

# WINDOWS PHONE 8 RELEASE NOTES: BUILD 8325.9686

---

May 11, 2012

[Pairing note](#)

[What's new](#)

[Known issues and breaking changes](#)

[System requirements](#)

[Installation and uninstallation](#)

[Change history](#)

## PAIRING NOTE

- Build 9686 of Windows Phone 8 is qualified to use with build 8325 of the Windows Driver Kit (WDK) for Windows Developer Preview.
  - The full Windows Phone 8 version is: WPMAIN.8325.9686.20120508-1225
  - The full Windows version is: fbl\_core1\_mobile\_dev\_wp\_8325\_0\_120422-1700
- Windows Phone 8 was tested against the following Qualcomm BSP:
  - **8960**: Version: Version: 1.30.120409.1120, Release Date: April 9, 2012
- This kit was tested against Visual Studio 11 Beta (Dev11) version 11.0.50214.1 BETAREL.

See [System requirements](#) to make sure your system can support the tools and kits. See [Installation and uninstallation](#) for information about the kits and tools that you need to get started.

## WHAT'S NEW

The following changes and features are new in this release:

- Phase 1 of the new kit setup experience is now available with this release.


In the **Apollo** kit folder, you can now select an app called Setup.exe, which installs the following components:

- Windows Phone Driver Kit
- Windows Phone Adaptation Kit
- Nokia Add-On Kit
- Windows Phone 8 tools such as Test Central, TShell, and Tungsten
- Windows Phone 8 Symbols

To minimize installation issues and ensure that you install the latest version of the components that is available with this kit release, you must uninstall previous versions of the development kits and the tools (i.e. you need to do a clean install). See the [Installation and uninstallation](#) section for updated kit and tools setup instructions. There are also some known issues with the new setup so check the [Known issues and breaking changes](#) section for more information.

- 📌 Setup.exe does not install/uninstall the WDK because this has not yet been integrated so you need to continue with the same manual install/uninstall steps for the WDK.

- Information about how you can turn off the new UI elements in the kit is provided for reference:

 If you need to release Windows Phone images externally, remove the following line from the ProductionOEMInput.xml file or from TestOEMInput.xml file:

```
<OptionalFeature>SHELLSTART8</OptionalFeature>
```

- **8325.9686 Fixed - Code defects and work items.xlsx** lists the code defects and feature work items that were resolved and that are included in this kit release.
- All remaining active Qualcomm dependencies are now being tracked in a Qualcomm BSP database. Because of this, the **Apollo Kit and Qualcomm BSP Scorecard.pdf** will no longer be provided as part of the Windows Phone 8 release notes package.

## KNOWN ISSUES AND BREAKING CHANGES

### ISSUES AND WORKAROUNDS RELATED TO SETUP.EXE

**Summary:** You may encounter the following issues when using the new Setup.exe. When possible, information about a workaround has been provided.

- You may need to uninstall Test Central and TShell before running Setup.exe. To do this, follow these steps:
  1. Click **Start**, and then point to **Control Panel**. Point to **Programs**, and then click **Programs and Features**.
  2. Find **Windows Phone 8 Test Central**, click the program name, and then click **Uninstall**.
  3. Follow the same procedure to uninstall TShell.
  4. Run Setup.exe after these two programs have been successfully uninstalled.
- You can install Test Central without installing the WDK by following these steps.
  1. In Setup.exe, under **Select products to install:**, expand **Windows Phone Driver Kit**.
  2. First, clear the checkbox **Core Components (fre)** and **Test Central**.
  3. Then, select **Test Central**, and then select **Core Components (fre)**.
  4. Proceed with the installation.
- You can use the Test Central UI by following these steps:
  1. In Setup.exe, under **Select products to install:**, select **Test Shell** and proceed with TShell installation.
  2. When finished, open an elevated Visual Studio 11 command line.
  3. From the command line, navigate to C:\Program Files (x86)\Windows Phone TShell.
  4. Run TestCentral.exe to see the Windows Phone 8 Test Central UI.

CORRECT DEPLOYTEST.EXE PATH CANNOT BE FOUND WHEN RUNNING TEST CASES THROUGH TEST CENTRAL

**Summary:** Due to Windows Phone 8 bug #156999 ([WPTC] Cannot find correct deploytest.exe path when running test cases through Test Central), you need to update the registry to make sure you can run test cases.

To address this issue, use the following workaround:

**Mitigation:** For a 32-bit machine where the OS is installed on the C: drive, change the path of HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Windows Phone 8 Test Central\TestPackages\TestToolsDir

**From:** C:\Program Files\Windows Kits\8.0\

**To:** C:\Program Files\Windows Kits\8.0\WP8\Tools\bin\i386

**Mitigation:** For a 32-bit machine where the OS is installed on the C: drive, change the path of HKEY\_LOCAL\_MACHINE\SOFTWARE\Wow6432Node\Microsoft\Windows Phone 8 Test Central\TestPackages\TestToolsDir

**From:** C:\Program Files (x86)\Windows Kits\8.0\

**To:** C:\Program Files (x86)\Windows Kits\8.0\WP8\Tools\bin\i386

Once you have updated the path for the registry key, you can then use TestD to deploy and run the test. The test tool finds the dependencies from the package and deploys the contents into the device.

If you need to use DeployD to deploy, you must copy the TestTool binaries located in C:\{Program Files}\Windows Kits\8.0\WP8\Tools\bin\i386 to the C:\{Test\_Central\_Installation\_Folder}\Packages\TestTools directory.

**ANNOUNCEMENT:** PARTNERS ARE REQUIRED TO BUILD ALL THEIR PACKAGES USING THE LATEST PACKAGING TOOLS

**Summary:** Microsoft now requires all packages to be built using the latest kit and the newest packaging tools. This ensures compatibility when Microsoft takes the prebuilt packages from OEM partners, which are used for internal Windows Phone images.

Microsoft will no longer accept packages that were generated using an older version of the kit.

**ANNOUNCEMENT:** ALL EXECUTABLE FILES NEED TO BE CONFIGURED TO WORK WITH THE SECURITY CHAMBER MODEL BEGINNING ON MAY 18

**Summary:** Beginning with a kit release that is currently planned to be released to partners on May 18, all executable files that run on the phone need to be configured to work properly with security chamber model of the OS. Executable files that are not configured properly to work with the security chamber model will fail to start.

This change will affect any executable files written by OEMs, such as services and applications that run in builds of the full OS. This change will not affect the Microsoft Manufacturing Operating System (MMOS). For more information about the security chamber model in Windows Phone 8, see the *Security model overview* topic Windows Phone 8 Partner Documentation.

**Mitigation:** When this change takes effect, the following components will run on a phone only if they are configured as described below:


- Services: OEMs must include services in package files, and each service must be defined in a **<Service>** element in the package XML file. For more information, see the *Creating services* topic in the Windows Phone 8 Partner Documentation.

- Applications: Applications built using the Windows Phone 8 SDK will be automatically configured to work properly with security chamber authorization when they are built into XAP files using the SDK tools in Visual Studio. No further work from OEMs will be required.
- Test applications: Test applications that are transferred directly to a phone by using TShell (rather than being included in a package) will run only if they are started from the C:\Data\Test or C:\Test folders on the phone.

## SYSTEM REQUIREMENTS

The following table contains the system requirements for Windows Phone 8.

