phone

Windows Phone 7 Application Certification Requirements



7

Microsoft Corporation

October 2010

Version 1.4

Applies To: Windows® Phone 7

© Microsoft Corporation. All Rights Reserved. This document is subject to updates.

Information in this document, including URL and other Internet Web site references, is subject to change without notice. Unless otherwise noted, the companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in examples herein are fictitious. No association with any real company, organization, product, domain name, e-mail address, logo, person, place, or event is intended or should be inferred. Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

© 2010 Microsoft Corporation. All rights reserved.

Microsoft Authenticode, Expression, MSDN, Visual Studio, Windows, Windows Phone, and XNA are trademarks of the Microsoft group of companies.

All other trademarks are property of their respective owners.

Contents

Windows Phone 7 Application Certification Requirements 4
1.0 Program Overview 4
1.1 What You Need to Know About the Submission and Certification Process 4
1.1.1 Process Outline 6
1.1.2 Code Signing 6
2.0 Application Policies 7
3.0 Content Policies 10
4.0 Application Submission Requirements 13
4.1 Installation Package Validation 13
4.2 Application Code Validation 14
4.3 Phone Capabilities Detection 14
4.4 Language Validation 15
4.5 Windows Phone Marketplace Iconography 16
4.6 Application Screenshot 16
5.0 Application Certification Requirements 17
5.1 Application Reliability 17
5.2 Performance and Resource Management 17
5.3 Phone Functionality 19
5.4 Security 19
5.5 Content Validation 20
5.6 Technical Support Information 20
6.0 Additional Requirements for Specific Application Types 21
6.1 Location Aware Application 21
6.2 Push Notifications Application 21
6.3 Applications Running under a Locked Screen 21
6.4 Music + Videos Hub Application 22
6.5 Applications that Play Media 23
6.6 Photo Extras 23
6.7 Photo Sharing Applications 24
7.0 Change History 25

© Microsoft Corporation. All Rights Reserved. This document is subject to updates.

Windows Phone 7 Application Certification Requirements

The purpose of this document is to provide the policies and technical requirements that a Windows Phone 7 application or game must meet to pass certification and to be eligible for listing in Windows Phone Marketplace.

1.0 Program Overview

A core principle that is applied in designing the certification process is that each individual policy or requirement is clear, objective, and testable. This transparency is designed to help developers easily design and test applications to meet these requirements.

The following list shows the pillars of the certification program:

1. Applications are reliable.
2. Applications make efficient use of resources.
3. Applications do not interfere with the phone functionality.
4. Applications are free of malicious software.

1.1 What You Need to Know About the Submission and Certification Process

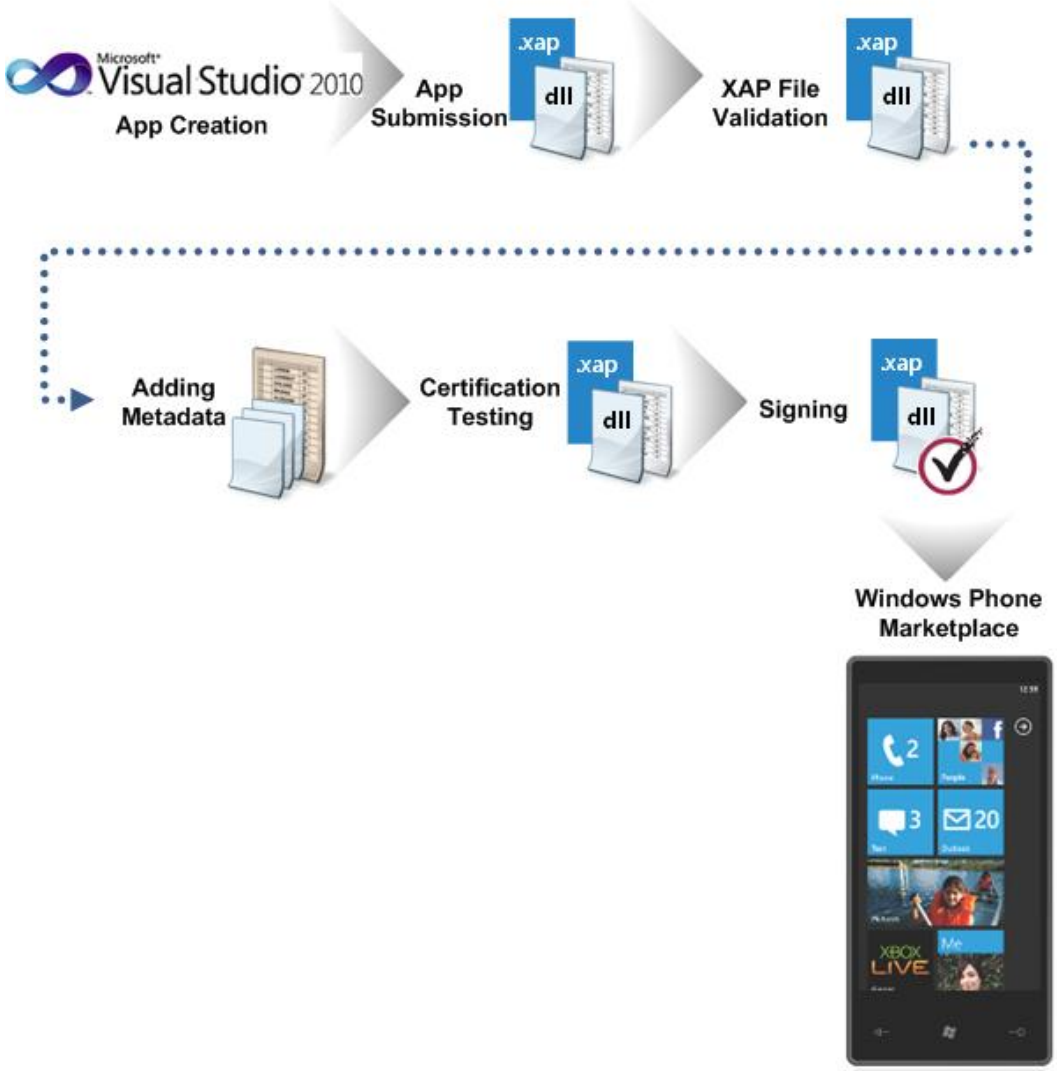
When your application is ready for publication, it must go through the certification process before it is eligible for listing in Windows Phone Marketplace. Your application does not have to be signed before submission.

