

Table of Contents

Configure Windows 10

Configure Windows telemetry in your organization

Windows 10, version 1709 basic level Windows diagnostic events and fields

Windows 10, version 1709 enhanced telemetry events and fields used by Windows Analytics

Windows 10, version 1703 basic level Windows diagnostic events and fields

Windows 10 diagnostic data for the Full telemetry level

Beginning your General Data Protection Regulation (GDPR) journey for Windows 10

Manage connections from Windows operating system components to Microsoft services

Manage Wi-Fi Sense in your company

Configure kiosk and shared devices running Windows 10 desktop editions

Set up a shared or guest PC with Windows 10

Set up a kiosk on Windows 10 Pro, Enterprise, or Education

Guidelines for choosing an app for assigned access (kiosk mode)

Create a Windows 10 kiosk that runs multiple apps

Configure Windows 10 Mobile devices

Set up a kiosk on Windows 10 Mobile or Windows 10 Mobile Enterprise

Use Windows Configuration Designer to configure Windows 10 Mobile devices

Use the Lockdown Designer app to create a Lockdown XML file

Configure Windows 10 Mobile using Lockdown XML

Settings and quick actions that can be locked down in Windows 10 Mobile

Product IDs in Windows 10 Mobile

Start layout XML for mobile editions of Windows 10 (reference)

Configure cellular settings for tablets and PCs

Configure Start, taskbar, and lock screen

Configure Windows Spotlight on the lock screen

Manage Windows 10 and Microsoft Store tips, tricks, and suggestions

Manage Windows 10 Start and taskbar layout

Cortana integration in your business or enterprise

Testing scenarios using Cortana in your business or organization

Set up and test Cortana with Office 365 in your organization

Set up and test Cortana with Microsoft Dynamics CRM (Preview feature) in your organization

Set up and test Cortana for Power BI in your organization

Set up and test custom voice commands in Cortana for your organization

Use Group Policy and mobile device management (MDM) settings to configure Cortana in your organization

Send feedback about Cortana at work back to Microsoft

Configure access to Microsoft Store

Provisioning packages for Windows 10

How provisioning works in Windows 10

Introduction to configuration service providers (CSPs)

Install Windows Configuration Designer

Create a provisioning package

Apply a provisioning package

Settings changed when you uninstall a provisioning package

Provision PCs with common settings for initial deployment (desktop wizard)

Provision PCs with apps

Use a script to install a desktop app in provisioning packages

Create a provisioning package with multivariant settings

PowerShell cmdlets for provisioning Windows 10 (reference)

Windows Configuration Designer command-line interface (reference)

Windows Configuration Designer provisioning settings (reference)

Lockdown features from Windows Embedded 8.1 Industry

User Experience Virtualization (UE-V) for Windows

Get Started with UE-V

Prepare a UE-V Deployment

Administering UE-V

Troubleshooting UE-V

Technical Reference for UE-V

Change history for Configure Windows 10

Configure Windows 10

10/17/2017 • 2 min to read • [Edit Online](#)

Enterprises often need to apply custom configurations to devices for their users. Windows 10 provides a number of features and methods to help you configure or lock down specific parts of Windows 10.

In this section

TOPIC	DESCRIPTION
Configure Windows telemetry in your organization	Use this article to make informed decisions about how you can configure Windows telemetry in your organization.
Windows 10, version 1709 basic diagnostic events and fields	Learn about diagnostic data that is collected at the basic level in Windows 10, version 1709.
Windows 10, version 1709 enhanced telemetry events and fields used by Windows Analytics	Learn about diagnostic data that is collected by Windows Analytics.
Windows 10, version 1703 basic diagnostic events and fields	Learn about diagnostic data that is collected at the basic level in Windows 10, version 1703.
Windows 10 diagnostic data for the Full telemetry level	Learn about the types of data that is collected at the full level in Windows 10, version 1703 and later.
Beginning your General Data Protection Regulation (GDPR) journey for Windows 10	Learn about Windows 10 and the upcoming GDPR-compliance requirements.
Manage connections from Windows operating system components to Microsoft services	Learn about the network connections that Windows components make to Microsoft and also the privacy settings that affect data that is shared with either Microsoft or apps and how they can be managed by an IT Pro.
Manage Wi-Fi Sense in your company	Wi-Fi Sense automatically connects you to Wi-Fi, so you can get online quickly in more places. It can connect you to open Wi-Fi hotspots it knows about through crowdsourcing, or to Wi-Fi networks your contacts have shared with you by using Wi-Fi Sense. The initial settings for Wi-Fi Sense are determined by the options you chose when you first set up your PC with Windows 10.
Configure kiosk and shared devices running Windows 10 desktop editions	These topics help you configure Windows 10 devices to be shared by multiple users or to run as a kiosk device that runs a single app.
Configure Windows 10 Mobile devices	These topics help you configure the features and apps and Start screen for a device running Windows 10 Mobile, as well as how to configure a kiosk device that runs a single app.
Configure cellular settings for tablets and PCs	Enterprises can provision cellular settings for tablets and PC with built-in cellular modems or plug-in USB modem dongles.

TOPIC	DESCRIPTION
Configure Start, taskbar, and lock screen	A standard, customized Start layout can be useful on devices that are common to multiple users and devices that are locked down for specialized purposes. Configuring the taskbar allows the organization to pin useful apps for their employees and to remove apps that are pinned by default.
Cortana integration in your business or enterprise	The world's first personal digital assistant helps users get things done, even at work. Cortana includes powerful configuration options specifically to optimize for unique small to medium-sized business and enterprise environments.
Configure access to Microsoft Store	IT Pros can configure access to Microsoft Store for client computers in their organization. For some organizations, business policies require blocking access to Microsoft Store.
Provisioning packages for Windows 10	Learn how to use the Windows Configuration Designer and provisioning packages to easily configure multiple devices.
Lockdown features from Windows Embedded 8.1 Industry	Many of the lockdown features available in Windows Embedded 8.1 Industry have been modified in some form for Windows 10.
Change history for Configure Windows 10	This topic lists new and updated topics in the Configure Windows 10 documentation for Windows 10 and Windows 10 Mobile.

Configure Windows telemetry in your organization

10/17/2017 • 27 min to read • [Edit Online](#)

Applies to

- Windows 10
- Windows 10 Mobile
- Windows Server

At Microsoft, we use Windows telemetry to inform our decisions and focus our efforts in providing the most robust, most valuable platform for your business and the people who count on Windows to enable them to be as productive as possible. Telemetry gives users a voice in the operating system's development. This guide describes the importance of Windows telemetry and how we protect that data. Additionally, it differentiates between telemetry and functional data. It also describes the telemetry levels that Windows supports. Of course, you can choose how much telemetry is shared with Microsoft, and this guide demonstrates how.

To frame a discussion about telemetry, it is important to understand Microsoft's privacy principles. We earn customer trust every day by focusing on six key privacy principles as described at privacy.microsoft.com. These principles guided the implementation of the Windows telemetry system in the following ways:

- **Control.** We offer customers control of the telemetry they share with us by providing easy-to-use management tools.
- **Transparency.** We provide information about the telemetry that Windows and Windows Server collects so our customers can make informed decisions.
- **Security.** We encrypt telemetry in transit from your device and protect that data at our secure data centers.
- **Strong legal protections.** We respect customers' local privacy laws and fight for legal protection of their privacy as a fundamental human right.
- **No content-based targeting.** We take steps to avoid and minimize the collection of customer content, such as the content of files, chats, or emails, through the Windows telemetry system. Customer content inadvertently collected is kept confidential and not used for user targeting.
- **Benefits to you.** We collect Windows telemetry to help provide you with an up-to-date, more secure, reliable and performant product, and to improve Windows for all our customers.

This article applies to Windows and Windows Server telemetry only. Other Microsoft or third-party apps, such as System Center Configuration Manager, System Center Endpoint Protection, or System Center Data Protection Manager, might send data to their cloud services in ways that are inconsistent with this guide. Their publishers are responsible for notifying users of their privacy policies, telemetry controls, and so on. This article describes the types of telemetry we may gather, the ways you might manage it in your organization, and some examples of how telemetry can provide you with valuable insights into your enterprise deployments. Microsoft uses the data to quickly identify and address issues affecting its customers.

Use this article to make informed decisions about how you might configure telemetry in your organization. Telemetry is a term that means different things to different people and organizations. For this article, we discuss telemetry as system data that is uploaded by the Connected User Experience and Telemetry component. The telemetry data is used to help keep Windows devices secure by identifying malware trends and other threats and to help Microsoft improve the quality of Windows and Microsoft services.

We are always striving to improve our documentation and welcome your feedback. You can provide feedback by contacting telmhelp@microsoft.com.

Not finding content you need? Windows 10 users, tell us what you want on [Feedback Hub](#).

Overview

In previous versions of Windows and Windows Server, Microsoft used telemetry to check for updated or new Windows Defender signatures, check whether Windows Update installations were successful, gather reliability information through the Reliability Analysis Component (RAC), and gather reliability information through the Windows Customer Experience Improvement Program (CEIP) on Windows. In Windows 10 and Windows Server 2016, you can control telemetry streams by using the Privacy option in Settings, Group Policy, or MDM.

For Windows 10, we invite IT pros to join the [Windows Insider Program](#) to give us feedback on what we can do to make Windows work better for your organization.

Understanding Windows telemetry

Windows as a Service is a fundamental change in how Microsoft plans, builds, and delivers the operating system. Historically, we released a major Windows version every few years. The effort required to deploy large and infrequent Windows versions was substantial. That effort included updating the infrastructure to support the upgrade. Windows as a Service accelerates the cadence to provide rich updates more frequently, and these updates require substantially less effort to roll out than earlier versions of Windows. Since it provides more value to organizations in a shorter timeframe, delivering Windows as a Service is a top priority for us.

The release cadence of Windows may be fast, so feedback is critical to its success. We rely on telemetry at each stage of the process to inform our decisions and prioritize our efforts.

What is Windows telemetry?

Windows telemetry is vital technical data from Windows devices about the device and how Windows and related software are performing. It's used in the following ways:

- Keep Windows up to date
- Keep Windows secure, reliable, and performant
- Improve Windows – through the aggregate analysis of the use of Windows
- Personalize Windows engagement surfaces

Here are some specific examples of Windows telemetry data:

- Type of hardware being used
- Applications installed and usage details
- Reliability information on device drivers

What is NOT telemetry?

Telemetry can sometimes be confused with functional data. Some Windows components and apps connect to Microsoft services directly, but the data they exchange is not telemetry. For example, exchanging a user's location for local weather or news is not an example of telemetry—it is functional data that the app or service requires to satisfy the user's request.

There are subtle differences between telemetry and functional data. Windows collects and sends telemetry in the background automatically. You can control how much information is gathered by setting the telemetry level. Microsoft tries to avoid collecting personal information wherever possible (for example, if a crash dump is collected and a document was in memory at the time of the crash). On the other hand, functional data can contain personal information. However, a user action, such as requesting news or asking Cortana a question, usually triggers collection and transmission of functional data.

If you're an IT pro that wants to manage Windows functional data sent from your organization to Microsoft, see [Manage connections from Windows operating system components to Microsoft services](#).

The following are specific examples of functional data:

- Current location for weather
- Bing searches
- Wallpaper and desktop settings synced across multiple devices

Telemetry gives users a voice

Windows and Windows Server telemetry gives every user a voice in the operating system's development and ongoing improvement. It helps us understand how Windows 10 and Windows Server 2016 behaves in the real world, focus on user priorities, and make informed decisions that benefit them. For our enterprise customers, representation in the dataset on which we will make future design decisions is a real benefit. The following sections offer real examples of these benefits.

Drive higher app and driver quality

Our ability to collect telemetry that drives improvements to Windows and Windows Server helps raise the bar for app and device driver quality. Telemetry helps us to quickly identify and fix critical reliability and security issues with apps and device drivers on given configurations. For example, we can identify an app that hangs on devices using a specific version of a video driver, allowing us to work with the app and device driver vendor to quickly fix the issue. The result is less downtime and reduced costs and increased productivity associated with troubleshooting these issues.

Real-world example of how Windows telemetry helps

There was a version of a video driver that was crashing on some devices running Windows 10, causing the device to reboot. We detected the problem in our telemetry, and immediately contacted the third-party developer who builds the video driver. Working with the developer, we provided an updated driver to Windows Insiders within 24 hours. Based on telemetry from the Windows Insiders' devices, we were able to validate the new version of the video driver, and rolled it out to the broad public as an update the next day. Telemetry helped us find, fix, and resolve this problem in just 48 hours, providing a better user experience and reducing costly support calls.

Improve end-user productivity

Windows telemetry also helps Microsoft better understand how customers use (or do not use) the operating system's features and related services. The insights we gain from this data helps us prioritize our engineering effort to directly impact our customers' experiences. Examples are:

- **Start menu.** How do people change the Start menu layout? Do they pin other apps to it? Are there any apps that they frequently unpin? We use this dataset to adjust the default Start menu layout to better reflect people's expectations when they turn on their device for the first time.
- **Cortana.** We use telemetry to monitor the scalability of our cloud service, improving search performance.
- **Application switching.** Research and observations from earlier Windows versions showed that people rarely used Alt+Tab to switch between applications. After discussing this with some users, we learned they loved the feature, saying that it would be highly productive, but they did not know about it previously. Based on this, we created the Task View button in Windows 10 to make this feature more discoverable. Later telemetry showed significantly higher usage of this feature.

These examples show how the use of telemetry data enables Microsoft to build or enhance features which can help organizations increase employee productivity while lowering help desk calls.

Insights into your own organization

Sharing information with Microsoft helps make Windows and other products better, but it can also help make your internal processes and user experiences better, as well. Microsoft is in the process of developing a set of analytics customized for your internal use. The first of these, called [Upgrade Readiness](#).

Upgrade Readiness

Upgrading to new operating system versions has traditionally been a challenging, complex, and slow process for many enterprises. Discovering applications and drivers and then testing them for potential compatibility issues have been among the biggest pain points.

To better help customers through this difficult process, Microsoft developed Upgrade Readiness to give enterprises the tools to plan and manage the upgrade process end to end and allowing them to adopt new Windows releases more quickly and on an ongoing basis.

With Windows telemetry enabled, Microsoft collects computer, application, and driver compatibility-related information for analysis. We then identify compatibility issues that can block your upgrade and suggest fixes when they are known to Microsoft.

Use Upgrade Readiness to get:

- A visual workflow that guides you from pilot to production
- Detailed computer, driver, and application inventory
- Powerful computer level search and drill-downs
- Guidance and insights into application and driver compatibility issues with suggested fixes
- Data driven application rationalization tools
- Application usage information, allowing targeted validation; workflow to track validation progress and decisions
- Data export to commonly used software deployment tools

The Upgrade Readiness workflow steps you through the discovery and rationalization process until you have a list of computers that are ready to be upgraded.

How is telemetry data handled by Microsoft?

Data collection

Windows 10 and Windows Server 2016 includes the Connected User Experience and Telemetry component, which uses Event Tracing for Windows (ETW) tracelogging technology that gathers and stores telemetry events and data. The operating system and some Microsoft management solutions, such as System Center, use the same logging technology.

1. Operating system features and some management applications are instrumented to publish events and data. Examples of management applications include Virtual Machine Manager (VMM), Server Manager, and Storage Spaces.
2. Events are gathered using public operating system event logging and tracing APIs.
3. You can configure the telemetry level by using MDM policy, Group Policy, or registry settings.
4. The Connected User Experience and Telemetry component transmits the telemetry data.

Info collected at the Enhanced and Full levels of telemetry is typically gathered at a fractional sampling rate, which can be as low as 1% of devices reporting data at those levels.

Data transmission

All telemetry data is encrypted using SSL and uses certificate pinning during transfer from the device to the Microsoft Data Management Service. With Windows 10, data is uploaded on a schedule that is sensitive to event priority, battery use, and network cost. Real-time events, such as Windows Defender Advanced Threat Protection, are always sent immediately. Normal events are not uploaded on metered networks, unless you are on a metered server connection. On a free network, normal events can be uploaded every 4 hours if on battery, or every 15 minutes if on A/C power. Diagnostic and crash data are only uploaded on A/C power and free networks.

Endpoints

The Microsoft Data Management Service routes data back to our secure cloud storage. Only Microsoft personnel with a valid business justification are permitted access.

The following table defines the endpoints for telemetry services:

SERVICE	ENDPOINT
Connected User Experience and Telemetry component	v10.vortex-win.data.microsoft.com settings-win.data.microsoft.com
Windows Error Reporting	watson.telemetry.microsoft.com
Online Crash Analysis	oca.telemetry.microsoft.com
OneDrive app for Windows 10	vortex.data.microsoft.com/collect/v1

Data use and access

The principle of least privileged access guides access to telemetry data. Microsoft does not share personal data of our customers with third parties, except at the customer's discretion or for the limited purposes described in the [Privacy Statement](#). Microsoft may share business reports with OEMs and third-party partners that include aggregated and anonymized telemetry information. Data-sharing decisions are made by an internal team including privacy, legal, and data management.

Retention

Microsoft believes in and practices information minimization. We strive to gather only the info we need and to store it only for as long as it's needed to provide a service or for analysis. Much of the info about how Windows and apps are functioning is deleted within 30 days. Other info may be retained longer, such as error reporting data or Microsoft Store purchase history.

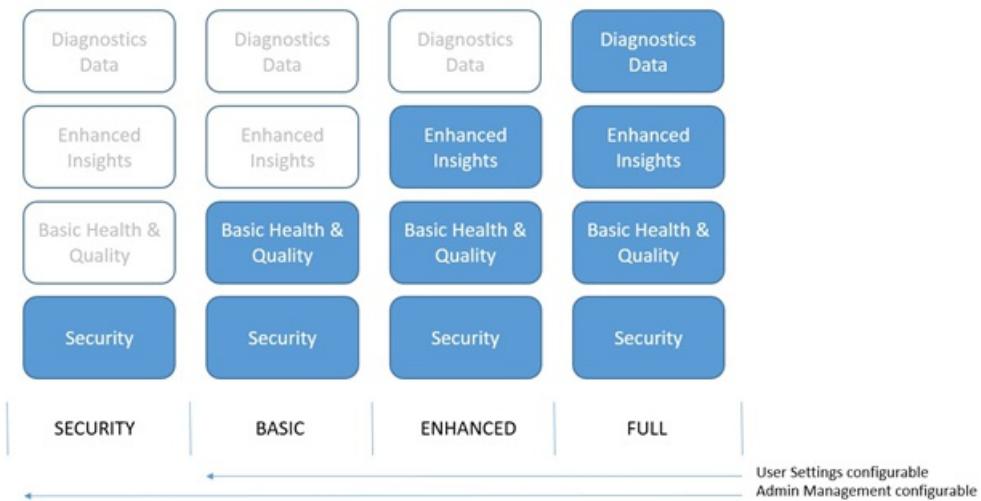
Telemetry levels

This section explains the different telemetry levels in Windows 10, Windows Server 2016, and System Center. These levels are available on all desktop and mobile editions of Windows 10, except for the **Security** level, which is limited to Windows 10 Enterprise, Windows 10 Education, Windows 10 Mobile Enterprise, Windows 10 IoT Core (IoT Core), and Windows Server 2016.

The telemetry data is categorized into four levels:

- **Security.** Information that's required to help keep Windows, Windows Server, and System Center secure, including data about the Connected User Experience and Telemetry component settings, the Malicious Software Removal Tool, and Windows Defender.
- **Basic.** Basic device info, including: quality-related data, app compatibility, app usage data, and data from the **Security** level.
- **Enhanced.** Additional insights, including: how Windows, Windows Server, System Center, and apps are used, how they perform, advanced reliability data, and data from both the **Basic** and the **Security** levels.
- **Full.** All data necessary to identify and help to fix problems, plus data from the **Security**, **Basic**, and **Enhanced** levels.

The levels are cumulative and are illustrated in the following diagram. Also, these levels apply to all editions of Windows Server 2016.



Security level

The Security level gathers only the telemetry info that is required to keep Windows devices, Windows Server, and guests protected with the latest security updates. This level is only available on Windows Server 2016, Windows 10 Enterprise, Windows 10 Education, Windows 10 Mobile Enterprise, and Windows IoT Core editions.

NOTE

If your organization relies on Windows Update for updates, you shouldn't use the **Security** level. Because no Windows Update information is gathered at this level, important information about update failures is not sent. Microsoft uses this information to fix the causes of those failures and improve the quality of our updates.

Windows Server Update Services (WSUS) and System Center Configuration Manager functionality is not affected at this level, nor is telemetry data about Windows Server features or System Center gathered.

The data gathered at this level includes:

- **Connected User Experience and Telemetry component settings.** If general telemetry data has been gathered and is queued, it is sent to Microsoft. Along with this telemetry, the Connected User Experience and Telemetry component may download a configuration settings file from Microsoft's servers. This file is used to configure the Connected User Experience and Telemetry component itself. The data gathered by the client for this request includes OS information, device id (used to identify what specific device is requesting settings) and device class (for example, whether the device is server or desktop).
- **Malicious Software Removal Tool (MSRT)** The MSRT infection report contains information, including device info and IP address.

NOTE

You can turn off the MSRT infection report. No MSRT information is included if MSRT is not used. If Windows Update is turned off, MSRT will not be offered to users. For more info, see Microsoft KB article [891716](#).

- **Windows Defender/Endpoint Protection.** Windows Defender and System Center Endpoint Protection requires some information to function, including: anti-malware signatures, diagnostic information, User Account Control settings, Unified Extensible Firmware Interface (UEFI) settings, and IP address.

NOTE

This reporting can be turned off and no information is included if a customer is using third-party antimalware software, or if Windows Defender is turned off. For more info, see [Windows Defender](#).

Microsoft recommends that Windows Update, Windows Defender, and MSRT remain enabled unless the enterprise uses alternative solutions such as Windows Server Update Services, System Center Configuration Manager, or a third-party antimalware solution. Windows Update, Windows Defender, and MSRT provide core Windows functionality such as driver and OS updates, including security updates.

For servers with default telemetry settings and no Internet connectivity, you should set the telemetry level to **Security**. This stops data gathering for events that would not be uploaded due to the lack of Internet connectivity.

No user content, such as user files or communications, is gathered at the **Security** telemetry level, and we take steps to avoid gathering any information that directly identifies a company or user, such as name, email address, or account ID. However, in rare circumstances, MSRT information may unintentionally contain personal information. For instance, some malware may create entries in a computer's registry that include information such as a username, causing it to be gathered. MSRT reporting is optional and can be turned off at any time.

Basic level

The Basic level gathers a limited set of data that's critical for understanding the device and its configuration. This level also includes the **Security** level data. This level helps to identify problems that can occur on a specific hardware or software configuration. For example, it can help determine if crashes are more frequent on devices with a specific amount of memory or that are running a specific driver version. The Connected User Experience and Telemetry component does not gather telemetry data about System Center, but it can transmit telemetry for other non-Windows applications if they have user consent.

The normal upload range for the Basic telemetry level is between 109 KB - 159 KB per day, per device.

The data gathered at this level includes:

- **Basic device data.** Helps provide an understanding about the types of Windows devices and the configurations and types of native and virtualized Windows Server 2016 in the ecosystem. Examples include:
 - Device attributes, such as camera resolution and display type
 - Internet Explorer version
 - Battery attributes, such as capacity and type
 - Networking attributes, such as number of network adapters, speed of network adapters, mobile operator network, and IMEI number
 - Processor and memory attributes, such as number of cores, architecture, speed, memory size, and firmware
 - Virtualization attribute, such as Second Level Address Translation (SLAT) support and guest operating system
 - Operating system attributes, such as Windows edition and virtualization state
 - Storage attributes, such as number of drives, type, and size
- **Connected User Experience and Telemetry component quality metrics.** Helps provide an understanding about how the Connected User Experience and Telemetry component is functioning, including % of uploaded events, dropped events, and the last upload time.
- **Quality-related information.** Helps Microsoft develop a basic understanding of how a device and its operating system are performing. Some examples are the device characteristics of a Connected Standby device, the number of crashes or hangs, and application state change details, such as how much processor time and memory were used, and the total uptime for an app.
- **Compatibility data.** Helps provide an understanding about which apps are installed on a device or virtual

machine and identifies potential compatibility problems.

- **General app data and app data for Internet Explorer add-ons.** Includes a list of apps that are installed on a native or virtualized instance of the OS and whether these apps function correctly after an upgrade. This app data includes the app name, publisher, version, and basic details about which files have been blocked from usage.
 - **App usage data.** Includes how an app is used, including how long an app is used, when the app has focus, and when the app is started
 - **Internet Explorer add-ons.** Includes a list of Internet Explorer add-ons that are installed on a device and whether these apps will work after an upgrade.
 - **System data.** Helps provide an understanding about whether a device meets the minimum requirements to upgrade to the next version of the operating system. System information includes the amount of memory, as well as information about the processor and BIOS.
 - **Accessory device data.** Includes a list of accessory devices, such as printers or external storage devices, that are connected to Windows PCs and whether these devices will function after upgrading to a new version of the operating system.
 - **Driver data.** Includes specific driver usage that's meant to help figure out whether apps and devices will function after upgrading to a new version of the operating system. This can help to determine blocking issues and then help Microsoft and our partners apply fixes and improvements.
- **Microsoft Store.** Provides information about how the Microsoft Store performs, including app downloads, installations, and updates. It also includes Microsoft Store launches, page views, suspend and resumes, and obtaining licenses.

Enhanced level

The Enhanced level gathers data about how Windows and apps are used and how they perform. This level also includes data from both the **Basic** and **Security** levels. This level helps to improve the user experience with the operating system and apps. Data from this level can be abstracted into patterns and trends that can help Microsoft determine future improvements.

This is the default level for Windows 10 Enterprise and Windows 10 Education editions, and the minimum level needed to quickly identify and address Windows, Windows Server, and System Center quality issues.

The normal upload range for the Enhanced telemetry level is between 239 KB - 348 KB per day, per device.

The data gathered at this level includes:

- **Operating system events.** Helps to gain insights into different areas of the operating system, including networking, Hyper-V, Cortana, storage, file system, and other components.
- **Operating system app events.** A set of events resulting from Microsoft applications and management tools that were downloaded from the Store or pre-installed with Windows or Windows Server, including Server Manager, Photos, Mail, and Microsoft Edge.
- **Device-specific events.** Contains data about events that are specific to certain devices, such as Surface Hub and Microsoft HoloLens. For example, Microsoft HoloLens sends Holographic Processing Unit (HPU)-related events.
- **Some crash dump types.** All crash dump types, except for heap dumps and full dumps.

If the Connected User Experience and Telemetry component detects a problem on Windows 10 that requires gathering more detailed instrumentation, the Connected User Experience and Telemetry component at the **Enhanced** telemetry level will only gather data about the events associated with the specific issue.

Limit Enhanced diagnostic data to the minimum required by Windows Analytics

Windows Analytics Device Health reports are powered by diagnostic data not included in the **Basic** level, such as crash reports and certain operating system events. In the past, organizations sending **Enhanced** or **Full** level diagnostic data were able to participate in Device Health. However, organizations that required detailed event and field level documentation were unable to move from **Basic** to **Enhanced**.

In Windows 10, version 1709, we introduce the **Limit Enhanced diagnostic data to the minimum required by Windows Analytics** feature. When enabled, this feature lets you send only the following subset of **Enhanced** level diagnostic data. For more info about Device Health, see the [Monitor the health of devices with Device Health](#) topic.

- **Operating system events.** Limited to a small set required for analytics reports and documented in the [Windows 10, version 1709 enhanced telemetry events and fields used by Windows Analytics](#) topic.
- **Some crash dump types.** All crash dump types, except for heap and full dumps.

To turn on this behavior for devices

1. Set the diagnostic data level to **Enhanced**, using either Group Policy or MDM.

a. Using Group Policy, set the **Computer Configuration/Administrative Templates/Windows Components/Data Collection and Preview Builds/Allow telemetry** setting to **2**.

-OR-

b. Using MDM, use the Policy CSP to set the **System/AllowTelemetry** value to **2**.

-AND-

2. Enable the **LimitEnhancedDiagnosticDataWindowsAnalytics** setting, using either Group Policy or MDM.

a. Using Group Policy, set the **Computer Configuration/Administrative Templates/Windows Components/Data collection and Preview builds/Limit Enhanced diagnostic data to the minimum required by Windows Analytics** setting to **Enabled**.

-OR-

b. Using MDM, use the Policy CSP to set the **System/LimitEnhancedDiagnosticDataWindowsAnalytics** value to **1**.

Full level

The **Full** level gathers data necessary to identify and to help fix problems, following the approval process described below. This level also includes data from the **Basic**, **Enhanced**, and **Security** levels.

Additionally, at this level, devices opted in to the [Windows Insider Program](#) will send events, such as reliability and app responsiveness, that can show Microsoft how pre-release binaries and features are performing. These events help us make decisions on which builds are flighted. All devices in the [Windows Insider Program](#) are automatically set to this level.

If a device experiences problems that are difficult to identify or repeat using Microsoft's internal testing, additional data becomes necessary. This data can include any user content that might have triggered the problem and is gathered from a small sample of devices that have both opted into the **Full** telemetry level and have exhibited the problem.

However, before more data is gathered, Microsoft's privacy governance team, including privacy and other subject matter experts, must approve the diagnostics request made by a Microsoft engineer. If the request is approved, Microsoft engineers can use the following capabilities to get the information:

- Ability to run a limited, pre-approved list of Microsoft certified diagnostic tools, such as msinfo32.exe,

powercfg.exe, and dxdiag.exe.

- Ability to get registry keys.
- All crash dump types, including heap dumps and full dumps.

Enterprise management

Sharing telemetry data with Microsoft provides many benefits to enterprises, so we do not recommend turning it off. For most enterprise customers, simply adjusting the telemetry level and managing specific components is the best option.

Customers can set the telemetry level in both the user interface and with existing management tools. Users can change the telemetry level in the **Diagnostic data** setting. In the **Settings** app, it is in **Privacy\Feedback & diagnostics**. They can choose between Basic, Enhanced, and Full. The Security level is not available.

IT pros can use various methods, including Group Policy and Mobile Device Management (MDM), to choose a telemetry level. If you're using Windows 10 Enterprise, Windows 10 Education, or Windows Server 2016, the Security telemetry level is available when managing the policy. Setting the telemetry level through policy overrides users' choices. The remainder of this section describes how to do that.

Manage your telemetry settings

We do not recommend that you turn off telemetry in your organization as valuable functionality may be impacted, but we recognize that in some scenarios this may be required. Use the steps in this section to do so for Windows, Windows Server, and System Center.

IMPORTANT

These telemetry levels only apply to Windows, Windows Server, and System Center components and apps that use the Connected User Experience and Telemetry component. Non-Windows components, such as Microsoft Office or other 3rd-party apps, may communicate with their cloud services outside of these telemetry levels. You should work with your app vendors to understand their telemetry policy, and how you can to opt in or opt out. For more information on how Microsoft Office uses telemetry, see [Overview of Office Telemetry](#).

You can turn on or turn off System Center telemetry gathering. The default is on and the data gathered at this level represents what is gathered by default when System Center telemetry is turned on. However, setting the operating system telemetry level to **Basic** will turn off System Center telemetry, even if the System Center telemetry switch is turned on.

The lowest telemetry setting level supported through management policies is **Security**. The lowest telemetry setting supported through the Settings UI is **Basic**. The default telemetry setting for Windows Server 2016 is **Enhanced**.

Configure the operating system telemetry level

You can configure your operating system telemetry settings using the management tools you're already using, such as Group Policy, MDM, or Windows Provisioning. You can also manually change your settings using Registry Editor. Setting your telemetry levels through a management policy overrides any device level settings.

Use the appropriate value in the table below when you configure the management policy.

LEVEL	DATA GATHERED	VALUE
Security	Security data only.	0

LEVEL	DATA GATHERED	VALUE
Basic	Security data, and basic system and quality data.	1
Enhanced	Security data, basic system and quality data, and enhanced insights and advanced reliability data.	2
Full	Security data, basic system and quality data, enhanced insights and advanced reliability data, and full diagnostics data.	3

Use Group Policy to set the telemetry level

Use a Group Policy object to set your organization's telemetry level.

1. From the Group Policy Management Console, go to **Computer Configuration > Administrative Templates > Windows Components > Data Collection and Preview Builds**.
2. Double-click **Allow Telemetry**.
3. In the **Options** box, select the level that you want to configure, and then click **OK**.

Use MDM to set the telemetry level

Use the [Policy Configuration Service Provider \(CSP\)](#) to apply the System/AllowTelemetry MDM policy.

Use Registry Editor to set the telemetry level

Use Registry Editor to manually set the registry level on each device in your organization or you can write a script to edit the registry. If a management policy already exists, such as Group Policy or MDM, it will override this registry setting.

1. Open Registry Editor, and go to **HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Policies\DataCollection**.
2. Right-click **DataCollection**, click New, and then click **DWORD (32-bit) Value**.
3. Type **AllowTelemetry**, and then press ENTER.
4. Double-click **AllowTelemetry**, set the desired value from the table above, and then click **OK**.
5. Click **File > Export**, and then save the file as a .reg file, such as **C:\AllowTelemetry.reg**. You can run this file from a script on each device in your organization.

Configure System Center 2016 telemetry

For System Center 2016 Technical Preview, you can turn off System Center telemetry by following these steps:

- Turn off telemetry by using the System Center UI Console settings workspace.
- For information about turning off telemetry for Service Management Automation and Service Provider Foundation, see [How to disable telemetry for Service Management Automation and Service Provider Foundation](#).

Additional telemetry controls

There are a few more settings that you can turn off that may send telemetry information:

- To turn off Windows Update telemetry, you have two choices. Either turn off Windows Update, or set your devices to be managed by an on premises update server, such as [Windows Server Update Services \(WSUS\)](#) or [System Center Configuration Manager](#).

- Turn off **Windows Defender Cloud-based Protection** and **Automatic sample submission** in **Settings > Update & security > Windows Defender**.
- Manage the Malicious Software Removal Tool in your organization. For more info, see Microsoft KB article [891716](#).
- Turn off **Linguistic Data Collection** in **Settings > Privacy**. At telemetry levels **Enhanced** and **Full**, Microsoft uses Linguistic Data Collection info to improve language model features such as autocomplete, spellcheck, suggestions, input pattern recognition, and dictionary.

NOTE

Microsoft does not intend to gather sensitive information, such as credit card numbers, usernames and passwords, email addresses, or other similarly sensitive information for Linguistic Data Collection. We guard against such events by using technologies to identify and remove sensitive information before linguistic data is sent from the user's device. If we determine that sensitive information has been inadvertently received, we delete the information.

Additional resources

FAQs

- [Cortana, Search, and privacy](#)
- [Windows 10 feedback, diagnostics, and privacy](#)
- [Windows 10 camera and privacy](#)
- [Windows 10 location service and privacy](#)
- [Microsoft Edge and privacy](#)
- [Windows 10 speech, inking, typing, and privacy](#)
- [Windows Hello and privacy](#)
- [Wi-Fi Sense](#)
- [Windows Update Delivery Optimization](#)

Blogs

- [Privacy and Windows 10](#)

Privacy Statement

- [Microsoft Privacy Statement](#)

TechNet

- [Manage connections from Windows operating system components to Microsoft services](#)

Web Pages

- [Privacy at Microsoft](#)

Not finding content you need? Windows 10 users, tell us what you want on [Feedback Hub](#).

Windows 10, version 1709 basic level Windows diagnostic events and fields

10/17/2017 • 165 min to read • [Edit Online](#)

Applies to

- Windows 10, version 1709

The Basic level gathers a limited set of information that is critical for understanding the device and its configuration including: basic device information, quality-related information, app compatibility, and Windows Store. When the level is set to Basic, it also includes the Security level information.

The Basic level helps to identify problems that can occur on a particular device hardware or software configuration. For example, it can help determine if crashes are more frequent on devices with a specific amount of memory or that are running a particular driver version. This helps Microsoft fix operating system or app problems.

Use this article to learn about diagnostic events, grouped by event area, and the fields within each event. A brief description is provided for each field. Every event generated includes common data, which collects device data.

You can learn more about Windows functional and diagnostic data through these articles:

- [Windows 10, version 1703 basic diagnostic events and fields](#)
- [Manage connections from Windows operating system components to Microsoft services](#)
- [Configure Windows telemetry in your organization](#)

Common data extensions

Common Data Extensions.App

The following fields are available:

- **expId** Associates a flight, such as an OS flight, or an experiment, such as a web site UX experiment, with an event.
- **userId** The userID as known by the application.
- **env** The environment from which the event was logged.
- **asId** An integer value that represents the app session. This value starts at 0 on the first app launch and increments after each subsequent app launch per boot session.

Common Data Extensions.CS

The following fields are available:

- **sig** A common schema signature that identifies new and modified event schemas.

Common Data Extensions.CUET

The following fields are available:

- **stId** Represents the Scenario Entry Point ID. This is a unique GUID for each event in a diagnostic scenario. This used to be Scenario Trigger ID.
- **alId** Represents the ETW ActivityId. Logged via TraceLogging or directly via ETW.
- **raId** Represents the ETW Related ActivityId. Logged via TraceLogging or directly via ETW.
- **op** Represents the ETW Op Code.
- **cat** Represents a bitmask of the ETW Keywords associated with the event.

- **flags** Represents the bitmap that captures various Windows specific flags.
- **cplId** The composer ID, such as Reference, Desktop, Phone, Holographic, Hub, IoT Composer.
- **tickets** A list of strings that represent entries in the HTTP header of the web request that includes this event.
- **bseq** Upload buffer sequence number in the format <buffer identifier>:<sequence number>
- **mon** Combined monitor and event sequence numbers in the format <monitor sequence>:<event sequence>

Common Data Extensions.Device

The following fields are available:

- **ver** Represents the major and minor version of the extension.
- **localId** Represents a locally defined unique ID for the device, not the human readable device name. Most likely equal to the value stored at HKLM\Software\Microsoft\SQMClient\MachineId
- **deviceClass** Represents the classification of the device, the device "family". For example, Desktop, Server, or Mobile.

Common Data Extensions.Envelope

The following fields are available:

- **ver** Represents the major and minor version of the extension.
- **name** Represents the uniquely qualified name for the event.
- **time** Represents the event date time in Coordinated Universal Time (UTC) when the event was generated on the client. This should be in ISO 8601 format.
- **popSample** Represents the effective sample rate for this event at the time it was generated by a client.
- **epoch** Represents the epoch and seqNum fields, which help track how many events were fired and how many events were uploaded, and enables identification of data lost during upload and de-duplication of events on the ingress server.
- **seqNum** Represents the sequence field used to track absolute order of uploaded events. It is an incrementing identifier for each event added to the upload queue. The Sequence helps track how many events were fired and how many events were uploaded and enables identification of data lost during upload and de-duplication of events on the ingress server.
- **iKey** Represents an ID for applications or other logical groupings of events.
- **flags** Represents a collection of bits that describe how the event should be processed by the Connected User Experience and Telemetry component pipeline. The lowest-order byte is the event persistence. The next byte is the event latency.
- **os** Represents the operating system name.
- **osVer** Represents the OS version, and its format is OS dependent.
- **appId** Represents a unique identifier of the client application currently loaded in the process producing the event; and is used to group events together and understand usage pattern, errors by application.
- **appVer** Represents the version number of the application. Used to understand errors by Version, Usage by Version across an app.
- **cV** Represents the Correlation Vector: A single field for tracking partial order of related telemetry events across component boundaries.

Common Data Extensions.OS

The following fields are available:

- **ver** Represents the major and minor version of the extension.
- **expId** Represents the experiment ID. The standard for associating a flight, such as an OS flight (pre-release build), or an experiment, such as a web site UX experiment, with an event is to record the flight / experiment IDs in Part A of the common schema.
- **locale** Represents the locale of the operating system.

- **bootId** An integer value that represents the boot session. This value starts at 0 on first boot after OS install and increments after every reboot.

Common Data Extensions.User

The following fields are available:

- **ver** Represents the major and minor version of the extension.
- **localId** Represents a unique user identity that is created locally and added by the client. This is not the user's account ID.

Common Data Extensions.XBL

The following fields are available:

- **nbf** Not before time
- **expId** Expiration time
- **sbx** XBOX sandbox identifier
- **dty** XBOX device type
- **did** XBOX device ID
- **xid** A list of base10-encoded XBOX User IDs.
- **uts** A bit field, with 2 bits being assigned to each user ID listed in xid. This field is omitted if all users are retail accounts.

Common Data Extensions.Consent UI Event

This User Account Control (UAC) telemetry point collects information on elevations that originate from low integrity levels. This occurs when a process running at low integrity level (IL) requires higher (administrator) privileges, and therefore requests for elevation via UAC (consent.exe). By better understanding the processes requesting these elevations, Microsoft can in turn improve the detection and handling of potentially malicious behavior in this path.

The following fields are available:

- **eventType** Represents the type of elevation: If it succeeded, was cancelled, or was auto-approved.
- **splitToken** Represents the flag used to distinguish between administrators and standard users.
- **friendlyName** Represents the name of the file requesting elevation from low IL.
- **elevationReason** Represents the distinction between various elevation requests sources (appcompat, installer, COM, MSI and so on).
- **exeName** Represents the name of the file requesting elevation from low IL.
- **signatureState** Represents the state of the signature, if it signed, unsigned, OS signed and so on.
- **publisherName** Represents the name of the publisher of the file requesting elevation from low IL.
- **cmdLine** Represents the full command line arguments being used to elevate.
- **Hash.Length** Represents the length of the hash of the file requesting elevation from low IL.
- **Hash** Represents the hash of the file requesting elevation from low IL.
- **HashAlgId** Represents the algorithm ID of the hash of the file requesting elevation from low IL.
- **telemetryFlags** Represents the details about the elevation prompt for CEIP data.
- **timeStamp** Represents the time stamp on the file requesting elevation.
- **fileVersionMS** Represents the major version of the file requesting elevation.
- **fileVersionLS** Represents the minor version of the file requesting elevation.

Common data fields

Common Data Fields.MS.Device.DeviceInventory.Change

These fields are added whenever Ms.Device.DeviceInventoryChange is included in the event.

The following fields are available:

- **syncId** A string used to group StartSync, EndSync, Add, and Remove operations that belong together. This field is unique by Sync period and is used to disambiguate in situations where multiple agents perform overlapping inventories for the same object.
- **objectType** Indicates the object type that the event applies to.
- **Action** The change that was invoked on a device inventory object.
- **inventoryId** Device ID used for Compatibility testing

Common Data Fields.TelClientSynthetic.PrivacySettingsAfterCreatorsUpdate.PreUpgradeSettings

These fields are added whenever PreUpgradeSettings is included in the event.

The following fields are available:

- **HKLM_SensorPermissionState.SensorPermissionState** The state of the Location service before the feature update completed.
- **HKLM_SensorPermissionState.HRESULT** The error code returned when trying to query the Location service for the device.
- **HKCU_SensorPermissionState.SensorPermissionState** The state of the Location service when a user signs on before the feature update completed.
- **HKCU_SensorPermissionState.HRESULT** The error code returned when trying to query the Location service for the current user.
- **HKLM_LocationPlatform.Status** The state of the location platform after the feature update has completed.
- **HKLM_LocationPlatform.HRESULT** The error code returned when trying to query the location platform for the device.
- **HKLM_LocationSyncEnabled.AcceptedPrivacyPolicy** The speech recognition state for the device before the feature update completed.
- **HKLM_LocationSyncEnabled.HRESULT** The error code returned when trying to query the Find My Device service for the device.
- **HKCU_LocationSyncEnabled.AcceptedPrivacyPolicy** The speech recognition state for the current user before the feature update completed.
- **HKCU_LocationSyncEnabled.HRESULT** The error code returned when trying to query the Find My Device service for the current user.
- **HKLM_AllowTelemetry.AllowTelemetry** The state of the Connected User Experiences and Telemetry component for the device before the feature update.
- **HKLM_AllowTelemetry.HRESULT** The error code returned when trying to query the Connected User Experiences and Telemetry component for the device.
- **HKLM_TIPC.Enabled** The state of TIPC for the device.
- **HKLM_TIPC.HRESULT** The error code returned when trying to query TIPC for the device.
- **HKCU_TIPC.Enabled** The state of TIPC for the current user.
- **HKCU_TIPC.HRESULT** The error code returned when trying to query TIPC for the current user.
- **HKLM_FlipAhead.FPEnabled** Is Flip Ahead enabled for the device before the feature update was completed?
- **HKLM_FlipAhead.HRESULT** The error code returned when trying to query Flip Ahead for the device.
- **HKCU_FlipAhead.FPEnabled** Is Flip Ahead enabled for the current user before the feature update was completed?
- **HKCU_FlipAhead.HRESULT** The error code returned when trying to query Flip Ahead for the current user.
- **HKLM_TailoredExperiences.TailoredExperiencesWithDiagnosticDataEnabled** Is Tailored Experiences with Diagnostics Data enabled for the current user after the feature update had completed?
- **HKCU_TailoredExperiences.HRESULT** The error code returned when trying to query Tailored Experiences with Diagnostics Data for the current user.

- **HKLM_AdvertisingID.Enabled** Is the advertising ID enabled for the device?
- **HKLM_AdvertisingID.HRESULT** The error code returned when trying to query the state of the advertising ID for the device.
- **HKCU_AdvertisingID.Enabled** Is the advertising ID enabled for the current user?
- **HKCU_AdvertisingID.HRESULT** The error code returned when trying to query the state of the advertising ID for the user.

Common Data Fields.TelClientSynthetic.PrivacySettingsAfterCreatorsUpdate.PostUpgradeSettings

These fields are added whenever PostUpgradeSettings is included in the event.

The following fields are available:

- **HKLM_SensorPermissionState.SensorPermissionState** The state of the Location service after the feature update has completed.
- **HKLM_SensorPermissionState.HRESULT** The error code returned when trying to query the Location service for the device.
- **HKCU_SensorPermissionState.SensorPermissionState** The state of the Location service when a user signs on after a feature update has completed.
- **HKCU_SensorPermissionState.HRESULT** The error code returned when trying to query the Location service for the current user.
- **HKLM_LocationPlatform.Status** The state of the location platform after the feature update has completed.
- **HKLM_LocationPlatform.HRESULT** The error code returned when trying to query the location platform for the device.
- **HKLM_LocationSyncEnabled.AcceptedPrivacyPolicy** The speech recognition state for the device after the feature update has completed.
- **HKLM_LocationSyncEnabled.HRESULT** The error code returned when trying to query the Find My Device service for the device.
- **HKCU_LocationSyncEnabled.AcceptedPrivacyPolicy** The speech recognition state for the current user after the feature update has completed.
- **HKCU_LocationSyncEnabled.HRESULT** The error code returned when trying to query the Find My Device service for the current user.
- **HKLM_AllowTelemetry.AllowTelemetry** The state of the Connected User Experiences and Telemetry component for the device after the feature update.
- **HKLM_AllowTelemetry.HRESULT** The error code returned when trying to query the Connected User Experiences and Telemetry component for the device.
- **HKLM_TIPC.Enabled** The state of TIPC for the device.
- **HKLM_TIPC.HRESULT** The error code returned when trying to query TIPC for the device.
- **HKCU_TIPC.Enabled** The state of TIPC for the current user.
- **HKCU_TIPC.HRESULT** The error code returned when trying to query TIPC for the current user.
- **HKLM_FlipAhead.FPEnabled** Is Flip Ahead enabled for the device after the feature update has completed?
- **HKLM_FlipAhead.HRESULT** The error code returned when trying to query Flip Ahead for the device.
- **HKCU_FlipAhead.FPEnabled** Is Flip Ahead enabled for the current user after the feature update has completed?
- **HKCU_FlipAhead.HRESULT** The error code returned when trying to query Flip Ahead for the current user.
- **HKLM_TailoredExperiences.TailoredExperiencesWithDiagnosticDataEnabled** Is Tailored Experiences with Diagnostics Data enabled for the current user after the feature update had completed?
- **HKCU_TailoredExperiences.HRESULT** The error code returned when trying to query Tailored Experiences with Diagnostics Data for the current user.
- **HKLM_AdvertisingID.Enabled** Is the advertising ID enabled for the device?
- **HKLM_AdvertisingID.HRESULT** The error code returned when trying to query the state of the advertising ID

for the device.

- **HKCU_AdvertisingID.Enabled** Is the advertising ID enabled for the current user?
- **HKCU_AdvertisingID.HRESULT** The error code returned when trying to query the state of the advertising ID for the user.

Appraiser events

Microsoft.Windows.Appraiser.General.RunContext

"This event indicates what should be expected in the data payload."

The following fields are available:

- **AppraiserBranch** The source branch in which the currently running version of Appraiser was built.
- **AppraiserProcess** The name of the process that launched Appraiser.
- **AppraiserVersion** The version of the Appraiser file generating the events.
- **Context** Indicates what mode Appraiser is running in. Example: Setup or Telemetry.
- **PCFP** An ID for the system calculated by hashing hardware identifiers.
- **Time** The client time of the event.

Microsoft.Windows.Appraiser.General.TelemetryRunHealth

A summary event indicating the parameters and result of a telemetry run. This allows the rest of the data sent over the course of the run to be properly contextualized and understood, which is then used to keep Windows up-to-date.

The following fields are available:

- **AppraiserBranch** The source branch in which the version of Appraiser that is running was built.
- **AppraiserDataVersion** The version of the data files being used by the Appraiser telemetry run.
- **AppraiserProcess** The name of the process that launched Appraiser.
- **AppraiserVersion** The file version (major, minor and build) of the Appraiser DLL, concatenated without dots.
- **AuxFinal** Obsolete, always set to false
- **AuxInitial** Obsolete, indicates if Appraiser is writing data files to be read by the Get Windows 10 app.
- **DeadlineDate** A timestamp representing the deadline date, which is the time until which appraiser will wait to do a full scan.
- **EnterpriseRun** Indicates if the telemetry run is an enterprise run, which means appraiser was run from the command line with an extra enterprise parameter.
- **FullSync** Indicates if Appraiser is performing a full sync, which means that full set of events representing the state of the machine are sent. Otherwise, only the changes from the previous run are sent.
- **InventoryFullSync** Indicates if inventory is performing a full sync, which means that the full set of events representing the inventory of machine are sent.
- **PCFP** An ID for the system calculated by hashing hardware identifiers.
- **PerfBackoff** Indicates if the run was invoked with logic to stop running when a user is present. Helps to understand why a run may have a longer elapsed time than normal.
- **PerfBackoffInsurance** Indicates if appraiser is running without performance backoff because it has run with perf backoff and failed to complete several times in a row.
- **RunAppraiser** Indicates if Appraiser was set to run at all. If this is false, it is understood that data events will not be received from this device.
- **RunDate** The date that the telemetry run was started, expressed as a filetime.
- **RunGeneralTel** Indicates if the generaltel.dll component was run. Generaltel collects additional telemetry on an infrequent schedule and only from machines at telemetry levels higher than Basic.
- **RunOnline** Indicates if appraiser was able to connect to Windows Update and therefore is making decisions

using up-to-date driver coverage information.

- **RunResult** The hresult of the Appraiser telemetry run.
- **SendingUtc** Indicates if the Appraiser client is sending events during the current telemetry run.
- **StoreHandleIsNotNull** Obsolete, always set to false
- **TelemetrySent** Indicates if telemetry was successfully sent.
- **ThrottlingUtc** Indicates if the Appraiser client is throttling its output of CUET events to avoid being disabled. This increases runtime but also telemetry reliability.
- **Time** The client time of the event.
- **VerboseMode** Indicates if appraiser ran in Verbose mode, which is a test-only mode with extra logging.
- **WhyFullSyncWithoutTablePrefix** Indicates the reason or reasons that a full sync was generated.

Microsoft.Windows.Appraiser.General.EnterpriseScenarioWithDiagTrackServiceRunning

The event that indicates that Appraiser has been triggered to run an enterprise scenario while the DiagTrack service is installed. This event can only be sent if a special flag is used to trigger the enterprise scenario.

The following fields are available:

- **PCFP** An ID for the system calculated by hashing hardware identifiers.
- **Time** The client time of the event.

Microsoft.Windows.Appraiser.General.InventoryApplicationFileAdd

This event represents the basic metadata about a file on the system. The file must be part of an app and either have a block in the compatibility database or are part of an anti-virus program.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **BinaryType** A binary type. Example: UNINITIALIZED, ZERO_BYTE, DATA_ONLY, DOS_MODULE, NE16_MODULE, PE32_UNKNOWN, PE32_I386, PE32_ARM, PE64_UNKNOWN, PE64_AMD64, PE64_ARM64, PE64_IA64, PE32_CLR_32, PE32_CLR_IL, PE32_CLR_IL_PREFER32, PE64_CLR_64
- **BinFileVersion** An attempt to clean up FileVersion at the client that tries to place the version into 4 octets.
- **BinProductVersion** An attempt to clean up ProductVersion at the client that tries to place the version into 4 octets.
- **BoeProgramId** If there is no entry in Add/Remove Programs, this is the ProgramID that is generated from the file metadata.
- **CompanyName** The company name of the vendor who developed this file.
- **FileId** A hash that uniquely identifies a file.
- **FileVersion** The File version field from the file metadata under Properties -> Details.
- **LinkDate** The date and time that this file was linked on.
- **LowerCaseLongPath** The full file path to the file that was inventoried on the device.
- **Name** The name of the file that was inventoried.
- **ProductName** The Product name field from the file metadata under Properties -> Details.
- **ProductVersion** The Product version field from the file metadata under Properties -> Details.
- **ProgramId** A hash of the Name, Version, Publisher, and Language of an application used to identify it.
- **Size** The size of the file (in hexadecimal bytes).

Microsoft.Windows.Appraiser.General.DecisionApplicationFileAdd

This event sends compatibility decision data about a file to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.

- **BlockAlreadyInbox** The uplevel runtime block on the file already existed on the current OS.
- **BlockingApplication** Are there any application issues that interfere with upgrade due to the file in question?
- **DisplayGenericMessage** Will be a generic message be shown for this file?
- **HardBlock** This file is blocked in the SDB.
- **HasUxBlockOverride** Does the file have a block that is overridden by a tag in the SDB?
- **MigApplication** Does the file have a MigXML from the SDB associated with it that applies to the current upgrade mode?
- **MigRemoval** Does the file have a MigXML from the SDB that will cause the app to be removed on upgrade?
- **NeedsDismissAction** Will the file cause an action that can be dismissed?
- **NeedsInstallPostUpgradeData** After upgrade, the file will have a post-upgrade notification to install a replacement for the app.
- **NeedsNotifyPostUpgradeData** Does the file have a notification that should be shown after upgrade?
- **NeedsReinstallPostUpgradeData** After upgrade, this file will have a post-upgrade notification to reinstall the app.
- **NeedsUninstallAction** The file must be uninstalled to complete the upgrade.
- **SdbBlockUpgrade** The file is tagged as blocking upgrade in the SDB,
- **SdbBlockUpgradeCanReinstall** The file is tagged as blocking upgrade in the SDB. It can be reinstalled after upgrade.
- **SdbBlockUpgradeUntilUpdate** The file is tagged as blocking upgrade in the SDB. If the app is updated, the upgrade can proceed.
- **SdbReinstallUpgrade** The file is tagged as needing to be reinstalled after upgrade in the SDB. It does not block upgrade.
- **SdbReinstallUpgradeWarn** The file is tagged as needing to be reinstalled after upgrade with a warning in the SDB. It does not block upgrade.
- **SoftBlock** The file is softblocked in the SDB and has a warning.

Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoBlockAdd

This event sends blocking data about any compatibility blocking entries hit on the system that are not directly related to specific applications or devices, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.

Microsoft.Windows.Appraiser.General.DecisionMatchingInfoBlockAdd

This event sends compatibility decision data about blocking entries on the system that are not keyed by either applications or devices, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.
- **BlockingApplication** Are there any application issues that interfere with upgrade due to matching info blocks?
- **DisplayGenericMessage** Will a generic message be shown for this block?
- **NeedsUninstallAction** Does the user need to take an action in setup due to a matching info block?
- **SdbBlockUpgrade** Is a matching info block blocking upgrade?
- **SdbBlockUpgradeCanReinstall** Is a matching info block blocking upgrade, but has the can reinstall tag?
- **SdbBlockUpgradeUntilUpdate** Is a matching info block blocking upgrade but has the until update tag?

Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPassiveAdd

This event sends compatibility database information about non-blocking compatibility entries on the system that are not keyed by either applications or devices, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.

Microsoft.Windows.Appraiser.General.DecisionMatchingInfoPassiveAdd

This event sends compatibility decision data about non-blocking entries on the system that are not keyed by either applications or devices, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **BlockingApplication** Are there any application issues that interfere with upgrade due to matching info blocks?
- **MigApplication** Is there a matching info block with a mig for the current mode of upgrade?

Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPostUpgradeAdd

This event sends compatibility database information about entries requiring reinstallation after an upgrade on the system that are not keyed by either applications or devices, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.

Microsoft.Windows.Appraiser.General.DecisionMatchingInfoPostUpgradeAdd

This event sends compatibility decision data about entries that require reinstall after upgrade. It's used to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **NeedsInstallPostUpgradeData** Will the file have a notification after upgrade to install a replacement for the app?
- **NeedsNotifyPostUpgradeData** Should a notification be shown for this file after upgrade?
- **NeedsReinstallPostUpgradeData** Will the file have a notification after upgrade to reinstall the app?
- **SdbReinstallUpgrade** The file is tagged as needing to be reinstalled after upgrade in the compatibility database (but is not blocking upgrade).

Microsoft.Windows.Appraiser.General.DatasourceDevicePnpAdd

This event sends compatibility data for a PNP device, to help keep Windows up-to-date.

The following fields are available:

- **ActiveNetworkConnection** Is the device an active network device?
- **AppraiserVersion** The version of the appraiser file generating the events.
- **IsBootCritical** Is the device boot critical?
- **WuDriverCoverage** Is there a driver uplevel for this device according to Windows Update?
- **WuDriverUpdateId** The Windows Update ID of the applicable uplevel driver.
- **WuPopulatedFromId** The expected uplevel driver matching ID based on driver coverage from Windows Update.

Microsoft.Windows.Appraiser.General.DecisionDevicePnpAdd

This event sends compatibility decision data about a PNP device to help keep Windows up-to-date.

The following fields are available:

- **AssociatedDriverWillNotMigrate** Will the driver associated with this plug-and-play device migrate?
- **AppraiserVersion** The version of the appraiser file generating the events.

- **AssociatedDriverIsBlocked** Is the driver associated with this PNP device blocked?
- **BlockAssociatedDriver** Should the driver associated with this PNP device be blocked?
- **BlockingDevice** Is this PNP device blocking upgrade?
- **BlockUpgradelfDriverBlocked** Is the PNP device both boot critical and does not have a driver included with the OS?
- **BlockUpgradelfDriverBlockedAndOnlyActiveNetwork** Is this PNP device the only active network device?
- **DisplayGenericMessage** Will a generic message be shown during Setup for this PNP device?
- **DriverAvailableInbox** Is a driver included with the operating system for this PNP device?
- **DriverAvailableOnline** Is there a driver for this PNP device on Windows Update?
- **DriverAvailableUplevel** Is there a driver on Windows Update or included with the operating system for this PNP device?
- **DriverBlockOverridden** Is there a driver block on the device that has been overridden?
- **NeedsDismissAction** Will the user need to dismiss a warning during Setup for this device?
- **NotRegressed** Does the device have a problem code on the source OS that is no better than the one it would have on the target OS?
- **SdbDeviceBlockUpgrade** Is there an SDB block on the PNP device that blocks upgrade?
- **SdbDriverBlockOverridden** Is there an SDB block on the PNP device that blocks upgrade, but that block was overridden?

Microsoft.Windows.Appraiser.General.DatasourceDriverPackageAdd

This event sends compatibility database data about driver packages to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.

Microsoft.Windows.Appraiser.General.DecisionDriverPackageAdd

This event sends decision data about driver package compatibility to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.
- **DriverBlockOverridden** Does the driver package have an SDB block that blocks it from migrating, but that block has been overridden?
- **DriverIsDeviceBlocked** Was the driver package blocked because of a device block?
- **DriverIsDriverBlocked** Is the driver package blocked because of a driver block?
- **DriverShouldNotMigrate** Should the driver package be migrated during upgrade?
- **SdbDriverBlockOverridden** Does the driver package have an SDB block that blocks it from migrating, but that block has been overridden?

Microsoft.Windows.Appraiser.General.InventorySystemBiosAdd

This event sends basic metadata about the BIOS to determine whether it has a compatibility block.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **BiosDate** The release date of the BIOS in UTC format.
- **BiosName** The name field from Win32_BIOS.
- **Manufacturer** The manufacturer field from Win32_ComputerSystem.
- **Model** The model field from Win32_ComputerSystem.

Microsoft.Windows.Appraiser.General.SystemMemoryAdd

This event sends data on the amount of memory on the system and whether it meets requirements, to help keep

Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **Blocking** Is the device from upgrade due to memory restrictions?
- **MemoryRequirementViolated** Was a memory requirement violated?
- **pageFile** The current committed memory limit for the system or the current process, whichever is smaller (in bytes).
- **ram** The amount of memory on the device.
- **ramKB** The amount of memory (in KB).
- **virtual** The size of the user-mode portion of the virtual address space of the calling process (in bytes).
- **virtualKB** The amount of virtual memory (in KB).

Microsoft.Windows.Appraiser.General.DecisionSystemBiosAdd

This event sends compatibility decision data about the BIOS to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **Blocking** Is the device blocked from upgrade due to a BIOS block?
- **HasBiosBlock** Does the device have a BIOS block?

Microsoft.Windows.Appraiser.General.DatasourceSystemBiosAdd

This event sends compatibility database information about the BIOS to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **SdbEntries** An array of fields indicating the SDB entries that apply to this BIOS.

Microsoft.Windows.Appraiser.General.SystemProcessorCompareExchangeAdd

This event sends data indicating whether the system supports the CompareExchange128 CPU requirement, to help keep Windows up to date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **Blocking** Is the upgrade blocked due to the processor?
- **CompareExchange128Support** Does the CPU support CompareExchange128?

Microsoft.Windows.Appraiser.General.SystemProcessorLahfSahfAdd

This event sends data indicating whether the system supports the LahfSahf CPU requirement, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **Blocking** Is the upgrade blocked due to the processor?
- **LahfSahfSupport** Does the CPU support LAHF/SAHF?

Microsoft.Windows.Appraiser.General.SystemProcessorNxAdd

This event sends data indicating whether the system supports the NX CPU requirement, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **Blocking** Is the upgrade blocked due to the processor?
- **NXDriverResult** The result of the driver used to do a non-deterministic check for NX support.
- **NXProcessorSupport** Does the processor support NX?

Microsoft.Windows.Appraiser.General.SystemProcessorPrefetchWAdd

This event sends data indicating whether the system supports the PrefetchW CPU requirement, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **Blocking** Is the upgrade blocked due to the processor?
- **PrefetchWSupport** Does the processor support PrefetchW?

Microsoft.Windows.Appraiser.General.SystemProcessorSse2Add

This event sends data indicating whether the system supports the SSE2 CPU requirement, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **Blocking** Is the upgrade blocked due to the processor?
- **SSE2ProcessorSupport** Does the processor support SSE2?

Microsoft.Windows.Appraiser.General.SystemWimAdd

This event sends data indicating whether the operating system is running from a compressed WIM file, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **IsWimBoot** Is the current operating system running from a compressed WIM file?
- **RegistryWimBootValue** The raw value from the registry that is used to indicate if the device is running from a WIM.

Microsoft.Windows.Appraiser.General.SystemTouchAdd

This event sends data indicating whether the system supports touch, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **IntegratedTouchDigitizerPresent** Is there an integrated touch digitizer?
- **MaximumTouches** The maximum number of touch points supported by the device hardware.

Microsoft.Windows.Appraiser.General.SystemWindowsActivationStatusAdd

This event sends data indicating whether the current operating system is activated, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **WindowsIsLicensedApiValue** The result from the API that's used to indicate if operating system is activated.
- **WindowsNotActivatedDecision** Is the current operating system activated?

Microsoft.Windows.Appraiser.General.InventoryLanguagePackAdd

This event sends data about the number of language packs installed on the system, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **HasLanguagePack** Does this device have 2 or more language packs?
- **LanguagePackCount** How many language packs are installed?

Microsoft.Windows.Appraiser.General.SystemWlanAdd

This event sends data indicating whether the system has WLAN, and if so, whether it uses an emulated driver that could block an upgrade, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **Blocking** Is the upgrade blocked because of an emulated WLAN driver?
- **HasWlanBlock** Does the emulated WLAN driver have an upgrade block?
- **WlanEmulatedDriver** Does the device have an emulated WLAN driver?
- **WlanExists** Does the device support WLAN at all?
- **WlanModulePresent** Are any WLAN modules present?
- **WlanNativeDriver** Does the device have a non-emulated WLAN driver?

Microsoft.Windows.Appraiser.General.InventoryMediaCenterAdd

This event sends true/false data about decision points used to understand whether Windows Media Center is used on the system, to help keep Windows up to date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **EverLaunched** Has Windows Media Center ever been launched?
- **HasConfiguredTv** Has the user configured a TV tuner through Windows Media Center?
- **HasExtendedUserAccounts** Are any Windows Media Center Extender user accounts configured?
- **HasWatchedFolders** Are any folders configured for Windows Media Center to watch?
- **IsDefaultLauncher** Is Windows Media Center the default app for opening music or video files?
- **IsPaid** Is the user running a Windows Media Center edition that implies they paid for Windows Media Center?
- **IsSupported** Does the running OS support Windows Media Center?

Microsoft.Windows.Appraiser.General.DecisionMediaCenterAdd

This event sends decision data about the presence of Windows Media Center, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **BlockingApplication** Is there any application issues that interfere with upgrade due to Windows Media Center?
- **MediaCenterActivelyUsed** If Windows Media Center is supported on the edition, has it been run at least once and are the MediaCenterIndicators are true?
- **MediaCenterIndicators** Do any indicators imply that Windows Media Center is in active use?
- **MediaCenterInUse** Is Windows Media Center actively being used?
- **MediaCenterPaidOrActivelyUsed** Is Windows Media Center actively being used or is it running on a supported edition?
- **NeedsDismissAction** Are there any actions that can be dismissed coming from Windows Media Center?

Microsoft.Windows.Appraiser.General.ChecksumTotalPictureCount

This event lists the types of objects and how many of each exist on the client device. This allows for a quick way to ensure that the records present on the server match what is present on the client.

The following fields are available:

- **DatasourceApplicationFile_RS2** The total DatasourceApplicationFile objects targeting Windows 10 version 1703 present on this device.
- **DatasourceDevicePnp_RS2** The total DatasourceDevicePnp objects targeting Windows 10 version 1703 present on this device.
- **DatasourceDriverPackage_RS2** The total DatasourceDriverPackage objects targeting Windows 10 version 1703 present on this device.
- **DataSourceMatchingInfoBlock_RS2** The total DataSourceMatchingInfoBlock objects targeting Windows 10 version 1703 present on this device.
- **DataSourceMatchingInfoPassive_RS2** The total DataSourceMatchingInfoPassive objects targeting Windows 10 version 1703 present on this device.
- **DataSourceMatchingInfoPostUpgrade_RS2** The total DataSourceMatchingInfoPostUpgrade objects targeting Windows 10 version 1703 present on this device.
- **DatasourceSystemBios_RS2** The total DatasourceSystemBios objects targeting Windows 10 version 1703 present on this device.
- **DecisionApplicationFile_RS2** The total DecisionApplicationFile objects targeting Windows 10 version 1703 present on this device.
- **DecisionDevicePnp_RS2** The total DecisionDevicePnp objects targeting Windows 10 version 1703 present on this device.
- **DecisionDriverPackage_RS2** The total DecisionDriverPackage objects targeting Windows 10 version 1703 present on this device.
- **DecisionMatchingInfoBlock_RS2** The total DecisionMatchingInfoBlock objects targeting Windows 10 version 1703 present on this device.
- **DecisionMatchingInfoPassive_RS2** The total DecisionMatchingInfoPassive objects targeting Windows 10 version 1703 present on this device.
- **DecisionMatchingInfoPostUpgrade_RS2** The total DecisionMatchingInfoPostUpgrade objects targeting Windows 10 version 1703 present on this device.
- **DecisionMediaCenter_RS2** The total DecisionMediaCenter objects targeting Windows 10 version 1703 present on this device.
- **DecisionSystemBios_RS2** The total DecisionSystemBios objects targeting Windows 10 version 1703 present on this device.
- **InventoryApplicationFile** The total InventoryApplicationFile objects that are present on this device.
- **InventoryLanguagePack** The total InventoryLanguagePack objects that are present on this device.
- **InventoryMediaCenter** The total InventoryMediaCenter objects that are present on this device.
- **InventorySystemBios** The total InventorySystemBios objects that are present on this device.
- **InventoryUplevelDriverPackage** The total InventoryUplevelDriverPackage objects that are present on this device.
- **PCFP** An ID for the system that is calculated by hashing hardware identifiers.
- **SystemMemory** The total SystemMemory objects that are present on this device.
- **SystemProcessorCompareExchange** The total SystemProcessorCompareExchange objects that are present on this device.
- **SystemProcessorLahfSahf** The total SystemProcessorLahfSahf objects that are present on this device.
- **SystemProcessorNx** The total SystemProcessorNx objects that are present on this device.
- **SystemProcessorPrefetchW** The total SystemProcessorPrefetchW objects that are present on this device.
- **SystemProcessorSse2** The total SystemProcessorSse2 objects that are present on this device.

- **SystemTouch** The total SystemTouch objects that are present on this device.
- **SystemWim** The total SystemWim objects that are present on this device
- **SystemWindowsActivationStatus** The total SystemWindowsActivationStatus objects that are present on this device.
- **SystemWlan** The total SystemWlan objects that are present on this device.
- **Wmdrm_RS2** The total Wmdrm objects targeting Windows 10 version 1703 present on this device.
- **DatasourceApplicationFile_RS3** "The total DecisionApplicationFile objects targeting the next release of Windows on this device."
- **DatasourceDevicePnp_RS3** The total DatasourceDevicePnp objects targeting the next release of Windows on this device.
- **DatasourceDriverPackage_RS3** The total DatasourceDriverPackage objects targeting the next release of Windows on this device.
- **DataSourceMatchingInfoBlock_RS3** The total DataSourceMatchingInfoBlock objects targeting the next release of Windows on this device.
- **DataSourceMatchingInfoPassive_RS3** The total DataSourceMatchingInfoPassive objects targeting the next release of Windows on this device.
- **DataSourceMatchingInfoPostUpgrade_RS3** The total DataSourceMatchingInfoPostUpgrade objects targeting the next release of Windows on this device.
- **DatasourceSystemBios_RS3** The total DatasourceSystemBios objects targeting the next release of Windows on this device.
- **DecisionApplicationFile_RS3** The total DecisionApplicationFile objects targeting the next release of Windows on this device.
- **DecisionDevicePnp_RS3** The total DecisionDevicePnp objects targeting the next release of Windows on this device.
- **DecisionDriverPackage_RS3** The total DecisionDriverPackage objects targeting the next release of Windows on this device.
- **DecisionMatchingInfoBlock_RS3** The total DecisionMatchingInfoBlock objects targeting the next release of Windows on this device.
- **DecisionMatchingInfoPassive_RS3** The total DataSourceMatchingInfoPassive objects targeting the next release of Windows on this device.
- **DecisionMatchingInfoPostUpgrade_RS3** The total DecisionMatchingInfoPostUpgrade objects targeting the next release of Windows on this device.
- **DecisionMediaCenter_RS3** The total DecisionMediaCenter objects targeting the next release of Windows on this device.
- **DecisionSystemBios_RS3** The total DecisionSystemBios objects targeting the next release of Windows on this device.
- **Wmdrm_RS3** The total Wmdrm objects targeting the next release of Windows on this device.

Microsoft.Windows.Appraiser.General.InventoryUplevelDriverPackageStartSync

This event indicates that a new set of InventoryUplevelDriverPackageAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemProcessorLahfSahfStartSync

This event indicates that a new set of SystemProcessorLahfSahfAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemProcessorSse2StartSync

This event indicates that a new set of SystemProcessorSse2Add events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.InventorySystemBiosStartSync

This event indicates that a new set of InventorySystemBiosAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionSystemBiosStartSync

This event indicates that a new set of DecisionSystemBiosAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemMemoryStartSync

This event indicates that a new set of SystemMemoryAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemProcessorCompareExchangeStartSync

This event indicates that a new set of SystemProcessorCompareExchangeAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemProcessorNxStartSync

This event indicates that a new set of SystemProcessorNxAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemProcessorPrefetchWStartSync

This event indicates that a new set of SystemProcessorPrefetchWAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemWimStartSync

This event indicates that a new set of SystemWimAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DatasourceSystemBiosStartSync

This event indicates that a new set of DatasourceSystemBiosAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemTouchStartSync

This event indicates that a new set of SystemTouchAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DatasourceDriverPackageEndSync

This event indicates that a full set of DatasourceDriverPackageAdd events has been sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemWlanStartSync

This event indicates that a new set of SystemWlanAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemWindowsActivationStatusStartSync

This event indicates that a new set of SystemWindowsActivationStatusAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionMediaCenterStartSync

This event indicates that a new set of DecisionMediaCenterAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.InventoryMediaCenterStartSync

This event indicates that a new set of InventoryMediaCenterAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionMatchingInfoPassiveStartSync

This event indicates that a new set of DecisionMatchingInfoPassiveAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPassiveStartSync

This event indicates that a new set of DataSourceMatchingInfoPassiveAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.InventoryApplicationFileStartSync

This event indicates indicates that a new set of InventoryApplicationFileAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionMatchingInfoPostUpgradeStartSync

This event indicates that a new set of DecisionMatchingInfoPostUpgradeAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.WmdrmStartSync

This event indicates that a new set of WmdrmAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPassiveEndSync

This event indicates that a full set of DataSourceMatchingInfoPassiveAdd events have been sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionMatchingInfoBlockStartSync

This event indicates that a new set of DecisionMatchingInfoBlockAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DatasourceApplicationFileStartSync

This event indicates that a new set of DatasourceApplicationFileAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DatasourceDevicePnpStartSync

This event indicates that a new set of DatasourceDevicePnpAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoBlockStartSync

This event indicates that a full set of DataSourceMatchingInfoBlockStAdd events have been sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionApplicationFileStartSync

This event indicates that a new set of DecisionApplicationFileAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.InventoryLanguagePackStartSync

This event indicates that a new set of InventoryLanguagePackAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPostUpgradeStartSync

This event indicates that a new set of DataSourceMatchingInfoPostUpgradeAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionDevicePnpStartSync

This event indicates that the DecisionDevicePnp object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DatasourceDriverPackageStartSync

This event indicates that a new set of DatasourceDriverPackageAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionDriverPackageStartSync

This event indicates that a new set of DecisionDriverPackageAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.WmdrmAdd

This event sends data about the usage of older digital rights management on the system, to help keep Windows up to date. This data does not indicate the details of the media using the digital rights management, only whether any such files exist. Collecting this data was critical to ensuring the correct mitigation for customers, and should be able to be removed once all mitigations are in place.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **BlockingApplication** Same as NeedsDismissAction
- **NeedsDismissAction** Indicates if a dismissible message is needed to warn the user about a potential loss of data due to DRM deprecation.
- **WmdrmApiResult** Raw value of the API used to gather DRM state.
- **WmdrmCdRipped** Indicates if the system has any files encrypted with personal DRM, which was used for ripped CDs.
- **WmdrmIndicators** WmdrmCdRipped OR WmdrmPurchased
- **WmdrmInUse** WmdrmIndicators AND dismissible block in setup was not dismissed.
- **WmdrmNonPermanent** Indicates if the system has any files with non-permanent licenses.
- **WmdrmPurchased** Indicates if the system has any files with permanent licenses.

Microsoft.Windows.Appraiser.General.InventoryUplevelDriverPackageAdd

This event is only runs during setup. It provides a listing of the uplevel driver packages that were downloaded before the upgrade. Is critical to understanding if failures in setup can be traced to not having sufficient uplevel drivers before the upgrade.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **BootCritical** Is the driver package marked as boot critical?

- **Build** The build value from the driver package.
- **CatalogFile** The name of the catalog file within the driver package.
- **Class** The device class from the driver package.
- **ClassGuid** The device class GUID from the driver package.
- **Date** The date from the driver package.
- **Inbox** Is the driver package of a driver that is included with Windows?
- **OriginalName** The original name of the INF file before it was renamed. Generally a path under \$WINDOWS.~BT\Drivers\DU
- **Provider** The provider of the driver package.
- **PublishedName** The name of the INF file, post-rename.
- **Revision** The revision of the driver package.
- **SignatureStatus** Indicates if the driver package is signed. Unknown:0, Unsigned:1, Signed: 2
- **VersionMajor** The major version of the driver package.
- **VersionMinor** The minor version of the driver package.

Microsoft.Windows.Appraiser.General.GatedRegChange

This event sends data about the results of running a set of quick-blocking instructions, to help keep Windows up to date.

The following fields are available:

- **NewData** The data in the registry value after the scan completed.
- **OldData** The previous data in the registry value before the scan ran.
- **PCFP** An ID for the system calculated by hashing hardware identifiers.
- **RegKey** The registry key name for which a result is being sent.
- **RegValue** The registry value for which a result is being sent.
- **Time** The client time of the event.

Microsoft.Windows.Appraiser.General.DatasourceApplicationFileRemove

This event indicates that the DatasourceApplicationFile object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DatasourceDevicePnpRemove

This event indicates that the DatasourceDevicePnp object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DatasourceDriverPackageRemove

This event indicates that the DatasourceDriverPackage object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemProcessorSse2Remove

This event indicates that the SystemProcessorSse2 object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.InventoryUplevelDriverPackageRemove

This event indicates that the InventoryUplevelDriverPackage object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionMediaCenterRemove

This event indicates that the DecisionMediaCenter object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.InventoryMediaCenterRemove

This event indicates that the InventoryMediaCenter object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DatasourceSystemBiosRemove

This event indicates that the DatasourceSystemBios object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionApplicationFileRemove

This event indicates Indicates that the DecisionApplicationFile object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionMatchingInfoPostUpgradeRemove

This event indicates that the DecisionMatchingInfoPostUpgrade object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemTouchRemove

"This event indicates that the SystemTouch object is no longer present. "

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemWindowsActivationStatusRemove

This event indicates that the SystemWindowsActivationStatus object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemWlanRemove

"This event indicates that the SystemWlan object is no longer present. "

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPostUpgradeRemove

This event indicates that the DataSourceMatchingInfoPostUpgrade object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemProcessorNxRemove

This event indicates that the SystemProcessorNx object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoBlockRemove

This event indicates that the DataSourceMatchingInfoBlock object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionDevicePnpRemove

This event indicates that the DecisionDevicePnp object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionMatchingInfoPassiveRemove

This event Indicates that the DecisionMatchingInfoPassive object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemMemoryRemove

This event that the SystemMemory object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionMatchingInfoBlockRemove

This event indicates that the DecisionMatchingInfoBlock object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPassiveRemove

This event indicates that the DataSourceMatchingInfoPassive object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.InventoryApplicationFileRemove

This event indicates that the InventoryApplicationFile object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemWimRemove

"This event indicates that the SystemWim object is no longer present. "

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.InventorySystemBiosRemove

"This event indicates that the InventorySystemBios object is no longer present. "

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.WmdrmRemove

This event indicates that the Wmdrm object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemProcessorLahfSahfRemove

"This event indicates that the SystemProcessorLahfSahf object is no longer present. "

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.InventoryLanguagePackRemove

This event indicates that the InventoryLanguagePack object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionDriverPackageRemove

This event indicates that the DecisionDriverPackage object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionSystemBiosRemove

This event indicates that the DecisionSystemBios object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemProcessorCompareExchangeRemove

"This event indicates that the SystemProcessorCompareExchange object is no longer present. "

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemProcessorPrefetchWRemove

This event indicates that the SystemProcessorPrefetchW object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.InventoryDriverBinaryEndSync

This event indicates that a full set of InventoryDriverBinaryAdd events has been sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Census events

Census.Battery

This event sends type and capacity data about the battery on the device, as well as the number of connected standby devices in use, type to help keep Windows up to date.