Supported operating systems	<ul style="list-style-type: none"> <li>• Windows 7 32-bit (x86) and 64-bit (x64)</li> </ul> <p>You must install all Windows 7 critical updates to avoid additional issues when using the Windows Phone 8 kit.</p>
Development environment	<ul style="list-style-type: none"> <li>• Visual Studio 11 Beta Professional or higher</li> </ul>

 If you are developing apps using the Windows Phone 8 SDK, check the SDK release notes to make sure that your system is compliant with the Windows Phone 8 SDK system requirements.

## INSTALLATION AND UNINSTALLATION


### UNINSTALLING EARLIER VERSIONS OF DEVELOPMENT KITS

If you are using a different version of the Windows Driver Kit (WDK), Windows Phone Driver Kit (WPK), or Windows Phone Adaptation Kit (WPAK) than the ones listed in the *Pairing note* section of this document, you first need to uninstall all of the programs associated with these development kits.

The uninstallation sequence is the reverse order of installation. The following list shows the order in which you must uninstall previous builds of the development kits and any tools or programs associated with them.

#### Windows Phone 8 Tools

1. Click **Start**, and then point to **Control Panel**. Point to **Programs**, and then click **Programs and Features**.
2. Identify the Windows Phone 8 tool that you need to install, click the program name, and then click **Uninstall**.
3. Follow these steps to uninstall another tool.

 Microsoft recommends that you always uninstall the earlier version of the Windows Phone 8 tools (such as Test Central, TShell, and so on) and install the version of the tools that came with the kit that you are using or have installed on your development workstation. This ensures that the tools are compatible with the kit.


## Development kits and associated components

Click **Start**, and then point to **Control Panel**. Point to **Programs**, and then click **Programs and Features**. For each program in the following list, click the program name, and then click **Uninstall**.

1. Windows Phone Adaptation Kit
2. Windows Phone Driver Kit
3. Windows Driver Kit ARM Additions
4. SDK ARM Additions
5. SDK ARM Redistributables
6. Windows Debugging VS Integration
7. Windows Driver Kit
8. SDK Debuggers
9. Windows Software Development Kit for Metro style apps
10. Windows Software Development Kit Redistributables
11. X86 Debuggers and Tools or X64 Debuggers and Tools
12. Windows Software Development Kit

If you installed earlier versions of the kits, you may have a version of one or more of the following programs installed. If you do, uninstall these:

- Windows Driver Kit Visual Studio 2010 Additions
- Application Verifier x86 External Package or Application Verifier x64 External Package

 In **Programs and Features**, show the **Installed On** column to view the dates when all programs were installed. Sorting the programs by their install dates offers an easy way to group together the programs and tools associated with the kits.

## INSTALLING THE TOOLS AND DEVELOPMENT KITS

### Visual Studio 11 Beta

- Download and install [Visual Studio 11 Beta](#).

Under **Downloads**, choose **Visual Studio**, and then choose **Ultimate**, **Premium**, or **Professional** for the version you want to download.

### Windows Driver Kit (WDK) and Windows Phone 8 kits (WPAK and WPAK)

Download the latest Apollo & WDK.zip packages from the Windows Phone Engineering Partners Connect site, and then unzip the files to your hard drive.

The WDK and Windows Phone 8 kit release (Apollo & WDK.zip) consists of the following files:

- Apollo.zip – Contains the Windows Phone 8 Adaptation Kit, Windows Phone 8 Driver Kit, and symbols.
- WDK.zip – Contains the Windows Driver Kit and associated symbols.
- Tools.zip – Contains the tools that currently are not included in Apollo.zip or WDK.zip.

You must install WDK, WPKD, and WPAK in the sequence specified here.

### WDK

1. Unzip the WDK (WDK.zip) file to your hard drive.
2. Open an elevated command prompt, navigate to the directory containing the UninstallKits.ps1 file, and then run the following command:

```
Powershell.exe -ExecutionPolicy bypass -file .\UninstallKits.ps1
```

3. Install WDK in the following order:
  - a. Run SDK\sdksetup.exe: right-click, and then click **Run as administrator**.
  - b. Run WDK\wdksetup.exe: right-click, and then click **Run as administrator**.
  - c. Run WOA\WOA\_SDK\sdkarmsetup.exe: right-click, and then click **Run as administrator**.
  - d. Run WOA\WOA\_WDK\wdkarmsetup.exe: right-click, and then click **Run as administrator**.
  - e. In an elevated Command Prompt window, browse to the WDK folder, and run **cscript managed.vbs**.
  - f. Reboot your computer.

### Windows Phone 8

1. In the **Apollo** folder, double-click **Setup.exe**.
2. In the Windows Phone Kits Setup screen, accept the terms in the License Agreement, and then select **Next**.
3. Under **Select products to install**: choose the programs you want to install:
  - Debugger Symbols
    - Debugger Symbols (fre)
    - Debugger Symbols (chk)
  - Nokia Add-On Kit
    - Add-On Kit (fre)
    - Add-On Kit (chk)
  - Windows Phone Adaptation Kit
    - Test Shell
    - Tungsten
    - Core Packages (fre)
    - Core Packages (chk)
  - Windows Phone Driver Kit
    - Test Central
    - Core Components (fre)
4. Once you have selected the programs you want to install, select **Next**, and then select **Install**.

### Symbols

To install the available Windows symbols:

1. Browse to the **WDK\Public\_MobileCore\_Symbols\MSI** folder, and then run the provided installers.
2. Add the paths of the symbols to the symbol path.

For example, you can use `.sympath` inside the debugger to add folder locations. For more information, see the topic *Symbol path* in the Windows Phone Partner Documentation.

For a large workgroup, consider creating a symbol server. For more information, see the topic *Symbol stores and symbol servers* in the Windows Phone Partner Documentation.

## Tools


You install some tools separately from Windows Phone 8 and WDK. To install tools such as VirtEth and BCDEdit, go to the **Tools** folder and run the installer for the tool that you want to install.

## VERIFYING THE INSTALLATION AND THE CONTENTS OF THE DEVELOPMENT KITS

### To verify that all the tools and programs were installed correctly

1. Click **Start**, and then point to **All Programs**. Browse to the **Windows Kits\Debugging Tools for Windows (x86)** folder.
2. In **Control Panel**, verify that the programs were installed, and then browse to **Windows Phone Driver Kit** and **Windows Phone Adaptation Kit**.

After successfully installing WDK, WPKD, and WPAK, the subdirectories are listed in the following directory.

 During early Windows Phone 8 development, the list of subdirectories may change. The list of subdirectories in the following table is provided as guidance and should not be used as a definitive list.

Location	Directories
Main directory	C:\Program Files\Windows Kits\8.0 for 32-bit OS C:\Program Files (x86)\Windows Kits\8.0 for 64-bit OS
Subdirectories under C:\Program Files\Windows Kits\8.0 or C:\Program Files (x86)\Windows Kits\8.0	bin build Catalogs CrossCertificates Debug Debuggers help Include Lib Redist Remote Shortcuts Testing Tools Windows Metadata WP8

## CHANGE HISTORY

---

### CHANGES IN THE 8325.9680 BUILD

**Pairing note:** NT Build version string: fbl\_core1\_mobile\_dev\_wp\_8325\_0\_120422-1700, Apollo Build: WPMAIN.8325.9680.20120501-1525

Windows Phone "Apollo" was tested on the following Qualcomm BSPs:

- **8660:** Version: Version: 1.30.120409.1120, Release Date: April 9, 2012

The kit was tested against Visual Studio 11 Beta (Dev11) version 11.0.50214.1 BETAREL.