The certification process involves static validation and automated testing of your application to verify that it meets all the policies and requirements. The following list shows the five major categories of policies and requirements:

- 2.0 Application Policies
- 3.0 Content Policies
- 4.0 Application Submission Requirements
- 5.0 Application Certification Requirements
- 6.0 Additional Requirements for Specific Application Types

© Microsoft Corporation. All Rights Reserved. This document is subject to updates.

The following is a simplified illustration of the submission and certification process.



© Microsoft Corporation. All Rights Reserved. This document is subject to updates.

1.1.1 Process Outline

The following is a brief outline of the submission and certification process:

1. Sign in to your account in the Windows Phone Developer Portal.
2. Create a new application submission.
3. Upload the application XAP file.
4. Enter the metadata for the application, such as title, description, category, and iconography.
5. Select the distribution countries and pricing.
6. The XAP file is validated while you are entering metadata.
7. If the XAP file validation succeeds, the submission process continues to Step 8; otherwise, the process terminates and you get a notification.
8. Select the option to publish immediately after passing the certification process or to wait until you decide to publish.
9. The XAP file is repackaged as described in Section 4.1.2.
10. The repackaged XAP file is deployed to a phone for the certification testing. Certification involves the automated and manual verification of the meeting of the requirements that are described in Sections 2, 3, 4, 5, and 6.
11. If the application meets all the requirements, the repackaged XAP and assembly files are signed, and the application is eligible for publication according to the option selected in Step 8.

    If the application fails one or more of the requirements, you get a failure report and the application is not published.

Note:
When you submit an application update for certification, it goes through the same process as the original application.

1.1.2 Code Signing

Code signing occurs automatically once the application has successfully passed the certification testing without any failure. The application and repackaged XAP files are signed with the Authenticode® certificate assigned to you when you registered for App Hub membership. Any signatures in a submitted application or XAP files will be replaced and are not retained.

Note:
All applications must be signed with the Microsoft issued Authenticode certificate before they can be installed and run on commercially available Windows Phones.

© Microsoft Corporation. All Rights Reserved. This document is subject to updates.

2.0 Application Policies

To protect the Windows Phone Marketplace service and users of the service, and to address mobile operator requirements, Microsoft has established the following policies for applications offered for distribution in the Windows Phone Marketplace. Microsoft reserves the right to update these policies as needed.

2.1 Your application must be fully functional when acquired from the Windows Phone Marketplace (except for additional data as permitted below), and may not require the user to pay, within the application experience and outside of Windows Phone Marketplace, to activate, unlock, or extend usage of the application. However, applications that do not appear in the Games Hub or are not commonly or elsewhere classified as a game may enable access to additional content for which payment is made outside of Windows Phone Marketplace if you have a pre-existing billing relationship with the user or you require the user to establish a new billing relationship via a web browser.

2.2 Your application may not sell, link to, or otherwise promote mobile voice plans.

2.3 Your application may not consist of, distribute, link to, or incent users to download, or otherwise promote alternate marketplaces for applications and/or games.

2.4 Your application must not jeopardize the security or functionality of (a) Windows Phone 7 devices or (b) the Windows Phone Marketplace.

2.5 The OTA (over the air) installation file for the application may not exceed 20 MB. Applications in excess of that size will be downloaded via Wi-Fi or through a tethered connection to a PC running the appropriate Microsoft software.

2.6 If your application includes a trial version, the trial version must reasonably represent the functionality and quality of the full application.

2.7 If your application includes or displays advertising, the advertising must comply with the Microsoft Advertising Creative Acceptance Policy Guide (http://advertising.microsoft.com/creative-specs).

2.8 If your application requires the download of a large additional data package (e.g. >50 MB) to enable the application to run as described, the application description must disclose the approximate size of the data package and that additional charges may apply depending on connectivity used to acquire data.

2.9 If your application enables chat, instant messaging, or other person-to-person communication and allows the user to setup or create his or her account or ID from the mobile device, the application must include a mechanism to verify that the user creating the account or ID is at least 13 years old.

2.10 The following requirements apply to applications that receive the location of a user's mobile device using Microsoft Location Service API:

- **2.10.1** Your application must determine location using Microsoft Location Service API.
- **2.10.2** Your application must not override, circumvent, or suppress any Microsoft toast or prompts related to the Location Service API.
- **2.10.3** Your application must not override or circumvent a user's choice to disable location services on the mobile device.
- **2.10.4** Your application must provide in-application settings that allow the user to enable and disable your application's access to and use of location from the Location Service API.

© Microsoft Corporation. All Rights Reserved. This document is subject to updates.