The following fields are available:

- **InternalBatteryCapabilties** Represents information about what the battery is capable of doing.
- **InternalBatteryCapacityCurrent** Represents the battery's current fully charged capacity in mWh (or relative). Compare this value to DesignedCapacity to estimate the battery's wear.
- **InternalBatteryCapacityDesign** Represents the theoretical capacity of the battery when new, in mWh.
- **InternalBatteryNumberOfCharges** Provides the number of battery charges. This is used when creating new products and validating that existing products meets targeted functionality performance.
- **IsAlwaysOnAlwaysConnectedCapable** Represents whether the battery enables the device to be AlwaysOnAlwaysConnected . Boolean value.

Census.Enterprise

This event sends data about Azure presence, type, and cloud domain use in order to provide an understanding of the use and integration of devices in an enterprise, cloud, and server environment.

The following fields are available:

- **AzureOSIDPresent** Represents the field used to identify an Azure machine.
- **AzureVMTType** Represents whether the instance is Azure VM PAAS, Azure VM IAAS or any other VMs.
- **CDJType** Represents the type of cloud domain joined for the machine.
- **CommercialId** Represents the GUID for the commercial entity which the device is a member of. Will be used to reflect insights back to customers.
- **ContainerType** The type of container, such as process or virtual machine hosted.
- **EnrollmentType** Represents the type of enrollment, such as MDM or Intune, for a particular device.
- **HashedDomain** The hashed representation of the user domain used for login.
- **IsCloudDomainJoined** Is this device joined to an Azure Active Directory (AAD) tenant? true/false
- **IsDERequirementMet** Represents if the device can do device encryption.
- **IsDeviceProtected** Represents if Device protected by BitLocker/Device Encryption
- **IsDomainJoined** Indicates whether a machine is joined to a domain.
- **IsEDPEnabled** Represents if Enterprise data protected on the device.
- **IsMDMEnrolled** Whether the device has been MDM Enrolled or not.
- **MPNId** Returns the Partner ID/MPN ID from Regkey.
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\DeployID
- **SCCMClientId** This ID correlate systems that send data to Compat Analytics (OMS) and other OMS based systems with systems in an Enterprise SCCM environment.
- **ServerFeatures** Represents the features installed on a Windows Server. This can be used by developers and administrators who need to automate the process of determining the features installed on a set of server computers.
- **SystemCenterID** The SCCM ID is an anonymized one-way hash of the Active Directory Organization identifier

Census.App

This event sends version data about the Apps running on this device, to help keep Windows up to date.

The following fields are available:

- **CensusVersion** The version of Census that generated the current data for this device.
- **IEVersion** Retrieves which version of Internet Explorer is running on this device.

Census.Camera

This event sends data about the resolution of cameras on the device, to help keep Windows up to date.

The following fields are available:

- **FrontFacingCameraResolution** Represents the resolution of the front facing camera in megapixels. If a front facing camera does not exist, then the value is 0.
- **RearFacingCameraResolution** Represents the resolution of the rear facing camera in megapixels. If a rear facing camera does not exist, then the value is 0.

Census.UserDisplay

This event sends data about the logical/physical display size, resolution and number of internal/external displays, and VRAM on the system, to help keep Windows up to date.

The following fields are available:

- **InternalPrimaryDisplayLogicalDPIX** Retrieves the logical DPI in the x-direction of the internal display.
- **InternalPrimaryDisplayLogicalDPIY** Retrieves the logical DPI in the y-direction of the internal display.
- **InternalPrimaryDisplayPhysicalDPIX** Retrieves the physical DPI in the x-direction of the internal display.
- **InternalPrimaryDisplayPhysicalDPIY** Retrieves the physical DPI in the y-direction of the internal display.
- **InternalPrimaryDisplayResolutionHorizontal** Retrieves the number of pixels in the horizontal direction of the internal display.
- **InternalPrimaryDisplayResolutionVertical** Retrieves the number of pixels in the vertical direction of the internal display.
- **InternalPrimaryDisplaySizePhysicalH** Retrieves the physical horizontal length of the display in mm. Used for calculating the diagonal length in inches .
- **InternalPrimaryDisplaySizePhysicalY** Retrieves the physical vertical length of the display in mm. Used for calculating the diagonal length in inches
- **InternalPrimaryDisplayType** Represents the type of technology used in the monitor, such as Plasma, LED, LCOS, etc.
- **NumberofExternalDisplays** Retrieves the number of external displays connected to the machine
- **NumberofInternalDisplays** Retrieves the number of internal displays in a machine.
- **VRAMDedicated** Retrieves the video RAM in MB.
- **VRAMDedicatedSystem** Retrieves the amount of memory on the dedicated video card.
- **VRAMSharedSystem** Retrieves the amount of RAM memory that the video card can use.

Census.Firmware

This event sends data about the BIOS and startup embedded in the device, to help keep Windows up to date.

The following fields are available:

- **FirmwareManufacturer** Represents the manufacturer of the device's firmware (BIOS).
- **FirmwareReleaseDate** Represents the date the current firmware was released.
- **FirmwareType** Represents the firmware type. The various types can be unknown, BIOS, UEFI.
- **FirmwareVersion** Represents the version of the current firmware.

Census.Flighting

This event sends Windows Insider data from customers participating in improvement testing and feedback programs, to help keep Windows up-to-date.

The following fields are available:

- **DeviceSampleRate** The telemetry sample rate assigned to the device.
- **EnablePreviewBuilds** Used to enable Windows Insider builds on a device.
- **FlightIds** A list of the different Windows Insider builds on this device.
- **FlightingBranchName** The name of the Windows Insider branch currently used by the device.
- **IsFlightsDisabled** Represents if the device is participating in the Windows Insider program.
- **MSA_Accounts** Represents a list of hashed IDs of the Microsoft Accounts that are flighting (pre-release builds) on this device.
- **SSRK** Retrieves the mobile targeting settings.

Census.Hardware

This event sends data about the device, including hardware type, OEM brand, model line, model, telemetry level setting, and TPM support, to help keep Windows up-to-date.

The following fields are available:

- **ActiveMicCount** The number of active microphones attached to the device.
- **ChassisType** Represents the type of device chassis, such as desktop or low profile desktop. The possible values can range between 1 - 36.
- **ComputerHardwareID** Identifies a device class that is represented by a hash of different SMBIOS fields.
- **D3DMaxFeatureLevel** The supported Direct3D version.
- **DeviceColor** Indicates a color of the device.
- **DeviceForm** Indicates the form as per the device classification.
- **DeviceName** The device name that is set by the user.
- **DigitizerSupport** Is a digitizer supported?
- **DUID** The device unique ID.
- **Gyroscope** Indicates whether the device has a gyroscope.
- **InventoryId** The device ID used for compatibility testing.
- **Magnetometer** Indicates whether the device has a magnetometer.
- **NFCProximity** Indicates whether the device supports NFC.
- **OEMDigitalMarkerFileName** The name of the file placed in the \Windows\system32\drivers directory that specifies the OEM and model name of the device.
- **OEMManufacturerName** The device manufacturer name. The OEMName for an inactive device is not reprocessed even if the clean OEM name is changed at a later date.
- **OEMModelBaseBoard** The baseboard model used by the OEM.
- **OEMModelBaseBoardVersion** Differentiates between developer and retail devices.
- **OEMModelName** The device model name.
- **OEMModelNumber** The device model number.
- **OEMModelSKU** The device edition that is defined by the manufacturer.
- **OEMModelSystemFamily** The system family set on the device by an OEM.
- **OEMModelSystemVersion** The system model version set on the device by the OEM.
- **OEMOptionalIdentifier** A Microsoft assigned value that represents a specific OEM subsidiary.
- **OEMSerialNumber** The serial number of the device that is set by the manufacturer.
- **PhoneManufacturer** The friendly name of the phone manufacturer.
- **PowerPlatformRole** The OEM preferred power management profile. It's used to help to identify the basic form

factor of the device.

- **SoCName** The firmware manufacturer of the device.
- **StudyID** Used to identify retail and non-retail device.
- **TelemetryLevel** The telemetry level the user has opted into, such as Basic or Enhanced.
- **TelemetryLevelLimitEnhanced** The telemetry level for Windows Analytics-based solutions.
- **TelemetrySettingAuthority** Determines who set the telemetry level, such as GP, MDM, or the user.
- **TPMVersion** The supported Trusted Platform Module (TPM) on the device. If no TPM is present, the value is 0.
- **VoiceSupported** Does the device have a cellular radio capable of making voice calls?

Census.Memory

This event sends data about the memory on the device, including ROM and RAM, to help keep Windows up to date.

The following fields are available:

- **TotalPhysicalRAM** Represents the physical memory (in MB).
- **TotalVisibleMemory** Represents the memory that is not reserved by the system.

Census.Network

This event sends data about the mobile and cellular network used by the device (mobile service provider, network, device ID, and service cost factors), to help keep Windows up to date.

The following fields are available:

- **IMEI0** Represents the International Mobile Station Equipment Identity. This number is usually unique and used by the mobile operator to distinguish different phone hardware. Microsoft does not have access to mobile operator billing data so collecting this data does not expose or identify the user. The two fields represent phone with dual sim coverage.
- **IMEI1** Represents the International Mobile Station Equipment Identity. This number is usually unique and used by the mobile operator to distinguish different phone hardware. Microsoft does not have access to mobile operator billing data so collecting this data does not expose or identify the user. The two fields represent phone with dual sim coverage.
- **MCC0** Represents the Mobile Country Code (MCC). It used with the Mobile Network Code (MNC) to uniquely identify a mobile network operator. The two fields represent phone with dual sim coverage.
- **MCC1** Represents the Mobile Country Code (MCC). It used with the Mobile Network Code (MNC) to uniquely identify a mobile network operator. The two fields represent phone with dual sim coverage.
- **MEID** Represents the Mobile Equipment Identity (MEID). MEID is a worldwide unique phone ID assigned to CDMA phones. MEID replaces electronic serial number (ESN), and is equivalent to IMEI for GSM and WCDMA phones. Microsoft does not have access to mobile operator billing data so collecting this data does not expose or identify the user.
- **MNC0** Retrieves the Mobile Network Code (MNC). It used with the Mobile Country Code (MCC) to uniquely identify a mobile network operator. The two fields represent phone with dual sim coverage.
- **MNC1** Retrieves the Mobile Network Code (MNC). It used with the Mobile Country Code (MCC) to uniquely identify a mobile network operator. The two fields represent phone with dual sim coverage.
- **MobileOperatorBilling** Represents the telephone company that provides services for mobile phone users.
- **MobileOperatorCommercialized** Represents which reseller and geography the phone is commercialized for. This is the set of values on the phone for who and where it was intended to be used. For example, the commercialized mobile operator code AT&T in the US would be ATT-US.
- **MobileOperatorNetwork0** Represents the operator of the current mobile network that the device is used on. (AT&T, T-Mobile, Vodafone). The two fields represent phone with dual sim coverage.
- **MobileOperatorNetwork1** Represents the operator of the current mobile network that the device is used on. (AT&T, T-Mobile, Vodafone). The two fields represent phone with dual sim coverage.

- **NetworkAdapterGUID** The GUID of the primary network adapter.
- **NetworkCost** Represents the network cost associated with a connection.
- **SPN0** Retrieves the Service Provider Name (SPN). For example, these might be AT&T, Sprint, T-Mobile, or Verizon. The two fields represent phone with dual sim coverage.
- **SPN1** Retrieves the Service Provider Name (SPN). For example, these might be AT&T, Sprint, T-Mobile, or Verizon. The two fields represent phone with dual sim coverage.

Census.OS

This event sends data about the operating system such as the version, locale, update service configuration, when and how it was originally installed, and whether it is a virtual device, to help keep Windows up to date.

The following fields are available:

- **ActivationChannel** Retrieves the retail license key or Volume license key for a machine.
- **AssignedAccessStatus** The kiosk configuration mode.
- **CompactOS** Indicates if the Compact OS feature from Win10 is enabled.
- **DeveloperUnlockStatus** "Represents if a device has been developer unlocked by the user or Group Policy. "
- **DeviceTimeZone** The time zone that is set on the device. Example: Pacific Standard Time
- **GenuineState** Retrieves the ID Value specifying the OS Genuine check.
- **InstallationType** Retrieves the type of OS installation. (Clean, Upgrade, Reset, Refresh, Update).
- **InstallLanguage** The first language installed on the user machine.
- **IsDeviceRetailDemo** Retrieves if the device is running in demo mode.
- **IsEduData** Returns Boolean if the education data policy is enabled.
- **IsPortableOperatingSystem** Retrieves whether OS is running Windows-To-Go
- **IsSecureBootEnabled** Retrieves whether Boot chain is signed under UEFI.
- **LanguagePacks** The list of language packages installed on the device.
- **LicenseStateReason** Retrieves why (or how) a system is licensed or unlicensed. The HRESULT may indicate an error code that indicates a key blocked error, or it may indicate that we are running an OS License granted by the MS store.
- **OA3xOriginalProductKey** Retrieves the License key stamped by the OEM to the machine.
- **OSEdition** Retrieves the version of the current OS.
- **OSInstallDateTime** Retrieves the date the OS was installed using ISO 8601 (Date part) == yyyy-mm-dd
- **OSInstallType** Retrieves a numeric description of what install was used on the device i.e. clean, upgrade, refresh, reset, etc
- **OSOOBEDateTime** Retrieves Out of Box Experience (OOBE) Date in Coordinated Universal Time (UTC).
- **OSSKU** Retrieves the Friendly Name of OS Edition.
- **OSSubscriptionStatus** Represents the existing status for enterprise subscription feature for PRO machines.
- **OSSubscriptionTypeId** Returns boolean for enterprise subscription feature for selected PRO machines.
- **OSTimeZoneBiasInMins** Retrieves the time zone set on machine.
- **OSUILocale** Retrieves the locale of the UI that is currently used by the OS.
- **ProductActivationResult** Returns Boolean if the OS Activation was successful.
- **ProductActivationTime** Returns the OS Activation time for tracking piracy issues.
- **ProductKeyID2** Retrieves the License key if the machine is updated with a new license key.
- **RACw7Id** Retrieves the Microsoft Reliability Analysis Component (RAC) Win7 Identifier. RAC is used to monitor and analyze system usage and reliability.
- **ServiceMachineIP** Retrieves the IP address of the KMS host used for anti-piracy.
- **ServiceMachinePort** Retrieves the port of the KMS host used for anti-piracy.
- **ServiceProductKeyID** Retrieves the License key of the KMS
- **SharedPCMode** Returns Boolean for education devices used as shared cart

- **Signature** Retrieves if it is a signature machine sold by Microsoft store.
- **SLICStatus** Whether a SLIC table exists on the device.
- **SLICVersion** Returns OS type/version from SLIC table.

Census.Processor

This event sends data about the processor (architecture, speed, number of cores, manufacturer, and model number), to help keep Windows up to date.

The following fields are available:

- **ProcessorArchitecture** Retrieves the processor architecture of the installed operating system. The complete list of values can be found in DimProcessorArchitecture.
- **ProcessorClockSpeed** Retrieves the clock speed of the processor in MHz.
- **ProcessorCores** Retrieves the number of cores in the processor.
- **ProcessorIdentifier** The processor identifier of a manufacturer.
- **ProcessorManufacturer** Retrieves the name of the processor's manufacturer.
- **ProcessorModel** Retrieves the name of the processor model.
- **ProcessorPhysicalCores** Number of physical cores in the processor.
- **ProcessorUpdateRevision** The microcode version.
- **SocketCount** Number of physical CPU sockets of the machine.

Census.Storage

This event sends data about the total capacity of the system volume and primary disk, to help keep Windows up to date.

The following fields are available:

- **PrimaryDiskTotalCapacity** Retrieves the amount of disk space on the primary disk of the device in MB.
- **PrimaryDiskType** Retrieves an enumerator value of type STORAGE_BUS_TYPE that indicates the type of bus to which the device is connected. This should be used to interpret the raw device properties at the end of this structure (if any).
- **SystemVolumeTotalCapacity** Retrieves the size of the partition that the System volume is installed on in MB.

Census.VM

This event sends data indicating whether virtualization is enabled on the device, and its various characteristics, to help keep Windows up to date.

The following fields are available:

- **CloudService** Indicates which cloud service, if any, that this virtual machine is running within.
- **HyperVisor** Retrieves whether the current OS is running on top of a Hypervisor.
- **IOMMUPresent** Represents if an input/output memory management unit (IOMMU) is present.
- **isVDI** Is the device using Virtual Desktop Infrastructure?
- **IsVirtualDevice** Retrieves that when the Hypervisor is Microsoft's Hyper-V Hypervisor or other Hv#HASH#1 Hypervisor, this field will be set to FALSE for the Hyper-V host OS and TRUE for any guest OS's. This field should not be relied upon for non-Hv#HASH#1 Hypervisors.
- **SLATSupported** Represents whether Second Level Address Translation (SLAT) is supported by the hardware.
- **VirtualizationFirmwareEnabled** Represents whether virtualization is enabled in the firmware.

Census.Xbox

This event sends data about the Xbox Console, such as Serial Number and Deviceld, to help keep Windows up to date.

The following fields are available:

- **XboxConsolePreferredLanguage** Retrieves the preferred language selected by the user on Xbox console.
- **XboxConsoleSerialNumber** Retrieves the serial number of the Xbox console.
- **XboxLiveDeviceId** Retrieves the unique device id of the console.
- **XboxLiveSandboxId** Retrieves the developer sandbox id if the device is internal to MS.

Census.Userdefault

This event sends data about the current user's default preferences for browser and several of the most popular extensions and protocols, to help keep Windows up to date.

The following fields are available:

- **DefaultApp** The current user's default program selected for the following extension or protocol:
.html,.htm,.jpg,.jpeg,.png,.mp3,.mp4,.mov,.pdf
- **DefaultBrowserProgId** The ProgramId of the current user's default browser

Census.UserNLS

This event sends data about the default app language, input, and display language preferences set by the user, to help keep Windows up to date.

The following fields are available:

- **DefaultAppLanguage** The current user Default App Language.
- **DisplayLanguage** The current user preferred Windows Display Language.
- **HomeLocation** The current user location, which is populated using GetUserGeoid() function.
- **KeyboardInputLanguages** The Keyboard input languages installed on the device.
- **SpeechInputLanguages** The Speech Input languages installed on the device.

Census.WU

This event sends data about the Windows update server and other App store policies, to help keep Windows up to date.

The following fields are available:

- **AppraiserGatedStatus** Indicates whether a device has been gated for upgrading.
- **AppStoreAutoUpdate** Retrieves the Appstore settings for auto upgrade. (Enable/Disabled).
- **AppStoreAutoUpdateMDM** Retrieves the App Auto Update value for MDM: 0 - Disallowed. 1 - Allowed. 2 - Not configured. Default: [2] Not configured
- **AppStoreAutoUpdatePolicy** Retrieves the Windows Store App Auto Update group policy setting
- **DelayUpgrade** Retrieves the Windows upgrade flag for delaying upgrades.
- **OSAssessmentFeatureOutOfDate** How many days has it been since a the last feature update was released but the device did not install it?
- **OSAssessmentForFeatureUpdate** Is the device is on the latest feature update?
- **OSAssessmentForQualityUpdate** Is the device on the latest quality update?
- **OSAssessmentForSecurityUpdate** Is the device on the latest security update?
- **OSAssessmentQualityOutOfDate** How many days has it been since a the last quality update was released but the device did not install it?
- **OSAssessmentReleaseInfoTime** The freshness of release information used to perform an assessment.
- **OSRollbackCount** The number of times feature updates have rolled back on the device.
- **OSRolledBack** A flag that represents when a feature update has rolled back during setup.
- **OSUninstalled** A flag that represents when a feature update is uninstalled on a device .
- **OSWUAutoUpdateOptions** Retrieves the auto update settings on the device.
- **UninstallActive** A flag that represents when a device has uninstalled a previous upgrade recently.

- **UpdateServiceURLConfigured** Retrieves if the device is managed by Windows Server Update Services (WSUS).
- **WUDeferUpdatePeriod** Retrieves if deferral is set for Updates
- **WUDeferUpgradePeriod** Retrieves if deferral is set for Upgrades
- **WUDODownloadMode** Retrieves whether DO is turned on and how to acquire/distribute updates Delivery Optimization (DO) allows users to deploy previously downloaded WU updates to other devices on the same network.
- **WUMachineId** Retrieves the Windows Update (WU) Machine Identifier.
- **WUPauseState** Retrieves WU setting to determine if updates are paused
- **WUServer** Retrieves the HTTP(S) URL of the WSUS server that is used by Automatic Updates and API callers (by default).

Census.Speech

This event is used to gather basic speech settings on the device.

The following fields are available:

- **AboveLockEnabled** Cortana setting that represents if Cortana can be invoked when the device is locked.
- **GPAallowInputPersonalization** Indicates if a Group Policy setting has enabled speech functionalities.
- **HolographicSpeechInputDisabled** Holographic setting that represents if the attached HMD devices have speech functionality disabled by the user.
- **HolographicSpeechInputDisabledRemote** Indicates if a remote policy has disabled speech functionalities for the HMD devices.
- **KWSEnabled** "Cortana setting that represents if a user has enabled the ""Hey Cortana"" keyword spotter (KWS)."
- **MDMAallowInputPersonalization** Indicates if an MDM policy has enabled speech functionalities.
- **RemotelyManaged** Indicates if the device is being controlled by a remote administrator (MDM or Group Policy) in the context of speech functionalities.
- **SpeakerIdEnabled** Cortana setting that represents if keyword detection has been trained to try to respond to a single user's voice.
- **SpeechServicesEnabled** Windows setting that represents whether a user is opted-in for speech services on the device.

Census.Security

This event provides information on about security settings used to help keep Windows up-to-date and secure.

- **AvailableSecurityProperties** Enumerates and reports state on the relevant security properties for Device Guard.
- **CGRunning** Is Credential Guard running?
- **DGState** A summary of the Device Guard state.
- **HVCIRunning** Is HVCI running?
- **RequiredSecurityProperties** Describes the required security properties to enable virtualization-based security.
- **SecureBootCapable** Is this device capable of running Secure Boot?
- **VBSState** Is virtualization-based security enabled, disabled, or running?

Diagnostic data events

TelClientSynthetic.AuthorizationInfo_Startup

This event sends data indicating that a device has undergone a change of telemetry opt-in level detected at UTC startup, to help keep Windows up to date.

The following fields are available:

- **CanAddMsaToMsTelemetry** True if UTC is allowed to add MSA user identity onto telemetry from the OS provider groups.
- **CanCollectAnyTelemetry** True if UTC is allowed to collect non-OS telemetry. Non-OS telemetry is responsible for providing its own opt-in mechanism.
- **CanCollectCoreTelemetry** True if UTC is allowed to collect data which is tagged with both MICROSOFT_KEYWORD_CRITICAL_DATA and MICROSOFT_EVENTTAG_CORE_DATA.
- **CanCollectHeartbeats** True if UTC is allowed to collect heartbeats.
- **CanCollectOsTelemetry** True if UTC is allowed to collect telemetry from the OS provider groups (often called Microsoft Telemetry).
- **CanPerformDiagnosticEscalations** True if UTC is allowed to perform all scenario escalations.
- **CanPerformScripting** True if UTC is allowed to perform scripting.
- **CanPerformTraceEscalations** True if UTC is allowed to perform scenario escalations with tracing actions.
- **CanReportScenarios** True if UTC is allowed to load and report scenario completion, failure, and cancellation events.
- **PreviousPermissions** Bitmask representing the previously configured permissions since the telemetry client was last started.
- **TransitionFromEverythingOff** True if this transition is moving from not allowing core telemetry to allowing core telemetry.

TelClientSynthetic.AuthorizationInfo_RuntimeTransition

This event sends data indicating that a device has undergone a change of telemetry opt-in level during the runtime of the device (not at UTC boot or offline), to help keep Windows up to date.

The following fields are available:

- **CanAddMsaToMsTelemetry** True if UTC is allowed to add MSA user identity onto telemetry from the OS provider groups.
- **CanCollectAnyTelemetry** True if UTC is allowed to collect non-OS telemetry. Non-OS telemetry is responsible for providing its own opt-in mechanism.
- **CanCollectCoreTelemetry** True if UTC is allowed to collect data which is tagged with both MICROSOFT_KEYWORD_CRITICAL_DATA and MICROSOFT_EVENTTAG_CORE_DATA.
- **CanCollectHeartbeats** True if UTC is allowed to collect heartbeats.
- **CanCollectOsTelemetry** True if UTC is allowed to collect telemetry from the OS provider groups (often called Microsoft Telemetry).
- **CanPerformDiagnosticEscalations** True if UTC is allowed to perform all scenario escalations.
- **CanPerformScripting** True if UTC is allowed to perform scripting.
- **CanPerformTraceEscalations** True if UTC is allowed to perform scenario escalations with tracing actions.
- **CanReportScenarios** True if UTC is allowed to load and report scenario completion, failure, and cancellation events.
- **PreviousPermissions** Bitmask representing the previously configured permissions since the telemetry opt-in level was last changed.
- **TransitionFromEverythingOff** True if this transition is moving from not allowing core telemetry to allowing core telemetry.

TelClientSynthetic.ConnectivityHeartBeat_0

This event sends data about the connectivity status of the Connected User Experience and Telemetry component that uploads telemetry events. If an unrestricted free network (such as Wi-Fi) is available, this event updates the last successful upload time. Otherwise, it checks whether a Connectivity Heartbeat event was fired in the past 24 hours, and if not, it fires an event. A Connectivity Heartbeat event also fires when a device recovers from costed

network to free network.

The following fields are available:

- **CensusExitCode** Returns last execution codes from census client run.
- **CensusStartTime** Returns timestamp corresponding to last successful census run.
- **CensusTaskEnabled** Returns Boolean value for the census task (Enable/Disable) on client machine.
- **LastConnectivityLossTime** Retrieves the last time the device lost free network.
- **LastConnectivityLossTime** Retrieves the last time the device lost free network.
- **NetworkState** Retrieves the network state: 0 = No network. 1 = Restricted network. 2 = Free network.
- **NoNetworkTime** Retrieves the time spent with no network (since the last time) in seconds.
- **RestrictedNetworkTime** Retrieves the time spent on a metered (cost restricted) network in seconds.

TelClientSynthetic.HeartBeat_5

This event sends data about the health and quality of the telemetry data from the given device, to help keep Windows up to date. It also enables data analysts to determine how 'trusted' the data is from a given device.

The following fields are available:

- **AgentConnectionErrorsCount** The number of non-timeout errors associated with the host/agent channel.
- **CensusExitCode** The last exit code of the Census task.
- **CensusStartTime** The time of the last Census run.
- **CensusTaskEnabled** Indicates whether Census is enabled.
- **ConsumerDroppedCount** The number of events dropped by the consumer layer of the telemetry client.
- **CriticalDataDbDroppedCount** The number of critical data sampled events that were dropped at the database layer.
- **CriticalDataThrottleDroppedCount** The number of critical data sampled events that were dropped because of throttling.
- **CriticalOverflowEntersCounter** The number of times a critical overflow mode was entered into the event database.
- **DbCriticalDroppedCount** The total number of dropped critical events in the event database.
- **DbDroppedCount** The number of events that were dropped because the database was full.
- **DecodingDroppedCount** The number of events dropped because of decoding failures.
- **EnteringCriticalOverflowDroppedCounter** The number of events that was dropped because a critical overflow mode was initiated.
- **EtwDroppedBufferCount** The number of buffers dropped in the CUET ETW session.
- **EtwDroppedCount** The number of events dropped by the ETW layer of the telemetry client.
- **EventSubStoreResetCounter** The number of times the event database was reset.
- **EventSubStoreResetSizeSum** The total size of the event database across all resets reports in this instance.
- **EventsUploaded** The number of events that have been uploaded.
- **Flags** Flags that indicate device state, such as network, battery, and opt-in state.
- **FullTriggerBufferDroppedCount** The number of events that were dropped because the trigger buffer was full.
- **HeartBeatSequenceNumber** A monotonically increasing heartbeat counter.
- **InvalidHttpCodeCount** The number of invalid HTTP codes received from Vortex.
- **LastAgentConnectionError** The last non-timeout error that happened in the host/agent channel.
- **LastEventSizeOffender** The name of the last event that exceeded the maximum event size.
- **LastInvalidHttpCode** The last invalid HTTP code received from Vortex.
- **MaxActiveAgentConnectionCount** The maximum number of active agents during this heartbeat timeframe.
- **MaxInUseScenarioCounter** The soft maximum number of scenarios loaded by the Connected User

Experience and Telemetry component.

- **PreviousHeartBeatTime** The time of last heartbeat event. This allows chaining of events.
- **SettingsHttpAttempts** The number of attempts to contact the OneSettings service.
- **SettingsHttpFailures** The number of failures from contacting the OneSettings service.
- **ThrottledDroppedCount** The number of events dropped due to throttling of noisy providers.
- **UploaderDroppedCount** The number of events dropped by the uploader layer of the telemetry client.
- **VortexFailuresTimeout** The number of timeout failures received from Vortex.
- **VortexHttpAttempts** The number of attempts to contact the Vortex service.
- **VortexHttpFailures4xx** The number of 400-499 error codes received from Vortex.
- **VortexHttpFailures5xx** The number of 500-599 error codes received from Vortex.

TelClientSynthetic.PrivacySettingsAfterCreatorsUpdate

This event sends basic data on privacy settings before and after a feature update. This is used to ensure that customer privacy settings are correctly migrated across feature updates.

The following fields are available:

- **PostUpgradeSettings** The privacy settings after a feature update.
- **PreUpgradeSettings** The privacy settings before a feature update.

DxgKernelTelemetry events

DxgKrnTelemetry.GPUAdapterInventoryV2

This event sends basic GPU and display driver information to keep Windows and display drivers up-to-date.

The following fields are available:

- **aiSeqId** The event sequence ID.
- **bootId** The system boot ID.
- **ComputePreemptionLevel** The maximum preemption level supported by GPU for compute payload.
- **DedicatedSystemMemoryB** The amount of system memory dedicated for GPU use (in bytes).
- **DedicatedVideoMemoryB** The amount of dedicated VRAM of the GPU (in bytes).
- **DisplayAdapterLuid** The display adapter LUID.
- **DriverDate** The date of the display driver.
- **DriverRank** The rank of the display driver.
- **DriverVersion** The display driver version.
- **GPUDeviceID** The GPU device ID.
- **GPUPreemptionLevel** The maximum preemption level supported by GPU for graphics payload.
- **GPURevisionID** The GPU revision ID.
- **GPUVendorID** The GPU vendor ID.
- **InterfaceId** The GPU interface ID.
- **IsDisplayDevice** Does the GPU have displaying capabilities?
- **IsHybridDiscrete** Does the GPU have discrete GPU capabilities in a hybrid device?
- **IsHybridIntegrated** Does the GPU have integrated GPU capabilities in a hybrid device?
- **IsLDA** Is the GPU comprised of Linked Display Adapters?
- **IsMiracastSupported** Does the GPU support Miracast?
- **IsMismatchLDA** Is at least one device in the Linked Display Adapters chain from a different vendor?
- **IsMPOSupported** Does the GPU support Multi-Plane Overlays?
- **IsMsMiracastSupported** Are the GPU Miracast capabilities driven by a Microsoft solution?
- **IsPostAdapter** Is this GPU the POST GPU in the device?

- **IsRenderDevice** Does the GPU have rendering capabilities?
- **IsSoftwareDevice** Is this a software implementation of the GPU?
- **MeasureEnabled** Is the device listening to MICROSOFT_KEYWORD_MEASURES?
- **SharedSystemMemoryB** The amount of system memory shared by GPU and CPU (in bytes).
- **SubSystemID** The subsystem ID.
- **SubVendorID** The GPU sub vendor ID.
- **TelemetryEnabled** Is the device listening to MICROSOFT_KEYWORD_TELEMETRY?
- **TellInvEvntTrigger** What triggered this event to be logged? Example: 0 (GPU enumeration) or 1 (DxgKrnITelemetry provider toggling)
- **version** The event version.
- **WDDMVersion** The Windows Display Driver Model version.
- **NumVidPnSources** The number of supported display output sources.
- **NumVidPnTargets** The number of supported display output targets.

Fault Reporting events

Microsoft.Windows.FaultReporting.AppCrashEvent

"This event sends data about crashes for both native and managed applications, to help keep Windows up to date. The data includes information about the crashing process and a summary of its exception record. It does not contain any Watson bucketing information. The bucketing information is recorded in a Windows Error Reporting (WER) event that is generated when the WER client reports the crash to the Watson service, and the WER event will contain the same ReportID (see field 14 of crash event, field 19 of WER event) as the crash event for the crash being reported. AppCrash is emitted once for each crash handled by WER (e.g. from an unhandled exception or FailFast or ReportException). Note that Generic Watson event types (e.g. from PLM) that may be considered crashes"" by a user DO NOT emit this event."

The following fields are available:

- **AppName** The name of the app that has crashed.
- **AppSessionGuid** GUID made up of process ID and is used as a correlation vector for process instances in the telemetry backend.
- **AppTimeStamp** The date/time stamp of the app.
- **AppVersion** The version of the app that has crashed.
- **ExceptionCode** The exception code returned by the process that has crashed.
- **ExceptionOffset** The address where the exception had occurred.
- **Flags** "Flags indicating how reporting is done. For example, queue the report, do not offer JIT debugging, or do not terminate the process after reporting. "
- **ModName** Exception module name (e.g. bar.dll).
- **ModTimeStamp** The date/time stamp of the module.
- **ModVersion** The version of the module that has crashed.
- **PackageFullName** Store application identity.
- **PackageRelativeAppId** Store application identity.
- **ProcessArchitecture** Architecture of the crashing process, as one of the PROCESSOR_ARCHITECTURE_* constants: 0: PROCESSOR_ARCHITECTURE_INTEL. 5: PROCESSOR_ARCHITECTURE_ARM. 9: PROCESSOR_ARCHITECTURE_AMD64. 12: PROCESSOR_ARCHITECTURE_ARM64.
- **ProcessCreateTime** The time of creation of the process that has crashed.
- **ProcessId** The ID of the process that has crashed.
- **ReportId** A GUID used to identify the report. This can be used to track the report across Watson.
- **TargetAppId** The kernel reported AppId of the application being reported.

- **TargetAppVer** The specific version of the application being reported
- **TargetAsId** The sequence number for the hanging process.

Feature update events

Microsoft.Windows.Upgrade.Uninstall.UninstallFailed

This event sends diagnostic data about failures when uninstalling a feature update, to help resolve any issues preventing customers from reverting to a known state

The following fields are available:

- **failureReason** Provides data about the uninstall initialization operation failure
- **hr** Provides the Win32 error code for the operation failure

Microsoft.Windows.Upgrade.Uninstall.UninstallFinalizedAndRebootTriggered

Indicates that the uninstall was properly configured and that a system reboot was initiated

The following fields are available:

- **name** Name of the event

Hang Reporting events

Microsoft.Windows.HangReporting.AppHangEvent

This event sends data about hangs for both native and managed applications, to help keep Windows up to date. It does not contain any Watson bucketing information. The bucketing information is recorded in a Windows Error Reporting (WER) event that is generated when the WER client reports the hang to the Watson service, and the WER event will contain the same ReportID (see field 13 of hang event, field 19 of WER event) as the hang event for the hang being reported. AppHang is reported only on PC devices. It handles classic Win32 hangs and is emitted only once per report. Some behaviors that may be perceived by a user as a hang are reported by app managers (e.g. PLM/RM/EM) as Watson Generics and will not produce AppHang events.

The following fields are available:

- **AppName** The name of the app that has hung.
- **AppSessionGuid** GUID made up of process id used as a correlation vector for process instances in the telemetry backend.
- **AppVersion** The version of the app that has hung.
- **PackageFullName** Store application identity.
- **PackageRelativeAppId** Store application identity.
- **ProcessArchitecture** Architecture of the hung process, as one of the PROCESSOR_ARCHITECTURE_* constants:
0: PROCESSOR_ARCHITECTURE_INTEL. 5: PROCESSOR_ARCHITECTURE_ARM. 9:
PROCESSOR_ARCHITECTURE_AMD64. 12: PROCESSOR_ARCHITECTURE_ARM64.
- **ProcessCreateTime** The time of creation of the process that has hung.
- **ProcessId** The ID of the process that has hung.
- **ReportId** A GUID used to identify the report. This can be used to track the report across Watson.
- **TargetAppId** The kernel reported AppId of the application being reported.
- **TargetAppVer** The specific version of the application being reported.
- **TargetAsId** The sequence number for the hanging process.
- **TypeCode** Bitmap describing the hang type.
- **WaitingOnAppName** If this is a cross process hang waiting for an application, this has the name of the application.
- **WaitingOnAppVersion** If this is a cross process hang, this has the version of the application for which it is

waiting.

- **WaitingOnPackageFullName** If this is a cross process hang waiting for a package, this has the full name of the package for which it is waiting.
- **WaitingOnPackageRelativeAppId** If this is a cross process hang waiting for a package, this has the relative application id of the package.

Inventory events

Microsoft.Windows.Inventory.Core.InventoryDeviceUsbHubClassStartSync

This event indicates that a new set of InventoryDeviceUsbHubClassAdd events will be sent

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events

Microsoft.Windows.Inventory.Core.InventoryDeviceUsbHubClassAdd

This event sends basic metadata about the USB hubs on the device

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events
- **TotalUserConnectablePorts** Total number of connectable USB ports
- **TotalUserConnectableTypeCPorts** Total number of connectable USB Type C ports

Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeVBARuleViolationsAdd

This event provides data on Microsoft Office VBA rule violations, including a rollup count per violation type, giving an indication of remediation requirements for an organization. The event identifier is a unique GUID, associated with the validation rule

The following fields are available:

- **Count** Count of total Microsoft Office VBA rule violations

Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeAddInAdd

This event provides data on the installed Office Add-ins.

- **AddInCLSID** The CLSID key office the Office addin.
- **AddInId** The ID of the Office addin.
- **BinFileTimestamp** The timestamp of the Office addin.
- **BinFileVersion** The version of the Office addin.
- **Description** The description of the Office addin.
- **FileId** The file ID of the Office addin.
- **FriendlyName** The friendly name of the Office addin.
- **FullPath** The full path to the Office addin.
- **LoadBehavior** A UInt32 that describes the load behavior.
- **LoadTime** The load time for the Office addin.
- **OfficeApplication** The Olffice application for this addin.
- **OfficeArchitecture** The architecture of the addin.
- **OfficeVersion** The Office version for this addin.
- **OutlookCrashingAddin** A boolean value that indicates if crashes have been found for this addin.
- **Provider** The provider name for this addin.

Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeVBAAAdd

This event provides a summary rollup count of conditions encountered while performing a local scan of Office

files, analyzing for known VBA programmability compatibility issues between legacy office version and ProPlus, and between 32 and 64-bit versions

The following fields are available:

- **Design** Count of files with design issues found
- **Design_x64** Count of files with 64 bit design issues found
- **DuplicateVBA** Count of files with duplicate VBA code
- **HasVBA** Count of files with VBA code
- **Inaccessible** Count of files that were inaccessible for scanning
- **Issues** Count of files with issues detected
- **Issues_x64** Count of files with 64-bit issues detected
- **IssuesNone** Count of files with no issues detected
- **IssuesNone_x64** Count of files with no 64-bit issues detected
- **Locked** Count of files that were locked, preventing scanning
- **NoVBA** Count of files with no VBA inside
- **Protected** Count of files that were password protected, preventing scanning
- **RemLimited** Count of files that require limited remediation changes
- **RemLimited_x64** Count of files that require limited remediation changes for 64-bit issues
- **RemSignificant** Count of files that require significant remediation changes
- **RemSignificant_x64** Count of files that require significant remediation changes for 64-bit issues
- **Score** Overall compatibility score calculated for scanned content
- **Score_x64** Overall 64-bit compatibility score calculated for scanned content
- **Total** Total number of files scanned
- **Validation** Count of files that require additional manual validation
- **Validation_x64** Count of files that require additional manual validation for 64-bit issues

Microsoft.Windows.Inventory.Core.InventoryApplicationFrameworkStartSync

This event indicates that a new set of InventoryApplicationFrameworkAdd events will be sent

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events

Microsoft.Windows.Inventory.Core.InventoryApplicationFrameworkAdd

This event provides the basic metadata about the frameworks an application may depend on

The following fields are available:

- **FileId** A hash that uniquely identifies a file
- **Frameworks** The list of frameworks this file depends on
- **InventoryVersion** The version of the inventory file generating the events
- **ProgramId** A hash of the Name, Version, Publisher, and Language of an application used to identify it

Microsoft.Windows.Inventory.Indicators.InventoryMiscellaneousUexIndicatorAdd

These events represent the basic metadata about the OS indicators installed on the system which are used for keeping the device up-to-date.

The following fields are available:

- **IndicatorValue** The indicator value
- **Value** Describes an operating system indicator that may be relevant for the device upgrade.

Microsoft.Windows.Inventory.Indicators.Checksum

This event summarizes the counts for the InventoryMiscellaneousUexIndicatorAdd events.

The following fields are available:

- **ChecksumDictionary** A count of each operating system indicator.
- **PCFP** Equivalent to the InventoryId field that is found in other core events.

Microsoft.Windows.Inventory.Core.InventoryApplicationAdd

This event sends basic metadata about an application on the system to help keep Windows up to date.

The following fields are available:

- **HiddenArp** Indicates whether a program hides itself from showing up in ARP.
- **InstallDate** The date the application was installed (a best guess based on folder creation date heuristics).
- **InstallDateArpLastModified** The date of the registry ARP key for a given application. Hints at install date but not always accurate. Passed as an array. Example: 4/11/2015 00:00:00
- **InstallDateFromLinkFile** The estimated date of install based on the links to the files. Passed as an array.
- **InstallDateMsi** The install date if the application was installed via MSI. Passed as an array.
- **InventoryVersion** The version of the inventory file generating the events.
- **Language** The language code of the program.
- **MsiPackageCode** A GUID that describes the MSI Package. Multiple 'Products' (apps) can make up an MsiPackage.
- **MsiProductCode** A GUID that describe the MSI Product.
- **Name** The name of the application
- **OSVersionAtInstallTime** The four octets from the OS version at the time of the application's install.
- **PackageFullName** The package full name for a Store application.
- **ProgramInstanceId** A hash of the file IDs in an app.
- **Publisher** The Publisher of the application. Location pulled from depends on the 'Source' field.
- **RootDirPath** The path to the root directory where the program was installed.
- **Source** How the program was installed (ARP, MSI, Appx, etc...)
- **StoreAppType** A sub-classification for the type of Windows Store app, such as UWP or Win8StoreApp.
- **Type** "One of ("Application", "Hotfix", "BOE", "Service", "Unknown"). Application indicates Win32 or Appx app, Hotfix indicates app updates (KBs), BOE indicates it's an app with no ARP or MSI entry, Service indicates that it is a service. Application and BOE are the ones most likely seen."
- **Version** The version number of the program.

Microsoft.Windows.Inventory.Core.InventoryApplicationRemove

This event indicates that a new set of InventoryDevicePnpAdd events will be sent.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Core.InventoryApplicationStartSync

This event indicates that a new set of InventoryApplicationAdd events will be sent.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Core.InventoryDeviceContainerRemove

This event indicates that the InventoryDeviceContainer object is no longer present.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Core.InventoryDriverPackageAdd

This event sends basic metadata about drive packages installed on the system to help keep Windows up-to-date.

The following fields are available:

- **Class** The class name for the device driver.
- **ClassGuid** The class GUID for the device driver.
- **Date** The driver package date.
- **Directory** The path to the driver package.
- **DriverInBox** Is the driver included with the operating system?
- **Inf** The INF name of the driver package.
- **InventoryVersion** The version of the inventory file generating the events.
- **Provider** The provider for the driver package.
- **SubmissionId** The HLK submission ID for the driver package.
- **Version** The version of the driver package.

Microsoft.Windows.Inventory.Core.InventoryDriverBinaryStartSync

This event indicates that a new set of InventoryDriverBinaryAdd events will be sent.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Core.InventoryDriverBinaryRemove

This event indicates that the InventoryDriverBinary object is no longer present.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Core.InventoryDriverPackageRemove

This event indicates that the InventoryDriverPackageRemove object is no longer present.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Core.InventoryDevicePnpRemove

This event indicates that the InventoryDevicePnpRemove object is no longer present.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Core.InventoryDeviceContainerAdd

This event sends basic metadata about a device container (such as a monitor or printer as opposed to a PNP device) to help keep Windows up-to-date.

The following fields are available:

- **Categories** A comma separated list of functional categories in which the container belongs.
- **DiscoveryMethod** The discovery method for the device container.
- **FriendlyName** The name of the device container.
- **InventoryVersion** The version of the inventory file generating the events.
- **IsActive** Is the device connected, or has it been seen in the last 14 days?

- **IsConnected** For a physically attached device, this value is the same as IsPresent. For wireless a device, this value represents a communication link.
- **IsMachineContainer** Is the container the root device itself?
- **IsNetworked** Is this a networked device?
- **IsPaired** Does the device container require pairing?
- **Manufacturer** The manufacturer name for the device container.
- **ModelId** A model GUID.
- **ModelName** The model name.
- **ModelNumber** The model number for the device container.
- **PrimaryCategory** The primary category for the device container.

Microsoft.Windows.Inventory.Core.InventoryDeviceContainerStartSync

This event indicates that a new set of InventoryDeviceContainerAdd events will be sent.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Core.InventoryDeviceMediaClassStartSync

This event indicates that a new set of InventoryDeviceMediaClassSAdd events will be sent.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Core.InventoryDriverPackageStartSync

This event indicates that a new set of InventoryDriverPackageAdd events will be sent.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Core.InventoryDeviceMediaClassRemove

This event indicates that the InventoryDeviceMediaClassRemove object is no longer present.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Core.InventoryDevicePnpStartSync

This event indicates that a new set of InventoryDevicePnpAdd events will be sent.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Core.InventoryDeviceMediaClassAdd

This event sends additional metadata about a PNP device that is specific to a particular class of devices to help keep Windows up to date while reducing overall size of data payload.

The following fields are available:

- **Audio_CaptureDriver** The Audio device capture driver endpoint.
- **Audio_RenderDriver** The Audio device render driver endpoint.
- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Core.InventoryDevicePnpAdd

This event represents the basic metadata about a PNP device and its associated driver

The following fields are available:

- **class** The device setup class of the driver loaded for the device
- **classGuid** The device class GUID from the driver package
- **COMPID** A JSON array that provides the value and order of the compatible ID tree for the device.
- **ContainerId** A system-supplied GUID that uniquely groups the functional devices associated with a single-function or multifunction device installed in the device.
- **description** The device description
- **deviceState** DeviceState is a bitmask of the following: DEVICE_IS_CONNECTED 0x0001 (currently only for container). DEVICE_IS_NETWORK_DEVICE 0x0002 (currently only for container). DEVICE_IS_PAIED 0x0004 (currently only for container). DEVICE_IS_ACTIVE 0x0008 (currently never set). DEVICE_IS_MACHINE 0x0010 (currently only for container). DEVICE_IS_PRESENT 0x0020 (currently always set). DEVICE_IS_HIDDEN 0x0040. DEVICE_IS_PRINTER 0x0080 (currently only for container). DEVICE_IS_WIRELESS 0x0100. DEVICE_IS_WIRELESS_FAT 0x0200. The most common values are therefore: 32 (0x20)= device is present. 96 (0x60)= device is present but hidden. 288 (0x120)= device is a wireless device that is present
- **DriverId** A unique identifier for the installed device.
- **DriverName** The name of the driver image file.
- **driverPackageStrongName** The immediate parent directory name in the Directory field of InventoryDriverPackage.
- **driverVerDate** The date of the driver loaded for the device
- **driverVerVersion** The version of the driver loaded for the device
- **enumerator** The bus that enumerated the device
- **HWID** A JSON array that provides the value and order of the HWID tree for the device.
- **Inf** The INF file name.
- **installState** The device installation state. One of these values: <https://msdn.microsoft.com/en-us/library/windows/hardware/ff543130.aspx>
- **InventoryVersion** The version of the inventory file generating the events.
- **lowerClassFilters** Lower filter class drivers IDs installed for the device.
- **lowerFilters** Lower filter drivers IDs installed for the device
- **manufacturer** The device manufacturer
- **matchingID** Represents the hardware ID or compatible ID that Windows uses to install a device instance
- **model** The device model
- **parentId** Device instance id of the parent of the device
- **ProblemCode** The current error code for the device.
- **provider** The device provider
- **service** The device service name#N##N##N##N##N#
- **STACKID** A JSON array that provides the value and order of the STACKID tree for the device.
- **upperClassFilters** Upper filter class drivers IDs installed for the device
- **upperFilters** Upper filter drivers IDs installed for the device

Microsoft.Windows.Inventory.Core.InventoryDriverBinaryAdd

This event provides the basic metadata about driver binaries running on the system

The following fields are available:

- **DriverCheckSum** The checksum of the driver file.
- **DriverCompany** The company name that developed the driver.
- **driverInBox** Is the driver included with the operating system?
- **driverIsKernelMode** Is it a kernel mode driver?
- **DriverName** The file name of the driver.

- **driverPackageStrongName** The strong name of the driver package
- **driverSigned** The strong name of the driver package
- **DriverTimeStamp** The low 32 bits of the time stamp of the driver file.
- **DriverType** A bitfield of driver attributes: 1. define DRIVER_MAP_DRIVER_TYPE_PRINTER 0x0001. 2. define DRIVER_MAP_DRIVER_TYPE_KERNEL 0x0002. 3. define DRIVER_MAP_DRIVER_TYPE_USER 0x0004. 4. define DRIVER_MAP_DRIVER_IS_SIGNED 0x0008. 5. define DRIVER_MAP_DRIVER_IS_INBOX 0x0010. 6. define DRIVER_MAP_DRIVER_IS_WINQUAL 0x0040. 7. define DRIVER_MAP_DRIVER_IS_SELF_SIGNED 0x0020. 8. define DRIVER_MAP_DRIVER_IS_CI_SIGNED 0x0080. 9. define DRIVER_MAP_DRIVER_HAS_BOOT_SERVICE 0x0100. 10. define DRIVER_MAP_DRIVER_TYPE_I386 0x10000. 11. define DRIVER_MAP_DRIVER_TYPE_IA64 0x20000. 12. define DRIVER_MAP_DRIVER_TYPE_AMD64 0x40000. 13. define DRIVER_MAP_DRIVER_TYPE_ARM 0x100000. 14. define DRIVER_MAP_DRIVER_TYPE_THUMB 0x200000. 15. define DRIVER_MAP_DRIVER_TYPE_ARMNT 0x400000. 16. define DRIVER_MAP_DRIVER_IS_TIME_STAMPED 0x800000.
- **DriverVersion** The version of the driver file.
- **ImageSize** The size of the driver file.
- **Inf** The name of the INF file.
- **InventoryVersion** The version of the inventory file generating the events.
- **Product** The product name that is included in the driver file.
- **ProductVersion** The product version that is included in the driver file.
- **service** The device service name
- **WdfVersion** The Windows Driver Framework version.

Microsoft.Windows.Inventory.Indicators.InventoryMiscellaneousUexIndicator

This event sends value data about the markers on custom devices, to help keep Windows up to date. The formal name for markers is UEX Indicators. See marker list for definitions.

The following fields are available:

- **IndicatorValue** Value of the marker/indicator
- **Key** Name of the marker/indicator

Microsoft.Windows.Inventory.Core.AmiTelCacheVersions

This event sends inventory component versions for the Device Inventory data.

The following fields are available:

- **aeinv** The version of the App inventory component.
- **devinv** The file version of the Device inventory component.

Microsoft.Windows.Inventory.Core.AmiTelCacheChecksum

This event captures basic checksum data about the device inventory items stored in the cache for use in validating data completeness for Microsoft.Windows.Inventory.Core events. The fields in this event may change over time, but they will always represent a count of a given object.

The following fields are available:

- **Device** A count of device objects in cache
- **DeviceCensus** A count of devicemetadata objects in cache
- **DriverPackageExtended** A count of driverpackageextended objects in cache
- **File** A count of file objects in cache
- **FileSigningInfo** A count of file signing info objects in cache.
- **Generic** A count of generic objects in cache
- **HwItem** A count of hwitem objects in cache

- **InventoryApplication** A count of application objects in cache
- **InventoryApplicationFile** A count of application file objects in cache
- **InventoryDeviceContainer** A count of device container objects in cache
- **InventoryDeviceInterface** A count of inventory device interface objects in cache.
- **InventoryDeviceMediaClass** A count of device media objects in cache
- **InventoryDevicePnp** A count of devicepnp objects in cache
- **InventoryDriverBinary** A count of driver binary objects in cache
- **InventoryDriverPackage** A count of device objects in cache
- **Metadata** A count of metadata objects in cache
- **Orphan** A count of orphan file objects in cache
- **Programs** A count of program objects in cache

Microsoft.Windows.Inventory.Core.InventoryDeviceInterfaceStartSync

This event indicates that a new set of InventoryDeviceInterfaceAdd events will be sent.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Core.InventoryDeviceInterfaceAdd

This event retrieves information about what sensor interfaces are available on the device.

The following fields are available:

- **Accelerometer3D** Indicates if an Accelerator3D sensor is found.
- **ActivityDetection** Indicates if an Activity Detection sensor is found.
- **AmbientLight** Indicates if an Ambient Light sensor is found.
- **Barometer** Indicates if a Barometer sensor is found.
- **Custom** Indicates if a Custom sensor is found.
- **EnergyMeter** Indicates if an Energy sensor is found.
- **FloorElevation** Indicates if a Floor Elevation sensor is found.
- **GeomagneticOrientation** Indicates if a Geo Magnetic Orientation sensor is found.
- **GravityVector** Indicates if a Gravity Detector sensor is found.
- **Gyrometer3D** Indicates if a Gyrometer3D sensor is found.
- **Humidity** Indicates if a Humidity sensor is found.
- **InventoryVersion** The version of the inventory file generating the events.
- **LinearAccelerometer** Indicates if a Linear Accelerometer sensor is found.
- **Magnetometer3D** Indicates if a Magnetometer3D sensor is found.
- **Orientation** Indicates if an Orientation sensor is found.
- **Pedometer** Indicates if a Pedometer sensor is found.
- **Proximity** Indicates if a Proximity sensor is found.
- **RelativeOrientation** Indicates if a Relative Orientation sensor is found.
- **SimpleDeviceOrientation** Indicates if a Simple Device Orientation sensor is found.
- **Temperature** Indicates if a Temperature sensor is found.

Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeAddInStartSync

This event indicates that a new sync is being generated for this object type.

There are no fields in this event.

Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeVBARuleViolationsStartSync

This event indicates that a new sync is being generated for this object type.

There are no fields in this event.

Microsoft.Windows.Inventory.General.InventoryMiscellaneousOfficeVBASync

This event indicates that a new sync is being generated for this object type.

There are no fields in this event.

Microsoft.Windows.Inventory.Indicators.InventoryMiscellaneousUexIndicatorRemove

This event is a counterpart to InventoryMiscellaneousUexIndicatorAdd that indicates that the item has been removed.

There are no fields in this event.

Microsoft.Windows.Inventory.Indicators.InventoryMiscellaneousUexIndicatorStartSync

This event indicates that a new set of InventoryMiscellaneousUexIndicatorAdd events will be sent.

There are no fields in this event.

OneDrive events

Microsoft.OneDrive.Sync.Updater.OfficeRegistration

This event determines the status of the OneDrive integration with Microsoft Office.

The following fields are available:

- **isValid** Is the Microsoft Office registration valid?

Microsoft.OneDrive.Sync.Updater.UpdateTierReg

This event determines status of the update tier registry values.

The following fields are available:

- **regReadEnterpriseHr** The HResult of the enterprise reg read value.
- **regReadTeamHr** The HResult of the team reg read value.

Microsoft.OneDrive.Sync.Updater.RepairResult

The event determines the result of the installation repair.

The following fields are available:

- **hr** The HResult of the operation.

Microsoft.OneDrive.Sync.Updater.UpdateXmlDownloadHResult

This event determines the status when downloading the OneDrive update configuration file.

The following fields are available:

- **hr** The HResult of the operation.

Microsoft.OneDrive.Sync.Updater.SetupBinaryDownloadHResult

This event indicates the status when downloading the OneDrive setup file.

The following fields are available:

- **hr** The HResult of the operation.

Microsoft.OneDrive.Sync.Updater.UpdateOverallResult

This event determines the outcome of the operation.

The following fields are available:

- **hr** The HResult of the operation.
- **IsLoggingEnabled** Is logging enabled?
- **UpdaterVersion** The version of the updater.

Microsoft.OneDrive.Sync.Updater.WebConnectionStatus

This event determines the error code that was returned when verifying Internet connectivity.

The following fields are available:

- **winInetError** The HResult of the operation.

Microsoft.OneDrive.Sync.Updater.OverlayIconStatus

This event indicates if the OneDrive overlay icon is working correctly. 0 = healthy; 1 = can be fixed; 2 = broken

The following fields are available:

- **32bit** The status of the OneDrive overlay icon on a 32-bit operating system.
- **64bit** The status of the OneDrive overlay icon on a 64-bit operating system.
- **SixtyFourBit** The status of the OneDrive overlay icon on a 32-bit operating system.
- **ThirtyTwoBit** The status of the OneDrive overlay icon on a 64-bit operating system.

Microsoft.OneDrive.Sync.Updater.ComponentInstallState

This event determines the installation state of dependent OneDrive components.

The following fields are available:

- **ComponentName** The name of the dependent component.
- **isInstalled** Is the dependent component installed?

Microsoft.OneDrive.Sync.Updater.CommonData

This event contains basic OneDrive configuration data that helps to diagnose failures.

The following fields are available:

- **AppVersion** The version of the app.
- **BuildArch** Is the architecture x86 or x64?
- **Environment** Is the device on the production or int service?
- **IsMSFTInternal** Is this an internal Microsoft device?
- **MachineGuid** The CEIP machine ID.
- **Market** Which market is this in?
- **OfficeVersion** The version of Office that is installed.
- **OneDriveDeviceId** The OneDrive device ID.
- **OSDeviceName** Only if the device is internal to Microsoft, the device name.
- **OSUserName** Only if the device is internal to Microsoft, the user name.
- **UserGuid** A unique global user identifier.

Microsoft.OneDrive.Sync.Setup.APIOperation

This event includes basic data about install and uninstall OneDrive API operations.

The following fields are available:

- **APIName** The name of the API.
- **Duration** How long the operation took.
- **IsSuccess** Was the operation successful?
- **ResultCode** The result code.

- **ScenarioName** The name of the scenario.

Microsoft.OneDrive.Sync.Setup.RegisterStandaloneUpdaterAPIOperation

This event is related to registering or unregistering the OneDrive update task.

The following fields are available:

- **APIName** The name of the API.
- **IsSuccess** Was the operation successful?
- **RegisterNewTaskResult** The HResult of the RegisterNewTask operation.
- **ScenarioName** The name of the scenario.
- **UnregisterOldTaskResult** The HResult of the UnregisterOldTask operation.

Microsoft.OneDrive.Sync.Setup.EndExperience

This event includes a success or failure summary of the installation.

The following fields are available:

- **APIName** The name of the API.
- **HResult** Indicates the result code of the event
- **IsSuccess** Was the operation successful?
- **ScenarioName** The name of the scenario.

Microsoft.OneDrive.Sync.Setup.OSUpgradeInstallationOperation

This event is related to the OS version when the OS is upgraded with OneDrive installed.

The following fields are available:

- **CurrentOneDriveVersion** The current version of OneDrive.
- **CurrentOSBuildBranch** The current branch of the operating system.
- **CurrentOSBuildNumber** The current build number of the operating system.
- **CurrentOSVersion** The current version of the operating system.
- **HResult** The HResult of the operation.
- **SourceOSBuildBranch** The source branch of the operating system.
- **SourceOSBuildNumber** The source build number of the operating system.
- **SourceOSVersion** The source version of the operating system.

Microsoft.OneDrive.Sync.Setup.SetupCommonData

This event contains basic OneDrive configuration data that helps to diagnose failures.

The following fields are available:

- **AppVersion** The version of the app.
- **BuildArchitecture** Is the architecture x86 or x64?
- **Environment** Is the device on the production or int service?
- **MachineGuid** The CEIP machine ID.
- **Market** Which market is this in?
- **MSFTInternal** Is this an internal Microsoft device?
- **OfficeVersionString** The version of Office that is installed.
- **OSDeviceName** Only if the device is internal to Microsoft, the device name.
- **OSUserName** Only if the device is internal to Microsoft, the user name.
- **UserGuid** The CEIP user ID.

Setup events

SetupPlatformTel.SetupPlatformTelActivityStarted

"This event sends basic metadata about the update installation process generated by SetupPlatform to help keep Windows up to date."

The following fields are available:

- **Name** The name of the dynamic update type. Example: GDR driver

SetupPlatformTel.SetupPlatformTelActivityEvent

This event sends basic metadata about the SetupPlatform update installation process, to help keep Windows up-to-date

The following fields are available:

- **ActivityId** Provides a unique Id to correlate events that occur between a activity start event, and a stop event
- **ActivityName** Provides a friendly name of the package type that belongs to the ActivityId (Setup, LanguagePack, GDR, Driver, etc.)
- **FieldName** Retrieves the event name/data point. Examples: InstallStartTime, InstallEndtime, OverallResult etc.
- **GroupName** Retrieves the groupname the event belongs to. Example: Install Information, DU Information, Disk Space Information etc.
- **value** Value associated with the corresponding event name. For example, time-related events will include the system time

SetupPlatformTel.SetupPlatformTelEvent

This service retrieves events generated by SetupPlatform, the engine that drives the various deployment scenarios.

The following fields are available:

- **FieldName** Retrieves the event name/data point. Examples: InstallStartTime, InstallEndtime, OverallResult etc.
- **Value** Retrieves the value associated with the corresponding event name (Field Name). For example: For time related events this will include the system time.
- **GroupName** Retrieves the groupname the event belongs to. Example: Install Information, DU Information, Disk Space Information etc.

Shared PC events

Microsoft.Windows.SharedPC.AccountManager.DeleteUserAccount

Activity for deletion of a user account for devices set up for Shared PC mode as part of the Transient Account Manager to help keep Windows up to date. Deleting unused user accounts on shared devices frees up disk space to improve Windows Update success rates.

The following fields are available:

- **accountType** The type of account that was deleted. Example: AD, AAD, or Local
- **userSid** The security identifier of the account.
- **wilActivity** Windows Error Reporting data collected when there is a failure in deleting a user account with the Transient Account Manager.

Microsoft.Windows.SharedPC.AccountManager.SinglePolicyEvaluation

Activity for run of the Transient Account Manager that determines if any user accounts should be deleted for devices set up for Shared PC mode to help keep Windows up to date. Deleting unused user accounts on shared devices frees up disk space to improve Windows Update success rates

The following fields are available:

- **wilActivity** Windows Error Reporting data collected when there is a failure in evaluating accounts to be deleted with the Transient Account Manager.
- **totalAccountCount** The number of accounts on a device after running the Transient Account Manager policies.
- **evaluationTrigger** When was the Transient Account Manager policies ran? Example: At log off or during maintenance hours

Software update events

SoftwareUpdateClientTelemetry.UpdateDetected

This event sends data about an AppX app that has been updated from the Windows Store, including what app needs an update and what version/architecture is required, in order to understand and address problems with apps getting required updates.

The following fields are available:

- **ApplicableUpdateInfo** Metadata for the updates which were detected as applicable
- **CallerApplicationName** The name provided by the caller who initiated API calls into the software distribution client
- **NumberOfApplicableUpdates** The number of updates which were ultimately deemed applicable to the system after the detection process is complete
- **RelatedCV** The previous Correlation Vector that was used before swapping with a new one
- **WUDeviceID** The unique device ID controlled by the software distribution client
- **IntentPFNs** Intended application-set metadata for atomic update scenarios.
- **ServiceGuid** An ID which represents which service the software distribution client is connecting to (Windows Update, Windows Store, etc.)

SoftwareUpdateClientTelemetry.SLSDiscovery

This event sends data about the ability of Windows to discover the location of a backend server with which it must connect to perform updates or content acquisition, in order to determine disruptions in availability of update services and provide context for Windows Update errors.

The following fields are available:

- **EventScenario** Indicates the purpose of sending this event - whether because the software distribution just started checking for content, or whether it was cancelled, succeeded, or failed
- **HResult** Indicates the result code of the event (success, cancellation, failure code HResult)
- **IsBackground** Indicates whether the SLS discovery event took place in the foreground or background
- **NextExpirationTime** Indicates when the SLS cab expires
- **ServiceID** An ID which represents which service the software distribution client is connecting to (Windows Update, Windows Store, etc.)
- **SusClientId** The unique device ID controlled by the software distribution client
- **UrlPath** Path to the SLS cab that was downloaded
- **WUAVersion** The version number of the software distribution client

SoftwareUpdateClientTelemetry.Commit

This event sends data on whether the Update Service has been called to execute an upgrade, to help keep Windows up to date.

The following fields are available:

- **BiosFamily** The family of the BIOS (Basic Input Output System).

- **BiosName** The name of the device BIOS.
- **BiosReleaseDate** The release date of the device BIOS.
- **BiosSKUNumber** The sku number of the device BIOS.
- **BIOSVendor** The vendor of the BIOS.
- **BiosVersion** The version of the BIOS.
- **BundleId** Identifier associated with the specific content bundle; should not be all zeros if the bundleID was found.
- **ClientVersion** The version number of the software distribution client.
- **DeviceModel** What is the device model.
- **EventInstanceId** A globally unique identifier for event instance.
- **EventScenario** State of call
- **EventType** "Possible values are ""Child"", ""Bundle"" or ""Driver""."
- **HandlerType** Indicates the kind of content (app, driver, windows patch, etc.)
- **RevisionNumber** Unique revision number of Update
- **ServerId** Identifier for the service to which the software distribution client is connecting, such as Windows Update and Windows Store.
- **SystemBIOSMajorRelease** Major version of the BIOS.
- **SystemBIOSMinorRelease** Minor version of the BIOS.
- **UpdateId** Unique Update ID
- **WUDeviceID** UniqueDeviceID
- **BundleRevisionNumber** Identifies the revision number of the content bundle
- **FlightId** The specific id of the flight the device is getting
- **CallerApplicationName** The name provided by the caller who initiated API calls into the software distribution client

SoftwareUpdateClientTelemetry.DownloadCheckpoint

This event provides a checkpoint between each of the Windows Update download phases for UUP content

The following fields are available:

- **EventScenario** Indicates the purpose of sending this event - whether because the software distribution just started checking for content, or whether it was cancelled, succeeded, or failed
- **ExtendedStatusCode** Secondary error code for certain scenarios where StatusCode wasn't specific enough
- **FileId** A hash that uniquely identifies a file
- **FileName** Name of the downloaded file
- **RelatedCV** The previous Correlation Vector that was used before swapping with a new one
- **StatusCode** Indicates the result of a CheckForUpdates event (success, cancellation, failure code HResult)
- **EventType** "Possible values are ""Child"", ""Bundle"" , ""Relase"" or ""Driver"""
- **CallerApplicationName** The name provided by the caller who initiated API calls into the software distribution client
- **ClientVersion** The version number of the software distribution client
- **FlightId** The unique identifier for each flight
- **RevisionNumber** Unique revision number of Update
- **ServiceGuid** An ID which represents which service the software distribution client is checking for content (Windows Update, Microsoft Store, etc.)
- **UpdateId** Unique Update ID
- **WUDeviceID** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue

SoftwareUpdateClientTelemetry.UpdateMetadataIntegrity

This event identifies whether updates have been tampered with and protects against man-in-the-middle attacks.

The following fields are available:

- **EventScenario** The purpose of this event, such as scan started, scan succeeded, or scan failed.
- **ExtendedStatusCode** The secondary status code of the event.
- **LeafCertId** Integral ID from the FragmentSigning data for certificate that failed.
- **MetadataIntegrityMode** The mode of the transport metadata integrity check. 0 = unknown; 1 = ignore; 2 = audit; 3 = enforce
- **MetadataSignature** A base64-encoded string of the signature associated with the update metadata (specified by revision ID).
- **RevisionId** The revision ID for a specific piece of content.
- **RevisionNumber** The revision number for a specific piece of content.
- **ServiceGuid** Identifies the service to which the software distribution client is connected, Example: Windows Update or Windows Store
- **SHA256OfLeafCertPublicKey** A base64 encoding of the hash of the Base64CertData in the FragmentSigning data of the leaf certificate.
- **SHA256OfTimestampToken** A base64-encoded string of hash of the timestamp token blob.
- **SignatureAlgorithm** The hash algorithm for the metadata signature.
- **StatusCode** The status code of the event.
- **TimestampTokenId** The time this was created. It is encoded in a timestamp blob and will be zero if the token is malformed.
- **UpdateId** The update ID for a specific piece of content.
- **TimestampTokenCertThumbprint** "The thumbprint of the encoded timestamp token. "
- **ValidityWindowInDays** The validity window that's in effect when verifying the timestamp.
- **ListOfSHA256OfIntermediateCerData** A semicolon delimited list of base64 encoding of hashes for the Base64CerData in the FragmentSigning data of an intermediate certificate.
- **RawMode** The raw unparsed mode string from the SLS response. This field is null if not applicable.
- **RawValidityWindowInDays** The raw unparsed validity window string in days of the timestamp token. This field is null if not applicable.
- **SHA256OfLeafCerData** A base64 encoding of the hash for the Base64CerData in the FragmentSigning data of the leaf certificate.
- **EndpointUrl** The endpoint URL where the device obtains update metadata. This is used to distinguish between test, staging, and production environments.
- **SLSPrograms** A test program to which a device may have opted in. Example: Insider Fast

SoftwareUpdateClientTelemetry.Download

This event sends tracking data about the software distribution client download of the content for that update, to help keep Windows up to date.

The following fields are available:

- **ActiveDownloadTime** How long the download took, in seconds, excluding time where the update wasn't actively being downloaded.
- **AppXBlockHashValidationFailureCount** A count of the number of blocks that have failed validation after being downloaded.
- **AppXDownloadScope** Indicates the scope of the download for application content. For streaming install scenarios, AllContent - non-streaming download, RequiredOnly - streaming download requested content required for launch, AutomaticOnly - streaming download requested automatic streams for the app, and Unknown - for events sent before download scope is determined by the Windows Update client.
- **BiosFamily** The family of the BIOS (Basic Input Output System).

- **BiosName** The name of the device BIOS.
- **BiosReleaseDate** The release date of the device BIOS.
- **BiosSKUNumber** The sku number of the device BIOS.
- **BIOSVendor** The vendor of the BIOS.
- **BiosVersion** The version of the BIOS.
- **BundleBytesDownloaded** How many bytes were downloaded for the specific content bundle.
- **BundleId** Identifier associated with the specific content bundle; should not be all zeros if the bundleID was found.
- **BundleRepeatFailFlag** Indicates whether this particular update bundle had previously failed to download.
- **BundleRevisionNumber** Identifies the revision number of the content bundle.
- **BytesDownloaded** How many bytes were downloaded for an individual piece of content (not the entire bundle).
- **CachedEngineVersion** For self-initiated healing, the version of the SIH engine that is cached on the device. If the SIH engine does not exist, the value is null.
- **CallerApplicationName** The name provided by the caller who initiated API calls into the software distribution client.
- **CbsDownloadMethod** Indicates whether the download was a full-file download or a partial/delta download.
- **CDNCountryCode** Two letter country abbreviation for the CDN's location.
- **CDNId** ID which defines which CDN the software distribution client downloaded the content from.
- **ClientManagedByWSUSServer** Indicates whether the client is managed by Windows Server Update Services (WSUS).
- **ClientVersion** The version number of the software distribution client.
- **CurrentMobileOperator** The mobile operator the device is currently connected to.
- **DeviceModel** What is the device model.
- **DeviceOEM** What OEM does this device belong to.
- **DownloadPriority** Indicates whether a download happened at background, normal, or foreground priority.
- **DownloadScenarioId** A unique ID for a given download used to tie together WU and DO events.
- **DownloadType** Differentiates the download type of SIH downloads between Metadata and Payload downloads.
- **Edition** Indicates the edition of Windows being used.
- **EventInstanceId** A globally unique identifier for event instance.
- **EventNamespaceID** Indicates whether the event succeeded or failed. Has the format EventType+Event where Event is Succeeded, Cancelled, Failed, etc.
- **EventScenario** Indicates the purpose of sending this event - whether because the software distribution just started downloading content, or whether it was cancelled, succeeded, or failed.
- **EventType** Possible values are Child, Bundle, or Driver.
- **ExtendedStatusCode** Secondary error code for certain scenarios where StatusCode wasn't specific enough.
- **FeatureUpdatePause** Indicates whether feature OS updates are paused on the device.
- **FlightBranch** The branch that a device is on if participating in flighting (pre-release builds).
- **FlightBuildNumber** If this download was for a flight (pre-release build), this indicates the build number of that flight.
- **FlightId** The specific id of the flight (pre-release build) the device is getting.
- **FlightRing** The ring (speed of getting builds) that a device is on if participating in flighting (pre-release builds).
- **HandlerType** Indicates what kind of content is being downloaded (app, driver, windows patch, etc.).
- **HardwareId** If this download was for a driver targeted to a particular device model, this ID indicates the model of the device.
- **HomeMobileOperator** The mobile operator that the device was originally intended to work with.

- **HostName** The hostname URL the content is downloading from.
- **IPVersion** Indicates whether the download took place over IPv4 or IPv6.
- **IsAOACDevice** Is it Always On, Always Connected?
- **IsDependentSet** Indicates whether a driver is a part of a larger System Hardware/Firmware Update
- **IsWUfBDualScanEnabled** Indicates if Windows Update for Business dual scan is enabled on the device.
- **IsWUfBEnabled** Indicates if Windows Update for Business is enabled on the device.
- **NetworkCostBitMask** Indicates what kind of network the device is connected to (roaming, metered, over data cap, etc.)
- **NetworkRestrictionStatus** "More general version of NetworkCostBitMask, specifying whether Windows considered the current network to be ""metered."""
- **PackageFullName** The package name of the content.
- **PhonePreviewEnabled** Indicates whether a phone was opted-in to getting preview builds, prior to flighting (pre-release builds) being introduced.
- **PlatformRole** The PowerPlatformRole as defined on MSDN
- **ProcessName** The process name of the caller who initiated API calls, in the event where CallerApplicationName was not provided.
- **ProcessorArchitecture** Processor architecture of the system (x86, AMD64, ARM).
- **QualityUpdatePause** Indicates whether quality OS updates are paused on the device.
- **RelatedCV** The previous Correlation Vector that was used before swapping with a new one
- **RepeatFailFlag** Indicates whether this specific piece of content had previously failed to download.
- **RevisionNumber** Identifies the revision number of this specific piece of content.
- **ServiceGuid** An ID which represents which service the software distribution client is installing content for (Windows Update, Windows Store, etc.).
- **Setup360Phase** If the download is for an operating system upgrade, this datapoint indicates which phase of the upgrade is underway.
- **ShippingMobileOperator** The mobile operator that a device shipped on.
- **StatusCode** Indicates the result of a Download event (success, cancellation, failure code HResult).
- **SystemBIOSMajorRelease** Major version of the BIOS.
- **SystemBIOSMinorRelease** Minor version of the BIOS.
- **TargetGroupId** For drivers targeted to a specific device model, this ID indicates the distribution group of devices receiving that driver.
- **TargetingVersion** For drivers targeted to a specific device model, this is the version number of the drivers being distributed to the device.
- **TargetMetadataVersion** For self-initiated healing, this is the target version of the SIH engine to download (if needed). If not, the value is null.
- **ThrottlingServiceHRESULT** Result code (success/failure) while contacting a web service to determine whether this device should download content yet.
- **TimeToEstablishConnection** Time (in ms) it took to establish the connection prior to beginning downloaded.
- **TotalExpectedBytes** The total count of bytes that the download is expected to be.
- **UpdateId** An identifier associated with the specific piece of content.
- **UpdateImportance** Indicates whether a piece of content was marked as Important, Recommended, or Optional.
- **UsedDO** Whether the download used the delivery optimization service.
- **UsedSystemVolume** Indicates whether the content was downloaded to the device's main system storage drive, or an alternate storage drive.
- **WUDeviceID** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue.
- **WUSetting** Indicates the users' current updating settings.