The following changes and features were new for this release:

- Starting with this release of the tools, the package merge tool, PkgMerge.exe, should be used on packages prior to imaging. For more information about the package merging tool and the packaging process, see the topics *Overview of packaging* and *Merging packages before imaging* in the Windows Phone "Apollo" Partner Documentation.

This release also contained the following known issues and breaking changes:

- For KDNNet over USB, the default values on boot have now changed to busparams=1 for the USB port on the phone and busparams=2 for the USB port on the debug board. This change is necessary to achieve full KDNNet convergence with Windows 8. No mitigation steps are needed. The old values busparams=255.3.0 for USB port on the phone and busparams=255.3.1 for USB port on the debug board still work.
- Partners need to include the correct security model settings to run tests and load files in C:\Data\Test. This change impacts OEMs that need to run Windows Phone tests.

This change isolates the required settings for executing tests from the SecurityPolicyBvt1 package into a new package named Microsoft.Phone.Test.BaseOs.SecurityModelTestSupport. In addition, both the Health and Test SKUs now include a new feature called SECURITYMODELTESTSUPPORT.

This new feature is required for SKUs that need to run tests. If the SKU does not have the feature, the package needs to be installed through image update on the device after the device boots.

If the new security setting is not included on an image that requires running tests, the test binaries will not be able to inject binaries from C:\Data\Test into other processes. After chamber authorization is enforced, binaries will not be able to execute without this change.

Partners must include the correct security model settings, SECURITYMODELTESTSUPPORT, in the required SKU/image.

- **Announcement:** All executable files need to be configured to work with the security chamber model beginning on May 18.

In a future kit release, all executable files that run on the phone need to be configured to work properly with security chamber model of the OS. Executable files that are not configured properly to work with the security chamber model will fail to start.

This change will affect any executable files written by OEMs, such as services and applications that run in builds of the full OS. This change will not affect the Microsoft Manufacturing Operating System (MMOS). For more information about the security chamber model in Windows Phone "Apollo", see the *Security model overview* topic Windows Phone "Apollo" Partner Documentation.



More information will be provided in the documentation and the kit release notes when this change takes effect.

## CHANGES IN THE 8325.9678 BUILD

**Pairing note:** NT Build version string: fbl\_core1\_mobile\_dev\_wp\_8325\_0\_120422-1700, Apollo Build: WPMMAIN.8325.9678.20120427-1526

Windows Phone "Apollo" was tested on the following Qualcomm BSPs:

- **8660:** Version: Version: 1.30.120409.1120, Release Date: April 9, 2012

The kit was tested against Visual Studio 11 Beta (Dev11) version 11.0.50214.1 BETAREL.

The following changes and features were new for this release:

- WDK build 8325 contains fixes for the CPU hints for power collapse and a fix for the SB bus bug that speeds up the 8960 boot time by 40 seconds. The known issue (Power collapse does not work with Windows build 8311) that was documented in the release notes for 8311.9669 is now fixed.
- This kit contained a change that requires all drivers and kernel-mode components to be digitally signed. Unsigned kernel-mode components and drivers will fail to load and may cause boot failures. For more information about signing drivers, see the *How to: Sign drivers* topic Windows Phone "Apollo" Partner Documentation.
- Windows Phone properties can now be modified from the Visual Studio UI. Previous versions of the WPDK required developers to manually edit the VcxProj files to set or modify the Windows Phone properties such as the BuildWindowsPhone property.

Support has been added to these properties to be displayed from the **Project Properties** page so manually editing the VcxProj file is no longer required.

- Bugs in the default values for the **<AdditionalDependencies>** and **<IncludePath>** settings have been fixed so you can now remove the previous overrides required in the VcxProj files. For more information about these changes, see the following:
  - In previous versions of the WPDK, user-mode components that were built with the WindowsApplicationForDrivers8.0 toolset and had the BuildWindowsPhone flag set to true linked against the following libraries: KERNEL32.LIB, USER32.LIB, GDI32.LIB, and ADVAPI32.LIB. Windows Phone does not support these libraries.

In this release, that issue has been resolved and the unsupported libraries have been removed from the **<AdditionalDependencies>** list. By default, these components will link to mincore.lib and all other libraries have been removed from the default list.

If you added any project-specific overrides for **<AdditionalDependencies>** to work around the previous issue, please remove these. Components that need to link against additional libraries supported by Windows Phone still need to append these dependencies to the **<AdditionalDependencies>** list.

- In previous versions of the WPDK, the Windows Phone CRT headers were installed to an incorrect folder. To work around this issue, user-mode components building with the WindowsApplicationForDrivers8.0 toolset had to add the following lines to the VcxProj file:

```
<PropertyGroup>
<IncludePath>$(WPKInstallDir)WP8\Tools\WPE\CRT\inc\CRT;$(IncludePath)</IncludePath>
</PropertyGroup>
```

The install path for the CRT headers has been fixed so the previous workaround should be removed. The WindowsApplicationForDrivers8.0 toolset now includes the CRT headers and libraries by default.

This release also contained the following known issues and breaking changes:

- **Announcement:** All executable files need to be configured to work with the security chamber model beginning on May 18.

In a future kit release, all executable files that run on the phone need to be configured to work properly with security chamber model of the OS. Executable files that are not configured properly to work with the security chamber model will fail to start.

This change will affect any executable files written by OEMs, such as services and applications that run in builds of the full OS. This change will not affect the Microsoft Manufacturing Operating System (MMOS). For more information about the security chamber model in Windows Phone "Apollo", see the *Security model overview* topic Windows Phone "Apollo" Partner Documentation.

More information will be provided in the documentation and the kit release notes when this change takes effect.

## CHANGES IN THE 8311.9669 BUILD

**Pairing note:** NT Build version string: fbl\_core1\_mobile\_dev\_wp\_8311\_0\_120409-1700, Apollo Build: WPMAIN.8311.9669.20120417-1412

Windows Phone "Apollo" was tested on the following Qualcomm BSPs:

- **8660:** Version 5095, Release date: December 6, 2011
- **8960:** Version: 1.30.120326.1105, Release Date: March 26, 2012

The kit was tested against Visual Studio 11 Beta (Dev11) version 11.0.50214.1 BETAREL.

The following changes and features were new for this release:

- WDK build 8311 contains a new compiler drop, encryption on KNet transport, and Windows changes that unblock Windows Phone "Apollo" features.
- A patched KNet.dll is no longer required for the kit. The WinDbg issue has been fixed in WDK build 8311.

This release also contained the following known issues and breaking changes:

- **Announcement:** Beginning with kit release 8323.xxxx and later, all kernel-mode device drivers must be signed.

This is only an announcement to prepare partners for the upcoming change. For more information about signing drivers, see the *How to: Sign drivers* topic Windows Phone "Apollo" Partner Documentation.

- Due to some isolated change lists and the changes that Windows build 8311 brought, the system is not entering low power states based on idle durations present in the system.

These issues will block partners who are working on any idle power management development work. Please continue to use the previous versions of the kit for idle power management development work.