- **2.10.5** If your application publishes or makes available location data obtained from the Location Service API to any other service or other person (including advertising networks), your application must implement a method to obtain opt-in consent. To "implement a method to obtain 'opt-in' consent," the application must (a) first describe how the location information will be used or shared; (b) obtain the user's express permission before publishing the location information as described; and (c) provide a mechanism through which the user can later opt out of having the location information published. Your application must periodically remind users or provide a visual indicator that location data is being sent to any other service or person.
- **2.10.6** Your application must request location and retain and use location data from the Location Service API only as necessary to deliver the location-aware features your application provides to users.
- **2.10.7** Your application must make a privacy policy available to users that informs them about how location data from the Location Service API is used and disclosed and the controls that users have over the use and sharing of location data.
- **2.10.8** You and your application must adopt measures to protect against unauthorized access to, use or disclosure of location data received from the Location Service API.

2.11 If your application publishes a user's personal information from the mobile device to any service or other person, the application must implement a method to obtain "opt-in" consent. "Personal information" means all information or data associated with an identifiable user, including but not limited to the following, whether stored on the mobile device or on a web-based server that is accessible from the mobile device:

- Contacts
- Photos
- Phone number
- SMS or other text communication
- Browsing history

To "implement a method to obtain 'opt-in' consent," the application must (a) first describe how the personal information will be used or shared; (b) obtain the user's express permission before publishing the information as described; and (c) provide a mechanism through which the user can later opt out of having the information published.

2.12 If your application allows users to purchase music content, it must include the Windows Phone music Marketplace (if available) as a purchase option. If the application also allows music content to be purchased from any source other than the Windows Phone music Marketplace, the application must include its own playback functionality for that music content.

© Microsoft Corporation. All Rights Reserved. This document is subject to updates.

2.13 If your application uses the Microsoft Push Notification Service (PNS), the application and the use of the PNS must comply with the following requirements:

- 2.13.1 The application must first describe the notifications to be provided and obtain the user's express permission (opt-in), and must provide a mechanism through which the user can opt out of receiving push notifications. All notifications provided using PNS must be consistent with the description provided to the user and must comply with all applicable 2.0 Application Policies and 3.0 Content Policies.
- 2.13.2 The application and its use of the PNS must not excessively use network capacity or bandwidth of the PNS or otherwise unduly burden a Windows Phone or other Microsoft device or service with excessive push notifications, as determined by Microsoft in its reasonable discretion, and must not harm or interfere with any Microsoft networks or servers or any third party servers or networks connected to the PNS.
- 2.13.3 The PNS may not be used to send notifications that are mission critical or otherwise could affect matters of life or death, including without limitation critical notifications related to a medical device or condition. MICROSOFT EXPRESSLY DISCLAIMS ANY WARRANTIES THAT THE USE OF PNS OR DELIVERY OF PNS NOTIFICATION WILL BE UNINTERRUPTED, ERROR FREE, OR OTHERWISE GUARANTEED TO OCCUR ON A REAL-TIME BASIS.

© Microsoft Corporation. All Rights Reserved. This document is subject to updates.

3.0 Content Policies

To protect the Windows Phone Marketplace service and users of the service, and to address mobile operator requirements, Microsoft has established the following policies for content offered for distribution in the Windows Phone Marketplace. Microsoft reserves the right to update these policies as needed.

If your application is a game that has been rated by an approved ratings board (currently ESRB, PEGI, and USK), you must submit the ratings certificate for the application and provide the appropriate content descriptors. If a game application is submitted with a rating that is equal to or less restrictive than ESRB "T", PEGI "12", or USK "12", it will be presumed to comply with these content policies.

3.1 Licensed Content, Name, Logo & Trademarks

Content allowed where:

- Content and application name are original or licensed.
- Copyrighted content that is used with permission. Use of branded items (logos/trademarks) has been approved by the brand owners.
- If an application depicts any mobile or wired telephone, handheld PDA, or any other data and voice communicator, it must be either generic or a Windows Phone device.
- It is the application provider's responsibility to determine if the application provider has the right to use the chosen name, content, logos, copyright, trademarks, online services & API's.

3.2 Illegal or Contemplates Harm

Content not allowed:

- Any content that is illegal under applicable local law, obscene, or indecent.
- Any content that depicts or encourages harm or violence against a person or animal in the real world.

3.3 Defamatory, Libelous, Slanderous, and Threatening

Content not allowed:

- Any content that is defamatory, libelous, slanderous, or threatening.
- Any content that facilitates or promotes content prohibited by these guidelines.

3.4 Hate Speech or Discriminatory

Content not allowed:

- Any content that advocates discrimination, hatred, or violence based on considerations of race, ethnicity, national origin, language, gender, age, disability, status as a veteran, religion, sexual orientation or expression, or that promotes organizations devoted to that purpose. Such content can include images or text that perpetuates a negative stereotype of a race, gender, sexual preference or religion. We are particularly inclined to act against content where there is evidence that the intent of posting was to harass, threaten, or insult an individual or group on one of these bases.

3.5 Alcohol, Tobacco, Weapons, and Drugs

Content not allowed:

- Any content that facilitates or promotes, whether directly or indirectly, the illegal (under applicable local law) or excessive sale or use of alcohol or tobacco products, drugs, or weapons is not allowed on any section/site, regardless of targeting.

© Microsoft Corporation. All Rights Reserved. This document is subject to updates.

3.6 Adult Related Content

Content not allowed:

- Sex / Nudity – Images that are sexually suggestive or provocative (e.g. sexually provocative touching, bondage, masturbation); provocative images that reveal nipples, genitals, buttocks, or pubic hair.
- Content that a reasonable person would consider to be adult or borderline adult content (images, text, or audio).
- Content that generally falls under the category of pornography.
- Content that depicts or suggests prostitution.
- Content depicting sexual fetishes.
- Content of a sexual nature depicting children or animals.