SoftwareUpdateClientTelemetry.CheckForUpdates

This event sends tracking data about the software distribution client check for content that is applicable to a device, to help keep Windows up to date

The following fields are available:

- **ActivityMatchingId** Contains a unique ID identifying a single CheckForUpdates session from initialization to completion.
- **AllowCachedResults** Indicates if the scan allowed using cached results.
- **BiosFamily** The family of the BIOS (Basic Input Output System).
- **BiosName** The name of the device BIOS.
- **BiosReleaseDate** The release date of the device BIOS.
- **BiosSKUNumber** The sku number of the device BIOS.
- **BIOSVendor** The vendor of the BIOS.
- **BiosVersion** The version of the BIOS.
- **CallerApplicationName** The name provided by the caller who initiated API calls into the software distribution client.
- **CapabilityDetectoidGuid** The GUID for a hardware applicability detectoid that could not be evaluated.
- **CDNCountryCode** Two letter country abbreviation for the CDN's location.
- **CDNId** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue.
- **ClientVersion** The version number of the software distribution client.
- **CurrentMobileOperator** The mobile operator the device is currently connected to.
- **DeviceModel** What is the device model.
- **DriverError** The error code hit during a driver scan. This is 0 if no error was encountered.
- **EventInstanceId** A globally unique identifier for event instance.
- **EventScenario** Indicates the purpose of sending this event - whether because the software distribution just started checking for content, or whether it was cancelled, succeeded, or failed.
- **ExtendedMetadataCabUrl** Hostname that is used to download an update.
- **ExtendedStatusCode** Secondary error code for certain scenarios where StatusCode wasn't specific enough.
- **FailedUpdateGuids** The GUIDs for the updates that failed to be evaluated during the scan.
- **FailedUpdatesCount** The number of updates that failed to be evaluated during the scan.
- **FlightBranch** The branch that a device is on if participating in flighting (pre-release builds).
- **FlightRing** The ring (speed of getting builds) that a device is on if participating in flighting (pre-release builds).
- **HomeMobileOperator** The mobile operator that the device was originally intended to work with.
- **IPVersion** Indicates whether the download took place over IPv4 or IPv6
- **IsWUfBDualScanEnabled** Indicates if Windows Update for Business dual scan is enabled on the device.
- **IsWUfBEnabled** Indicates if Windows Update for Business is enabled on the device.
- **MetadataIntegrityMode** The mode of the update transport metadata integrity check. 0-Unknown, 1-Ignor, 2-Audit, 3-Enforce
- **MSIError** The last error that was encountered during a scan for updates.
- **NetworkConnectivityDetected** Indicates the type of network connectivity that was detected. 0 - IPv4, 1 - IPv6
- **NumberOfApplicationsCategoryScanEvaluated** The number of categories (apps) for which an app update scan checked
- **NumberOfLoop** The number of round trips the scan required
- **NumberOfNewUpdatesFromServiceSync** The number of updates which were seen for the first time in this scan
- **NumberOfUpdatesEvaluated** The total number of updates which were evaluated as a part of the scan

- **NumFailedMetadataSignatures** The number of metadata signatures checks which failed for new metadata synced down.
- **Online** Indicates if this was an online scan.
- **PhonePreviewEnabled** Indicates whether a phone was getting preview build, prior to flighting (pre-release builds) being introduced.
- **ProcessName** The process name of the caller who initiated API calls, in the event where CallerApplicationName was not provided.
- **RelatedCV** The previous Correlation Vector that was used before swapping with a new one
- **ScanDurationInSeconds** The number of seconds a scan took
- **ScanEnqueueTime** The number of seconds it took to initialize a scan
- **ServiceGuid** An ID which represents which service the software distribution client is checking for content (Windows Update, Windows Store, etc.).
- **ServiceUrl** The environment URL a device is configured to scan with
- **ShippingMobileOperator** The mobile operator that a device shipped on.
- **StatusCode** Indicates the result of a CheckForUpdates event (success, cancellation, failure code HResult).
- **SyncType** Describes the type of scan the event was
- **SystemBIOSMajorRelease** Major version of the BIOS.
- **SystemBIOSMinorRelease** Minor version of the BIOS.
- **TotalNumMetadataSignatures** The total number of metadata signatures checks done for new metadata that was synced down.
- **WUDeviceID** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue.
- **ApplicableUpdateInfo** Metadata for the updates which were detected as applicable
- **NumberOfApplicableUpdates** The number of updates which were ultimately deemed applicable to the system after the detection process is complete
- **WebServiceRetryMethods** Web service method requests that needed to be retried to complete operation.
- **BranchReadinessLevel** The servicing branch configured on the device.
- **DeferralPolicySources** Sources for any update deferral policies defined (GPO = 0x10, MDM = 0x100, Flight = 0x1000, UX = 0x10000).
- **DeferredUpdates** Update IDs which are currently being deferred until a later time
- **DriverExclusionPolicy** Indicates if the policy for not including drivers with Windows Update is enabled.
- **FeatureUpdateDeferral** The deferral period configured for feature OS updates on the device (in days).
- **FeatureUpdatePause** Indicates whether feature OS updates are paused on the device.
- **FeatureUpdatePausePeriod** The pause duration configured for feature OS updates on the device (in days).
- **QualityUpdateDeferral** The deferral period configured for quality OS updates on the device (in days).
- **QualityUpdatePause** Indicates whether quality OS updates are paused on the device.
- **QualityUpdatePausePeriod** The pause duration configured for quality OS updates on the device (in days).
- **IntentPFNs** Intended application-set metadata for atomic update scenarios.
- **PausedUpdates** A list of Updatelds which are currently being paused.
- **PauseFeatureUpdatesEndTime** If feature OS updates are paused on the device, this is the date and time for the end of the pause time window.
- **PauseFeatureUpdatesStartTime** If feature OS updates are paused on the device, this is the date and time for the beginning of the pause time window.
- **PauseQualityUpdatesEndTime** If quality OS updates are paused on the device, this is the date and time for the end of the pause time window.
- **PauseQualityUpdatesStartTime** If quality OS updates are paused on the device, this is the date and time for the beginning of the pause time window.
- **CachedEngineVersion** For self-initiated healing, the version of the SIH engine that is cached on the device. If

the SIH engine does not exist, the value is null.

- **TargetMetadataVersion** For self-initiated healing, this is the target version of the SIH engine to download (if needed). If not, the value is null.
- **Context** Gives context on where the error has occurred. Example: AutoEnable, GetSLSData, AddService, Misc, or Unknown
- **DriverSyncPassPerformed** Were drivers scanned this time?

SoftwareUpdateClientTelemetry.Install

This event sends tracking data about the software distribution client installation of the content for that update, to help keep Windows up to date.

The following fields are available:

- **BiosFamily** The family of the BIOS (Basic Input Output System).
- **BiosName** The name of the device BIOS.
- **BiosReleaseDate** The release date of the device BIOS.
- **BiosSKUNumber** The sku number of the device BIOS.
- **BIOSVendor** The vendor of the BIOS.
- **BiosVersion** The version of the BIOS.
- **BundleBytesDownloaded** How many bytes were downloaded for the specific content bundle?
- **BundleId** Identifier associated with the specific content bundle; should not be all zeros if the bundleID was found.
- **BundleRepeatFailFlag** Has this particular update bundle previously failed to install?
- **BundleRevisionNumber** Identifies the revision number of the content bundle.
- **CachedEngineVersion** For self-initiated healing, the version of the SIH engine that is cached on the device. If the SIH engine does not exist, the value is null.
- **CallerApplicationName** The name provided by the caller who initiated API calls into the software distribution client.
- **CbsDownloadMethod** Was the download a full download or a partial download?
- **ClientManagedByWSUSServer** Is the client managed by Windows Server Update Services (WSUS)?
- **ClientVersion** The version number of the software distribution client.
- **CSIErrortype** The stage of CBS installation where it failed.
- **CurrentMobileOperator** Mobile operator that device is currently connected to.
- **DeviceModel** What is the device model.
- **DeviceOEM** What OEM does this device belong to.
- **DownloadPriority** The priority of the download activity.
- **DownloadScenarioId** A unique ID for a given download used to tie together WU and DO events.
- **DriverPingBack** Contains information about the previous driver and system state.
- **Edition** Indicates the edition of Windows being used.
- **EventInstanceId** A globally unique identifier for event instance.
- **EventNamespaceId** Indicates whether the event succeeded or failed. Has the format EventType+Event where Event is Succeeded, Cancelled, Failed, etc.
- **EventScenario** Indicates the purpose of sending this event - whether because the software distribution just started installing content, or whether it was cancelled, succeeded, or failed.
- **EventType** Possible values are Child, Bundle, or Driver.
- **ExtendedErrorCode** The extended error code.
- **ExtendedStatusCode** Secondary error code for certain scenarios where StatusCode wasn't specific enough.
- **FeatureUpdatePause** Are feature OS updates paused on the device?
- **FlightBranch** The branch that a device is on if participating in the Windows Insider Program.

- **FlightBuildNumber** If this installation was for a Windows Insider build, this is the build number of that build.
- **FlightId** The specific ID of the Windows Insider build the device is getting.
- **FlightRing** The ring that a device is on if participating in the Windows Insider Program.
- **HandlerType** Indicates what kind of content is being installed. Example: app, driver, Windows update
- **HardwareId** If this install was for a driver targeted to a particular device model, this ID indicates the model of the device.
- **HomeMobileOperator** The mobile operator that the device was originally intended to work with.
- **IntentPFNs** Intended application-set metadata for atomic update scenarios.
- **IsAOACDevice** Is it Always On, Always Connected? (Mobile device usage model)
- **IsDependentSet** Is the driver part of a larger System Hardware/Firmware update?
- **IsFinalOutcomeEvent** Does this event signal the end of the update/upgrade process?
- **IsFirmware** Is this update a firmware update?
- **IsSuccessFailurePostReboot** Did it succeed and then fail after a restart?
- **IsWUfBDualScanEnabled** Is Windows Update for Business dual scan enabled on the device?
- **IsWUfBEnabled** Is Windows Update for Business enabled on the device?
- **MergedUpdate** Was the OS update and a BSP update merged for installation?
- **MsiAction** The stage of MSI installation where it failed.
- **MsiProductCode** The unique identifier of the MSI installer.
- **PackageFullName** The package name of the content being installed.
- **PhonePreviewEnabled** Indicates whether a phone was getting preview build, prior to flighting being introduced.
- **PlatformRole** The PowerPlatformRole as defined on MSDN.
- **ProcessName** The process name of the caller who initiated API calls, in the event where CallerApplicationName was not provided.
- **ProcessorArchitecture** Processor architecture of the system (x86, AMD64, ARM).
- **QualityUpdatePause** Are quality OS updates paused on the device?
- **RelatedCV** The previous Correlation Vector that was used before swapping with a new one
- **RepeatFailFlag** Indicates whether this specific piece of content had previously failed to install.
- **RepeatSuccessInstallFlag** Indicates whether this specific piece of content had previously installed successful, for example if another user had already installed it.
- **RevisionNumber** The revision number of this specific piece of content.
- **ServiceGuid** An ID which represents which service the software distribution client is installing content for (Windows Update, Windows Store, etc.).
- **Setup360Phase** If the install is for an operating system upgrade, indicates which phase of the upgrade is underway.
- **ShippingMobileOperator** The mobile operator that a device shipped on.
- **StatusCode** Indicates the result of an installation event (success, cancellation, failure code HResult).
- **SystemBIOSMajorRelease** Major version of the BIOS.
- **SystemBIOSMinorRelease** Minor version of the BIOS.
- **TargetGroupId** For drivers targeted to a specific device model, this ID indicates the distribution group of devices receiving that driver.
- **TargetingVersion** For drivers targeted to a specific device model, this is the version number of the drivers being distributed to the device.
- **TransactionCode** The ID which represents a given MSI installation
- **UpdateId** Unique update ID
- **UpdateImportance** Indicates whether a piece of content was marked as Important, Recommended, or Optional.

- **UsedSystemVolume** Indicates whether the content was downloaded and then installed from the device's main system storage drive, or an alternate storage drive.
- **WUDeviceID** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue.
- **WUSetting** Indicates the user's current updating settings.

SoftwareUpdateClientTelemetry.DownloadHeartbeat

This event allows tracking of ongoing downloads and contains data to explain the current state of the download

The following fields are available:

- **BundleID** Identifier associated with the specific content bundle. If this value is found, it shouldn't report as all zeros
- **BytesTotal** Total bytes to transfer for this content
- **BytesTransferred** Total bytes transferred for this content at the time of heartbeat
- **ConnectionStatus** Indicates the connectivity state of the device at the time of heartbeat
- **CurrentError** Last (transient) error encountered by the active download
- **DownloadFlags** Flags indicating if power state is ignored
- **DownloadState** Current state of the active download for this content (queued, suspended, or progressing)
- **IsNetworkMetered** "Indicates whether Windows considered the current network to be ?metered"""
- **MOAppDownloadLimit** Mobile operator cap on size of application downloads, if any
- **MOUpdateDownloadLimit** Mobile operator cap on size of operating system update downloads, if any
- **PowerState** Indicates the power state of the device at the time of heartbeat (DC, AC, Battery Saver, or Connected Standby)
- **RelatedCV** "The previous correlation vector that was used by the client, before swapping with a new one "
- **ResumeCount** Number of times this active download has resumed from a suspended state
- **ServiceID** "Identifier for the service to which the software distribution client is connecting (Windows Update, Windows Store, etc) "
- **SuspendCount** Number of times this active download has entered a suspended state
- **SuspendReason** Last reason for why this active download entered a suspended state
- **CallerApplicationName** Name provided by the caller who initiated API calls into the software distribution client
- **ClientVersion** The version number of the software distribution client
- **EventType** "Possible values are ""Child"", ""Bundle"" , or ""Driver"""
- **FlightId** The unique identifier for each flight
- **RevisionNumber** Identifies the revision number of this specific piece of content
- **ServiceGuid** Identifier for the service to which the software distribution client is connecting (Windows Update, Windows Store, etc)
- **UpdateId** "Identifier associated with the specific piece of content "
- **WUDeviceID** "Unique device id controlled by the software distribution client "

Update events

Update360Telemetry.UpdateAgentPostRebootResult

This event collects information for both Mobile and Desktop regarding the post reboot phase of the new UUP (Unified Update Platform) update scenario

The following fields are available:

- **ErrorCode** The error code returned for the current post reboot phase
- **FlightId** The unique identifier for each flight

- **ObjectId** Unique value for each Update Agent mode
- **RelatedCV** Correlation vector value generated from the latest USO scan
- **Result** Indicates the Hresult
- **Scenarioid** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **SessionId** Unique value for each Update Agent mode attempt
- **UpdateId** Unique ID for each update
- **PostRebootResult** Indicates the Hresult

Update360Telemetry.UpdateAgent_Initialize

This event sends data during the initialize phase of updating Windows.

The following fields are available:

- **ErrorCode** The error code returned for the current initialize phase.
- **FlightId** Unique ID for each flight.
- **FlightMetadata** Contains the FlightId and the build being flighted.
- **ObjectId** Unique value for each Update Agent mode.
- **RelatedCV** Correlation vector value generated from the latest USO scan.
- **Scenarioid** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **SessionData** Contains instructions to update agent for processing FODs and DUCIs (Null for other scenarios).
- **SessionId** Unique value for each Update Agent mode attempt .
- **UpdateId** Unique ID for each update.
- **Result** Result of the initialize phase of update. 0 = Succeeded, 1 = Failed, 2 = Cancelled, 3 = Blocked, 4 = BlockCancelled

Update360Telemetry.UpdateAgent_DownloadRequest

This event sends data during the download request phase of updating Windows.

The following fields are available:

- **ErrorCode** The error code returned for the current download request phase.
- **ObjectId** Unique value for each Update Agent mode.
- **PackageCountOptional** Number of optional packages requested.
- **PackageCountRequired** Number of required packages requested.
- **PackageCountTotal** Total number of packages needed.
- **RelatedCV** Correlation vector value generated from the latest USO scan.
- **Scenarioid** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **SessionId** Unique value for each Update Agent mode attempt.
- **PackageSizeCanonical** Size of canonical packages in bytes
- **PackageSizeDiff** Size of diff packages in bytes
- **PackageSizeExpress** Size of express packages in bytes
- **Result** Result of the download request phase of update.
- **FlightId** Unique ID for each flight.
- **UpdateId** Unique ID for each update.
- **PackageCountTotalCanonical** Total number of canonical packages.
- **PackageCountTotalDiff** Total number of diff packages.
- **PackageCountTotalExpress** Total number of express packages.

- **DeletedCorruptFiles** Indicates if UpdateAgent found any corrupt payload files and whether the payload was deleted.
- **RangeRequestState** Represents the state of the download range request.

Update360Telemetry.UpdateAgent_Install

This event sends data during the install phase of updating Windows.

The following fields are available:

- **ErrorCode** The error code returned for the current install phase.
- **ObjectId** Unique value for each Update Agent mode.
- **RelatedCV** Correlation vector value generated from the latest scan.
- **Scenarioid** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **SessionId** Unique value for each Update Agent mode attempt.
- **Result** "Result of the install phase of update. 0 = Succeeded 1 = Failed, 2 = Cancelled, 3 = Blocked, 4 = BlockCancelled "
- **FlightId** Unique ID for each flight.
- **UpdateId** Unique ID for each update.

Update360Telemetry.UpdateAgent_ModeStart

This event sends data for the start of each mode during the process of updating Windows.

The following fields are available:

- **Mode** Indicates that the Update Agent mode that has started. 1 = Initialize, 2 = DownloadRequest, 3 = Install, 4 = Commit
- **ObjectId** Unique value for each Update Agent mode.
- **RelatedCV** The correlation vector value generated from the latest scan.
- **Scenarioid** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **SessionId** Unique value for each Update Agent mode attempt.
- **FlightId** Unique ID for each flight.
- **UpdateId** Unique ID for each update.

Update360Telemetry.UpdateAgent_SetupBoxLaunch

This event sends data during the launching of the setup box when updating Windows.

The following fields are available:

- **ObjectId** Unique value for each Update Agent mode.
- **Quiet** Indicates whether setup is running in quiet mode. 0 = false 1 = true
- **RelatedCV** Correlation vector value generated from the latest scan.
- **Scenarioid** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **SessionId** Unique value for each Update Agent mode attempt.
- **FlightId** Unique ID for each flight.
- **UpdateId** Unique ID for each update.
- **SetupMode** Setup mode 1 = predownload, 2 = install, 3 = finalize
- **SandboxSize** The size of the sandbox folder on the device.

Update notification events

Microsoft.Windows.UpdateNotificationPipeline.JavascriptJavascriptCriticalGenericMessage

This event indicates that Javascript is reporting a schema and a set of values for critical telemetry

The following fields are available:

- **CampaignConfigVersion** Configuration version for the current campaign
- **CampaignID** Currently campaign that's running on UNP
- **ConfigCatalogVersion** Current catalog version of UNP
- **ContentVersion** Content version for the current campaign on UNP
- **CV** Correlation vector
- **DetectorVersion** Most recently run detector version for the current campaign on UNP
- **GlobalEventCounter** Client-side counter that indicates the event ordering sent by the user
- **key1** Interaction data for the UI
- **key10** Interaction data for the UI
- **key11** Interaction data for the UI
- **key12** Interaction data for the UI
- **key13** Interaction data for the UI
- **key14** Interaction data for the UI
- **key15** Interaction data for the UI
- **key16** Interaction data for the UI
- **key17** Interaction data for the UI
- **key18** Interaction data for the UI
- **key19** Interaction data for the UI
- **key2** Interaction data for the UI
- **key20** Interaction data for the UI
- **key21** Interaction data for the UI
- **key22** Interaction data for the UI
- **key23** Interaction data for the UI
- **key24** Interaction data for the UI
- **key25** Interaction data for the UI
- **key26** Interaction data for the UI
- **key27** Interaction data for the UI
- **key28** Interaction data for the UI
- **key29** Interaction data for the UI
- **key3** Interaction data for the UI
- **key30** Interaction data for the UI
- **key4** Interaction data for the UI
- **key5** Interaction data for the UI
- **key6** Interaction data for the UI
- **key7** Interaction data for the UI
- **key8** Interaction data for the UI
- **key9** Interaction data for the UI
- **PackageVersion** Current package version of UNP
- **schema** Type of UI interaction

Microsoft.Windows.UpdateNotificationPipeline.UNPCampaignHeartbeat

This event is sent at the start of each campaign, to be used as a heartbeat

The following fields are available:

- **CampaignConfigVersion** Configuration version for the current campaign
- **CampaignID** Currently campaign that's running on UNP
- **ConfigCatalogVersion** Current catalog version of UNP
- **ContentVersion** Content version for the current campaign on UNP
- **CV** Correlation vector
- **DetectorVersion** Most recently run detector version for the current campaign on UNP
- **GlobalEventCounter** Client-side counter that indicates the event ordering sent by the user
- **PackageVersion** Current UNP package version

Microsoft.Windows.UpdateNotificationPipeline.UNPCampaignManagerCleaningCampaign

This event indicates that the Campaign Manager is cleaning up the campaign content

The following fields are available:

- **CampaignConfigVersion** Configuration version for the current campaign
- **CampaignID** Current campaign that's running on UNP
- **ConfigCatalogVersion** Current catalog version of UNP
- **ContentVersion** Content version for the current campaign on UNP
- **CV** Correlation vector
- **DetectorVersion** Most recently run detector version for the current campaign on UNP
- **GlobalEventCounter** Client-side counter that indicates the event ordering sent by the user
- **PackageVersion** Current UNP package version

Microsoft.Windows.UpdateNotificationPipeline.UnpCampaignManagerGetIsCampaignCompleteFailed

This event is sent when a campaign completion status query fails

The following fields are available:

- **CampaignConfigVersion** Configuration version for the current campaign
- **CampaignID** Current campaign that's running on UNP
- **ConfigCatalogVersion** Current catalog version of UNP
- **ContentVersion** Content version for the current campaign on UNP
- **CV** Correlation vector
- **DetectorVersion** Most recently run detector version for the current campaign on UNP
- **GlobalEventCounter** Client-side counter that indicates the event ordering sent by the user
- **HRESULT** HRESULT of the failure
- **PackageVersion** Current UNP package version

Microsoft.Windows.UpdateNotificationPipeline.UNPCampaignManagerHeartbeat

This event is sent at the start of the CampaignManager event and is intended to be used as a heartbeat

The following fields are available:

- **CampaignConfigVersion** Configuration version for the current campaign
- **CampaignID** Currently campaign that's running on UNP
- **ConfigCatalogVersion** Current catalog version of UNP
- **ContentVersion** Content version for the current campaign on UNP
- **CV** Correlation vector
- **DetectorVersion** Most recently run detector version for the current campaign on UNP
- **GlobalEventCounter** Client-side counter that indicates the event ordering sent by the user
- **PackageVersion** Current UNP package version

Microsoft.Windows.UpdateNotificationPipeline.UnpCampaignManagerRunCampaignFailed

This event is sent when the Campaign Manager encounters an unexpected error while running the campaign

The following fields are available:

- **CampaignConfigVersion** Configuration version for the current campaign
- **CampaignID** Currently campaign that's running on UNP
- **ConfigCatalogVersion** Current catalog version of UNP
- **ContentVersion** Content version for the current campaign on UNP
- **CV** Correlation vector
- **DetectorVersion** Most recently run detector version for the current campaign on UNP
- **GlobalEventCounter** Client-side counter that indicates the event ordering sent by the user
- **HRESULT** HRESULT of the failure#N#
- **PackageVersion** Current UNP package version

Upgrade events

Setup360Telemetry.PreDownloadUX

The event sends data regarding OS updates and upgrades from Windows 7, Windows 8, and Windows 10. Specifically, the Setup360Telemetry.PredownloadUX indicates the outcome of the PredownloadUX portion of the update process.

The following fields are available:

- **ClientId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **HostOSBuildNumber** The build number of the previous operating system.
- **HostOsSkuName** The OS edition which is running the Setup360 instance (previous operating system).
- **InstanceId** Unique GUID that identifies each instance of setuphost.exe.
- **ReportId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, this is the GUID for the install.wim.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that can be used to diagnose errors.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **SetupVersionBuildNumber** The build number of Setup360 (build number of the target OS).
- **State** The exit state of the Setup360 run. Example: succeeded, failed, blocked, cancelled
- **TestId** A string to uniquely identify a group of events.
- **Wuid** Windows Update client ID.

Setup360Telemetry.UnexpectedEvent

This event sends data indicating that the device has invoked the unexpected event phase of the upgrade, to help keep Windows up to date.

The following fields are available:

- **ClientId** With Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **HostOSBuildNumber** The build number of the previous OS.
- **HostOsSkuName** The OS edition which is running Setup360 instance (previous OS).
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe

- **ReportId** With Windows Update, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that can be used to diagnose errors.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **TestId** A string to uniquely identify a group of events.
- **Wuid** This is the Windows Update Client ID. With Windows Update, this is the same as the clientID.

Setup360Telemetry.PreInstallQuiet

This event sends data indicating that the device has invoked the preinstall quiet phase of the upgrade, to help keep Windows up to date.

The following fields are available:

- **ClientId** With Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **HostOSBuildNumber** The build number of the previous OS.
- **HostOsSkuName** The OS edition which is running Setup360 instance (previous OS).
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe
- **ReportId** With Windows Update, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback etc.
- **Setup360Result** The result of Setup360. This is an HRESULT error code that can be used to diagnose errors.
- **Setup360Scenario** Setup360 flow type (Boot, Media, Update, MCT)
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **TestId** A string to uniquely identify a group of events.
- **Wuid** This is the Windows Update Client ID. With Windows Update, this is the same as the clientID.

Setup360Telemetry.Finalize

This event sends data indicating that the device has invoked the finalize phase of the upgrade, to help keep Windows up-to-date.

The following fields are available:

- **ClientId** With Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **HostOSBuildNumber** The build number of the previous OS.
- **HostOsSkuName** The OS edition which is running Setup360 instance (previous OS).
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe
- **ReportId** With Windows Update, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened

- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that is used to diagnose errors.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **TestId** A string to uniquely identify a group of events.
- **Wuid** This is the Windows Update Client ID. With Windows Update, this is the same as the clientId.

Setup360Telemetry.PostRebootInstall

This event sends data indicating that the device has invoked the postrebootinstall phase of the upgrade, to help keep Windows up-to-date.

The following fields are available:

- **ClientId** With Windows Update, this is the Windows Update client ID that is passed to Setup. In Media setup, the default value is Media360, but can be overwritten by the caller to a unique value.
- **HostOSBuildNumber** The build number of the previous OS.
- **HostOsSkuName** The OS edition which is running Setup360 instance (previous OS).
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe.
- **ReportId** With Windows Update, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that's used to diagnose errors.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **TestId** A string to uniquely identify a group of events.
- **Wuid** This is the Windows Update Client ID. With Windows Update, this is the same as ClientId.

Setup360Telemetry.PreDownloadQuiet

This event sends data indicating that the device has invoked the predownload quiet phase of the upgrade, to help keep Windows up to date.

The following fields are available:

- **ClientId** Using Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **HostOSBuildNumber** The build number of the previous OS.
- **HostOsSkuName** The OS edition which is running Setup360 instance (previous operating system).
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe
- **ReportId** Using Windows Update, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that is used to diagnose errors.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, canceled

- **TestId** A string to uniquely identify a group of events.
- **Wuld** This is the Windows Update Client ID. Using Windows Update, this is the same as the clientId.

Setup360Telemetry.OsUninstall

The event sends data regarding OS updates and upgrades from Windows 7, Windows 8, and Windows 10. Specifically, the Setup360Telemetry.OSUninstall indicates the outcome of an OS uninstall.

The following fields are available:

- **ClientId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **HostOSBuildNumber** The build number of the previous OS.
- **HostOsSkuName** The OS edition which is running the Setup360 instance (previous OS).
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe.
- **ReportId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, this is the GUID for the install.wim.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that is used to diagnose errors.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).
- **State** Exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **TestId** A string to uniquely identify a group of events.
- **Wuld** Windows Update client ID.

Setup360Telemetry.Downlevel

This event sends data indicating that the device has invoked the downlevel phase of the upgrade. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **ClientId** If using Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, the default value is Media360, but it can be overwritten by the caller to a unique value.
- **HostOSBuildNumber** The build number of the downlevel OS.
- **HostOsSkuName** The operating system edition which is running Setup360 instance (downlevel OS).
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe.
- **ReportId** In the Windows Update scenario, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. It's an HRESULT error code that can be used to diagnose errors.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **SetupVersionBuildNumber** The build number of Setup360 (build number of the target OS).
- **State** Exit state of given Setup360 run. Example: succeeded, failed, blocked, cancelled
- **TestId** A string that uniquely identifies a group of events.
- **Wuld** This is the Windows Update Client ID. In the Windows Update scenario, this is the same as the clientId.

Setup360Telemetry.PreInstallUX

This event sends data regarding OS updates and upgrades from Windows 7, Windows 8, and Windows 10.

Specifically, the Setup360Telemetry.PreinstallUX indicates the outcome of the PreinstallUX portion of the update process.

The following fields are available:

- **ClientId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **HostOSBuildNumber** The build number of the previous OS.
- **HostOsSkuName** The OS edition which is running the Setup360 instance (previous OS).
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe.
- **ReportId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, this is the GUID for the install.wim.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that is used to diagnose errors.
- **Setup360Scenario** The Setup360 flow type, Example: Boot, Media, Update, MCT
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **TestId** A string to uniquely identify a group of events.
- **Wuid** Windows Update client ID.

Setup360Telemetry.Setup360

This event sends data about OS deployment scenarios, to help keep Windows up-to-date.

The following fields are available:

- **FieldName** Retrieves the data point.
- **FlightData** Specifies a unique identifier for each group of Windows Insider builds.
- **InstanceId** Retrieves a unique identifier for each instance of a setup session.
- **ReportId** Retrieves the report ID.
- **Scenarioid** Retrieves the deployment scenario.
- **Value** Retrieves the value associated with the corresponding FieldName.
- **ClientId** Retrieves the upgrade ID: Upgrades via Windows Update - specifies the WU clientID. All other deployment - static string.

Windows as a Service diagnostic events

Microsoft.Windows.WaaSMedic.SummaryEvent

This event provides the results from the WaaSMedic engine

The following fields are available:

- **detectionSummary** Result of each detection that ran
- **featureAssessmentImpact** Windows as a Service (WaaS) Assessment impact on feature updates
- **insufficientSessions** True, if the device has enough activity to be eligible for update diagnostics. False, if otherwise
- **isManaged** Indicates the device is managed for updates
- **isWUConnected** Indicates the device is connected to Windows Update
- **noMoreActions** All available WaaSMedic diagnostics have run. There are no pending diagnostics and corresponding actions
- **qualityAssessmentImpact** Windows as a Service (WaaS) Assessment impact for quality updates

- **remediationSummary** Result of each operation performed on a device to fix an invalid state or configuration that's preventing the device from getting updates. For example, if Windows Update service is turned off, the fix is to turn the it back on
- **usingBackupFeatureAssessment** The WaaSMedic engine contacts Windows as a Service (WaaS) Assessment to determine whether the device is up-to-date. If WaaS Assessment isn't available, the engine falls back to backup feature assessments, which are determined programmatically on the client#N#
- **usingBackupQualityAssessment** The WaaSMedic engine contacts Windows as a Service (WaaS) Assessment to determine whether the device is up-to-date. If WaaS Assessment isn't available, the engine falls back to backup quality assessments, which are determined programmatically on the client#N#
- **versionString** Installed version of the WaaSMedic engine
- **hrEngineResult** Indicates the WaaSMedic engine operation error codes

Microsoft.Windows.WaaSMedic.Summary

This event provides the results of the WaaSMedic diagnostic run

The following fields are available:

- **detectionSummary** Result of each detection that ran
- **remediationSummary** Result of each operation performed on a device to fix an invalid state or configuration that's preventing the device from getting updates. For example, if Windows Update service is turned off, the fix is to turn the it back on
- **versionString** Installed version of the WaaSMedic engine
- **featureAssessmentImpact** Windows as a Service (WaaS) Assessment impact on feature updates
- **insufficientSessions** True, if the device has enough activity to be eligible for update diagnostics. False, if otherwise
- **isManaged** Indicates the device is managed for updates
- **isWUConnected** Indicates the device is connected to Windows Update
- **noMoreActions** All available WaaSMedic diagnostics have run. There are no pending diagnostics and corresponding actions
- **qualityAssessmentImpact** Windows as a Service (WaaS) Assessment impact for quality updates
- **usingBackupFeatureAssessment** The WaaSMedic engine contacts Windows as a Service (WaaS) Assessment to determine whether the device is up-to-date. If WaaS Assessment isn't available, the engine falls back to backup feature assessments, which are determined programmatically on the client
- **usingBackupQualityAssessment** The WaaSMedic engine contacts Windows as a Service (WaaS) Assessment to determine whether the device is up-to-date. If WaaS Assessment isn't available, the engine falls back to backup quality assessments, which are determined programmatically on the client

Windows Error Reporting events

Microsoft.Windows.WERVertical.OSCrash

This event sends binary data from the collected dump file whenever a bug check occurs, to help keep Windows up to date. This is the OneCore version of this event.

The following fields are available:

- **BootId** UInt32 identifying the boot number for this device.
- **BugCheckCode** "UInt64 ""bugcheck code"" that identifies a proximate cause of the bug check."
- **BugCheckParameter1** UInt64 parameter providing additional information.
- **BugCheckParameter2** UInt64 parameter providing additional information.
- **BugCheckParameter3** UInt64 parameter providing additional information.
- **BugCheckParameter4** UInt64 parameter providing additional information.

- **DumpFileAttributes** Codes that identify the type of data contained in the dump file
- **DumpFileSize** Size of the dump file
- **IsValidDumpFile** True if the dump file is valid for the debugger, false otherwise
- **ReportId** WER Report Id associated with this bug check (used for finding the corresponding report archive in Watson).

Windows Store events

Microsoft.Windows.StoreAgent.Telemetry.AbortedInstallation

This event is sent when an installation or update is canceled by a user or the system and is used to help keep Windows Apps up to date and secure.

The following fields are available:

- **AggregatedPackageFullNames** The names of all packages to be downloaded and installed.
- **AttemptNumber** Number of retry attempts before it was canceled.
- **BundleId** The Item Bundle ID.
- **CategoryId** The Item Category ID.
- **ClientAppId** The identity of the app that initiated this operation.
- **HResult** The result code of the last action performed before this operation.
- **IntentPFNs** Intent Product Family Name
- **IsBundle** Is this a bundle?
- **IsInteractive** Was this requested by a user?
- **IsMandatory** Was this a mandatory update?
- **IsRemediation** Was this a remediation install?
- **IsRestore** Is this automatically restoring a previously acquired product?
- **IsUpdate** Flag indicating if this is an update.
- **IsWin32** Flag indicating if this is a Win32 app (not used).
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **PFN** The product family name of the product being installed.
- **ProductId** The identity of the package or packages being installed.
- **SystemAttemptNumber** The total number of automatic attempts at installation before it was canceled.
- **UpdateId** Update ID (if this is an update)
- **UserAttemptNumber** The total number of user attempts at installation before it was canceled.
- **WUContentId** The Windows Update content ID

Microsoft.Windows.StoreAgent.Telemetry.EndAcquireLicense

This event is sent after the license is acquired when a product is being installed. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **AggregatedPackageFullNames** Includes a set of package full names for each app that is part of an atomic set.
- **AttemptNumber** The total number of attempts to acquire this product.
- **BundleId** The bundle ID
- **CategoryId** The identity of the package or packages being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **HResult** HResult code to show the result of the operation (success/failure).
- **IntentPFNs** Intent Product Family Name

- **IsBundle** Is this a bundle?
- **IsInteractive** Did the user initiate the installation?
- **IsMandatory** Is this a mandatory update?
- **IsRemediation** Is this repairing a previous installation?
- **IsRestore** Is this happening after a device restore?
- **IsUpdate** Is this an update?
- **IsWin32** Flag indicating if this is a Win32app.
- **ParentBundledId** The product's parent bundle ID.
- **ParentBundleId** The parent bundle ID (if it's part of a bundle).
- **PFN** Product Family Name of the product being installed.
- **ProductId** The Store Product ID for the product being installed.
- **SystemAttemptNumber** The number of attempts by the system to acquire this product.
- **UpdateId** The update ID (if this is an update)
- **UserAttemptNumber** The number of attempts by the user to acquire this product
- **WUContentId** The Windows Update content ID

Microsoft.Windows.StoreAgent.Telemetry.EndDownload

This event happens during the app update or installation when content is being downloaded at the end of the process to report success or failure. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **AggregatedPackageFullNames** The name of all packages to be downloaded and installed.
- **AttemptNumber** Number of retry attempts before it was canceled.
- **BundleId** The identity of the Windows Insider build associated with this product.
- **CategoryId** The identity of the package or packages being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **DownloadSize** The total size of the download.
- **ExtendedHResult** Any extended HResult error codes.
- **HResult** The result code of the last action performed.
- **IntentPFNs** Intent Product Family Name
- **IsBundle** Is this a bundle?
- **IsInteractive** Is this initiated by the user?
- **IsMandatory** Is this a mandatory installation?
- **IsRemediation** Is this repairing a previous installation?
- **IsRestore** Is this a restore of a previously acquired product?
- **IsUpdate** Is this an update?
- **IsWin32** Flag indicating if this is a Win32 app (unused).
- **ParentBundledId** The parent bundle ID (if it's part of a bundle).
- **PFN** The Product Family Name of the app being download.
- **ProductId** The Store Product ID for the product being installed.
- **SystemAttemptNumber** The number of attempts by the system to download.
- **UpdateId** Update ID (if this is an update)
- **UserAttemptNumber** The number of attempts by the user to download.
- **WUContentId** The Windows Update content ID.

Microsoft.Windows.StoreAgent.Telemetry.EndFrameworkUpdate

This event happens when an app update requires an updated Framework package and the process starts to download it. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **HResult** The result code of the last action performed before this operation.

Microsoft.Windows.StoreAgent.Telemetry.EndGetInstalledContentIds

This event is sent after sending the inventory of the products installed to determine whether updates for those products are available. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **HResult** The result code of the last action performed before this operation.

Microsoft.Windows.StoreAgent.Telemetry.EndInstall

This event is sent after a product has been installed. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **AggregatedPackageFullNames** The names of all packages to be downloaded and installed.
- **AttemptNumber** The number of retry attempts before it was canceled.
- **BundleId** The identity of the build associated with this product.
- **CategoryId** The identity of the package or packages being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **ExtendedHResult** The extended HResult error code.
- **HResult** The result code of the last action performed.
- **IntentPFNs** Intent Product Family Name
- **IsBundle** Is this a bundle?
- **IsInteractive** Is this an interactive installation?
- **IsMandatory** Is this a mandatory installation?
- **IsRemediation** Is this repairing a previous installation?
- **IsRestore** Is this automatically restoring a previously acquired product?
- **IsUpdate** Is this an update?
- **IsWin32** Flag indicating if this a Win32 app (unused).
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **PFN** Product Family Name of the product being installed.
- **ProductId** The Store Product ID for the product being installed.
- **SystemAttemptNumber** The total number of system attempts.
- **UpdateId** Update ID (if this is an update)
- **UserAttemptNumber** The total number of user attempts.
- **WUContentId** The Windows Update content ID

Microsoft.Windows.StoreAgent.Telemetry.EndScanForUpdates

This event is sent after a scan for product updates to determine if there are packages to install. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **ClientAppId** The identity of the app that initiated this operation.
- **HResult** The result code of the last action performed.
- **IsApplicability** Is this request to only check if there are any applicable packages to install?
- **IsInteractive** Is this user requested?
- **IsOnline** Is the request doing an online check?

Microsoft.Windows.StoreAgent.Telemetry.EndSearchUpdatePackages

This event is sent after searching for update packages to install. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **AggregatedPackageFullNames** The names of all packages to be downloaded and installed.
- **AttemptNumber** The total number of retry attempts before it was canceled.
- **BundleId** The identity of the build associated with this product.
- **CategoryId** The identity of the package or packages being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **HResult** The result code of the last action performed.
- **IntentPFNs** The licensing identity of this package.
- **IsBundle** Is this a bundle?
- **IsInteractive** Is this user requested?
- **IsMandatory** Is this a mandatory update?
- **IsRemediation** Is this repairing a previous installation?
- **IsRestore** Is this restoring previously acquired content?
- **IsUpdate** Is this an update?
- **IsWin32** Flag indicating if this a Win32 app (unused).
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **PFN** The name of the package or packages requested for install.
- **ProductId** The Store Product ID for the product being installed.
- **SystemAttemptNumber** The total number of system attempts.
- **UpdateId** Update ID (if this is an update)
- **UserAttemptNumber** The total number of user attempts.
- **WUContentId** The Windows Update content ID

Microsoft.Windows.StoreAgent.Telemetry.EndStageUserData

This event is sent between download and installation to see if there is app data that needs to be restored from the cloud. It's used to keep Windows up-to-date and secure.

The following fields are available:

- **AttemptNumber** The total number of retry attempts before it was canceled.
- **BundleId** The identity of the build associated with this product.
- **CategoryId** The identity of the package or packages being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **HResult** The result code of the last action performed.
- **IsBundle** Is this a bundle?
- **IsInteractive** Is this user requested?
- **IsMandatory** Is this a mandatory update?
- **IsRemediation** Is this repairing a previous installation?
- **IsRestore** Is this restoring previously acquired content?
- **IsUpdate** Is this an update?
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **PFN** The name of the package or packages requested for install.
- **ProductId** The Store Product ID for the product being installed.
- **SystemAttemptNumber** The total number of system attempts.
- **UserAttemptNumber** The total number of user attempts.
- **WUContentId** The Windows Update content ID

- **IntentPFNs** The licensing identity of this package.
- **AggregatedPackageFullNames** The name of all packages to be downloaded and installed.

Microsoft.Windows.StoreAgent.Telemetry.InstallOperationRequest

This event happens at the beginning of the install process when an app update or new app is installed. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **BundleId** The identity of the build associated with this product.
- **CatalogId** If this product is from a private catalog, the Store Product ID for the product being installed.
- **ProductId** The Store Product ID for the product being installed.
- **Skuid** Specific edition ID being installed.
- **VolumePath** The disk path of the installation.

Microsoft.Windows.StoreAgent.Telemetry.PauseInstallation

This event is sent when a product install or update is paused either by a user or the system. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **AttemptNumber** The total number of retry attempts before it was canceled.
- **BundleId** The identity of the build associated with this product.
- **CategoryId** The identity of the package or packages being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **IsBundle** Is this a bundle?
- **IsInteractive** Is this user requested?
- **IsMandatory** Is this a mandatory update?
- **IsRemediation** Is this repairing a previous installation?
- **IsRestore** Is this restoring previously acquired content?
- **IsUpdate** Is this an update?
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **PFN** The Product Full Name.
- **PreviousHResult** The result code of the last action performed before this operation.
- **PreviousInstallState** Previous state before the installation or update was paused.
- **ProductId** The Store Product ID for the product being installed.
- **RelatedCV** Correlation Vector of a previous performed action on this product.
- **SystemAttemptNumber** The total number of system attempts.
- **UserAttemptNumber** The total number of user attempts.
- **WUContentId** The Windows Update content ID
- **IntentPFNs** The licensing identity of this package.
- **AggregatedPackageFullNames** The names of all packages to be downloaded and installed.

Microsoft.Windows.StoreAgent.Telemetry.ResumeInstallation

This event happens when a product install or update is resumed either by a user or the system. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **AttemptNumber** The number of retry attempts before it was canceled.
- **BundleId** The identity of the build associated with this product.
- **CategoryId** The identity of the package or packages being installed.

- **ClientAppId** The identity of the app that initiated this operation.
- **IsBundle** Is this a bundle?
- **IsInteractive** Is this user requested?
- **IsMandatory** Is this a mandatory update?
- **IsRemediation** Is this repairing a previous installation?
- **IsRestore** Is this restoring previously acquired content?
- **IsUpdate** Is this an update?
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **PFN** The name of the package or packages requested for install.
- **PreviousHResult** The previous HResult error code.
- **PreviousInstallState** Previous state before the installation was paused.
- **ProductId** The Store Product ID for the product being installed.
- **RelatedCV** Correlation Vector for the original install before it was resumed.
- **SystemAttemptNumber** The total number of system attempts.
- **UserAttemptNumber** The total number of user attempts.
- **WUContentId** The Windows Update content ID
- **IntentPFNs** Intent Product Family Name
- **AggregatedPackageFullNames** The names of all packages to be downloaded and installed.
- **HResult** The result code of the last action performed before this operation.
- **IsUserRetry** Did the user initiate the retry?

Microsoft.Windows.StoreAgent.Telemetry.UpdateAppOperationRequest

This event happens an app for a user needs to be updated. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **PFamN** The name of the product that is requested for update.

Microsoft.Windows.StoreAgent.Telemetry.CancelInstallation

This event is sent when an app update or installation is canceled while in interactive mode. This can be canceled by the user or the system. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **AttemptNumber** Total number of installation attempts.
- **BundleId** The identity of the Windows Insider build that is associated with this product.
- **CategoryId** The identity of the package or packages being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **IsBundle** Is this a bundle?
- **IsInteractive** Was this requested by a user?
- **IsMandatory** Is this a mandatory update?
- **IsRemediation** Is this repairing a previous installation?
- **IsRestore** Is this an automatic restore of a previously acquired product?
- **IsUpdate** Is this a product update?
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **PFN** The name of all packages to be downloaded and installed.
- **PreviousHResult** The previous HResult code.
- **PreviousInstallState** Previous installation state before it was canceled.
- **ProductId** The name of the package or packages requested for installation.
- **RelatedCV** Correlation Vector of a previous performed action on this product.

- **SystemAttemptNumber** Total number of automatic attempts to install before it was canceled.
- **UserAttemptNumber** Total number of user attempts to install before it was canceled.
- **WUContentId** The Windows Update content ID
- **IntentPFNs** Intent Product Family Name
- **AggregatedPackageFullNames** The names of all package or packages to be downloaded and installed.

Microsoft.Windows.StoreAgent.Telemetry.SearchForUpdateOperationRequest

This event is sent when searching for update packages to install. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **CatalogId** The Store Product ID for the product being installed.
- **ProductId** The Store Product ID for the product being installed.
- **Skuid** Specific edition of the app being updated.

Microsoft.Windows.StoreAgent.Telemetry.EndUpdateMetadataPrepare

This event happens after a scan for available app updates. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **HResult** The result code of the last action performed.

Microsoft.Windows.StoreAgent.Telemetry.CompleteInstallOperationRequest

This event is sent after the app installations or updates. It's used to help keep Windows up-to-date and secure

The following fields are available:

- **CatalogId** The Store Product ID of the app being installed.
- **HResult** HResult code of the action being performed.
- **IsBundle** Is this a bundle?
- **PackageFamilyName** The name of the package being installed.
- **ProductId** The Store Product ID of the product being installed.
- **Skuid** Specific edition of the item being installed.

Microsoft.Windows.StoreAgent.Telemetry.ResumeOperationRequest

This event happens when a product install or update is resumed by a user and on installation retries. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **ProductId** The Store Product ID for the product being installed.

Microsoft.Windows.StoreAgent.Telemetry.FulfillmentComplete

This event is sent at the end of an app install or update and is used to track the very end of the install or update process.

The following fields are available:

- **FailedRetry** Was the installation or update retry successful?
- **HResult** The HResult code of the operation.
- **PFN** The Package Family Name of the app that is being installed or updated.
- **ProductId** The product ID of the app that is being updated or installed.

Microsoft.Windows.StoreAgent.Telemetry.FulfillmentInitiate

This event is sent at the beginning of an app install or update and is used to track the very beginning of the install

or update process.

The following fields are available:

- **PFN** The Package Family Name of the app that is being installed or updated.
- **ProductId** The product ID of the app that is being updated or installed.

Windows Update Delivery Optimization events

Microsoft.OSG.DU.DeliveryOptClient.DownloadCompleted

This event describes when a download has completed with Delivery Optimization. It's used to understand and address problems regarding downloads.

The following fields are available:

- **background** Is the download a background download?
- **bytesFromCDN** The number of bytes received from a CDN source.
- **bytesFromGroupPeers** The number of bytes received from a peer in the same domain group.
- **bytesFromIntPeers** The number of bytes received from peers not in the same LAN or in the same domain group.
- **bytesFromPeers** The number of bytes received from a peer in the same LAN.
- **bytesRequested** The total number of bytes requested for download.
- **cdnConnectionCount** The total number of connections made to the CDN.
- **cdnErrorCodes** A list of CDN connection errors since the last FailureCDNCommunication event.
- **cdnErrorCounts** The number of times each error in cdnErrorCodes was encountered.
- **cdnIp** The IP address of the source CDN.
- **clientTeId** A random number used for device sampling.
- **doErrorCode** The Delivery Optimization error code that was returned.
- **downlinkBps** The maximum measured available download bandwidth (in bytes per second).
- **downlinkUsageBps** The download speed (in bytes per second).
- **downloadMode** The download mode used for this file download session.
- **fileID** The ID of the file being downloaded.
- **fileSize** The size of the file being downloaded.
- **groupConnectionCount** The total number of connections made to peers in the same group.
- **internetConnectionCount** The total number of connections made to peers not in the same LAN or the same group.
- **lanConnectionCount** The total number of connections made to peers in the same LAN.
- **numPeers** The total number of peers used for this download.
- **restrictedUpload** Is the upload restricted?
- **scenarioID** The ID of the scenario.
- **sessionID** The ID of the download session.
- **totalTimeMs** Duration of the download (in seconds).
- **updateID** The ID of the update being downloaded.
- **uplinkBps** The maximum measured available upload bandwidth (in bytes per second).
- **uplinkUsageBps** The upload speed (in bytes per second).
- **experimentId** When running a test, this is used to correlate with other events that are part of the same test.
- **isVpn** Is the device connected to a Virtual Private Network?
- **usedMemoryStream** Did the download use memory streaming?

Microsoft.OSG.DU.DeliveryOptClient.DownloadPaused

This event represents a temporary suspension of a download with Delivery Optimization. It's used to understand and address problems regarding downloads.

The following fields are available:

- **background** Is the download a background download?
- **clientTeId** A random number used for device sampling.
- **errorCode** The error code that was returned.
- **fileID** The ID of the file being paused.
- **reasonCode** The reason for pausing the download.
- **scenarioID** The ID of the scenario.
- **sessionId** The ID of the download session.
- **updateID** The ID of the update being paused.
- **experimentId** When running a test, this is used to correlate with other events that are part of the same test.
- **isVpn** Is the device connected to a Virtual Private Network?

Microsoft.OSG.DU.DeliveryOptClient.JobError

This event represents a Windows Update job error. It allows for investigation of top errors.

The following fields are available:

- **clientTeId** A random number used for device sampling.
- **errorCode** The error code returned.
- **experimentId** When running a test, this is used to correlate with other events that are part of the same test.
- **fileID** The ID of the file being downloaded.
- **jobID** The Windows Update job ID.

Microsoft.OSG.DU.DeliveryOptClient.DownloadCanceled

This event describes when a download was canceled with Delivery Optimization. It's used to understand and address problems regarding downloads.

The following fields are available:

- **background** Is the download being done in the background?
- **bytesFromCDN** The number of bytes received from a CDN source.
- **bytesFromGroupPeers** The number of bytes received from a peer in the same group.
- **bytesFromIntPeers** The number of bytes received from peers not in the same LAN or in the same group.
- **bytesFromPeers** The number of bytes received from a peer in the same LAN.
- **cdnErrorCodes** A list of CDN connection errors since the last FailureCDNCommunication event.
- **cdnErrorCounts** The number of times each error in cdnErrorCodes was encountered.
- **clientTeId** A random number used for device sampling.
- **doErrorCode** The Delivery Optimization error code that was returned.
- **errorCode** The error code that was returned.
- **experimentId** When running a test, this is used to correlate events that are part of the same test.
- **fileID** The ID of the file being downloaded.
- **isVpn** Is the device connected to a Virtual Private Network?
- **scenarioID** The ID of the scenario.
- **sessionId** The ID of the file download session.
- **updateID** The ID of the update being downloaded.
- **usedMemoryStream** Did the download use memory streaming?

Microsoft.OSG.DU.DeliveryOptClient.DownloadStarted

This event describes the start of a new download with Delivery Optimization. It's used to understand and address problems regarding downloads.

The following fields are available:

- **background** Is the download a background download?
- **cdnUrl** The URL of the CDN.
- **clientTeId** A random number used for device sampling.
- **deviceProfile** Identifies the usage or form factor. Example: Desktop or Xbox
- **diceRoll** The dice roll value used in sampling events.
- **doClientVersion** The version of the Delivery Optimization client.
- **doErrorCode** The Delivery Optimization error code that was returned.
- **downloadMode** The download mode used for this file download session.
- **errorCode** The error code that was returned.
- **experimentId** When running a test, this is used to correlate with other events that are part of the same test.
- **fileID** The ID of the file being downloaded.
- **filePath** The path where the file will be written.
- **groupID** ID for the group.
- **isVpn** Is the device connected to a Virtual Private Network?
- **jobID** The ID of the Windows Update job.
- **minDiskSizeGB** The minimum disk size (in GB) required for Peering.
- **minDiskSizePolicyEnforced** Is the minimum disk size enforced via policy?
- **minFileSizePolicy** The minimum content file size policy to allow the download using Peering.
- **peerID** The ID for this Delivery Optimization client.
- **scenarioID** The ID of the scenario.
- **sessionId** The ID of the download session.
- **updateID** The ID of the update being downloaded.
- **usedMemoryStream** Did the download use memory streaming?
- **costFlags** A set of flags representing network cost.

Microsoft.OSG.DU.DeliveryOptClient.FailureCdnCommunication

This event represents a failure to download from a CDN with Delivery Optimization. It's used to understand and address problems regarding downloads.

The following fields are available:

- **cdnIp** The IP address of the CDN.
- **cdnUrl** The URL of the CDN.
- **clientTeId** A random number used for device sampling.
- **errorCode** The error code that was returned.
- **errorCount** The total number of times this error code was seen since the last FailureCdnCommunication event was encountered.
- **httpStatusCode** The HTTP status code returned by the CDN.
- **sessionId** The ID of the download session.
- **cdnHeaders** The HTTP headers returned by the CDN.
- **experimentId** When running a test, this is used to correlate with other events that are part of the same test.
- **fileID** The ID of the file being downloaded.
- **isHeadRequest** The type of HTTP request that was sent to the CDN. Example: HEAD or GET
- **requestSize** The size of the range requested from the CDN.
- **responseSize** The size of the range response received from the CDN.

Windows Update events

Microsoft.Windows.Update.DeviceUpdateAgent.UpdateAgentModeStart

This event sends data for the start of each mode during the process of updating device manifest assets via the UUP (Unified Update Platform) update scenario, which is used to install a device manifest describing a set of driver packages.

The following fields are available:

- **flightId** The unique identifier for each flight
- **mode** Indicates that the Update Agent mode that has started. 1 = Initialize, 2 = DownloadRequest, 3 = Install, 4 = Commit
- **objectId** Unique value for each Update Agent mode
- **relatedCV** Correlation vector value generated from the latest scan
- **scenarioid** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **sessionId** Unique value for each Update Agent mode attempt
- **updateId** Unique ID for each update

Microsoft.Windows.Update.DeviceUpdateAgent.UpdateAgentInitialize

This event sends data for initializing a new update session for the new device manifest UUP (Unified Update Platform) update scenario, which is used to install a device manifest describing a set of driver packages

The following fields are available:

- **errorCode** The error code returned for the current initialize phase
- **flightId** The unique identifier for each flight
- **flightMetadata** Contains the FlightId and the build being flighted
- **objectId** Unique value for each Update Agent mode
- **relatedCV** Correlation vector value generated from the latest USO scan
- **result** Result of the initialize phase of update. 0 = Succeeded, 1 = Failed, 2 = Cancelled, 3 = Blocked, 4 = BlockCancelled
- **scenarioid** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate#N#
- **sessionData** Contains instructions to update agent for processing FODs and DUCIs (Null for other scenarios)
- **sessionId** "Unique value for each Update Agent mode attempt "
- **updateId** Unique ID for each update

Microsoft.Windows.Update.DeviceUpdateAgent.UpdateAgentCommit

This event collects information regarding the final commit phase of the new device manifest UUP (Unified Update Platform) update scenario, which is used to install a device manifest describing a set of driver packages

The following fields are available:

- **errorCode** The error code returned for the current session initialization
- **flightId** The unique identifier for each flight
- **objectId** The unique GUID for each diagnostics session
- **relatedCV** A correlation vector value, generated from the latest USO scan
- **result** Outcome of the initialization of the session
- **scenarioid** Identifies the Update scenario
- **sessionId** The unique value for each update session
- **updateId** The unique identifier for each Update

Microsoft.Windows.Update.DeviceUpdateAgent.UpdateAgentInstall

This event collects information regarding the install phase of the new device manifest UUP (Unified Update Platform) update scenario, which is used to install a device manifest describing a set of driver packages

The following fields are available:

- **errorCode** The error code returned for the current install phase
- **flightId** The unique identifier for each flight
- **objectId** Unique value for each Update Agent mode
- **relatedCV** Correlation vector value generated from the latest scan
- **result** Result of the install phase of update. 0 = Succeeded 1 = Failed, 2 = Cancelled, 3 = Blocked, 4 = BlockCancelled
- **scenarioid** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **sessionId** Unique value for each Update Agent mode attempt
- **updateId** Unique ID for each update

Microsoft.Windows.Update.DeviceUpdateAgent.UpdateAgentDownloadRequest

This event collects information regarding the download request phase of the new device manifest UUP (Unified Update Platform) update scenario, which is used to install a device manifest describing a set of driver packages

The following fields are available:

- **deletedCorruptFiles** Indicates if UpdateAgent found any corrupt payload files and whether the payload was deleted
- **errorCode** The error code returned for the current session initialization
- **flightId** The unique identifier for each flight
- **objectId** Unique value for each Update Agent mode
- **packageCountOptional** Number of optional packages requested
- **packageCountRequired** Number of required packages requested
- **packageCountTotal** Total number of packages needed
- **packageCountTotalCanonical** Total number of canonical packages
- **packageCountTotalDiff** Total number of diff packages
- **packageCountTotalExpress** Total number of express packages
- **packageSizeCanonical** Size of canonical packages in bytes
- **packageSizeDiff** Size of diff packages in bytes
- **packageSizeExpress** Size of express packages in bytes
- **rangeRequestState** Represents the state of the download range request
- **relatedCV** Correlation vector value generated from the latest USO scan
- **result** Result of the download request phase of update
- **scenarioid** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **sessionId** Unique value for each Update Agent mode attempt
- **updateId** Unique ID for each update

Microsoft.Windows.Update.Orchestrator.GameActive

This event indicates that an enabled GameMode process prevented the device from restarting to complete an update

The following fields are available:

- **eventScenario** Indicates the purpose of sending this event - whether because the software distribution just

started checking for content, or whether it was cancelled, succeeded, or failed

- **gameModeReason** Name of the enabled GameMode process that prevented the device from restarting to complete an update
- **wuDeviceId** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue

Microsoft.Windows.Update.DataMigrationFramework.DmfMigrationCompleted

This event sends data collected at the end of the Data Migration Framework (DMF) and parameters involved in its invocation, to help keep Windows up to date.

The following fields are available:

- **MigrationDurationInMilliseconds** How long the DMF migration took (in milliseconds)
- **MigrationEndTime** A system timestamp of when the DMF migration completed.
- **RevisionNumbers** A collection of revision numbers for the updates associated with the DMF session.
- **UpdateIds** A collection of GUIDs for updates that are associated with the DMF session.
- **WuClientId** The GUID of the Windows Update client responsible for triggering the DMF migration

Microsoft.Windows.Update.DataMigrationFramework.DmfMigrationStarted

This event sends data collected at the beginning of the Data Migration Framework (DMF) and parameters involved in its invocation, to help keep Windows up to date.

The following fields are available:

- **MigrationMicrosoftPhases** Revision numbers for the updates that were installed.
- **MigrationOEMPhases** WU Update IDs for the updates that were installed.
- **MigrationStartTime** The timestamp representing the beginning of the DMF migration
- **WuClientId** The GUID of the Windows Update client invoking DMF
- **RevisionNumbers** A collection of the revision numbers associated with the UpdateIds.
- **UpdateIds** A collection of GUIDs identifying the upgrades that are running.

Microsoft.Windows.Update.DataMigrationFramework.MigratorResult

This event sends DMF migrator data to help keep Windows up to date.

The following fields are available:

- **CurrentStep** This is the last step the migrator reported before returning a result. This tells us how far through the individual migrator the device was before failure.
- **ErrorCode** The result (as an HRESULT) of the migrator that just completed.
- **MigratorId** A GUID identifying the migrator that just completed.
- **MigratorName** The name of the migrator that just completed.
- **RunDurationInSeconds** The time it took for the migrator to complete.
- **TotalSteps** Migrators report progress in number of completed steps against the total steps. This is the total number of steps.

Microsoft.Windows.Update.Orchestrator.Download

This event sends launch data for a Windows Update download to help keep Windows up to date.

The following fields are available:

- **deferReason** Reason for download not completing
- **detectionDeferralReason** Reason for download not completing
- **errorCode** An error code represented as a hexadecimal value
- **eventScenario** End to end update session ID.

- **flightID** Unique update ID.
- **interactive** Identifies if session is user initiated.
- **revisionNumber** Update revision number.
- **updateId** Update ID.
- **updateScenarioType** The update session type.
- **wuDeviceid** Unique device ID used by Windows Update.

Microsoft.Windows.Update.Orchestrator.FlightInapplicable

This event sends data on whether the update was applicable to the device, to help keep Windows up to date.

The following fields are available:

- **EventPublishedTime** time that the event was generated
- **revisionNumber** Revision Number of the Update
- **updateId** Unique Update ID
- **UpdateStatus** Integer that describes Update state
- **wuDeviceid** Unique Device ID
- **flightID** Unique Update ID
- **updateScenarioType** The update session type.

Microsoft.Windows.Update.Orchestrator.PostInstall

This event sends data about lite stack devices (mobile, IOT, anything non-PC) immediately before data migration is launched to help keep Windows up to date.

The following fields are available:

- **batteryLevel** Current battery capacity in mWh or percentage left.
- **bundleId** Update grouping ID.
- **bundleRevisionnumber** Bundle revision number.
- **errorCode** Hex code for the error message, to allow lookup of the specific error.
- **eventScenario** End to end update session ID.
- **flightID** Unique update ID.
- **sessionType** Interactive vs. Background.
- **wuDeviceid** Unique device ID used by Windows Update.

Microsoft.Windows.Update.Orchestrator.RebootFailed

This event sends information about whether an update required a reboot and reasons for failure to help keep Windows up to date.

The following fields are available:

- **batteryLevel** Current battery capacity in mWh or percentage left.
- **deferReason** Reason for install not completing.
- **EventPublishedTime** The time that the reboot failure occurred.
- **flightID** Unique update ID.
- **installRebootDeferration** Reason for reboot not occurring.
- **rebootOutsideOfActiveHours** Indicates the timing that the reboot was to occur to ensure the correct update process and experience is provided to keep Windows up to date.
- **RebootResults** Hex code indicating failure reason. Typically, we expect this to be a specific USO generated hex code.
- **revisionNumber** Update revision number.
- **updateId** Update ID.

- **updateScenarioType** The update session type.
- **uxRebootstate** Indicates the exact state of the user experience at the time the required reboot was initiated to ensure the correct update process and experience is provided to keep Windows up to date.
- **wuDeviceid** Unique device ID used by Windows Update.

Microsoft.Windows.Update.Orchestrator.UpdatePolicyCacheRefresh

This event sends data on whether Update Management Policies were enabled on a device, to help keep Windows up to date.

The following fields are available:

- **configuredPoliciescount** Policy Count
- **policiesNamevaluesource** Policy Name
- **policyCacherefreshTime** Refresh time
- **updateInstallUXSetting** This shows whether a user has set policies via UX option
- **wuDeviceid** Unique device ID used by Windows Update.

Microsoft.Windows.Update.Orchestrator.UpdateRebootRequired

This event sends data about whether an update required a reboot to help keep Windows up to date.

The following fields are available:

- **revisionNumber** Update revision number.
- **updateId** Update ID.
- **wuDeviceid** Unique device ID used by Windows Update.
- **flightID** Unique update ID.
- **interactive** Indicates the reboot initiation stage of the update process was entered as a result of user action or not.
- **uxRebootstate** Indicates the exact state of the user experience at the time the required reboot was initiated to ensure the correct update process and experience is provided to keep Windows up to date.
- **updateScenarioType** The update session type.

Microsoft.Windows.Update.Ux.MusNotification.RebootScheduled

This event sends data about a required reboot that is scheduled with no user interaction, to help keep Windows up to date.

The following fields are available:

- **activeHoursApplicable** True, If Active Hours applicable on this device. False, otherwise.
- **forcedReboot** True, if a reboot is forced on the device. Otherwise, this is False
- **rebootArgument** Argument for the reboot task. It also represents specific reboot related action.
- **rebootOutsideOfActiveHours** True, if a reboot is scheduled outside of active hours. False, otherwise.
- **rebootScheduledByUser** True, if a reboot is scheduled by user. False, if a reboot is scheduled automatically.
- **revisionNumber** Revision number of the update that is getting installed with this reboot.
- **scheduledRebootTime** Time of the scheduled reboot
- **updateId** Update ID of the update that is getting installed with this reboot.
- **wuDeviceid** Unique device ID used by Windows Update.
- **rebootState** The state of the reboot.

Microsoft.Windows.Update.Orchestrator.Detection

This event sends launch data for a Windows Update scan to help keep Windows up to date.

The following fields are available:

- **deferReason** Reason why the device could not check for updates.
- **detectionBlockReason** Reason for detection not completing.
- **detectionDeferReason** A log of deferral reasons for every update state.
- **errorCode** The returned error code.
- **eventScenario** End to end update session ID, or indicates the purpose of sending this event - whether because the software distribution just started installing content, or whether it was cancelled, succeeded, or failed.
- **flightID** A unique update ID.
- **interactive** Identifies if session is User Initiated.
- **revisionNumber** Update revision number.
- **updateId** Update ID.
- **updateScenarioType** The update session type.
- **wuDeviceid** Unique device ID used by Windows Update.

Microsoft.Windows.Update.Orchestrator.InitiatingReboot

This event sends data about an Orchestrator requesting a reboot from power management to help keep Windows up to date.

The following fields are available:

- **EventPublishedTime** Time of the event.
- **revisionNumber** Revision number of the update.
- **updateId** Update ID.
- **wuDeviceid** Unique device ID used by Windows Update.
- **flightID** Unique update ID
- **interactive** Indicates the reboot initiation stage of the update process was entered as a result of user action or not.
- **rebootOutsideOfActiveHours** Indicates the timing that the reboot was to occur to ensure the correct update process and experience is provided to keep Windows up to date.
- **uxRebootstate** Indicates the exact state of the user experience at the time the required reboot was initiated to ensure the correct update process and experience is provided to keep Windows up to date.
- **updateScenarioType** The update session type.

Microsoft.Windows.Update.Ux.MusUpdateSettings.RebootScheduled

This event sends basic information for scheduling a device restart to install security updates. It's used to help keep Windows up-to-date.

The following fields are available:

- **activeHoursApplicable** Is the restart respecting Active Hours?
- **rebootArgument** The arguments that are passed to the OS for the restarted.
- **rebootOutsideOfActiveHours** Was the restart scheduled outside of Active Hours?
- **rebootScheduledByUser** Was the restart scheduled by the user? If the value is false, the restart was scheduled by the device.
- **rebootState** The state of the restart.
- **revisionNumber** The revision number of the OS being updated.
- **scheduledRebootTime** Time of the scheduled reboot
- **updateId** The Windows Update device GUID.
- **wuDeviceid** The Windows Update device GUID.
- **forcedReboot** True, if a reboot is forced on the device. Otherwise, this is False

Microsoft.Windows.Update.Ux.MusNotification.RebootNoLongerNeeded

This event is sent when a security update has successfully completed.

The following fields are available:

- **UtcTime** The Coordinated Universal Time that the restart was no longer needed.

Microsoft.Windows.Update.Ux.MusNotification.ToastDisplayedToScheduleReboot

This event is sent when a toast notification is shown to the user about scheduling a device restart.

The following fields are available:

- **UtcTime** The Coordinated Universal Time when the toast notification was shown.

Microsoft.Windows.Update.Orchestrator.RestoreRebootTask

This event sends data indicating that a reboot task is missing unexpectedly on a device and the task is restored because a reboot is still required, to help keep Windows up to date.

The following fields are available:

- **RebootTaskRestoredTime** Time at which this reboot task was restored.
- **revisionNumber** Update revision number.
- **updateId** Update ID.
- **wuDeviceid** Device id on which the reboot is restored

Microsoft.Windows.Update.Orchestrator.SystemNeeded

This event sends data about why a device is unable to reboot, to help keep Windows up to date.

The following fields are available:

- **eventScenario** End to end update session ID.
- **revisionNumber** Update revision number.
- **systemNeededReason** Reason ID
- **updateId** Update ID.
- **wuDeviceid** Unique device ID used by Windows Update.
- **rebootOutsideOfActiveHours** Indicates the timing that the reboot was to occur to ensure the correct update process and experience is provided to keep Windows up to date.
- **uxRebootstate** Indicates the exact state of the user experience at the time the required reboot was initiated to ensure the correct update process and experience is provided to keep Windows up to date.
- **updateScenarioType** The update session type.

Microsoft.Windows.Update.UpdateStackServicing.CheckForUpdates

This event sends data about the UpdateStackServicing check for updates, to help keep Windows up to date.

The following fields are available:

- **BspVersion** The version of the BSP.
- **CallerApplicationName** The name of the USS scheduled task. Example UssScheduled or UssBoot
- **ClientVersion** The version of the client.
- **CommercializationOperator** The name of the operator.
- **DetectionVersion** The string returned from the GetDetectionVersion export of the downloaded detection DLL.
- **DeviceName** The name of the device.
- **EventInstanceId** The USS session ID.
- **EventScenario** The scenario of the event. Example: Started, Failed, or Succeeded
- **OemName** The name of the manufacturer.
- **ServiceGuid** The GUID of the service.

- **StatusCode** The HRESULT code of the operation.
- **WUDeviceID** The Windows Update device ID.

Microsoft.Windows.Update.Orchestrator.CommitFailed

This event tracks when a device needs to restart after an update but did not.

The following fields are available:

- **errorCode** The error code that was returned.
- **wuDeviceid** The Windows Update device GUID.

Microsoft.Windows.Update.Orchestrator.Install

This event sends launch data for a Windows Update install to help keep Windows up to date.

The following fields are available:

- **batteryLevel** Current battery capacity in mWh or percentage left.
- **deferReason** Reason for install not completing.
- **eventScenario** End to end update session ID.
- **interactive** Identifies if session is user initiated.
- **wuDeviceid** Unique device ID used by Windows Update.
- **flightUpdate** Flight update
- **installRebootinitiatetime** The time it took for a reboot to be attempted.
- **minutesToCommit** The time it took to install updates.
- **revisionNumber** Update revision number.
- **updateId** Update ID.
- **errorCode** The error code represented by a hexadecimal value.
- **installCommitfailedtime** The time it took for a reboot to happen but the upgrade failed to progress.
- **flightID** Unique update ID
- **ForcedRebootReminderSet** A boolean value that indicates if a forced reboot will happen for updates.
- **rebootOutsideOfActiveHours** Indicates the timing that the reboot was to occur to ensure the correct update process and experience is provided to keep Windows up to date.
- **uxRebootstate** Indicates the exact state of the user experience at the time the required reboot was initiated to ensure the correct update process and experience is provided to keep Windows up to date.
- **updateScenarioType** The update session type.

Microsoft.Windows.Update.Orchestrator.PreShutdownStart

This event is generated right before the shutdown and commit operations

The following fields are available:

- **wuDeviceid** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue

Microsoft.Windows.Update.Orchestrator.DeferRestart

This event indicates that a restart required for installing updates was postponed

The following fields are available:

- **filteredDeferReason** Indicates the raised, but ignorable, reasons that the USO didn't restart (for example, user active or low battery)
- **raisedDeferReason** Indicates the reason that the USO didn't restart. For example, user active or low battery
- **wuDeviceid** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue

- **eventScenario** Indicates the purpose of sending this event - whether because the software distribution just started checking for content, or whether it was cancelled, succeeded, or failed

Microsoft.Windows.Update.Orchestrator.DisplayNeeded

Reboot postponed due to needing a display

The following fields are available:

- **displayNeededReason** Reason the display is needed
- **eventScenario** Indicates the purpose of sending this event - whether because the software distribution just started checking for content, or whether it was cancelled, succeeded, or failed
- **rebootOutsideOfActiveHours** Indicates the timing that the reboot was to occur to ensure the correct update process and experience is provided to keep Windows up to date
- **revisionNumber** Revision number of the update
- **updateId** Update ID
- **updateScenarioType** The update session type
- **uxRebootstate** Indicates the exact state of the user experience at the time the required reboot was initiated to ensure the correct update process and experience is provided to keep Windows up to date
- **wuDeviceid** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue

Microsoft.Windows.Update.NotificationUx.RebootScheduled

Indicates when a reboot is scheduled by the system or a user for a security, quality, or feature update

The following fields are available:

- **activeHoursApplicable** True, If Active Hours applicable on this device. False, otherwise
- **rebootArgument** Argument for the reboot task. It also represents specific reboot related action
- **rebootOutsideOfActiveHours** True, if a reboot is scheduled outside of active hours. False, otherwise
- **rebootScheduledByUser** True, if a reboot is scheduled by user. False, if a reboot is scheduled automatically
- **rebootState** The state of the reboot
- **revisionNumber** Revision number of the update that is getting installed with this reboot
- **scheduledRebootTime** Time of the scheduled reboot
- **updateId** ID of the update that is getting installed with this reboot
- **wuDeviceid** Unique device ID used by Windows Update
- **scheduledRebootTimeInUTC** Time of the scheduled reboot in Coordinated Universal Time

Windows 10, version 1709 enhanced telemetry events and fields used by Windows Analytics

10/17/2017 • 9 min to read • [Edit Online](#)

Applies to

- Windows 10, version 1709 and later

Windows Analytics Device Health reports are powered by diagnostic data not included in the Basic level. This includes crash reports and certain OS telemetry events. Organizations sending Enhanced or Full level diagnostic data were able to participate in Device Health, but some organizations which required detailed event and field level documentation were unable to move from Basic to Enhanced.

In Windows 10, version 1709, we introduce a new feature: "Limit Enhanced diagnostic data to the minimum required by Windows Analytics". When enabled, this feature limits the operating system telemetry events included in the Enhanced level to only those described below. Note that the Enhanced level also includes limited crash reports, which are not described below. For more information on the Enhanced level, see [Configure Windows telemetry in your organization](#).

KernelProcess.AppStateChangeSummary

This event summarizes application usage and performance characteristics to help Microsoft improve performance and reliability. Organizations can use this event with Windows Analytics to gain insights into application reliability.

The following fields are available:

- **CommitChargeAtExit_Sum:** Total memory commit charge for a process when it exits
- **CommitChargePeakAtExit_Sum:** Total peak memory commit charge for a process when it exits
- **ContainerId:** Server Silo Container ID
- **CrashCount:** Number of crashes for a process instance
- **CycleCountAtExit_Sum:** Total processor cycles for a process when it exited
- **ExtraInfoFlags:** Flags indicating internal states of the logging
- **GhostCount_Sum:** Total number of instances where the application stopped responding
- **HandleCountAtExit_Sum:** Total handle count for a process when it exits
- **HangCount_Max:** Maximum number of hangs detected
- **HangCount_Sum:** Total number of application hangs detected
- **HardFaultCountAtExit_Sum:** Total number of hard page faults detected for a process when it exits
- **HeartbeatCount:** Heartbeats logged for this summary
- **HeartbeatSuspendedCount:** Heartbeats logged for this summary where the process was suspended
- **LaunchCount:** Number of process instances started
- **LicenseType:** Reserved for future use
- **ProcessDurationMS_Sum:** Total duration of wall clock process instances
- **ReadCountAtExit_Sum:** Total IO reads for a process when it exited
- **ReadSizeInKBAtExit_Sum:** Total IO read size for a process when it exited
- **ResumeCount:** Number of times a process instance has resumed
- **RunningDurationMS_Sum:** Total uptime
- **SuspendCount:** Number of times a process instance was suspended

- **TargetAppId:** Application identifier
- **TargetAppType:** Application type
- **TargetAppVer:** Application version
- **TerminateCount:** Number of times a process terminated
- **WriteCountAtExit_Sum:** Total number of IO writes for a process when it exited
- **WriteSizeInKBAtExit_Sum:** Total size of IO writes for a process when it exited

Microsoft.OSG.OSS.CredProvFramework.ReportResultStop

This event indicates the result of an attempt to authenticate a user with a credential provider. It helps Microsoft to improve logon reliability. Using this event with Windows Analytics can help organizations monitor and improve logon success for different methods (for example, biometric) on managed devices.

The following fields are available:

- **CredTileProviderId:** ID of the Credential Provider
- **IsConnectedUser:** Flag indicating whether a user is connected or not
- **IsPLAPTile:** Flag indicating whether this credential tile is a pre-logon access provider or not
- **IsRemoteSession:** Flag indicating whether the session is remote or not
- **IsV2CredProv:** Flag indicating whether the credential provider of V2 or not
- **OpitonalStatusText:** Status text
- **ProcessImage:** Image path to the process
- **ProviderId:** Credential provider ID
- **ProviderStatusIcon:** Indicates which status icon should be displayed
- **ReturnCode:** Output of the ReportResult function
- **SessionId:** Session identifier
- **Sign-in error status:** The sign-in error status
- **SubStatus:** Sign-in error sub-status
- **UserTag:** Count of the number of times a user has selected a provider

Microsoft.Windows.Kernel.Power.OSStateChange

This event denotes the transition between operating system states (e.g., On, Off, Sleep, etc.). By using this event with Windows Analytics, organizations can use this to monitor reliability and performance of managed devices

The following fields are available:

- **AcPowerOnline:** If "TRUE," the device is using AC power. If "FALSE," the device is using battery power.
- **ActualTransitions:** The number of transitions between operating system states since the last system boot
- **BatteryCapacity:** Maximum battery capacity in mWh
- **BatteryCharge:** Current battery charge as a percentage of total capacity
- **BatteryDischarging:** Flag indicating whether the battery is discharging or charging
- **BootId:** Total boot count since the operating system was installed
- **BootTimeUTC:** Date and time of a particular boot event (identified by BootId)
- **EnergyChangeV2:** A snapshot value in mWh reflecting a change in power usage
- **EnergyChangeV2Flags:** Flags for disambiguating EnergyChangeV2 context
- **EventSequence:** A sequential number used to evaluate the completeness of the data
- **LastStateTransition:** ID of the last operating system state transition
- **LastStateTransitionSub:** ID of the last operating system sub-state transition
- **StateDurationMS:** Number of milliseconds spent in the last operating system state

- **StateTransition:** ID of the operating system state the system is transitioning to
- **StateTransitionSub:** ID of the operating system sub-state the system is transitioning to
- **TotalDurationMS:** Total time (in milliseconds) spent in all states since the last boot
- **TotalUptimeMS:** Total time (in milliseconds) the device was in Up or Running states since the last boot
- **TransitionsToOn:** Number of transitions to the Powered On state since the last boot
- **UptimeDeltaMS:** Total time (in milliseconds) added to Uptime since the last event

Microsoft.Windows.LogonController.LogonAndUnlockSubmit

Sends details of the user attempting to sign into or unlock the device.

The following fields are available:

- **isSystemManagedAccount:** Indicates if the user's account is System Managed
- **isUnlockScenario:** Flag indicating whether the event is a Logon or an Unlock
- **PartA_UserSid:** The security identifier of the user
- **userType:** Indicates the user type: 0 = unknown; 1 = local; 2 = Active Directory domain user; 3 = Microsoft Account; 4 = Azure Active Directory user

Microsoft.Windows.LogonController.SignInFailure

Sends details about any error codes detected during a failed sign-in.

The following fields are available:

- **ntsStatus:** The NTSTATUS error code status returned from an attempted sign-in
- **ntsSubstatus:** The NTSTATUS error code sub-status returned from an attempted sign-in

Microsoft.Windows.Security.Biometrics.Service.BioServiceActivityCapture

Indicates that a biometric capture was compared to known templates

The following fields are available:

- **captureDetail:** Result of biometric capture, either matched to an enrollment or an error
- **captureSuccessful:** Indicates whether a biometric capture was successfully matched or not
- **hardwareId:** ID of the sensor that collected the biometric capture
- **isSecureSensor:** Flag indicating whether a biometric sensor was in enhanced security mode
- **isTrustletRunning:** Indicates whether an enhanced security component is currently running
- **isVsmCfg:** Flag indicating whether virtual secure mode is configured or not

Microsoft.Windows.Security.Certificates.PinRulesCaCertUsedAnalytics

The Microsoft.Windows.Security.Certificates.Pin*Analytics events summarize which server certificates the client encounters. By using this event with Windows Analytics, organizations can use this to determine potential scope and impact of pending certificate revocations or expirations.