- If you use one of the sample OEMInput image files under %ProgramFiles(x86)%\Windows Kits\8.0\WP8\OEMInputSamples as the basis for your input file, you must move the following DEBUGGERON optional feature reference from **<MSOptionalFeatures>** to **<InternalOptionalFeatures>**.

```
<OptionalFeature>DEBUGGERON</OptionalFeature>
```

- A new KNet drop is now available as part of the kit. The tools have been updated to reflect the supported state of the tools and operating system. You must use the latest version of the debugger so it works with the new KNet implementation.
- If you are using KNet on Fluid 8660, the Ethernet debug port now has to be set to busparams=255.4.0. This change does not impact KNet over USB.
- Phone-specific GUIDs that were previously added to ksmedia.h have been moved to ksmedia\_phone.h. If you are using these phone-specific GUIDs from ksmedia.h, you must now include ksmedia\_phone.h. If your code does not compile, add the following:

```
#include <ksmedia_phone.h>
```

- The WDK toolset adds several libraries, such as kernel32.lib, to the dependency list and the WPK fails to remove these libraries. Because these libraries are not supported on Windows Phone "Apollo", this dependency will prevent the user-mode component from loading.

To work around this issue, add the following lines to the end of the project file to override the dependency list:

```
<ItemDefinitionGroup>  
  <Link>  
    <AdditionalDependencies>mincore.lib;msvcrt.lib;msvcprt.lib;vcomp.lib</AdditionalDependencies>  
  </Link>  
</ItemDefinitionGroup>
```

- Support for SYSCRT is not available: api-ms-win-core-crt-\*.lib libraries are missing from the WPK. Microsoft recommends linking user-mode services and driver components against SYSCRT instead of the full CRT. However, when support for the full CRT was added to the WPK, the SYSCRT libs were mistakenly removed. As a result, the SYSCRT libs api-ms-win-core-crt-11.lib and api-ms-win-core-crt-12.lib are not included in the kit.

To work around this issue, use the WPK as-is and link against the full CRT. The full CRT .libs are installed by default to the C:\Program Files (x86)\Windows Kits\8.0\WP8\Tools\WPE\CRT\lib\arm directory.

The SYSCRT files will be available in future versions of the kit.

- The incorrect folder was added to the **IncludePath** property for the CRT. As a result, projects using the CRT and STL will fail due to missing header files. To work around this issue, you needed to add the following lines to your VcxProj file:

```
<PropertyGroup>
  <IncludePath>$(WPKInstallDir)WP8\Tools\WPE\CRT\inc\CRT;$(IncludePath)</IncludePath>
</PropertyGroup>
```

## CHANGES IN THE 8297.9664 BUILD

**Pairing note:** NT Build version string: fbl\_core1\_mobile\_dev\_wp\_8297\_0\_120330-0756, Apollo Build: WPMMAIN.8297.9664.20120410-1356

Windows Phone "Apollo" was tested on the following Qualcomm BSPs:

- **8660:** Version 5095, Release date: December 6, 2011
- **8960:** Version: 1.30.120326.1105, Release Date: March 26, 2012

The kit was tested against Visual Studio 11 Beta (Dev11) version 11.0.50214.1 BETAREL.

The following changes and features were new for this release:

- The kit was tested against a newer version of the 8960 BSP.

This release also contained the following known issues and breaking changes:

- During the first boot after an image is created or has been updated, the driver database is refreshed and may cause some drivers not to load if they have not been instantiated properly. Qualcomm and OEM drivers that have been created without INFs need to be modified to work with this kit.
- This release required a patch for KDNet.dll, which contains an issue that caused WinDbg to hang during phone bootup. The patch is specific to the kit and was available as a separate download on Connect.

This issue has been fixed beginning with kit release 8311.xxxx.

- The WDK toolset adds several libraries, such as kernel32.lib, to the dependency list and the WPK fails to remove these libraries. Because these libraries are not supported on Windows Phone "Apollo", this dependency will prevent the user-mode component from loading.

To work around this issue, add the following lines to the end of the project file to override the dependency list:

```
<ItemDefinitionGroup>
  <Link>
    <AdditionalDependencies>mincore.lib;msvcrt.lib;msvcprt.lib;vcomp.lib</AdditionalDependencies>
  </Link>
</ItemDefinitionGroup>
```

- Support for SYSCRT is not available: api-ms-win-core-crt-\*.lib libraries are missing from the WPK. Microsoft recommends linking user-mode services and driver components against SYSCRT instead of the full CRT. However, when support for the full CRT was added to the WPK, the SYSCRT libs

were mistakenly removed. As a result, the SYSCRT libs api-ms-win-core-crt-11.lib and api-ms-win-core-crt-12.lib are not included in the kit.

To work around this issue, use the WPKD as-is and link against the full CRT. The full CRT .libs are installed by default to the C:\Program Files (x86)\Windows Kits\8.0\WP8\Tools\WPE\CRT\lib\arm directory.

The SYSCRT files will be available in future versions of the kit.

- The incorrect folder was added to the **IncludePath** property for the CRT. As a result, projects using the CRT and STL will fail due to missing header files. To work around this issue, you needed to add the following lines to your VcxProj file:

```
<PropertyGroup>
  <IncludePath>$(WPKInstallDir)WP8\Tools\WPE\CRT\inc\CRT;$(IncludePath)</IncludePath>
</PropertyGroup>
```

## CHANGES IN THE 8297.9657 BUILD

**Pairing note:** NT Build version string: fbl\_core1\_mobile\_dev\_wp\_8297\_0\_120330-0756, Apollo Build: WPMMAIN.8297.9657.20120403-1543

Windows Phone "Apollo" was tested on the following Qualcomm BSPs:

- **8660:** Version 5095, Release date: December 6, 2011
- **8960:** Version: 1.30.120305.0000 (1103.0002), Release Date: March 5, 2012

The kit was tested against Visual Studio 11 Beta (Dev11) version 11.0.50214.1 BETAREL.

The following changes and features were new for this release:

- **<SHELLSKIPOOBE>** was no longer a required entry in the OEMInput.xml file.
- A new debugging application (debugoutput.exe) and application extension (Decode.dll) were available with this release. The application, extension, and information on how to run the tool were available in the **Tools** folder.
- The instructions for specifying a test certificate for package signing has been moved to the See the *Specifying a test certificate for package signing* topic in the Windows Phone "Apollo" Partner Documentation. If you do not have access to this documentation and need to review the steps, see the release notes for 8297.9650.

This release also contained the following known issues and breaking changes:

- Timeouts on inter-task and intra-task page navigation commands were enabled as part of the navigation subsystem. The default timeout value is 60 seconds. If clients do not complete the operation within this amount of time, the process will be terminated. As Windows Phone "Apollo" stabilizes, the default timeout will be continually reduced up to only a few seconds. Also, clients responding to navigation commands with an invalid response will be terminated.

To avoid your processes from being terminated, clients must complete navigation commands within the timeout period.

- Metabase and its corresponding configuration component are no longer needed and have been removed from the image. If your configuration service provider tries to retrieve metadata from Metabase, it will not work.

OEMs can no longer use the Metabase configuration service provider to add or set security access rights to settings. You need to rely on the Windows Phone “Apollo” security app container model to retrieve access to settings.

- Some APIs exported by have been refactored. Specifically, the following changes have been made:
  - **api-ms-win-security-base-l1-1-0.dll** was renamed to **api-ms-win-security-base-l1-1-1.dll**
  - **api-ms-win-eventing-classicproviderlegacy-l1-1-0.dll** was renamed to **api-ms-win-eventing-legacy-l1-1-0.dll**
  - **NeedCurrentDirectoryForExePath** APIs were moved from **api-ms-win-core-path-l1-1-0.dll** to **api-ms-win-core-processenvironment-l1-1-1.dll**

For OEMs, no public APIs were removed, but the renamed DLLs require that you re-link against the new .lib versions.