3.7 Certain Types of Illegal Activity

Content not allowed:

- Any content that facilitates or promotes illegal gambling, illegal adult content and/or pornography, child pornography, bestiality, piracy, illegal online pharmacies, illegal drugs, or criminal or terrorist activities.
- Any content that instructs users how to make bombs or weapons, drugs, or solicits involvement in behavior that is violent or illegal under applicable local law.
- Unauthorized use of another entity's intellectual property, including but not limited to: software, music, art, and other copyrighted, trademarked or patented materials or trade secrets.
- Any content that facilitates or promotes underage drinking, consumption of illegal drugs, or socially irresponsible behavior due to alcohol or drug consumption (e.g., drinking and driving).

3.8 Violence

Content not allowed:

- Realistic or gratuitous violence, including depictions of the following:
    - Decapitation, impaling, blood splatter/blood spurting/blood pooling, or gore
    - Exploding body parts
    - Guns/weapons pointed toward user/audience
    - Strangulation/choking
    - People or creatures on fire
    - Cruelty to animals
    - Audio of humans or animals suffering
- Involuntary or physically-resisted sexual interactions with violent or illicit overtones
- Rape, sexual assault
- Molestation, physical child abuse
- Requests or instructions to injure or otherwise harm a real-world person or group of people
- Glorification of crimes against humanity such as genocide and torture

© Microsoft Corporation. All Rights Reserved. This document is subject to updates.

3.9 Excessive Profanity

Content not allowed:

- Any content with the excessive use of profanity or adult language.

Note:
Microsoft reserves the right to update these content policies as needed to protect the Windows Phone Marketplace service or the users of the service.

© Microsoft Corporation. All Rights Reserved. This document is subject to updates.

4.0 Application Submission Requirements

The following requirements are validated during the submission process. The process involves the checking of the metadata and the validating of the XAP file that you upload. An application that does not meet one or more requirements fails the submission process.

4.1 Installation Package Validation

The assembly and data files must be packaged as an XAP file package. Visual Studio® 2010 Express for Windows Phone generates the necessary XAP package and manifest files.

- **4.1.1 List of Package Requirements**

  The maximum size of the XAP package file is 225 MB.

  The XAP package must contain the following:

  a. A valid Windows Phone application manifest file, named WMAppManifest.xml. For more information, see the Application Manifest File for Windows Phone topic.

  b. The <Title> element in the WMAppManifest.xml file must contain the application title. The <Title> element must not be empty. The **Application title** entered in Step 2 of the submission process to Windows Phone Marketplace and the title displayed on the phone must be the same.

     For more information about setting the application and tile titles, see How to: Create and Modify and Application Tile.

  c. A valid .NET application manifest file, named AppManifest.xml.

  d. The assembly files as specified in the AppManifest.xml file.

  e. The application icon that you want displayed on the phone **app list**. Games must use the application tile image in place of the application icon.

  f. The application tile image that you want displayed when the user pins the application to the quick launch area on the phone **start experience**.

     The following table shows the size and file type requirements for icon and tile images that are included in the XAP package.

Required Icons in the XAP Package	Where Used	Pixels	File Type
Application Icon	In the **app list** on the phone	62 x 62	PNG
Application Tile Image	In the **start experience** on the phone	173 x 173	PNG

© Microsoft Corporation. All Rights Reserved. This document is subject to updates.

- **4.1.2 XAP File Repacking Process**

  When you submit the XAP file to Windows Phone Marketplace, the file is decompressed, validated, and repackaged.

  Repackaging involves the following steps:

  a. The Windows Phone application manifest is provisioned with a product identifier for each application.
  b. The security capabilities are rediscovered and listed in the manifest.
  c. The phone experience hub type is set in the Windows Phone manifest (e.g. Music + Videos Hub).
  d. The Digital Right Management header file is created and named WMAppPRHeader.xml.
  e. The original contents of the package, the updated Windows Phone application manifest, and the Digital Rights Management header file are compressed into a new XAP package.

4.2 Application Code Validation

The following list shows the code requirements:

- **4.2.1** You must develop your application using the documented APIs that are supported on the Windows Phone Application Platform. For more information, see the Class Library Reference for Windows Phone topic.
- **4.2.2** The application must not invoke native code via PInvoke or COM interoperability. If it does, it will fail the certification process.
- **4.2.3** The application must be compiled using retail configuration instead of debug. The application must not contain debugging symbols or output.
- **4.2.4** The application must not redistribute the Windows Phone assemblies. However, you can re-distribute panorama, pivot, and map assemblies.
- **4.2.5** The application must not call any APIs in the Microsoft.Xna.Framework.Game assembly or the Microsoft.Xna.Framework.Graphics assembly when using any methods from the System.Windows.Controls namespace.

4.3 Phone Capabilities Detection

When you create a Windows Phone project using Visual Studio 2010 Express for Windows Phone, the Windows Phone application manifest file is auto-generated that includes a list of all the phone capabilities supported by Windows Phone. The phone capabilities listed on the application manifest file are displayed to the user during application purchase. In addition, operating system grants the security permissions to the application according to the capabilities listed on the manifest file. For more information, see the Application Manifest File for Windows Phone topic.

The application submission process uses Microsoft intermediate language (MSIL) code analysis to detect phone capabilities. The phone capabilities detected on your application are listed on the Windows Phone application manifest file replacing existing capabilities, as described in Step b, Section 4.1.2.