The following fields are available:

- **certBinary:** Binary blob of public certificate as presented to the client (does not include any private keys)
- **certThumbprint:** Certificate thumbprint

Microsoft.Windows.Security.Certificates.PinRulesCheckedAnalytics

The Microsoft.Windows.Security.Certificates.Pin*Analytics events summarize which server certificates the client encounters. By using this event with Windows Analytics, organizations can use this to determine potential scope and impact of pending certificate revocations or expirations.

The following fields are available:

- **caThumbprints:** Intermediate certificate thumbprints
- **rootThumbprint:** Root certificate thumbprint
- **serverName:** Server name associated with the certificate
- **serverThumbprint:** Server certificate thumbprint
- **statusBits:** Certificate status

Microsoft.Windows.Security.Certificates.PinRulesServerCertUsedAnalytics

The Microsoft.Windows.Security.Certificates.Pin*Analytics events summarize which server certificates the client encounters. By using this event with Windows Analytics, organizations can use this to determine potential scope and impact of pending certificate revocations or expirations.

The following fields are available:

- **certBinary:** Binary blob of public certificate as presented to the client (does not include any private keys)
- **certThumbprint:** Certificate thumbprint

Microsoft.Windows.Security.Winlogon.SystemBootStop

System boot has completed.

The following field is available:

- **ticksSinceBoot:** Duration of boot event (milliseconds)

Microsoft.Windows.Shell.Desktop.LogonFramework.AllLogonTasks

This event summarizes the logon procedure to help Microsoft improve performance and reliability. By using this event with Windows Analytics organizations can help identify logon problems on managed devices.

The following fields are available:

- **isAadUser:** Indicates whether the current logon is for an Azure Active Directory account
- **isDomainUser:** Indicates whether the current logon is for a domain account
- **isMSA:** Indicates whether the current logon is for a Microsoft Account
- **logonOptimizationFlags:** Flags indicating optimization settings for this logon session
- **logonTypeFlags:** Flags indicating logon type (first logon vs. a later logon)
- **systemManufacturer:** Device manufacturer
- **systemProductName:** Device product name
- **wilActivity:** Indicates errors in the task to help Microsoft improve reliability.

Microsoft.Windows.Shell.Desktop.LogonFramework.LogonTask

This event describes system tasks which are part of the user logon sequence and helps Microsoft to improve reliability.

The following fields are available:

- **isStartWaitTask:** Flag indicating whether the task starts a background task
- **isWaitMethod:** Flag indicating the task is waiting on a background task
- **logonTask:** Indicates which logon step is currently occurring
- **wilActivity:** Indicates errors in the task to help Microsoft improve reliability.

Microsoft.Windows.Shell.Explorer.DesktopReady

Initialization of Explorer is complete.

Microsoft-Windows-Security-EFS-EDPAudit-ApplicationLearning.EdpAuditLogApplicationLearning

For a device subject to Windows Information Protection policy, learning events are generated when an app encounters a policy boundary (for example, trying to open a work document from a personal app). These events help the WIP administrator tune policy rules and prevent unnecessary user disruption.

The following fields are available:

- **actiontype:** Indicates what type of resource access the app was attempting (for example, opening a local document vs. a network resource) when it encountered a policy boundary. Useful for Windows Information Protection administrators to tune policy rules.
- **appIdType:** Based on the type of application, this indicates what type of app rule a Windows Information Protection administrator would need to create for this app.
- **appname:** App that triggered the event
- **status:** Indicates whether errors occurred during WIP learning events

Win32kTraceLogging.AppInteractivitySummary

Summarizes which app windows are being used (for example, have focus) to help Microsoft improve compatibility and user experience. Also helps organizations (by using Windows Analytics) to understand and improve application reliability on managed devices.

The following fields are available:

- **AggregationDurationMS:** Actual duration of aggregation period (in milliseconds)
- **AggregationFlags:** Flags denoting aggregation settings
- **AggregationPeriodMS:** Intended duration of aggregation period (in milliseconds)
- **AggregationStartTime:** Start date and time of AppInteractivity aggregation
- **AppId:** Application ID for usage
- **AppSessionId:** GUID identifying the application's usage session
- **AppVersion:** Version of the application that produced this event
- **AudioInMS:** Audio capture duration (in milliseconds)
- **AudioOutMS:** Audio playback duration (in milliseconds)
- **BackgroundMouseSec:** Indicates that there was a mouse hover event while the app was in the background
- **BitPeriodMS:** Length of the period represented by InFocusBitmap
- **CommandLineHash:** A hash of the command line
- **CompositionDirtyGeneratedSec:** Represents the amount of time (in seconds) during which the active app reported that it had an update
- **CompositionDirtyPropagatedSec:** Total time (in seconds) that a separate process with visuals hosted in an app signaled updates
- **CompositionRenderedSec:** Time (in seconds) that an app's contents were rendered

- **EventSequence:** [need more info]
- **FocusLostCount:** Number of times that an app lost focus during the aggregation period
- **GameInputSec:** Time (in seconds) there was user input using a game controller
- **HidInputSec:** Time (in seconds) there was user input using devices other than a game controller
- **InFocusBitmap:** Series of bits representing application having and losing focus
- **InFocusDurationMS:** Total time (in milliseconds) the application had focus
- **InputSec:** Total number of seconds during which there was any user input
- **InteractiveTimeoutPeriodMS:** Total time (in milliseconds) that inactivity expired interactivity sessions
- **KeyboardInputSec:** Total number of seconds during which there was keyboard input
- **MonitorFlags:** Flags indicating app use of individual monitor(s)
- **MonitorHeight:** Number of vertical pixels in the application host monitor resolution
- **MonitorWidth:** Number of horizontal pixels in the application host monitor resolution
- **MouseInputSec:** Total number of seconds during which there was mouse input
- **NewProcessCount:** Number of new processes contributing to the aggregate
- **PartATransform_AppSessionGuidToUserId:** Flag which influences how other parts of the event are constructed
- **PenInputSec:** Total number of seconds during which there was pen input
- **SpeechRecognitionSec:** Total number of seconds of speech recognition
- **SummaryRound:** Incrementing number indicating the round (batch) being summarized
- **TargetAsId:** Flag which influences how other parts of the event are constructed
- **TotalUserOrDisplayActiveDurationMS:** Total time the user or the display was active (in milliseconds)
- **TouchInputSec:** Total number of seconds during which there was touch input
- **UserActiveDurationMS:** Total time that the user was active including all input methods
- **UserActiveTransitionCount:** Number of transitions in and out of user activity
- **UserOrDisplayActiveDurationMS:** Total time the user was using the display
- **ViewFlags:** Flags denoting properties of an app view (for example, special VR view or not)
- **WindowFlags:** Flags denoting runtime properties of an app window
- **WindowHeight:** Number of vertical pixels in the application window
- **WindowWidth:** Number of horizontal pixels in the application window

Windows 10, version 1703 basic level Windows diagnostic events and fields

7/28/2017 • 151 min to read • [Edit Online](#)

Applies to

- Windows 10, version 1703 and later

The Basic level gathers a limited set of information that is critical for understanding the device and its configuration including: basic device information, quality-related information, app compatibility, and Microsoft Store. When the level is set to Basic, it also includes the Security level information. The Basic level also helps to identify problems that can occur on a particular device hardware or software configuration. For example, it can help determine if crashes are more frequent on devices with a specific amount of memory or that are running a particular driver version. This helps Microsoft fix operating system or app problems.

Use this article to learn about diagnostic events, grouped by event area, and the fields within each event. A brief description is provided for each field. Every event generated includes common data, which collects device data. You can learn more about Windows functional and diagnostic data through these articles:

- [Manage connections from Windows operating system components to Microsoft services](#)
- [Configure Windows telemetry in your organization](#)

NOTE

Updated July 2017 to document new and modified events. We've added new fields to several Appraiser events to prepare for upgrades to the next release of Windows and we've added a brand-new event, Census.Speech, to collect basic details about speech settings and configuration.

Common data extensions

Common Data Extensions.App

The following fields are available:

- **expId** Associates a flight, such as an OS flight, or an experiment, such as a web site UX experiment, with an event.
- **userId** The userID as known by the application.
- **env** The environment from which the event was logged.
- **asId** An integer value that represents the app session. This value starts at 0 on the first app launch and increments after each subsequent app launch per boot session.

Common Data Extensions.CS

The following fields are available:

- **sig** A common schema signature that identifies new and modified event schemas.

Common Data Extensions.CUET

The following fields are available:

- **stId** Represents the Scenario Entry Point ID. This is a unique GUID for each event in a diagnostic scenario. This used to be Scenario Trigger ID.

- **aid** Represents the ETW ActivityId. Logged via TraceLogging or directly via ETW.
- **raId** Represents the ETW Related ActivityId. Logged via TraceLogging or directly via ETW.
- **op** Represents the ETW Op Code.
- **cat** Represents a bitmask of the ETW Keywords associated with the event.
- **flags** Represents the bitmap that captures various Windows specific flags.
- **cplId** The composer ID, such as Reference, Desktop, Phone, Holographic, Hub, IoT Composer.
- **tickets** A list of strings that represent entries in the HTTP header of the web request that includes this event.
- **bseq** Upload buffer sequence number in the format <buffer identifier>:<sequence number>
- **mon** Combined monitor and event sequence numbers in the format <monitor sequence>:<event sequence>

Common Data Extensions.Device

The following fields are available:

- **ver** Represents the major and minor version of the extension.
- **localId** Represents a locally defined unique ID for the device, not the human readable device name. Most likely equal to the value stored at HKLM\Software\Microsoft\SQMClient\MachineId
- **deviceClass** Represents the classification of the device, the device "family". For example, Desktop, Server, or Mobile.

Common Data Extensions.Envelope

The following fields are available:

- **ver** Represents the major and minor version of the extension.
- **name** Represents the uniquely qualified name for the event.
- **time** Represents the event date time in Coordinated Universal Time (UTC) when the event was generated on the client. This should be in ISO 8601 format.
- **popSample** Represents the effective sample rate for this event at the time it was generated by a client.
- **epoch** Represents the epoch and seqNum fields, which help track how many events were fired and how many events were uploaded, and enables identification of data lost during upload and de-duplication of events on the ingress server.
- **seqNum** Represents the sequence field used to track absolute order of uploaded events. It is an incrementing identifier for each event added to the upload queue. The Sequence helps track how many events were fired and how many events were uploaded and enables identification of data lost during upload and de-duplication of events on the ingress server.
- **iKey** Represents an ID for applications or other logical groupings of events.
- **flags** Represents a collection of bits that describe how the event should be processed by the Connected User Experience and Telemetry component pipeline. The lowest-order byte is the event persistence. The next byte is the event latency.
- **os** Represents the operating system name.
- **osVer** Represents the OS version, and its format is OS dependent.
- **appId** Represents a unique identifier of the client application currently loaded in the process producing the event; and is used to group events together and understand usage pattern, errors by application.
- **appVer** Represents the version number of the application. Used to understand errors by Version, Usage by Version across an app.
- **cV** Represents the Correlation Vector: A single field for tracking partial order of related telemetry events across component boundaries.

Common Data Extensions.OS

The following fields are available:

- **ver** Represents the major and minor version of the extension.

- **expId** Represents the experiment ID. The standard for associating a flight, such as an OS flight (pre-release build), or an experiment, such as a web site UX experiment, with an event is to record the flight / experiment IDs in Part A of the common schema.
- **locale** Represents the locale of the operating system.
- **bootId** An integer value that represents the boot session. This value starts at 0 on first boot after OS install and increments after every reboot.

Common Data Extensions.User

The following fields are available:

- **ver** Represents the major and minor version of the extension.
- **localId** Represents a unique user identity that is created locally and added by the client. This is not the user's account ID.

Common Data Extensions.XBL

The following fields are available:

- **nbf** Not before time
- **expId** Expiration time
- **sbx** XBOX sandbox identifier
- **dty** XBOX device type
- **did** XBOX device ID
- **xid** A list of base10-encoded XBOX User IDs.
- **uts** A bit field, with 2 bits being assigned to each user ID listed in xid. This field is omitted if all users are retail accounts.

Common Data Extensions.Consent UI Event

This User Account Control (UAC) telemetry point collects information on elevations that originate from low integrity levels. This occurs when a process running at low integrity level (IL) requires higher (administrator) privileges, and therefore requests for elevation via UAC (consent.exe). By better understanding the processes requesting these elevations, Microsoft can in turn improve the detection and handling of potentially malicious behavior in this path.

The following fields are available:

- **eventType** Represents the type of elevation: If it succeeded, was cancelled, or was auto-approved.
- **splitToken** Represents the flag used to distinguish between administrators and standard users.
- **friendlyName** Represents the name of the file requesting elevation from low IL.
- **elevationReason** Represents the distinction between various elevation requests sources (appcompat, installer, COM, MSI and so on).
- **exeName** Represents the name of the file requesting elevation from low IL.
- **signatureState** Represents the state of the signature, if it signed, unsigned, OS signed and so on.
- **publisherName** Represents the name of the publisher of the file requesting elevation from low IL.
- **cmdLine** Represents the full command line arguments being used to elevate.
- **Hash.Length** Represents the length of the hash of the file requesting elevation from low IL.
- **Hash** Represents the hash of the file requesting elevation from low IL.
- **HashAlgId** Represents the algorithm ID of the hash of the file requesting elevation from low IL.
- **telemetryFlags** Represents the details about the elevation prompt for CEIP data.
- **timeStamp** Represents the time stamp on the file requesting elevation.
- **fileVersionMS** Represents the major version of the file requesting elevation.
- **fileVersionLS** Represents the minor version of the file requesting elevation.

Common data fields

Common Data Fields.MS.Device.DeviceInventory.Change

These fields are added whenever Ms.Device.DeviceInventoryChange is included in the event.

The following fields are available:

- **syncId** A string used to group StartSync, EndSync, Add, and Remove operations that belong together. This field is unique by Sync period and is used to disambiguate in situations where multiple agents perform overlapping inventories for the same object.
- **objectType** Indicates the object type that the event applies to.
- **Action** The change that was invoked on a device inventory object.
- **inventoryId** Device ID used for Compatibility testing

Common Data Fields.TelClientSynthetic.PrivacySettingsAfterCreatorsUpdate.PreUpgradeSettings

These fields are added whenever PreUpgradeSettings is included in the event.

The following fields are available:

- **HKLM_SensorPermissionState.SensorPermissionState** The state of the Location service before the feature update completed.
- **HKLM_SensorPermissionState.HRESULT** The error code returned when trying to query the Location service for the device.
- **HKCU_SensorPermissionState.SensorPermissionState** The state of the Location service when a user signs on before the feature update completed.
- **HKCU_SensorPermissionState.HRESULT** The error code returned when trying to query the Location service for the current user.
- **HKLM_LocationPlatform.Status** The state of the location platform after the feature update has completed.
- **HKLM_LocationPlatform.HRESULT** The error code returned when trying to query the location platform for the device.
- **HKLM_LocationSyncEnabled.AcceptedPrivacyPolicy** The speech recognition state for the device before the feature update completed.
- **HKLM_LocationSyncEnabled.HRESULT** The error code returned when trying to query the Find My Device service for the device.
- **HKCU_LocationSyncEnabled.AcceptedPrivacyPolicy** The speech recognition state for the current user before the feature update completed.
- **HKCU_LocationSyncEnabled.HRESULT** The error code returned when trying to query the Find My Device service for the current user.
- **HKLM_AllowTelemetry.AllowTelemetry** The state of the Connected User Experiences and Telemetry component for the device before the feature update.
- **HKLM_AllowTelemetry.HRESULT** The error code returned when trying to query the Connected User Experiences and Telemetry component for the device.
- **HKLM_TIPC.Enabled** The state of TIPC for the device.
- **HKLM_TIPC.HRESULT** The error code returned when trying to query TIPC for the device.
- **HKCU_TIPC.Enabled** The state of TIPC for the current user.
- **HKCU_TIPC.HRESULT** The error code returned when trying to query TIPC for the current user.
- **HKLM_FlipAhead.FPEnabled** Is Flip Ahead enabled for the device before the feature update was completed?
- **HKLM_FlipAhead.HRESULT** The error code returned when trying to query Flip Ahead for the device.
- **HKCU_FlipAhead.FPEnabled** Is Flip Ahead enabled for the current user before the feature update was completed?
- **HKCU_FlipAhead.HRESULT** The error code returned when trying to query Flip Ahead for the current user.

- **HKLM_TailoredExperiences.TailoredExperiencesWithDiagnosticDataEnabled** Is Tailored Experiences with Diagnostics Data enabled for the current user after the feature update had completed?
- **HKCU_TailoredExperiences.HRESULT** The error code returned when trying to query Tailored Experiences with Diagnostics Data for the current user.
- **HKLM_AdvertisingID.Enabled** Is the advertising ID enabled for the device?
- **HKLM_AdvertisingID.HRESULT** The error code returned when trying to query the state of the advertising ID for the device.
- **HKCU_AdvertisingID.Enabled** Is the advertising ID enabled for the current user?
- **HKCU_AdvertisingID.HRESULT** The error code returned when trying to query the state of the advertising ID for the user.

Common Data Fields.TelClientSynthetic.PrivacySettingsAfterCreatorsUpdate.PostUpgradeSettings

These fields are added whenever PostUpgradeSettings is included in the event.

The following fields are available:

- **HKLM_SensorPermissionState.SensorPermissionState** The state of the Location service after the feature update has completed.
- **HKLM_SensorPermissionState.HRESULT** The error code returned when trying to query the Location service for the device.
- **HKCU_SensorPermissionState.SensorPermissionState** The state of the Location service when a user signs on after a feature update has completed.
- **HKCU_SensorPermissionState.HRESULT** The error code returned when trying to query the Location service for the current user.
- **HKLM_LocationPlatform.Status** The state of the location platform after the feature update has completed.
- **HKLM_LocationPlatform.HRESULT** The error code returned when trying to query the location platform for the device.
- **HKLM_LocationSyncEnabled.AcceptedPrivacyPolicy** The speech recognition state for the device after the feature update has completed.
- **HKLM_LocationSyncEnabled.HRESULT** The error code returned when trying to query the Find My Device service for the device.
- **HKCU_LocationSyncEnabled.AcceptedPrivacyPolicy** The speech recognition state for the current user after the feature update has completed.
- **HKCU_LocationSyncEnabled.HRESULT** The error code returned when trying to query the Find My Device service for the current user.
- **HKLM_AllowTelemetry.AllowTelemetry** The state of the Connected User Experiences and Telemetry component for the device after the feature update.
- **HKLM_AllowTelemetry.HRESULT** The error code returned when trying to query the Connected User Experiences and Telemetry component for the device.
- **HKLM_TIPC.Enabled** The state of TIPC for the device.
- **HKLM_TIPC.HRESULT** The error code returned when trying to query TIPC for the device.
- **HKCU_TIPC.Enabled** The state of TIPC for the current user.
- **HKCU_TIPC.HRESULT** The error code returned when trying to query TIPC for the current user.
- **HKLM_FlipAhead.FPEnabled** Is Flip Ahead enabled for the device after the feature update has completed?
- **HKLM_FlipAhead.HRESULT** The error code returned when trying to query Flip Ahead for the device.
- **HKCU_FlipAhead.FPEnabled** Is Flip Ahead enabled for the current user after the feature update has completed?
- **HKCU_FlipAhead.HRESULT** The error code returned when trying to query Flip Ahead for the current user.
- **HKLM_TailoredExperiences.TailoredExperiencesWithDiagnosticDataEnabled** Is Tailored Experiences with Diagnostics Data enabled for the current user after the feature update had completed?

- **HKCU_TailoredExperiences.HRESULT** The error code returned when trying to query Tailored Experiences with Diagnostics Data for the current user.
- **HKLM_AdvertisingID.Enabled** Is the advertising ID enabled for the device?
- **HKLM_AdvertisingID.HRESULT** The error code returned when trying to query the state of the advertising ID for the device.
- **HKCU_AdvertisingID.Enabled** Is the advertising ID enabled for the current user?
- **HKCU_AdvertisingID.HRESULT** The error code returned when trying to query the state of the advertising ID for the user.

Appraiser events

Microsoft.Windows.Appraiser.General.ChecksumTotalPictureCount

This event lists the types of objects and how many of each exist on the client device. This allows for a quick way to ensure that the records present on the server match what is present on the client.

The following fields are available:

- **DatasourceApplicationFile_RS3** The total DecisionApplicationFile objects targeting the next release of Windows on this device. on this device.
- **DatasourceDevicePnp_RS3** The total DatasourceDevicePnp objects targeting the next release of Windows on this device.
- **DatasourceDriverPackage_RS3** The total DatasourceDriverPackage objects targeting the next release of Windows on this device.
- **DataSourceMatchingInfoBlock_RS3** The total DataSourceMatchingInfoBlock objects targeting the next release of Windows on this device.
- **DataSourceMatchingInfoPassive_RS3** The total DataSourceMatchingInfoPassive objects targeting the next release of Windows on this device.
- **DataSourceMatchingInfoPostUpgrade_RS3** The total DataSourceMatchingInfoPostUpgrade objects targeting the next release of Windows on this device.
- **DatasourceSystemBios_RS3** The total DatasourceSystemBios objects targeting the next release of Windows on this device.
- **DecisionApplicationFile_RS3** The total DecisionApplicationFile objects targeting the next release of Windows on this device.
- **DecisionDevicePnp_RS3** The total DecisionDevicePnp objects targeting the next release of Windows on this device.
- **DecisionDriverPackage_RS3** The total DecisionDriverPackage objects targeting the next release of Windows on this device.
- **DecisionMatchingInfoBlock_RS3** The total DecisionMatchingInfoBlock objects targeting the next release of Windows on this device.
- **DecisionMatchingInfoPassive_RS3** The total DataSourceMatchingInfoPostUpgrade objects targeting the next release of Windows on this device.
- **DecisionMatchingInfoPostUpgrade_RS3** The total DecisionMatchingInfoPostUpgrade objects targeting the next release of Windows on this device.
- **DecisionMediaCenter_RS3** The total DecisionMediaCenter objects targeting the next release of Windows on this device.
- **DecisionSystemBios_RS3** The total DecisionSystemBios objects targeting the next release of Windows on this device.
- **PCFP** An ID for the system that is calculated by hashing hardware identifiers.
- **InventoryApplicationFile** The total InventoryApplicationFile objects that are present on this device.
- **InventoryMediaCenter** The total InventoryMediaCenter objects that are present on this device.

- **InventoryLanguagePack** The total InventoryLanguagePack objects that are present on this device.
- **InventoryUplevelDriverPackage** The total InventoryUplevelDriverPackage objects that are present on this device.
- **InventorySystemBios** The total InventorySystemBios objects that are present on this device.
- **SystemProcessorCompareExchange** The total SystemProcessorCompareExchange objects that are present on this device.
- **SystemProcessorLahfSahf** The total SystemProcessorLahfSahf objects that are present on this device.
- **SystemMemory** The total SystemMemory objects that are present on this device.
- **SystemProcessorPrefetchW** The total SystemProcessorPrefetchW objects that are present on this device.
- **SystemProcessorSse2** The total SystemProcessorSse2 objects that are present on this device.
- **SystemProcessorNx** The total SystemProcessorNx objects that are present on this device.
- **SystemWlan** The total SystemWlan objects that are present on this device.
- **SystemWim** The total SystemWim objects that are present on this device
- **SystemTouch** The total SystemTouch objects that are present on this device.
- **SystemWindowsActivationStatus** The total SystemWindowsActivationStatus objects that are present on this device.
- **Wmdrm_RS3** The total Wmdrm objects targeting the next release of Windows on this device.

Microsoft.Windows.Appraiser.General.ChecksumTotalPictureIdHashSha256

This event lists the types of objects and the hashed values of all the identifiers for each one. This allows for a more in-depth way to ensure that the records present on the server match what is present on the client.

The following fields are available:

- **DatasourceApplicationFile_RS3** The total DatasourceApplicationFile objects targeting the next release of Windows on this device.
- **DatasourceDevicePnp_RS3** The total DatasourceDevicePnp objects targeting the next release of Windows on this device.
- **DatasourceDriverPackage_RS3** The total DatasourceDriverPackage objects targeting the next release of Windows on this device.
- **DataSourceMatchingInfoBlock_RS3** The total DataSourceMatchingInfoBlock objects targeting the next release of Windows on this device.
- **DataSourceMatchingInfoPassive_RS3** The total DataSourceMatchingInfoPassive objects targeting the next release of Windows on this device.
- **DataSourceMatchingInfoPostUpgrade_RS3** The total DataSourceMatchingInfoPostUpgrade objects targeting the next release of Windows on this device.
- **DatasourceSystemBios_RS3** The total DatasourceSystemBios objects targeting the next release of Windows on this device.
- **DecisionApplicationFile_RS3** The total DecisionApplicationFile objects targeting the next release of Windows on this device.
- **DecisionDevicePnp_RS3** The total DecisionDevicePnp objects targeting the next release of Windows on this device.
- **DecisionDriverPackage_RS3** The total DecisionDriverPackage objects targeting the next release of Windows on this device.
- **DecisionMatchingInfoBlock_RS3** The total DecisionMatchingInfoBlock objects targeting the next release of Windows on this device.
- **DecisionMatchingInfoPassive_RS3** The total DataSourceMatchingInfoPostUpgrade objects targeting the next release of Windows on this device.
- **DecisionMatchingInfoPostUpgrade_RS3** The total DecisionMatchingInfoPostUpgrade objects targeting the next release of Windows on this device.

- **DecisionMediaCenter_RS3** The total DecisionMediaCenter objects targeting the next release of Windows on this device.
- **DecisionSystemBios_RS3** The total DecisionSystemBios objects targeting the next release of Windows on this device.
- **PCFP** An ID for the system that is calculated by hashing hardware identifiers.
- **InventoryApplicationFile** The SHA256 hash of InventoryApplicationFile objects that are present on this device.
- **InventoryMediaCenter** The SHA256 hash of InventoryMediaCenter objects that are present on this device.
- **InventoryLanguagePack** The SHA256 hash of InventoryLanguagePack objects that are present on this device.
- **InventoryUplevelDriverPackage** The SHA256 hash of InventoryUplevelDriverPackage objects that are present on this device.
- **InventorySystemBios** The SHA256 hash of InventorySystemBios objects that are present on this device.
- **SystemProcessorCompareExchange** The SHA256 hash of SystemProcessorCompareExchange objects that are present on this device.
- **SystemProcessorLahfSahf** The SHA256 hash of SystemProcessorLahfSahf objects that are present on this device.
- **SystemMemory** The SHA256 hash of SystemMemory objects that are present on this device.
- **SystemProcessorPrefetchW** The SHA256 hash of SystemProcessorPrefetchW objects that are present on this device.
- **SystemProcessorSse2** The SHA256 hash of SystemProcessorSse2 objects that are present on this device.
- **SystemProcessorNx** The SHA256 hash of SystemProcessorNx objects that are present on this device.
- **SystemWlan** The SHA256 hash of SystemWlan objects that are present on this device.
- **SystemWim** The SHA256 hash of SystemWim objects that are present on this device.
- **SystemTouch** The SHA256 hash of SystemTouch objects that are present on this device.
- **SystemWindowsActivationStatus** The SHA256 hash of SystemWindowsActivationStatus objects that are present on this device.
- **Wmdrm_RS3** The total Wmdrm objects targeting the next release of Windows on this device.

Microsoft.Windows.Appraiser.General.DatasourceApplicationFileAdd

This event sends compatibility information about a file to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file that is generating the events.
- **AvDisplayName** If it is an anti-virus app, this is its display name.
- **CompatModelIndex** The compatibility prediction for this file.
- **HasCitData** Is the file present in CIT data?
- **HasUpgradeExe** Does the anti-virus app have an upgrade.exe file?
- **IsAv** Is the file an anti-virus reporting EXE?
- **ResolveAttempted** This will always be an empty string when sending telemetry.
- **SdbEntries** An array of fields that indicates the SDB entries that apply to this file.

Microsoft.Windows.Appraiser.General.DatasourceApplicationFileRemove

This event indicates that the DatasourceApplicationFile object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DatasourceApplicationFileStartSync

This event indicates that a new set of DatasourceApplicationFileAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DatasourceDevicePnpAdd

This event sends compatibility data for a PNP device, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.
- **ActiveNetworkConnection** Is the device an active network device?
- **IsBootCritical** Is the device boot critical?
- **SdbEntries** An array of fields indicating the SDB entries that apply to this device.
- **WuDriverCoverage** Is there a driver uplevel for this device according to Windows Update?
- **WuDriverUpdateID** The Windows Update ID of the applicable uplevel driver.
- **WuPopulatedFromID** The expected uplevel driver matching ID based on driver coverage from Windows Update.

Microsoft.Windows.Appraiser.General.DatasourceDevicePnpRemove

This event indicates that the DatasourceDevicePnp object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DatasourceDevicePnpStartSync

This event indicates that a new set of DatasourceDevicePnpAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DatasourceDriverPackageAdd

This event sends compatibility database data about driver packages to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.
- **SdbEntries** An array of fields indicating the SDB entries that apply to this driver package.

Microsoft.Windows.Appraiser.General.DatasourceDriverPackageRemove

This event indicates that the DatasourceDriverPackage object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DatasourceDriverPackageStartSync

This event indicates that a new set of DatasourceDriverPackageAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoBlockAdd

This event sends blocking data about any compatibility blocking entries hit on the system that are not directly related to specific applications or devices, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.
- **SdbEntries** An array of fields indicating the SDB entries that apply to this file.

Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoBlockRemove

This event indicates that the DataSourceMatchingInfoBlock object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoBlockStartSync

This event indicates that a full set of DataSourceMatchingInfoBlockStAdd events have been sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPassiveAdd

This event sends compatibility database information about non-blocking compatibility entries on the system that are not keyed by either applications or devices, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.
- **SdbEntries** An array of fields indicating the SDB entries that apply to this file.

Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPassiveRemove

This event indicates that the DataSourceMatchingInfoPassive object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPassiveStartSync

This event indicates that a new set of DataSourceMatchingInfoPassiveAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPostUpgradeAdd

This event sends compatibility database information about entries requiring reinstallation after an upgrade on the system that are not keyed by either applications or devices, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.
- **SdbEntries** An array of fields indicating the SDB entries that apply to this file.

Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPostUpgradeRemove

This event indicates that the DataSourceMatchingInfoPostUpgrade object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DataSourceMatchingInfoPostUpgradeStartSync

This event indicates that a new set of DataSourceMatchingInfoPostUpgradeAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DatasourceSystemBiosAdd

This event sends compatibility database information about the BIOS to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **SdbEntries** An array of fields indicating the SDB entries that apply to this BIOS.

Microsoft.Windows.Appraiser.General.DatasourceSystemBiosRemove

This event indicates that the DatasourceSystemBios object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DatasourceSystemBiosStartSync

This event indicates that a new set of DatasourceSystemBiosAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionApplicationFileAdd

This event sends compatibility decision data about a file to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.
- **BlockAlreadyInbox** The uplevel runtime block on the file already existed on the current OS.
- **BlockingApplication** Are there any application issues that interfere with upgrade due to the file in question?
- **DisplayGenericMessage** Will be a generic message be shown for this file?
- **HardBlock** This file is blocked in the SDB.
- **HasUxBlockOverride** Does the file have a block that is overridden by a tag in the SDB?
- **MigApplication** Does the file have a MigXML from the SDB associated with it that applies to the current upgrade mode?
- **MigRemoval** Does the file have a MigXML from the SDB that will cause the app to be removed on upgrade?
- **NeedsDismissAction** Will the file cause an action that can be dismissed?
- **NeedsInstallPostUpgradeData** After upgrade, the file will have a post-upgrade notification to install a replacement for the app.
- **NeedsNotifyPostUpgradeData** Does the file have a notification that should be shown after upgrade?
- **NeedsReinstallPostUpgradeData** After upgrade, this file will have a post-upgrade notification to reinstall the app.
- **NeedsUninstallAction** The file must be uninstalled to complete the upgrade.
- **SdbBlockUpgrade** The file is tagged as blocking upgrade in the SDB,
- **SdbBlockUpgradeCanReinstall** The file is tagged as blocking upgrade in the SDB. It can be reinstalled after upgrade.
- **SdbBlockUpgradeUntilUpdate** The file is tagged as blocking upgrade in the SDB. If the app is updated, the upgrade can proceed.
- **SdbReinstallUpgrade** The file is tagged as needing to be reinstalled after upgrade in the SDB. It does not block upgrade.
- **SdbReinstallUpgradeWarn** The file is tagged as needing to be reinstalled after upgrade with a warning in the SDB. It does not block upgrade.

- **SoftBlock** The file is softblocked in the SDB and has a warning.

Microsoft.Windows.Appraiser.General.DecisionApplicationFileRemove

This event indicates Indicates that the DecisionApplicationFile object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionApplicationFileStartSync

This event indicates that a new set of DecisionApplicationFileAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionDevicePnpAdd

This event sends compatibility decision data about a PNP device to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.
- **AssociatedDriverIsBlocked** Is the driver associated with this PNP device blocked?
- **BlockAssociatedDriver** Should the driver associated with this PNP device be blocked?
- **BlockUpgradelfDriverBlocked** Is the PNP device both boot critical and does not have a driver included with the OS?
- **BlockUpgradelfDriverBlockedAndOnlyActiveNetwork** Is this PNP device the only active network device?
- **BlockingDevice** Is this PNP device blocking upgrade?
- **DisplayGenericMessage** Will a generic message be shown during Setup for this PNP device?
- **DriverAvailableInbox** Is a driver included with the operating system for this PNP device?
- **DriverAvailableOnline** Is there a driver for this PNP device on Windows Update?
- **DriverAvailableUplevel** Is there a driver on Windows Update or included with the operating system for this PNP device?
- **DriverBlockOverridden** Is there is a driver block on the device that has been overridden?
- **NeedsDismissAction** Will the user would need to dismiss a warning during Setup for this device?
- **NotRegressed** Does the device have a problem code on the source OS that is no better than the one it would have on the target OS?
- **SdbDeviceBlockUpgrade** Is there an SDB block on the PNP device that blocks upgrade?
- **SdbDriverBlockOverridden** Is there an SDB block on the PNP device that blocks upgrade, but that block was overridden?

Microsoft.Windows.Appraiser.General.DecisionDevicePnpRemove

This event indicates that the DecisionDevicePnp object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionDevicePnpStartSync

This event indicates that the DecisionDevicePnp object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionDriverPackageAdd

This event sends decision data about driver package compatibility to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.
- **DriverBlockOverridden** Does the driver package have an SDB block that blocks it from migrating, but that block has been overridden?
- **DriverIsDeviceBlocked** Was the driver package was blocked because of a device block?
- **DriverIsDriverBlocked** Is the driver package blocked because of a driver block?
- **DriverShouldNotMigrate** Should the driver package be migrated during upgrade?
- **SdbDriverBlockOverridden** Does the driver package have an SDB block that blocks it from migrating, but that block has been overridden?

Microsoft.Windows.Appraiser.General.DecisionDriverPackageRemove

This event indicates that the DecisionDriverPackage object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionDriverPackageStartSync

This event indicates that a new set of DecisionDriverPackageAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionMatchingInfoBlockAdd

This event sends compatibility decision data about blocking entries on the system that are not keyed by either applications or devices, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the appraiser file generating the events.
- **BlockingApplication** Are there are any application issues that interfere with upgrade due to matching info blocks?
- **DisplayGenericMessage** Will a generic message be shown for this block?
- **NeedsUninstallAction** Does the user need to take an action in setup due to a matching info block?
- **SdbBlockUpgrade** Is a matching info block blocking upgrade?
- **SdbBlockUpgradeCanReinstall** Is a matching info block blocking upgrade, but has the can reinstall tag?
- **SdbBlockUpgradeUntilUpdate** Is a matching info block blocking upgrade but has the until update tag?

Microsoft.Windows.Appraiser.General.DecisionMatchingInfoBlockRemove

This event indicates that the DecisionMatchingInfoBlock object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionMatchingInfoBlockStartSync

This event indicates that a new set of DecisionMatchingInfoBlockAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionMatchingInfoPassiveAdd

This event sends compatibility decision data about non-blocking entries on the system that are not keyed by either applications or devices, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **BlockingApplication** Are there any application issues that interfere with upgrade due to matching info blocks?
- **MigApplication** Is there a matching info block with a mig for the current mode of upgrade?

Microsoft.Windows.Appraiser.General.DecisionMatchingInfoPassiveRemove

This event Indicates that the DecisionMatchingInfoPassive object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionMatchingInfoPassiveStartSync

This event indicates that a new set of DecisionMatchingInfoPassiveAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionMatchingInfoPostUpgradeAdd

This event sends compatibility decision data about entries that require reinstall after upgrade. It's used to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **NeedsInstallPostUpgradeData** Will the file have a notification after upgrade to install a replacement for the app?
- **NeedsNotifyPostUpgradeData** Should a notification be shown for this file after upgrade?
- **NeedsReinstallPostUpgradeData** Will the file have a notification after upgrade to reinstall the app?
- **SdbReinstallUpgrade** The file is tagged as needing to be reinstalled after upgrade in the compatibility database (but is not blocking upgrade).

Microsoft.Windows.Appraiser.General.DecisionMatchingInfoPostUpgradeRemove

This event indicates that the DecisionMatchingInfoPostUpgrade object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionMatchingInfoPostUpgradeStartSync

This event indicates that a new set of DecisionMatchingInfoPostUpgradeAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionMediaCenterAdd

This event sends decision data about the presence of Windows Media Center, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **BlockingApplication** Is there any application issues that interfere with upgrade due to Windows Media

Center?

- **MediaCenterActivelyUsed** If Windows Media Center is supported on the edition, has it been run at least once and are the MediaCenterIndicators are true?
- **MediaCenterInUse** Is Windows Media Center actively being used?
- **MediaCenterIndicators** Do any indicators imply that Windows Media Center is in active use?
- **MediaCenterPaidOrActivelyUsed** Is Windows Media Center actively being used or is it running on a supported edition?
- **NeedsDismissAction** Are there any actions that can be dismissed coming from Windows Media Center?

Microsoft.Windows.Appraiser.General.DecisionMediaCenterRemove

This event indicates that the DecisionMediaCenter object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionMediaCenterStartSync

This event indicates that a new set of DecisionMediaCenterAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionSystemBiosAdd

This event sends compatibility decision data about the BIOS to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **Blocking** Is the device blocked from upgrade due to a BIOS block?
- **HasBiosBlock** Does the device have a BIOS block?

Microsoft.Windows.Appraiser.General.DecisionSystemBiosRemove

This event indicates that the DecisionSystemBios object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.DecisionSystemBiosStartSync

This event indicates that a new set of DecisionSystemBiosAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.EnterpriseScenarioWithDiagTrackServiceRunning

The event that indicates that Appraiser has been triggered to run an enterprise scenario while the DiagTrack service is installed. This event can only be sent if a special flag is used to trigger the enterprise scenario.

The following fields are available:

- **Time** The client time of the event.
- **PCFP** An ID for the system calculated by hashing hardware identifiers.

Microsoft.Windows.Appraiser.General.GatedRegChange

This event sends data about the results of running a set of quick-blocking instructions, to help keep Windows up to date.

The following fields are available:

- **Time** The client time of the event.
- **PCFP** An ID for the system calculated by hashing hardware identifiers.
- **RegKey** The registry key name for which a result is being sent.
- **RegValue** The registry value for which a result is being sent.
- **OldData** The previous data in the registry value before the scan ran.
- **NewData** The data in the registry value after the scan completed.

Microsoft.Windows.Appraiser.General.InventoryApplicationFileAdd

This event represents the basic metadata about a file on the system. The file must be part of an app and either have a block in the compatibility database or are part of an anti-virus program.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **BinFileVersion** An attempt to clean up FileVersion at the client that tries to place the version into 4 octets.
- **BinProductVersion** An attempt to clean up ProductVersion at the client that tries to place the version into 4 octets.
- **BinaryType** A binary type. Example: UNINITIALIZED, ZERO_BYTE, DATA_ONLY, DOS_MODULE, NE16_MODULE, PE32_UNKNOWN, PE32_I386, PE32_ARM, PE64_UNKNOWN, PE64_AMD64, PE64_ARM64, PE64_IA64, PE32_CLR_32, PE32_CLR_IL, PE32_CLR_IL_PREFER32, PE64_CLR_64
- **BoeProgramId** If there is no entry in Add/Remove Programs, this is the ProgramID that is generated from the file metadata.
- **CompanyName** The company name of the vendor who developed this file.
- **FileId** A hash that uniquely identifies a file.
- **FileVersion** The File version field from the file metadata under Properties -> Details.
- **LinkDate** The date and time that this file was linked on.
- **LowerCaseLongPath** The full file path to the file that was inventoried on the device.
- **Name** The name of the file that was inventoried.
- **ProductName** The Product name field from the file metadata under Properties -> Details.
- **ProductVersion** The Product version field from the file metadata under Properties -> Details.
- **ProgramId** A hash of the Name, Version, Publisher, and Language of an application used to identify it.
- **Size** The size of the file (in hexadecimal bytes).

Microsoft.Windows.Appraiser.General.InventoryApplicationFileRemove

This event indicates that the InventoryApplicationFile object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.InventoryApplicationFileStartSync

This event indicates indicates that a new set of InventoryApplicationFileAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.InventoryLanguagePackAdd

This event sends data about the number of language packs installed on the system, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **HasLanguagePack** Does this device have 2 or more language packs?
- **LanguagePackCount** How many language packs are installed?

Microsoft.Windows.Appraiser.General.InventoryLanguagePackRemove

This event indicates that the InventoryLanguagePack object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.InventoryLanguagePackStartSync

This event indicates that a new set of InventoryLanguagePackAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.InventoryMediaCenterAdd

This event sends true/false data about decision points used to understand whether Windows Media Center is used on the system, to help keep Windows up to date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **EverLaunched** Has Windows Media Center ever been launched?
- **HasConfiguredTv** Has the user configured a TV tuner through Windows Media Center?
- **HasExtendedUserAccounts** Are any Windows Media Center Extender user accounts configured?
- **HasWatchedFolders** Are any folders configured for Windows Media Center to watch?
- **IsDefaultLauncher** Is Windows Media Center the default app for opening music or video files?
- **IsPaid** Is the user running a Windows Media Center edition that implies they paid for Windows Media Center?
- **IsSupported** Does the running OS support Windows Media Center?

Microsoft.Windows.Appraiser.General.InventoryMediaCenterRemove

This event indicates that the InventoryMediaCenter object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.InventoryMediaCenterStartSync

This event indicates that a new set of InventoryMediaCenterAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.InventorySystemBiosAdd

This event sends basic metadata about the BIOS to determine whether it has a compatibility block.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **BiosDate** The release date of the BIOS in UTC format.
- **BiosName** The name field from Win32_BIOS.
- **Manufacturer** The manufacturer field from Win32_ComputerSystem.
- **Model** The model field from Win32_ComputerSystem.

Microsoft.Windows.Appraiser.General.InventorySystemBiosRemove

This event indicates that the InventorySystemBios object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.InventorySystemBiosStartSync

This event indicates that a new set of InventorySystemBiosAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.InventoryUplevelDriverPackageAdd

This event is only runs during setup. It provides a listing of the uplevel driver packages that were downloaded before the upgrade. Is critical to understanding if failures in setup can be traced to not having sufficient uplevel drivers before the upgrade.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **BootCritical** Is the driver package marked as boot critical?
- **Build** The build value from the driver package.
- **CatalogFile** The name of the catalog file within the driver package.
- **ClassGuid** The device class GUID from the driver package.
- **Class** The device class from the driver package.
- **Date** The date from the driver package.
- **SignatureStatus** Indicates if the driver package is signed. Unknown:0, Unsigned:1, Signed: 2
- **Inbox** Is the driver package of a driver that is included with Windows?
- **VersionMajor** The major version of the driver package.
- **VersionMinor** The minor version of the driver package.
- **OriginalName** The original name of the INF file before it was renamed. Generally a path under \$WINDOWS.~BT\Drivers\DU
- **Provider** The provider of the driver package.
- **PublishedName** The name of the INF file, post-rename.
- **Revision** The revision of the driver package.

Microsoft.Windows.Appraiser.General.InventoryUplevelDriverPackageRemove

This event indicates that the InventoryUplevelDriverPackage object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.InventoryUplevelDriverPackageStartSync

This event indicates that a new set of InventoryUplevelDriverPackageAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.IsOnlineTelemetryOutputter

This event indicates if Appraiser was able to connect successfully to Windows Update to get driver availability information.

The following fields are available:

- **Time** The client time of the event.
- **PCFP** A unique hardware identifier that is calculated by hashing hardware identifiers.
- **IsOnlineRun** Was the device able to connect to Windows Update to get driver availability information?

Microsoft.Windows.Appraiser.General.IsOnlineWuDriverDataSource

This event indicates if Appraiser was able to connect to Windows Update to gather driver coverage information.

The following fields are available:

- **Time** The client time of the event.
- **PCFP** A unique hardware identifier that is calculated by hashing hardware identifiers.
- **IsOnlineRun** Was the device able to connect to Windows Update to get driver availability information?
- **TargetVersion** The abbreviated name for the OS version against which Windows Update was queried.

Microsoft.Windows.Appraiser.General.RunContext

This event indicates what should be expected in the data payload.

The following fields are available:

- **AppraiserBranch** The source branch in which the currently running version of Appraiser was built.
- **AppraiserVersion** The version of the Appraiser file generating the events.
- **Context** Indicates what mode Appraiser is running in. Example: Setup or Telemetry.
- **Time** The client time of the event.
- **AppraiserProcess** The name of the process that launched Appraiser.
- **PCFP** An ID for the system calculated by hashing hardware identifiers.

Microsoft.Windows.Appraiser.General.SetupAdlStatus

This event indicates if Appraiser used data files from the setup image or more up-to-date data files downloaded from a Microsoft server.

The following fields are available:

- **Time** The client time of the event.
- **PCFP** An ID for the system calculated by hashing hardware identifiers.
- **Result** The last result of the operation to determine if there is a data file to download.
- **OneSettingsInitialized** Was the query to OneSettings, where the information is stored on if there is a data file to download, initialized?
- **Url** The URL of the data file to download. This will be an empty string if there is no data file to download.
- **UsingAlternateData** Is the client using alternate data file or using the data file in the setup image?

Microsoft.Windows.Appraiser.General.SystemMemoryAdd

This event sends data on the amount of memory on the system and whether it meets requirements, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **Blocking** Is the device from upgrade due to memory restrictions?
- **MemoryRequirementViolated** Was a memory requirement violated?
- **pageFile** The current committed memory limit for the system or the current process, whichever is smaller (in bytes).
- **ram** The amount of memory on the device.

- **ramKB** The amount of memory (in KB).
- **virtual** The size of the user-mode portion of the virtual address space of the calling process (in bytes).
- **virtualKB** The amount of virtual memory (in KB).

Microsoft.Windows.Appraiser.General.SystemMemoryRemove

This event indicates that the SystemMemory object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemMemoryStartSync

This event indicates that a new set of SystemMemoryAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemProcessorCompareExchangeAdd

This event sends data indicating whether the system supports the CompareExchange128 CPU requirement, to help keep Windows up to date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **Blocking** Is the upgrade blocked due to the processor?
- **CompareExchange128Support** Does the CPU support CompareExchange128?

Microsoft.Windows.Appraiser.General.SystemProcessorCompareExchangeRemove

This event indicates that the SystemProcessorCompareExchange object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemProcessorCompareExchangeStartSync

This event indicates that a new set of SystemProcessorCompareExchangeAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemProcessorLahfSahfAdd

This event sends data indicating whether the system supports the LahfSahf CPU requirement, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file generating the events.
- **Blocking** Is the upgrade blocked due to the processor?
- **LahfSahfSupport** Does the CPU support LAHF/SAHF?

Microsoft.Windows.Appraiser.General.SystemProcessorLahfSahfRemove

This event indicates that the SystemProcessorLahfSahf object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemProcessorLahfSahfStartSync

This event indicates that a new set of SystemProcessorLahfSahfAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemProcessorNxAdd

This event sends data indicating whether the system supports the NX CPU requirement, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **Blocking** Is the upgrade blocked due to the processor?
- **NXDriverResult** The result of the driver used to do a non-deterministic check for NX support.
- **NXProcessorSupport** Does the processor support NX?

Microsoft.Windows.Appraiser.General.SystemProcessorNxRemove

This event indicates that the SystemProcessorNx object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemProcessorNxStartSync

This event indicates that a new set of SystemProcessorNxAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemProcessorPrefetchWAdd

This event sends data indicating whether the system supports the PrefetchW CPU requirement, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **Blocking** Is the upgrade blocked due to the processor?
- **PrefetchWSupport** Does the processor support PrefetchW?

Microsoft.Windows.Appraiser.General.SystemProcessorPrefetchWRemove

This event indicates that the SystemProcessorPrefetchW object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemProcessorPrefetchWStartSync

This event indicates that a new set of SystemProcessorPrefetchWAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemProcessorSse2Add

This event sends data indicating whether the system supports the SSE2 CPU requirement, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **Blocking** Is the upgrade blocked due to the processor?
- **SSE2ProcessorSupport** Does the processor support SSE2?

Microsoft.Windows.Appraiser.General.SystemProcessorSse2Remove

This event indicates that the SystemProcessorSse2 object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemProcessorSse2StartSync

This event indicates that a new set of SystemProcessorSse2Add events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemTouchAdd

This event sends data indicating whether the system supports touch, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **IntegratedTouchDigitizerPresent** Is there an integrated touch digitizer?
- **MaximumTouches** The maximum number of touch points supported by the device hardware.

Microsoft.Windows.Appraiser.General.SystemTouchRemove

This event indicates that the SystemTouch object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemTouchStartSync

This event indicates that a new set of SystemTouchAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemWimAdd

This event sends data indicating whether the operating system is running from a compressed WIM file, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **IsWimBoot** Is the current operating system running from a compressed WIM file?
- **RegistryWimBootValue** The raw value from the registry that is used to indicate if the device is running from a WIM.

Microsoft.Windows.Appraiser.General.SystemWimRemove

This event indicates that the SystemWim object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemWimStartSync

This event indicates that a new set of SystemWimAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemWindowsActivationStatusAdd

This event sends data indicating whether the current operating system is activated, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **WindowsIsLicensedApiValue** The result from the API that's used to indicate if operating system is activated.
- **WindowsNotActivatedDecision** Is the current operating system activated?

Microsoft.Windows.Appraiser.General.SystemWindowsActivationStatusRemove

This event indicates that the SystemWindowsActivationStatus object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemWindowsActivationStatusStartSync

This event indicates that a new set of SystemWindowsActivationStatusAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemWlanAdd

This event sends data indicating whether the system has WLAN, and if so, whether it uses an emulated driver that could block an upgrade, to help keep Windows up-to-date.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **Blocking** Is the upgrade blocked because of an emulated WLAN driver?
- **HasWlanBlock** Does the emulated WLAN driver have an upgrade block?
- **WlanEmulatedDriver** Does the device have an emulated WLAN driver?
- **WlanExists** Does the device support WLAN at all?
- **WlanModulePresent** Are any WLAN modules present?
- **WlanNativeDriver** Does the device have a non-emulated WLAN driver?

Microsoft.Windows.Appraiser.General.SystemWlanRemove

This event indicates that the SystemWlan object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.SystemWlanStartSync

This event indicates that a new set of SystemWlanAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.TelemetryRunHealth

A summary event indicating the parameters and result of a telemetry run. This allows the rest of the data sent over the course of the run to be properly contextualized and understood, which is then used to keep Windows up-to-date.

The following fields are available:

- **PerfBackoff** Indicates if the run was invoked with logic to stop running when a user is present. Helps to understand why a run may have a longer elapsed time than normal.
- **RunAppraiser** Indicates if Appraiser was set to run at all. If this is false, it is understood that data events will not be received from this device.
- **ThrottlingUtc** Indicates if the Appraiser client is throttling its output of CUET events to avoid being disabled. This increases runtime but also telemetry reliability.
- **AuxInitial** Obsolete, indicates if Appraiser is writing data files to be read by the Get Windows 10 app.
- **Time** The client time of the event.
- **RunDate** The date that the telemetry run was stated, expressed as a filetime.
- **AppraiserProcess** The name of the process that launched Appraiser.
- **AppraiserVersion** The file version (major, minor and build) of the Appraiser DLL, concatenated without dots.
- **SendingUtc** Indicates if the Appraiser client is sending events during the current telemetry run.
- **DeadlineDate** A timestamp representing the deadline date, which is the time until which appraiser will wait to do a full scan.
- **AppraiserBranch** The source branch in which the version of Appraiser that is running was built.
- **EnterpriseRun** Indicates if the telemetry run is an enterprise run, which means appraiser was run from the command line with an extra enterprise parameter.
- **RunGeneralTel** Indicates if the generaltel.dll component was run. Generaltel collects additional telemetry on an infrequent schedule and only from machines at telemetry levels higher than Basic.
- **PerfBackoffInsurance** Indicates if appraiser is running without performance backoff because it has run with perf backoff and failed to complete several times in a row.
- **AuxFinal** Obsolete, always set to false
- **StoreHandleIsNotNull** Obsolete, always set to false
- **VerboseMode** Indicates if appraiser ran in Verbose mode, which is a test-only mode with extra logging.
- **AppraiserDataVersion** The version of the data files being used by the Appraiser telemetry run.
- **FullSync** Indicates if Appraiser is performing a full sync, which means that full set of events representing the state of the machine are sent. Otherwise, only the changes from the previous run are sent.
- **InventoryFullSync** Indicates if inventory is performing a full sync, which means that the full set of events representing the inventory of machine are sent.
- **PCFP** An ID for the system calculated by hashing hardware identifiers.
- **RunOnline** Indicates if appraiser was able to connect to Windows Update and therefore is making decisions using up-to-date driver coverage information.
- **TelemetrySent** Indicates if telemetry was successfully sent.
- **WhyFullSyncWithoutTablePrefix** Indicates the reason or reasons that a full sync was generated.
- **RunResult** The HRESULT of the Appraiser telemetry run.

Microsoft.Windows.Appraiser.General.WmdrmAdd

This event sends data about the usage of older digital rights management on the system, to help keep Windows up to date. This data does not indicate the details of the media using the digital rights management, only whether any such files exist. Collecting this data was critical to ensuring the correct mitigation for customers, and should be able to be removed once all mitigations are in place.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.
- **WmdrmCdRipped** Indicates if the system has any files encrypted with personal DRM, which was used for ripped CDs.
- **WmdrmNonPermanent** Indicates if the system has any files with non-permanent licenses.
- **WmdrmPurchased** Indicates if the system has any files with permanent licenses.
- **WmdrmApiResult** Raw value of the API used to gather DRM state.
- **WmdrmInUse** WmdrmIndicators AND dismissible block in setup was not dismissed.
- **WmdrmIndicators** WmdrmCdRipped OR WmdrmPurchased
- **NeedsDismissAction** Indicates if a dismissible message is needed to warn the user about a potential loss of data due to DRM deprecation.
- **BlockingApplication** Same as NeedsDismissAction

Microsoft.Windows.Appraiser.General.WmdrmRemove

This event indicates that the Wmdrm object is no longer present.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Microsoft.Windows.Appraiser.General.WmdrmStartSync

This event indicates that a new set of WmdrmAdd events will be sent.

The following fields are available:

- **AppraiserVersion** The version of the Appraiser file that is generating the events.

Census events

Census.App

This event sends version data about the Apps running on this device, to help keep Windows up to date.

The following fields are available:

- **IEVersion** Retrieves which version of Internet Explorer is running on this device.
- **CensusVersion** The version of Census that generated the current data for this device.

Census.Battery

This event sends type and capacity data about the battery on the device, as well as the number of connected standby devices in use, type to help keep Windows up to date.

The following fields are available:

- **InternalBatteryCapabilities** Represents information about what the battery is capable of doing.
- **InternalBatteryCapacityCurrent** Represents the battery's current fully charged capacity in mWh (or relative). Compare this value to DesignedCapacity to estimate the battery's wear.
- **InternalBatteryCapacityDesign** Represents the theoretical capacity of the battery when new, in mWh.
- **IsAlwaysOnAlwaysConnectedCapable** Represents whether the battery enables the device to be AlwaysOnAlwaysConnected . Boolean value.
- **InternalBatteryNumberOfCharges** Provides the number of battery charges. This is used when creating new products and validating that existing products meets targeted functionality performance.

Census.Camera

This event sends data about the resolution of cameras on the device, to help keep Windows up to date.

The following fields are available:

- **FrontFacingCameraResolution** Represents the resolution of the front facing camera in megapixels. If a front facing camera does not exist, then the value is 0.
- **RearFacingCameraResolution** Represents the resolution of the rear facing camera in megapixels. If a rear facing camera does not exist, then the value is 0.

Census.Enterprise

This event sends data about Azure presence, type, and cloud domain use in order to provide an understanding of the use and integration of devices in an enterprise, cloud, and server environment.

The following fields are available:

- **IsCloudDomainJoined** Is this device joined to an Azure Active Directory (AAD) tenant? true/false
- **IsMDMEnrolled** Whether the device has been MDM Enrolled or not.
- **ServerFeatures** Represents the features installed on a Windows Server. This can be used by developers and administrators who need to automate the process of determining the features installed on a set of server computers.
- **CommercialId** Represents the GUID for the commercial entity which the device is a member of. Will be used to reflect insights back to customers.
- **AzureVMTType** Represents whether the instance is Azure VM PAAS, Azure VM IAAS or any other VMs.
- **AzureOSIDPresent** Represents the field used to identify an Azure machine.
- **IsDomainJoined** Indicates whether a machine is joined to a domain.
- **HashedDomain** The hashed representation of the user domain used for login.
- **SystemCenterID** The SCCM ID is an anonymized one-way hash of the Active Directory Organization identifier
- **MPNId** Returns the Partner ID/MPN ID from Regkey.
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\DeployID
- **SCCMClientId** This ID correlate systems that send data to Compat Analytics (OMS) and other OMS based systems with systems in an Enterprise SCCM environment.
- **CDJType** Represents the type of cloud domain joined for the machine.
- **IsDeviceProtected** Represents if Device protected by BitLocker/Device Encryption
- **IsDRequirementMet** Represents if the device can do device encryption.
- **IsEDPEnabled** Represents if Enterprise data protected on the device.
- **ContainerType** The type of container, such as process or virtual machine hosted.

Census.Firmware

This event sends data about the BIOS and startup embedded in the device, to help keep Windows up to date.

The following fields are available:

- **FirmwareManufacturer** Represents the manufacturer of the device's firmware (BIOS).
- **FirmwareReleaseDate** Represents the date the current firmware was released.
- **FirmwareType** Represents the firmware type. The various types can be unknown, BIOS, UEFI.
- **FirmwareVersion** Represents the version of the current firmware.

Census.Flighting

This event sends Windows Insider data from customers participating in improvement testing and feedback programs, to help keep Windows up-to-date.

The following fields are available:

- **FlightIds** A list of the different Windows Insider builds on this device.
- **MSA_Accounts** Represents a list of hashed IDs of the Microsoft Accounts that are flighting (pre-release builds)

on this device.

- **IsFlightDisabled** Represents if the device is participating in the Windows Insider program.
- **FlightingBranchName** The name of the Windows Insider branch currently used by the device.
- **DeviceSampleRate** The telemetry sample rate assigned to the device.
- **EnablePreviewBuilds** Used to enable Windows Insider builds on a device.
- **SSRK** Retrieves the mobile targeting settings.

Census.Hardware

This event sends data about the device, including hardware type, OEM brand, model line, model, telemetry level setting, and TPM support, to help keep Windows up-to-date.

The following fields are available:

- **ChassisType** Represents the type of device chassis, such as desktop or low profile desktop. The possible values can range between 1 - 36.
- **ComputerHardwareID** Identifies a device class that is represented by a hash of different SMBIOS fields.
- **DeviceColor** Indicates a color of the device.
- **DeviceName** The device name that is set by the user.
- **OEMDigitalMarkerFileName** The name of the file placed in the \Windows\system32\drivers directory that specifies the OEM and model name of the device.
- **OEMManufacturerName** The device manufacturer name. The OEMName for an inactive device is not reprocessed even if the clean OEM name is changed at a later date.
- **OEMModelNumber** The device model number.
- **OEMModelName** The device model name.
- **OEMModelSKU** The device edition that is defined by the manufacturer.
- **OEMOptionalIdentifier** A Microsoft assigned value that represents a specific OEM subsidiary.
- **OEMSerialNumber** The serial number of the device that is set by the manufacturer.
- **PhoneManufacturer** The friendly name of the phone manufacturer.
- **SoCName** The firmware manufacturer of the device.
- **DUID** The device unique ID.
- **InventoryId** The device ID used for compatibility testing.
- **VoiceSupported** Does the device have a cellular radio capable of making voice calls?
- **PowerPlatformRole** The OEM preferred power management profile. It's used to help to identify the basic form factor of the device.
- **TPMVersion** The supported Trusted Platform Module (TPM) on the device. If no TPM is present, the value is 0.
- **StudyID** Used to identify retail and non-retail device.
- **TelemetryLevel** The telemetry level the user has opted into, such as Basic or Enhanced.
- **TelemetrySettingAuthority** Determines who set the telemetry level, such as GP, MDM, or the user.
- **DeviceForm** Indicates the form as per the device classification.
- **DigitizerSupport** Is a digitizer supported?
- **OEMModelBaseBoard** The baseboard model used by the OEM.
- **OEMModelSystemFamily** The system family set on the device by an OEM.
- **OEMModelBaseBoardVersion** Differentiates between developer and retail devices.
- **ActiveMicCount** The number of active microphones attached to the device.
- **OEMModelSystemVersion** The system model version set on the device by the OEM.

Census.Memory

This event sends data about the memory on the device, including ROM and RAM, to help keep Windows up to date.

The following fields are available:

- **TotalPhysicalRAM** Represents the physical memory (in MB).
- **TotalVisibleMemory** Represents the memory that is not reserved by the system.

Census.Network

This event sends data about the mobile and cellular network used by the device (mobile service provider, network, device ID, and service cost factors), to help keep Windows up to date.

The following fields are available:

- **MobileOperatorBilling** Represents the telephone company that provides services for mobile phone users.
- **MobileOperatorCommercialized** Represents which reseller and geography the phone is commercialized for. This is the set of values on the phone for who and where it was intended to be used. For example, the commercialized mobile operator code AT&T in the US would be ATT-US.
- **NetworkCost** Represents the network cost associated with a connection.
- **IMEIO** Represents the International Mobile Station Equipment Identity. This number is usually unique and used by the mobile operator to distinguish different phone hardware. Microsoft does not have access to mobile operator billing data so collecting this data does not expose or identify the user. The two fields represent phone with dual sim coverage.
- **SPN0** Retrieves the Service Provider Name (SPN). For example, these might be AT&T, Sprint, T-Mobile, or Verizon. The two fields represent phone with dual sim coverage.
- **MobileOperatorNetwork0** Represents the operator of the current mobile network that the device is used on. (AT&T, T-Mobile, Vodafone). The two fields represent phone with dual sim coverage.
- **MCC0** Represents the Mobile Country Code (MCC). It used with the Mobile Network Code (MNC) to uniquely identify a mobile network operator. The two fields represent phone with dual sim coverage.
- **MNC0** Retrieves the Mobile Network Code (MNC). It used with the Mobile Country Code (MCC) to uniquely identify a mobile network operator. The two fields represent phone with dual sim coverage.
- **IMEI1** Represents the International Mobile Station Equipment Identity. This number is usually unique and used by the mobile operator to distinguish different phone hardware. Microsoft does not have access to mobile operator billing data so collecting this data does not expose or identify the user. The two fields represent phone with dual sim coverage.
- **SPN1** Retrieves the Service Provider Name (SPN). For example, these might be AT&T, Sprint, T-Mobile, or Verizon. The two fields represent phone with dual sim coverage.
- **MobileOperatorNetwork1** Represents the operator of the current mobile network that the device is used on. (AT&T, T-Mobile, Vodafone). The two fields represent phone with dual sim coverage.
- **MCC1** Represents the Mobile Country Code (MCC). It used with the Mobile Network Code (MNC) to uniquely identify a mobile network operator. The two fields represent phone with dual sim coverage.
- **MNC1** Retrieves the Mobile Network Code (MNC). It used with the Mobile Country Code (MCC) to uniquely identify a mobile network operator. The two fields represent phone with dual sim coverage.
- **MEID** Represents the Mobile Equipment Identity (MEID). MEID is a worldwide unique phone ID assigned to CDMA phones. MEID replaces electronic serial number (ESN), and is equivalent to IMEI for GSM and WCDMA phones. Microsoft does not have access to mobile operator billing data so collecting this data does not expose or identify the user.
- **NetworkAdapterGUID** The GUID of the primary network adapter.

Census.OS

This event sends data about the operating system such as the version, locale, update service configuration, when and how it was originally installed, and whether it is a virtual device, to help keep Windows up to date.

The following fields are available:

- **GenuineState** Retrieves the ID Value specifying the OS Genuine check.

- **IsPortableOperatingSystem** Retrieves whether OS is running Windows-To-Go
- **IsSecureBootEnabled** Retrieves whether Boot chain is signed under UEFI.
- **InstallationType** Retrieves the type of OS installation. (Clean, Upgrade, Reset, Refresh, Update).
- **OSInstallType** Retrieves a numeric description of what install was used on the device i.e. clean, upgrade, refresh, reset, etc
- **OSOOBEDateTime** Retrieves Out of Box Experience (OOBE) Date in Coordinated Universal Time (UTC).
- **OSSKU** Retrieves the Friendly Name of OS Edition.
- **OSTimeZoneBiasInMins** Retrieves the time zone set on machine.
- **OSUILocale** Retrieves the locale of the UI that is currently used by the OS.
- **RACw7Id** Retrieves the Microsoft Reliability Analysis Component (RAC) Win7 Identifier. RAC is used to monitor and analyze system usage and reliability.
- **CompactOS** Indicates if the Compact OS feature from Win10 is enabled.
- **Signature** Retrieves if it is a signature machine sold by Microsoft store.
- **IsDeviceRetailDemo** Retrieves if the device is running in demo mode.
- **ActivationChannel** Retrieves the retail license key or Volume license key for a machine.
- **LicenseStateReason** Retrieves why (or how) a system is licensed or unlicensed. The HRESULT may indicate an error code that indicates a key blocked error, or it may indicate that we are running an OS License granted by the MS store.
- **OA3xOriginalProductKey** Retrieves the License key stamped by the OEM to the machine.
- **ProductKeyID2** Retrieves the License key if the machine is updated with a new license key.
- **ServiceMachineIP** Retrieves the IP address of the KMS host used for anti-piracy.
- **ServiceProductKeyID** Retrieves the License key of the KMS
- **LanguagePacks** The list of language packages installed on the device.
- **InstallLanguage** The first language installed on the user machine.
- **IsEduData** Returns Boolean if the education data policy is enabled.
- **SharedPCMode** Returns Boolean for education devices used as shared cart
- **SLICVersion** Returns OS type/version from SLIC table.
- **SLICStatus** Whether a SLIC table exists on the device.
- **OSEdition** Retrieves the version of the current OS.
- **ProductActivationTime** Returns the OS Activation time for tracking piracy issues.
- **ProductActivationResult** Returns Boolean if the OS Activation was successful.
- **OSSubscriptionTypeId** Returns boolean for enterprise subscription feature for selected PRO machines.
- **OSSubscriptionStatus** Represents the existing status for enterprise subscription feature for PRO machines.
- **ServiceMachinePort** Retrieves the port of the KMS host used for anti-piracy.
- **DeviceTimeZone** The time zone that is set on the device. Example: Pacific Standard Time
- **DeveloperUnlockStatus** Represents if a device has been developer unlocked by the user or Group Policy.

Census.Processor

This event sends data about the processor (architecture, speed, number of cores, manufacturer, and model number), to help keep Windows up to date.

The following fields are available:

- **ProcessorCores** Retrieves the number of cores in the processor.
- **ProcessorPhysicalCores** Number of physical cores in the processor.
- **ProcessorArchitecture** Retrieves the processor architecture of the installed operating system. The complete list of values can be found in DimProcessorArchitecture.
- **ProcessorClockSpeed** Retrieves the clock speed of the processor in MHz.
- **ProcessorManufacturer** Retrieves the name of the processor's manufacturer.

- **ProcessorModel** Retrieves the name of the processor model.
- **SocketCount** Number of physical CPU sockets of the machine.
- **ProcessorIdentifier** The processor identifier of a manufacturer.

Census.Speech

This event is used to gather basic speech settings on the device.

The following fields are available:

- **AboveLockEnabled** Cortana setting that represents if Cortana can be invoked when the device is locked.
- **GPAallowInputPersonalization** Indicates if a Group Policy setting has enabled speech functionalities.
- **HolographicSpeechInputDisabled** Holographic setting that represents if the attached HMD devices have speech functionality disabled by the user.
- **HolographicSpeechInputDisabledRemote** Indicates if a remote policy has disabled speech functionalities for the HMD devices.
- **KWSEnabled** Cortana setting that represents if a user has enabled the "Hey Cortana" keyword spotter (KWS).
- **MDMAllowInputPersonalization** Indicates if an MDM policy has enabled speech functionalities.
- **RemotelyManaged** Indicates if the device is being controlled by a remote administrator (MDM or Group Policy) in the context of speech functionalities.
- **SpeakerIdEnabled** Cortana setting that represents if keyword detection has been trained to try to respond to a single user's voice.
- **SpeechServicesEnabled** Windows setting that represents whether a user is opted-in for speech services on the device.

Census.Storage

This event sends data about the total capacity of the system volume and primary disk, to help keep Windows up to date.

The following fields are available:

- **PrimaryDiskTotalCapacity** Retrieves the amount of disk space on the primary disk of the device in MB.
- **SystemVolumeTotalCapacity** Retrieves the size of the partition that the System volume is installed on in MB.
- **PrimaryDiskType** Retrieves an enumerator value of type STORAGE_BUS_TYPE that indicates the type of bus to which the device is connected. This should be used to interpret the raw device properties at the end of this structure (if any).

Census.Userdefault

This event sends data about the current user's default preferences for browser and several of the most popular extensions and protocols, to help keep Windows up to date.

The following fields are available:

- **DefaultBrowserProgId** The ProgramId of the current user's default browser
- **DefaultApp** The current user's default program selected for the following extension or protocol:
.html,.htm,.jpg,.jpeg,.png,.mp3,.mp4,.mov,.pdf

Census.UserDisplay

This event sends data about the logical/physical display size, resolution and number of internal/external displays, and VRAM on the system, to help keep Windows up to date.

The following fields are available:

- **InternalPrimaryDisplayLogicalDPIX** Retrieves the logical DPI in the x-direction of the internal display.
- **InternalPrimaryDisplayLogicalDPIY** Retrieves the logical DPI in the y-direction of the internal display.
- **InternalPrimaryDisplayPhysicalDPIX** Retrieves the physical DPI in the x-direction of the internal display.

- **InternalPrimaryDisplayPhysicalDPIY** Retrieves the physical DPI in the y-direction of the internal display.
- **InternalPrimaryDisplayResolutionHorizontal** Retrieves the number of pixels in the horizontal direction of the internal display.
- **InternalPrimaryDisplayResolutionVertical** Retrieves the number of pixels in the vertical direction of the internal display.
- **InternalPrimaryDisplaySizePhysicalH** Retrieves the physical horizontal length of the display in mm. Used for calculating the diagonal length in inches .
- **InternalPrimaryDisplaySizePhysicalY** Retrieves the physical vertical length of the display in mm. Used for calculating the diagonal length in inches
- **NumberofInternalDisplays** Retrieves the number of internal displays in a machine.
- **NumberofExternalDisplays** Retrieves the number of external displays connected to the machine
- **VRAMDedicated** Retrieves the video RAM in MB.
- **VRAMDedicatedSystem** Retrieves the amount of memory on the dedicated video card.
- **VRAMSharedSystem** Retrieves the amount of RAM memory that the video card can use.

Census.UserNLS

This event sends data about the default app language, input, and display language preferences set by the user, to help keep Windows up to date.

The following fields are available:

- **DefaultAppLanguage** The current user Default App Language.
- **HomeLocation** The current user location, which is populated using GetUserGeoid() function.
- **DisplayLanguage** The current user preferred Windows Display Language.
- **SpeechInputLanguages** The Speech Input languages installed on the device.
- **KeyboardInputLanguages** The Keyboard input languages installed on the device.

Census.VM

This event sends data indicating whether virtualization is enabled on the device, and its various characteristics, to help keep Windows up to date.

The following fields are available:

- **VirtualizationFirmwareEnabled** Represents whether virtualization is enabled in the firmware.
- **SLATSupported** Represents whether Second Level Address Translation (SLAT) is supported by the hardware.
- **IOMMUPresent** Represents if an input/output memory management unit (IOMMU) is present.
- **IsVirtualDevice** Retrieves that when the Hypervisor is Microsoft's Hyper-V Hypervisor or other Hv#1 Hypervisor, this field will be set to FALSE for the Hyper-V host OS and TRUE for any guest OS's. This field should not be relied upon for non-Hv#1 Hypervisors.
- **HyperVisor** Retrieves whether the current OS is running on top of a Hypervisor.

Census.WU

This event sends data about the Windows update server and other App store policies, to help keep Windows up to date.

The following fields are available:

- **WUMachineId** Retrieves the Windows Update (WU) Machine Identifier.
- **WUServer** Retrieves the HTTP(S) URL of the WSUS server that is used by Automatic Updates and API callers (by default).
- **WUDODownloadMode** Retrieves whether DO is turned on and how to acquire/distribute updates Delivery Optimization (DO) allows users to deploy previously downloaded WU updates to other devices on the same network.

- **OSWUAutoUpdateOptions** Retrieves the auto update settings on the device.
- **AppStoreAutoUpdate** Retrieves the Appstore settings for auto upgrade. (Enable/Disabled).
- **AppStoreAutoUpdatePolicy** Retrieves the Microsoft Store App Auto Update group policy setting
- **AppStoreAutoUpdateMDM** Retrieves the App Auto Update value for MDM: 0 - Disallowed. 1 - Allowed. 2 - Not configured. Default: [2] Not configured
- **DelayUpgrade** Retrieves the Windows upgrade flag for delaying upgrades.
- **UpdateServiceURLConfigured** Retrieves if the device is managed by Windows Server Update Services (WSUS).
- **WUDeferUpgradePeriod** Retrieves if deferral is set for Upgrades
- **WUDeferUpdatePeriod** Retrieves if deferral is set for Updates
- **WUPauseState** Retrieves WU setting to determine if updates are paused
- **OSUninstalled** A flag that represents when a feature update is uninstalled on a device .
- **OSRolledBack** A flag that represents when a feature update has rolled back during setup.
- **OSRollbackCount** The number of times feature updates have rolled back on the device.
- **UninstallActive** A flag that represents when a device has uninstalled a previous upgrade recently.
- **AppraiserGatedStatus** Indicates whether a device has been gated for upgrading.

Census.Xbox

This event sends data about the Xbox Console, such as Serial Number and Deviceld, to help keep Windows up to date.

The following fields are available:

- **XboxLiveDeviceId** Retrieves the unique device id of the console.
- **XboxConsoleSerialNumber** Retrieves the serial number of the Xbox console.
- **XboxLiveSandboxId** Retrieves the developer sandbox id if the device is internal to MS.
- **XboxConsolePreferredLanguage** Retrieves the preferred language selected by the user on Xbox console.

Diagnostic data events

TelClientSynthetic.AuthorizationInfo_RuntimeTransition

This event sends data indicating that a device has undergone a change of telemetry opt-in level during the runtime of the device (not at UTC boot or offline), to help keep Windows up to date.

The following fields are available:

- **CanAddMsaToMsTelemetry** True if UTC is allowed to add MSA user identity onto telemetry from the OS provider groups.
- **CanCollectAnyTelemetry** True if UTC is allowed to collect non-OS telemetry. Non-OS telemetry is responsible for providing its own opt-in mechanism.
- **CanCollectCoreTelemetry** True if UTC is allowed to collect data which is tagged with both MICROSOFT_KEYWORD_CRITICAL_DATA and MICROSOFT_EVENTTAG_CORE_DATA.
- **CanCollectHeartbeats** True if UTC is allowed to collect heartbeats.
- **CanCollectOsTelemetry** True if UTC is allowed to collect telemetry from the OS provider groups (often called Microsoft Telemetry).
- **CanPerformDiagnosticEscalations** True if UTC is allowed to perform all scenario escalations.
- **CanPerformScripting** True if UTC is allowed to perform scripting.
- **CanPerformTraceEscalations** True if UTC is allowed to perform scenario escalations with tracing actions.
- **CanReportScenarios** True if UTC is allowed to load and report scenario completion, failure, and cancellation events.
- **TransitionFromEverythingOff** True if this transition is moving from not allowing core telemetry to allowing

core telemetry.

- **PreviousPermissions** Bitmask representing the previously configured permissions since the telemetry opt-in level was last changed.

TelClientSynthetic.AuthorizationInfo_Startup

This event sends data indicating that a device has undergone a change of telemetry opt-in level detected at UTC startup, to help keep Windows up to date.

The following fields are available:

- **TransitionFromEverythingOff** True if this transition is moving from not allowing core telemetry to allowing core telemetry.
- **CanCollectAnyTelemetry** True if UTC is allowed to collect non-OS telemetry. Non-OS telemetry is responsible for providing its own opt-in mechanism.
- **CanCollectHeartbeats** True if UTC is allowed to collect heartbeats.
- **CanCollectCoreTelemetry** True if UTC is allowed to collect data which is tagged with both MICROSOFT_KEYWORD_CRITICAL_DATA and MICROSOFT_EVENTTAG_CORE_DATA.
- **CanCollectOsTelemetry** True if UTC is allowed to collect telemetry from the OS provider groups (often called Microsoft Telemetry).
- **CanReportScenarios** True if UTC is allowed to load and report scenario completion, failure, and cancellation events.
- **CanAddMsaToMsTelemetry** True if UTC is allowed to add MSA user identity onto telemetry from the OS provider groups.
- **CanPerformTraceEscalations** True if UTC is allowed to perform scenario escalations with tracing actions.
- **CanPerformDiagnosticEscalations** True if UTC is allowed to perform all scenario escalations.
- **CanPerformScripting** True if UTC is allowed to perform scripting.
- **PreviousPermissions** Bitmask representing the previously configured permissions since the telemetry client was last started.

TelClientSynthetic.ConnectivityHeartBeat_0

This event sends data about the connectivity status of the Connected User Experience and Telemetry component that uploads telemetry events. If an unrestricted free network (such as Wi-Fi) is available, this event updates the last successful upload time. Otherwise, it checks whether a Connectivity Heartbeat event was fired in the past 24 hours, and if not, it fires an event. A Connectivity Heartbeat event also fires when a device recovers from costed network to free network.

The following fields are available:

- **CensusExitCode** Returns last execution codes from census client run.
- **CensusStartTime** Returns timestamp corresponding to last successful census run.
- **CensusTaskEnabled** Returns Boolean value for the census task (Enable/Disable) on client machine.
- **LastConnectivityLossTime** Retrieves the last time the device lost free network.
- **NetworkState** Retrieves the network state: 0 = No network. 1 = Restricted network. 2 = Free network.
- **NoNetworkTime** Retrieves the time spent with no network (since the last time) in seconds.
- **RestrictedNetworkTime** Retrieves the time spent on a metered (cost restricted) network in seconds.
- **LastConntectivityLossTime** Retrieves the last time the device lost free network.