- Support for SYSCRT is not available: **api-ms-win-core-crt-\*.lib** libraries are missing from the WPK. Microsoft recommends linking user-mode services and driver components against SYSCRT instead of the full CRT. However, when support for the full CRT was added to the WPK, the SYSCRT libs were mistakenly removed. As a result, the SYSCRT libs **api-ms-win-core-crt-11.lib** and **api-ms-win-core-crt-12.lib** are not included in the kit.

There are two workarounds for this issue:

- Copy the **api-ms-win-core-crt-\*.lib** files from the previous kit and modify the project file(s) to manually link against these .libs, or
- Use the WPK as-is and link against the full CRT. The full CRT .libs are installed by default to the **C:\Program Files (x86)\Windows Kits\8.0\WP8\Tools\WPE\CRT\lib\arm** directory.

The SYSCRT files will be available and used by default in future versions of the kit.

- The incorrect folder was to the **IncludePath** property for the CRT. As a result, projects using the CRT and STL will fail due to missing header files. To work around this issue, you needed to add the following lines to your VcxProj file:

```
<PropertyGroup>
  <IncludePath>$(WPKInstallDir)WP8\Tools\WPE\CRT\inc\CRT;$(IncludePath)</IncludePath>
</PropertyGroup>
```

## CHANGES IN THE 8297.9650 BUILD

**Pairing note:** NT Build version string: **fbl\_core1\_mobile\_dev\_8297\_0\_120314-1700**, Apollo Build: **WP8\_MOBILECORE.8297.9650.20120325-2215**


Windows Phone “Apollo” was tested on the following Qualcomm BSPs:

- **8660:** Version 5095, Release date: December 6, 2011
- **8960:** Version: 1.30.120305.0000 (1103.0002), Release Date: March 5, 2012

The kit was tested against Visual Studio 11 Beta (Dev11) version 11.0.50214.1 BETAREL.


The following changes and features were new for this release:

- This build contained a new compiler build, encryption on KDNET KDNET transport, and fixes for several breaking changes.
- New kit and symbol installers were added.
- New input file format for generating images. For more information, see the *OEMInput file contents* and *How to: Build a phone image* topics in the Windows Phone Partner Documentation.
- Beginning with this release, you can now see the new Windows Phone “Apollo” UI when you build and flash a Windows Phone image.

 If you need to release Windows Phone images externally, you must turn off the new UI elements. To do this, remove the following line from the ProductionOEMInput.xml file or from TestOEMInput.xml file:

```
<OptionalFeature>SHELLSTART8</OptionalFeature>
```

- Recommendation to group packages by feature area for clarity.

 The following example is not an exhaustive list of the types of packages you need to build your phone image. Please consult with Qualcomm for the full list of packages for a particular BSP.

```
<PackageFiles>
<!--Insert OEM/SoC packages here -->
<!-- SV DMA HAL Extension -->
<!-- SV Graphics -->
<!-- Qualcomm WDDM DirectX Drivers -->
<!-- Video Processing Engine (Video Accelerator) -->
<!-- SV Camera -->
<!-- SV Audio -->
<!-- SV RIL -->
<!-- SV Network -->
<!-- OEM Device Identification -->
<!-- SV UEFI\Bootloader -->
<!-- SV\OEM Bus -->
<!-- Hardware Configuration for I2C and SPI -->
<!-- OEM Battery and Power -->
<!-- OEM Security\Crypto -->
<!-- OEM Platform Specific -->
<!-- ACPI Features -->
<!-- Qualcomm Crash Dump Injector Device -->
<!--... -->
</PackageFiles>
```

This release also contained the following known issues and breaking changes:

- Package Generation now requires catalog files to be signed. To perform this task, a certificate must be specified for use with pkggen.exe. If a certificate is not provided, usage of pkggen.exe will return an error message.

Steps on how to specify a signing certificate for pkggen.exe were provided. This information is now documented in the *Specifying a test certificate for package signing* topic in the Windows Phone "Apollo" Partner Documentation.

- Several example SKU files were missing from the main kit and were made available in a separate .zip file that also went out as part of the release.
- <SHELLSKIPOOBE>, which was marked as an optional feature in the OEMInput.xml file could not be removed or the phone will not boot. This is no longer a requirement as of build 8297.9657.
- When using the MobileCore production image for the Fluid 8660, the system may crash during OOBE even when you used the image with <SHELLSKIPOOBE>. To work around this issue, In the **Set date/time** screen, make sure that the **Send current location to MSFT** check box is clear or not selected, and then reboot the device.
- The incorrect folder was to the **IncludePath** property for the CRT. As a result, projects using the CRT and STL will fail due to missing header files. To work around this issue, you needed to add the following lines to your VcxProj file:

```
<PropertyGroup>
  <IncludePath>$(WPDKInstallDir)WP8\Tools\WPE\CRT\inc\CRT;$(IncludePath)</IncludePath>
</PropertyGroup>
```

- A new input file format will be used with ImgGen.cmd, replacing the UOSInput.xml file that was used previously. When using the new input files, OEMs specify a set of properties that are used by the imaging tool to automatically determine what type of image to generate and which Microsoft packages to include in the image. OEMs also specify a list of all the SV and OEM packages to include in the image.

If you are creating images, you now have to create an OEMInput.xml file instead of a UOSInput.xml file.

- For Windows Phone "Apollo", the language models are now part of optional packages that OEMs may include at their discretion. Most keyboards are usable without these packages, but they will not have the advantage of features such as auto-correction and hit-target resizing.

This new model only supports EN-US currently. When more language models are supported, OEMs must specify the language models to install on the device. This is done during the generation of the new answer file.

- Support for SYSCRT is not available: api-ms-win-core-crt-\*.lib libraries are missing from the WPK. Microsoft recommends linking user-mode services and driver components against SYSCRT instead of the full CRT. However, when support for the full CRT was added to the WPK, the SYSCRT libs were mistakenly removed. As a result, the SYSCRT libs api-ms-win-core-crt-11.lib and api-ms-win-core-crt-12.lib are not included in the kit.

There are two workarounds for this issue:

- Copy the api-ms-win-core-crt-\*.lib files from the previous kit and modify the project file(s) to manually link against these .libs, or



- Use the WPK as-is and link against the full CRT. The full CRT .libs are installed by default to the C:\Program Files (x86)\Windows Kits\8.0\WP8\Tools\WPE\CRT\lib\arm directory.

The SYSCRT files will be available and used by default in future versions of the kit.

- In this release of the kit, only SHA-256 signing is supported. When signing the boot critical drivers using the command line SignTool, specify the **/fd sha256** option to specify the SHA-256 file digest algorithm. For more information about test signing a driver using SignTool, see the MSDN article [Test-Signing a Driver File](#).
- Preboot crash dump (or offline crash dump) on the Qualcomm 8960 Fluid has been enabled. This feature collects system information after an abnormal device reset and is intended to be a fallback mechanism for collecting debug information if the application processor hangs, which would prevent Windows crash dump generation.