© Microsoft Corporation. All Rights Reserved. This document is subject to updates.

Note:
You can submit an application with obfuscated code and the detection process still applies.

Note:
The capability detection process does not discover Windows Phone APIs invoked via .NET reflection. As a result, the application will not have the security permissions required to run properly and will result in a failure during certification.

Microsoft recommends that you test the application with the same phone capabilities that are generated during the application submission process. To do this, you can run the Windows Phone Capability Detection tool and populate the application manifest file with the phone capabilities generated by the tool. After completing this step and building the application, if errors are reported, it might indicate that the application uses undocumented APIs or invokes APIs via .NET reflection. For more information, see the How to: Use the Windows Phone Capability Detection Tool topic.

Note:
The capability list can change across application updates. When you submit an application update for certification, it goes through the same process as the original application.

4.4 Language Validation

Every application is targeted to publish to at least one specific geographic market and language. You can target multiple markets and submit your application in multiple languages. The language detection process includes the evaluation of the metadata that is used to describe the application and the UI text that is used within the application.

The following list shows the supported languages in Windows Phone 7:

- English
- French
- Italian
- German
- Spanish

An application must be localized in at least one of the supported application languages on the Windows Phone Marketplace. You will be prompted to submit a description only for the languages listed above. The users see this description when browsing the Windows Phone Marketplace catalog.

© Microsoft Corporation. All Rights Reserved. This document is subject to updates.

4.5 Windows Phone Marketplace Iconography

For each application, you must submit one icon to represent your application in the Windows Phone Marketplace catalog. This icon must match closely the icon provided in the XAP package. Users see this icon when browsing the application catalog on the phone before making a purchase.

Note:
Do not use transparent PNG image files for the following phone application icons.

The following table lists the phone application icon requirements.

Icon	Where Used	Size (Pixels)	File Type
Small mobile app tile artwork (required)	In the phone Marketplace catalog	99 x 99	PNG
Large mobile app tile artwork (required)	In the phone Marketplace catalog	173 x 173	PNG
Large PC app tile artwork (required)	In the PC Marketplace catalog	200 x 200	PNG
Background art (optional)	Background panorama	1000 x 800	PNG

4.6 Application Screenshot

For each application, you must provide at least one or up to a maximum of eight screenshots. Users see these screenshots in the details page of the catalog before they make a purchase.

Screenshots must only contain application graphics, and must not include any emulator chrome. Graphically-enhanced screenshots are not allowed.

Note:
Do not use transparent PNG image files for application screenshots.

The following table lists the size and file type requirements for application screenshots.

Screenshot	Size (Pixels)	File Type
Details page screenshot	480 x 800	PNG

© Microsoft Corporation. All Rights Reserved. This document is subject to updates.

5.0 Application Certification Requirements

An application that is submitted for certification must meet all of the following requirements. An application that does not meet one or more of the requirements fails the certification process.

Note:
These requirements apply equally to an application that implements game functionality, commonly referred to as a game.

5.1 Application Reliability

- **5.1.1 Multiple Devices Support**

  The application must run on any Windows Phone 7 device, regardless of model, screen size, keyboard hardware, and manufacturer.

- **5.1.2 Application Termination**

  The application must handle exceptions raised by the .NET Framework and not terminate unexpectedly. During the certification process, the application is monitored for unexpected termination. An application that terminates unexpectedly fails certification.

  When handling exceptions, an application must provide a user-friendly error message. You may present a message that is relevant to the context of the application. The application must continue to run and remain responsive to user input after the exception is handled.

  An application that displays generic or unhelpful error messages will fail certification.

- **5.1.3 Application Does not Hang**

  The application must not become unresponsive to user input because of an operation within the application. For time consuming activities such as downloading data over network connections, the application must display a visual progress indicator. When there is a visual progress indicator, you must implement a UI element that provides the user with an option to cancel the time consuming activity.

5.2 Performance and Resource Management

- **5.2.1 Launch Time**

  a. The application must render the first screen within 5 seconds after launch.

     The application may provide a splash screen image in a file called SplashScreenImage.jpg in the root of the XAP package while the application is still trying to load. However, the first application screen must be rendered within the 5 second requirement even when there is a splash screen.

     Microsoft recommends that the application provides a splash screen image only when it takes longer than 1 second to load the first screen.

  b. Within 20 seconds after launch, the application must be responsive to user input.

© Microsoft Corporation. All Rights Reserved. This document is subject to updates.

- **5.2.2 Application Responsiveness After Being Closed**

  A Windows Phone application is closed and terminated when the user navigates away from the application. When an application is started after being closed, its launch time must meet the requirements in Section 5.2.1. It is recommended that the application appears to be a fresh instance. For more information, see the Execution Model Overview for Windows Phone topic.

- **5.2.3 Application Responsiveness After Being Deactivated**

  A Windows Phone application is deactivated when the user presses the Start button or if the device timeout causes the lock screen to engage. A Windows Phone application is also deactivated when it invokes a Launcher or a Chooser API. When activated, the application launch time must meet the requirements in Section 5.2.1.

  Microsoft recommends that the application reestablishes the state of the application that the user experienced before the application was deactivated. For more information, see the Execution Model Overview for Windows Phone topic.

  Note:
  If the application does not complete the action taken in the Activated or Deactivated event handlers within ten seconds it will be terminated by the operating system. Such applications will fail certification per the 5.1.2 Application Termination requirement.