TelClientSynthetic.HeartBeat_5

This event sends data about the health and quality of the telemetry data from the given device, to help keep Windows up to date. It also enables data analysts to determine how 'trusted' the data is from a given device.

The following fields are available:

- **PreviousHeartBeatTime** The time of last heartbeat event. This allows chaining of events.
- **EtwDroppedCount** The number of events dropped by the ETW layer of the telemetry client.
- **ConsumerDroppedCount** The number of events dropped by the consumer layer of the telemetry client.
- **DecodingDroppedCount** The number of events dropped because of decoding failures.
- **ThrottledDroppedCount** The number of events dropped due to throttling of noisy providers.
- **DbDroppedCount** The number of events that were dropped because the database was full.
- **EventSubStoreResetCounter** The number of times the event database was reset.
- **EventSubStoreResetSizeSum** The total size of the event database across all resets reports in this instance.
- **CriticalOverflowEntersCounter** The number of times a critical overflow mode was entered into the event database.
- **EnteringCriticalOverflowDroppedCounter** The number of events that was dropped because a critical overflow mode was initiated.
- **UploaderDroppedCount** The number of events dropped by the uploader layer of the telemetry client.
- **InvalidHttpCodeCount** The number of invalid HTTP codes received from Vortex.
- **LastInvalidHttpCode** The last invalid HTTP code received from Vortex.
- **MaxInUseScenarioCounter** The soft maximum number of scenarios loaded by the Connected User Experience and Telemetry component.
- **LastEventSizeOffender** The name of the last event that exceeded the maximum event size.
- **SettingsHttpAttempts** The number of attempts to contact the OneSettings service.
- **SettingsHttpFailures** The number of failures from contacting the OneSettings service.
- **VortexHttpAttempts** The number of attempts to contact the Vortex service.
- **EventsUploaded** The number of events that have been uploaded.
- **DbCriticalDroppedCount** The total number of dropped critical events in the event database.
- **VortexHttpFailures4xx** The number of 400-499 error codes received from Vortex.
- **VortexHttpFailures5xx** The number of 500-599 error codes received from Vortex.
- **VortexFailuresTimeout** The number of timeout failures received from Vortex.
- **HeartBeatSequenceNumber** A monotonically increasing heartbeat counter.
- **EtwDroppedBufferCount** The number of buffers dropped in the CUET ETW session.
- **FullTriggerBufferDroppedCount** The number of events that were dropped because the trigger buffer was full.
- **CriticalDataThrottleDroppedCount** The number of critical data sampled events that were dropped because of throttling.
- **CriticalDataDbDroppedCount** The number of critical data sampled events that were dropped at the database layer.
- **MaxActiveAgentConnectionCount** The maximum number of active agents during this heartbeat timeframe.
- **AgentConnectionErrorsCount** The number of non-timeout errors associated with the host/agent channel.
- **LastAgentConnectionError** The last non-timeout error that happened in the host/agent channel.
- **Flags** Flags that indicate device state, such as network, battery, and opt-in state.
- **CensusTaskEnabled** Indicates whether Census is enabled.
- **CensusExitCode** The last exit code of the Census task.
- **CensusStartTime** The time of the last Census run.

TelClientSynthetic.PrivacySettingsAfterCreatorsUpdate

This event sends basic data on privacy settings before and after a feature update. This is used to ensure that customer privacy settings are correctly migrated across feature updates.

The following fields are available:

- **PostUpgradeSettings** The privacy settings after a feature update.

- **PreUpgradeSettings** The privacy settings before a feature update.

DxgKernelTelemetry events

DxgKrnTelemetry.GPUAdapterInventoryV2

This event sends basic GPU and display driver information to keep Windows and display drivers up-to-date.

The following fields are available:

- **version** The event version.
- **bootId** The system boot ID.
- **aiSeqId** The event sequence ID.
- **MeasureEnabled** Is the device listening to MICROSOFT_KEYWORD_MEASURES?
- **TelemetryEnabled** Is the device listening to MICROSOFT_KEYWORD_TELEMETRY?
- **InterfaceId** The GPU interface ID.
- **GPUVendorId** The GPU vendor ID.
- **GPUDeviceId** The GPU device ID.
- **SubVendorId** The GPU sub vendor ID.
- **SubSystemId** The subsystem ID.
- **GPURevisionId** The GPU revision ID.
- **DriverVersion** The display driver version.
- **DriverDate** The date of the display driver.
- **DriverRank** The rank of the display driver.
- **IsMiracastSupported** Does the GPU support Miracast?
- **IsMsMiracastSupported** Are the GPU Miracast capabilities driven by a Microsoft solution?
- **IsHybridDiscrete** Does the GPU have discrete GPU capabilities in a hybrid device?
- **IsHybridIntegrated** Does the GPU have integrated GPU capabilities in a hybrid device?
- **IsMPOSupported** Does the GPU support Multi-Plane Overlays?
- **IsLDA** Is the GPU comprised of Linked Display Adapters?
- **IsMismatchLDA** Is at least one device in the Linked Display Adapters chain from a different vendor?
- **IsPostAdapter** Is this GPU the POST GPU in the device?
- **IsSoftwareDevice** Is this a software implementation of the GPU?
- **IsRenderDevice** Does the GPU have rendering capabilities?
- **IsDisplayDevice** Does the GPU have displaying capabilities?
- **WDDMVersion** The Windows Display Driver Model version.
- **DisplayAdapterLuid** The display adapter LUID.
- **GPUPreemptionLevel** The maximum preemption level supported by GPU for graphics payload.
- **ComputePreemptionLevel** The maximum preemption level supported by GPU for compute payload.
- **TellInvEvntTrigger** What triggered this event to be logged? Example: 0 (GPU enumeration) or 1 (DxgKrnTelemetry provider toggling)
- **DedicatedVideoMemoryB** The amount of dedicated VRAM of the GPU (in bytes).
- **DedicatedSystemMemoryB** The amount of system memory dedicated for GPU use (in bytes).
- **SharedSystemMemoryB** The amount of system memory shared by GPU and CPU (in bytes).
- **NumVidPnSources** The number of supported display output sources.
- **NumVidPnTargets** The number of supported display output targets.

Fault Reporting events

Microsoft.Windows.FaultReporting.AppCrashEvent

This event sends data about crashes for both native and managed applications, to help keep Windows up to date. The data includes information about the crashing process and a summary of its exception record. It does not contain any Watson bucketing information. The bucketing information is recorded in a Windows Error Reporting (WER) event that is generated when the WER client reports the crash to the Watson service, and the WER event will contain the same ReportID (see field 14 of crash event, field 19 of WER event) as the crash event for the crash being reported. AppCrash is emitted once for each crash handled by WER (e.g. from an unhandled exception or FailFast or ReportException). Note that Generic Watson event types (e.g. from PLM) that may be considered "crashes" by a user DO NOT emit this event.

The following fields are available:

- **ProcessId** The ID of the process that has crashed.
- **ProcessCreateTime** The time of creation of the process that has crashed.
- **ExceptionCode** The exception code returned by the process that has crashed.
- **ExceptionOffset** The address where the exception had occurred.
- **AppName** The name of the app that has crashed.
- **AppVersion** The version of the app that has crashed.
- **AppTimeStamp** The date/time stamp of the app.
- **ModName** Exception module name (e.g. bar.dll).
- **ModVersion** The version of the module that has crashed.
- **ModTimeStamp** The date/time stamp of the module.
- **PackageFullName** Store application identity.
- **PackageRelativeAppId** Store application identity.
- **ProcessArchitecture** Architecture of the crashing process, as one of the PROCESSOR_ARCHITECTURE_* constants: 0: PROCESSOR_ARCHITECTURE_INTEL. 5: PROCESSOR_ARCHITECTURE_ARM. 9: PROCESSOR_ARCHITECTURE_AMD64. 12: PROCESSOR_ARCHITECTURE_ARM64.
- **ReportId** A GUID used to identify the report. This can be used to track the report across Watson.
- **Flags** Flags indicating how reporting is done. For example, queue the report, do not offer JIT debugging, or do not terminate the process after reporting.
- **AppSessionGuid** GUID made up of process ID and is used as a correlation vector for process instances in the telemetry backend.
- **TargetAppId** The kernel reported AppId of the application being reported.
- **TargetAppVer** The specific version of the application being reported
- **TargetAsId** The sequence number for the hanging process.

Hang Reporting events

Microsoft.Windows.HangReporting.AppHangEvent

This event sends data about hangs for both native and managed applications, to help keep Windows up to date. It does not contain any Watson bucketing information. The bucketing information is recorded in a Windows Error Reporting (WER) event that is generated when the WER client reports the hang to the Watson service, and the WER event will contain the same ReportID (see field 13 of hang event, field 19 of WER event) as the hang event for the hang being reported. AppHang is reported only on PC devices. It handles classic Win32 hangs and is emitted only once per report. Some behaviors that may be perceived by a user as a hang are reported by app managers (e.g. PLM/RM/EM) as Watson Generics and will not produce AppHang events.

The following fields are available:

- **AppName** The name of the app that has hung.
- **TypeCode** Bitmap describing the hang type.
- **ProcessId** The ID of the process that has hung.

- **UTCReplace_TargetAppId** The kernel reported AppId of the application being reported.
- **ProcessCreateTime** The time of creation of the process that has hung.
- **UTCReplace_TargetAppVer** The specific version of the application being reported.
- **WaitingOnAppName** If this is a cross process hang waiting for an application, this has the name of the application.
- **PackageRelativeAppId** Store application identity.
- **ProcessArchitecture** Architecture of the hung process, as one of the PROCESSOR_ARCHITECTURE_* constants: 0: PROCESSOR_ARCHITECTURE_INTEL. 5: PROCESSOR_ARCHITECTURE_ARM. 9: PROCESSOR_ARCHITECTURE_AMD64. 12: PROCESSOR_ARCHITECTURE_ARM64.
- **WaitingOnPackageRelativeAppId** If this is a cross process hang waiting for a package, this has the relative application id of the package.
- **WaitingOnAppVersion** If this is a cross process hang, this has the version of the application for which it is waiting.
- **AppSessionGuid** GUID made up of process id used as a correlation vector for process instances in the telemetry backend.
- **WaitingOnPackageFullName** If this is a cross process hang waiting for a package, this has the full name of the package for which it is waiting.
- **PackageFullName** Store application identity.
- **AppVersion** The version of the app that has hung.
- **ReportId** A GUID used to identify the report. This can be used to track the report across Watson.
- **TargetAppId** The kernel reported AppId of the application being reported.
- **TargetAppVer** The specific version of the application being reported.
- **TargetAsId** The sequence number for the hanging process.

Inventory events

Microsoft.Windows.Inventory.Core.AMiTelCacheChecksum

This event captures basic checksum data about the device inventory items stored in the cache for use in validating data completeness for Microsoft.Windows.Inventory.Core events. The fields in this event may change over time, but they will always represent a count of a given object.

The following fields are available:

- **Device** A count of device objects in cache
- **DeviceCensus** A count of devicecensus objects in cache
- **DriverPackageExtended** A count of driverpackageextended objects in cache
- **File** A count of file objects in cache
- **Generic** A count of generic objects in cache
- **HwiItem** A count of hwitem objects in cache
- **InventoryApplication** A count of application objects in cache
- **InventoryApplicationFile** A count of application file objects in cache
- **InventoryDeviceContainer** A count of device container objects in cache
- **InventoryDeviceMediaClass** A count of device media objects in cache
- **InventoryDevicePnp** A count of devicepnp objects in cache
- **InventoryDriverBinary** A count of driver binary objects in cache
- **InventoryDriverPackage** A count of device objects in cache
- **Metadata** A count of metadata objects in cache
- **Orphan** A count of orphan file objects in cache
- **Programs** A count of program objects in cache

- **FileSigningInfo** A count of file signing info objects in cache.
- **InventoryDeviceInterface** A count of inventory device interface objects in cache.

Microsoft.Windows.Inventory.Core.AMiTelCacheVersions

This event sends inventory component versions for the Device Inventory data.

The following fields are available:

- **aeinv** The version of the App inventory component.
- **devinv** The file version of the Device inventory component.

Microsoft.Windows.Inventory.Core.InventoryApplicationAdd

This event sends basic metadata about an application on the system to help keep Windows up to date.

The following fields are available:

- **ProgramInstanceId** A hash of the file IDs in an app.
- **Name** The name of the application. Location pulled from depends on 'Source' field.
- **Type** One of ("Application", "Hotfix", "BOE", "Service", "Unknown"). Application indicates Win32 or Appx app, Hotfix indicates app updates (KBs), BOE indicates it's an app with no ARP or MSI entry, Service indicates that it is a service. Application and BOE are the ones most likely seen.
- **Publisher** The Publisher of the application. Location pulled from depends on the 'Source' field.
- **Version** The version number of the program.
- **Language** The language code of the program.
- **Source** How the program was installed (ARP, MSI, Appx, etc...)
- **MsiProductCode** A GUID that describe the MSI Product.
- **MsiPackageCode** A GUID that describes the MSI Package. Multiple 'Products' (apps) can make up an MsiPackage.
- **HiddenArp** Indicates whether a program hides itself from showing up in ARP.
- **OSVersionAtInstallTime** The four octets from the OS version at the time of the application's install.
- **RootDirPath** The path to the root directory where the program was installed.
- **InstallDate** The date the application was installed (a best guess based on folder creation date heuristics)
- **InstallDateMsi** The install date if the application was installed via MSI. Passed as an array.
- **InstallDateFromLinkFile** The estimated date of install based on the links to the files. Passed as an array.
- **InstallDateArpLastModified** The date of the registry ARP key for a given application. Hints at install date but not always accurate. Passed as an array.
- **PartB_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **objectInstanceId** ProgramId (a hash of Name, Version, Publisher, and Language of an application used to identify it).
- **PackageFullName** The package full name for a Store application.
- **InventoryVersion** The version of the inventory file generating the events.
- **StoreAppType** A sub-classification for the type of Microsoft Store app, such as UWP or Win8StoreApp.

Microsoft.Windows.Inventory.Core.InventoryApplicationRemove

This event indicates that a new set of InventoryDevicePnpAdd events will be sent.

The following fields are available:

- **PartB_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Core.InventoryApplicationStartSync

This event indicates that a new set of InventoryApplicationAdd events will be sent.

The following fields are available:

- **PartB_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Core.InventoryDeviceContainerAdd

This event sends basic metadata about a device container (such as a monitor or printer as opposed to a PNP device) to help keep Windows up-to-date.

The following fields are available:

- **ModelName** The model name.
- **ModelId** A model GUID.
- **PrimaryCategory** The primary category for the device container.
- **Categories** A comma separated list of functional categories in which the container belongs.
- **IsConnected** For a physically attached device, this value is the same as IsPresent. For wireless a device, this value represents a communication link.
- **IsActive** Is the device connected, or has it been seen in the last 14 days?
- **IsPaired** Does the device container require pairing?
- **IsNetworked** Is this a networked device?
- **IsMachineContainer** Is the container the root device itself?
- **FriendlyName** The name of the device container.
- **DiscoveryMethod** The discovery method for the device container.
- **ModelNumber** The model number for the device container.
- **Manufacturer** The manufacturer name for the device container.
- **PartB_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **objectInstanceId** ContainerId
- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Core.InventoryDeviceContainerRemove

This event indicates that the InventoryDeviceContainer object is no longer present.

The following fields are available:

- **PartB_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Core.InventoryDeviceContainerStartSync

This event indicates that a new set of InventoryDeviceContainerAdd events will be sent.

The following fields are available:

- **PartB_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Core.InventoryDeviceInterfaceAdd

This event retrieves information about what sensor interfaces are available on the device.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.
- **Accelerometer3D** Indicates if an Accelerator3D sensor is found.
- **ActivityDetection** Indicates if an Activity Detection sensor is found.
- **AmbientLight** Indicates if an Ambient Light sensor is found.

- **Barometer** Indicates if a Barometer sensor is found.
- **Custom** Indicates if a Custom sensor is found.
- **FloorElevation** Indicates if a Floor Elevation sensor is found.
- **GeomagneticOrientation** Indicates if a Geo Magnetic Orientation sensor is found.
- **GravityVector** Indicates if a Gravity Detector sensor is found.
- **Gyrometer3D** Indicates if a Gyrometer3D sensor is found.
- **Humidity** Indicates if a Humidity sensor is found.
- **LinearAccelerometer** Indicates if a Linear Accelerometer sensor is found.
- **Magnetometer3D** Indicates if a Magnetometer3D sensor is found.
- **Orientation** Indicates if an Orientation sensor is found.
- **Pedometer** Indicates if a Pedometer sensor is found.
- **Proximity** Indicates if a Proximity sensor is found.
- **RelativeOrientation** Indicates if a Relative Orientation sensor is found.
- **SimpleDeviceOrientation** Indicates if a Simple Device Orientation sensor is found.
- **Temperature** Indicates if a Temperature sensor is found.

Microsoft.Windows.Inventory.Core.InventoryDeviceInterfaceStartSync

This event indicates that a new set of InventoryDeviceInterfaceAdd events will be sent.

The following fields are available:

- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Core.InventoryDeviceMediaClassAdd

This event sends additional metadata about a PNP device that is specific to a particular class of devices to help keep Windows up to date while reducing overall size of data payload.

The following fields are available:

- **PartB_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.
- **Audio_CaptureDriver** The Audio device capture driver endpoint.
- **Audio_RenderDriver** The Audio device render driver endpoint.

Microsoft.Windows.Inventory.Core.InventoryDeviceMediaClassRemove

This event indicates that the InventoryDeviceMediaClassRemove object is no longer present.

The following fields are available:

- **PartB_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Core.InventoryDeviceMediaClassStartSync

This event indicates that a new set of InventoryDeviceMediaClassSAdd events will be sent.

The following fields are available:

- **PartB_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Core.InventoryDevicePnpAdd

This event sends basic metadata about a PNP device and its associated driver to help keep Windows up-to-date.

The following fields are available:

- **HWID** A JSON array that provides the value and order of the HWID tree for the device.
- **COMPID** A JSON array that provides the value and order of the compatible ID tree for the device.
- **InstallState** The device installation state. One of these values: <https://msdn.microsoft.com/en-us/library/windows/hardware/ff543130.aspx>
- **Enumerator** The bus that enumerated the device.
- **ContainerId** A system-supplied GUID that uniquely groups the functional devices associated with a single-function or multifunction device installed in the device.
- **DeviceState** DeviceState is a bitmask of the following: DEVICE_IS_CONNECTED 0x0001 (currently only for container). DEVICE_IS_NETWORK_DEVICE 0x0002 (currently only for container). DEVICE_IS_PAIED 0x0004 (currently only for container). DEVICE_IS_ACTIVE 0x0008 (currently never set). DEVICE_IS_MACHINE 0x0010 (currently only for container). DEVICE_IS_PRESENT 0x0020 (currently always set). DEVICE_IS_HIDDEN 0x0040. DEVICE_IS_PRINTER 0x0080 (currently only for container). DEVICE_IS_WIRELESS 0x0100. DEVICE_IS_WIRELESS_FAT 0x0200. The most common values are therefore: 32 (0x20)= device is present. 96 (0x60)= device is present but hidden. 288 (0x120)= device is a wireless device that is present.
- **ParentId** Device instance id of the parent of the device.
- **STACKID** A JSON array that provides the value and order of the STACKID tree for the device.
- **Description** The device description.
- **MatchingID** Represents the hardware ID or compatible ID that Windows uses to install a device instance.
- **Class** The device setup class of the driver loaded for the device.
- **ClassGuid** The device setup class guid of the driver loaded for the device.
- **Manufacturer** The device manufacturer.
- **Model** The device model.
- **Inf** The INF file name.
- **DriverVerVersion** The version of the driver loaded for the device.
- **DriverVerDate** The date of the driver loaded for the device.
- **Provider** The device provider.
- **DriverPackageStrongName** The immediate parent directory name in the Directory field of InventoryDriverPackage.
- **Service** The device service name.
- **LowerClassFilters** Lower filter class drivers IDs installed for the device.
- **LowerFilters** Lower filter drivers IDs installed for the device.
- **UpperClassFilters** Upper filter class drivers IDs installed for the device.
- **UpperFilters** Upper filter drivers IDs installed for the device.
- **PartB_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **DriverId** A unique identifier for the installed device.
- **DriverName** The name of the driver image file.
- **InventoryVersion** The version of the inventory file generating the events.
- **ProblemCode** The current error code for the device.

Microsoft.Windows.Inventory.Core.InventoryDevicePnpRemove

This event indicates that the InventoryDevicePnpRemove object is no longer present.

The following fields are available:

- **PartB_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Core.InventoryDevicePnpStartSync

This event indicates that a new set of InventoryDevicePnpAdd events will be sent.

The following fields are available:

- **PartB_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Core.InventoryDriverBinaryAdd

This event sends basic metadata about driver files running on the system to help keep Windows up-to-date.

The following fields are available:

- **DriverName** The file name of the driver.
- **Inf** The name of the INF file.
- **DriverPackageStrongName** The strong name of the driver package.
- **DriverCompany** The company name that developed the driver.
- **DriverCheckSum** The checksum of the driver file.
- **DriverTimeStamp** The low 32 bits of the time stamp of the driver file.
- **DriverType** A bitfield of driver attributes: 1. define DRIVER_MAP_DRIVER_TYPE_PRINTER 0x0001. 2. define DRIVER_MAP_DRIVER_TYPE_KERNEL 0x0002. 3. define DRIVER_MAP_DRIVER_TYPE_USER 0x0004. 4. define DRIVER_MAP_DRIVER_IS_SIGNED 0x0008. 5. define DRIVER_MAP_DRIVER_IS_INBOX 0x0010. 6. define DRIVER_MAP_DRIVER_IS_WINQUAL 0x0040. 7. define DRIVER_MAP_DRIVER_IS_SELF_SIGNED 0x0020. 8. define DRIVER_MAP_DRIVER_IS_CI_SIGNED 0x0080. 9. define DRIVER_MAP_DRIVER_HAS_BOOT_SERVICE 0x0100. 10. define DRIVER_MAP_DRIVER_TYPE_I386 0x10000. 11. define DRIVER_MAP_DRIVER_TYPE_IA64 0x20000. 12. define DRIVER_MAP_DRIVER_TYPE_AMD64 0x40000. 13. define DRIVER_MAP_DRIVER_TYPE_ARM 0x100000. 14. define DRIVER_MAP_DRIVER_TYPE_THUMB 0x200000. 15. define DRIVER_MAP_DRIVER_TYPE_ARMNT 0x400000. 16. define DRIVER_MAP_DRIVER_IS_TIME_STAMPED 0x800000.
- **DriverInBox** Is the driver included with the operating system?
- **DriverSigned** Is the driver signed?
- **DriverIsKernelMode** Is it a kernel mode driver?
- **DriverVersion** The version of the driver file.
- **ImageSize** The size of the driver file.
- **Product** The product name that is included in the driver file.
- **ProductVersion** The product version that is included in the driver file.
- **WdfVersion** The Windows Driver Framework version.
- **Service** The name of the service that is installed for the device.
- **PartB_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Core.InventoryDriverBinaryRemove

This event indicates that the InventoryDriverBinary object is no longer present.

The following fields are available:

- **PartB_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Core.InventoryDriverBinaryStartSync

This event indicates that a new set of InventoryDriverBinaryAdd events will be sent.

The following fields are available:

- **PartB_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Core.InventoryDriverPackageAdd

This event sends basic metadata about drive packages installed on the system to help keep Windows up-to-date.

The following fields are available:

- **Inf** The INF name of the driver package.
- **ClassGuid** The class GUID for the device driver.
- **Class** The class name for the device driver.
- **Directory** The path to the driver package.
- **Date** The driver package date.
- **Version** The version of the driver package.
- **Provider** The provider for the driver package.
- **SubmissionId** The HLK submission ID for the driver package.
- **PartB_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Core.InventoryDriverPackageRemove

This event indicates that the InventoryDriverPackageRemove object is no longer present.

The following fields are available:

- **PartB_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Core.InventoryDriverPackageStartSync

This event indicates that a new set of InventoryDriverPackageAdd events will be sent.

The following fields are available:

- **PartB_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **InventoryVersion** The version of the inventory file generating the events.

Microsoft.Windows.Inventory.Indicators.Checksum

This event summarizes the counts for the InventoryMiscellaneousUexIndicatorAdd events.

The following fields are available:

- **ChecksumDictionary** A count of each operating system indicator.
- **PCFP** Equivalent to the InventoryId field that is found in other core events.

Microsoft.Windows.Inventory.Indicators.InventoryMiscellaneousUexIndicatorAdd

These events represent the basic metadata about the OS indicators installed on the system which are used for keeping the device up-to-date.

The following fields are available:

- **PartB_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.
- **IndicatorValue** The indicator value

Microsoft.Windows.Inventory.Indicators.InventoryMiscellaneousUexIndicatorRemove

This event is a counterpart to InventoryMiscellaneousUexIndicatorAdd, indicating that the item has been removed.

There are no additional unique fields in this event.

The following fields are available:

- **PartB_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.

Microsoft.Windows.Inventory.Indicators.InventoryMiscellaneousUexIndicatorStartSync

This event indicates that a new set of InventoryMiscellaneousUexIndicatorAdd events will be sent.

The following fields are available:

- **PartB_Ms.Device.DeviceInventoryChange** See the Common Data Fields section.

OneDrive events

Microsoft.OneDrive.Sync.Setup.APIOperation

This event includes basic data about install and uninstall OneDrive API operations.

The following fields are available:

- **APIName** The name of the API.
- **ScenarioName** The name of the scenario.
- **Duration** How long the operation took.
- **isSuccess** Was the operation successful?
- **ResultCode** The result code.

Microsoft.OneDrive.Sync.Setup.EndExperience

This event includes a success or failure summary of the installation.

The following fields are available:

- **APIName** The name of the API.
- **ScenarioName** The name of the scenario.
- **Hresult** The HResult of the operation.
- **isSuccess** Was the operation successful?

Microsoft.OneDrive.Sync.Setup.OSUpgradeInstallationOperation

This event is related to the OS version when the OS is upgraded with OneDrive installed.

The following fields are available:

- **HResult** The HResult of the operation.
- **SourceOSVersion** The source version of the operating system.
- **SourceOSBuildNumber** The source build number of the operating system.
- **SourceOSBuildBranch** The source branch of the operating system.
- **CurrentOSVersion** The current version of the operating system.
- **CurrentOSBuildNumber** The current build number of the operating system.
- **CurrentOSBuildBranch** The current branch of the operating system.
- **CurrentOneDriveVersion** The current version of OneDrive.

Microsoft.OneDrive.Sync.Setup.RegisterStandaloneUpdaterAPIOperation

This event is related to registering or unregistering the OneDrive update task.

The following fields are available:

- **APIName** The name of the API.
- **ScenarioName** The name of the scenario.
- **UnregisterOldTaskResult** The HResult of the UnregisterOldTask operation.
- **RegisterNewTaskResult** The HResult of the RegisterNewTask operation.
- **isSuccess** Was the operation successful?

Microsoft.OneDrive.Sync.Setup.SetupCommonData

This event contains basic OneDrive configuration data that helps to diagnose failures.

The following fields are available:

- **AppVersion** The version of the app.
- **OfficeVersion** The version of Office that is installed.
- **BuildArch** Is the architecture x86 or x64?
- **Market** Which market is this in?
- **OneDriveDeviceId** The OneDrive device ID.
- **MachineGuid** The CEIP machine ID.
- **IsMSFTInternal** Is this an internal Microsoft device?
- **OSDeviceName** Only if the device is internal to Microsoft, the device name.
- **OSUserName** Only if the device is internal to Microsoft, the user name.
- **Environment** Is the device on the production or int service?
- **OfficeVersionString** The version of Office that is installed.
- **BuildArchitecture** Is the architecture x86 or x64?
- **UserGuid** The CEIP user ID.
- **MSFTInternal** Is this an internal Microsoft device?

Microsoft.OneDrive.Sync.Updater.CommonData

This event contains basic OneDrive configuration data that helps to diagnose failures.

The following fields are available:

- **AppVersion** The version of the app.
- **OfficeVersion** The version of Office that is installed.
- **BuildArch** Is the architecture x86 or x64?
- **Market** Which market is this in?
- **OneDriveDeviceId** The OneDrive device ID.
- **MachineGuid** The CEIP machine ID.
- **IsMSFTInternal** Is this an internal Microsoft device?
- **OSDeviceName** Only if the device is internal to Microsoft, the device name.
- **OSUserName** Only if the device is internal to Microsoft, the user name.
- **Environment** Is the device on the production or int service?
- **UserGuid** A unique global user identifier.

Microsoft.OneDrive.Sync.Updater.ComponentInstallState

This event determines the installation state of dependent OneDrive components.

The following fields are available:

- **ComponentName** The name of the dependent component.
- **isInstalled** Is the dependent component installed?

Microsoft.OneDrive.Sync.Updater.OfficeRegistration

This event determines the status of the OneDrive integration with Microsoft Office.

The following fields are available:

- **isValid** Is the Microsoft Office registration valid?

Microsoft.OneDrive.Sync.Updater.OverlayIconStatus

This event indicates if the OneDrive overlay icon is working correctly. 0 = healthy; 1 = can be fixed; 2 = broken

The following fields are available:

- **32bit** The status of the OneDrive overlay icon on a 32-bit operating system.
- **64bit** The status of the OneDrive overlay icon on a 64-bit operating system.

Microsoft.OneDrive.Sync.Updater.RepairResult

This event determines the result of the installation repair.

The following fields are available:

- **hr** The HResult of the operation.

Microsoft.OneDrive.Sync.Updater.SetupBinaryDownloadHResult

This event indicates the status when downloading the OneDrive setup file.

The following fields are available:

- **hr** The HResult of the operation.

Microsoft.OneDrive.Sync.Updater.UpdateOverallResult

This event determines the outcome of the operation.

The following fields are available:

- **UpdaterVersion** The version of the updater.
- **IsLoggingEnabled** Is logging enabled?
- **hr** The HResult of the operation.

Microsoft.OneDrive.Sync.Updater.UpdateTierReg

This event determines status of the update tier registry values.

The following fields are available:

- **regReadEnterpriseHr** The HResult of the enterprise reg read value.
- **regReadTeamHr** The HResult of the team reg read value.

Microsoft.OneDrive.Sync.Updater.UpdateXmlDownloadHResult

This event determines the status when downloading the OneDrive update configuration file.

The following fields are available:

- **hr** The HResult of the operation.

Microsoft.OneDrive.Sync.Updater.WebConnectionStatus

This event determines the error code that was returned when verifying Internet connectivity.

The following fields are available:

- **winInetError** The HResult of the operation.

Setup events

SetupPlatformTel.SetupPlatformTelActivityEvent

This event sends a unique ID that can be used to bind Setup Platform events together, to help keep Windows up to date.

The following fields are available:

- **FieldName** Retrieves the event name/data point. Examples: InstallStartTime, InstallEndtime, OverallResult etc.
- **GroupName** Retrieves the groupname the event belongs to. Example: Install Information, DU Information, Disk Space Information etc.
- **Value** Retrieves the value associated with the corresponding event name. For example: For time-related events, this will include the system time.
- **ActivityId** Provides a unique Id to correlate events that occur between a activity start event, and a stop event
- **ActivityName** Provides a friendly name of the package type that belongs to the ActivityId (Setup, LanguagePack, GDR, Driver, etc.)

SetupPlatformTel.SetupPlatformTelActivityStarted

This event sends basic metadata about the update installation process generated by SetupPlatform to help keep Windows up to date.

The following fields are available:

- **Name** The name of the dynamic update type. Example: GDR driver

SetupPlatformTel.SetupPlatformTelActivityStopped

This event sends basic metadata about the update installation process generated by SetupPlatform to help keep Windows up to date.

SetupPlatformTel.SetupPlatformTelEvent

This service retrieves events generated by SetupPlatform, the engine that drives the various deployment scenarios.

The following fields are available:

- **FieldName** Retrieves the event name/data point. Examples: InstallStartTime, InstallEndtime, OverallResult etc.
- **Value** Retrieves the value associated with the corresponding event name (Field Name). For example: For time related events this will include the system time.
- **GroupName** Retrieves the groupname the event belongs to. Example: Install Information, DU Information, Disk Space Information etc.

Shared PC events

Microsoft.Windows.SharedPC.AccountManager.DeleteUserAccount

Activity for deletion of a user account for devices set up for Shared PC mode as part of the Transient Account Manager to help keep Windows up to date. Deleting unused user accounts on shared devices frees up disk space to improve Windows Update success rates.

The following fields are available:

- **wilActivity** Windows Error Reporting data collected when there is a failure in deleting a user account with the Transient Account Manager.
- **userSid** The security identifier of the account.
- **accountType** The type of account that was deleted. Example: AD, AAD, or Local

Microsoft.Windows.SharedPC.AccountManager.SinglePolicyEvaluation

Activity for run of the Transient Account Manager that determines if any user accounts should be deleted for devices set up for Shared PC mode to help keep Windows up to date. Deleting unused user accounts on shared devices frees up disk space to improve Windows Update success rates

The following fields are available:

- **wilActivity** Windows Error Reporting data collected when there is a failure in evaluating accounts to be deleted with the Transient Account Manager.

- **totalAccountCount** The number of accounts on a device after running the Transient Account Manager policies.
- **evaluationTrigger** When was the Transient Account Manager policies ran? Example: At log off or during maintenance hours

Software update events

SoftwareUpdateClientTelemetry.CheckForUpdates

This event sends tracking data about the software distribution client check for content that is applicable to a device, to help keep Windows up to date

The following fields are available:

- **EventScenario** Indicates the purpose of sending this event - whether because the software distribution just started checking for content, or whether it was cancelled, succeeded, or failed.
- **EventInstanceId** A globally unique identifier for event instance.
- **DeviceModel** What is the device model.
- **BiosName** The name of the device BIOS.
- **BIOSVendor** The vendor of the BIOS.
- **BiosVersion** The version of the BIOS.
- **BiosReleaseDate** The release date of the device BIOS.
- **SystemBIOSMajorRelease** Major version of the BIOS.
- **SystemBIOSMinorRelease** Minor version of the BIOS.
- **BiosFamily** The family of the BIOS (Basic Input Output System).
- **BiosSKUNumber** The sku number of the device BIOS.
- **ClientVersion** The version number of the software distribution client.
- **WUDeviceID** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue.
- **CallerApplicationName** The name provided by the caller who initiated API calls into the software distribution client.
- **ProcessName** The process name of the caller who initiated API calls, in the event where CallerApplicationName was not provided.
- **ServiceGuid** An ID which represents which service the software distribution client is checking for content (Windows Update, Microsoft Store, etc.).
- **StatusCode** Indicates the result of a CheckForUpdates event (success, cancellation, failure code HResult).
- **ExtendedStatusCode** Secondary error code for certain scenarios where StatusCode wasn't specific enough.
- **FlightRing** The ring (speed of getting builds) that a device is on if participating in flighting (pre-release builds).
- **FlightBranch** The branch that a device is on if participating in flighting (pre-release builds).
- **RelatedCV** The previous Correlation Vector that was used before swapping with a new one
- **IsWUfBEnabled** Indicates if Windows Update for Business is enabled on the device.
- **IsWUfBDualScanEnabled** Indicates if Windows Update for Business dual scan is enabled on the device.
- **ShippingMobileOperator** The mobile operator that a device shipped on.
- **CurrentMobileOperator** The mobile operator the device is currently connected to.
- **HomeMobileOperator** The mobile operator that the device was originally intended to work with.
- **PhonePreviewEnabled** Indicates whether a phone was getting preview build, prior to flighting (pre-release builds) being introduced.
- **ActivityMatchingId** Contains a unique ID identifying a single CheckForUpdates session from initialization to completion.
- **SyncType** Describes the type of scan the event was

- **IPVersion** Indicates whether the download took place over IPv4 or IPv6
- **NumberOfApplicationsCategoryScanEvaluated** The number of categories (apps) for which an app update scan checked
- **ScanDurationInSeconds** The number of seconds a scan took
- **ScanEnqueueTime** The number of seconds it took to initialize a scan
- **NumberOfLoop** The number of round trips the scan required
- **NumberOfUpdatesEvaluated** The total number of updates which were evaluated as a part of the scan
- **NumberOfNewUpdatesFromServiceSync** The number of updates which were seen for the first time in this scan
- **ServiceUrl** The environment URL a device is configured to scan with
- **Online** Indicates if this was an online scan.
- **AllowCachedResults** Indicates if the scan allowed using cached results.
- **MetadataIntegrityMode** The mode of the update transport metadata integrity check. 0-Unknown, 1-Ignor, 2-Audit, 3-Enforce
- **TotalNumMetadataSignatures** The total number of metadata signatures checks done for new metadata that was synced down.
- **NumFailedMetadataSignatures** The number of metadata signatures checks which failed for new metadata synced down.
- **MSIError** The last error that was encountered during a scan for updates.
- **DriverError** The error code hit during a driver scan. This is 0 if no error was encountered.
- **FailedUpdatesCount** The number of updates that failed to be evaluated during the scan.
- **FailedUpdateGuids** The GUIDs for the updates that failed to be evaluated during the scan.
- **CapabilityDetectoidGuid** The GUID for a hardware applicability detectoid that could not be evaluated.
- **ExtendedMetadataCabUrl** Hostname that is used to download an update.
- **CDNId** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue.
- **CDNCountryCode** Two letter country abbreviation for the CDN's location.
- **NetworkConnectivityDetected** Indicates the type of network connectivity that was detected. 0 - IPv4, 1 - IPv6
- **NumberOfApplicableUpdates** The number of updates which were ultimately deemed applicable to the system after the detection process is complete
- **ApplicableUpdateInfo** Metadata for the updates which were detected as applicable
- **WebServiceRetryMethods** Web service method requests that needed to be retried to complete operation.
- **DeferredUpdates** Update IDs which are currently being deferred until a later time
- **BranchReadinessLevel** The servicing branch configured on the device.
- **DeferralPolicySources** Sources for any update deferral policies defined (GPO = 0x10, MDM = 0x100, Flight = 0x1000, UX = 0x10000).
- **QualityUpdateDeferral** The deferral period configured for quality OS updates on the device (in days).
- **QualityUpdatePause** Indicates whether quality OS updates are paused on the device.
- **QualityUpdatePausePeriod** The pause duration configured for quality OS updates on the device (in days).
- **FeatureUpdateDeferral** The deferral period configured for feature OS updates on the device (in days).
- **FeatureUpdatePause** Indicates whether feature OS updates are paused on the device.
- **FeatureUpdatePausePeriod** The pause duration configured for feature OS updates on the device (in days).
- **DriverExclusionPolicy** Indicates if the policy for not including drivers with Windows Update is enabled.
- **TargetMetadataVersion** For self-initiated healing, this is the target version of the SIH engine to download (if needed). If not, the value is null.
- **CachedEngineVersion** For self-initiated healing, the version of the SIH engine that is cached on the device. If the SIH engine does not exist, the value is null.

- **SearchFilter** Contains information indicating filters applied while checking for content applicable to the device. For example, to filter out all content which may require a reboot.
- **IntentPFNs** Intended application-set metadata for atomic update scenarios.
- **PausedUpdates** A list of UpdateIDs which are currently being paused.
- **PauseQualityUpdatesStartTime** If quality OS updates are paused on the device, this is the date and time for the beginning of the pause time window.
- **PauseQualityUpdatesEndTime** If quality OS updates are paused on the device, this is the date and time for the end of the pause time window.
- **PauseFeatureUpdatesStartTime** If feature OS updates are paused on the device, this is the date and time for the beginning of the pause time window.
- **PauseFeatureUpdatesEndTime** If feature OS updates are paused on the device, this is the date and time for the end of the pause time window.
- **Context** Gives context on where the error has occurred. Example: AutoEnable, GetSLSData, AddService, Misc, or Unknown
- **DriverSyncPassPerformed** Were drivers scanned this time?

SoftwareUpdateClientTelemetry.Commit

This event sends data on whether the Update Service has been called to execute an upgrade, to help keep Windows up to date.

The following fields are available:

- **EventScenario** State of call
- **EventInstanceId** A globally unique identifier for event instance.
- **DeviceModel** What is the device model.
- **BiosName** The name of the device BIOS.
- **BIOSVendor** The vendor of the BIOS.
- **BiosVersion** The version of the BIOS.
- **BiosReleaseDate** The release date of the device BIOS.
- **SystemBIOSMajorRelease** Major version of the BIOS.
- **SystemBIOSMinorRelease** Minor version of the BIOS.
- **BiosFamily** The family of the BIOS (Basic Input Output System).
- **BiosSKUNumber** The sku number of the device BIOS.
- **ClientVersion** The version number of the software distribution client.
- **WUDeviceID** UniqueDeviceID
- **ServerId** Identifier for the service to which the software distribution client is connecting, such as Windows Update and Microsoft Store.
- **EventType** Possible values are "Child", "Bundle", or "Driver".
- **UpdateId** Unique Update ID
- **BundleId** Identifier associated with the specific content bundle; should not be all zeros if the bundleID was found.
- **RevisionNumber** Unique revision number of Update
- **HandlerType** Indicates the kind of content (app, driver, windows patch, etc.)
- **BundleRevisionNumber** Identifies the revision number of the content bundle
- **FlightId** The specific id of the flight the device is getting
- **CallerApplicationName** The name provided by the caller who initiated API calls into the software distribution client

SoftwareUpdateClientTelemetry.Download

This event sends tracking data about the software distribution client download of the content for that update, to

help keep Windows up to date.

The following fields are available:

- **EventScenario** Indicates the purpose of sending this event - whether because the software distribution just started downloading content, or whether it was cancelled, succeeded, or failed.
- **EventInstanceId** A globally unique identifier for event instance.
- **DeviceModel** What is the device model.
- **BiosName** The name of the device BIOS.
- **BIOSVendor** The vendor of the BIOS.
- **BiosVersion** The version of the BIOS.
- **BiosReleaseDate** The release date of the device BIOS.
- **SystemBIOSMajorRelease** Major version of the BIOS.
- **SystemBIOSMinorRelease** Minor version of the BIOS.
- **BiosFamily** The family of the BIOS (Basic Input Output System).
- **BiosSKUNumber** The sku number of the device BIOS.
- **ClientVersion** The version number of the software distribution client.
- **WUDeviceID** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue.
- **CallerApplicationName** The name provided by the caller who initiated API calls into the software distribution client.
- **ProcessName** The process name of the caller who initiated API calls, in the event where CallerApplicationName was not provided.
- **ServiceGuid** An ID which represents which service the software distribution client is installing content for (Windows Update, Microsoft Store, etc.).
- **StatusCode** Indicates the result of a Download event (success, cancellation, failure code HResult).
- **ExtendedStatusCode** Secondary error code for certain scenarios where StatusCode wasn't specific enough.
- **FlightRing** The ring (speed of getting builds) that a device is on if participating in flighting (pre-release builds).
- **FlightBranch** The branch that a device is on if participating in flighting (pre-release builds).
- **RelatedCV** The previous Correlation Vector that was used before swapping with a new one
- **IsWUfBEnabled** Indicates if Windows Update for Business is enabled on the device.
- **IsWUfBDualScanEnabled** Indicates if Windows Update for Business dual scan is enabled on the device.
- **ShippingMobileOperator** The mobile operator that a device shipped on.
- **CurrentMobileOperator** The mobile operator the device is currently connected to.
- **HomeMobileOperator** The mobile operator that the device was originally intended to work with.
- **PhonePreviewEnabled** Indicates whether a phone was opted-in to getting preview builds, prior to flighting (pre-release builds) being introduced.
- **IPVersion** Indicates whether the download took place over IPv4 or IPv6.
- **NetworkCostBitMask** Indicates what kind of network the device is connected to (roaming, metered, over data cap, etc.)
- **NetworkRestrictionStatus** More general version of NetworkCostBitMask, specifying whether Windows considered the current network to be "metered."
- **TimeToEstablishConnection** Time (in ms) it took to establish the connection prior to beginning downloaded.
- **HostName** The hostname URL the content is downloading from.
- **CDNId** ID which defines which CDN the software distribution client downloaded the content from.
- **CDNCountryCode** Two letter country abbreviation for the CDN's location.
- **ActiveDownloadTime** How long the download took, in seconds, excluding time where the update wasn't actively being downloaded.
- **IsDependentSet** Indicates whether a driver is a part of a larger System Hardware/Firmware Update

- **TargetingVersion** For drivers targeted to a specific device model, this is the version number of the drivers being distributed to the device.
- **HardwareId** If this download was for a driver targeted to a particular device model, this ID indicates the model of the device.
- **UpdateImportance** Indicates whether a piece of content was marked as Important, Recommended, or Optional.
- **TargetGroupId** For drivers targeted to a specific device model, this ID indicates the distribution group of devices receiving that driver.
- **RepeatFailFlag** Indicates whether this specific piece of content had previously failed to download.
- **BytesDownloaded** How many bytes were downloaded for an individual piece of content (not the entire bundle).
- **TotalExpectedBytes** The total count of bytes that the download is expected to be.
- **ThrottlingServiceHRESULT** Result code (success/failure) while contacting a web service to determine whether this device should download content yet.
- **EventType** Possible values are Child, Bundle, or Driver.
- **UpdateId** An identifier associated with the specific piece of content.
- **RevisionNumber** Identifies the revision number of this specific piece of content.
- **BundleId** Identifier associated with the specific content bundle; should not be all zeros if the bundleID was found.
- **BundleRevisionNumber** Identifies the revision number of the content bundle.
- **HandlerType** Indicates what kind of content is being downloaded (app, driver, windows patch, etc.).
- **DownloadPriority** Indicates whether a download happened at background, normal, or foreground priority.
- **FlightId** The specific id of the flight (pre-release build) the device is getting.
- **Setup360Phase** If the download is for an operating system upgrade, this datapoint indicates which phase of the upgrade is underway.
- **UsedDO** Whether the download used the delivery optimization service.
- **CbsDownloadMethod** Indicates whether the download was a full-file download or a partial/delta download.
- **UsedSystemVolume** Indicates whether the content was downloaded to the device's main system storage drive, or an alternate storage drive.
- **FlightBuildNumber** If this download was for a flight (pre-release build), this indicates the build number of that flight.
- **BundleBytesDownloaded** How many bytes were downloaded for the specific content bundle.
- **BundleRepeatFailFlag** Indicates whether this particular update bundle had previously failed to download.
- **DownloadScenarioId** A unique ID for a given download used to tie together WU and DO events.
- **PackageFullName** The package name of the content.
- **AppXBlockHashValidationFailureCount** A count of the number of blocks that have failed validation after being downloaded.
- **CachedEngineVersion** For self-initiated healing, the version of the SIH engine that is cached on the device. If the SIH engine does not exist, the value is null.
- **TargetMetadataVersion** For self-initiated healing, this is the target version of the SIH engine to download (if needed). If not, the value is null.
- **DownloadType** Differentiates the download type of SIH downloads between Metadata and Payload downloads.
- **WUSetting** Indicates the users' current updating settings.
- **ProcessorArchitecture** Processor architecture of the system (x86, AMD64, ARM).
- **PlatformRole** The PowerPlatformRole as defined on MSDN
- **IsAOACDevice** Is it Always On, Always Connected?
- **EventNamespaceID** Indicates whether the event succeeded or failed. Has the format EventType+Event where

Event is Succeeded, Cancelled, Failed, etc.

- **Edition** Indicates the edition of Windows being used.
- **DeviceOEM** What OEM does this device belong to.
- **ClientManagedByWSUSServer** Indicates whether the client is managed by Windows Server Update Services (WSUS).
- **QualityUpdatePause** Indicates whether quality OS updates are paused on the device.
- **FeatureUpdatePause** Indicates whether feature OS updates are paused on the device.
- **AppXDownloadScope** Indicates the scope of the download for application content. For streaming install scenarios, AllContent - non-streaming download, RequiredOnly - streaming download requested content required for launch, AutomaticOnly - streaming download requested automatic streams for the app, and Unknown - for events sent before download scope is determined by the Windows Update client.

SoftwareUpdateClientTelemetry.Install

This event sends tracking data about the software distribution client installation of the content for that update, to help keep Windows up to date.

The following fields are available:

- **EventScenario** Indicates the purpose of sending this event - whether because the software distribution just started installing content, or whether it was cancelled, succeeded, or failed.
- **EventInstanceId** A globally unique identifier for event instance.
- **DeviceModel** What is the device model.
- **BiosName** The name of the device BIOS.
- **BIOSVendor** The vendor of the BIOS.
- **BiosVersion** The version of the BIOS.
- **BiosReleaseDate** The release date of the device BIOS.
- **SystemBIOSMajorRelease** Major version of the BIOS.
- **SystemBIOSMinorRelease** Minor version of the BIOS.
- **BiosFamily** The family of the BIOS (Basic Input Output System).
- **BiosSKUNumber** The sku number of the device BIOS.
- **ClientVersion** The version number of the software distribution client.
- **WUDeviceID** The unique identifier of a specific device, used to identify how many devices are encountering success or a particular issue.
- **CallerApplicationName** The name provided by the caller who initiated API calls into the software distribution client.
- **ProcessName** The process name of the caller who initiated API calls, in the event where CallerApplicationName was not provided.
- **ServiceGuid** An ID which represents which service the software distribution client is installing content for (Windows Update, Microsoft Store, etc.).
- **StatusCode** Indicates the result of an installation event (success, cancellation, failure code HResult).
- **ExtendedStatusCode** Secondary error code for certain scenarios where StatusCode wasn't specific enough.
- **FlightRing** The ring that a device is on if participating in the Windows Insider Program.
- **FlightBranch** The branch that a device is on if participating in the Windows Insider Program.
- **RelatedCV** The previous Correlation Vector that was used before swapping with a new one
- **IsWUfBEnabled** Is Windows Update for Business enabled on the device?
- **IsWUfBDualScanEnabled** Is Windows Update for Business dual scan enabled on the device?
- **ShippingMobileOperator** The mobile operator that a device shipped on.
- **CurrentMobileOperator** Mobile operator that device is currently connected to.
- **HomeMobileOperator** The mobile operator that the device was originally intended to work with.

- **PhonePreviewEnabled** Indicates whether a phone was getting preview build, prior to flighting being introduced.
- **TargetGroupId** For drivers targeted to a specific device model, this ID indicates the distribution group of devices receiving that driver.
- **RepeatFailFlag** Indicates whether this specific piece of content had previously failed to install.
- **EventType** Possible values are Child, Bundle, or Driver.
- **TargetingVersion** For drivers targeted to a specific device model, this is the version number of the drivers being distributed to the device.
- **UpdateImportance** Indicates whether a piece of content was marked as Important, Recommended, or Optional.
- **IsFirmware** Is this update a firmware update?
- **IsFinalOutcomeEvent** Does this event signal the end of the update/upgrade process?
- **IsDependentSet** Is the driver part of a larger System Hardware/Firmware update?
- **DriverPingBack** Contains information about the previous driver and system state.
- **ExtendedErrorCode** The extended error code.
- **CSIErrortype** The stage of CBS installation where it failed.
- **MsiAction** The stage of MSI installation where it failed.
- **MsiProductCode** The unique identifier of the MSI installer.
- **TransactionCode** The ID which represents a given MSI installation
- **HardwareId** If this install was for a driver targeted to a particular device model, this ID indicates the model of the device.
- **IsSuccessFailurePostReboot** Did it succeed and then fail after a restart?
- **UpdateId** Unique update ID
- **RevisionNumber** The revision number of this specific piece of content.
- **BundleId** Identifier associated with the specific content bundle; should not be all zeros if the bundleID was found.
- **BundleRevisionNumber** Identifies the revision number of the content bundle.
- **HandlerType** Indicates what kind of content is being installed. Example: app, driver, Windows update
- **FlightId** The specific ID of the Windows Insider build the device is getting.
- **Setup360Phase** If the install is for an operating system upgrade, indicates which phase of the upgrade is underway.
- **UsedSystemVolume** Indicates whether the content was downloaded and then installed from the device's main system storage drive, or an alternate storage drive.
- **FlightBuildNumber** If this installation was for a Windows Insider build, this is the build number of that build.
- **BundleRepeatFailFlag** Has this particular update bundle previously failed to install?
- **PackageFullName** The package name of the content being installed.
- **CachedEngineVersion** For self-initiated healing, the version of the SIH engine that is cached on the device. If the SIH engine does not exist, the value is null.
- **BundleBytesDownloaded** How many bytes were downloaded for the specific content bundle?
- **CbsDownloadMethod** Was the download a full download or a partial download?
- **ClientManagedByWSUSServer** Is the client managed by Windows Server Update Services (WSUS)?
- **DeviceOEM** What OEM does this device belong to.
- **DownloadPriority** The priority of the download activity.
- **DownloadScenaroid** A unique ID for a given download used to tie together WU and DO events.
- **Edition** Indicates the edition of Windows being used.
- **EventNamespaceID** Indicates whether the event succeeded or failed. Has the format EventType+Event where Event is Succeeded, Cancelled, Failed, etc.

- **IsAOACDevice** Is it Always On, Always Connected? (Mobile device usage model)
- **PlatformRole** The PowerPlatformRole as defined on MSDN.
- **ProcessorArchitecture** Processor architecture of the system (x86, AMD64, ARM).
- **RepeatSuccessInstallFlag** Indicates whether this specific piece of content had previously installed successful, for example if another user had already installed it.
- **WUSetting** Indicates the user's current updating settings.
- **IntentPFNs** Intended application-set metadata for atomic update scenarios.
- **QualityUpdatePause** Are quality OS updates paused on the device?
- **FeatureUpdatePause** Are feature OS updates paused on the device?
- **MergedUpdate** Was the OS update and a BSP update merged for installation?

SoftwareUpdateClientTelemetry.SLSDiscovery

This event sends data about the ability of Windows to discover the location of a backend server with which it must connect to perform updates or content acquisition, in order to determine disruptions in availability of update services and provide context for Windows Update errors.

The following fields are available:

- **EventScenario** Indicates the purpose of sending this event - whether because the software distribution just started checking for content, or whether it was cancelled, succeeded, or failed
- **SusClientId** The unique device ID controlled by the software distribution client
- **WUAVersion** The version number of the software distribution client
- **ServiceID** An ID which represents which service the software distribution client is connecting to (Windows Update, Microsoft Store, etc.)
- **UrlPath** Path to the SLS cab that was downloaded
- **HRESULT** Indicates the result code of the event (success, cancellation, failure code HRESULT)
- **IsBackground** Indicates whether the SLS discovery event took place in the foreground or background
- **NextExpirationTime** Indicates when the SLS cab expires

SoftwareUpdateClientTelemetry.UpdateDetected

This event sends data about an AppX app that has been updated from the Microsoft Store, including what app needs an update and what version/architecture is required, in order to understand and address problems with apps getting required updates.

The following fields are available:

- **CallerApplicationName** The name provided by the caller who initiated API calls into the software distribution client
- **ApplicableUpdateInfo** Metadata for the updates which were detected as applicable
- **NumberOfApplicableUpdates** The number of updates which were ultimately deemed applicable to the system after the detection process is complete
- **WUDeviceID** The unique device ID controlled by the software distribution client
- **RelatedCV** The previous Correlation Vector that was used before swapping with a new one
- **EventScenario** Indicates the purpose of sending this event - whether because the software distribution just started checking for content, or whether it was cancelled, succeeded, or failed
- **EventInstanceId** A globally unique identifier for event instance
- **DeviceModel** The device's model as defined in system bios
- **BiosName** The name of the device's system bios
- **BIOSVendor** The vendor of the device's system bios
- **BiosVersion** The version of the device's system bios
- **BiosReleaseDate** The release date of the device's system bios

- **SystemBIOSMajorRelease** The major release version of the device's system system
- **SystemBIOSMinorRelease** The minor release version of the device's system system
- **BiosFamily** The device's family as defined in system bios
- **BiosSKUNumber** The device's SKU as defined in system bios
- **ClientVersion** The version number of the software distribution client
- **ProcessName** The process name of the caller who initiated API calls, in the event where CallerApplicationName was not provided
- **ServiceGuid** An ID which represents which service the software distribution client is connecting to (Windows Update, Microsoft Store, etc.)
- **StatusCode** Indicates the result code of the event (success, cancellation, failure code HResult)
- **ExtendedStatusCode** Secondary status code for certain scenarios where StatusCode wasn't specific enough
- **FlightRing** The ring (speed of getting builds) that a device is on if participating in flighting (pre-release builds).
- **FlightBranch** The branch that a device is on if participating in flighting (pre-release builds).
- **ShippingMobileOperator** The mobile operator that a device shipped on.
- **CurrentMobileOperator** The mobile operator the device is currently connected to.
- **HomeMobileOperator** The mobile operator that the device was originally intended to work with
- **PhonePreviewEnabled** Indicates whether a phone was getting preview build, prior to flighting (pre-release builds) being introduced.
- **ActivityMatchingId** Contains a unique ID identifying a single CheckForUpdates session from initialization to completion
- **SyncType** Describes the type of scan the event was
- **IPVersion** Indicates whether the download took place over IPv4 or IPv6
- **NumberOfApplicationsCategoryScanEvaluated** The number of categories (apps) for which an app update scan checked
- **ScanDurationInSeconds** The number of seconds a scan took
- **ScanEnqueueTime** The number of seconds it took to initialize a scan
- **NumberOfLoop** The number of round trips the scan required
- **NumberOfUpdatesEvaluated** The total number of updates which were evaluated as a part of the scan
- **NumberOfNewUpdatesFromServiceSync** The number of updates which were seen for the first time in this scan
- **ServiceUrl** The environment URL a device is configured to scan with
- **IntentPFNs** Intended application-set metadata for atomic update scenarios.

SoftwareUpdateClientTelemetry.UpdateMetadataIntegrity

This event identifies whether updates have been tampered with and protects against man-in-the-middle attacks.

The following fields are available:

- **EventScenario** The purpose of this event, such as scan started, scan succeeded, or scan failed.
- **ServiceGuid** Identifies the service to which the software distribution client is connected, Example: Windows Update or Microsoft Store
- **MetadataIntegrityMode** The mode of the transport metadata integrity check. 0 = unknown; 1 = ignore; 2 = audit; 3 = enforce
- **StatusCode** The status code of the event.
- **ExtendedStatusCode** The secondary status code of the event.
- **RevisionId** The revision ID for a specific piece of content.
- **UpdateId** The update ID for a specific piece of content.
- **RevisionNumber** The revision number for a specific piece of content.
- **TimestampTokenId** The time this was created. It is encoded in a timestamp blob and will be zero if the token

is malformed.

- **LeafCertId** Integral ID from the FragmentSigning data for certificate that failed.
- **SHA256OfLeafCertPublicKey** A base64 encoding of the hash of the Base64CertData in the FragmentSigning data of the leaf certificate.
- **MetadataSignature** A base64-encoded string of the signature associated with the update metadata (specified by revision ID).
- **SignatureAlgorithm** The hash algorithm for the metadata signature.
- **SHA256OfTimestampToken** A base64-encoded string of hash of the timestamp token blob.
- **ValidityWindowInDays** The validity window that's in effect when verifying the timestamp.
- **TimestampTokenCertThumbprint** The thumbprint of the encoded timestamp token.
- **RawMode** The raw unparsed mode string from the SLS response. This field is null if not applicable.
- **RawValidityWindowInDays** The raw unparsed validity window string in days of the timestamp token. This field is null if not applicable.
- **SHA256OfLeafCerData** A base64 encoding of the hash for the Base64CerData in the FragmentSigning data of the leaf certificate.
- **ListOfSHA256OfIntermediateCerData** A semicolon delimited list of base64 encoding of hashes for the Base64CerData in the FragmentSigning data of an intermediate certificate.
- **EndpointUrl** The endpoint URL where the device obtains update metadata. This is used to distinguish between test, staging, and production environments.
- **SLSPrograms** A test program to which a device may have opted in. Example: Insider Fast

Update events

Update360Telemetry.UpdateAgent_DownloadRequest

This event sends data during the download request phase of updating Windows.

The following fields are available:

- **ErrorCode** The error code returned for the current download request phase.
- **PackageCountTotal** Total number of packages needed.
- **PackageCountRequired** Number of required packages requested.
- **PackageCountOptional** Number of optional packages requested.
- **ObjectId** Unique value for each Update Agent mode.
- **SessionId** Unique value for each Update Agent mode attempt.
- **Scenarioid** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **RelatedCV** Correlation vector value generated from the latest USO scan.
- **Result** Result of the download request phase of update.
- **PackageSizeCanonical** Size of canonical packages in bytes
- **PackageSizeDiff** Size of diff packages in bytes
- **PackageSizeExpress** Size of express packages in bytes
- **FlightId** Unique ID for each flight.
- **UpdateId** Unique ID for each update.
- **PackageCountTotalCanonical** Total number of canonical packages.
- **PackageCountTotalDiff** Total number of diff packages.
- **PackageCountTotalExpress** Total number of express packages.
- **RangeRequestState** Represents the state of the download range request.
- **DeletedCorruptFiles** Indicates if UpdateAgent found any corrupt payload files and whether the payload was deleted.

Update360Telemetry.UpdateAgent_Initialize

This event sends data during the initialize phase of updating Windows.

The following fields are available:

- **ErrorCode** The error code returned for the current initialize phase.
- **SessionData** Contains instructions to update agent for processing FODs and DUICs (Null for other scenarios).
- **UpdateId** Unique ID for each update.
- **FlightId** Unique ID for each flight.
- **FlightMetadata** Contains the FlightId and the build being flighted.
- **ObjectId** Unique value for each Update Agent mode.
- **SessionId** Unique value for each Update Agent mode attempt .
- **Scenarioid** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **RelatedCV** Correlation vector value generated from the latest USO scan.
- **Result** Result of the initialize phase of update. 0 = Succeeded, 1 = Failed, 2 = Cancelled, 3 = Blocked, 4 = BlockCancelled

Update360Telemetry.UpdateAgent_Install

This event sends data during the install phase of updating Windows.

The following fields are available:

- **ErrorCode** The error code returned for the current install phase.
- **ObjectId** Unique value for each Update Agent mode.
- **SessionId** Unique value for each Update Agent mode attempt.
- **Scenarioid** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **RelatedCV** Correlation vector value generated from the latest scan.
- **Result** Result of the install phase of update. 0 = Succeeded 1 = Failed, 2 = Cancelled, 3 = Blocked, 4 = BlockCancelled
- **FlightId** Unique ID for each flight.
- **UpdateId** Unique ID for each update.

Update360Telemetry.UpdateAgent_ModeStart

This event sends data for the start of each mode during the process of updating Windows.

The following fields are available:

- **Mode** Indicates that the Update Agent mode that has started. 1 = Initialize, 2 = DownloadRequest, 3 = Install, 4 = Commit
- **ObjectId** Unique value for each Update Agent mode.
- **SessionId** Unique value for each Update Agent mode attempt.
- **Scenarioid** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **RelatedCV** The correlation vector value generated from the latest scan.
- **FlightId** Unique ID for each flight.
- **UpdateId** Unique ID for each update.

Update360Telemetry.UpdateAgent_SetupBoxLaunch

This event sends data during the launching of the setup box when updating Windows.

The following fields are available:

- **Quiet** Indicates whether setup is running in quiet mode. 0 = false 1 = true
- **ObjectId** Unique value for each Update Agent mode.
- **SessionId** Unique value for each Update Agent mode attempt.
- **Scenarioid** The scenario ID. Example: MobileUpdate, DesktopLanguagePack, DesktopFeatureOnDemand, or DesktopDriverUpdate
- **RelatedCV** Correlation vector value generated from the latest scan.
- **FlightId** Unique ID for each flight.
- **UpdateId** Unique ID for each update.
- **SetupMode** Setup mode 1 = predownload, 2 = install, 3 = finalize
- **SandboxSize** The size of the sandbox folder on the device.

Upgrade events

Setup360Telemetry.Downlevel

This event sends data indicating that the device has invoked the downlevel phase of the upgrade. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **ClientId** If using Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, the default value is Media360, but it can be overwritten by the caller to a unique value.
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe.
- **ReportId** In the Windows Update scenario, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Wuid** This is the Windows Update Client ID. In the Windows Update scenario, this is the same as the clientId.
- **TestId** A string that uniquely identifies a group of events.
- **State** Exit state of given Setup360 run. Example: succeeded, failed, blocked, cancelled
- **HostOsSkuName** The operating system edition which is running Setup360 instance (downlevel OS).
- **HostOSBuildNumber** The build number of the downlevel OS.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. It's an HRESULT error code that can be used to diagnose errors.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **SetupVersionBuildNumber** The build number of Setup360 (build number of the target OS).

Setup360Telemetry.Finalize

This event sends data indicating that the device has invoked the finalize phase of the upgrade, to help keep Windows up-to-date.

The following fields are available:

- **ClientId** With Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe
- **ReportId** With Windows Update, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Wuid** This is the Windows Update Client ID. With Windows Update, this is the same as the clientId.
- **TestId** A string to uniquely identify a group of events.
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **HostOsSkuName** The OS edition which is running Setup360 instance (previous OS).

- **HostOSBuildNumber** The build number of the previous OS.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that is used to diagnose errors.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).

Setup360Telemetry.OsUninstall

The event sends data regarding OS updates and upgrades from Windows 7, Windows 8, and Windows 10. Specifically, the Setup360Telemetry.OSUninstall indicates the outcome of an OS uninstall.

The following fields are available:

- **ClientId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe.
- **ReportId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, this is the GUID for the install.wim.
- **Wuid** Windows Update client ID.
- **TestId** A string to uniquely identify a group of events.
- **State** Exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **HostOsSkuName** The OS edition which is running the Setup360 instance (previous OS).
- **HostOSBuildNumber** The build number of the previous OS.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that is used to diagnose errors.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).

Setup360Telemetry.PostRebootInstall

This event sends data indicating that the device has invoked the postrebootinstall phase of the upgrade, to help keep Windows up-to-date.

The following fields are available:

- **ClientId** With Windows Update, this is the Windows Update client ID that is passed to Setup. In Media setup, the default value is Media360, but can be overwritten by the caller to a unique value.
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe.
- **ReportId** With Windows Update, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Wuid** This is the Windows Update Client ID. With Windows Update, this is the same as ClientId.
- **TestId** A string to uniquely identify a group of events.
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **HostOsSkuName** The OS edition which is running Setup360 instance (previous OS).
- **HostOSBuildNumber** The build number of the previous OS.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that's used to diagnose errors.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential

failure happened

- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).

Setup360Telemetry.PreDownloadQuiet

This event sends data indicating that the device has invoked the predownload quiet phase of the upgrade, to help keep Windows up to date.

The following fields are available:

- **ClientId** Using Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe
- **ReportId** Using Windows Update, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Wuid** This is the Windows Update Client ID. Using Windows Update, this is the same as the clientId.
- **TestId** A string to uniquely identify a group of events.
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, canceled
- **HostOsSkuName** The OS edition which is running Setup360 instance (previous operating system).
- **HostOSBuildNumber** The build number of the previous OS.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that is used to diagnose errors.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).

Setup360Telemetry.PreDownloadUX

The event sends data regarding OS updates and upgrades from Windows 7, Windows 8, and Windows 10.

Specifically, the Setup360Telemetry.PredownloadUX indicates the outcome of the PredownloadUX portion of the update process.

The following fields are available:

- **ClientId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **InstanceId** Unique GUID that identifies each instance of setuphost.exe.
- **ReportId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, this is the GUID for the install.wim.
- **Wuid** Windows Update client ID.
- **TestId** A string to uniquely identify a group of events.
- **State** The exit state of the Setup360 run. Example: succeeded, failed, blocked, cancelled
- **HostOsSkuName** The OS edition which is running the Setup360 instance (previous operating system).
- **HostOSBuildNumber** The build number of the previous operating system.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that can be used to diagnose errors.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **SetupVersionBuildNumber** The build number of Setup360 (build number of the target OS).

Setup360Telemetry.PreInstallQuiet

This event sends data indicating that the device has invoked the preinstall quiet phase of the upgrade, to help keep Windows up to date.

The following fields are available:

- **ClientId** With Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe
- **ReportId** With Windows Update, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Wuid** This is the Windows Update Client ID. With Windows Update, this is the same as the clientId.
- **TestId** A string to uniquely identify a group of events.
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **HostOsSkuName** The OS edition which is running Setup360 instance (previous OS).
- **HostOSBuildNumber** The build number of the previous OS.
- **Setup360Scenario** Setup360 flow type (Boot, Media, Update, MCT)
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback etc.
- **Setup360Result** The result of Setup360. This is an HRESULT error code that can be used to diagnose errors.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).

Setup360Telemetry.PreInstallUX

This event sends data regarding OS updates and upgrades from Windows 7, Windows 8, and Windows 10. Specifically, the Setup360Telemetry.PreinstallUX indicates the outcome of the PreinstallUX portion of the update process.

The following fields are available:

- **ClientId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **InstanceId** A unique GUID that identifies each instance of setuphost.exe.
- **ReportId** For Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, this is the GUID for the install.wim.
- **Wuid** Windows Update client ID.
- **TestId** A string to uniquely identify a group of events.
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **HostOsSkuName** The OS edition which is running the Setup360 instance (previous OS).
- **HostOSBuildNumber** The build number of the previous OS.
- **Setup360Scenario** The Setup360 flow type, Example: Boot, Media, Update, MCT
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that is used to diagnose errors.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).

Setup360Telemetry.Setup360

This event sends data about OS deployment scenarios, to help keep Windows up-to-date.

The following fields are available:

- **InstanceId** Retrieves a unique identifier for each instance of a setup session.

- **ReportId** Retrieves the report ID.
- **FlightData** Specifies a unique identifier for each group of Windows Insider builds.
- **Scenarioid** Retrieves the deployment scenario.
- **FieldName** Retrieves the data point.
- **Value** Retrieves the value associated with the corresponding FieldName.
- **ClientId** Retrieves the upgrade ID: Upgrades via Windows Update - specifies the WU clientID. All other deployment - static string.

Setup360Telemetry.UnexpectedEvent

This event sends data indicating that the device has invoked the unexpected event phase of the upgrade, to help keep Windows up to date.

The following fields are available:

- **ClientId** With Windows Update, this will be the Windows Update client ID that is passed to Setup. In Media setup, default value is Media360, but can be overwritten by the caller to a unique value.
- **Instanceld** A unique GUID that identifies each instance of setuphost.exe
- **ReportId** With Windows Update, this is the updateID that is passed to Setup. In media setup, this is the GUID for the install.wim.
- **Wuid** This is the Windows Update Client ID. With Windows Update, this is the same as the clientId.
- **TestId** A string to uniquely identify a group of events.
- **State** The exit state of a Setup360 run. Example: succeeded, failed, blocked, cancelled
- **HostOsSkuName** The OS edition which is running Setup360 instance (previous OS).
- **HostOSBuildNumber** The build number of the previous OS.
- **Setup360Scenario** The Setup360 flow type. Example: Boot, Media, Update, MCT
- **Setup360Mode** The phase of Setup360. Example: Predownload, Install, Finalize, Rollback
- **Setup360Result** The result of Setup360. This is an HRESULT error code that can be used to diagnose errors.
- **Setup360Extended** Extension of result - more granular information about phase/action when the potential failure happened
- **SetupVersionBuildNumber** The build number of Setup360 (build number of target OS).

Windows Error Reporting events

Microsoft.Windows.WERVertical.OSCrash

This event sends binary data from the collected dump file whenever a bug check occurs, to help keep Windows up to date. This is the OneCore version of this event.

The following fields are available:

- **ReportId** WER Report Id associated with this bug check (used for finding the corresponding report archive in Watson).
- **BugCheckCode** UInt64 "bugcheck code" that identifies a proximate cause of the bug check.
- **BugCheckParameter1** UInt64 parameter providing additional information.
- **BootId** UInt32 identifying the boot number for this device.
- **BugCheckParameter2** UInt64 parameter providing additional information.
- **BugCheckParameter4** UInt64 parameter providing additional information.
- **BugCheckParameter3** UInt64 parameter providing additional information.
- **IsValidDumpFile** True if the dump file is valid for the debugger, false otherwise
- **DumpFileSize** Size of the dump file
- **DumpFileAttributes** Codes that identify the type of data contained in the dump file

Microsoft Store events

Microsoft.Windows.StoreAgent.Telemetry.AbortedInstallation

This event is sent when an installation or update is canceled by a user or the system and is used to help keep Windows Apps up to date and secure.

The following fields are available:

- **PFN** The product family name of the product being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **HResult** The result code of the last action performed before this operation.
- **IsUpdate** Flag indicating if this is an update.
- **AttemptNumber** Number of retry attempts before it was canceled.
- **CategoryId** The Item Category ID.
- **ProductId** The identity of the package or packages being installed.
- **IsInteractive** Was this requested by a user?
- **IsRemediation** Was this a remediation install?
- **BundleId** The Item Bundle ID.
- **IsMandatory** Was this a mandatory update?
- **SystemAttemptNumber** The total number of automatic attempts at installation before it was canceled.
- **UserAttemptNumber** The total number of user attempts at installation before it was canceled.
- **IsRestore** Is this automatically restoring a previously acquired product?
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **IsBundle** Is this a bundle?
- **WUContentId** The Windows Update content ID
- **AggregatedPackageFullNames** The names of all packages to be downloaded and installed.

Microsoft.Windows.StoreAgent.Telemetry.BeginGetInstalledContentIds

This event is sent when an inventory of the apps installed is started to determine whether updates for those apps are available. It's used to help keep Windows up-to-date and secure.

Microsoft.Windows.StoreAgent.Telemetry.BeginUpdateMetadataPrepare

This event is sent when the Store Agent cache is refreshed with any available package updates. It's used to help keep Windows up-to-date and secure.

Microsoft.Windows.StoreAgent.Telemetry.CancelInstallation

This event is sent when an app update or installation is canceled while in interactive mode. This can be canceled by the user or the system. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **IsInteractive** Was this requested by a user?
- **AttemptNumber** Total number of installation attempts.
- **BundleId** The identity of the Windows Insider build that is associated with this product.
- **PreviousHResult** The previous HResult code.
- **ClientAppId** The identity of the app that initiated this operation.
- **CategoryId** The identity of the package or packages being installed.
- **PFN** The name of all packages to be downloaded and installed.
- **ProductId** The name of the package or packages requested for installation.
- **IsUpdate** Is this a product update?
- **IsRemediation** Is this repairing a previous installation?

- **RelatedCV** Correlation Vector of a previous performed action on this product.
- **PreviousInstallState** Previous installation state before it was canceled.
- **IsMandatory** Is this a mandatory update?
- **SystemAttemptNumber** Total number of automatic attempts to install before it was canceled.
- **UserAttemptNumber** Total number of user attempts to install before it was canceled.
- **IsRestore** Is this an automatic restore of a previously acquired product?
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **IsBundle** Is this a bundle?
- **WUContentId** The Windows Update content ID
- **AggregatedPackageFullNames** The names of all package or packages to be downloaded and installed.

Microsoft.Windows.StoreAgent.Telemetry.CompleteInstallOperationRequest

This event is sent after the app installations or updates. It's used to help keep Windows up-to-date and secure

The following fields are available:

- **IsBundle** Is this a bundle?
- **ProductId** The Store Product ID of the product being installed.
- **Skuid** Specific edition of the item being installed.
- **CatalogId** The Store Product ID of the app being installed.
- **PackageFamilyName** The name of the package being installed.
- **HResult** HResult code of the action being performed.

Microsoft.Windows.StoreAgent.Telemetry.EndAcquireLicense

This event is sent after the license is acquired when a product is being installed. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **PFN** Product Family Name of the product being installed.
- **HResult** HResult code to show the result of the operation (success/failure).
- **ProductId** The Store Product ID for the product being installed.
- **IsInteractive** Did the user initiate the installation?
- **CategoryId** The identity of the package or packages being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **IsRemediation** Is this repairing a previous installation?
- **UpdateId** The update ID (if this is an update)
- **AttemptNumber** The total number of attempts to acquire this product.
- **IsUpdate** Is this an update?
- **IsMandatory** Is this a mandatory update?
- **SystemAttemptNumber** The number of attempts by the system to acquire this product.
- **UserAttemptNumber** The number of attempts by the user to acquire this product
- **IsRestore** Is this happening after a device restore?
- **IsBundle** Is this a bundle?
- **WUContentId** The Windows Update content ID
- **ParentBundledId** The product's parent bundle ID.
- **AggregatedPackageFullNames** Includes a set of package full names for each app that is part of an atomic set.

Microsoft.Windows.StoreAgent.Telemetry.EndDownload

This event happens during the app update or installation when content is being downloaded at the end of the

process to report success or failure. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **PFN** The Product Family Name of the app being download.
- **IsRemediation** Is this repairing a previous installation?
- **DownloadSize** The total size of the download.
- **ClientAppId** The identity of the app that initiated this operation.
- **CategoryId** The identity of the package or packages being installed.
- **IsUpdate** Is this an update?
- **HResult** The result code of the last action performed.
- **IsInteractive** Is this initiated by the user?
- **AttemptNumber** Number of retry attempts before it was canceled.
- **BundleId** The identity of the Windows Insider build associated with this product.
- **ProductId** The Store Product ID for the product being installed.
- **IsMandatory** Is this a mandatory installation?
- **SystemAttemptNumber** The number of attempts by the system to download.
- **UserAttemptNumber** The number of attempts by the user to download.
- **IsRestore** Is this a restore of a previously acquired product?
- **ParentBundleId** The parent bundle ID (if it's part of a bundle).
- **IsBundle** Is this a bundle?
- **WUContentId** The Windows Update content ID.
- **ExtendedHResult** Any extended HResult error codes.
- **AggregatedPackageFullNames** The name of all packages to be downloaded and installed.

Microsoft.Windows.StoreAgent.Telemetry.EndFrameworkUpdate

This event happens when an app update requires an updated Framework package and the process starts to download it. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **HResult** The result code of the last action performed before this operation.

Microsoft.Windows.StoreAgent.Telemetry.EndGetInstalledContentIds

This event is sent after sending the inventory of the products installed to determine whether updates for those products are available. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **HResult** The result code of the last action performed before this operation.

Microsoft.Windows.StoreAgent.Telemetry.EndInstall

This event is sent after a product has been installed. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **BundleId** The identity of the build associated with this product.
- **PFN** Product Family Name of the product being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **CategoryId** The identity of the package or packages being installed.
- **ProductId** The Store Product ID for the product being installed.
- **AttemptNumber** The number of retry attempts before it was canceled.
- **HResult** The result code of the last action performed.

- **IsRemediation** Is this repairing a previous installation?
- **IsInteractive** Is this an interactive installation?
- **IsUpdate** Is this an update?
- **IsMandatory** Is this a mandatory installation?
- **SystemAttemptNumber** The total number of system attempts.
- **UserAttemptNumber** The total number of user attempts.
- **IsRestore** Is this automatically restoring a previously acquired product?
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **IsBundle** Is this a bundle?
- **WUContentId** The Windows Update content ID
- **ExtendedHResult** The extended HResult error code.
- **AggregatedPackageFullNames** The names of all packages to be downloaded and installed.