You can disable this feature by setting

**HKLM\System\ControlSet001\Control\CrashControl\WpDisablePrebootCrashDump** to 1.

- The resolution for Windows Phone bug 80728 (Device shows up as an unrecognized/unknown device after booting to MainOS (KDNET/USB)) partially includes convergence work necessary to align with the Windows 8 KDNET extensibility model. As a result, if you are using KDNET Ethernet on Fluid 8660 you now have to set busparams=2.
- An app chamber is created for configuration using a ProvXML file during device cold boot. This means that any configuration service provider that is designed to be used in a ProvXML file should have a capability associated with it, and the capability must be added to the app chamber's allow list.

The ProvXML file has been moved to the following location:

Previous location	New location
%SystemRoot%\System32\Provisioning\OEM	%SystemDrive%\Programs\PhoneProvisioner_OEM\OEM

In addition to this, the **\$(runtime.coldBootProvxmlOEM)** macro is now defined to be used for the ProvXML folder in the package definition. This change was a necessary part of the update to the new Windows Phone "Apollo" app chamber-based security model. Some configuration through ProvXML during cold boot may fail if the security capability is not set properly.

OEMs should put the ProvXML file under the

%SystemDrive%\Programs\PhoneProvisioner\_OEM\OEM directory. If the OEM ProvXML does not work, contact Microsoft to coordinate additional capabilities for the app chamber's allow list.

- ConfigManager2 has been updated to conform to the Windows Phone "Apollo" security model, and ConfigManager clients that run in an app container or a service chamber must now include the ID\_CAP\_CSP\_FOUNDATION capability.

With this change, ConfigManager2 is now more secure and supports recover rollback, which is the restoration of settings left in an indeterminate state by abnormal process termination or device reboot. ConfigManager clients running in an app container or a non-TCB service chamber will get access-denied errors if the proper capabilities are not included.

OEM apps that call the **DMPProcessConfigXML** function will now run in an app container and should include the proper capability in their allow list.

## CHANGES IN THE 8283.9632 BUILD

**Pairing note:** NT Build version string: fbl\_core1\_mobile\_dev\_8283\_0\_120216-1700, Apollo Build: WP8\_PD.8283.9632.20120306-1842

Windows Phone "Apollo" was tested on the following Qualcomm BSPs:

- **8660:** Version 5095
- **8960:** Version: 1.30.120130 (1102.0005)

The following changes and features were new for this release:

- The version of Visual Studio 11 Beta that you need to install has been clarified in the *System requirements* and *Installation and uninstallation* sections.
- An updated version of Windows Phone Test Central (WPTC) is now available. To install Test Central, navigate to the **Tools** folder, and double-click WPTC\_MC-arm-fre.msi.
- Windows Phone "Apollo" kit and Qualcomm BSP scorecard (8283.9632 Apollo Kit and Qualcomm BSP Scorecard.pdf), which included some corrected entries.

This release also contained the following known issues and breaking changes:

- Visual Studio 2010 was linking to a legacy version of CRT. When the WDK moved to Visual Studio 11 Beta, the link was updated to the newer, supported version of the CRT. However, when the WSDK was moved to Visual Studio 11 Beta, support for the old CRT remained, but the CRT is incompatible with the STL. Support for the newer CRT and compatibility with the STL will be available in the next Windows Phone "Apollo" kit.

## CHANGES IN THE 8283.9628 BUILD

**Pairing note:** NT Build version string: fbl\_core1\_mobile\_dev\_8283\_0\_120216-1700, Apollo Build: WP8\_PD.8283.9628.20120229-2156

Windows Phone "Apollo" was tested on the following Qualcomm BSPs:

- **8660:** Version 5095
- **8960:** Version: 1.30.120130 (1102.0005)

The following changes and features were new for this release:

- **Visual Studio 11 Beta is now required to build kernel-mode drivers and user-mode components.** See [Installation and uninstallation](#) for information about where you can download Visual Studio 11 Beta.

The Windows Phone "Apollo" Partner Documentation released on March 5, 2012 contains relevant instructions for using Visual Studio 11 Beta to create kernel-mode drivers and user-mode components.

- Fixed a typo in the *Pairing note* section of the release notes.

- The Windows Phone Driver Kit (WPKD), Windows Phone Adaptation Kit (WPAK), and Windows Phone Test Central (WPTC) installers have been renamed.
  - WPKD.msi is now WPKD\_MC-arm-fre.msi
  - WPAK.msi is now WPKD\_MC-arm-fre.msi
  - WPTC.msi is now WPTC\_MC-arm-fre.msi
- The Windows Phone "Apollo" kit now includes the sensor class extension library `wpsensorclxstub.lib`, which is required to pick up the exports you need to link to the sensor class extension. This addresses link errors caused by the missing library file.

This release also contained the following known issues and breaking changes:

- You cannot use the new linker to create ARM images with a subsystem version earlier than 6.02. If your project specifies a subsystem version, you may see linker warnings and build breaks, as in the following example:

```
warning LNK4010: invalid subsystem version number <version>
```

If you see the linker warning, stop specifying a subsystem version in project files. If you are using the **LINKER\_SUBSYSTEM** variable in your project to specify the CONSOLE subsystem, use **UMTYPE** instead. For example, replace the following:

```
LINKER_SUBSYSTEM=/SUBSYSTEM:CONSOLE,6.00
```

With:

```
UMTYPE=console
```

Note that the latter statement is case-sensitive.

- The way the new Dev11 compiler (ARM-only) issues warnings about potentially uninitialized variables is different from the previous compiler. This might result in new compiler warnings and build breaks.

```
warning C4701: potentially uninitialized local variable '<identifier>' used
```

If you see the compiler warning, add initializers to the variable(s) detailed in the warning(s). For example, change the following:

```
int * pPointer;
```

To:

```
int * pPointer = NULL;
```

It is good practice to always initialize local variables, especially pointers; unnecessary initialization usually does not measurably affect generated code quality.

- The FFUTool used to flash devices with .ffu images now enforces a check where the image being flashed must match the target device. New device images include an embedded device ID and an on-device file that contains the same ID. These values will be compared by the FFUTool when

flashing to confirm that the image is appropriate for the device. When the ID values do not match, the FFUTool fails and displays a message. If the ID values match, flashing proceeds normally.

Existing images currently are not provisioned with the correct device ID. If you try to flash a new image using the new version of FFUTool, an error occurs. The Windows Phone Team has added a **-force** option to FFUTool with which you can override the check behavior. If you have a lab automation setup, this change does not affect that because the lab does not use FFUTool directly. Using the **-force** option is necessary during this transition for certain platforms. You may also need to use the **-force** option when flashing an earlier image, made before this change, with a newer version of FFUTool.

In the following flashing scenarios, using the **-force** flag is safe:

- **Dogphone** — Device name is either "Nokia..Dogphone." or "Qualcomm Snapdragon.Snapdragon Product Family.Snapdragon Product."
  - Earlier safe image platform ID: "{6ACA3782-8388-4f9c-996C-508FB42AF391}"
  - Most recent safe image platform ID: "Nokia..Dogphone."
- **8960 Fluid** — Device name is either "Qualcomm.MSM8960.MSM8960 Fluid." or "Qualcomm Snapdragon.Snapdragon Product Family.Snapdragon Product."
  - Earlier safe image platform ID: "{39D1D2F2-4EBD-4A8D-8683-A538F9FD0A3A}"
  - Most recent safe image platform ID: "Qualcomm.MSM8960.MSM8960 Fluid."
- **8660 Fluid** — Name "Qualcomm.MSM8660.MSM8660 Fluid."
  - Earlier safe image has a blank ID string. The tool will show "... , image targets: ."
  - Most recent safe image platform ID: "Qualcomm.MSM8660.MSM8660 Fluid."
- A ProvXML file creates two configuration app chamber files during device cold boot:
  - One for processing an OEM XML file
  - One for processing a Microsoft XML file

This means that configuration service providers (CSPs) that are designed to be used in ProvXML file should have a capability associated with them, and the capability should be added to the PhoneProvisioner's app chamber's allow list.