- **5.2.4 Use of Back Button**

  To maintain a consistent user experience, the Back button must only be used for backwards navigation in the application.

  a. Pressing the Back button from the first screen of an application must exit the application.
  b. Pressing the Back button must return the application to the previous page.
  c. If the current page displays a context menu or a dialog, the pressing of the Back button must close the menu or dialog and cancel the backward navigation to the previous page.
  d. For games, when the Back button is pressed during gameplay, the game can choose to present a pause context menu or dialog or navigate the user to the prior menu screen. Pressing the Back button again while in a paused context menu or dialog closes the menu or dialog.

- **5.2.5 Memory Consumption**

  An application must not exceed 90 MB of RAM usage, except on devices that have more than 256 MB of memory. You can use the **DeviceExtendedProperties** class to query the amount of memory that is available on the device and modify the application behavior at runtime to take advantage of additional memory. For more information, see the DeviceExtendedProperties class in MSDN.

  Note:
  The DeviceTotalMemory value returned by **DeviceExtendedProperties** indicates the physical RAM size in bytes. This value is less than the actual amount of device memory. For an application to pass certification, Microsoft recommends that the value returned by **ApplicationPeakMemoryUsage** is less than 90 MB when the **DeviceTotalMemory** is less than or equal to 256 MB.

© Microsoft Corporation. All Rights Reserved. This document is subject to updates.

- **5.2.6 Trial Applications**

  An application must not invoke either of the Trial APIs in a tight loop. For example, a game application must not invoke either of the Trial APIs while in a game loop. The API should be called infrequently and the value returned should be cached. For information about best practices for creating trial applications, see Trial Applications for Windows Phone.

5.3 Phone Functionality

The application must not delay or prevent the ability of the user to initiate a call, answer an incoming call, or end a call. The application must not delay or prevent the ability of the user to send or receive SMS or MMS messages.

The application must not hang or terminate unexpectedly when there is an incoming phone call, SMS, or MMS message.

5.4 Security

- **5.4.1 Malicious Software Screening**

  The application must be free of viruses, malware, and any malicious software.

- **5.4.2 MSIL Type Safety Verification**

  Windows Phone 7 operating system implements multiple sandbox mechanisms to help protect the integrity of the device and the applications running on the device. The Common Language Runtime (CLR) on Windows Phone 7 relies on the type-safe execution of application code to help enforce security and isolation mechanisms.

  An application must implement type-safe MSIL code to pass certification. For more information about C# unsafe code, see Unsafe Code and Pointers (C# Programming Guide).

- **5.4.3 Security Transparency Verification**

  The Windows Phone Application Platform does not allow an application to run security critical code. An application that invokes security critical code will fail certification. For more information about the .NET Security model, see Security Changes in the .NET Framework 4.

© Microsoft Corporation. All Rights Reserved. This document is subject to updates.

5.5 Content Validation

Content validation includes the evaluation of the metadata that describes the application as well as the UI text and media within the application. For example, an application that is submitted in French requires a product description in French. In addition to evaluating the description, the certification process reviews UI text, screenshots, and icons, in French.

Additional content validation criteria are described in Section 3.0. The application content (e.g. text, visual elements) must be visible and legible regardless of the phone theme. For example, if the phone theme changes from black background to white background, the text and visual elements of your application must be visible or legible.

When you submit separate XAP packages for each language, each XAP package is evaluated as individual submissions. By submitting a single XAP package for all supported languages you reduce the certification feedback time.

5.6 Technical Support Information

An application must include the application name, version information, and technical support contact information that are easily discoverable.

© Microsoft Corporation. All Rights Reserved. This document is subject to updates.

6.0 Additional Requirements for Specific Application Types

An application that uses specific phone capabilities must meet the following requirements in addition to all the requirements specified above.

6.1 Location Aware Application

A location aware application can access the phone location by using the classes in System.Devices.Location namespace. For more information, see the Location for Windows Phone topic.

Users have the ability to turn off the Location Service on the phone from the System Settings page. Location aware applications must not hang or terminate unexpectedly when the Location Service is turned off on the phone. One recommendation is to present a user friendly message to indicate that location data is not available. Another recommendation is to provide the user with the ability to view and disable the use of location data.

6.2 Push Notifications Application

The Microsoft Push Notification Service provides a dedicated, resilient, and persistent channel for pushing notifications from a web service to a mobile device. For more information, see the Push Notifications for Windows Phone topic in MSDN®.

An application that binds a push notification channel to a toast notification must meet the following requirements:

- **6.2.1 Configurable Functionality**

  In the UI or settings menu, the application must provide the user with the ability to independently disable toast notifications.

- **6.2.2 Initial Push Notifications Functionality**

  On first use of HttpNotificationChannel.BindtoShellToast method, the application must ask the user for explicit permission to receive a toast notification.

6.3 Applications Running under a Locked Screen

An application in the foreground can continue to run when the phone screen is locked by setting the PhoneApplicationService.ApplicationIdleDetectionMode property.

By setting up your application to run when the phone screen is locked, a user is able to access the application quickly upon unlock. However, when your application runs under a locked screen, it could consume power outside of the user's control. For this reason, your application must minimize power usage when running under a locked screen, and adhere to the following requirements.

- **6.3.1 Minimize Power Usage When Running Under a Locked Screen** All applications that run under a locked screen must stop any UI updates, active timers, and other non-critical processing when notified that the screen is locked. When the screen locks, the operating system notifies the application by raising the PhoneApplicationFrame.Obscured event with the ObscuredEventArgs.IsLocked property set to true.

© Microsoft Corporation. All Rights Reserved. This document is subject to updates.