Microsoft.Windows.StoreAgent.Telemetry.EndScanForUpdates

This event is sent after a scan for product updates to determine if there are packages to install. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **HResult** The result code of the last action performed.
- **IsApplicability** Is this request to only check if there are any applicable packages to install?
- **IsInteractive** Is this user requested?
- **ClientAppId** The identity of the app that initiated this operation.
- **IsOnline** Is the request doing an online check?

Microsoft.Windows.StoreAgent.Telemetry.EndSearchUpdatePackages

This event is sent after searching for update packages to install. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **IsRemediation** Is this repairing a previous installation?
- **IsUpdate** Is this an update?
- **ClientAppId** The identity of the app that initiated this operation.
- **HResult** The result code of the last action performed.
- **ProductId** The Store Product ID for the product being installed.
- **AttemptNumber** The total number of retry attempts before it was canceled.
- **IsInteractive** Is this user requested?
- **PFN** The name of the package or packages requested for install.
- **BundleId** The identity of the build associated with this product.
- **CategoryId** The identity of the package or packages being installed.
- **IsMandatory** Is this a mandatory update?
- **SystemAttemptNumber** The total number of system attempts.
- **UserAttemptNumber** The total number of user attempts.
- **IsRestore** Is this restoring previously acquired content?
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **IsBundle** Is this a bundle?
- **WUContentId** The Windows Update content ID
- **AggregatedPackageFullNames** The names of all packages to be downloaded and installed.

Microsoft.Windows.StoreAgent.Telemetry.EndStageUserData

This event is sent between download and installation to see if there is app data that needs to be restored from the cloud. It's used to keep Windows up-to-date and secure.

The following fields are available:

- **IsInteractive** Is this user requested?
- **PFN** The name of the package or packages requested for install.
- **IsUpdate** Is this an update?
- **CategoryId** The identity of the package or packages being installed.
- **HResult** The result code of the last action performed.
- **AttemptNumber** The total number of retry attempts before it was canceled.
- **ProductId** The Store Product ID for the product being installed.
- **BundleId** The identity of the build associated with this product.
- **IsRemediation** Is this repairing a previous installation?
- **ClientAppId** The identity of the app that initiated this operation.
- **IsMandatory** Is this a mandatory update?
- **SystemAttemptNumber** The total number of system attempts.
- **UserAttemptNumber** The total number of system attempts.
- **IsRestore** Is this restoring previously acquired content?
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **IsBundle** Is this a bundle?
- **WUContentId** The Windows Update content ID
- **AggregatedPackageFullNames** The name of all packages to be downloaded and installed.

Microsoft.Windows.StoreAgent.Telemetry.EndUpdateMetadataPrepare

This event happens after a scan for available app updates. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **HResult** The result code of the last action performed.

Microsoft.Windows.StoreAgent.Telemetry.FulfillmentComplete

This event is sent at the end of an app install or update and is used to track the very end of the install or update process.

The following fields are available:

- **ProductId** The product ID of the app that is being updated or installed.
- **PFN** The Package Family Name of the app that is being installed or updated.
- **FailedRetry** Was the installation or update retry successful?
- **HResult** The HResult code of the operation.

Microsoft.Windows.StoreAgent.Telemetry.FulfillmentInitiate

This event is sent at the beginning of an app install or update and is used to track the very beginning of the install or update process.

The following fields are available:

- **ProductId** The product ID of the app that is being updated or installed.
- **PFN** The Package Family Name of the app that is being installed or updated.

Microsoft.Windows.StoreAgent.Telemetry.InstallOperationRequest

This event happens at the beginning of the install process when an app update or new app is installed. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **CatalogId** If this product is from a private catalog, the Store Product ID for the product being installed.
- **BundleId** The identity of the build associated with this product.
- **Skuid** Specific edition ID being installed.
- **ProductId** The Store Product ID for the product being installed.
- **VolumePath** The disk path of the installation.

Microsoft.Windows.StoreAgent.Telemetry.PauseInstallation

This event is sent when a product install or update is paused either by a user or the system. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **RelatedCV** Correlation Vector of a previous performed action on this product.
- **IsRemediation** Is this repairing a previous installation?
- **PreviousHResult** The result code of the last action performed before this operation.
- **ProductId** The Store Product ID for the product being installed.
- **IsUpdate** Is this an update?
- **PreviousInstallState** Previous state before the installation or update was paused.
- **CategoryId** The identity of the package or packages being installed.
- **ClientAppId** The identity of the app that initiated this operation.
- **AttemptNumber** The total number of retry attempts before it was canceled.
- **IsInteractive** Is this user requested?
- **BundleId** The identity of the build associated with this product.
- **PFN** The Product Full Name.
- **IsMandatory** Is this a mandatory update?
- **SystemAttemptNumber** The total number of system attempts.
- **UserAttemptNumber** The total number of user attempts.
- **IsRestore** Is this restoring previously acquired content?
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **IsBundle** Is this a bundle?
- **WUContentId** The Windows Update content ID
- **AggregatedPackageFullNames** The names of all packages to be downloaded and installed.

Microsoft.Windows.StoreAgent.Telemetry.ResumeInstallation

This event happens when a product install or update is resumed either by a user or the system. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **RelatedCV** Correlation Vector for the original install before it was resumed.
- **AttemptNumber** The number of retry attempts before it was canceled.
- **BundleId** The identity of the build associated with this product.
- **PreviousHResult** The previous HResult error code.
- **ClientAppId** The identity of the app that initiated this operation.
- **CategoryId** The identity of the package or packages being installed.
- **PFN** The name of the package or packages requested for install.
- **IsUpdate** Is this an update?
- **PreviousInstallState** Previous state before the installation was paused.
- **IsRemediation** Is this repairing a previous installation?

- **IsInteractive** Is this user requested?
- **ProductId** The Store Product ID for the product being installed.
- **IsMandatory** Is this a mandatory update?
- **SystemAttemptNumber** The total number of system attempts.
- **UserAttemptNumber** The total number of user attempts.
- **IsRestore** Is this restoring previously acquired content?
- **ParentBundleId** The product ID of the parent (if this product is part of a bundle).
- **IsBundle** Is this a bundle?
- **WUContentId** The Windows Update content ID
- **AggregatedPackageFullNames** The names of all packages to be downloaded and installed.
- **IsUserRetry** Did the user initiate the retry?
- **HResult** The result code of the last action performed before this operation.

Microsoft.Windows.StoreAgent.Telemetry.ResumeOperationRequest

This event happens when a product install or update is resumed by a user and on installation retries. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **ProductId** The Store Product ID for the product being installed.

Microsoft.Windows.StoreAgent.Telemetry.SearchForUpdateOperationRequest

This event is sent when searching for update packages to install. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **ProductId** The Store Product ID for the product being installed.
- **Skuid** Specific edition of the app being updated.
- **CatalogId** The Store Product ID for the product being installed.

Microsoft.Windows.StoreAgent.Telemetry.UpdateAppOperationRequest

This event happens when an app for a user needs to be updated. It's used to help keep Windows up-to-date and secure.

The following fields are available:

- **PfamN** The name of the product that is requested for update.

Windows Update Delivery Optimization events

Microsoft.OSG.DU.DeliveryOptClient.DownloadCanceled

This event describes when a download was canceled with Delivery Optimization. It's used to understand and address problems regarding downloads.

The following fields are available:

- **bytesFromIntPeers** The number of bytes received from peers not in the same LAN or in the same group.
- **fileID** The ID of the file being downloaded.
- **sessionID** The ID of the file download session.
- **scenarioID** The ID of the scenario.
- **bytesFromCDN** The number of bytes received from a CDN source.
- **updateID** The ID of the update being downloaded.
- **background** Is the download being done in the background?
- **bytesFromPeers** The number of bytes received from a peer in the same LAN.

- **clientTeId** A random number used for device sampling.
- **bytesFromGroupPeers** The number of bytes received from a peer in the same group.
- **errorCode** The error code that was returned.
- **doErrorCode** The Delivery Optimization error code that was returned.
- **cdnErrorCodes** A list of CDN connection errors since the last FailureCDNCommunication event.
- **cdnErrorCounts** The number of times each error in cdnErrorCodes was encountered.
- **experimentId** When running a test, this is used to correlate events that are part of the same test.
- **isVpn** Is the device connected to a Virtual Private Network?
- **usedMemoryStream** Did the download use memory streaming?

Microsoft.OSG.DU.DeliveryOptClient.DownloadCompleted

This event describes when a download has completed with Delivery Optimization. It's used to understand and address problems regarding downloads.

The following fields are available:

- **sessionId** The ID of the download session.
- **scenarioID** The ID of the scenario.
- **bytesFromIntPeers** The number of bytes received from peers not in the same LAN or in the same domain group.
- **updateID** The ID of the update being downloaded.
- **fileSize** The size of the file being downloaded.
- **bytesFromCDN** The number of bytes received from a CDN source.
- **fileID** The ID of the file being downloaded.
- **background** Is the download a background download?
- **bytesFromPeers** The number of bytes received from a peer in the same LAN.
- **totalTime** How long did the download take (in seconds)?
- **restrictedUpload** Is the upload restricted?
- **clientTeId** A random number used for device sampling.
- **bytesFromGroupPeers** The number of bytes received from a peer in the same domain group.
- **downloadMode** The download mode used for this file download session.
- **doErrorCode** The Delivery Optimization error code that was returned.
- **numPeers** The total number of peers used for this download.
- **cdnConnectionCount** The total number of connections made to the CDN.
- **lanConnectionCount** The total number of connections made to peers in the same LAN.
- **groupConnectionCount** The total number of connections made to peers in the same group.
- **internetConnectionCount** The total number of connections made to peers not in the same LAN or the same group.
- **cdnIp** The IP address of the source CDN.
- **downlinkBps** The maximum measured available download bandwidth (in bytes per second).
- **uplinkBps** The maximum measured available upload bandwidth (in bytes per second).
- **downlinkUsageBps** The download speed (in bytes per second).
- **uplinkUsageBps** The upload speed (in bytes per second).
- **totalTimeMs** Duration of the download (in seconds).
- **cdnErrorCodes** A list of CDN connection errors since the last FailureCDNCommunication event.
- **cdnErrorCounts** The number of times each error in cdnErrorCodes was encountered.
- **bytesRequested** The total number of bytes requested for download.
- **experimentId** When running a test, this is used to correlate with other events that are part of the same test.

- **isVpn** Is the device connected to a Virtual Private Network?
- **usedMemoryStream** Did the download use memory streaming?

Microsoft.OSG.DU.DeliveryOptClient.DownloadPaused

This event represents a temporary suspension of a download with Delivery Optimization. It's used to understand and address problems regarding downloads.

The following fields are available:

- **updateID** The ID of the update being paused.
- **errorCode** The error code that was returned.
- **scenarioID** The ID of the scenario.
- **background** Is the download a background download?
- **sessionId** The ID of the download session.
- **clientTeId** A random number used for device sampling.
- **reasonCode** The reason for pausing the download.
- **fileID** The ID of the file being paused.
- **experimentId** When running a test, this is used to correlate with other events that are part of the same test.
- **isVpn** Is the device connected to a Virtual Private Network?

Microsoft.OSG.DU.DeliveryOptClient.DownloadStarted

This event describes the start of a new download with Delivery Optimization. It's used to understand and address problems regarding downloads.

The following fields are available:

- **experimentId** When running a test, this is used to correlate with other events that are part of the same test.
- **errorCode** The error code that was returned.
- **doErrorCode** The Delivery Optimization error code that was returned.
- **peerID** The ID for this Delivery Optimization client.
- **doClientVersion** The version of the Delivery Optimization client.
- **jobID** The ID of the Windows Update job.
- **sessionId** The ID of the download session.
- **updateID** The ID of the update being downloaded.
- **scenarioID** The ID of the scenario.
- **fileID** The ID of the file being downloaded.
- **cdnUrl** The URL of the CDN.
- **filePath** The path where the file will be written.
- **groupId** ID for the group.
- **background** Is the download a background download?
- **downloadMode** The download mode used for this file download session.
- **minFileSizePolicy** The minimum content file size policy to allow the download using Peering.
- **diceRoll** The dice roll value used in sampling events.
- **deviceProfile** Identifies the usage or form factor. Example: Desktop or Xbox
- **isVpn** Is the device connected to a Virtual Private Network?
- **usedMemoryStream** Did the download use memory streaming?
- **minDiskSizePolicyEnforced** Is the minimum disk size enforced via policy?
- **minDiskSizeGB** The minimum disk size (in GB) required for Peering.
- **clientTeId** A random number used for device sampling.
- **costFlags** A set of flags representing network cost.

Microsoft.OSG.DU.DeliveryOptClient.FailureCdnCommunication

This event represents a failure to download from a CDN with Delivery Optimization. It's used to understand and address problems regarding downloads.

The following fields are available:

- **experimentId** When running a test, this is used to correlate with other events that are part of the same test.
- **fileID** The ID of the file being downloaded.
- **errorCode** The error code that was returned.
- **httpStatusCode** The HTTP status code returned by the CDN.
- **errorCount** The total number of times this error code was seen since the last FailureCdnCommunication event was encountered.
- **sessionId** The ID of the download session.
- **cdnUrl** The URL of the CDN.
- **cdnIp** The IP address of the CDN.
- **cdnHeaders** The HTTP headers returned by the CDN.
- **clientTeId** A random number used for device sampling.
- **isHeadRequest** The type of HTTP request that was sent to the CDN. Example: HEAD or GET
- **requestSize** The size of the range requested from the CDN.
- **responseSize** The size of the range response received from the CDN.

Microsoft.OSG.DU.DeliveryOptClient.JobError

This event represents a Windows Update job error. It allows for investigation of top errors.

The following fields are available:

- **jobID** The Windows Update job ID.
- **fileID** The ID of the file being downloaded.
- **errorCode** The error code returned.
- **clientTeId** A random number used for device sampling.
- **experimentId** When running a test, this is used to correlate with other events that are part of the same test.

Windows Update events

Microsoft.Windows.Update.DataMigrationFramework.DmfMigrationCompleted

This event sends data collected at the end of the Data Migration Framework (DMF) and parameters involved in its invocation, to help keep Windows up to date.

The following fields are available:

- **MigrationEndtime** A system timestamp of when the DMF migration completed.
- **UpdateIds** A collection of GUIDs for updates that are associated with the DMF session.
- **WuClientid** The GUID of the Windows Update client responsible for triggering the DMF migration.
- **MigrationDurationinmilliseconds** How long the DMF migration took (in milliseconds).
- **RevisionNumbers** A collection of revision numbers for the updates associated with the DMF session.

Microsoft.Windows.Update.DataMigrationFramework.DmfMigrationStarted

This event sends data collected at the beginning of the Data Migration Framework (DMF) and parameters involved in its invocation, to help keep Windows up to date.

The following fields are available:

- **UpdateIds** A collection of GUIDs identifying the upgrades that are running.

- **MigrationStarttime** The timestamp representing the beginning of the DMF migration.
- **MigrationOEMphases** The number of OEM-authored migrators scheduled to be ran by DMF for this upgrade.
- **WuClientid** The GUID of the Windows Update client invoking DMF.
- **MigrationMicrosoftphases** The number of Microsoft-authored migrators scheduled to be ran by DMF for this upgrade.
- **RevisionNumbers** A collection of the revision numbers associated with the Updatelds.

Microsoft.Windows.Update.DataMigrationFramework.MigratorResult

This event sends DMF migrator data to help keep Windows up to date.

The following fields are available:

- **MigratorGuid** A GUID identifying the migrator that just completed.
- **RunDurationInSeconds** The time it took for the migrator to complete.
- **CurrentStep** This is the last step the migrator reported before returning a result. This tells us how far through the individual migrator the device was before failure.
- **MigratorName** The name of the migrator that just completed.
- **MigratorId** A GUID identifying the migrator that just completed.
- **ErrorCode** The result (as an HRESULT) of the migrator that just completed.
- **TotalSteps** Migrators report progress in number of completed steps against the total steps. This is the total number of steps.

Microsoft.Windows.Update.Orchestrator.CommitFailed

This events tracks when a device needs to restart after an update but did not.

The following fields are available:

- **wuDeviceid** The Windows Update device GUID.
- **errorCode** The error code that was returned.

Microsoft.Windows.Update.Orchestrator.Detection

This event sends launch data for a Windows Update scan to help keep Windows up to date.

The following fields are available:

- **wuDeviceid** Unique device ID used by Windows Update.
- **revisionNumber** Update revision number.
- **eventScenario** End to end update session ID, or indicates the purpose of sending this event - whether because the software distribution just started installing content, or whether it was cancelled, succeeded, or failed.
- **deferReason** Reason why the device could not check for updates.
- **detectionBlockreason** Reason for detection not completing.
- **interactive** Identifies if session is User Initiated.
- **updateId** Update ID.
- **detectionDeferralReason** A log of deferral reasons for every update state.
- **flightID** A unique update ID.
- **updateScenarioType** The update session type.
- **errorCode** The returned error code.

Microsoft.Windows.Update.Orchestrator.Download

This event sends launch data for a Windows Update download to help keep Windows up to date.

The following fields are available:

- **detectionDeferralReason** Reason for download not completing

- **wuDeviceid** Unique device ID used by Windows Update.
- **interactive** Identifies if session is user initiated.
- **revisionNumber** Update revision number.
- **deferReason** Reason for download not completing
- **updateId** Update ID.
- **eventScenario** End to end update session ID.
- **errorCode** An error code represented as a hexadecimal value
- **flightID** Unique update ID.
- **updateScenarioType** The update session type.

Microsoft.Windows.Update.Orchestrator.FlightInapplicable

This event sends data on whether the update was applicable to the device, to help keep Windows up to date.

The following fields are available:

- **updateId** Unique Update ID
- **revisionNumber** Revision Number of the Update
- **UpdateStatus** Integer that describes Update state
- **EventPublishedTime** time that the event was generated
- **wuDeviceid** Unique Device ID
- **flightID** Unique Update ID
- **updateScenarioType** The update session type.

Microsoft.Windows.Update.Orchestrator.InitiatingReboot

This event sends data about an Orchestrator requesting a reboot from power management to help keep Windows up to date.

The following fields are available:

- **revisionNumber** Revision number of the update.
- **EventPublishedTime** Time of the event.
- **updateId** Update ID.
- **wuDeviceid** Unique device ID used by Windows Update.
- **flightID** Unique update ID
- **interactive** Indicates the reboot initiation stage of the update process was entered as a result of user action or not.
- **rebootOutsideOfActiveHours** Indicates the timing that the reboot was to occur to ensure the correct update process and experience is provided to keep Windows up to date.
- **uxRebootstate** Indicates the exact state of the user experience at the time the required reboot was initiated to ensure the correct update process and experience is provided to keep Windows up to date.
- **updateScenarioType** The update session type.

Microsoft.Windows.Update.Orchestrator.Install

This event sends launch data for a Windows Update install to help keep Windows up to date.

The following fields are available:

- **eventScenario** End to end update session ID.
- **deferReason** Reason for install not completing.
- **interactive** Identifies if session is user initiated.
- **wuDeviceid** Unique device ID used by Windows Update.
- **batteryLevel** Current battery capacity in mWh or percentage left.

- **installCommitFailedTime** The time it took for a reboot to happen but the upgrade failed to progress.
- **errorCode** The error code represented by a hexadecimal value.
- **updateId** Update ID.
- **revisionNumber** Update revision number.
- **flightID** Unique update ID
- **installRebootInitiateTime** The time it took for a reboot to be attempted.
- **flightUpdate** Flight update
- **minutesToCommit** The time it took to install updates.
- **ForcedRebootReminderSet** A boolean value that indicates if a forced reboot will happen for updates.
- **rebootOutsideOfActiveHours** Indicates the timing that the reboot was to occur to ensure the correct update process and experience is provided to keep Windows up to date.
- **uxRebootstate** Indicates the exact state of the user experience at the time the required reboot was initiated to ensure the correct update process and experience is provided to keep Windows up to date.
- **updateScenarioType** The update session type.

Microsoft.Windows.Update.Orchestrator.PostInstall

This event sends data about lite stack devices (mobile, IOT, anything non-PC) immediately before data migration is launched to help keep Windows up to date.

The following fields are available:

- **wuDeviceid** Unique device ID used by Windows Update.
- **eventScenario** End to end update session ID.
- **sessionType** Interactive vs. Background.
- **bundleRevisionnumber** Bundle revision number.
- **batteryLevel** Current battery capacity in mWh or percentage left.
- **bundleId** Update grouping ID.
- **errorCode** Hex code for the error message, to allow lookup of the specific error.
- **flightID** Unique update ID.

Microsoft.Windows.Update.Orchestrator.RebootFailed

This event sends information about whether an update required a reboot and reasons for failure to help keep Windows up to date.

The following fields are available:

- **updateId** Update ID.
- **batteryLevel** Current battery capacity in mWh or percentage left.
- **RebootResults** Hex code indicating failure reason. Typically, we expect this to be a specific USO generated hex code.
- **installRebootDeferreason** Reason for reboot not occurring.
- **revisionNumber** Update revision number.
- **EventPublishedTime** The time that the reboot failure occurred.
- **deferReason** Reason for install not completing.
- **wuDeviceid** Unique device ID used by Windows Update.
- **flightID** Unique update ID.
- **rebootOutsideOfActiveHours** Indicates the timing that the reboot was to occur to ensure the correct update process and experience is provided to keep Windows up to date.
- **uxRebootstate** Indicates the exact state of the user experience at the time the required reboot was initiated to ensure the correct update process and experience is provided to keep Windows up to date.
- **updateScenarioType** The update session type.

Microsoft.Windows.Update.Orchestrator.RestoreRebootTask

This event sends data indicating that a reboot task is missing unexpectedly on a device and the task is restored because a reboot is still required, to help keep Windows up to date.

The following fields are available:

- **RebootTaskRestoredTime** Time at which this reboot task was restored.
- **wuDeviceid** Device id on which the reboot is restored
- **revisionNumber** Update revision number.
- **updateId** Update ID.

Microsoft.Windows.Update.Orchestrator.SystemNeeded

This event sends data about why a device is unable to reboot, to help keep Windows up to date.

The following fields are available:

- **eventScenario** End to end update session ID.
- **wuDeviceid** Unique device ID used by Windows Update.
- **systemNeededReason** Reason ID
- **updateId** Update ID.
- **revisionNumber** Update revision number.
- **rebootOutsideOfActiveHours** Indicates the timing that the reboot was to occur to ensure the correct update process and experience is provided to keep Windows up to date.
- **uxRebootstate** Indicates the exact state of the user experience at the time the required reboot was initiated to ensure the correct update process and experience is provided to keep Windows up to date.
- **updateScenarioType** The update session type.

Microsoft.Windows.Update.Orchestrator.UpdatePolicyCacheRefresh

This event sends data on whether Update Management Policies were enabled on a device, to help keep Windows up to date.

The following fields are available:

- **wuDeviceid** Unique device ID used by Windows Update.
- **policyCacherefreshtime** Refresh time
- **policiesNamevaluesource** Policy Name
- **updateInstallUXsetting** This shows whether a user has set policies via UX option
- **configuredPoliciescount** Policy Count

Microsoft.Windows.Update.Orchestrator.UpdateRebootRequired

This event sends data about whether an update required a reboot to help keep Windows up to date.

The following fields are available:

- **updateId** Update ID.
- **revisionNumber** Update revision number.
- **wuDeviceid** Unique device ID used by Windows Update.
- **flightID** Unique update ID.
- **interactive** Indicates the reboot initiation stage of the update process was entered as a result of user action or not.
- **uxRebootstate** Indicates the exact state of the user experience at the time the required reboot was initiated to ensure the correct update process and experience is provided to keep Windows up to date.
- **updateScenarioType** The update session type.

Microsoft.Windows.Update.UpdateStackServicing.CheckForUpdates

This event sends data about the UpdateStackServicing check for updates, to help keep Windows up to date.

The following fields are available:

- **EventScenario** The scenario of the event. Example: Started, Failed, or Succeeded
- **StatusCode** The HRESULT code of the operation.
- **CallerApplicationName** The name of the USS scheduled task. Example UssScheduled or UssBoot
- **ClientVersion** The version of the client.
- **EventInstanceId** The USS session ID.
- **WUDeviceID** The Windows Update device ID.
- **ServiceGuid** The GUID of the service.
- **BspVersion** The version of the BSP.
- **OemName** The name of the manufacturer.
- **DeviceName** The name of the device.
- **CommercializationOperator** The name of the operator.
- **DetectionVersion** The string returned from the GetDetectionVersion export of the downloaded detection DLL.

Microsoft.Windows.Update.Ux.MusNotification.RebootNoLongerNeeded

This event is sent when a security update has successfully completed.

The following fields are available:

- **UtcTime** The Coordinated Universal Time that the restart was no longer needed.

Microsoft.Windows.Update.Ux.MusNotification.RebootScheduled

This event sends data about a required reboot that is scheduled with no user interaction, to help keep Windows up to date.

The following fields are available:

- **updateId** Update ID of the update that is getting installed with this reboot.
- **ScheduledRebootTime** Time of the scheduled reboot.
- **wuDeviceid** Unique device ID used by Windows Update.
- **revisionNumber** Revision number of the update that is getting installed with this reboot.
- **forcedreboot** True, if a reboot is forced on the device. False, otherwise.
- **rebootArgument** Argument for the reboot task. It also represents specific reboot related action.
- **rebootScheduledByUser** True, if a reboot is scheduled by user. False, if a reboot is scheduled automatically.
- **activeHoursApplicable** True, If Active Hours applicable on this device. False, otherwise.
- **rebootOutsideOfActiveHours** True, if a reboot is scheduled outside of active hours. False, otherwise.
- **rebootState** The state of the reboot.

Microsoft.Windows.Update.Ux.MusNotification.ToastDisplayedToScheduleReboot

This event is sent when a toast notification is shown to the user about scheduling a device restart.

The following fields are available:

- **UtcTime** The Coordinated Universal Time when the toast notification was shown.

Microsoft.Windows.Update.Ux.MusUpdateSettings.RebootScheduled

This event sends basic information for scheduling a device restart to install security updates. It's used to help keep Windows up-to-date.

The following fields are available:

- **ScheduledRebootTime** The time that the device was restarted.
- **updateId** The Windows Update device GUID.
- **revisionNumber** The revision number of the OS being updated.
- **wuDeviceid** The Windows Update device GUID.
- **forcedreboot** Is the restart that's being scheduled a forced restart?
- **rebootArgument** The arguments that are passed to the OS for the restarted.
- **rebootScheduledByUser** Was the restart scheduled by the user? If the value is false, the restart was scheduled by the device.
- **activeHoursApplicable** Is the restart respecting Active Hours?
- **rebootOutsideOfActiveHours** Was the restart scheduled outside of Active Hours?
- **rebootState** The state of the restart.

Winlogon events

Microsoft.Windows.Security.Winlogon.SetupCompleteLogon

This event signals the completion of the setup process. It happens only once during the first logon.

Windows 10 diagnostic data for the Full telemetry level

10/17/2017 • 13 min to read • [Edit Online](#)

Applies to:

- Windows 10, version 1703 and later

Microsoft collects Windows diagnostic data to keep Windows up-to-date, secure, and operating properly. It also helps us improve Windows and, for users who have turned on “tailored experiences”, can be used to provide more relevant tips and recommendations to tailor Microsoft products to the user’s needs. This article describes all types diagnostic data collected by Windows at the Full telemetry level (inclusive of data collected at Basic), with comprehensive examples of data we collect per each type. For additional, detailed technical descriptions of Basic data items, see [Windows 10, version 1709 Basic level diagnostic events and fields](#) and [Windows 10, version 1703 Basic level diagnostic events and fields](#).

The data covered in this article is grouped into the following categories:

- Common Data (diagnostic header information)
- Device, Connectivity, and Configuration data
- Product and Service Usage data
- Product and Service Performance data
- Software Setup and Inventory data
- Browsing History data
- Inking, Typing, and Speech Utterance data

NOTE

The majority of diagnostic data falls into the first four categories.

Common data

Most diagnostic events contain a header of common data:

CATEGORY NAME	EXAMPLES

CATEGORY NAME	EXAMPLES
Common Data	<p>Information that is added to most diagnostic events, if relevant and available:</p> <ul style="list-style-type: none"> • OS name, version, build, and locale • User ID -- a unique identifier associated with the user's Microsoft Account (if one is used) or local account. The user's Microsoft Account identifier is not collected from devices configured to send Basic diagnostic data • Xbox UserID • Environment from which the event was logged -- Application ID of app or component that logged the event, Session GUID. Used to track events over a given period of time such the period an app is running or between boots of the OS. • The diagnostic event name, Event ID, ETW opcode, version, schema signature, keywords, and flags • HTTP header information including IP address. This is not the IP address of the device but the source address in the network packet header received by the diagnostics ingestion service. • Various IDs that are used to correlate and sequence related events together. • Device ID. This is not the user provided device name, but an ID that is unique for that device. • Device class -- Desktop, Server, or Mobile • Event collection time • Diagnostic level -- Basic or Full, Sample level -- for sampled data, what sample level is this device opted into

Device, Connectivity, and Configuration data

This type of data includes details about the device, its configuration and connectivity capabilities, and status.

CATEGORY NAME	EXAMPLES
Device properties	<p>Information about the OS and device hardware, such as:</p> <ul style="list-style-type: none"> • OS - version name, Edition • Installation type, subscription status, and genuine OS status • Processor architecture, speed, number of cores, manufacturer, and model • OEM details --manufacturer, model, and serial number • Device identifier and Xbox serial number • Firmware/BIOS -- type, manufacturer, model, and version • Memory -- total memory, video memory, speed, and how much memory is available after the device has reserved memory • Storage -- total capacity and disk type • Battery -- charge capacity and InstantOn support • Hardware chassis type, color, and form factor • Is this a virtual machine?

CATEGORY NAME	EXAMPLES
Device capabilities	<p>Information about the specific device capabilities such as:</p> <ul style="list-style-type: none"> • Camera -- whether the device has a front facing, a rear facing camera, or both. • Touch screen -- does the device include a touch screen? If so, how many hardware touch points are supported? • Processor capabilities -- CompareExchange128, LahfSahf, NX, PrefetchW, and SSE2 • Trusted Platform Module (TPM) – whether present and what version • Virtualization hardware -- whether an IOMMU is present, SLAT support, is virtualization enabled in the firmware • Voice – whether voice interaction is supported and the number of active microphones • Number of displays, resolutions, DPI • Wireless capabilities • OEM or platform face detection • OEM or platform video stabilization and quality level set • Advanced Camera Capture mode (HDR vs. LowLight), OEM vs. platform implementation, HDR probability, and Low Light probability
Device preferences and settings	<p>Information about the device settings and user preferences such as:</p> <ul style="list-style-type: none"> • User Settings – System, Device, Network & Internet, Personalization, Cortana, Apps, Accounts, Time & Language, Gaming, Ease of Access, Privacy, Update & Security • User-provided device name • Whether device is domain-joined, or cloud-domain joined (i.e. part of a company-managed network) • Hashed representation of the domain name • MDM (mobile device management) enrollment settings and status • BitLocker, Secure Boot, encryption settings, and status • Windows Update settings and status • Developer Unlock settings and status • Default app choices • Default browser choice • Default language settings for app, input, keyboard, speech, and display • App store update settings • Enterprise OrganizationID, Commercial ID

CATEGORY NAME	EXAMPLES
Device peripherals	<p>Information about the device peripherals such as:</p> <ul style="list-style-type: none"> • Peripheral name, device model, class, manufacturer and description • Peripheral device state, install state, and checksum • Driver name, package name, version, and manufacturer • HWID - A hardware vendor defined ID to match a device to a driver INF file • Driver state, problem code, and checksum • Whether driver is kernel mode, signed, and image size
Device network info	<p>Information about the device network configuration such as:</p> <ul style="list-style-type: none"> • Network system capabilities • Local or Internet connectivity status • Proxy, gateway, DHCP, DNS details and addresses • Paid or free network • Wireless driver is emulated or not • Access point mode capable • Access point manufacturer, model, and MAC address • WDI Version • Name of networking driver service • Wi-Fi Direct details • Wi-Fi device hardware ID and manufacturer • Wi-Fi scan attempt counts and item counts • Mac randomization is supported/enabled or not • Number of spatial streams and channel frequencies supported • Manual or Auto Connect enabled • Time and result of each connection attempt • Airplane mode status and attempts • Interface description provided by the manufacturer • Data transfer rates • Cipher algorithm • Mobile Equipment ID (IMEI) and Mobile Country Code (MCCO) • Mobile operator and service provider name • Available SSIDs and BSSIDs • IP Address type -- IPv4 or IPv6 • Signal Quality percentage and changes • Hotspot presence detection and success rate • TCP connection performance • Miracast device names • Hashed IP address

Product and Service Usage data

This type of data includes details about the usage of the device, operating system, applications and services.

CATEGORY NAME	EXAMPLES
---------------	----------

CATEGORY NAME	EXAMPLES
App usage	<p>Information about Windows and application usage such as:</p> <ul style="list-style-type: none"> • OS component and app feature usage • User navigation and interaction with app and Windows features. This could potentially include user input, such as name of a new alarm set, user menu choices, or user favorites. • Time of and count of app/component launches, duration of use, session GUID, and process ID • App time in various states – running foreground or background, sleeping, or receiving active user interaction • User interaction method and duration – whether and length of time user used the keyboard, mouse, pen, touch, speech, or game controller • Cortana launch entry point/reason • Notification delivery requests and status • Apps used to edit images and videos • SMS, MMS, VCard, and broadcast message usage statistics on primary or secondary line • Incoming and Outgoing calls and Voicemail usage statistics on primary or secondary line • Emergency alerts are received or displayed statistics • Content searches within an app • Reading activity -- bookmarking used, print used, layout changed
App or product state	<p>Information about Windows and application state such as:</p> <ul style="list-style-type: none"> • Start Menu and Taskbar pins • Online/Offline status • App launch state -- with deep-link such as Groove launched with an audio track to play, or share contract such as MMS launched to share a picture. • Personalization impressions delivered • Whether the user clicked or hovered on UI controls or hotspots • User feedback Like or Dislike or rating was provided • Caret location or position within documents and media files -- how much of a book has been read in a single session or how much of a song has been listened to.
Login properties	<ul style="list-style-type: none"> • Login success or failure • Login sessions and state

Product and Service Performance data

This type of data includes details about the health of the device, operating system, apps and drivers.

CATEGORY NAME	DESCRIPTION AND EXAMPLES

CATEGORY NAME	DESCRIPTION AND EXAMPLES
Device health and crash data	<p>Information about the device and software health such as:</p> <ul style="list-style-type: none"> • Error codes and error messages, name and ID of the app, and process reporting the error • DLL library predicted to be the source of the error -- xyz.dll • System generated files -- app or product logs and trace files to help diagnose a crash or hang • System settings such as registry keys • User generated files – .doc, .ppt, .csv files where they are indicated as a potential cause for a crash or hang • Details and counts of abnormal shutdowns, hangs, and crashes • Crash failure data – OS, OS component, driver, device, 1st and 3rd party app data • Crash and Hang dumps <ul style="list-style-type: none"> ◦ The recorded state of the working memory at the point of the crash. ◦ Memory in use by the kernel at the point of the crash. ◦ Memory in use by the application at the point of the crash. ◦ All the physical memory used by Windows at the point of the crash. ◦ Class and function name within the module that failed.

CATEGORY NAME	DESCRIPTION AND EXAMPLES
Device performance and reliability data	<p>Information about the device and software performance such as:</p> <ul style="list-style-type: none"> • User Interface interaction durations -- Start Menu display times, browser tab switch times, app launch and switch times, and Cortana and search performance and reliability. • Device on/off performance -- Device boot, shutdown, power on/off, lock/unlock times, and user authentication times (fingerprint and face recognition durations). • In-app responsiveness -- time to set alarm, time to fully render in-app navigation menus, time to sync reading list, time to start GPS navigation, time to attach picture MMS, and time to complete a Microsoft Store transaction. • User input responsiveness – onscreen keyboard invocation times for different languages, time to show auto-complete words, pen or touch latencies, latency for handwriting recognition to words, Narrator screen reader responsiveness, and CPU score. • UI and media performance and glitches/smoothness -- video playback frame rate, audio glitches, animation glitches (stutter when bringing up Start), graphics score, time to first frame, play/pause/stop/seek responsiveness, time to render PDF, dynamic streaming of video from OneDrive performance • Disk footprint -- Free disk space, out of memory conditions, and disk score. • Excessive resource utilization – components impacting performance or battery life through high CPU usage during different screen and power states • Background task performance -- download times, Windows Update scan duration, Windows Defender Antivirus scan times, disk defrag times, mail fetch times, service startup and state transition times, and time to index on-device files for search results • Peripheral and devices -- USB device connection times, time to connect to a wireless display, printing times, network availability and connection times (time to connect to Wi-Fi, time to get an IP address from DHCP etc.), smart card authentication times, automatic brightness environmental response times • Device setup -- first setup experience times (time to install updates, install apps, connect to network etc.), time to recognize connected devices (printer and monitor), and time to setup Microsoft Account. • Power and Battery life – power draw by component (Process/CPU/GPU/Display), hours of screen off time, sleep state transition details, temperature and thermal throttling, battery drain in a power state (screen off or screen on), processes and components requesting power use during screen off, auto-brightness details, time device is plugged into AC vs. battery, battery state transitions • Service responsiveness - Service URI, operation, latency, service success/error codes, and protocol. • Diagnostic heartbeat – regular signal to validate the health of the diagnostics system

CATEGORY NAME	DESCRIPTION AND EXAMPLES
Movies	<p>Information about movie consumption functionality on the device. This isn't intended to capture user viewing, listening or habits.</p> <ul style="list-style-type: none"> • Video Width, height, color pallet, encoding (compression) type, and encryption type • Instructions for how to stream content for the user -- the smooth streaming manifest of chunks of content files that must be pieced together to stream the content based on screen resolution and bandwidth • URL for a specific two second chunk of content if there is an error • Full screen viewing mode details
Music & TV	<p>Information about music and TV consumption on the device. This isn't intended to capture user viewing, listening or habits.</p> <ul style="list-style-type: none"> • Service URL for song being downloaded from the music service – collected when an error occurs to facilitate restoration of service • Content type (video, audio, surround audio) • Local media library collection statistics -- number of purchased tracks, number of playlists • Region mismatch -- User OS Region, and Xbox Live region
Reading	<p>Information about reading consumption functionality on the device. This isn't intended to capture user viewing, listening or habits.</p> <ul style="list-style-type: none"> • App accessing content and status and options used to open a Microsoft Store book • Language of the book • Time spent reading content • Content type and size details
Photos App	<p>Information about photos usage on the device. This isn't intended to capture user viewing, listening or habits.</p> <ul style="list-style-type: none"> • File source data -- local, SD card, network device, and OneDrive • Image & video resolution, video length, file sizes types and encoding • Collection view or full screen viewer use and duration of view

CATEGORY NAME	DESCRIPTION AND EXAMPLES
On-device file query	<p>Information about local search activity on the device such as:</p> <ul style="list-style-type: none"> • Kind of query issued and index type (ConstraintIndex, SystemIndex) • Number of items requested and retrieved • File extension of search result user interacted with • Launched item kind, file extension, index of origin, and the App ID of the opening app. • Name of process calling the indexer and time to service the query. • A hash of the search scope (file, Outlook, OneNote, IE history) • The state of the indices (fully optimized, partially optimized, being built)
Purchasing	<p>Information about purchases made on the device such as:</p> <ul style="list-style-type: none"> • Product ID, edition ID and product URI • Offer details -- price • Order requested date/time • Store client type -- web or native client • Purchase quantity and price • Payment type -- credit card type and PayPal
Entitlements	<p>Information about entitlements on the device such as:</p> <ul style="list-style-type: none"> • Service subscription status and errors • DRM and license rights details -- Groove subscription or OS volume license • Entitlement ID, lease ID, and package ID of the install package • Entitlement revocation • License type (trial, offline vs online) and duration • License usage session

Software Setup and Inventory data

This type of data includes software installation and update information on the device.

CATEGORY NAME	DATA EXAMPLES

CATEGORY NAME	DATA EXAMPLES
Installed Applications and Install History	<p>Information about apps, drivers, update packages, or OS components installed on the device such as:</p> <ul style="list-style-type: none"> • App, driver, update package, or component's Name, ID, or Package Family Name • Product, SKU, availability, catalog, content, and Bundle IDs • OS component, app or driver publisher, language, version and type (Win32 or UWP) • Install date, method, and install directory, count of install attempts • MSI package code and product code • Original OS version at install time • User or administrator or mandatory installation/update • Installation type – clean install, repair, restore, OEM, retail, upgrade, and update
Device update information	<p>Information about Windows Update such as:</p> <ul style="list-style-type: none"> • Update Readiness analysis of device hardware, OS components, apps, and drivers (progress, status, and results) • Number of applicable updates, importance, type • Update download size and source -- CDN or LAN peers • Delay upgrade status and configuration • OS uninstall and rollback status and count • Windows Update server and service URL • Windows Update machine ID • Windows Insider build details

Browsing History data

This type of data includes details about web browsing in the Microsoft browsers.

CATEGORY NAME	DESCRIPTION AND EXAMPLES
Microsoft browser data	<p>Information about Address bar and search box performance on the device such as:</p> <ul style="list-style-type: none"> • Text typed in address bar and search box • Text selected for Ask Cortana search • Service response time • Auto-completed text if there was an auto-complete • Navigation suggestions provided based on local history and favorites • Browser ID • URLs (which may include search terms) • Page title

Inking Typing and Speech Utterance data

This type of data gathers details about the voice, inking, and typing input features on the device.

CATEGORY NAME	DESCRIPTION AND EXAMPLES
Voice, inking, and typing	<p>Information about voice, inking and typing features such as:</p> <ul style="list-style-type: none"> • Type of pen used (highlighter, ball point, pencil), pen color, stroke height and width, and how long it is used • Pen gestures (click, double click, pan, zoom, rotate) • Palm Touch x,y coordinates • Input latency, missed pen signals, number of frames, strokes, first frame commit time, sample rate • Ink strokes written, text before and after the ink insertion point, recognized text entered, Input language - processed to remove identifiers, sequencing information, and other data (such as email addresses and numeric values) which could be used to reconstruct the original content or associate the input to the user. • Text input from Windows Mobile on-screen keyboards except from password fields and private sessions - processed to remove identifiers, sequencing information, and other data (such as email addresses, and numeric values) which could be used to reconstruct the original content or associate the input to the user. • Text of speech recognition results -- result codes and recognized text • Language and model of the recognizer, System Speech language • App ID using speech features • Whether user is known to be a child • Confidence and Success/Failure of speech recognition

Beginning your General Data Protection Regulation (GDPR) journey for Windows 10

9/25/2017 • 26 min to read • [Edit Online](#)

This article provides info about the GDPR, including what it is, and the products Microsoft provides to help you to become compliant.

Introduction

On May 25, 2018, a European privacy law is due to take effect that sets a new global bar for privacy rights, security, and compliance.

The General Data Protection Regulation, or GDPR, is fundamentally about protecting and enabling the privacy rights of individuals. The GDPR establishes strict global privacy requirements governing how you manage and protect personal data while respecting individual choice — no matter where data is sent, processed, or stored.

Microsoft and our customers are now on a journey to achieve the privacy goals of the GDPR. At Microsoft, we believe privacy is a fundamental right, and we believe that the GDPR is an important step forward for clarifying and enabling individual privacy rights. But we also recognize that the GDPR will require significant changes by organizations all over the world.

We have outlined our commitment to the GDPR and how we are supporting our customers within the [Get GDPR compliant with the Microsoft Cloud](#) blog post by our Chief Privacy Officer [Brendon Lynch](#) and the [Earning your trust with contractual commitments to the General Data Protection Regulation](#)” blog post by [Rich Sauer](#) - Microsoft Corporate Vice President & Deputy General Counsel.

Although your journey to GDPR-compliance may seem challenging, we're here to help you. For specific information about the GDPR, our commitments and how to begin your journey, please visit the [GDPR section of the Microsoft Trust Center](#).

GDPR and its implications

The GDPR is a complex regulation that may require significant changes in how you gather, use and manage personal data. Microsoft has a long history of helping our customers comply with complex regulations, and when it comes to preparing for the GDPR, we are your partner on this journey.

The GDPR imposes rules on organizations that offer goods and services to people in the European Union (EU), or that collect and analyze data tied to EU residents, no matter where those businesses are located. Among the key elements of the GDPR are the following:

- **Enhanced personal privacy rights.** Strengthened data protection for residents of EU by ensuring they have the right to access to their personal data, to correct inaccuracies in that data, to erase that data, to object to processing of their personal data, and to move it.
- **Increased duty for protecting personal data.** Reinforced accountability of organizations that process personal data, providing increased clarity of responsibility in ensuring compliance.
- **Mandatory personal data breach reporting.** Organizations that control personal data are required to report personal data breaches that pose a risk to the rights and freedoms of individuals to their supervisory authorities without undue delay, and, where feasible, no later than 72 hours once they become aware of the breach.

As you might anticipate, the GDPR can have a significant impact on your business, potentially requiring you to update privacy policies, implement and strengthen data protection controls and breach notification procedures, deploy highly transparent policies, and further invest in IT and training. Microsoft Windows 10 can help you effectively and efficiently address some of these requirements.

Personal and sensitive data

As part of your effort to comply with the GDPR, you will need to understand how the regulation defines personal and sensitive data and how those definitions relate to data held by your organization.

The GDPR considers personal data to be any information related to an identified or identifiable natural person. That can include both direct identification (such as, your legal name) and indirect identification (such as, specific information that makes it clear it is you the data references). The GDPR also makes clear that the concept of personal data includes online identifiers (such as, IP addresses, mobile device IDs) and location data.

The GDPR introduces specific definitions for genetic data (such as, an individual's gene sequence) and biometric data. Genetic data and biometric data along with other sub categories of personal data (personal data revealing racial or ethnic origin, political opinions, religious or philosophical beliefs, or trade union membership; data concerning health; or data concerning a person's sex life or sexual orientation) are treated as sensitive personal data under the GDPR. Sensitive personal data is afforded enhanced protections and generally requires an individual's explicit consent where these data are to be processed.

Examples of info relating to an identified or identifiable natural person (data subject)

This list provides examples of several types of info that will be regulated through GDPR. This is not an exhaustive list.

- Name
- Identification number (such as, SSN)
- Location data (such as, home address)
- Online identifier (such as, e-mail address, screen names, IP address, device IDs)
- Pseudonymous data (such as, using a key to identify individuals)
- Genetic data (such as, biological samples from an individual)
- Biometric data (such as, fingerprints, facial recognition)

Getting started on the journey towards GDPR compliance

Given how much is involved to become GDPR-compliant, we strongly recommend that you don't wait to prepare until enforcement begins. You should review your privacy and data management practices now. We recommend that you begin your journey to GDPR compliance by focusing on four key steps:

- **Discover.** Identify what personal data you have and where it resides.
- **Manage.** Govern how personal data is used and accessed.
- **Protect.** Establish security controls to prevent, detect, and respond to vulnerabilities and data breaches.
- **Report.** Act on data requests, report data breaches, and keep required documentation.



For each of the steps, we've outlined example tools, resources, and features in various Microsoft solutions, which can be used to help you address the requirements of that step. While this article isn't a comprehensive "how to," we've included links for you to find out more details, and more information is available in the [GDPR section of the Microsoft Trust Center](#).

Windows 10 security and privacy

As you work to comply with the GDPR, understanding the role of your desktop and laptop client machines in creating, accessing, processing, storing and managing data that may qualify as personal and potentially sensitive data under the GDPR is important. Windows 10 provides capabilities that will help you comply with the GDPR requirements to implement appropriate technical and organizational security measures to protect personal data.

With Windows 10, your ability to protect, detect and defend against the types of attacks that can lead to data breaches is greatly improved. Given the stringent requirements around breach notification within the GDPR, ensuring that your desktop and laptop systems are well defended will lower the risks you face that could result in costly breach analysis and notification.

In this section, we'll talk about how Windows 10 provides capabilities that fit squarely in the **Protect** stage of your journey, including these 4 scenarios:

- **Threat protection: Pre-breach threat resistance.** Disrupt the malware and hacking industry by moving the playing field to one where they lose the attack vectors that they depend on.
- **Threat protection: Post-breach detection and response.** Detect, investigate, and respond to advanced threats and data breaches on your networks.
- **Identity protection.** Next generation technology to help protect your user's identities from abuse.
- **Information protection.** Comprehensive data protection while meeting compliance requirements and maintaining user productivity.

These capabilities, discussed in more detail below with references to specific GDPR requirements, are built on top of advanced device protection that maintains the integrity and security of the operating system and data.

A key provision within the GDPR is data protection by design and by default, and helping with your ability to meet this provision are features within Windows 10 such as the Trusted Platform Module (TPM) technology designed to provide hardware-based, security-related functions. A TPM chip is a secure crypto-processor that is designed to carry out cryptographic operations.

The chip includes multiple physical security mechanisms to make it tamper resistant, and malicious software is unable to tamper with the security functions of the TPM. Some of the key advantages of using TPM technology are that you can:

- Generate, store, and limit the use of cryptographic keys.
- Use TPM technology for platform device authentication by using the TPM's unique RSA key, which is burned

into itself.

- Help to ensure platform integrity by taking and storing security measurements.

Additional advanced device protection relevant to your operating without data breaches include Windows Trusted Boot to help maintain the integrity of the system by ensuring malware is unable to start before system defenses.

Threat protection: Pre-breach threat resistance

The GDPR requires you to implement appropriate technical and organizational security measures to protect personal data.

Your ability to meet this requirement to implement appropriate technical security measures should reflect the threats you face in today's increasingly hostile IT environment. Today's security threat landscape is one of aggressive and tenacious threats. In previous years, malicious attackers mostly focused on gaining community recognition through their attacks or the thrill of temporarily taking a system offline. Since then, attacker's motives have shifted toward making money, including holding devices and data hostage until the owner pays the demanded ransom.

Modern attacks increasingly focus on large-scale intellectual property theft; targeted system degradation that can result in financial loss; and now even cyberterrorism that threatens the security of individuals, businesses, and national interests all over the world. These attackers are typically highly trained individuals and security experts, some of whom are in the employ of nation states that have large budgets and seemingly unlimited human resources. Threats like these require an approach that can meet this challenge.

Not only are these threats a risk to your ability to maintain control of any personal or sensitive data you may have, but they are a material risk to your overall business as well. Consider recent data from Ponemon Institute, Verizon, and Microsoft:

- The average cost of the type of data breach the GDPR will expect you to report is \$3.5M. (Ponemon Institute).
- 63% of these breaches involve weak or stolen passwords that the GDPR expects you to address. (2016 Data Breach Investigations Report, Verizon Enterprise).
- Over 300,000 new malware samples are created and spread every day making your task to address data protection even more challenging. (Microsoft Malware Protection Center, Microsoft).

As seen with recent ransomware attacks, once called the "black plague" of the Internet, attackers are going after bigger targets that can afford to pay more, with potentially catastrophic consequences. Desktops and laptops, that contain personal and sensitive data, are commonly targeted where control over data might be lost.

In response to these threats and as a part of your mechanisms to resist these types of breaches so that you remain in compliance with the GDPR, Windows 10 provides built in technology, detailed below including the following:

- Windows Defender Antivirus to respond to emerging threats on data.
- Microsoft Edge to systematically disrupt phishing, malware, and hacking attacks.
- Windows Defender Device Guard to block all unwanted applications on client machines.

Responding to emerging data threats

Windows Defender Antivirus is a built-in antimalware solution that provides security and antimalware management for desktops, portable computers, and servers. In Windows 10, it uses a multi-pronged approach to improve antimalware:

- **Cloud-delivered protection.** Helps to detect and block new malware within seconds, even if the malware has never been seen before.
- **Rich local context.** Improves how malware is identified. Windows 10 informs Windows Defender Antivirus

not only about content like files and processes, but also where the content came from, where it's been stored, and more.

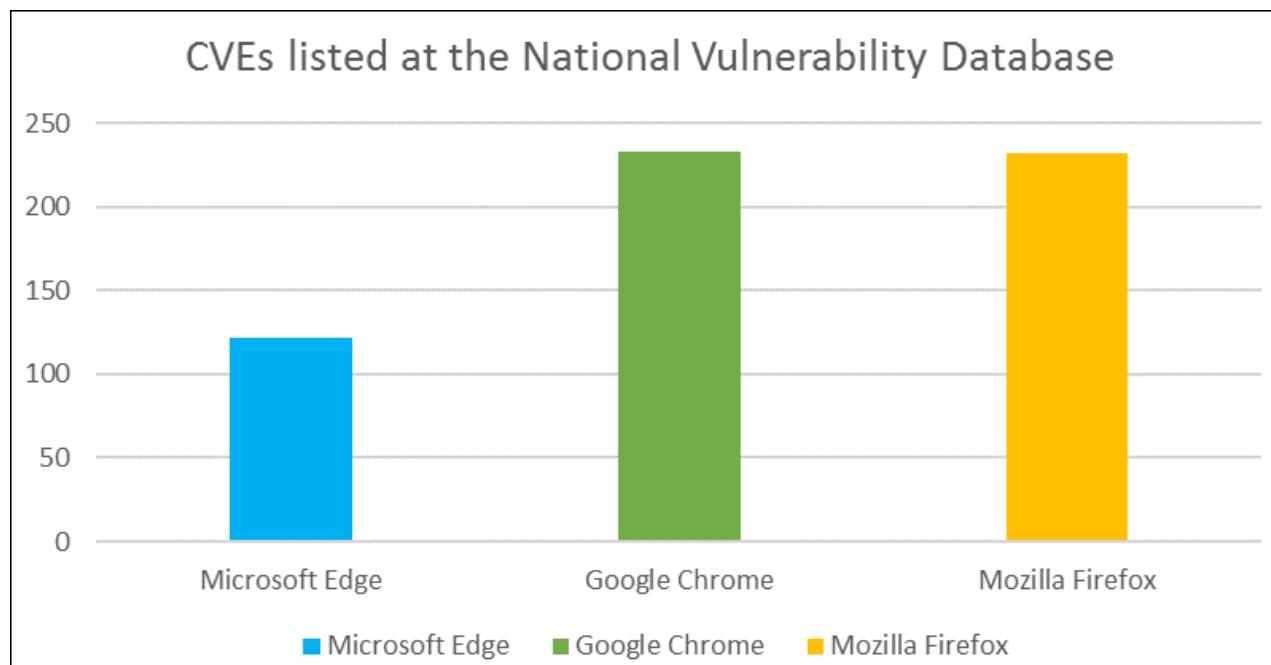
- **Extensive global sensors.** Help to keep Windows Defender Antivirus current and aware of even the newest malware. This is accomplished in two ways: by collecting the rich local context data from end points and by centrally analyzing that data.
- **Tamper proofing.** Helps to guard Windows Defender Antivirus itself against malware attacks. For example, Windows Defender Antivirus uses Protected Processes, which prevents untrusted processes from attempting to tamper with Windows Defender Antivirus components, its registry keys, and so on.
- **Enterprise-level features.** Give IT pros the tools and configuration options necessary to make Windows Defender Antivirus an enterprise-class antimalware solution.

Systemically disrupting phishing, malware, and hacking attacks

In today's threat landscape, your ability to provide those mechanisms should be tied to the specific data-focused attacks you face through phishing, malware and hacking due to the browser-related attacks.

As part of Windows 10, Microsoft has brought you Microsoft Edge, our safest and most secure browser to-date. Over the past two years, we have been continuously innovating, and we're proud of the progress we've made. This quality of engineering is reflected by the reduction of Common Vulnerabilities and Exposures (CVE) when comparing Microsoft Edge with Internet Explorer over the past year. Browser-related attacks on personal and sensitive data that you will need to protect under the GDPR means this innovation in Windows 10 is important.

While no modern browser — or any complex application — is free of vulnerabilities, many of the vulnerabilities for Microsoft Edge have been responsibly reported by professional security researchers who work with the Microsoft Security Response Center (MSRC) and the Microsoft Edge team to ensure customers are protected well before any attacker might use these vulnerabilities in the wild. Even better, there is no evidence that any vulnerabilities have been exploited in the wild as zero-day attacks.



However, many businesses worldwide have come under increasing threat of targeted attacks, where attackers are crafting specialized attacks against a specific business, attempting to take control of corporate networks and data.

Blocking all unwanted apps

Application Control is your best defense in a world where there are more than 300,000 new malware samples each day. As part of Windows 10, Windows Defender Device Guard is a combination of enterprise-related hardware and software security features that, when configured together, will lock a device down so that it can only run trusted applications that you define in your code integrity policies. If the app isn't trusted it can't run, period.

With hardware that meets basic requirements, it also means that even if an attacker manages to get control of the Windows kernel, he or she will be much less likely to be able to run malicious executable code. With appropriate hardware, Windows Defender Device Guard can use the new virtualization-based security in Windows 10 to isolate the Code Integrity service from the Microsoft Windows kernel itself. In this case, the Code Integrity service runs alongside the kernel in a Windows hypervisor-protected container.

Windows Defender Device Guard protects threats that can expose personal or sensitive data to attack, including:

- Exposure to new malware, for which the "signature" is not yet known
- Exposure to unsigned code (most malware is unsigned)
- Malware that gains access to the kernel and then, from within the kernel, captures sensitive information or damages the system
- DMA-based attacks, for example, attacks launched from a malicious device that read secrets from memory, making the enterprise more vulnerable to attack; and
- Exposure to boot kits or to a physically present attacker at boot time.

Threat protection: Post-breach detection and response

The GDPR includes explicit requirements for breach notification where a personal data breach means, "a breach of security leading to the accidental or unlawful destruction, loss, alteration, unauthorized disclosure of, or access to, personal data transmitted, stored or otherwise processed."

As noted in the Windows Security Center white paper, [Post Breach: Dealing with Advanced Threats](#), "*Unlike pre-breach, post-breach assumes a breach has already occurred – acting as a flight recorder and Crime Scene Investigator (CSI). Post-breach provides security teams the information and toolset needed to identify, investigate, and respond to attacks that otherwise will stay undetected and below the radar.*"

Insightful security telemetry

For nearly two decades, Microsoft has been turning threats into useful intelligence that can help fortify our platform and protect customers. Today, with the immense computing advantages afforded by the cloud, we are finding new ways to use our rich analytics engines driven by threat intelligence to protect our customers.

By applying a combination of automated and manual processes, machine learning and human experts, we can create an Intelligent Security Graph that learns from itself and evolves in real-time, reducing our collective time to detect and respond to new incidents across our products.



The scope of Microsoft's threat intelligence spans, literally, billions of data points: 35 billion messages scanned monthly, 1 billion customers across enterprise and consumer segments accessing 200+ cloud services, and 14 billion authentications performed daily. All this data is pulled together on your behalf by Microsoft to create the Intelligent Security Graph that can help you protect your front door dynamically to stay secure, remain productive, and meet the requirements of the GDPR.

Detecting attacks and forensic investigation

Even the best endpoint defenses may be breached eventually, as cyberattacks become more sophisticated and targeted.

Windows Defender Advanced Threat Protection (ATP) helps you detect, investigate, and respond to advanced attacks and data breaches on your networks. GDPR expects you to protect against attacks and breaches through technical security measures to ensure the ongoing confidentiality, integrity, and availability of personal data.

Among the key benefits of ATP are the following:

- Detecting the undetectable - sensors built deep into the operating system kernel, Windows security experts, and unique optics from over 1 billion machines and signals across all Microsoft services.
- Built in, not bolted on - agentless with high performance and low impact, cloud-powered; easy management with no deployment.
- Single pane of glass for Windows security - explore 6 months of rich machine timeline that unifies security events from Windows Defender ATP, Windows Defender Antivirus.
- Power of the Microsoft graph - leverages the Microsoft Intelligence Security Graph to integrate detection and exploration with Office 365 ATP subscription, to track back and respond to attacks.

Read more at [What's new in the Windows Defender ATP Creators Update preview](#).

To provide Detection capabilities, Windows 10 improves our OS memory and kernel sensors to enable detection of attackers who are employing in-memory and kernel-level attacks – shining a light into previously dark spaces where attackers hid from conventional detection tools. We've already successfully leveraged this new technology against zero-days attacks on Windows.

The screenshot shows the Windows Defender Security Center Alert interface. The main window displays an alert titled "Process privilege escalation due to kernel exploit" from "cont-jonathan" on "02-06-2017 | 18:08:53". The alert is categorized as "High" severity under "Privilege Escalation". The "Execution details" pane shows the affected process is "WINWORD.EXE" (User: A CONTOSO\jonathan, icon: WINWORD). The "File details" pane shows the file path as "C:\Program Files\Microsoft Office\Office16\WINWORD.DLL" and the file size as 1.8 MB. The "Detections" pane indicates "No detections found". The "Alerts" section shows 1 alert. The "Observed worldwide" section shows a count of 86.9k, first seen 2 years ago, and last seen 15 hours ago. The "Alert Process Tree" diagram shows a complex hierarchy of processes including ntkrnlpe.exe, smss.exe, winlogon.exe, userinit.exe, explorer.exe, and WINWORD.EXE, with red dots indicating malicious activity. A note in the tree states: "Access token modified. The WINWORD.EXE access token was modified to the SYSTEM token". The bottom of the interface notes: "Incident graph is not available for this alert".

We continue to upgrade our detections of ransomware and other advanced attacks, applying our behavioral and machine-learning detection library to counter changing attack trends. Our historical detection capability ensures

new detection rules apply to up to six months of stored data to detect attacks that previously went unnoticed. Customers can also add customized detection rules or IOCs to augment the detection dictionary.

Customers asked us for a single pane of glass across the entire Windows security stack. Windows Defender Antivirus detections and Windows Defender Device Guard blocks are the first to surface in the Windows Defender ATP portal interleaved with Windows Defender ATP detections. The new user entity adds identity as a pivot, providing insight into actions, relationships, and alerts that span machines and allow us to track attackers moving laterally across the network.

Our alert page now includes a new process tree visualization that aggregates multiple detections and related events into a single view that helps security teams reduce the time to resolve cases by providing the information required to understand and resolve incidents without leaving the alert page.

Security Operations (SecOps) can hunt for evidence of attacks, such as file names or hashes, IP addresses or URLs, behaviors, machines, or users. They can do this immediately by searching the organization's cloud inventory, across all machines – and going back up to 6 months in time – even if machines are offline, have been reimaged, or no longer exist.

The screenshot shows the Windows Defender Security Center interface. At the top, it displays 'Windows Defender Security Center' and 'User'. Below this, the user profile for 'cont-jonathanw' is shown, including their name ('jonathan wolcott'), job title ('Sales Manager'), department ('Sales'), and last seen activity ('a month ago'). A link to 'More details on this user in ATA' and a 'Contact via SfB' button are also present. To the right, a section titled 'Logged on machines' shows '2' machines, with one entry for 'cont-jonathanw (Local admin)'. Below this, a table lists 'Alerts related to this user' with columns for 'Last activity', 'Title', 'Machine', 'Severity', 'Status', and 'Assigned to'. The table contains 10 entries from February 2017, mostly categorized under 'Suspicious Activity' or 'Reconnaissance'. At the bottom, there's a timeline from Oct 2016 to Mar 2017, a 'Logon types: All' dropdown, and a table showing 'Observed in organization from' with data for 'Machine', 'Total observed users', 'Most frequent user', and 'Least frequent user'.

When detecting an attack, security teams can now take immediate action: isolate machines, ban files from the network, kill or quarantine running processes or files, or retrieve an investigation package from a machine to provide forensic evidence – with a click of a button. Because while detecting advanced attacks is important – shutting them down is even more so.

The screenshot shows the Windows Defender Security Center interface. On the left, there's a sidebar with icons for Home, Machine, Network, and Identity. The main area shows a machine named 'cont-jonathanw'. Under 'Actions', there are options to 'Collect investigation package' (which is available), 'Isolate machine', and 'Action center'. Below this is a section for 'Alerts related to this machine' with a table of recent events:

Last activity	Title	User
02.06.2017 16:40:31	A malicious PowerShell Cmdlet was invoked on the machine. Suspicious Activity	contoso\jonathan.wolcott
02.06.2017 16:39:12	A suspicious remote shell was detected. Command And Control	contoso\jonathan.wolcott
02.06.2017 16:38:12	A process was injected with potentially malicious code Installation	contoso\jonathan.wolcott
02.06.2017 16:37:47	Process privilege escalation due to kernel exploit Privilege Escalation	contoso\jonathan.wolcott
02.06.2017 16:37:11	Abnormal code execution was observed Exploit	contoso\jonathan.wolcott
02.06.2017 15:49:45	A known vulnerable driver was loaded Privilege Escalation	nt authority\system

On the right, there's an 'Action Center' pane with sections for 'Investigation package collection' and 'Machine isolation', both of which show successful submissions. A note at the bottom says: 'For submitted actions to take effect, machine must be connected to the network.' There's also a 'Close' button.

Identity Protection

Identify and access management is another area where the GDPR has placed special emphasis by calling for mechanisms to grant and restrict access to data subject personal data (for example, role-based access, segregation of duties).

Multi-factor protection

Biometric authentication – using your face, iris, or fingerprint to unlock your devices – is much safer than traditional passwords. You – uniquely you – plus your device are the keys to your apps, data, and even websites and services – not a random assortment of letters and numbers that are easily forgotten, hacked, or written down and pinned to a bulletin board.

Your ability to protect personal and sensitive data, that may be stored or accessed through desktop or laptops will be further enhanced by adopting advanced authentication capabilities such as Windows Hello for Business and Windows Hello companion devices. Windows Hello for Business, part of Windows 10, gives users a personal, secured experience where the device is authenticated based on their presence. Users can log in with a look or a touch, with no need for a password.

In conjunction with Windows Hello for Business, biometric authentication uses fingerprints or facial recognition and is more secure, more personal, and more convenient. If an application supports Hello, Windows 10 enables you to authenticate applications, enterprise content, and even certain online experiences without a password being stored on your device or in a network server at all. Windows Hello for Business works with the Companion Device Framework to enhance the user authentication experience. Using the Windows Hello Companion Device Framework, a companion device can provide a rich experience for Windows Hello even when biometrics are not available (for example, if the Windows 10 desktop lacks a camera for face authentication or fingerprint reader device).

There are numerous ways one can use the Windows Hello Companion Device Framework to build a great Windows unlock experience with a companion device. For example, users can:

- Work offline (for example, while traveling on a plane)
- Attach their companion device to PC via USB, touch the button on the companion device, and automatically unlock their PC.
- Carry a phone in their pocket that is already paired with their PC over Bluetooth. Upon hitting the spacebar

on their PC, their phone receives a notification. Approve it and the PC simply unlocks.

- Tap their companion device to an NFC reader to quickly unlock their PC.
- Wear a fitness band that has already authenticated the wearer. Upon approaching PC, and by performing a special gesture (like clapping), the PC unlocks.

Protection against attacks by isolating user credentials

As noted in the [Windows 10 Credential Theft Mitigation Guide](#), "the tools and techniques criminals use to carry out credential theft and reuse attacks improve, malicious attackers are finding it easier to achieve their goals.

Credential theft often relies on operational practices or user credential exposure, so effective mitigations require a holistic approach that addresses people, processes, and technology. In addition, these attacks rely on the attacker stealing credentials after compromising a system to expand or persist access, so organizations must contain breaches rapidly by implementing strategies that prevent attackers from moving freely and undetected in a compromised network."

An important design consideration for Windows 10 was mitigating credential theft — in particular, derived credentials. Windows Defender Credential Guard provides significantly improved security against derived credential theft and reuse by implementing a significant architectural change in Windows designed to help eliminate hardware-based isolation attacks rather than simply trying to defend against them.

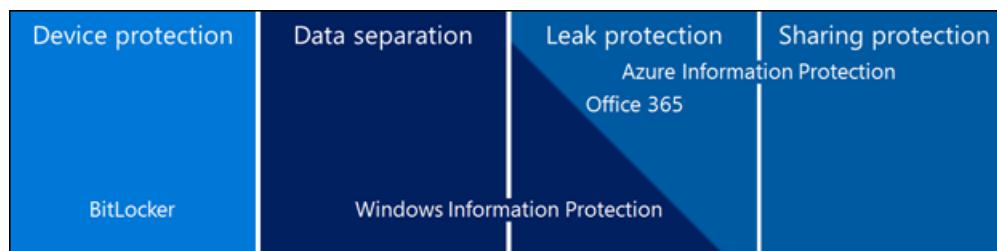
When Credential Manager domain credentials, NTLM, and Kerberos derived credentials are protected using virtualization-based security, the credential theft attack techniques and tools used in many targeted attacks are blocked. Malware running in the operating system with administrative privileges can't extract secrets that are protected by virtualization-based security. While Windows Defender Credential Guard is a powerful mitigation, persistent threat attacks will likely shift to new attack techniques and you should also incorporate Windows Defender Device Guard, as described above, and other security strategies and architectures.

Information Protection

The GDPR is focused on information protection regarding data that is considered as personal or sensitive in relation to a natural person, or data subject. Device protection, protection against threats, and identity protection are all important elements of a Defense in Depth strategy surrounding a layer of information protection in your laptop and desktop systems.

As to the protection of data, the GDPR recognizes that in assessing data security risk, consideration should be given to the risks that are presented such as accidental loss, unauthorized disclosure of, or access to, personal data transmitted, stored or otherwise processed. It also recommends that measures taken to maintain an appropriate level of security should consider the state-of-the-art and the costs of implementation in relation to the risks among other factors.

Windows 10 provides built in risk mitigation capabilities for today's threat landscape. In this section, we will look at the types of technologies that will help your journey toward GDPR compliance and at the same time provide you with solid overall data protection as part of a comprehensive information protection strategy.



Encryption for lost or stolen devices

The GDPR calls for mechanisms that implement appropriate technical security measures to confirm the ongoing confidentiality, integrity, and availability of both personal data and processing systems. BitLocker Encryption, first introduced as part of Microsoft's Next-Generation Secure Computing Base architecture in 2004 and made available with Windows Vista, is a built-in data protection feature that integrates with the operating system and addresses

the threats of data theft or exposure from lost, stolen, or inappropriately decommissioned computers.

BitLocker provides the most protection when used with a Trusted Platform Module (TPM) version 1.2 or later. The TPM is a hardware component installed in many newer computers by the computer manufacturers. It works with BitLocker to protect user data and to ensure that a computer has not been tampered with while the system was offline.

Data on a lost or stolen computer is vulnerable to unauthorized access, either by running a software-attack tool against it or by transferring the computer's hard disk to a different computer. BitLocker helps mitigate unauthorized data access by enhancing file and system protections. BitLocker also helps render data inaccessible when BitLocker-protected computers are decommissioned or recycled.

Related to BitLocker are Encrypted Hard Drives, a new class of hard drives that are self-encrypting at a hardware level and allow for full disk hardware encryption. Encrypted Hard Drives use the rapid encryption that is provided by BitLocker Drive Encryption to enhance data security and management.

By offloading the cryptographic operations to hardware, Encrypted Hard Drives increase BitLocker performance and reduce CPU usage and power consumption. Because Encrypted Hard Drives encrypt data quickly, enterprise devices can expand BitLocker deployment with minimal impact on productivity.

Some of the benefits of Encrypted Hard Drives include:

- **Better performance.** Encryption hardware, integrated into the drive controller, allows the drive to operate at full data rate with no performance degradation.
- **Strong security based in hardware.** Encryption is always "on" and the keys for encryption never leave the hard drive. User authentication is performed by the drive before it will unlock, independently of the operating system
- **Ease of use.** Encryption is transparent to the user because it is on by default. There is no user interaction needed to enable encryption. Encrypted Hard Drives are easily erased using on-board encryption key; there is no need to re-encrypt data on the drive.
- **Lower cost of ownership.** There is no need for new infrastructure to manage encryption keys, since BitLocker leverages your Active Directory Domain Services infrastructure to store recovery information. Your device operates more efficiently because processor cycles don't need to be used for the encryption process.

Preventing accidental data leaks to unauthorized users

Part of the reality of your operating in a mobile-first, cloud-first world is the notion that some laptops will have multiple purposes – both business and personal. Yet that data that is considered as personal and sensitive regarding EU residents considered as "data subjects" must be protected in line with the requirements of the GDPR.

Windows Information Protection helps people separate their work and personal data and keeps data encrypted wherever it's stored. Your employees can safely use both work and personal data on the same device without switching applications. Windows Information Protection helps end users avoid inadvertent data leaks by sending a warning when copy/pasting information in non-corporate applications – end users can still proceed but the action will be logged centrally.

For example, employees can't send protected work files from a personal email account instead of their work account. They also can't accidentally post personal or sensitive data from a corporate site into a tweet. Windows Information Protection also helps ensure that they aren't saving personal or sensitive data in a public cloud storage location.

Capabilities to classify, assign permissions and share data

Windows Information Protection is designed to coexist with advanced data loss prevention (DLP) capabilities found in Office 365 ProPlus, Azure Information Protection, and Azure Rights Management. Advanced DLP prevents printing, for example, or protects work data that is emailed outside your company.

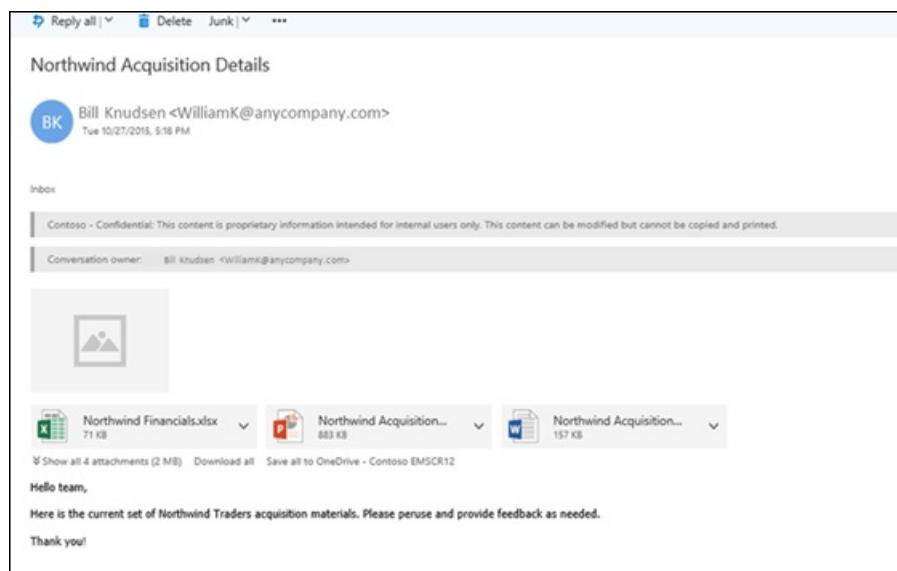
To continuously protect your data, regardless of where it is stored, with whom it is shared, or if the device is running iOS, Android or Windows, the classification and protection needs to be built into the file itself, so this protection can travel with the data wherever it goes. Microsoft Azure Information Protection (AIP) is designed to provide this persistent data protection both on-premises and in the cloud.

Data classification is an important part of any data governance plan. Adopting a classification scheme that applies throughout your business can be particularly helpful in responding to what the GDPR calls data subject (for example, your EU employee or customer) requests, because it enables enterprises to identify more readily and process personal data requests.

Azure Information Protection can be used to help you classify and label your data at the time of creation or modification. Protection in the form of encryption, which the GDPR recognizes may be appropriate at times, or visual markings can then be applied to data needing protection.

With Azure Information Protection, you can either query for data marked with a sensitivity label or intelligently identify sensitive data when a file or email is created or modified. Once identified, you can automatically classify and label the data – all based on the company's desired policy.

Azure Information Protection also helps your users share sensitive data in a secure manner. In the example below, information about a sensitive acquisition was encrypted and restricted to a group of people who were granted only a limited set of permissions on the information – they could modify the content but could not copy or print it.



Related content for associated Windows 10 solutions

- **Windows Hello for Business:** <https://www.youtube.com/watch?v=WOvoXQdj-9E> and <https://docs.microsoft.com/en-us/windows/access-protection/hello-for-business/hello-identity-verification>
- **Windows Defender Antivirus:** <https://www.youtube.com/watch?v=P1aNEy09Nal> and <https://docs.microsoft.com/en-us/windows/threat-protection/windows-defender-antivirus/windows-defender-antivirus-in-windows-10>
- **Windows Defender Advanced Threat Protection:** <https://www.youtube.com/watch?v=qxeGa3pxlwg> and <https://docs.microsoft.com/en-us/windows/threat-protection/windows-defender-atp/windows-defender-advanced-threat-protection>
- **Windows Defender Device Guard:** <https://www.youtube.com/watch?v=F-pTkesjkhI> and <https://docs.microsoft.com/en-us/windows/device-security/device-guard/device-guard-deployment-guide>
- **Windows Defender Credential Guard:** <https://www.youtube.com/watch?v=F-pTkesjkhI> and <https://docs.microsoft.com/en-us/windows/access-protection/credential-guard/credential-guard>

- **Windows Information Protection:** <https://www.youtube.com/watch?v=wLkQOmK7-Jg> and <https://docs.microsoft.com/en-us/windows/threat-protection/windows-information-protection/protect-enterprise-data-using-wip>
- Windows 10 Security Guide: <https://technet.microsoft.com/en-us/itpro/windows/keep-secure/windows-10-security-guide>

Disclaimer

This article is a commentary on the GDPR, as Microsoft interprets it, as of the date of publication. We've spent a lot of time with GDPR and like to think we've been thoughtful about its intent and meaning. But the application of GDPR is highly fact-specific, and not all aspects and interpretations of GDPR are well-settled.

As a result, this article is provided for informational purposes only and should not be relied upon as legal advice or to determine how GDPR might apply to you and your organization. We encourage you to work with a legally-qualified professional to discuss GDPR, how it applies specifically to your organization, and how best to ensure compliance.

MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED, OR STATUTORY, AS TO THE INFORMATION IN THIS ARTICLE. This article is provided "as-is." Information and views expressed in this article, including URL and other Internet website references, may change without notice.

This article does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this article for your internal, reference purposes only.

Published September 2017

Version 1.0

© 2017 Microsoft. All rights reserved.

Manage connections from Windows operating system components to Microsoft services

10/26/2017 • 55 min to read • [Edit Online](#)

Applies to

- Windows 10
- Windows Server 2016

If you're looking for content on what each telemetry level means and how to configure it in your organization, see [Configure Windows telemetry in your organization](#).

Learn about the network connections that Windows components make to Microsoft and also the privacy settings that affect data that is shared with either Microsoft or apps and how they can be managed by an IT Pro.

If you want to minimize connections from Windows to Microsoft services, or configure particular privacy settings, this article covers the settings that you could consider. You can configure telemetry at the lowest level for your edition of Windows, and also evaluate which other connections Windows makes to Microsoft services you want to turn off in your environment from the list in this article.

You can configure telemetry at the Security level, turn off Windows Defender telemetry and MSRT reporting, and turn off all other connections to Microsoft network endpoints as described in this article to help prevent Windows from sending any data to Microsoft. There are many reasons why these communications are enabled by default, such as updating malware definitions and maintain current certificate revocation lists, which is why we strongly recommend against this. This data helps us deliver a secure, reliable, and more delightful personalized experience.