Some CSP configuration using ProvXML during cold boot might fail if the security capability is not properly set. OEMs should use only the CSPs that have capability listed in the phone provisioner's OEM app chamber's allow list when you create the OEM ProvXML file to be used in cold boot.

- The generic USB function class driver (GenericUsbFnClass.sys) will no longer activate the USB bus. Because Nokia uses the generic USB function class driver, activating the bus is a step that can now be skipped when using the generic driver, although the current code allows this.

GenericUsbFnClass.sys previously issued the activate bus control code during bootup, which made the USB available when drivers were loaded. With this change, the generic driver is now more transparent, and user-mode code now has to activate the bus in the same way that kernel-mode code does.

- Windows Phone "Apollo" sensor native API users now need to make an API call to run in connected standby mode. This change does not apply to Windows Runtime (WinRT) or to managed runtime. For the sensor to run in idle, the sensor needs to call the **SensorEnableIdleOperation** function. This function is documented in the Windows Phone "Apollo" Partner Documentation.
- When going through OOBE on the 8960, you may encounter the following issues:

1. The Time Zone screen does not allow scrolling
2. There are no options in the "What's your home region?" section.

These issues are minor and do not prevent the completion of OOBE. Windows Phone "Apollo" bug 70882 is tracking this work item.

## CHANGES IN THE 8186.9618 BUILD

**Pairing note:** NT Build version string: fbl\_core1\_mobile\_dev\_wp\_8186\_0\_120129-2040, Apollo Build: 9618

Windows Phone "Apollo" was tested on the following Qualcomm BSPs:

- **8660:** Version 5095
- **8960:** Version 1102

This release also contained the following known issues and breaking changes:

- A sensor library, wpsensorclxstub.lib, required to link to the sensor class extension, is missing from the kit. Windows Phone "Apollo" bug 78727 is tracking this work item and the library will be formally added to the kit in a future release.
- When going through OOBE on the 8960, you may encounter the following issues:
  1. The Time Zone screen does not allow scrolling
  2. There are no options in the "What's your home region?" section.

These issues are minor and do not prevent the completion of OOBE. Windows Phone "Apollo" bug 70882 is tracking this work item.

## CHANGES IN THE 8186.9612 BUILD

**Pairing note:** NT Build version string: fbl\_core1\_mobile\_dev\_wp\_8186\_0\_120129-2040, Apollo Build: 9612

Windows Phone "Apollo" was tested on the following Qualcomm BSPs:

- **8660:** Version 5095
- **8960:** Version 1102

The following changes and features were new for this release:

- This release required an update to the MobileCore symbols.
- Cellular COM API reference has been removed. The cellular interface is being reengineered and information about a replacement API will be provided in a future release of the Windows Phone "Apollo" Partner Documentation.
- This release includes new hardware abstraction layer (HAL) timer extensions when you build an image.

This release also contained the following known issues and breaking changes:

- A sensor library, wpsensorclxstub.lib, required to link to the sensor class extension, is missing from the kit. Windows Phone "Apollo" bug 78727 is tracking this work item and the library will be formally added to the kit in a future release.
- When going through OOBE on the 8960, you may encounter the following issues:
  1. The Time Zone screen does not allow scrolling
  2. There are no options in the "What's your home region?" section.

These issues are minor and do not prevent the completion of OOBE.

## CHANGES IN THE 8186.9604 BUILD

**Pairing note:** NT Build version string: fbl\_core1\_mobile\_dev\_8186\_0\_120118-1706, Apollo Build: 9604

Windows Phone "Apollo" was tested on the following Qualcomm board support packages (BSPs):

- **8660:** Version 5095
- **8960:** Version 1100

The following changes and features were new for this release:

- A new package, Microsoft.HAL.TIMEREXT.spkg, that contains HAL timer extensions was released to unblock development on 8960 V3 hardware. The package is intended to replace the package included in the image. For this release, you have to manually add the package to the UOSInput.xml file to include the contents of the package in an image.
- A preliminary build of Windows Phone Test Central is available with this release. The initial set of tests is a subset of the **Suite 1 – Chassis** tests. Preliminary Test Central documentation was also available in this release.
- A preliminary version of Tungsten, a Windows desktop tool designed to analyze an event trace log (ETL) collected from Windows Phones, is available in this release. The Tungsten *Getting Started Guide*, which contains information about Tungsten and how to use it, is also available.
- A preliminary EFI battery charging app was added to the kit. The app was available through the Microsoft.firmware.batt\_soc100\_950mA.spkg package, which had to be manually added to the UOSInput.xml file (in this release) to include the contents of the package in an image.

This release also contains the following known issues and breaking changes:

- A sensor library, wpsensorclxstub.lib, which is required to link to the sensor class extension, is missing from the kit. Windows Phone "Apollo" bug 78727 is tracking this work item and the library will be formally added to the kit in a future release.
- Using EFI drop 1060 for 8960 Fluid, if you press and hold the **Volume Up** button during boot, FFUapp does not open. This was being tracked by bug 42547, but it has since been resolved as By Design.

## CHANGES IN THE 8175.9592 BUILD

**Pairing note:** NT Build version string: fbl\_core1\_mobile\_dev\_8175\_0\_120101-1600, Apollo Build: 9592

Windows Phone "Apollo" was tested on the following Qualcomm BSPs:

- **8660:** Version 5095
- **8960:** Version 1070

The following changes and features were new for this release:

- A package, Microsoft.BaseOS.CodeIntegrity.spkg, which disables Code Integrity so that you can use imagines to run unsigned code, is available. The change is temporary until Windows Phone has a system that can sign binaries in Windows Phone "Apollo". For this release, you must manually add the package to the UOSInput.xml file to include the contents of the package in an image.

- A preliminary UEFI battery charging app was added to the kit. The app was available through the Microsoft.firmware.batt\_soc100\_950mA.spkg package, which had to be manually added to the UOSInput.xml file (in this release) to include the contents of the package in an image.
- Storedisk images are no longer provided. Going forward, full flash update (FFU) image generation is the expected flashing method.
- Preboot crash dump on Qualcomm 8660 Fluid has been enabled. This change was introduced to help debug hard system hangs. Using the preboot crash dump, you can debug subsystem memory errors after an abnormal device reset. This feature can be disabled by setting **HKLM\System\ControlSet001\Control\CrashControl\WpDisablePrebootCrashDump** to 1.

This release also contained the following known issues and breaking changes:

- When flashing devices using 8175.9592 and the Qualcomm 8960 BSP, the device might fail to boot with an error and indicate that the device might need to reboot. Analysis of the memory dump will show a bugcheck of type PANIC\_STACK\_SWITCH. If you encounter this issue, please contact Qualcomm for assistance.
- Using UEFI drop 1060 for 8960 Fluid, if you press and hold the **Volume Up** button during boot, FFUapp does not open. An active bug that is tracking this issue has been assigned to Qualcomm.
- Some APIs in mincore.lib have moved to other libs, including the following:
  1. **DevObj** APIs have been removed.
  2. The **RegisterWaitForSingleObjectEx**, **SetThreadpoolTimerEx**, and **SetThreadpoolWaitEx** APIs have been removed.