- **6.3.2 Applications that play audio under a locked screen**
  - **6.3.2.1** The minimum battery life of the phone must be greater than six hours while the application plays audio under a locked screen.
  - **6.3.2.2** If an application is not playing audio when the phone is locked, the application must remain idle while the phone screen is locked.
- **6.3.3 Applications that do not play audio under a locked screen**
  - **6.3.3.1** The minimum battery life of the phone must be greater than 120 hours while the application is running under a locked screen.

6.4 Music + Videos Hub Application

An application in the Music + Videos Hub provides an integrated music and video experience on the phone as its primary function. An application that calls the Microsoft.Devices.MediaHistory or Microsoft.Devices.MediaHistoryItem classes is considered as a Music + Videos Hub application and will appear in the Marquee list when installed on the phone. The submission process detects that the application uses these classes and automatically updates the hub type to Music + Videos in the Windows Phone application manifest.

An application in the Music + Videos Hub must meet the following requirements:

- **6.4.1** The application functionality must be related to video and/or music media playback.
- **6.4.2** When a user taps a tile associated with the application in the “History” or “Now Playing” area of the Music + Videos Hub, the application must either (a) launch the playback experience for the content identified in the tile, or (b) launch a view that provides information about the previously played media content and allows the user to resume. The application must not launch to the main or default landing page when the user taps on a content tile in the “History,” Now Playing,” or “New” area of the Music + Videos Hub.

  An application in the Music + Videos Hub may override music already playing in the background.

  Note:
  Requirements in Section 6.5 do not apply to Music + Videos Hub applications.
- **6.4.3** The application must update the “History” area of the Music + Videos Hub when the application plays media.
- **6.4.4** The application must update the “New” area of the Music + Videos Hub when media is added to the device or when the user creates an “object” in the application (for example, a radio station is created, a music tag is created.)
- **6.4.5** When the media is associated with a container, the hub tile in “New” and “History” in the Music + Videos Hub must represent a valid container, such as album, artist, playlist, radio station, rather than individual media items.
- **6.4.6** The hub tiles in the Music + Videos Hub must not contain advertisements, media feeds, or other unsolicited content. To help avoid confusion for users, hub tiles should not contain album art unless the album plays when the hub tile is pressed. If your application plays a stream, the hub tile graphic should describe the stream that is being played.
- **6.4.7** Comply with the iconography rules for the Music + Videos Hub as documented in the How to: Integrate with the Music and Video Hub for Windows Phone topic.

© Microsoft Corporation. All Rights Reserved. This document is subject to updates.

6.5 Applications that Play Media

An application can play media in the background while it is running even when its primary function is not about music or video. An application that plays music, audio, or sound effects (except applications in the Music + Videos Hub) must meet the following requirements:

- **6.5.1 Initial Launch Functionality**

  When the user is already playing music on the phone when the application is launched, the application must not pause, resume, or stop the active music in the phone MediaQueue by calling the Microsoft.Xna.Framework.Media.MediaPlayer class.

  If the application plays its own background music or adjusts background music volume, it must ask the user for consent to stop playing/adjust the background music (e.g. message dialog or settings menu).

  Note:
  This requirement does not apply to applications that play sound effects through the Microsoft.Xna.Framework.Audio.SoundEffect class, as sound effects will be mixed with the MediaPlayer. The SoundEffect class should not be used to play background music.

  Note:
  This requirement does not apply to Music + Videos Hub applications that are described in Section 6.4

- **6.5.2 Configurable Functionality**

  If the application needs to play its own background music or adjust background music volume, it must provide the user with the configurable settings for its music.

- **6.5.3 Applications that Play a Video or Audio Segment**

  An application may interrupt the currently playing music to play a non-interactive full motion video or audio segment (e.g. cut-scene or media clip) without asking for user consent. If music was playing prior to the segment, the application must resume music when the segment has completed.

6.6 Photo Extras

Photo extras allow you to integrate your photo altering application with the Windows Phone built-in photo application. Through the extras menu in the photo viewer, end users can access your application without leaving the primary Windows Phone photo application. You enable the “extras” menu in the photo viewer of the phone by creating an Extras.xml file in the root of the XAP package. For more information, see the Photo Extras Application Extensibility for Windows Phone topic.

© Microsoft Corporation. All Rights Reserved. This document is subject to updates.

An application with Extras.xml in the root of the XAP package must meet the following requirements:

- **6.6.1** The application must implement the primary functionality associated with photo manipulation.
- **6.6.2** The Extras.xml file in the root XAP package must be valid according to the description in the Photo Extras Application Extensibility for Windows Phone topic.
- **6.6.3** The application must support two kinds of launch behaviors:
  a. When an application is launched from the application list without an input photo, the application must invoke the PhotoChooser method to enable the user to pick up a photo or capture a new one.
  b. When an application is launched from the “extras” menu in the photo viewer with an input photo in the JPEG file format, the application must implement a functionality to allow the user to manipulate the input photo without any photo selection steps.

Microsoft recommends that the application notifies the user when it saves the edited photo.

6.7 Photo Sharing Applications

A photo sharing application can use the Windows Phone built-in share picker functionality. In the photo viewer, the user can use the share picker to upload the photo through your photo upload application. Your photo upload application can use the standard list of messaging, email accounts, and other upload services. To extend the share picker to include your photo upload application, create an E0F0E49A-3EB1-4970-B780-45DA41EC7C28.xml file in the root of the XAP package. For more information, see the Share Picker Extensibility for Windows Phone topic.