To help make it easier to deploy settings to restrict connections from Windows 10 to Microsoft, you can apply the [Windows Restricted Traffic Limited Functionality Baseline](#). This baseline was created in the same way as the [Windows security baselines](#) that are often used to efficiently configure Windows to a known secure state. Running the Windows Restricted Traffic Limited Functionality Baseline on devices in your organization will allow you to quickly configure all of the settings covered in this document. However, some of the settings reduce the functionality and security configuration of your device and are therefore not recommended. Make sure you've chosen the right settings configuration for your environment before applying. You should not extract this package to the windows\system32 folder because it will not apply correctly. Applying this baseline is equivalent to applying the Windows 10 steps covered in this article.

We are always striving to improve our documentation and welcome your feedback. You can provide feedback by contacting telmhelp@microsoft.com.

Not finding content you need? Windows 10 users, tell us what you want on [Feedback Hub](#).

What's new in Windows 10, version 1709

Here's a list of changes that were made to this article for Windows 10, version 1709:

- Added the Phone calls section.
- Added the Storage Health section.

What's new in Windows 10, version 1703

Here's a list of changes that were made to this article for Windows 10, version 1703:

- Added an MDM policy for Font streaming.
- Added an MDM policy for Network Connection Status Indicator.
- Added an MDM policy for the Microsoft Account Sign-In Assistant.
- Added instructions for removing the Sticky Notes app.
- Added registry paths for some Group Policies

- Added the Find My Device section
- Added the Tasks section
- Added the App Diagnostics section
- Added the following Group Policies:
 - Prevent managing SmartScreen Filter
 - Turn off Compatibility View
 - Turn off Automatic Download and Install of updates
 - Do not connect to any Windows Update locations
 - Turn off access to all Windows Update features
 - Specify Intranet Microsoft update service location
 - Enable Windows NTP client
 - Turn off Automatic download of the ActiveX VersionList
 - Allow Automatic Update of Speech Data
 - Accounts: Block Microsoft Accounts
 - Do not use diagnostic data for tailored experiences

Settings

The following sections list the components that make network connections to Microsoft services by default. You can configure these settings to control the data that is sent to Microsoft. To prevent Windows from sending any data to Microsoft, configure telemetry at the Security level, turn off Windows Defender telemetry and MSRT reporting, and turn off all of these connections.

If you're running Windows 10, they will be included in the next update for the Long Term Servicing Branch.

Settings for Windows 10 Enterprise, version 1703

See the following table for a summary of the management settings for Windows 10 Enterprise, version 1703.

SETTING	UI	GROUP POLICY	MDM POLICY	REGISTRY	COMMAND LINE
1. Automatic Root Certificates Update		✓			
2. Cortana and Search	✓	✓	✓	✓	✓
3. Date & Time	✓	✓		✓	
4. Device metadata retrieval		✓		✓	
5. Find My Device		✓			
6. Font streaming		✓		✓	
7. Insider Preview builds	✓	✓	✓	✓	✓
8. Internet Explorer	✓	✓		✓	
9. Live Tiles		✓		✓	

SETTING	UI	GROUP POLICY	MDM POLICY	REGISTRY	COMMAND LINE
10. Mail synchronization	✓		✓	✓	
11. Microsoft Account		✓	✓	✓	
12. Microsoft Edge	✓	✓	✓	✓	✓
13. Network Connection Status Indicator		✓		✓	
14. Offline maps	✓	✓		✓	
15. OneDrive		✓		✓	
16. Preinstalled apps	✓				✓
17. Settings > Privacy					
17.1 General	✓	✓	✓	✓	
17.2 Location	✓	✓	✓	✓	
17.3 Camera	✓	✓	✓	✓	
17.4 Microphone	✓	✓	✓	✓	
17.5 Notifications	✓	✓	✓	✓	
17.6 Speech, inking, & typing	✓	✓	✓	✓	
17.7 Account info	✓	✓	✓	✓	
17.8 Contacts	✓	✓	✓	✓	
17.9 Calendar	✓	✓	✓	✓	
17.10 Call history	✓	✓	✓	✓	
17.11 Email	✓	✓	✓	✓	
17.12 Messaging	✓	✓	✓	✓	

SETTING	UI	GROUP POLICY	MDM POLICY	REGISTRY	COMMAND LINE
17.13 Phone calls	✓	✓	✓	✓	
17.14 Radios	✓	✓	✓	✓	
17.15 Other devices	✓	✓	✓	✓	
17.16 Feedback & diagnostics	✓	✓	✓	✓	
17.17 Background apps	✓	✓	✓		
17.18 Motion	✓	✓	✓	✓	
17.19 Tasks	✓	✓	✓	✓	
17.20 App Diagnostics	✓	✓	✓	✓	
18. Software Protection Platform		✓	✓	✓	
19. Storage Health		✓			
20. Sync your settings	✓	✓	✓	✓	
21. Teredo		✓		✓	✓
22. Wi-Fi Sense	✓	✓		✓	
23. Windows Defender		✓	✓	✓	
24. Windows Media Player	✓				✓
25. Windows Spotlight	✓	✓	✓	✓	
26. Microsoft Store		✓		✓	
27. Windows Update Delivery Optimization	✓	✓	✓	✓	
28. Windows Update	✓	✓	✓		

Settings for Windows Server 2016 with Desktop Experience

See the following table for a summary of the management settings for Windows Server 2016 with Desktop Experience.

SETTING	UI	GROUP POLICY	REGISTRY	COMMAND LINE
1. Automatic Root Certificates Update		✓	✓	
2. Cortana and Search	✓	✓	✓	
3. Date & Time	✓	✓	✓	
4. Device metadata retrieval		✓	✓	
6. Font streaming		✓	✓	
7. Insider Preview builds	✓	✓	✓	
8. Internet Explorer	✓	✓	✓	
9. Live Tiles		✓	✓	
11. Microsoft Account		✓	✓	
13. Network Connection Status Indicator		✓	✓	
15. OneDrive		✓		
17. Settings > Privacy				
17.1 General	✓	✓	✓	
18. Software Protection Platform		✓	✓	
21. Teredo		✓	✓	✓
23. Windows Defender		✓	✓	
24. Windows Media Player				✓
26. Microsoft Store		✓	✓	
28. Windows Update		✓	✓	

Settings for Windows Server 2016 Server Core

See the following table for a summary of the management settings for Windows Server 2016 Server Core.

SETTING	GROUP POLICY	REGISTRY	COMMAND LINE
1. Automatic Root Certificates Update	✓	✓	
3. Date & Time	✓	✓	
6. Font streaming	✓	✓	
13. Network Connection Status Indicator	✓		
18. Software Protection Platform	✓		
21. Teredo	✓		✓
23. Windows Defender	✓	✓	
28. Windows Update	✓	✓	

Settings for Windows Server 2016 Nano Server

See the following table for a summary of the management settings for Windows Server 2016 Nano Server.

SETTING	REGISTRY	COMMAND LINE
1. Automatic Root Certificates Update	✓	
3. Date & Time	✓	
21. Teredo		✓
28. Windows Update	✓	

Settings

Use the following sections for more information about how to configure each setting.

1. Automatic Root Certificates Update

The Automatic Root Certificates Update component is designed to automatically check the list of trusted authorities on Windows Update to see if an update is available. For more information, see [Automatic Root Certificates Update Configuration](#). Although not recommended, you can turn off Automatic Root Certificates Update, which also prevents updates to the disallowed certificate list and the pin rules list.

Caution

By not automatically downloading the root certificates, the device might have not be able to connect to some websites.

For Windows 10, Windows Server 2016 with Desktop Experience, and Windows Server 2016 Server Core:

- Enable the Group Policy: **Computer Configuration > Administrative Templates > System > Internet Communication Management > Internet Communication Settings > Turn off Automatic Root Certificates Update**

-and-

1. Navigate to **Computer Configuration > Windows Settings > Security Settings > Public Key Policies**.
2. Double-click **Certificate Path Validation Settings**.
3. On the **Network Retrieval** tab, select the **Define these policy settings** check box.
4. Clear the **Automatically update certificates in the Microsoft Root Certificate Program (recommended)** check box, and then click **OK**.

-or-

- Create the registry path **HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\SystemCertificates\AuthRoot** and then add a REG_DWORD registry setting, called **DisableRootAutoUpdate**, with a value of 1.

-and-

1. Navigate to **Computer Configuration > Windows Settings > Security Settings > Public Key Policies**.
2. Double-click **Certificate Path Validation Settings**.
3. On the **Network Retrieval** tab, select the **Define these policy settings** check box.
4. Clear the **Automatically update certificates in the Microsoft Root Certificate Program (recommended)** check box, and then click **OK**.

On Windows Server 2016 Nano Server:

- Create the registry path **HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\SystemCertificates\AuthRoot** and then add a REG_DWORD registry setting, called **DisableRootAutoUpdate**, with a value of 1.

NOTE

CRL and OCSP network traffic is currently whitelisted and will still show up in network traces. CRL and OCSP checks are made to the issuing certificate authorities. Microsoft is one of them, but there are many others, such as DigiCert, Thawte, Google, Symantec, and VeriSign.

2. Cortana and Search

Use either Group Policy or MDM policies to manage settings for Cortana. For more info, see [Cortana, Search, and privacy: FAQ](#).

2.1 Cortana and Search Group Policies

Find the Cortana Group Policy objects under **Computer Configuration > Administrative Templates > Windows Components > Search**.

POLICY	DESCRIPTION
Allow Cortana	Choose whether to let Cortana install and run on the device. Disable this policy to turn off Cortana.
Allow search and Cortana to use location	Choose whether Cortana and Search can provide location-aware search results. Disable this policy to block access to location information for Cortana.
Do not allow web search	Choose whether to search the web from Windows Desktop Search. Enable this policy to remove the option to search the Internet from Cortana.
Don't search the web or display web results in Search	Choose whether to search the web from Cortana. Enable this policy to stop web queries and results from showing in Search.

POLICY	DESCRIPTION
Set what information is shared in Search	Control what information is shared with Bing in Search. If you enable this policy and set it to Anonymous info , usage information will be shared but not search history, Microsoft Account information, or specific location.

You can also apply the Group Policies using the following registry keys:

POLICY	REGISTRY PATH
Allow Cortana	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\Windows Search!AllowCortana REG_DWORD: 0
Allow search and Cortana to use location	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\Windows Search!AllowSearchToUseLocation REG_DWORD: 0
Do not allow web search	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\Windows Search!DisableWebSearch REG_DWORD: 1
Don't search the web or display web results in Search	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\Windows Search!ConnectedSearchUseWeb REG_DWORD: 0
Set what information is shared in Search	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\Windows Search!ConnectedSearchPrivacy REG_DWORD: 3

In Windows 10, version 1507 and Windows 10, version 1511, when you enable the **Don't search the web or display web results in Search** Group Policy, you can control the behavior of whether Cortana searches the web to display web results. However, this policy only covers whether or not web search is performed. There could still be a small amount of network traffic to Bing.com to evaluate if certain Cortana components are up-to-date or not. In order to turn off that network activity completely, you can create a Windows Firewall rule to prevent outbound traffic.

IMPORTANT

These steps are not required for devices running Windows 10, version 1607 or Windows Server 2016.

1. Expand **Computer Configuration > Windows Settings > Security Settings > Windows Firewall with Advanced Security > Windows Firewall with Advanced Security - <LDAP name>**, and then click **Outbound Rules**.
2. Right-click **Outbound Rules**, and then click **New Rule**. The **New Outbound Rule Wizard** starts.
3. On the **Rule Type** page, click **Program**, and then click **Next**.
4. On the **Program** page, click **This program path**, type **%windir%\systemapps\Microsoft.Windows.Cortana_cw5n1h2txyewy\SearchUI.exe**, and then click **Next**.
5. On the **Action** page, click **Block the connection**, and then click **Next**.
6. On the **Profile** page, ensure that the **Domain**, **Private**, and **Public** check boxes are selected, and then click **Next**.
7. On the **Name** page, type a name for the rule, such as **Cortana firewall configuration**, and then click **Finish**.
8. Right-click the new rule, click **Properties**, and then click **Protocols and Ports**.
9. Configure the **Protocols and Ports** page with the following info, and then click **OK**.
 - Protocol: **Internet (TCP)**
 - Local Port: **445**
 - Remote Port: **445**
 - Local Address: **Any**
 - Remote Address: **Any**
 - Source: **Windows Firewall**
 - Destination: **Windows Firewall**
 - Direction: **Outbound**
 - Priority: **Medium**
 - Profile: **All profiles**

- For **Protocol type**, choose **TCP**.
- For **Local port**, choose **All Ports**.
- For **Remote port**, choose **All ports**.

If your organization tests network traffic, do not use a network proxy as Windows Firewall does not block proxy traffic. Instead, use a network traffic analyzer. Based on your needs, there are many network traffic analyzers available at no cost.

2.2 Cortana and Search MDM policies

For Windows 10 only, the following Cortana MDM policies are available in the [Policy CSP](#).

POLICY	DESCRIPTION
Experience/AllowCortana	Choose whether to let Cortana install and run on the device.
Search/AllowSearchToUseLocation	Choose whether Cortana and Search can provide location-aware search results. Default: Allowed

3. Date & Time

You can prevent Windows from setting the time automatically.

- To turn off the feature in the UI: **Settings > Time & language > Date & time > Set time automatically**
-or-
• Create a REG_SZ registry setting in
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W32Time\Parameters\Type with a value of **NoSync**.

After that, configure the following:

- Disable the Group Policy: **Computer Configuration > Administrative Templates > System > Enable Windows NTP Server > Windows Time Service > Configure Windows NTP Client**

NOTE

This is only available on Windows 10, version 1703 and later. If you're using Windows 10, version 1607, the Group Policy setting is **Computer Configuration > Administrative Templates > System > Windows Time Service > Time Providers > Enable Windows NTP Client**

-or -

- Create a new REG_DWORD registry setting
HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\W32time\TimeProviders\NtpClient!Enabled and set it to 0 (zero).

4. Device metadata retrieval

To prevent Windows from retrieving device metadata from the Internet, apply the Group Policy: **Computer Configuration > Administrative Templates > System > Device Installation > Prevent device metadata retrieval from the Internet**.

You can also create a new REG_DWORD registry setting
HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\Device Metadata!PreventDeviceMetadataFromNetwork to 1 (one).

5. Find My Device

To turn off Find My Device:

- Turn off the feature in the UI

-or-

- Disable the Group Policy: **Computer Configuration > Administrative Template > Windows Components > Find My Device > Turn On/Off Find My Device**

You can also create a new REG_DWORD registry setting

HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\FindMyDevice\AllowFindMyDevice to 0 (zero).

6. Font streaming

Fonts that are included in Windows but that are not stored on the local device can be downloaded on demand.

If you're running Windows 10, version 1607, Windows Server 2016, or later:

- Disable the Group Policy: **Computer Configuration > Administrative Templates > Network > Fonts > Enable Font Providers**.
- Create a new REG_DWORD registry setting
HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\System\EnableFontProviders to 0 (zero).
- In Windows 10, version 1703, you can apply the System/AllowFontProviders MDM policy from the [Policy CSP](#) where:
 - **false**. Font streaming is disabled.
 - **true**. Font streaming is enabled.

If you're running Windows 10, version 1507 or Windows 10, version 1511, create a REG_DWORD registry setting called **DisableFontProviders** in **HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\FontCache\Parameters**, with a value of 1.

NOTE

After you apply this policy, you must restart the device for it to take effect.

7. Insider Preview builds

The Windows Insider Preview program lets you help shape the future of Windows, be part of the community, and get early access to releases of Windows 10.

NOTE

This setting stops communication with the Windows Insider Preview service that checks for new builds. Windows Insider Preview builds only apply to Windows 10 and are not available for Windows Server 2016.

To turn off Insider Preview builds for a released version of Windows 10:

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Data Collection and Preview Builds > Toggle user control over Insider builds**.

To turn off Insider Preview builds for Windows 10:

NOTE

If you're running a preview version of Windows 10, you must roll back to a released version before you can turn off Insider Preview builds.

- Turn off the feature in the UI: **Settings > Update & security > Windows Insider Program > Stop Insider Preview builds**.

-or-

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Data Collection and Preview Builds > Toggle user control over Insider builds**.

-or -

- Create a new REG_DWORD registry setting
HKEY_LOCAL_MACHINE\SOFTWARE\ Policies\Microsoft\Windows\PreviewBuilds!AllowBuildPreview to 0 (zero)
 -or-
- Apply the System/AllowBuildPreview MDM policy from the [Policy CSP](#) where:
 - 0. Users cannot make their devices available for downloading and installing preview software.
 - 1. Users can make their devices available for downloading and installing preview software.
 - 2. (default) Not configured. Users can make their devices available for download and installing preview software.
 -or-
- Create a provisioning package: **Runtime settings > Policies > System > AllowBuildPreview**, where:
 - 0. Users cannot make their devices available for downloading and installing preview software.
 - 1. Users can make their devices available for downloading and installing preview software.
 - 2. (default) Not configured. Users can make their devices available for download and installing preview software.

8. Internet Explorer

Use Group Policy to manage settings for Internet Explorer. You can find the Internet Explorer Group Policy objects under **Computer Configuration > Administrative Templates > Windows Components > Internet Explorer**.

POLICY	DESCRIPTION
Turn on Suggested Sites	Choose whether an employee can configure Suggested Sites. Default: Enabled You can also turn this off in the UI by clearing the Internet Options > Advanced > Enable Suggested Sites check box.
Allow Microsoft services to provide enhanced suggestions as the user types in the Address Bar	Choose whether an employee can configure enhanced suggestions, which are presented to the employee as they type in the address bar. Default: Enabled
Turn off the auto-complete feature for web addresses	Choose whether auto-complete suggests possible matches when employees are typing web address in the address bar. Default: Disabled You can also turn this off in the UI by clearing the Internet Options > Advanced > Use inline AutoComplete in the Internet Explorer Address Bar and Open Dialog check box.
Turn off browser geolocation	Choose whether websites can request location data from Internet Explorer. Default: Disabled
Prevent managing SmartScreen filter	Choose whether employees can manage the SmartScreen Filter in Internet Explorer. Default: Disabled

Alternatively, you could use the registry to set the Group Policies.

POLICY	REGISTRY PATH
Turn on Suggested Sites	HKEY_LOCAL_MACHINE\SOFTWARE\ Policies\Microsoft\Internet Explorer\Suggested Sites!Enabled REG_DWORD: 0

POLICY	REGISTRY PATH
Allow Microsoft services to provide enhanced suggestions as the user types in the Address Bar	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Internet Explorer\AllowServicePoweredQSA REG_DWORD: 0
Turn off the auto-complete feature for web addresses	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Explorer\AutoComplete!AutoSuggest REG_SZ: No
Turn off browser geolocation	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Internet Explorer\Geolocation!PolicyDisableGeolocation REG_DWORD: 1
Prevent managing SmartScreen filter	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Internet Explorer\PhishingFilter!EnabledV9 REG_DWORD: 0

There are three more Group Policy objects that are used by Internet Explorer:

PATH	POLICY	DESCRIPTION
Computer Configuration > Administrative Templates > Windows Components > Internet Explorer > Compatibility View > Turn off Compatibility View	Choose whether employees can configure Compatibility View.	Choose whether an employee can swipe across a screen or click forward to go to the next pre-loaded page of a website. Default: Disabled
Computer Configuration > Administrative Templates > Windows Components > Internet Explorer > Internet Control Panel > Advanced Page	Turn off the flip ahead with page prediction feature	Choose whether an employee can swipe across a screen or click forward to go to the next pre-loaded page of a website. Default: Enabled
Computer Configuration > Administrative Templates > Windows Components > RSS Feeds	Turn off background synchronization for feeds and Web Slices	Choose whether to have background synchronization for feeds and Web Slices. Default: Enabled

You can also use registry entries to set these Group Policies.

POLICY	REGISTRY PATH
Choose whether employees can configure Compatibility View.	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\MicrosoftEdge\BrowserEmulation!MSCompatibilityMode REG_DWORD: 0
Turn off the flip ahead with page prediction feature	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Internet Explorer\FlipAhead!Enabled REG_DWORD: 0
Turn off background synchronization for feeds and Web Slices	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Internet Explorer\Feeds!BackgroundSyncStatus REG_DWORD: 0

To turn off the home page, enable the Group Policy: **User Configuration > Administrative Templates > Windows Components > Internet Explorer > Disable changing home page settings**, and set it to **about:blank**.

To configure the First Run Wizard, enable the Group Policy: **User Configuration > Administrative Templates > Windows Components > Internet Explorer > Prevent running First Run wizard**, and set it to **Go directly to home page**.

To configure the behavior for a new tab, enable the Group Policy: **User Configuration > Administrative Templates > Windows Components > Internet Explorer > Specify default behavior for a new tab**, and set it to **about:blank**.

8.1 ActiveX control blocking

ActiveX control blocking periodically downloads a new list of out-of-date ActiveX controls that should be blocked.

You can turn this off by:

- Apply the Group Policy: **User Configuration > Administrative Templates > Windows Components > Internet Explorer > Security Features > Add-on Management > Turn off Automatic download of the ActiveX VersionList**

-or-
- Changing the REG_DWORD registry setting **HKEY_CURRENT_USER\Software\Microsoft\Internet Explorer\VersionManager\DownloadVersionList** to 0 (zero).

For more info, see [Out-of-date ActiveX control blocking](#).

9. Live Tiles

To turn off Live Tiles:

- Apply the Group Policy: **User Configuration > Administrative Templates > Start Menu and Taskbar > Notifications > Turn Off notifications network usage**

-or-
- Create a REG_DWORD registry setting called **HKEY_CURRENT_USER\Software\Policies\Microsoft\Windows\CurrentVersion\PushNotifications!NoCloudApplicationNotification**, with a value of 1 (one).

In Windows 10 Mobile, you must also unpin all tiles that are pinned to Start.

10. Mail synchronization

To turn off mail synchronization for Microsoft Accounts that are configured on a device:

- In **Settings > Accounts > Your email and accounts**, remove any connected Microsoft Accounts.

-or-
- Remove any Microsoft Accounts from the Mail app.

-or-
- Apply the Accounts/AllowMicrosoftAccountConnection MDM policy from the [Policy CSP](#) where 0 is not allowed and 1 is allowed. This does not apply to Microsoft Accounts that have already been configured on the device.

To turn off the Windows Mail app:

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Windows Mail > Turn off Windows Mail application**

-or-
- Create a REG_DWORD registry setting called **HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows Mail!ManualLaunchAllowed**, with a value of 0 (zero).

11. Microsoft Account

To prevent communication to the Microsoft Account cloud authentication service. Many apps and system components that depend on Microsoft Account authentication may lose functionality. Some of them could be in unexpected ways.

- Apply the Group Policy: **Computer Configuration > Windows Settings > Security Settings > Local Policies > Security Options > Accounts: Block Microsoft Accounts** and set it to **Users can't add Microsoft accounts**.

-or-
- Create a REG_DWORD registry setting called **HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Policies\System!NoConnectedUser**, with a value of 3. To disable the Microsoft Account Sign-In Assistant:

- Apply the Accounts/AllowMicrosoftAccountSignInAssistant MDM policy from the [Policy CSP](#) where 0 is turned off and 1 is turned on.
- Change the Start REG_DWORD registry setting in **HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\wlidsvc** to a value of **4**.

12. Microsoft Edge

Use either Group Policy or MDM policies to manage settings for Microsoft Edge. For more info, see [Microsoft Edge and privacy: FAQ](#).

12.1 Microsoft Edge Group Policies

Find the Microsoft Edge Group Policy objects under **Computer Configuration > Administrative Templates > Windows Components > Microsoft Edge**.

POLICY	DESCRIPTION
Configure Autofill	Choose whether employees can use autofill on websites. Default: Enabled
Configure Do Not Track	Choose whether employees can send Do Not Track headers. Default: Disabled
Configure Password Manager	Choose whether employees can save passwords locally on their devices. Default: Enabled
Configure search suggestions in Address bar	Choose whether the address bar shows search suggestions. Default: Enabled
Configure Windows Defender SmartScreen Filter (Windows 10, version 1703) Configure SmartScreen Filter (Windows Server 2016)	Choose whether Windows Defender SmartScreen is turned on or off. Default: Enabled
Allow web content on New Tab page	Choose whether a new tab page appears. Default: Enabled
Configure Start pages	Choose the Start page for domain-joined devices. Set this to <about:blank>
Prevent the First Run webpage from opening on Microsoft Edge	Choose whether employees see the First Run webpage. Default: Disabled

The Windows 10, version 1511 Microsoft Edge Group Policy names are:

POLICY	DESCRIPTION
Turn off autofill	Choose whether employees can use autofill on websites. Default: Enabled
Allow employees to send Do Not Track headers	Choose whether employees can send Do Not Track headers. Default: Disabled
Turn off password manager	Choose whether employees can save passwords locally on their devices. Default: Enabled
Turn off address bar search suggestions	Choose whether the address bar shows search suggestions. Default: Enabled
Turn off the SmartScreen Filter	Choose whether SmartScreen is turned on or off. Default: Enabled

POLICY	DESCRIPTION
Open a new tab with an empty tab	Choose whether a new tab page appears. Default: Enabled
Configure corporate Home pages	Choose the corporate Home page for domain-joined devices. Set this to about:blank

Alternatively, you can configure the Microsoft Group Policies using the following registry entries:

POLICY	REGISTRY PATH
Configure Autofill	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\MicrosoftEdge\Main!Use FormSuggest <br/ > REG_SZ: no
Configure Do Not Track	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\MicrosoftEdge\Main!DoNotTrack REG_DWORD: 1
Configure Password Manager	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\MicrosoftEdge\Main!FormSuggest Passwords REG_SZ: no
Configure search suggestions in Address bar	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\MicrosoftEdge\SearchScopes!ShowSearchSuggestionsGlobal REG_DWORD: 0
Configure Windows Defender SmartScreen Filter (Windows 10, version 1703)	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\MicrosoftEdge\PhishingFilter!EnabledV9 REG_DWORD: 0
Allow web content on New Tab page	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\MicrosoftEdge\SearchScopes!AllowWebContentOnNewTabPage REG_DWORD: 0
Configure corporate Home pages	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\MicrosoftEdge\ServiceUI!ProvisionedHomePages REG_DWORD: 0

12.2 Microsoft Edge MDM policies

The following Microsoft Edge MDM policies are available in the [Policy CSP](#).

POLICY	DESCRIPTION
Browser/AllowAutoFill	Choose whether employees can use autofill on websites. Default: Allowed
Browser/AllowDoNotTrack	Choose whether employees can send Do Not Track headers. Default: Not allowed
Browser/AllowMicrosoftCompatibilityList	Specify the Microsoft compatibility list in Microsoft Edge. Default: Enabled
Browser/AllowPasswordManager	Choose whether employees can save passwords locally on their devices. Default: Allowed
Browser/AllowSearchSuggestionsinAddressBar	Choose whether the address bar shows search suggestions.. Default: Allowed

POLICY	DESCRIPTION
Browser/AllowSmartScreen	Choose whether SmartScreen is turned on or off. Default: Allowed
Browser/FirstRunURL	Choose the home page for Microsoft Edge on Windows Mobile 10. Default: blank

For a complete list of the Microsoft Edge policies, see [Available policies for Microsoft Edge](#).

13. Network Connection Status Indicator

Network Connection Status Indicator (NCSI) detects Internet connectivity and corporate network connectivity status. NCSI sends a DNS request and HTTP query to <http://www.msftconnecttest.com/connecttest.txt> to determine if the device can communicate with the Internet. For more info about NCSI, see [The Network Connection Status Icon](#).

In versions of Windows 10 prior to Windows 10, version 1607 and Windows Server 2016, the URL was <http://www.msftncsi.com>.

You can turn off NCSI by doing one of the following:

- Enable the Group Policy: **Computer Configuration > Administrative Templates > System > Internet Communication Management > Internet Communication Settings > Turn off Windows Network Connectivity Status Indicator active tests**
- In Windows 10, version 1703 and later, apply the Connectivity/DisallowNetworkConnectivityActiveTests MDM policy.

NOTE

After you apply this policy, you must restart the device for the policy setting to take effect.

-or-

- Create a REG_DWORD registry setting called **HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\NetworkConnectivityStatusIndicator!NoActiveProbe** with a value of 1 (one).

14. Offline maps

You can turn off the ability to download and update offline maps.

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Maps > Turn off Automatic Download and Update of Map Data**

-or-

- Create a REG_DWORD registry setting called **HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\Maps!AutoDownloadAndUpdateMapData**, with a value of 0 (zero).

-and-

- In Windows 10, version 1607 and later, apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Maps > Turn off unsolicited network traffic on the Offline Maps settings page**

-or-

- Create a REG_DWORD registry setting called **HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\Maps!AllowUntriggeredNetworkTrafficOnSettingsPage**, with a value of 0 (zero).

15. OneDrive

To turn off OneDrive in your organization:

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > OneDrive > Prevent the usage of OneDrive for file storage**

-or-

- Create a REG_DWORD registry setting called
HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\OneDrive!DisableFileSyncNGSC, with a value of 1 (one).

-and-

- Create a REG_DWORD registry setting called
HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\OneDrive\PreventNetworkTrafficPreUserSignIn, with a value of 1 (one).

16. Preinstalled apps

Some preinstalled apps get content before they are opened to ensure a great experience. You can remove these using the steps in this section.

To remove the News app:

- Right-click the app in Start, and then click **Uninstall**.

-or-

- Remove the app for new user accounts. From an elevated command prompt, run the following Windows PowerShell command:
Get-AppxProvisionedPackage -Online | Where-Object {\$_ .PackageName -Like "Microsoft.BingNews"} | ForEach-Object { Remove-AppxProvisionedPackage -Online -PackageName \$_ .PackageName}

-and-

Remove the app for the current user. From an elevated command prompt, run the following Windows PowerShell command:
Get-AppxPackage Microsoft.BingNews | Remove-AppxPackage

To remove the Weather app:

- Remove the app for new user accounts. From an elevated command prompt, run the following Windows PowerShell command:
Get-AppxProvisionedPackage -Online | Where-Object {\$_ .PackageName -Like "Microsoft.BingWeather"} | ForEach-Object { Remove-AppxProvisionedPackage -Online -PackageName \$_ .PackageName}

-and-

Remove the app for the current user. From an elevated command prompt, run the following Windows PowerShell command:
Get-AppxPackage Microsoft.BingWeather | Remove-AppxPackage

To remove the Money app:

- Right-click the app in Start, and then click **Uninstall**.

-or-

- Remove the app for new user accounts. From an elevated command prompt, run the following Windows PowerShell command:
Get-AppxProvisionedPackage -Online | Where-Object {\$_ .PackageName -Like "Microsoft.BingFinance"} | ForEach-Object { Remove-AppxProvisionedPackage -Online -PackageName \$_ .PackageName}

-and-

Remove the app for the current user. From an elevated command prompt, run the following Windows PowerShell command:
Get-AppxPackage Microsoft.BingFinance | Remove-AppxPackage

To remove the Sports app:

- Right-click the app in Start, and then click **Uninstall**.

-or-

- Remove the app for new user accounts. From an elevated command prompt, run the following Windows PowerShell command:
Get-AppxProvisionedPackage -Online | Where-Object {\$_ .PackageName -Like "Microsoft.BingSports"} | ForEach-Object { Remove-AppxProvisionedPackage -Online -PackageName \$_ .PackageName}

-and-

Remove the app for the current user. From an elevated command prompt, run the following Windows PowerShell command: **Get-AppxPackage Microsoft.BingSports | Remove-AppxPackage**

To remove the Twitter app:

- Right-click the app in Start, and then click **Uninstall**.

-or-

- Remove the app for new user accounts. From an elevated command prompt, run the following Windows PowerShell command:
Get-AppxProvisionedPackage -Online | Where-Object {\$_ .PackageName -Like "* .Twitter"} | ForEach-Object { Remove-AppxProvisionedPackage -Online -PackageName \$_ .PackageName}

-and-

Remove the app for the current user. From an elevated command prompt, run the following Windows PowerShell command: **Get-AppxPackage * .Twitter | Remove-AppxPackage**

To remove the XBOX app:

- Remove the app for new user accounts. From an elevated command prompt, run the following Windows PowerShell command:
Get-AppxProvisionedPackage -Online | Where-Object {\$_ .PackageName -Like "Microsoft.XboxApp"} | ForEach-Object { Remove-AppxProvisionedPackage -Online -PackageName \$_ .PackageName}

-and-

Remove the app for the current user. From an elevated command prompt, run the following Windows PowerShell command: **Get-AppxPackage Microsoft.XboxApp | Remove-AppxPackage**

To remove the Sway app:

- Right-click the app in Start, and then click **Uninstall**.

-or-

- Remove the app for new user accounts. From an elevated command prompt, run the following Windows PowerShell command:
Get-AppxProvisionedPackage -Online | Where-Object {\$_ .PackageName -Like "Microsoft.Office.Sway"} | ForEach-Object { Remove-AppxProvisionedPackage -Online -PackageName \$_ .PackageName}

-and-

Remove the app for the current user. From an elevated command prompt, run the following Windows PowerShell command: **Get-AppxPackage Microsoft.Office.Sway | Remove-AppxPackage**

To remove the OneNote app:

- Remove the app for new user accounts. From an elevated command prompt, run the following Windows PowerShell command:
Get-AppxProvisionedPackage -Online | Where-Object {\$_ .PackageName -Like "Microsoft.Office.OneNote"} | ForEach-Object { Remove-AppxProvisionedPackage -Online -PackageName \$_ .PackageName}

-and-

Remove the app for the current user. From an elevated command prompt, run the following Windows PowerShell command: **Get-AppxPackage Microsoft.Office.OneNote | Remove-AppxPackage**

To remove the Get Office app:

- Right-click the app in Start, and then click **Uninstall**.
-or-
- Remove the app for new user accounts. From an elevated command prompt, run the following Windows PowerShell command:
Get-AppxProvisionedPackage -Online | Where-Object {\$_PackageName -Like "Microsoft.MicrosoftOfficeHub"} | ForEach-Object { Remove-AppxProvisionedPackage -Online -PackageName \$_PackageName}

-and-

Remove the app for the current user. From an elevated command prompt, run the following Windows PowerShell command: **Get-AppxPackage Microsoft.MicrosoftOfficeHub | Remove-AppxPackage**

To remove the Get Skype app:

- Right-click the Sports app in Start, and then click **Uninstall**.
-or-
- Remove the app for new user accounts. From an elevated command prompt, run the following Windows PowerShell command:
Get-AppxProvisionedPackage -Online | Where-Object {\$_PackageName -Like "Microsoft.SkypeApp"} | ForEach-Object { Remove-AppxProvisionedPackage -Online -PackageName \$_PackageName}

-and-

Remove the app for the current user. From an elevated command prompt, run the following Windows PowerShell command: **Get-AppxPackage Microsoft.SkypeApp | Remove-AppxPackage**

To remove the Sticky notes app:

- Remove the app for new user accounts. From an elevated command prompt, run the following Windows PowerShell command:
Get-AppxProvisionedPackage -Online | Where-Object {\$_PackageName -Like "Microsoft.MicrosoftStickyNotes"} | ForEach-Object { Remove-AppxProvisionedPackage -Online -PackageName \$_PackageName}

-and-

Remove the app for the current user. From an elevated command prompt, run the following Windows PowerShell command: **Get-AppxPackage Microsoft.MicrosoftStickyNotes | Remove-AppxPackage**

17. Settings > Privacy

Use Settings > Privacy to configure some settings that may be important to your organization. Except for the Feedback & Diagnostics page, these settings must be configured for every user account that signs into the PC.

- [17.1 General](#)
- [17.2 Location](#)
- [17.3 Camera](#)
- [17.4 Microphone](#)
- [17.5 Notifications](#)
- [17.6 Speech, inking, & typing](#)
- [17.7 Account info](#)
- [17.8 Contacts](#)
- [17.9 Calendar](#)
- [17.10 Call history](#)

- [17.11 Email](#)
- [17.12 Messaging](#)
- [17.13 Radios](#)
- [17.14 Other devices](#)
- [17.15 Feedback & diagnostics](#)
- [17.16 Background apps](#)
- [17.17 Motion](#)
- [17.18 Tasks](#)
- [17.19 App Diagnostics](#)

17.1 General

General includes options that don't fall into other areas.

Windows 10, version 1703 options

To turn off **Let apps use advertising ID to make ads more interesting to you based on your app usage (turning this off will reset your ID)**:

NOTE

When you turn this feature off in the UI, it turns off the advertising ID, not just resets it.

- Turn off the feature in the UI.
-or-
- Apply the Group Policy: **Computer Configuration > Administrative Templates > System > User Profiles > Turn off the advertising ID**.
-or-
- Create a REG_DWORD registry setting called **Enabled** in
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\AdvertisingInfo, with a value of 0 (zero).
-or-
- Create a REG_DWORD registry setting in
HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\AdvertisingInfo!DisabledByGroupPolicy, with a value of 1 (one).

To turn off **Let websites provide locally relevant content by accessing my language list**:

- Turn off the feature in the UI.
-or-
- Create a new REG_DWORD registry setting called **HttpAcceptLanguageOptOut** in
HKEY_CURRENT_USER\Control Panel\International\User Profile, with a value of 1.

To turn off **Let Windows track app launches to improve Start and search results**:

- Turn off the feature in the UI.
-or-
- Create a REG_DWORD registry setting called **Start_TrackProgs** with value of 0 (zero) in
HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\Advanced

To turn off **Let apps use my advertising ID for experiences across apps (turning this off will reset your ID)**:

NOTE

When you turn this feature off in the UI, it turns off the advertising ID, not just resets it.

- Turn off the feature in the UI.
-or-
- Apply the Group Policy: **Computer Configuration > Administrative Templates > System > User Profiles > Turn off the advertising ID**.
-or-
- Create a REG_DWORD registry setting called **Enabled** in
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\AdvertisingInfo, with a value of 0 (zero).
-or-
- Create a REG_DWORD registry setting in
HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\AdvertisingInfo!DisabledByGroupPolicy, with a value of 1 (one).

To turn off **Turn on SmartScreen Filter to check web content (URLs) that Microsoft Store apps use**:

- Turn off the feature in the UI.
-or-
- In Windows Server 2016, apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Microsoft Edge > Configure SmartScreen Filter**. In Windows 10, version 1703, apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Microsoft Edge > Configure Windows Defender SmartScreen Filter**.

In Windows Server 2016, apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > File Explorer > Configure Windows SmartScreen**. In Windows 10, version 1703 , apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > File Explorer > Configure Windows Defender SmartScreen**.

- or-
- Apply the Browser/AllowSmartScreen MDM policy from the [Policy CSP](#) where 0 is turned off and 1 is turned on.
-or-
 - Create a provisioning package, using:
 - For Internet Explorer: **Runtime settings > Policies > Browser > AllowSmartScreen**
 - For Microsoft Edge: **Runtime settings > Policies > MicrosoftEdge > AllowSmartScreen**
-or-
 - Create a REG_DWORD registry setting called **EnableWebContentEvaluation** in
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\AppHost, with a value of 0 (zero).
-or-
 - Create a REG_DWORD registry setting in
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\System!EnableSmartScreen, with a value of 0 (zero).

To turn off **Send Microsoft info about how I write to help us improve typing and writing in the future**:

NOTE

If the telemetry level is set to either **Basic** or **Security**, this is turned off automatically.

- Turn off the feature in the UI.
-or-
- Apply the TextInput/AllowLinguisticDataCollection MDM policy from the [Policy CSP](#) where:
 - **0.** Not allowed
 - **1.** Allowed (default)

To turn off **Let websites provide locally relevant content by accessing my language list**:

- Turn off the feature in the UI.
-or-
- Create a new REG_DWORD registry setting called **HttpAcceptLanguageOptOut** in **HKEY_CURRENT_USER\Control Panel\International\User Profile**, with a value of 1.

To turn off **Let apps on my other devices open apps and continue experiences on this devices**:

- Turn off the feature in the UI.
-or-
- Disable the Group Policy: **Computer Configuration > Administrative Templates > System > Group Policy > Continue experiences on this device**.
-or-
- Create a REG_DWORD registry setting in **HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\System!EnableCdp**, with a value of 0 (zero).

To turn off **Let apps on my other devices use Bluetooth to open apps and continue experiences on this device**:

- Turn off the feature in the UI.

17.2 Location

In the **Location** area, you choose whether devices have access to location-specific sensors and which apps have access to the device's location.

To turn off **Location for this device**:

- Click the **Change** button in the UI.
-or-
- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Location and Sensors > Turn off location**.
-or-
- Create a REG_DWORD registry setting in **HKEY_LOCAL_MACHINE\Policies\Microsoft\Windows\AppPrivacy!LetAppsAccessLocation**, with a value of 2 (two).
-or-
- Apply the System/AllowLocation MDM policy from the [Policy CSP](#), where:
 - **0.** Turned off and the employee can't turn it back on.
 - **1.** Turned on, but lets the employee choose whether to use it. (default)

- 2. Turned on and the employee can't turn it off.

NOTE

You can also set this MDM policy in System Center Configuration Manager using the [WMI Bridge Provider](#).

-or-

- Create a provisioning package, using **Runtime settings > Policies > System > AllowLocation**, where
 - **No.** Turns off location service.
 - **Yes.** Turns on location service. (default)

To turn off **Location**:

- Turn off the feature in the UI.
- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps access location**
 - Set the **Select a setting** box to **Force Deny**.

-or-

- Create a REG_DWORD registry setting in
HKEY_LOCAL_MACHINE\Policies\Microsoft\Windows\LocationAndSensors!DisableLocation, with a value of 1 (one).

-or-

To turn off **Location history**:

- Erase the history using the **Clear** button in the UI.

To turn off **Choose apps that can use your location**:

- Turn off each app using the UI.

17.3 Camera

In the **Camera** area, you can choose which apps can access a device's camera.

To turn off **Let apps use my camera**:

- Turn off the feature in the UI.
- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps access the camera**
 - Set the **Select a setting** box to **Force Deny**.
- Create a REG_DWORD registry setting in
HKEY_LOCAL_MACHINE\Policies\Microsoft\Windows\AppPrivacy!LetAppsAccessCamera, with a value of 2 (two).
- Apply the Camera/AllowCamera MDM policy from the [Policy CSP](#), where:
 - **0.** Apps can't use the camera.
 - **1.** Apps can use the camera.

NOTE

You can also set this MDM policy in System Center Configuration Manager using the [WMI Bridge Provider](#).

-or-

- Create a provisioning package with use Windows ICD, using **Runtime settings > Policies > Camera > AllowCamera**, where:

- **0.** Apps can't use the camera.
- **1.** Apps can use the camera.

To turn off **Choose apps that can use your camera**:

- Turn off the feature in the UI for each app.

17.4 Microphone

In the **Microphone** area, you can choose which apps can access a device's microphone.

To turn off **Let apps use my microphone**:

- Turn off the feature in the UI.

-or-

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps access the microphone**

- Set the **Select a setting** box to **Force Deny**.

-or-

- Apply the Privacy/LetAppsAccessMicrophone MDM policy from the [Policy CSP](#), where:

- **0.** User in control
- **1.** Force allow
- **2.** Force deny

-or-

- Create a REG_DWORD registry setting in
HKEY_LOCAL_MACHINE\Policies\Microsoft\Windows\AppPrivacy!LetAppsAccessMicrophone, with a value of 2 (two)

To turn off **Choose apps that can use your microphone**:

- Turn off the feature in the UI for each app.

17.5 Notifications

In the **Notifications** area, you can choose which apps have access to notifications.

To turn off **Let apps access my notifications**:

- Turn off the feature in the UI.

-or-

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps access notifications**

- Set the **Select a setting** box to **Force Deny**.

-or-

- Apply the Privacy/LetAppsAccessNotifications MDM policy from the [Policy CSP](#), where:

- **0.** User in control

- 1. Force allow
 - 2. Force deny
- or-
- Create a REG_DWORD registry setting in
HKEY_LOCAL_MACHINE\Policies\Microsoft\Windows\AppPrivacy!LetAppsAccessNotifications, with a value of 2 (two)

17.6 Speech, inking, & typing

In the **Speech, Inking, & Typing** area, you can let Windows and Cortana better understand your employee's voice and written input by sampling their voice and writing, and by comparing verbal and written input to contact names and calendar entrees.

NOTE

For more info on how to disable Cortana in your enterprise, see [Cortana](#) in this article.

To turn off the functionality:

- Click the **Stop getting to know me** button, and then click **Turn off**.
- or-
- Enable the Group Policy: **Computer Configuration > Administrative Templates > Control Panel > Regional and Language Options > Handwriting personalization > Turn off automatic learning**
- or-
- Create a REG_DWORD registry setting in
HKEY_LOCAL_MACHINE\Policies\Microsoft\InputPersonalization!RestrictImplicitInkCollection, with a value of 1 (one).
- or-
- Create a REG_DWORD registry setting called **AcceptedPrivacyPolicy** in
HKEY_CURRENT_USER\Software\Microsoft\Personalization\Settings, with a value of 0 (zero).
- and-
- Create a REG_DWORD registry setting called **HarvestContacts** in
HKEY_CURRENT_USER\Software\Microsoft\InputPersonalization\TrainedDataStore, with a value of 0 (zero).

If you're running at least Windows 10, version 1703, you can turn off updates to the speech recognition and speech synthesis models:

- Disable the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Speech > Allow automatically update of Speech Data**

If you're running at least Windows 10, version 1607, you can turn off updates to the speech recognition and speech synthesis models:

Apply the Speech/AllowSpeechModelUpdate MDM policy from the [Policy CSP](#), where:

- **0** (default). Not allowed.
 - **1**. Allowed.
- or-
- Create a REG_DWORD registry setting called **ModelDownloadAllowed** in
HKEY_LOCAL_MACHINE\Software\Microsoft\Speech_OneCore\Preferences, with a value of 0 (zero).

17.7 Account info

In the **Account Info** area, you can choose which apps can access your name, picture, and other account info.

To turn off **Let apps access my name, picture, and other account info**:

- Turn off the feature in the UI.
-or-
 - Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps access account information**
 - Set the **Select a setting** box to **Force Deny**.
- Apply the Privacy/LetAppsAccessAccountInfo MDM policy from the [Policy CSP](#), where:
 - **0.** User in control
 - **1.** Force allow
 - **2.** Force deny
-or-
- Create a REG_DWORD registry setting in
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\AppPrivacy!LetAppsAccessAccountInfo, with a value of 2 (two).

To turn off **Choose the apps that can access your account info**:

- Turn off the feature in the UI for each app.

17.8 Contacts

In the **Contacts** area, you can choose which apps can access an employee's contacts list.

To turn off **Choose apps that can access contacts**:

- Turn off the feature in the UI for each app.
-or-
 - Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps access contacts**
 - Set the **Select a setting** box to **Force Deny**.
- Apply the Privacy/LetAppsAccessContacts MDM policy from the [Policy CSP](#), where:
 - **0.** User in control
 - **1.** Force allow
 - **2.** Force deny

17.9 Calendar

In the **Calendar** area, you can choose which apps have access to an employee's calendar.

To turn off **Let apps access my calendar**:

- Turn off the feature in the UI.
-or-
 - Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps access the calendar**
 - Set the **Select a setting** box to **Force Deny**.
- Apply the Privacy/LetAppsAccessCalendar MDM policy from the [Policy CSP](#), where:

- **0.** User in control
 - **1.** Force allow
 - **2.** Force deny
- or-
- Create a REG_DWORD registry setting in
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\AppPrivacy!LetAppsAccessCalendar, with a value of 2 (two).

To turn off **Choose apps that can access calendar**:

- Turn off the feature in the UI for each app.

17.10 Call history

In the **Call history** area, you can choose which apps have access to an employee's call history.

To turn off **Let apps access my call history**:

- Turn off the feature in the UI.
- or-
- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps access call history**
 - Set the **Select a setting** box to **Force Deny**.
 - or-
 - Apply the Privacy/LetAppsAccessCallHistory MDM policy from the [Policy CSP](#), where:
 - **0.** User in control
 - **1.** Force allow
 - **2.** Force deny
- or-
- Create a REG_DWORD registry setting in
HKEY_LOCAL_MACHINE\Policies\Microsoft\Windows\AppPrivacy!LetAppsAccessCallHistory, with a value of 2 (two).

17.11 Email

In the **Email** area, you can choose which apps have can access and send email.

To turn off **Let apps access and send email**:

- Turn off the feature in the UI.
- or-
- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps access email**
 - Set the **Select a setting** box to **Force Deny**.
 - or-
 - Apply the Privacy/LetAppsAccessEmail MDM policy from the [Policy CSP](#), where:
 - **0.** User in control
 - **1.** Force allow
 - **2.** Force deny
- or-
- Create a REG_DWORD registry setting in

HKEY_LOCAL_MACHINE\Policies\Microsoft\Windows\AppPrivacy!LetAppsAccessEmail, with a value of 2 (two).

17.12 Messaging

In the **Messaging** area, you can choose which apps can read or send messages.

To turn off **Let apps read or send messages (text or MMS)**:

- Turn off the feature in the UI.
-or-
- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps access messaging**
 - Set the **Select a setting** box to **Force Deny**.
 - or-
- Apply the Privacy/LetAppsAccess<Messaging MDM policy from the [Policy CSP](#), where:
 - **0.** User in control
 - **1.** Force allow
 - **2.** Force deny
-or-
- Create a REG_DWORD registry setting in
HKEY_LOCAL_MACHINE\Software\ Policies\Microsoft\Windows\AppPrivacy!LetAppsAccessMessaging, with a value of 2 (two).

To turn off **Choose apps that can read or send messages**:

- Turn off the feature in the UI for each app.

17.13 Phone calls

In the **Phone calls** area, you can choose which apps can make phone calls.

To turn off **Let apps make phone calls**:

- Turn off the feature in the UI.
-or-
- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps make phone calls**
 - Set the **Select a setting** box to **Force Deny**.
 - or-
- Apply the Privacy/LetAppsAccessPhone MDM policy from the [Policy CSP](#), where:
 - **0.** User in control
 - **1.** Force allow
 - **2.** Force deny
-or-
- Create a REG_DWORD registry setting in
HKEY_LOCAL_MACHINE\ Policies\Microsoft\Windows\AppPrivacy!LetAppsAccessPhone, with a value of 2 (two).

To turn off **Choose apps that can make phone calls**:

- Turn off the feature in the UI for each app.

17.14 Radios

In the **Radios** area, you can choose which apps can turn a device's radio on or off.

To turn off **Let apps control radios**:

- Turn off the feature in the UI.
-or-
- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps control radios**
 - Set the **Select a setting** box to **Force Deny**.
-or-
- Apply the Privacy/LetAppsAccessRadios MDM policy from the [Policy CSP](#), where:
 - **0.** User in control
 - **1.** Force allow
 - **2.** Force deny
-or-
- Create a REG_DWORD registry setting in
HKEY_LOCAL_MACHINE\Policies\Microsoft\Windows\AppPrivacy!LetAppsAccessRadios, with a value of 2 (two).

To turn off **Choose apps that can control radios**:

- Turn off the feature in the UI for each app.

17.15 Other devices

In the **Other Devices** area, you can choose whether devices that aren't paired to PCs, such as an Xbox One, can share and sync info.

To turn off **Let apps automatically share and sync info with wireless devices that don't explicitly pair with your PC, tablet, or phone**:

- Turn off the feature in the UI.
-or-
- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps sync with devices**
-or-
- Apply the Privacy/LetAppsSyncWithDevices MDM policy from the [Policy CSP](#), where:
 - **0.** User in control
 - **1.** Force allow
 - **2.** Force deny
-or-
- Create a REG_DWORD registry setting in
HKEY_LOCAL_MACHINE\Policies\Microsoft\Windows\AppPrivacy!LetAppsSyncWithDevices, with a value of 2 (two).

To turn off **Let your apps use your trusted devices (hardware you've already connected, or comes with your PC, tablet, or phone)**:

- Turn off the feature in the UI.
-or-
- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps access trusted devices**
 - Set the **Select a setting** box to **Force Deny**.

17.16 Feedback & diagnostics

In the **Feedback & Diagnostics** area, you can choose how often you're asked for feedback and how much diagnostic and usage information is sent to Microsoft.

To change how frequently **Windows should ask for my feedback**:

NOTE

Feedback frequency only applies to user-generated feedback, not diagnostic and usage data sent from the device.

- To change from **Automatically (Recommended)**, use the drop-down list in the UI.
-or-
- Enable the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Data Collection and Preview Builds > Do not show feedback notifications**
-or-
- Create a REG_DWORD registry setting in
HKEY_LOCAL_MACHINE\ Policies\Microsoft\Windows\ DataCollection!DoNotShowFeedbackNotifications,
with a value of 1 (one).
-or-
- Create the registry keys (REG_DWORD type):
 - HKEY_CURRENT_USER\Software\Microsoft\Siuf\Rules\PeriodInNanoSeconds
 - HKEY_CURRENT_USER\Software\Microsoft\Siuf\Rules\NumberOfSIUFInPeriod

Based on these settings:

SETTING	PERIODINNANOSECONDS	NUMBEROFSIUFINPERIOD
Automatically	Delete the registry setting	Delete the registry setting
Never	0	0
Always	100000000	Delete the registry setting
Once a day	864000000000	1
Once a week	604800000000	1

To change the level of diagnostic and usage data sent when you **Send your device data to Microsoft**:

- Click either the **Basic** or **Full** options.
-or-
- Apply the Group Policy: **Computer Configuration\Administrative Templates\Windows Components\ Data Collection And Preview Builds\Allow Telemetry**
-or-
- Create a REG_DWORD registry setting in
HKEY_LOCAL_MACHINE\ Software\ Policies\Microsoft\Windows\ DataCollection\AllowTelemetry, with a
value of 0 (zero).
-or-
- Apply the System/AllowTelemetry MDM policy from the [Policy CSP](#), where:

- **0.** Maps to the **Security** level.
- **1.** Maps to the **Basic** level.
- **2.** Maps to the **Enhanced** level.
- **3.** Maps to the **Full** level.

-or-

- Create a provisioning package, using **Runtime settings > Policies > System > AllowTelemetry**, where:
 - **0.** Maps to the **Security** level.
 - **1.** Maps to the **Basic** level.
 - **2.** Maps to the **Enhanced** level.
 - **3.** Maps to the **Full** level.

To turn off tailored experiences with relevant tips and recommendations by using your diagnostics data:

- Turn off the feature in the UI.
- or-
- Apply the Group Policy: **User Configuration > Administrative Templates > Windows Components > Cloud Content > Do not use diagnostic data for tailored experiences**

17.17 Background apps

In the **Background Apps** area, you can choose which apps can run in the background.

To turn off **Let apps run in the background**:

- Turn off the feature in the UI for each app.
- or-
- Apply the Group Policy (only applicable for Windows 10, version 1703): **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps run in the background**
 - Set the **Select a setting** box to **Force Deny**.
- or-
- Apply the Privacy/LetAppsRunInBackground MDM policy from the [Policy CSP](#), where:
 - **0.** User in control
 - **1.** Force allow
 - **2.** Force deny

17.18 Motion

In the **Motion** area, you can choose which apps have access to your motion data.

To turn off **Let Windows and your apps use your motion data and collect motion history**:

- Turn off the feature in the UI.
- or-
- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps access motion**
- or-
- Apply the Privacy/LetAppsAccessMotion MDM policy from the [Policy CSP](#), where:
 - **0.** User in control

- **1.** Force allow
 - **2.** Force deny
- or-
- Create a REG_DWORD registry setting in
HKEY_LOCAL_MACHINE\Policies\Microsoft\Windows\AppPrivacy!LetAppsAccessMotion, with a value of 2 (two).

17.19 Tasks

In the **Tasks** area, you can choose which apps have access to your tasks.

To turn this off:

- Turn off the feature in the UI.
- or-
- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps access Tasks**
 - Set the **Select a setting** box to **Force Deny**.
- or-
- Apply the Privacy/LetAppsAccessTasks MDM policy from the [Policy CSP](#), where:
 - **0.** User in control
 - **1.** Force allow
 - **2.** Force deny

17.20 App Diagnostics

In the **App diagnostics** area, you can choose which apps have access to your diagnostic information.

To turn this off:

- Turn off the feature in the UI.
- or-
- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > App Privacy > Let Windows apps access diagnostic information about other apps**
- or-
- Apply the Privacy/LetAppsGetDiagnosticInfo MDM policy from the [Policy CSP](#), where:
 - **0.** User in control
 - **1.** Force allow
 - **2.** Force deny

18. Software Protection Platform

Enterprise customers can manage their Windows activation status with volume licensing using an on-premise Key Management Server. You can opt out of sending KMS client activation data to Microsoft automatically by doing one of the following:

For Windows 10:

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Software Protection Platform > Turn off KMS Client Online AVS Validation**
- or-
- Create a REG_DWORD registry setting in
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\AppPrivacy!LetAppsAccessContacts, with a value of 2 (two).

-or-

- Apply the Licensing/DisallowKMSClientOnlineAVSValidation MDM policy from the [Policy CSP](#) where 0 is disabled (default) and 1 is enabled.

For Windows Server 2016 with Desktop Experience or Windows Server 2016 Server Core:

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Software Protection Platform > Turn off KMS Client Online AVS Validation**

-or-

- Create a REG_DWORD registry setting in **HKEY_LOCAL_MACHINE\Policies\Microsoft\Windows NT\CurrentVersion\Software Protection Platform!NoGenTicket**, with a value of 1 (one).

The Windows activation status will be valid for a rolling period of 180 days with weekly activation status checks to the KMS.

19. Storage health

Enterprise customers can manage updates to the Disk Failure Prediction Model.

For Windows 10:

- Apply the Group Policy: **Computer Configuration > Administrative Templates > System > Storage Health > Allow downloading updates to the Disk Failure Prediction Model**

20. Sync your settings

You can control if your settings are synchronized:

- In the UI: **Settings > Accounts > Sync your settings**

-or-

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Sync your settings > Do not sync**

-or-

- Create a REG_DWORD registry setting in **HKEY_LOCAL_MACHINE\Policies\Microsoft\Windows\SettingSync!DisableSettingSync**, with a value of 2 (two) and **HKEY_LOCAL_MACHINE\Policies\Microsoft\Windows\SettingSync!DisableSettingSyncUserOverride**, with a value of 1 (one).

-or-

- Apply the Experience/AllowSyncMySettings MDM policy from the [Policy CSP](#) where 0 is not allowed and 1 is allowed.

-or-

- Create a provisioning package, using **Runtime settings > Policies > Experience > AllowSyncMySettings**, where
 - **No.** Settings are not synchronized.
 - **Yes.** Settings are synchronized. (default)

To turn off Messaging cloud sync:

- Create a REG_DWORD registry setting called **CloudServiceSyncEnabled** in **HKEY_CURRENT_USER\SOFTWARE\Microsoft\Messaging**, with a value of 0 (zero).

21. Teredo

You can disable Teredo by using Group Policy or by using the netsh.exe command. For more info on Teredo, see [Internet Protocol Version 6, Teredo, and Related Technologies](#).

NOTE

If you disable Teredo, some XBOX gaming features and Windows Update Delivery Optimization will not work.

- Enable the Group Policy: **Computer Configuration > Administrative Templates > Network > TCPIP Settings > IPv6 Transition Technologies > Set Teredo State** and set it to **Disabled State**.
-or-
- Create a new REG_SZ registry setting called in
HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\TCPIP\v6Transition!Teredo_State, with a value of **Disabled**.
-or-
- From an elevated command prompt, run **netsh interface teredo set state disabled**

22. Wi-Fi Sense

Wi-Fi Sense automatically connects devices to known hotspots and to the wireless networks the person's contacts have shared with them.

To turn off **Connect to suggested open hotspots** and **Connect to networks shared by my contacts**:

- Turn off the feature in the UI.
-or-
- Disable the Group Policy: **Computer Configuration > Administrative Templates > Network > WLAN Service > WLAN Settings > Allow Windows to automatically connect to suggested open hotspots, to networks shared by contacts, and to hotspots offering paid services**.
-or-
- Create a new REG_DWORD registry setting called **AutoConnectAllowedOEM** in
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\WcmSvc\wifinetworkmanager\config, with a value of 0 (zero).
-or-
- Change the Windows Provisioning setting, WiFiSenseAllowed, to 0 (zero). For more info, see the Windows Provisioning Settings reference doc, [WiFiSenseAllowed](#).
-or-
- Use the Unattended settings to set the value of WiFiSenseAllowed to 0 (zero). For more info, see the Unattended Windows Setup reference doc, [WiFiSenseAllowed](#).

When turned off, the Wi-Fi Sense settings still appear on the Wi-Fi Settings screen, but they're non-functional and they can't be controlled by the employee.

23. Windows Defender

You can disconnect from the Microsoft Antimalware Protection Service.

- Disable the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Windows Defender Antivirus > MAPS > Join Microsoft MAPS**
-or-
- Delete the registry setting **HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows Defender\Updates!DefinitionUpdateFileSharesSources**.
-or-
- For Windows 10 only, apply the Defender/AllowCloudProtection MDM policy from the [Defender CSP](#).
-or-

- Use the registry to set the REG_DWORD value **HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows Defender\Spynet\SpyNetReporting** to 0 (zero).

-and-

From an elevated Windows PowerShell prompt, run **set-mppreference -Mapsreporting 0**

You can stop sending file samples back to Microsoft.

- Set the Group Policy **Computer Configuration > Administrative Templates > Windows Components > Windows Defender Antivirus > MAPS > Send file samples when further analysis is required** to **Always Prompt or Never Send**.

-or-

- For Windows 10 only, apply the Defender/SubmitSamplesConsent MDM policy from the [Policy CSP](#), where:

- **0.** Always prompt.
- **1.** (default) Send safe samples automatically.
- **2.** Never send.
- **3.** Send all samples automatically.

-or-

- Use the registry to set the REG_DWORD value **HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows Defender\Spynet\SubmitSamplesConsent** to 0 (zero) to always prompt or 2 to never send.

You can stop downloading definition updates:

- Enable the Group Policy **Computer Configuration > Administrative Templates > Windows Components > Windows Defender Antivirus > Signature Updates > Define the order of sources for downloading definition updates** and set it to **FileShares**.

-and-

- Disable the Group Policy **Computer Configuration > Administrative Templates > Windows Components > Windows Defender Antivirus > Signature Updates > Define file shares for downloading definition updates** and set it to nothing.

-or-

- Create a new REG_SZ registry setting in **HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows Defender\Updates!FallbackOrder**, with a value of **FileShares**.

For Windows 10 only, you can stop Enhanced Notifications:

- Turn off the feature in the UI.

You can also use the registry to turn off Malicious Software Reporting Tool telemetry by setting the REG_DWORD value **HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\MRT\DontReportInfectionInformation** to 1.

24. Windows Media Player

To remove Windows Media Player on Windows 10:

- From the **Programs and Features** control panel, click **Turn Windows features on or off**, under **Media Features**, clear the **Windows Media Player** check box, and then click **OK**.

-or-

- Run the following DISM command from an elevated command prompt: **dism /online /Disable-Feature /FeatureName:WindowsMediaPlayer**

To remove Windows Media Player on Windows Server 2016:

- Run the following DISM command from an elevated command prompt: **dism /online /Disable-Feature /FeatureName:WindowsMediaPlayer**

25. Windows Spotlight

Windows Spotlight provides features such as different background images and text on the lock screen, suggested apps, Microsoft account notifications, and Windows tips. You can control it by using the user interface, MDM policy, or through Group Policy.

If you're running Windows 10, version 1607 or later, you only need to enable the following Group Policy:

- User Configuration > Administrative Templates > Windows Components > Cloud Content > Turn off all Windows spotlight features**

NOTE

This must be done within 15 minutes after Windows 10 is installed. Alternatively, you can create an image with this setting.

-or-

- For Windows 10 only, apply the Experience/AllowWindowsSpotlight MDM policy from the **Policy CSP**, with a value of 0 (zero).

-or-

- Create a new REG_DWORD registry setting in **HKEY_CURRENT_USER\Software\Policies\Microsoft\Windows\CloudContent!DisableWindowsSpotlightFeatures**, with a value of 1 (one).

If you're not running Windows 10, version 1607 or later, you can use the other options in this section.

- Configure the following in **Settings**:

- Personalization > Lock screen > Background > Windows spotlight**, select a different background, and turn off **Get fun facts, tips, tricks and more on your lock screen**.

NOTE

In Windows 10, version 1507 and Windows 10, version 1511, this setting was called **Show me tips, tricks, and more on the lock screen**.

- Personalization > Start > Occasionally show suggestions in Start**.

- System > Notifications & actions > Show me tips about Windows**.

-or-

- Apply the Group Policies:

- Computer Configuration > Administrative Templates > Control Panel > Personalization > Force a specific default lock screen image**.

- Add a location in the **Path to local lock screen image** box.

- Set the **Turn off fun facts, tips, tricks, and more on lock screen** check box.

NOTE

This will only take effect if the policy is applied before the first logon. If you cannot apply the **Force a specific default lock screen image** policy before the first logon to the device, you can apply this policy: **Computer Configuration > Administrative Templates > Control Panel > Personalization > Do not display the lock screen**. Alternatively, you can create a new REG_SZ registry setting in **HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\Personalization!LockScreenImage**, with a value of **C:\windows\web\screen\lockscreen.jpg** and create a new REG_DWORD registry setting in **HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\Personalization!LockScreenOverlaysDisable**, with a value of 1 (one).

- **Computer Configuration > Administrative Templates > Windows Components > Cloud Content > Do not show Windows tips.**

-or-

- Create a new REG_DWORD registry setting in **HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\CloudContent!DisableSoftLanding**, with a value of 1 (one).

- **Computer Configuration > Administrative Templates > Windows Components > Cloud Content > Turn off Microsoft consumer experiences.**

-or-

- Create a new REG_DWORD registry setting in **HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\CloudContent!DisableWindowsConsumerFeatures**, with a value of 1 (one).

For more info, see [Windows Spotlight on the lock screen](#).

26. Microsoft Store

You can turn off the ability to launch apps from the Microsoft Store that were preinstalled or downloaded. This will also turn off automatic app updates, and the Microsoft Store will be disabled. On Windows Server 2016, this will block Microsoft Store calls from Universal Windows Apps.

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Store > Disable all apps from Microsoft Store.**

-or-

- Create a new REG_DWORD registry setting in **HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\WindowsStore!DisableStoreApps**, with a value of 1 (one).

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Store > Turn off Automatic Download and Install of updates.**

-or-

- Create a new REG_DWORD registry setting in **HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\WindowsStore!AutoDownload**, with a value of 2 (two).

Disable the Group Policy: **Computer Configuration > Administrative Templates > System > Group Policy > Configure web-to-app linking with URI handlers**

27. Windows Update Delivery Optimization

Windows Update Delivery Optimization lets you get Windows updates and Microsoft Store apps from sources in addition to Microsoft, which not only helps when you have a limited or unreliable Internet connection, but can also help you reduce the amount of bandwidth needed to keep all of your organization's PCs up-to-date. If you have Delivery Optimization turned on, PCs on your network may send and receive updates and apps to other PCs on your local network, if you choose, or to PCs on the Internet.

By default, PCs running Windows 10 Enterprise and Windows 10 Education will only use Delivery Optimization to get and receive updates for PCs and apps on your local network.

Use the UI, Group Policy, MDM policies, or Windows Provisioning to set up Delivery Optimization.

In Windows 10, version 1607, you can stop network traffic related to Windows Update Delivery Optimization by setting **Download Mode** to **Simple** (99) or **Bypass** (100), as described below.

27.1 Settings > Update & security

You can set up Delivery Optimization from the **Settings** UI.

- Go to **Settings** > **Update & security** > **Windows Update** > **Advanced options** > **Choose how updates are delivered**.

27.2 Delivery Optimization Group Policies

You can find the Delivery Optimization Group Policy objects under **Computer Configuration** > **Administrative Templates** > **Windows Components** > **Delivery Optimization**.

POLICY	DESCRIPTION
Download Mode	Lets you choose where Delivery Optimization gets or sends updates and apps, including <ul style="list-style-type: none">• None. Turns off Delivery Optimization.• Group. Gets or sends updates and apps to PCs on the same local network domain.• Internet. Gets or sends updates and apps to PCs on the Internet.• LAN. Gets or sends updates and apps to PCs on the same NAT only.• Simple. Simple download mode with no peering.• Bypass. Use BITS instead of Windows Update Delivery Optimization.
Group ID	Lets you provide a Group ID that limits which PCs can share apps and updates. Note: This ID must be a GUID.
Max Cache Age	Lets you specify the maximum time (in seconds) that a file is held in the Delivery Optimization cache. The default value is 259200 seconds (3 days).
Max Cache Size	Lets you specify the maximum cache size as a percentage of disk size. The default value is 20, which represents 20% of the disk.
Max Upload Bandwidth	Lets you specify the maximum upload bandwidth (in KB/second) that a device uses across all concurrent upload activity. The default value is 0, which means unlimited possible bandwidth.

You can also set the **Download Mode** policy by creating a new REG_DWORD registry setting in **HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\DeliveryOptimization!DODownloadMode**, with a value of 100 (one hundred).

27.3 Delivery Optimization MDM policies

The following Delivery Optimization MDM policies are available in the [Policy CSP](#).

POLICY	DESCRIPTION
--------	-------------

POLICY	DESCRIPTION
DeliveryOptimization/DODownloadMode	Lets you choose where Delivery Optimization gets or sends updates and apps, including <ul style="list-style-type: none"> • 0. Turns off Delivery Optimization. • 1. Gets or sends updates and apps to PCs on the same NAT only. • 2. Gets or sends updates and apps to PCs on the same local network domain. • 3. Gets or sends updates and apps to PCs on the Internet. • 99. Simple download mode with no peering. • 100. Use BITS instead of Windows Update Delivery Optimization.
DeliveryOptimization/DOGroupID	Lets you provide a Group ID that limits which PCs can share apps and updates. Note This ID must be a GUID.
DeliveryOptimization/DOMaxCacheAge	Lets you specify the maximum time (in seconds) that a file is held in the Delivery Optimization cache. The default value is 259200 seconds (3 days).
DeliveryOptimization/DOMaxCacheSize	Lets you specify the maximum cache size as a percentage of disk size. The default value is 20, which represents 20% of the disk.
DeliveryOptimization/DOMaxUploadBandwidth	Lets you specify the maximum upload bandwidth (in KB/second) that a device uses across all concurrent upload activity. The default value is 0, which means unlimited possible bandwidth.

27.4 Delivery Optimization Windows Provisioning

If you don't have an MDM server in your enterprise, you can use Windows Provisioning to configure the Delivery Optimization policies

Use Windows ICD, included with the [Windows Assessment and Deployment Kit \(Windows ADK\)](#), to create a provisioning package for Delivery Optimization.

1. Open Windows ICD, and then click **New provisioning package**.
2. In the **Name** box, type a name for the provisioning package, and then click **Next**.
3. Click the **Common to all Windows editions** option, click **Next**, and then click **Finish**.
4. Go to **Runtime settings > Policies > DeliveryOptimization** to configure the policies.

For more info about Delivery Optimization in general, see [Windows Update Delivery Optimization: FAQ](#).

28. Windows Update

You can turn off Windows Update by setting the following registry entries:

- Add a REG_DWORD value called **DoNotConnectToWindowsUpdateInternetLocations** to **HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\WindowsUpdate** and set the value to 1.
-and-
- Add a REG_DWORD value called **DisableWindowsUpdateAccess** to **HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\WindowsUpdate** and set the value to 1.
-and-
- Add a REG_DWORD value called **UseWUServer** to

HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\WindowsUpdate\AU and set the value to 1.

-or-

- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Windows Update > Do not connect to any Windows Update Internet locations.**
- and-
- Apply the Group Policy: **Computer Configuration > Administrative Templates > System > Internet Communication Management > Internet Communication Settings > Turn off access to all Windows Update features.**
- and-
- Apply the Group Policy: **Computer Configuration > Administrative Templates > Windows Components > Windows Update > Specify intranet Microsoft update service location** and set the **Set the alternate download server** to "".

You can turn off automatic updates by doing one of the following. This is not recommended.

- Add a REG_DWORD value called **AutoDownload** to
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\WindowsStore\WindowsUpdate and set the value to 5.
- or-
- For Windows 10 only, apply the Update/AllowAutoUpdate MDM policy from the [Policy CSP](#), where:
 - **0.** Notify the user before downloading the update.
 - **1.** Auto install the update and then notify the user to schedule a device restart.
 - **2** (default). Auto install and restart.
 - **3.** Auto install and restart at a specified time.
 - **4.** Auto install and restart without end-user control.
 - **5.** Turn off automatic updates.

To learn more, see [Device update management](#) and [Configure Automatic Updates by using Group Policy](#).

Manage Wi-Fi Sense in your company

7/28/2017 • 4 min to read • [Edit Online](#)

Applies to:

- Windows 10
- Windows 10 Mobile

Learn more about what features and functionality are supported in each Windows edition at [Compare Windows 10 Editions](#).

Wi-Fi Sense learns about open Wi-Fi hotspots your Windows PC or Windows phone connects to by collecting information about the network, like whether the open Wi-Fi network has a high-quality connection to the Internet. By using that information from your device and from other Wi-Fi Sense customers' devices too, Wi-Fi Sense builds a database of these high-quality networks. When you're in range of one of these Wi-Fi hotspots, you automatically get connected to it.

The initial settings for Wi-Fi Sense are determined by the options you chose when you first set up your PC with Windows 10.

Note

Wi-Fi Sense isn't available in all countries or regions.

How does Wi-Fi Sense work?

Wi-Fi Sense connects your employees to open Wi-Fi networks. Typically, these are the open (no password required) Wi-Fi hotspots you see when you're out and about.

How to manage Wi-Fi Sense in your company

In a company environment, you will most likely deploy Windows 10 to your employees' PCs using your preferred deployment method and then manage their settings globally. With that in mind, you have a few options for managing how your employees will use Wi-Fi Sense.

Important

Turning off Wi-Fi Sense stops employees from connecting automatically to open hotspots.

Using Group Policy (available starting with Windows 10, version 1511)

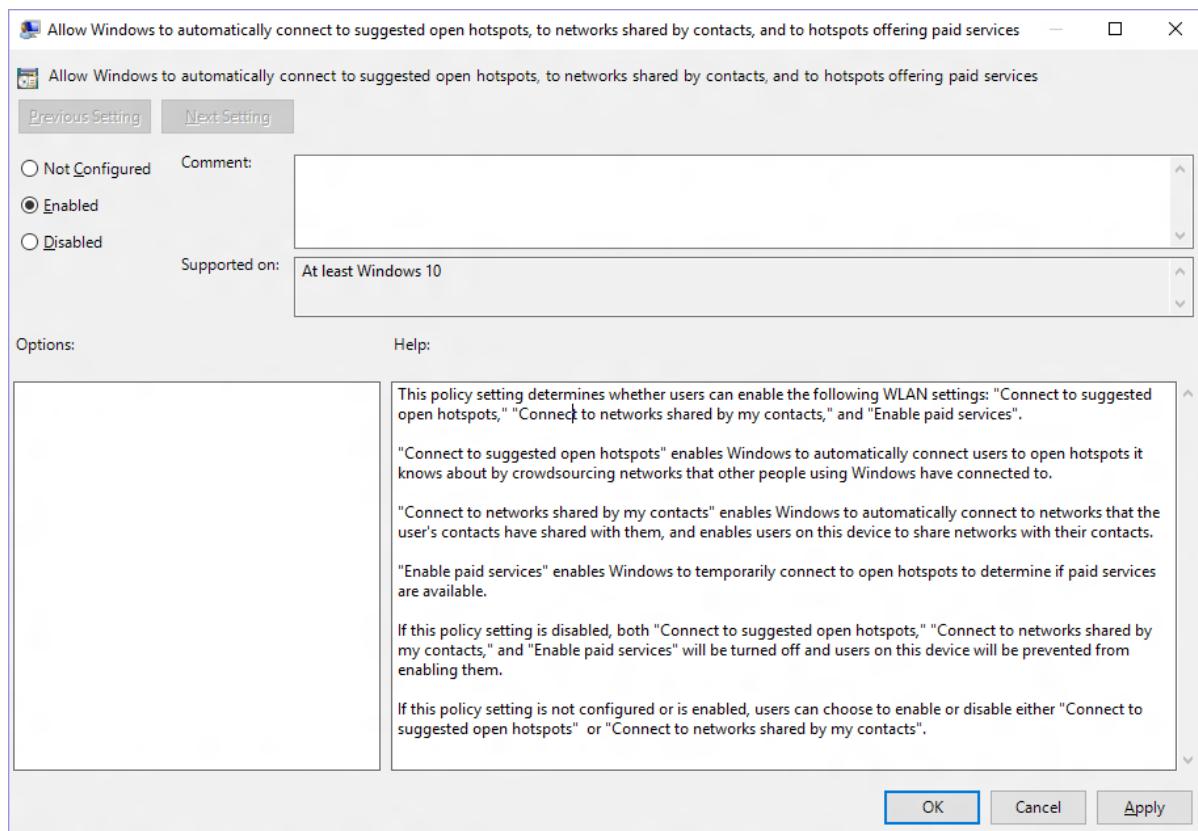
You can manage your Wi-Fi Sense settings by using Group Policy and your Group Policy editor.

To set up Wi-Fi Sense using Group Policy

1. Open your Group Policy editor and go to the

Computer Configuration\Administrative Templates\Network\WLAN Service\WLAN Settings\Allow Windows to automatically connect to suggested open hotspots, to networks shared by contacts, and to hotspots offering paid services

setting.



- Turn Wi-Fi Sense on (enabled) or off (disabled), based on your company's environment.

Using the Registry Editor

You can manage your Wi-Fi Sense settings by using registry keys and the Registry Editor.

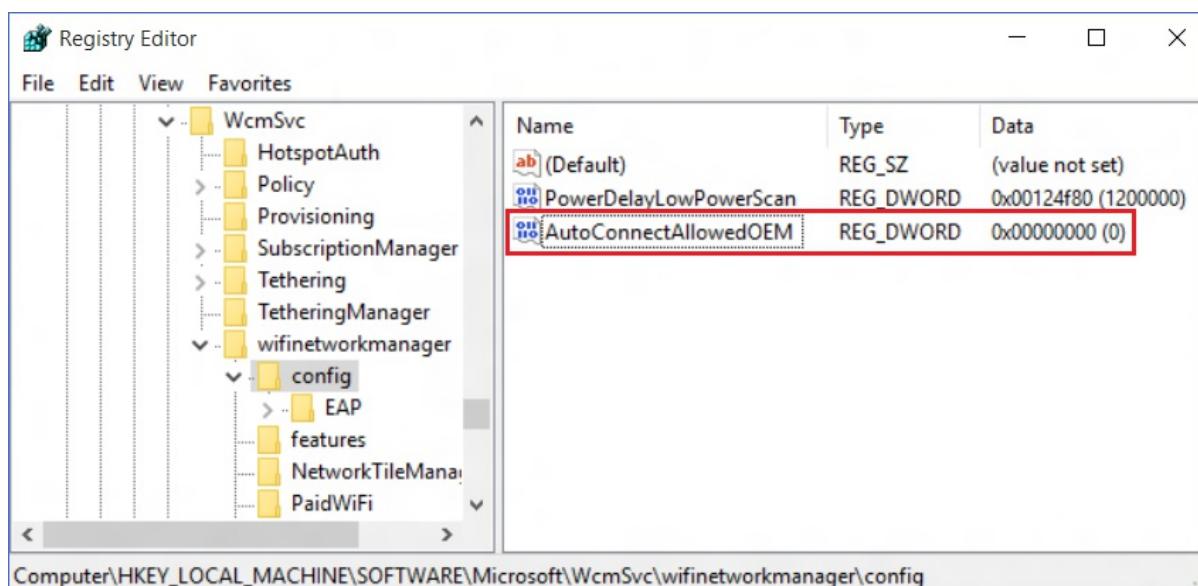
To set up Wi-Fi Sense using the Registry Editor

- Open your Registry Editor and go to

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\WcmSvc\wifinetworkmanager\config\

- Create and set a new **DWORD (32-bit) Value** named, **AutoConnectAllowedOEM**, with a **Value data of 0 (zero)**.