This change was made so that mincore.lib will only export documented APIs on MSDN or on documented import libs. You should transition to a suitable replacement in mincore.lib for any API that is missing. If you have no suitable alternative, please contact the Windows Phone team.

## CHANGES IN THE 8141.9571 BUILD

**Pairing note:** NT Build version string: 8141.0.armchk.fbl\_core1\_mobile\_dev.111102-1406, Apollo Build: 9571

Windows Phone "Apollo" was tested against the following Qualcomm BSPs:

- **8660:** Version 508002, released November 1, 2011
- **8960:** Version 1.20.111031 (1070), released October 31, 2011

The following changes and features were new for this release:

- Windows Phone "Apollo" CHK and FRE storedisk images.
- FRE MS packages, which can be used to build an FFU image.
- FFU flashing tools compatible with the 8660 and the 8960 Fluid development platforms.
- Matching NT public symbols.
- Matching Apollo symbols for graphics subsystem.
- Matching Mobile SDK.
- The UOSInput.xml file has been fixed to address the image generation error from the previous release. You no longer have to use the patched UOSInput.xml file that shipped with the previous release.

- This is the last release expected to include a storedisk OS image. Qualcomm is already shipping packages for its binaries (or will be shipping these very soon) and partners are expected to use packages and to build only their own images.
- ChassisTests.zip, which contains DLLs for the available Windows Phone "Apollo" chassis tests.

This release also contained the following known issues and breaking changes:

- Known issue with the debugger not connecting to the device when using KDNet over USB. The workaround is to use a USB2 connection instead, and then to follow the procedure in the Windows Phone "Apollo" Partner Documentation topic *How to: Set up a USB connection for kernel-mode debugging*.
- There is no built-in support for creating a new kernel-mode driver project in Visual Studio 2010 with this build of the WDK and WPK. To work around these limitations, open a sample KMDF project in VS2010, and then customize the project to fit your needs. For more information about the available sample KMDF projects and where to find them, see [KMDF samples](#). Driver project files have the .vcxproj extension. For more information about building driver projects, see [Building a Driver](#) and the BuildEnvironment.docx document that accompanies WDK.
- A floppy drive in a development workstation causes an issue with creating the volume tracker list when you try to build an image. This has been fixed in later builds.
- The provided storedisk images in this release do not show the Windows Phone UI. To interact with the phone, you can use TShell and other tools. This is not an issue for a FFU images.
- Random bug checks are seen during boot:PHASE1\_INITIALIZATION\_FAILED(32) on 8960 Fluid devices. Rebooting the device temporarily fixes the issue.
- OOBE could not be successfully completed on the 8960 Fluid platform. A workaround for this issue is to press the **Back** or **Start** button to leave the OOBE screen, or to reboot the device.
- Using UEFI drop 1060 for the 8960 Fluid, the reset command does not work. Qualcomm is aware of the issue, and expects to include a fix in a future Qualcomm BSP drop.
- Using UEFI drop 1060 for the 8960 Fluid, holding the camera button during boot does not open FFUapp. This behavior is inconsistent with the experience on the 8660 Fluid and an active bug has been assigned to Qualcomm.

## CHANGES IN THE 8141.9559 BUILD

**Pairing note:** NT Build version string: 8141.0.armchk.fbl\_core1\_mobile\_dev.111102-1406, Apollo Build: 9559

Beginning with this release, there are officially two main kits for Windows Phone, with separate installers. The Windows Phone kits are:

- **Windows Phone Driver Kit (WPK)**
- **Windows Phone Adaptation Kit (WPAK)**

For more information about how to set up your development computer, and to learn more about the contents of these kits, see *Getting started* in the WP\_Apollo\_Documentation.chm file.

This release also contained the following known issues and breaking changes:



- Known image generation issue with the UOSInput.xml file – The “default” XML file results in an image generation error. Build 8141 includes a patched UOSInput.xml file that you can use to correct the issue in this build. This has been fixed in later builds.
- An ATL\_ASSERT may be triggered when booting. The only workaround is to reboot the device. This has been fixed and is not an issue in later builds.
- The package generator has been updated to validate package project file extensions. Any existing packages that use the new schema and that are not using the .pkg.xml extension will cause build breaks. This change only affects developers who are using package projects with the new schema. A package is using the new schema if “\_NEW\_SCHEMA=1” is specified in the corresponding SOURCES file.
- The **OwnerType** attribute has been changed from optional to required. Any existing package projects that are using the new schema need to include this attribute or the build will fail. You must specify the **OwnerType** attribute in the package project file, with a value of Microsoft, OEM, SiliconVendor, or Operator.

## CHANGES IN THE 8124.9543 BUILD

**Pairing note:** NT Build version string: 8124.0.armchk.fbl\_core1\_mobile\_dev.110929-1700, Apollo Build: 8124.9543.20111019-1808

The 8124 build of the WPKD includes the following changes:

- Packaging, imaging, and flashing tools, and related files. These are installed in %ProgramFiles%\Windows Kits\8.0\WP8\tools\bin\i386 or %ProgramFiles(x86)%\Windows Kits\8.0\WP8\tools\bin\i386.
- Prebuilt driver packages and OS packages for building images. These are installed in %ProgramFiles%\Windows Kits\8.0\WP8\mspackages or %ProgramFiles(x86)%\Windows Kits\8.0\WP8\mspackages.

For more information about the individual files that are new in build 8124, see New Files In Build 8124.txt.

## CHANGES IN THE 8117.9531 BUILD

**Pairing note:** NT Build version string: 8117.0.armchk.fbl\_core1\_mobile\_dev.110919-1700, Apollo Build: 8117.9531.20110930-0947

The 8117 build of the WPKD includes the following changes:

- New kernel-mode headers:
  - wpmbbextensiondef.h
  - WpAmbient.h
  - WpGyro.h
  - WpMagnetometer.h
- Removed kernel-mode headers:
  - usbfnccommon.h was removed because of refactoring. APIs that were in this file are now in usbfnc.h, usbfncx.h, and usbfniocctl.h.

- New user-mode headers and libraries:
  - Rilapitypes.idl
  - Rilapi.h
  - Rilproxy.lib

## CHANGES IN THE 8070.9509 BUILD

**Pairing note:** NT Build version string: 8070.0.armchk.fbl\_core1\_mobile\_dev.110820-1322, Apollo Build: 8070.9509.20110830-2055

The 8070 build of the WPKD includes the following changes:

- New kernel-mode headers and libraries:
  - gnssdriver.h
  - hwn.h
  - hwnclx.h
  - kusbfnclasslib.h
  - usbfh.h
  - usbfhclx.h
  - usbfhcommon.h
  - usbfhioctl.h
  - WpAccelerometer.h
  - WpCrDmp.h
  - WpOrientation.h
  - WpProximity.h
  - WpSensors.h
  - mshwnclxstub.lib
  - Orientation.lib
  - wpsensors.lib

## COPYRIGHT INFORMATION

---

This document supports a preliminary release of a software product that may be changed significantly before final commercial release. This document is provided for informational purposes only and Microsoft makes no warranties, either express or implied, in this document. Information in this document, including URL and other Internet Web site references, is subject to change without notice. The entire risk of the use or the results from the use of this document remains with the user. 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.

©2012 Microsoft. All rights reserved. Microsoft, Windows, and Zune are trademarks of the Microsoft group of companies.