A photo sharing application that uses the Windows Phone built-in share picker functionality must meet the following requirements:

- **6.7.1** The application must implement the primary functionality associated with photo uploading.
- **6.7.2** The application must support two kinds of launch behaviors:
  a. When an application is launched from the application list without an input photo, the application must invoke the PhotoChooser method to enable the user to pick up a photo or capture a new one.
  b. When the application is launched from the share picker through the photo viewer with an input photo in the JPEG file format, the application must implement a functionality to allow the user to upload or share the input photo without any photo selection steps.

Microsoft recommends that the application notify the user when the photo uploading is completed.

© Microsoft Corporation. All Rights Reserved. This document is subject to updates.

7.0 Change History

Date	Document Version	Change Description
July 2010	1.1	Section 3.5. Alcohol, Tobacco, Weapons, and Drugs. Removed the following text: Any content that facilitates the use of weapons in the real world.
July 2010	1.1	Section 4.1.1. List of Package Requirements. Reduced the maximum size of XAP file from 500 MB to 400 MB.
July 2010	1.1	Section 4.1.1. List of Package Requirements. Removed the Appendix topic Windows Phone Application Manifest File and redirected the link to the MSDN topic.
July 2010	1.1	Section 4.5. Windows Phone Marketplace Iconography. Changed optional panoramic background art size from 1426 x 800 to 1422 x 800.
July 2010	1.1	Section 6.3. Applications Running Under a Locked Screen. Changed the reference to PhoneApplicationService.ApplicationIdleDetectionMode from method to property.
August 2010	1.2	Section 2.1. Application Policies. Updated the text to clarify the requirements for games.
August 2010	1.2	Section 2.10. Application Policies. Added a new section about applications using Microsoft Location Service API.
August 2010	1.2	Section 3.0. Content Policies. Updated the section overview to state that a game that has been rated by the approved ratings boards must submit the ratings certificate. Previous statement indicated that it was optional to submit the ratings.
August 2010	1.2	Section 4.1.1. (2) List of Package Requirements. Added a requirement that the <Title> element in the application manifest cannot be empty.
August 2010	1.2	Section 4.2.5. Application Code Validation. Added a new requirement about XNA Framework assemblies.
August 2010	1.2	Section 4.5. Windows Phone Marketplace Iconography. Change the panoramic background art size from 1422 x 800 to 1000 x 800.

© Microsoft Corporation. All Rights Reserved. This document is subject to updates.

Date	Document Version	Change Description
August 2010	1.2	5.2.1. Launch Time. Added a clarification to the original requirement.
August 2010	1.2	5.2.2. Application Responsiveness After Being Closed. Revised the original text.
August 2010	1.2	5.2.3. Application Responsiveness After Being Deactivated. Added a new requirement.
August 2010	1.2	5.2.4. Use of Back Button. Revised the original text.
August 2010	1.2	5.2.5. Memory Consumption. Added a new requirement.
August 2010	1.2	Section 5. Technical Support Information. Added a new requirement.
August 2010	1.2	Section 6.2. Push Notification Application. Removed references to tile notifications.
August 2010	1.2	6.5.3. Applications that Play a Video or Audio Segment. Revised the statement.
August 2010	1.2	7.0. Change History. Added a new section that provides details of changes to the document.
September 2010	1.3	Section 2.1. Application Policies. Added a clarification regarding applications in the Games Hub.
October 2010	1.4	Section 4.1.1. List of Package Requirements. Changed the maximum size from 400 to 225 MB. Step b. Added a clarification that the application title displayed on the phone must be the same as the title entered in Step 2 of the submission process to Windows Phone Marketplace. Step e. Changed **start experience** to **app list**. Removed the statement “The application is optional for games.” Added a requirement that games must use the application tile image in place of the application icon. Step f. Added a column in the table about where the application icon and tile image shows up on the phone. And added screenshots of the **start experience** and **app list**.

© Microsoft Corporation. All Rights Reserved. This document is subject to updates.

Date	Document Version	Change Description
October 2010	1.4	4.3. Phone Capabilities Detection. Added information about the Windows Phone Capability Detection Tool.
October 2010	1.4	4.5. Windows Phone Marketplace Iconography. Changed the icon descriptions in the table to match the UI in Step 3 of the application submission process. Added a column to the table describing where artwork is used.
October 2010	1.4	4.6. Application Screenshot. Revised this requirement to describe that screenshots must not include any emulator chrome.
October 2010	1.4	5.2.5. Memory Consumption. Revised the first paragraph and added a note about DeviceExtendedProperties and ApplicationMemoryUsage.
October 2010	1.4	5.2.6. Trial Applications. This is a new requirement.
October 2010	1.4	6.3. Applications Running under a Locked Screen. Updated section with a description of the advantages and challenges of running an application under a locked screen. Separated guidance for audio and/or video applications, and all other applications. Added requirements for minimum battery life and idle behavior while the application is running under a locked screen. Removed the requirement **6.3.1 Configurable Functionality**.
October 2010	1.4	6.4.1. Music + Videos Hub Application. Updated text to 'video and/or music media playback'.
October 2010	1.4	6.4.2. Music + Videos Hub Application. Revised this requirement to add a second implementation option.
October 2010	1.4	6.4.6. Music + Videos Hub Application. Added a section to the requirement that states that hub tiles should not contain album art unless the album plays when the hub tile is pressed.
October 2010	1.4	6.5. Applications that Play Media. Updated the first paragraph to better describe the kinds of applications or games that must meet this requirement.
October 2010	1.4	6.7. Photo Sharing Applications. This is a new requirement.

© Microsoft Corporation. All Rights Reserved. This document is subject to updates.