Setting this value to **0** turns off Wi-Fi Sense and all Wi-Fi sense features. When turned off, the Wi-Fi Sense settings still appear on the **Wi-Fi Settings** screen, but can't be controlled by the employee and all of the Wi-Fi Sense features are turned off. For more info, see [How to configure Wi-Fi Sense on Windows 10 in an enterprise](#).



Using the Windows Provisioning settings

You can manage your Wi-Fi Sense settings by changing the Windows provisioning setting, **WiFiSenseAllowed**.

To set up Wi-Fi Sense using WiFiSenseAllowed

- Change the Windows Provisioning setting, **WiFiSenseAllowed**, to **0**.

Setting this value to **0** turns off Wi-Fi Sense and all Wi-Fi sense features. When turned off, the Wi-Fi Sense settings still appear on the **Wi-Fi Settings** screen, but can't be controlled by the employee and all of the Wi-Fi Sense features are turned off. For more info, see the Windows Provisioning settings reference topic, [WiFiSenseAllowed](#).

Using Unattended Windows Setup settings

If your company still uses Unattend, you can manage your Wi-Fi Sense settings by changing the Unattended Windows Setup setting, **WiFiSenseAllowed**.

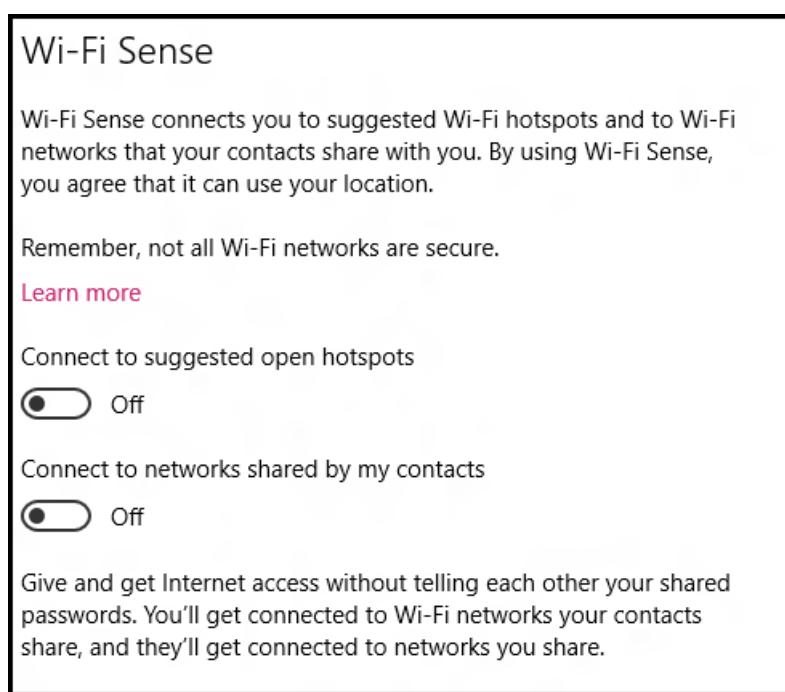
To set up Wi-Fi Sense using WiFiSenseAllowed

- Change the Unattended Windows Setup setting, **WiFiSenseAllowed**, to **0**.

Setting this value to **0** turns off Wi-Fi Sense and all Wi-Fi sense features. When turned off, the Wi-Fi Sense settings still appear on the **Wi-Fi Settings** screen, but can't be controlled by the employee and all of the Wi-Fi Sense features are turned off. For more info, see the Unattended Windows Setup Reference topic, [WiFiSenseAllowed](#).

How employees can change their own Wi-Fi Sense settings

If you don't turn off the ability for your employees to use Wi-Fi Sense, they can turn it on locally by selecting **Settings > Network & Internet > Wi-Fi > Manage Wi-Fi settings**, and then turning on **Connect to suggested open hotspots**.



Important

The service that was used to share networks with Facebook friends, Outlook.com contacts, or Skype contacts is no longer available. This means:

The **Connect to networks shared by my contacts** setting will still appear in **Settings > Network & Internet > Wi-Fi > Manage Wi-Fi settings** on your PC and in **Settings > Network & wireless > Wi-Fi > Wi-Fi Sense** on your phone. However, this setting will have no effect now. Regardless of what it's set to, networks won't be shared with your contacts. Your contacts won't be connected to networks you've shared with them, and you won't be connected to networks they've shared with you.

Even if you selected **Automatically connect to networks shared by your contacts** when you first set up your Windows 10 device, you still won't be connected to networks your contacts have shared with you.

If you select the **Share network with my contacts** check box the first time you connect to a new network, the network won't be shared.

Related topics

- [Wi-Fi Sense and Privacy](#)
- [How to configure Wi-Fi Sense on Windows 10 in an enterprise](#)

Configure kiosk and shared devices running Windows desktop editions

10/17/2017 • 1 min to read • [Edit Online](#)

Some desktop devices in an enterprise serve a special purpose, such as a common PC in a touchdown space that any employee can sign in to, or a PC in the lobby that customers can use to view your product catalog. Windows 10 is easy to configure for shared use or for use as a kiosk (single app).

In this section

TOPIC	DESCRIPTION
Set up a shared or guest PC with Windows 10	Windows 10, version 1607, introduced <i>shared PC mode</i> , which optimizes Windows 10 for shared use scenarios, such as touchdown spaces in an enterprise and temporary customer use in retail.
Set up a kiosk on Windows 10 Pro, Enterprise, or Education	You can configure a device running Windows 10 Pro, Windows 10 Enterprise, or Windows 10 Education as a kiosk device, so that users can only interact with a single application that you select.
Guidelines for choosing an app for assigned access (kiosk mode)	You can choose almost any Windows app for assigned access; however, some apps may not provide a good user experience. This topic provides guidelines to help you choose an appropriate app for a kiosk device.
Create a Windows 10 kiosk that runs multiple apps	Learn how to configure a device running Windows 10 Enterprise or Windows 10 Education so that users can only run a few specific apps. The result is similar to a kiosk device, but with multiple apps available. For example, you might set up a library computer so that users can search the catalog and browse the Internet, but can't run any other apps or change computer settings.

Set up a shared or guest PC with Windows 10

7/28/2017 • 17 min to read • [Edit Online](#)

Applies to

- Windows 10

Windows 10, version 1607, introduced *shared PC mode*, which optimizes Windows 10 for shared use scenarios, such as touchdown spaces in an enterprise and temporary customer use in retail. You can apply shared PC mode to Windows 10 Pro, Pro Education, Education, and Enterprise.

NOTE

If you're interested in using Windows 10 for shared PCs in a school, see [Use Set up School PCs app](#) which provides a simple way to configure PCs with shared PC mode plus additional settings specific for education.

Shared PC mode concepts

A Windows 10 PC in shared PC mode is designed to be management- and maintenance-free with high reliability. In shared PC mode, only one user can be signed in at a time. When the PC is locked, the currently signed in user can always be signed out at the lock screen.

Account models

It is intended that shared PCs are joined to an Active Directory or Azure Active Directory domain by a user with the necessary rights to perform a domain join as part of a setup process. This enables any user that is part of the directory to sign-in to the PC. If using Azure Active Directory Premium, any domain user can also be configured to sign in with administrative rights. Additionally, shared PC mode can be configured to enable a **Guest** option on the sign-in screen, which doesn't require any user credentials or authentication, and creates a new local account each time it is used. Windows 10, version 1703, introduces a **kiosk mode** account. Shared PC mode can be configured to enable a **Kiosk** option on the sign-in screen, which doesn't require any user credentials or authentication, and creates a new local account each time it is used to run a specified app in assigned access (kiosk) mode.

Account management

When the account management service is turned on in shared PC mode, accounts are automatically deleted. Account deletion applies to Active Directory, Azure Active Directory, and local accounts that are created by the **Guest** and **Kiosk** options. Account management is performed both at sign-off time (to make sure there is enough disk space for the next user) as well as during system maintenance time periods. Shared PC mode can be configured to delete accounts immediately at sign-out or when disk space is low. In Windows 10, version 1703, an inactive option is added which deletes accounts if they haven't signed in after a specified number of days.

Maintenance and sleep

Shared PC mode is configured to take advantage of maintenance time periods which run while the PC is not in use. Therefore, sleep is strongly recommended so that the PC can wake up when it is not in use to perform maintenance, clean up accounts, and run Windows Update. The recommended settings can be set by choosing **SetPowerPolicies** in the list of shared PC options. Additionally, on devices without Advanced Configuration and Power Interface (ACPI) wake alarms, shared PC mode will always override real-time clock (RTC) wake alarms to be allowed to wake the PC from sleep (by default, RTC wake alarms are off). This ensures that the widest variety of hardware will take advantage of maintenance periods.

While shared PC mode does not configure Windows Update itself, it is strongly recommended to configure

Windows Update to automatically install updates and reboot (if necessary) during maintenance hours. This will help ensure the PC is always up to date and not interrupting users with updates.

Use one of the following methods to configure Windows Update:

- Group Policy: Set **Computer Configuration > Administrative Templates > Windows Components > Windows Update > Configure Automatic Updates** to and check **Install during automatic maintenance**.
- MDM: Set **Update/AllowAutoUpdate** to .
- Provisioning: In Windows Imaging and Configuration Designer (ICD), set **Policies/Update/AllowAutoUpdate** to .

[Learn more about the AllowAutoUpdate settings](#)

App behavior

Apps can take advantage of shared PC mode with the following three APIs:

- **IsEnabled** - This informs apps when the PC has been configured for shared use scenarios. For example, an app might only download content on demand on a device in shared PC mode, or might skip first run experiences.
- **ShouldAvoidLocalStorage** - This informs apps when the PC has been configured to not allow the user to save to the local storage of the PC. Instead, only cloud save locations should be offered by the app or saved automatically by the app.
- **IsEducationEnvironment** - This informs apps when the PC is used in an education environment. Apps may want to handle telemetry differently or hide advertising functionality.

Customization

Shared PC mode exposes a set of customizations to tailor the behavior to your requirements. These customizations are the options that you'll set either using MDM or a provisioning package as explained in [Configuring shared PC mode on Windows](#). The options are listed in the following table.

SETTING	VALUE
EnableSharedPCMode	Set as True . If this is not set to True , shared PC mode is not turned on and none of the other settings apply. This setting controls this API: IsEnabled Some of the remaining settings in SharedPC are optional, but we strongly recommend that you also set EnableAccountManager to True .
AccountManagement: AccountModel	This option controls how users can sign-in on the PC. Choosing domain-joined will enable any user in the domain to sign-in. Specifying the guest option will add the Guest option to the sign-in screen and enable anonymous guest access to the PC. - Only guest allows anyone to use the PC as a local standard (non-admin) account. - Domain-joined only allows users to sign in with an Active Directory or Azure AD account. - Domain-joined and guest allows users to sign in with an Active Directory, Azure AD, or local standard account.

SETTING	VALUE
AccountManagement: DeletionPolicy	<p>- Delete immediately will delete the account on sign-out.</p> <p>- Delete at disk space threshold will start deleting accounts when available disk space falls below the threshold you set for DiskLevelDeletion, and it will stop deleting accounts when the available disk space reaches the threshold you set for DiskLevelCaching. Accounts are deleted in order of oldest accessed to most recently accessed.</p> <p>Example: The caching number is 50 and the deletion number is 25. Accounts will be cached while the free disk space is above 25%. When the free disk space is less than 25% (the deletion number) at a maintenance period, accounts will be deleted (oldest last used first) until the free disk space is above 50% (the caching number). Accounts will be deleted immediately at sign off of an account if free space is under the deletion threshold and disk space is very low, regardless if the PC is actively in use or not.</p> <p>- Delete at disk space threshold and inactive threshold will apply the same disk space checks as noted above, but also delete accounts if they have not signed in within the number of days specified by InactiveThreshold</p>
AccountManagement: DiskLevelCaching	If you set DeletionPolicy to Delete at disk space threshold , set the percent of total disk space to be used as the disk space threshold for account caching.
AccountManagement: DiskLevelDeletion	If you set DeletionPolicy to Delete at disk space threshold , set the percent of total disk space to be used as the disk space threshold for account deletion.
AccountManagement: InactiveThreshold	If you set DeletionPolicy to Delete at disk space threshold and inactive threshold , set the number of days after which an account that has not signed in will be deleted.
AccountManagement: EnableAccountManager	Set as True to enable automatic account management. If this is not set to true, no automatic account management will be done.
AccountManagement: KioskModeAUMID	Set an Application User Model ID (AUMID) to enable the kiosk account on the sign-in screen. A new account will be created and will use assigned access to only run the app specified by the AUMID. Note that the app must be installed on the PC. Set the name of the account using KioskModeUserTileDisplayText , or a default name will be used. Find the Application User Model ID of an installed app
AccountManagement: KioskModeUserTileDisplayText	Sets the display text on the kiosk account if KioskModeAUMID has been set.
Customization: MaintenanceStartTime	By default, the maintenance start time (which is when automatic maintenance tasks run, such as Windows Update) is midnight. You can adjust the start time in this setting by entering a new start time in minutes from midnight. For example, if you want maintenance to begin at 2 AM, enter 120 as the value.
Customization: MaxPageFileSizeMB	Adjusts the maximum page file size in MB. This can be used to fine-tune page file behavior, especially on low end PCs.

SETTING	VALUE
Customization: RestrictLocalStorage	Set as True to restrict the user from saving or viewing local storage when using File Explorer. This setting controls this API: ShouldAvoidLocalStorage
Customization: SetEduPolicies	Set to True for PCs that will be used in a school. For more information, see Windows 10 configuration recommendations for education customers . This setting controls this API: IsEducationEnvironment
Customization: SetPowerPolicies	When set as True : - Prevents users from changing power settings - Turns off hibernate - Overrides all power state transitions to sleep (e.g. lid close)
Customization: SignInOnResume	This setting specifies if the user is required to sign in with a password when the PC wakes from sleep.
Customization: SleepTimeout	Specifies all timeouts for when the PC should sleep. Enter the amount of idle time in seconds. If you don't set sleep timeout, the default of 1 hour applies.

Configuring shared PC mode on Windows

You can configure Windows to be in shared PC mode in a couple different ways:

- Mobile device management (MDM): Shared PC mode is enabled by the [SharedPC configuration service provider \(CSP\)](#). Your MDM policy can contain any of the options listed in the [Customization](#) section. The following image shows a Microsoft Intune policy with the shared PC options added as OMA-URI settings. [Learn more about Windows 10 policy settings in Microsoft Intune](#).

Edit Policy: SharedPCPolicy

***General**

Configure a policy containing settings for your environment.

*** Name:** SharedPCPolicy

Description:

OMA-URI Settings

Add one or more OMA-URI settings that control functionality on Windows devices.
[More information about custom OMA-URI settings for Windows 10](#)

Setting name	Setting description	OMA-URI
SetEduPolicies		/Vendor/MSFT/SharedPC/SetEduPolicies
SetPowerPolicies		/Vendor/MSFT/SharedPC/SetPowerPolicies
MaintenanceStartTime		/Vendor/MSFT/SharedPC/MaintenanceStartTime
SignInOnResume		/Vendor/MSFT/SharedPC/SignInOnResume
SleepTimeout		/Vendor/MSFT/SharedPC/SleepTimeout
AccountModel		/Vendor/MSFT/SharedPC/AccountModel
DeletionPolicy		/Vendor/MSFT/SharedPC/DeletionPolicy
DiskLevelDeletion		/Vendor/MSFT/SharedPC/DiskLevelDeletion
DiskLevelCaching		/Vendor/MSFT/SharedPC/DiskLevelCaching
EnableAccountManager		/Vendor/MSFT/SharedPC/EnableAccountManager
EnabledSharedPCM		/Vendor/MSFT/SharedPC/EnableSharedPCM

- A provisioning package created with the Windows Configuration Designer: You can apply a provisioning package when you initially set up the PC (also known as the out-of-box-experience or OOBE), or you can apply the provisioning package to a Windows 10 PC that is already in use. The provisioning package is created in Windows Configuration Designer. Shared PC mode is enabled by the [SharedPC configuration service provider \(CSP\)](#), exposed in Windows Configuration Designer as **SharedPC**.

Available customizations

View: All settings

Search

- OOBE
- Policies
- ProvisioningCommands
- SharedPC**
 - AccountManagement
 - EnableSharedPCM
 - PolicyCustomization
- SMISettings
- Start
- TabletMode
- TakeATest
- UnifiedWriteFilter
- UniversalAppInstall
- UniversalAppUninstall
- UsbErrorsOEMOverride
- WeakCharger
- Workplace

SharedPC

EnableSharedPCM	NOT CONFIGURED
-----------------	----------------

SharedPC/AccountManagement

AccountModel	NOT CONFIGURED
DeletionPolicy	NOT CONFIGURED
DiskLevelCaching	Numeric only, Default = 50
DiskLevelDeletion	Numeric only, Default = 25
EnableAccountManager	NOT CONFIGURED

SharedPC/PolicyCustomization

MaintenanceStartTime	Numeric only, Default = 0
SetEduPolicies	NOT CONFIGURED
SetPowerPolicies	NOT CONFIGURED
SignInOnResume	NOT CONFIGURED
SleepTimeout	Numeric only, Default = 3600

- WMI bridge: Environments that use Group Policy can use the [MDM Bridge WMI Provider](#) to configure the [MDM_SharedPC class](#). For example, open PowerShell as an administrator and enter the following:

```
$sharedPC = Get-CimInstance -Namespace "root\cimv2\mdm\dmmap" -ClassName "MDM_SharedPC"
$sharedPC.EnableSharedPCMode = $True
$sharedPC.SetEduPolicies = $True
$sharedPC.SetPowerPolicies = $True
$sharedPC.MaintenanceStartTime = 0
$sharedPC.SignInOnResume = $True
$sharedPC.SleepTimeout = 0
$sharedPC.EnableAccountManager = $True
$sharedPC.AccountModel = 2
$sharedPC.DeletionPolicy = 1
$sharedPC.DiskLevelDeletion = 25
$sharedPC.DiskLevelCaching = 50
$sharedPC.RestrictLocalStorage = $False
$sharedPC.KioskModeAUMID = ""
$sharedPC.KioskModeUserTileDisplayText = ""
$sharedPC.InactiveThreshold = 0
Set-CimInstance -CimInstance $sharedPC
Get-CimInstance -Namespace $namespaceName -ClassName $MDM_SharedPCClass
```

Create a provisioning package for shared use

- [install Windows Configuration Designer](#)
- Open Windows Configuration Designer.
- On the **Start page**, select **Advanced provisioning**.
- Enter a name and (optionally) a description for the project, and click **Next**.
- Select **All Windows desktop editions**, and click **Next**.
- Click **Finish**. Your project opens in Windows Configuration Designer.
- Go to **Runtime settings > SharedPC**. Select the desired settings for shared PC mode.
- On the **File** menu, select **Save**.
- On the **Export** menu, select **Provisioning package**.
- Change **Owner** to **IT Admin**, which will set the precedence of this provisioning package higher than provisioning packages applied to this device from other sources, and then select **Next**.
- Set a value for **Package Version**.

TIP

You can make changes to existing packages and change the version number to update previously applied packages.

- (Optional) In the **Provisioning package security** window, you can choose to encrypt the package and enable package signing.

- Enable package encryption** - If you select this option, an auto-generated password will be shown on the screen.
- Enable package signing** - If you select this option, you must select a valid certificate to use for signing the package. You can specify the certificate by clicking **Select...** and choosing the certificate you want to use to sign the package.

IMPORTANT

We recommend that you include a trusted provisioning certificate in your provisioning package. When the package is applied to a device, the certificate is added to the system store and any package signed with that certificate thereafter can be applied silently.

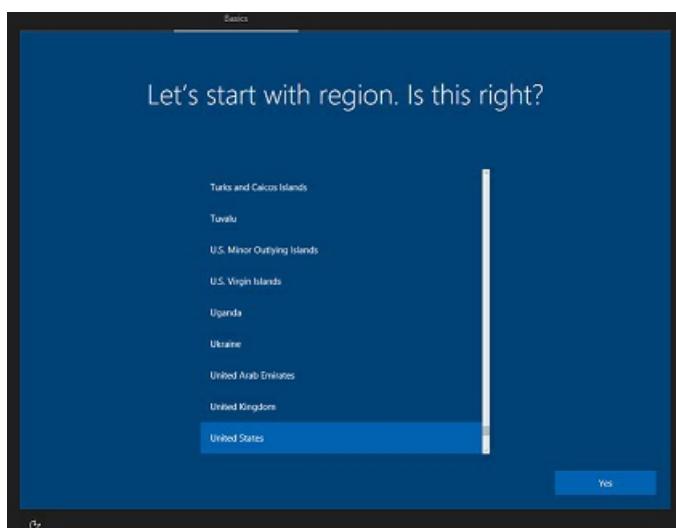
13. Click **Next** to specify the output location where you want the provisioning package to go once it's built. By default, Windows Configuration Designer uses the project folder as the output location. Optionally, you can click **Browse** to change the default output location.
14. Click **Next**.
15. Click **Build** to start building the package. The project information is displayed in the build page and the progress bar indicates the build status. If you need to cancel the build, click **Cancel**. This cancels the current build process, closes the wizard, and takes you back to the **Customizations Page**.
16. If your build fails, an error message will show up that includes a link to the project folder. You can scan the logs to determine what caused the error. Once you fix the issue, try building the package again. If your build is successful, the name of the provisioning package, output directory, and project directory will be shown.
 - If you choose, you can build the provisioning package again and pick a different path for the output package. To do this, click **Back** to change the output package name and path, and then click **Next** to start another build.
 - If you are done, click **Finish** to close the wizard and go back to the **Customizations Page**.
17. Select the **output location** link to go to the location of the package. You can provide that .ppkg to others through any of the following methods:
 - Shared network folder
 - SharePoint site
 - Removable media (USB/SD) (select this option to apply to a PC during initial setup)

Apply the provisioning package

You can apply the provisioning package to a PC during initial setup or to a PC that has already been set up.

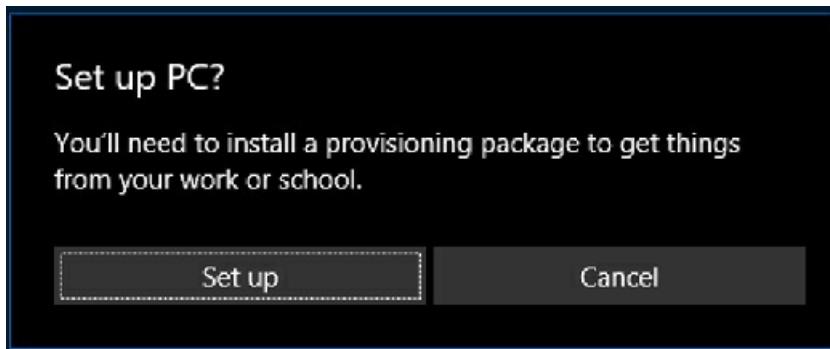
During initial setup

1. Start with a PC on the setup screen.



2. Insert the USB drive. If nothing happens when you insert the USB drive, press the Windows key five times.
 - If there is only one provisioning package on the USB drive, the provisioning package is applied.

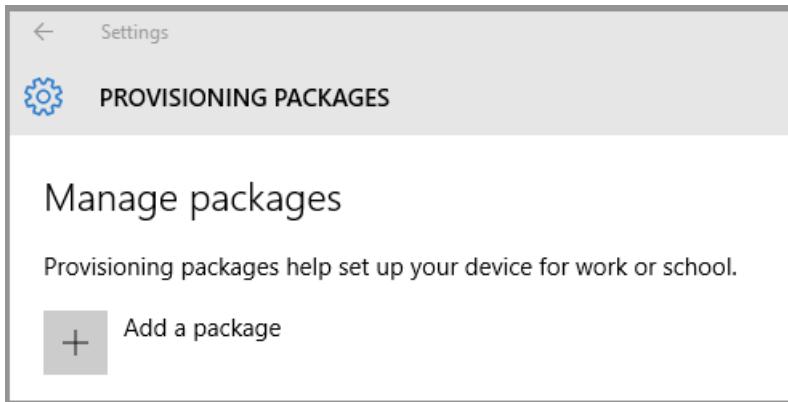
- If there is more than one provisioning package on the USB drive, the **Set up device?** message displays. Click **Set up**, and select the provisioning package that you want to install.



- Complete the setup process.

After setup

On a desktop computer, navigate to **Settings > Accounts > Work access > Add or remove a management package > Add a package**, and selects the package to install.



NOTE

If you apply the setup file to a computer that has already been set up, existing accounts and data might be lost.

Guidance for accounts on shared PCs

- We recommend no local admin accounts on the PC to improve the reliability and security of the PC.
- When a PC is set up in shared PC mode with the default deletion policy, accounts will be cached automatically until disk space is low. Then, accounts will be deleted to reclaim disk space. This account management happens automatically. Both Azure AD and Active Directory domain accounts are managed in this way. Any accounts created through **Guest** and **Kiosk** will also be deleted automatically at sign out.
- On a Windows PC joined to Azure Active Directory:
 - By default, the account that joined the PC to Azure AD will have an admin account on that PC. Global administrators for the Azure AD domain will also have admin accounts on the PC.
 - With Azure AD Premium, you can specify which accounts have admin accounts on a PC using the **Additional administrators on Azure AD Joined devices** setting on the Azure portal.
- Local accounts that already exist on a PC won't be deleted when turning on shared PC mode. New local accounts that are created using **Settings > Accounts > Other people > Add someone else to this PC** after shared PC mode is turned on won't be deleted. However, any new local accounts created by the **Guest** and **Kiosk** options on the sign-in screen (if enabled) will automatically be deleted at sign-out.
- If admin accounts are necessary on the PC
 - Ensure the PC is joined to a domain that enables accounts to be signed on as admin, or

- Create admin accounts before setting up shared PC mode, or
- Create exempt accounts before signing out when turning shared pc mode on.
- The account management service supports accounts that are exempt from deletion.
 - An account can be marked exempt from deletion by adding the account SID to the `HKEY_LOCAL_MACHINE\SOFTARE\Microsoft\Windows\CurrentVersion\SharedPC\Exemptions\` registry key.
 - To add the account SID to the registry key using PowerShell:


```
$adminName = "LocalAdmin" $adminPass = 'Pa$$word123' iex "net user /add $adminName $adminPass" $user = New-Object System.Security.Principal.NTAccount($adminName) $sid = $user.Translate([System.Security.Principal.SecurityIdentifier]) $sid = $sid.Value; New-Item -Path "HKLM:\Software\Microsoft\Windows\CurrentVersion\SharedPC\Exemptions\$sid" -Force
```

Policies set by shared PC mode

Shared PC mode sets local group policies to configure the device. Some of these are configurable using the shared pc mode options.

IMPORTANT

It is not recommended to set additional policies on PCs configured for **Shared PC Mode**. The shared PC mode has been optimized to be fast and reliable over time with minimal to no manual maintenance required.

POLICY NAME	VALUE	WHEN SET?
Admin Templates > Control Panel > Personalization		
Prevent enabling lock screen slide show	Enabled	Always
Prevent changing lock screen and logon image	Enabled	Always
Admin Templates > System > Power Management > Button Settings		
Select the Power button action (plugged in)	Sleep	SetPowerPolicies=True
Select the Power button action (on battery)	Sleep	SetPowerPolicies=True
Select the Sleep button action (plugged in)	Sleep	SetPowerPolicies=True
Select the lid switch action (plugged in)	Sleep	SetPowerPolicies=True
Select the lid switch action (on battery)	Sleep	SetPowerPolicies=True

Admin Templates > System > Power Management > Sleep Settings

Require a password when a computer wakes (plugged in)	Enabled	SignInOnResume=True
Require a password when a computer wakes (on battery)	Enabled	SignInOnResume=True
Specify the system sleep timeout (plugged in)	<i>SleepTimeout</i>	SetPowerPolicies=True
Specify the system sleep timeout (on battery)	<i>SleepTimeout</i>	SetPowerPolicies=True
Turn off hybrid sleep (plugged in)	Enabled	SetPowerPolicies=True
Turn off hybrid sleep (on battery)	Enabled	SetPowerPolicies=True
Specify the unattended sleep timeout (plugged in)	<i>SleepTimeout</i>	SetPowerPolicies=True
Specify the unattended sleep timeout (on battery)	<i>SleepTimeout</i>	SetPowerPolicies=True
Allow standby states (S1-S3) when sleeping (plugged in)	Enabled	SetPowerPolicies=True
Allow standby states (S1-S3) when sleeping (on battery)	Enabled	SetPowerPolicies=True
Specify the system hibernate timeout (plugged in)	Enabled, 0	SetPowerPolicies=True
Specify the system hibernate timeout (on battery)	Enabled, 0	SetPowerPolicies=True

Admin Templates>System>Power Management>Video and Display Settings

Turn off the display (plugged in)	<i>SleepTimeout</i>	SetPowerPolicies=True
Turn off the display (on battery)	<i>SleepTimeout</i>	SetPowerPolicies=True

Admin Templates>System>Power Management>Energy Saver Settings

Energy Saver Battery Threshold (on battery)	70	SetPowerPolicies=True
---	----	-----------------------

Admin Templates>System>Logon

Show first sign-in animation	Disabled	Always
Hide entry points for Fast User Switching	Enabled	Always
Turn on convenience PIN sign-in	Disabled	Always
Turn off picture password sign-in	Enabled	Always
Turn off app notification on the lock screen	Enabled	Always
Allow users to select when a password is required when resuming from connected standby	Disabled	SignInOnResume=True
Block user from showing account details on sign-in	Enabled	Always

Admin Templates>System>User Profiles

Turn off the advertising ID	Enabled	SetEduPolicies=True
-----------------------------	---------	---------------------

Admin Templates>Windows Components

Do not show Windows Tips	Enabled	SetEduPolicies=True
Turn off Microsoft consumer experiences	Enabled	SetEduPolicies=True
Microsoft Passport for Work	Disabled	Always
Prevent the usage of OneDrive for file storage	Enabled	Always

Admin Templates>Windows Components>Biometrics

Allow the use of biometrics	Disabled	Always
Allow users to log on using biometrics	Disabled	Always
Allow domain users to log on using biometrics	Disabled	Always

Admin Templates>Windows Components>Data Collection and Preview Builds

Toggle user control over Insider builds	Disabled	Always
Disable pre-release features or settings	Disabled	Always
Do not show feedback notifications	Enabled	Always
Allow Telemetry	Basic, 0	SetEduPolicies=True

Admin Templates>Windows Components>File Explorer

Show lock in the user tile menu	Disabled	Always
---------------------------------	----------	--------

Admin Templates>Windows Components>Maintenance Scheduler

Automatic Maintenance Activation Boundary	<i>MaintenanceStartTime</i>	Always
Automatic Maintenance Random Delay	Enabled, 2 hours	Always
Automatic Maintenance WakeUp Policy	Enabled	Always

Admin Templates>Windows Components>Windows Hello for Business

Use phone sign-in	Disabled	Always
Use Windows Hello for Business	Disabled	Always
Use biometrics	Disabled	Always

Admin Templates>Windows Components>OneDrive

Prevent the usage of OneDrive for file storage	Enabled	Always
--	---------	--------

Windows Settings>Security Settings>Local Policies>Security Options

Interactive logon: Do not display last user name	Enabled, Disabled when account model is only guest	Always
Interactive logon: Sign-in last interactive user automatically after a system-initiated restart	Disabled	Always
Shutdown: Allow system to be shut down without having to log on	Disabled	Always
User Account Control: Behavior of the elevation prompt for standard users	Auto deny	Always

Set up a kiosk on Windows 10 Pro, Enterprise, or Education

10/17/2017 • 14 min to read • [Edit Online](#)

Applies to

- Windows 10

Looking for Windows Embedded 8.1 Industry information? See [Assigned Access](#)

A single-use or *kiosk* device is easy to set up in Windows 10 for desktop editions.

- Use the [Provision kiosk devices wizard](#) in Windows Configuration Designer (Windows 10, version 1607 or later) to create a provisioning package that configures a kiosk device running either a Universal Windows app or a Classic Windows application (Windows 10 Enterprise or Education only). In Windows 10, version 1709, you can use the [Provision kiosk devices wizard](#) to configure a kiosk device running a Universal Windows app for Windows 10 Pro.

or

- For a kiosk device to run a Universal Windows app, use the [assigned access](#) feature (Windows 10 Pro, Enterprise, or Education).

or

- For a kiosk device to run a Classic Windows application, use [Shell Launcher](#) to set a custom user interface as the shell (Windows 10 Enterprise or Education only).

To return the device to the regular shell, see [Sign out of assigned access](#).

NOTE

A Universal Windows app is built on the Universal Windows Platform (UWP), which was first introduced in Windows 8 as the Windows Runtime. A Classic Windows application uses the Classic Windows Platform (CWP) (e.g., COM, Win32, WPF, WinForms, etc.) and is typically launched using an .EXE or .DLL file.

Set up a kiosk using Windows Configuration Designer

When you use the **Provision kiosk devices** wizard in Windows Configuration Designer, you can configure the kiosk to run either a Universal Windows app or a Classic Windows application.

IMPORTANT

When you build a provisioning package, you may include sensitive information in the project files and in the provisioning package (.ppkg) file. Although you have the option to encrypt the .ppkg file, project files are not encrypted. You should store the project files in a secure location and delete the project files when they are no longer needed.

[Install Windows Configuration Designer](#), then open Windows Configuration Designer and select **Provision kiosk devices**. After you name your project, and click **Next**, configure the settings as shown in the following table.

1

Set up device

Enable device setup if you want to configure settings on this page.

If enabled:

Enter a name for the device.

(Optional) Select a license file to upgrade Windows 10 to a different edition. [See the permitted upgrades.](#)

Toggle **Configure devices for shared use** off. This setting optimizes Windows 10 for shared use scenarios and isn't necessary for a kiosk scenario.

You can also select to remove pre-installed software from the device.

Enabled

Device name

Enter a unique 15-character name for the device. For help generating a unique name, you can use %SERIAL%, which includes a hardware-specific serial number, or you can use %RAND:x%, which generates random characters of x length.

Example device name values:

Contoso-%SERIAL%

Fabrikam-%RAND:5%

|

Required

Enter product key

Optional: Enter a product key to upgrade Windows.

XXXXXX-XXXXXX-XXXXXX-XXXXXX-XXXXXX

Configure devices for shared use

Allow students to quickly login with their credentials or as an anonymous guest, and store all their work in the cloud

No

Remove pre-installed software

Optional: remove pre-installed software without keeping any user data

No

2

Set up network

Enable network setup if you want to configure settings on this page.

If enabled:

Toggle **On** or **Off** for wireless network connectivity. If you select **On**, enter the SSID, the network type (**Open** or **WPA2-Personal**), and (if **WPA2-Personal**) the password for the wireless network.

Enabled

Set up network

Connect devices to a Wi-Fi network

On

Network SSID*

|

Required

Network type*

Open

3

Account Management

Enable account management if you want to configure settings on this page.

If enabled:

You can enroll the device in Active Directory, enroll in Azure Active Directory, or create a local administrator account on the device

To enroll the device in Active Directory, enter the credentials for a least-privileged user account to join the device to the domain.

Before you use a Windows Configuration Designer wizard to configure bulk Azure AD enrollment, [set up Azure AD join in your organization](#). The **maximum number of devices per user** setting in your Azure AD tenant determines how many times the bulk token that you get in the wizard can be used. To enroll the device in Azure AD, select that option and enter a friendly name for the bulk token you will get using the wizard. Set an expiration date for the token (maximum is 30 days from the date you get the token). Click **Get bulk token**. In the **Let's get you signed in** window, enter an account that has permissions to join a device to Azure AD, and then the password. Click **Accept** to give Windows Configuration Designer the necessary permissions.

Warning: You must run Windows Configuration Designer on Windows 10 to configure Azure Active Directory enrollment using any of the wizards.

To create a local administrator account, select that option and enter a user name and password.

Important: If you create a local account in the provisioning package, you must change the password using the **Settings** app every 42 days. If the password is not changed during that period, the account might be locked out and unable to sign in.

4

Add applications

You can provision the kiosk app in the **Add applications** step. You can install multiple applications, both Classic Windows (Win32) apps and Universal Windows Platform (UWP) apps, in a provisioning package. The settings in this step vary according to the application that you select. For help with the settings, see [Provision PCs with apps](#)

Warning: If you click the plus button to add an application, you must specify an application for the provisioning package to validate. If you click the plus button in error, select any executable file in **Installer Path**, and then a **Cancel** button becomes available, allowing you to complete the provisioning package without an application.

Enabled

Manage Organization/School Accounts

Improve security and remote management by enrolling devices into Active Directory

- Enroll into Active Directory
- Enroll in Azure AD
- Local Admin

Create a local administrator account

User name *

Required

Password *

Required

Add an Application

Application name

Specify the name of the application

Required

Installer Path

Specify the installer file

Required

5

Add certificates

To provision the device with a certificate for the kiosk app, click **Add a certificate**. Enter a name for the certificate, and then browse to and select the certificate to be used.

Add a certificate

Certificate name

Specify a name for the certificate

Required

Certificate path

Choose the certificate file

Browse

Required

6

Configure kiosk account and app

Important: You must use the Windows Configuration Designer app from Microsoft Store to select a Classic Windows application as the kiosk app in a provisioning package.

You can create a local standard user account that will be used to run the kiosk app. If you toggle **No**, make sure that you have an existing user account to run the kiosk app.

If you want to create an account, enter the user name and password, and then toggle **Yes** or **No** to automatically sign in the account when the device starts.

In **Configure the kiosk mode app**, enter the name of the user account that will run the kiosk mode app. Select the type of app to run in kiosk mode, and then enter the path or filename (for a Classic Windows app) or the AUMID (for a Universal Windows app). For a Classic Windows app, you can use the filename if the path to the file is in the PATH environment variable, otherwise the full path is required.

Create a kiosk user account

Create a local standard user account to run the kiosk mode app.

 Yes

User name

Required

Password

Required

Auto sign-in

 No

When the device starts, automatically sign in to the kiosk user account. We recommend auto sign-in only when the device is physically secured.

Configure the kiosk mode app

Lock down the device to use the selected app in kiosk mode. The app must be provisioned for the system or installed for the user.

User name

Required

App type



Enter either the full path to the app or a

filename in the Path environment variable

You can include environment variables in the path, such as "c:\program files\internet explorer\iexplore.exe
www.microsoft.com"

7

Configure kiosk common settings

On this step, select your options for tablet mode, the user experience on the Welcome and shutdown screens, and the timeout settings.

Set tablet mode

Display in tablet mode with the On-Screen Keyboard when the kiosk user signs in.

No

Customize user experience

Configure welcome and shutdown screens.

No

Configure power settings

Turn off timeout settings.

No

Finish

You can set a password to protect your provisioning package. You must enter this password when you apply the provisioning package to a device.

Summary

Protect your package

Protect the contents of your package by specifying a password. The password length must be 8-16 characters.

No

NOTE

If you want to use the advanced editor in Windows Configuration Designer, specify the user account and app (by AUMID) in **Runtime settings > AssignedAccess > AssignedAccessSettings**

[Learn how to apply a provisioning package.](#)

Assigned access method for Universal Windows apps

Using assigned access, Windows 10 runs the designated Universal Windows app above the lockscreen, so that the assigned access account has no access to any other functionality on the device. You have these choices for setting up assigned access:

METHOD	ACCOUNT TYPE	WINDOWS 10 EDITION
Use Settings on the PC	Local standard	Pro, Enterprise, Education
Apply a mobile device management (MDM) policy	All (domain, local standard, local administrator, etc)	Pro (1709 only), Enterprise, Education
Create a provisioning package using Windows Configuration Designer	All (domain, local standard, local administrator, etc)	Pro (1709 only), Enterprise, Education
Run a PowerShell script	Local standard	Pro, Enterprise, Education

Requirements

- A domain or local user account.
- A Universal Windows app that is installed or provisioned for that account and is an above lock screen app.

For more information, see [Guidelines for choosing an app for assigned access](#). For details on building an

above lock screen app, see [Kiosk apps for assigned access: Best practices](#).

The app can be your own company app that you have made available in your own app Store. To set up assigned access using MDM or PowerShell, you also need the Application User Model ID (AUMID) for the app. [Learn how to get the AUMID](#).

The Universal Windows app must be able to handle multiple views and cannot launch other apps or dialogs.

NOTE

Assigned access does not work on a device that is connected to more than one monitor.

Set up assigned access in PC settings

1. Go to **Start > Settings > Accounts > Other users**.
2. Choose **Set up assigned access**.
3. Choose an account.
4. Choose an app. Only apps that can run above the lock screen will be displayed. For more information, see [Guidelines for choosing an app for assigned access](#).
5. Close **Settings** – your choices are saved automatically, and will be applied the next time that user account logs on.

To remove assigned access, choose **Turn off assigned access and sign out of the selected account**.

Set up assigned access in MDM

Assigned Access has one setting, KioskModeApp. In the KioskModeApp setting, you enter the user account name and AUMID for the app to run in kiosk mode.

[Learn how to get the AUMID](#).

[See the technical reference for the Assigned Access configuration service provider](#).

Set up assigned access using Windows PowerShell

You can use any of the following PowerShell cmdlets to set up assigned access on multiple devices.

To open PowerShell on Windows 10, search for PowerShell and find **Windows PowerShell Desktop app** in the results. Run PowerShell as administrator.

```
Set-AssignedAccess -AppUserModelId <AUMID> -UserName <username>
```

```
Set-AssignedAccess -AppUserModelId <AUMID> -UserSID <usersid>
```

```
Set-AssignedAccess -AppName <CustomApp> -UserName <username>
```

```
Set-AssignedAccess -AppName <CustomApp> -UserSID <usersid>
```

NOTE

To set up assigned access using `-AppName`, the user account that you specify for assigned access must have logged on at least once.

[Learn how to get the AUMID](#).

[Learn how to get the AppName](#) (see **Parameters**).

[Learn how to get the SID](#).

To remove assigned access, using PowerShell, run the following cmdlet.

```
Clear-AssignedAccess
```

Set up automatic logon

When your kiosk device restarts, whether from an update or power outage, you can log on the assigned access account manually or you can configure the device to log on to the assigned access account automatically. Make sure that Group Policy settings applied to the device do not prevent automatic logon.

Edit the registry to have an account automatically logged on.

1. Open Registry Editor (regedit.exe).

NOTE

If you are not familiar with Registry Editor, [learn how to modify the Windows registry](#).

2. Go to

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\WindowsNT\CurrentVersion\Winlogon

3. Set the values for the following keys.

- *AutoAdminLogon*: set value as **1**.
- *DefaultUserName*: set value as the account that you want logged in.
- *DefaultPassword*: set value as the password for the account.

NOTE

If *DefaultUserName* and *DefaultPassword* aren't there, add them as **New > String Value**.

- *DefaultDomainName*: set value for domain, only for domain accounts. For local accounts, do not add this key.

4. Close Registry Editor. The next time the computer restarts, the account will be logged on automatically.

Sign out of assigned access

To exit the assigned access (kiosk) app, press **Ctrl + Alt + Del**, and then sign in using another account. When you press **Ctrl + Alt + Del** to sign out of assigned access, the kiosk app will exit automatically. If you sign in again as the assigned access account or wait for the login screen timeout, the kiosk app will be re-launched. The assigned access user will remain signed in until an admin account opens **Task Manager > Users** and signs out the user account.

If you press **Ctrl + Alt + Del** and do not sign in to another account, after a set time, assigned access will resume. The default time is 30 seconds, but you can change that in the following registry key:

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Authentication\LogonUI

To change the default time for assigned access to resume, add *IdleTimeOut* (DWORD) and enter the value data as milliseconds in hexadecimal.

Shell Launcher for Classic Windows applications

Using Shell Launcher, you can configure a kiosk device that runs a Classic Windows application as the user interface. The application that you specify replaces the default shell (explorer.exe) that usually runs when a user logs on.

NOTE

You can also configure a kiosk device that runs a Classic Windows application by using the [Provision kiosk devices wizard](#).

WARNING

Shell Launcher doesn't support a custom shell with an application that launches a different process and exits. For example, you cannot specify **write.exe** in Shell Launcher. Shell Launcher launches a custom shell and monitors the process to identify when the custom shell exits. **Write.exe** creates a 32-bit wordpad.exe process and exits. Because Shell Launcher is not aware of the newly created wordpad.exe process, Shell Launcher will take action based on the exit code of **Write.exe**, such as restarting the custom shell.

Requirements

- A domain or local user account.
- A Classic Windows application that is installed for that account. The app can be your own company application or a common app like Internet Explorer.

See the [technical reference for the shell launcher component](#).

Configure Shell Launcher

To set a Classic Windows application as the shell, you first turn on the Shell Launcher feature, and then you can set your custom shell as the default using PowerShell.

To turn on Shell Launcher in Windows features

1. Go to Control Panel > **Programs and features** > **Turn Windows features on or off**.
2. Expand **Device Lockdown**.
3. Select **Shell Launcher** and **OK**.

Alternatively, you can turn on Shell Launcher using Windows Configuration Designer in a provisioning package, using **SMISettings > ShellLauncher**, or the Deployment Image Servicing and Management (DISM.exe) tool.

To turn on Shell Launcher using DISM

1. Open a command prompt as an administrator.
2. Enter the following command.

```
Dism /online /Enable-Feature /all /FeatureName:Client-EmbeddedShellLauncher
```

To set your custom shell

Modify the following PowerShell script as appropriate. The comments in the sample script explain the purpose of each section and tell you where you will want to change the script for your purposes. Save your script with the extension .ps1, open Windows PowerShell as administrator, and run the script on the kiosk device.

```
# Check if shell launcher license is enabled
function Check-ShellLauncherLicenseEnabled
{
    [string]$source = @"
using System;
using System.Runtime.InteropServices;

static class CheckShellLauncherLicense
{
    const int S_OK = 0;

    public static bool IsShellLauncherLicenseEnabled()
    {
        int enabled = 0;

        if (NativeMethods.SLGetWindowsInformationDWORD("EmbeddedFeature-ShellLauncher-Enabled", out enabled) != S_OK)
        {
            enabled = 0;
        }

        return (enabled != 0);
    }

    static class NativeMethods
    {
        [DllImport("Slc.dll")]
        internal static extern int SLGetWindowsInformationDWORD([MarshalAs(UnmanagedType.LPWSTR)]string valueName, out int value);
    }
}

{@
    $type = Add-Type -TypeDefintion $source -PassThru

    return $type[0]::IsShellLauncherLicenseEnabled()
}

[bool]$result = $false

$result = Check-ShellLauncherLicenseEnabled
"`nShell Launcher license enabled is set to " + $result
if (-not($result))
{
    "`nThis device doesn't have required license to use Shell Launcher"
    exit
}

$COMPUTER = "localhost"
$NAMESPACE = "root\standardcimv2\embedded"

# Create a handle to the class instance so we can call the static methods.
try {
    $ShellLauncherClass = [wmiclass]"\"$COMPUTER\${$NAMESPACE}:WESL_UserSetting"
} catch [Exception] {
    write-host $_.Exception.Message;
    write-host "Make sure Shell Launcher feature is enabled"
    exit
}
```

```

# This well-known security identifier (SID) corresponds to the BUILTIN\Administrators group.

$Admins_SID = "S-1-5-32-544"

# Create a function to retrieve the SID for a user account on a machine.

function Get-UsernameSID($AccountName) {

    $NTUserObject = New-Object System.Security.Principal.NTAccount($AccountName)
    $NTUserSID = $NTUserObject.Translate([System.Security.Principal.SecurityIdentifier])

    return $NTUserSID.Value
}

# Get the SID for a user account named "Cashier". Rename "Cashier" to an existing account on your system to
# test this script.

$Cashier_SID = Get-UsernameSID("Cashier")

# Define actions to take when the shell program exits.

$restart_shell = 0
$restart_device = 1
$shutdown_device = 2

# Examples. You can change these examples to use the program that you want to use as the shell.

# This example sets the command prompt as the default shell, and restarts the device if the command prompt is
# closed.

$ShellLauncherClass.SetDefaultShell("cmd.exe", $restart_device)

# Display the default shell to verify that it was added correctly.

$DefaultShellObject = $ShellLauncherClass.GetDefaultShell()

```
nDefault Shell is set to " + $DefaultShellObject.Shell + " and the default action is set to " +
$DefaultShellObject.defaultaction

Set Internet Explorer as the shell for "Cashier", and restart the machine if Internet Explorer is closed.

$ShellLauncherClass.SetCustomShell($Cashier_SID, "c:\program files\internet explorer\iexplore.exe
www.microsoft.com", ($null), ($null), $restart_shell)

Set Explorer as the shell for administrators.

$ShellLauncherClass.SetCustomShell($Admins_SID, "explorer.exe")

View all the custom shells defined.

```
nCurrent settings for custom shells:"`n
Get-WmiObject -namespace $NAMESPACE -computer $COMPUTER -class WESL_UserSetting | Select Sid, Shell,
DefaultAction

# Enable Shell Launcher

$ShellLauncherClass.SetEnabled($TRUE)

$IsShellLauncherEnabled = $ShellLauncherClass.IsEnabled()

```
nEnabled is set to " + $IsShellLauncherEnabled.Enabled

Remove the new custom shells.

$ShellLauncherClass.RemoveCustomShell($Admins_SID)

$ShellLauncherClass.RemoveCustomShell($Cashier_SID)

```

```
Disable Shell Launcher

$ShellLauncherClass.SetEnabled($FALSE)

$IsShellLauncherEnabled = $ShellLauncherClass.IsEnabled()

"`nEnabled is set to " + $IsShellLauncherEnabled.Enabled
```

## Other settings to lock down

For a more secure kiosk experience, we recommend that you make the following configuration changes to the device:

- Put device in **Tablet mode**.

If you want users to be able to use the touch (on screen) keyboard, go to **Settings > System > Tablet mode** and choose **On**.

- Hide **Ease of access** feature on the logon screen.

Go to **Control Panel > Ease of Access > Ease of Access Center**, and turn off all accessibility tools.

- Disable the hardware power button.

Go to **Power Options > Choose what the power button does**, change the setting to **Do nothing**, and then **Save changes**.

- Remove the power button from the sign-in screen.

Go to **Computer Configuration > Windows Settings > Security Settings > Local Policies > Security Options > Shutdown: Allow system to be shut down without having to log on** and select **Disabled**.

- Disable the camera.

Go to **Settings > Privacy > Camera**, and turn off **Let apps use my camera**.

- Turn off app notifications on the lock screen.

Go to **Group Policy Editor > Computer Configuration > Administrative Templates\System\Logon\Turn off app notifications on the lock screen**.

- Disable removable media.

Go to **Group Policy Editor > Computer Configuration > Administrative Templates\System\Device Installation\Device Installation Restrictions**. Review the policy settings available in **Device Installation Restrictions** for the settings applicable to your situation.

### NOTE

To prevent this policy from affecting a member of the Administrators group, in **Device Installation Restrictions**, enable **Allow administrators to override Device Installation Restriction policies**.

## Related topics

- [Set up a kiosk on Windows 10 Mobile or Windows 10 Mobile Enterprise](#)

Not finding content you need? Windows 10 users, tell us what you want on [Feedback Hub](#).

# Guidelines for choosing an app for assigned access (kiosk mode)

10/20/2017 • 3 min to read • [Edit Online](#)

## Applies to

- Windows 10

You can use assigned access to restrict customers at your business to using only one Windows app so your device acts like a kiosk. Administrators can use assigned access to restrict a selected user account to access a single Windows app. You can choose almost any Windows app for assigned access; however, some apps may not provide a good user experience.

The following guidelines may help you choose an appropriate Windows app for your assigned access experience.

## General guidelines

- Windows apps must be provisioned or installed for the assigned access account before they can be selected as the assigned access app. [Learn how to provision and install apps](#).
- Updating a Windows app can sometimes change the Application User Model ID (AUMID) of the app. If this happens, you must update the assigned access settings to launch the updated app, because assigned access uses the AUMID to determine which app to launch.
- Apps that are generated using the [Desktop App Converter \(Desktop Bridge\)](#) cannot be used as kiosk apps.

## Guidelines for Windows apps that launch other apps

Some Windows apps can launch other apps. Assigned access prevents Windows apps from launching other apps.

Avoid selecting Windows apps that are designed to launch other apps as part of their core functionality.

## Guidelines for web browsers

Microsoft Edge and any third-party web browsers that can be set as a default browser have special permissions beyond that of most Windows apps. Microsoft Edge is not supported for assigned access.

If you use a web browser as your assigned access app, consider the following tips:

- You can download browsers that are optimized to be used as a kiosk from the Microsoft Store.
- You can create your own web browser Windows app by using the WebView class. Learn more about developing your own web browser app:
  - [Creating your own browser with HTML and JavaScript](#)
  - [WebView class](#)
  - [A web browser built with JavaScript as a Windows app](#)

### To block access to the file system from Internet Explorer's web address bar

- On the Start screen, type the following: `gpedit.msc`
- Press **Enter** or click the gpedit icon to launch the group policy editor.
- In the group policy editor, navigate to **User Configuration > Administrative Templates > Start Menu and Taskbar**.

4. Select **Remove Run menu from Start Menu**, select **Disabled**, and click **Apply**. Disabling this policy prevents users from entering the following into the Internet Explorer Address Bar:

- A UNC path (\server\share)
- A local drive (C:\)
- A local folder (\temp)

## Secure your information

Avoid selecting Windows apps that may expose the information you don't want to show in your kiosk, since kiosk usually means anonymous access and locates in a public setting like a shopping mall. For example, an app that has a file picker allows the user to gain access to files and folders on the user's system, avoid selecting this type of apps if they provide unnecessary data access.

## App configuration

Some apps may require additional configurations before they can be used appropriately in assigned access . For example, Microsoft OneNote requires you to set up a Microsoft account for the assigned access user account before OneNote will open in assigned access. Check the guidelines published by your selected app and do the setup accordingly.

## Develop your kiosk app

Assigned access in Windows 10 leverages the new lock framework. When an assigned access user signs in, the selected kiosk app is launched above lock . The kiosk app is actually running as an above lock screen app.

Follow the [best practices guidance for developing a kiosk app for assigned access](#).

## Test your assigned access experience

The above guidelines may help you select or develop an appropriate Windows app for your assigned access experience. Once you have selected your app, we recommend that you thoroughly test the assigned access experience to ensure that your device provides a good customer experience.

## Learn more

[Customizing Your Device Experience with Assigned Access](#)

# Create a Windows 10 kiosk that runs multiple apps

10/30/2017 • 22 min to read • [Edit Online](#)

## Applies to

- Windows 10

A [kiosk device](#) typically runs a single app, and users are prevented from accessing any features or functions on the device outside of the kiosk app. In Windows 10, version 1709, the [AssignedAccess configuration service provider \(CSP\)](#) has been expanded to make it easy for administrators to create kiosks that run more than one app. You can configure multi-app kiosks using a provisioning package.

### NOTE

For devices running versions of Windows 10 earlier than version 1709, you can [create AppLocker rules](#) to configure a multi-app kiosk.

The benefit of a multi-app kiosk, or fixed-purpose device, is to provide an easy-to-understand experience for individuals by putting in front of them only the things they need to use, and removing from their view the things they don't need to access.

### WARNING

The assigned access feature is intended for corporate-owned fixed-purpose devices, like kiosks. When the multi-app assigned access configuration is applied on the device, certain policies are enforced system-wide, and will impact other users on the device. Deleting the multi-app configuration will remove the assigned access lockdown profiles associated with the users, but it cannot revert all the enforced policies (such as Start layout). A factory reset is needed to clear all the policies enforced via assigned access.

Process:

1. [Create XML file](#)
2. [Add XML file to provisioning package](#)
3. [Apply provisioning package to device](#)

If you don't want to use a provisioning package, you can deploy the configuration XML file using [mobile device management \(MDM\)](#) or you can configure assigned access using the [MDM Bridge WMI Provider](#).

## Prerequisites

- Windows Configuration Designer (Windows 10, version 1709)
- The kiosk device must be running Windows 10 (S, Pro, Enterprise, or Education), version 1709

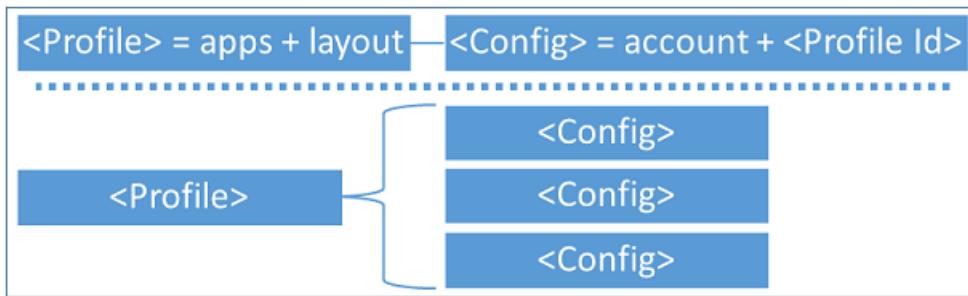
## Create XML file

Let's start by looking at the basic structure of the XML file.

- A configuration xml can define multiple *profiles*. Each profile has a unique **Id** and defines a set of applications that are allowed to run, whether the taskbar is visible, and can include a custom Start layout.
- A configuration xml can have multiple *config* sections. Each config section associates a non-admin user

account to a default profile **Id**.

- Multiple config sections can be associated to the same profile.
- A profile has no effect if it's not associated to a config section.



You can start your file by pasting the following XML (or any other examples in this topic) into a XML editor, and saving the file as *filename.xml*. Each section of this XML is explained in this topic.

```
<?xml version="1.0" encoding="utf-8" ?>
<AssignedAccessConfiguration xmlns="http://schemas.microsoft.com/AssignedAccess/2017/config">
 <Profiles>
 <Profile Id="">
 <AllAppsList>
 <AllowedApps/>
 </AllAppsList>
 <StartLayout/>
 <Taskbar/>
 </Profile>
 </Profiles>
 <Configs>
 <Config>
 <Account/>
 <DefaultProfile Id="" />
 </Config>
 </Configs>
</AssignedAccessConfiguration>
```

## Profile

A profile section in the XML has the following entries:

- **Id**
- **AllowedApps**
- **StartLayout**
- **Taskbar**

### Id

The profile **Id** is a GUID attribute to uniquely identify the profile. You can create a GUID using a GUID generator. The GUID just needs to be unique within this XML file.

```
<Profiles>
 <Profile Id="{9A2A490F-10F6-4764-974A-43B19E722C23}">...</Profile>
</Profiles>
```

### AllowedApps

**AllowedApps** is a list of applications that are allowed to run. Apps can be Universal Windows Platform (UWP) apps or Classic Windows desktop apps.

Based on the purpose of the kiosk device, define the list of applications that are allowed to run. This list can contain both UWP apps and desktop apps. When the multi-app kiosk configuration is applied to a device, AppLocker rules will be generated to allow the apps that are listed in the configuration.

#### NOTE

You cannot manage AppLocker rules that are generated by the multi-app kiosk configuration in [MMC snap-ins](#). Avoid applying AppLocker rules to devices running the multi-app kiosk configuration.

- For UWP apps, you need to provide the App User Model ID (AUMID). [Learn how to get the AUMID](#), or [get the AUMID from the Start Layout XML](#).
- For desktop apps, you need to specify the full path of the executable, which can contain one or more system environment variables in the form of %variableName% (i.e. %systemroot%, %windir%).

Here are the predefined assigned access AppLocker rules for **UWP apps**:

1. Default rule is to allow all users to launch the signed package apps.
2. The package app deny list is generated at runtime when the assigned access user signs in. Based on the installed/provisioned package apps available for the user account, assigned access generates the deny list. This list will exclude the default allowed inbox package apps which are critical for the system to function, and then exclude the allowed packages that enterprises defined in the assigned access configuration. If there are multiple apps within the same package, all these apps will be excluded. This deny list will be used to prevent the user from accessing the apps which are currently available for the user but not in the allowed list.

#### NOTE

Multi-app kiosk mode doesn't block the enterprise or the users from installing UWP apps. When a new UWP app is installed during the current assigned access user session, this app will not be in the deny list. When the user signs out and signs in again, the app will be included in the deny list. If this is an enterprise-deployed line-of-business app and you want to allow it to run, update the assigned access configuration to include it in the allowed app list.

Here are the predefined assigned access AppLocker rules for **desktop apps**:

1. Default rule is to allow all users to launch the desktop programs signed with Microsoft Certificate in order for the system to boot and function. The rule also allows the admin user group to launch all desktop programs.
2. There is a predefined inbox desktop app deny list for the assigned access user account, and this deny list is adjusted based on the desktop app allow list that you defined in the multi-app configuration.
3. Enterprise-defined allowed desktop apps are added in the AppLocker allow list.

The following example allows Groove Music, Movies & TV, Photos, Weather, Calculator, Paint, and Notepad apps to run on the device.

```
<AllAppsList>
 <AllowedApps>
 <App AppUserModelId="Microsoft.ZuneMusic_8wekyb3d8bbwe!Microsoft.ZuneMusic" />
 <App AppUserModelId="Microsoft.ZuneVideo_8wekyb3d8bbwe!Microsoft.ZuneVideo" />
 <App AppUserModelId="Microsoft.Windows.Photos_8wekyb3d8bbwe!App" />
 <App AppUserModelId="Microsoft.BingWeather_8wekyb3d8bbwe!App" />
 <App AppUserModelId="Microsoft.WindowsCalculator_8wekyb3d8bbwe!App" />
 <App DesktopAppPath="%windir%\system32\mspaint.exe" />
 <App DesktopAppPath="C:\Windows\System32\notepad.exe" />
 </AllowedApps>
</AllAppsList>
```

## StartLayout

After you define the list of allowed applications, you can customize the Start layout for your kiosk experience. You can choose to pin all the allowed apps on the Start screen or just a subset, depending on whether you want the end user to directly access them on the Start screen.

The easiest way to create a customized Start layout to apply to other Windows 10 devices is to set up the Start screen on a test device and then export the layout. For detailed steps, see [Customize and export Start layout](#).

A few things to note here:

- The test device on which you customize the Start layout should have the same OS version that is installed on the device where you plan to deploy the multi-app assigned access configuration.
- Since the multi-app assigned access experience is intended for fixed-purpose devices, to ensure the device experiences are consistent and predictable, use the *full* Start layout option instead of the *partial* Start layout.
- There are no apps pinned on the taskbar in the multi-app mode, and it is not supported to configure Taskbar layout using the `<CustomTaskbarLayoutCollection>` tag in a layout modification XML as part of the assigned access configuration.
- The following example uses DesktopApplicationLinkPath to pin the desktop app to start. When the desktop app doesn't have a shortcut link on the target device, [learn how to provision .lnk files using Windows Configuration Designer](#).

This example pins Groove Music, Movies & TV, Photos, Weather, Calculator, Paint, and Notepad apps on Start.

```
<StartLayout>
 <![CDATA[<LayoutModificationTemplate
 xmlns:defaultlayout="http://schemas.microsoft.com/Start/2014/FullDefaultLayout"
 xmlns:start="http://schemas.microsoft.com/Start/2014/StartLayout" Version="1"
 xmlns="http://schemas.microsoft.com/Start/2014/LayoutModification">
 <LayoutOptions StartTileGroupCellWidth="6" />
 <DefaultLayoutOverride>
 <StartLayoutCollection>
 <defaultlayout:StartLayout GroupCellWidth="6">
 <start:Group Name="Group1">
 <start:Tile Size="4x4" Column="0" Row="0"
 AppUserModelID="Microsoft.ZuneMusic_8wekyb3d8bbwe!Microsoft.ZuneMusic" />
 <start:Tile Size="2x2" Column="4" Row="2"
 AppUserModelID="Microsoft.ZuneVideo_8wekyb3d8bbwe!Microsoft.ZuneVideo" />
 <start:Tile Size="2x2" Column="4" Row="0"
 AppUserModelID="Microsoft.Windows.Photos_8wekyb3d8bbwe!App" />
 <start:Tile Size="2x2" Column="4" Row="4"
 AppUserModelID="Microsoft.BingWeather_8wekyb3d8bbwe!App" />
 <start:Tile Size="4x2" Column="0" Row="4"
 AppUserModelID="Microsoft.WindowsCalculator_8wekyb3d8bbwe!App" />
 </start:Group>
 <start:Group Name="Group2">
 <start:DesktopApplicationTile Size="2x2" Column="2" Row="0"
 DesktopApplicationLinkPath="%ALLUSERSPROFILE%\Microsoft\Windows\Start Menu\Programs\Accessories\Paint.lnk" />
 <start:DesktopApplicationTile Size="2x2" Column="0" Row="0"
 DesktopApplicationLinkPath="%APPDATA%\Microsoft\Windows\Start Menu\Programs\Accessories\Notepad.lnk" />
 </start:Group>
 </defaultlayout:StartLayout>
 </StartLayoutCollection>
 </DefaultLayoutOverride>
 </LayoutModificationTemplate>
]]>
</StartLayout>
```

### NOTE

If an app is not installed for the user but is included in the Start layout XML, the app will not be shown on the Start screen.



### Taskbar

Define whether you want to have the taskbar present in the kiosk device. For tablet-based or touch-enabled all-in-one kiosks, when you don't attach a keyboard and mouse, you can hide the taskbar as part of the multi-app experience if you want.

The following example exposes the taskbar to the end user:

```
<Taskbar ShowTaskbar="true"/>
```

The following example hides the taskbar:

```
<Taskbar ShowTaskbar="false"/>
```

#### NOTE

This is different from the **Automatically hide the taskbar** option in tablet mode, which shows the taskbar when swiping up from or moving the mouse pointer down to the bottom of the screen. Setting **ShowTaskbar** as **false** will always keep the taskbar hidden.

### Configs

Under **Configs**, define which user account will be associated with the profile. When this user account signs in on the device, the associated assigned access profile will be enforced, including the allowed apps, Start layout, and taskbar configuration, as well as other local group policies or mobile device management (MDM) policies set as part of the multi-app experience.

The full multi-app assigned access experience can only work for non-admin users. It's not supported to associate an admin user with the assigned access profile; doing this in the XML file will result in unexpected/unsupported experiences when this admin user signs in.

The account can be local, domain, or Azure Active Directory (Azure AD). Groups are not supported.

- Local account can be entered as `machinename\account` or `.\account` or just `account`.
- Domain account should be entered as `domain\account`.
- Azure AD account must be specified in this format: `AzureAD\{email address}`. **AzureAD** must be provided AS IS (consider it's a fixed domain name), then follow with the Azure AD email address, e.g.  
`AzureAD\someone@contoso.onmicrosoft.com`.

### WARNING

Assigned access can be configured via WMI or CSP to run its applications under a domain user or service account, rather than a local account. However, use of domain user or service accounts introduces risks that an attacker subverting the assigned access application might gain access to sensitive domain resources that have been inadvertently left accessible to any domain account. We recommend that customers proceed with caution when using domain accounts with assigned access, and consider the domain resources potentially exposed by the decision to do so.

Before applying the multi-app configuration, make sure the specified user account is available on the device, otherwise it will fail.

### NOTE

For both domain and Azure AD accounts, it's not required that target account is explicitly added to the device. As long as the device is AD-joined or Azure AD-joined, the account can be discovered in the domain forest or tenant that the device is joined to. For local accounts, it is required that the account exist before you configure the account for assigned access.

```
<Configs>
 <Config>
 <Account>MultiAppKioskUser</Account>
 <DefaultProfile Id="{9A2A490F-10F6-4764-974A-43B19E722C23}" />
 </Config>
</Configs>
```

## Add XML file to provisioning package

Before you add the XML file to a provisioning package, you can [validate your configuration XML against the XSD](#).

Use the Windows Configuration Designer tool to create a provisioning package. [Learn how to install Windows Configuration Designer](#).

### IMPORTANT

When you build a provisioning package, you may include sensitive information in the project files and in the provisioning package (.ppkg) file. Although you have the option to encrypt the .ppkg file, project files are not encrypted. You should store the project files in a secure location and delete the project files when they are no longer needed.

1. Open Windows Configuration Designer (by default, %systemdrive%\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Imaging and Configuration Designer\x86\ICD.exe).
2. Choose **Advanced provisioning**.
3. Name your project, and click **Next**.
4. Choose **All Windows desktop editions** and click **Next**.
5. On **New project**, click **Finish**. The workspace for your package opens.
6. Expand **Runtime settings > AssignedAccess > MultiAppAssignedAccessSettings**.
7. In the center pane, click **Browse** to locate and select the assigned access configuration XML file that you created.

## AssignedAccess/MultiAppAssignedAccessSettings

MultiAppAssignedAccessSettings

|

Browse...

8. **(Optional):** If you want to apply the provisioning package after device initial setup and there is an admin user already available on the kiosk device, skip this step.) Create an admin user account in **Runtime settings > Accounts > Users**. Provide a **UserName** and **Password**, and select **UserGroup** as **Administrators**. With this account, you can view the provisioning status and logs if needed.
9. **(Optional):** If you already have a non-admin account on the kiosk device, skip this step.) Create a local standard user account in **Runtime settings > Accounts > Users**. Make sure the **UserName** is the same as the account that you specify in the configuration XML. Select **UserGroup** as **Standard Users**.
10. On the **File** menu, select **Save**.
11. On the **Export** menu, select **Provisioning package**.
12. Change **Owner** to **IT Admin**, which will set the precedence of this provisioning package higher than provisioning packages applied to this device from other sources, and then select **Next**.
13. Optional. In the **Provisioning package security** window, you can choose to encrypt the package and enable package signing.
  - **Enable package encryption** - If you select this option, an auto-generated password will be shown on the screen.
  - **Enable package signing** - If you select this option, you must select a valid certificate to use for signing the package. You can specify the certificate by clicking **Browse** and choosing the certificate you want to use to sign the package.
14. Click **Next** to specify the output location where you want the provisioning package to go when it's built. By default, Windows Imaging and Configuration Designer (ICD) uses the project folder as the output location.  
Optionally, you can click **Browse** to change the default output location.
15. Click **Next**.
16. Click **Build** to start building the package. The provisioning package doesn't take long to build. The project information is displayed in the build page and the progress bar indicates the build status.  
If you need to cancel the build, click **Cancel**. This cancels the current build process, closes the wizard, and takes you back to the **Customizations Page**.
17. If your build fails, an error message will show up that includes a link to the project folder. You can scan the logs to determine what caused the error. Once you fix the issue, try building the package again.  
If your build is successful, the name of the provisioning package, output directory, and project directory will be shown.
  - If you choose, you can build the provisioning package again and pick a different path for the output package. To do this, click **Back** to change the output package name and path, and then click **Next** to start another build.
  - If you are done, click **Finish** to close the wizard and go back to the **Customizations Page**.
18. Copy the provisioning package to the root directory of a USB drive.

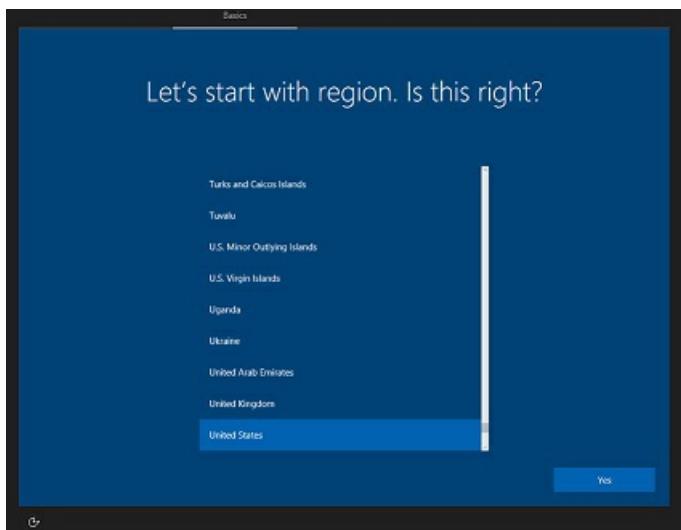
## Apply provisioning package to device

Provisioning packages can be applied to a device during the first-run experience (out-of-box experience or

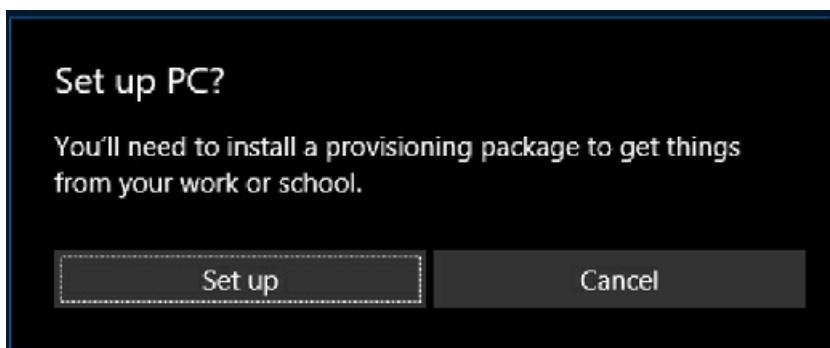
"OOBE") and after ("runtime").

#### During initial setup, from a USB drive

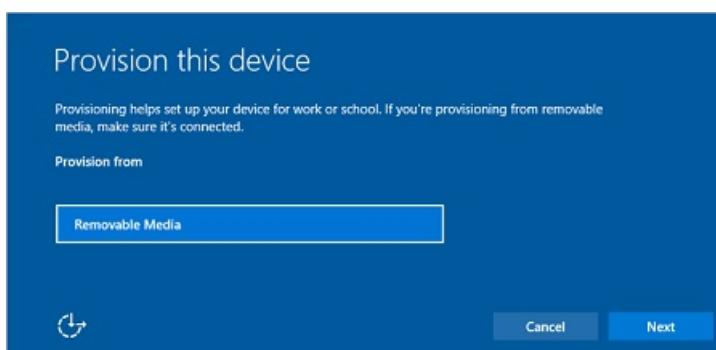
1. Start with a computer on the first-run setup screen. If the PC has gone past this screen, reset the PC to start over. To reset the PC, go to **Settings > Update & security > Recovery > Reset this PC**.



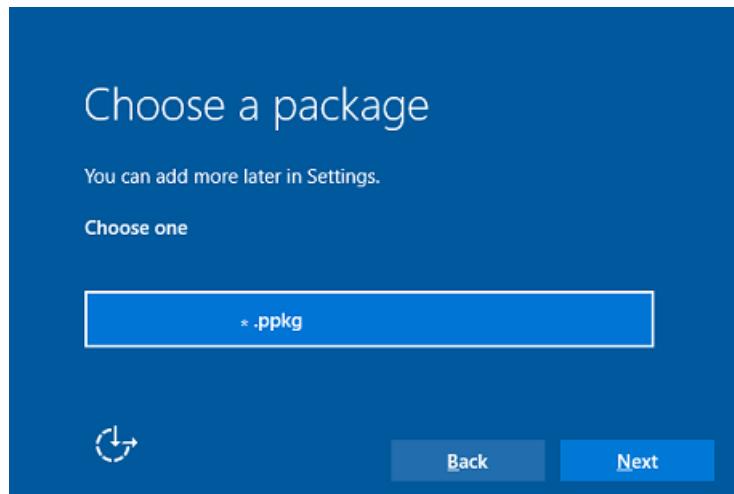
2. Insert the USB drive. Windows Setup will recognize the drive and ask if you want to set up the device. Select **Set up**.



3. The next screen asks you to select a provisioning source. Select **Removable Media** and tap **Next**.



4. Select the provisioning package (\*.ppkg) that you want to apply, and tap **Next**.



5. Select **Yes, add it**.

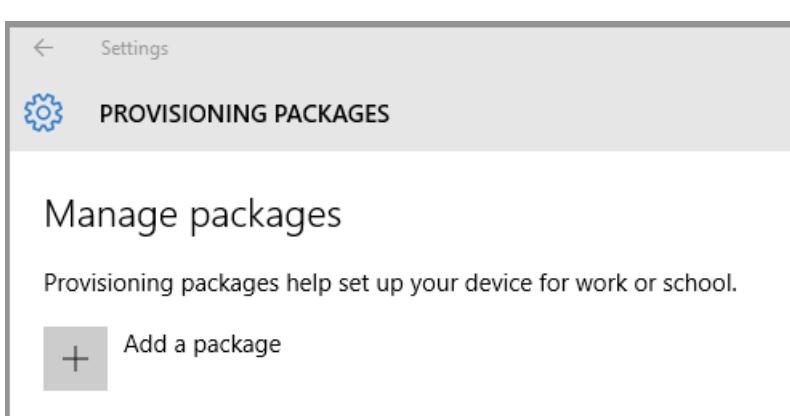


#### After setup, from a USB drive, network folder, or SharePoint site

1. Sign in with an admin account.
2. Insert the USB drive to a desktop computer, navigate to **Settings > Accounts > Access work or school > Add or remove a provisioning package > Add a package**, and select the package to install.

#### NOTE

If your provisioning package doesn't include the assigned access user account creation, make sure the account you specified in the multi-app configuration XML exists on the device.



#### Validate provisioning

- Go to **Settings > Accounts > Access work or school**, and then click **Add or remove a provisioning package**. You should see a list of packages that were applied to the device, including the one you applied for the multi-app configuration.
- Optionally, run Event Viewer (eventvwr.exe) and look through logs under **Applications and Services Logs >**

## Use MDM to deploy the multi-app configuration

Multi-app kiosk mode is enabled by the [AssignedAccess configuration service provider \(CSP\)](#). Your MDM policy can contain the assigned access configuration XML.

If your device is enrolled with a MDM server which supports applying the assigned access configuration, you can use it to apply the setting remotely.

The OMA-URI for multi-app policy is `./Device/Vendor/MSFT/AssignedAccess/Configuration`.

## Use MDM Bridge WMI Provider to configure assigned access

Environments that use WMI can use the [MDM Bridge WMI Provider](#) to configure the MDM\_AssignedAccess class. See [PowerShell Scripting with WMI Bridge Provider](#) for more details about using a PowerShell script to configure AssignedAccess.

Here's an example to set AssignedAccess configuration:

1. Download the [psexec tool](#).
2. Run `psexec.exe -i -s cmd.exe`.
3. In the command prompt launched by psexec.exe, enter `powershell.exe` to open PowerShell.
4. Execute the following script:

```
$nameSpaceName="root\cimv2\mdm\dmmap"
$className="MDM_AssignedAccess"
$obj = Get-CimInstance -Namespace $nameSpaceName -ClassName $className
$obj.Configuration = @"
<?xml version="1.0" encoding="utf-8"?>
<AssignedAccessConfiguration xmlns="http://schemas.microsoft.com/AssignedAccess/2017/config">
 <Profiles>
 <Profile Id="{9A2A490F-10F6-4764-974A-43B19E722C23}">
 <AllAppsList>
 <AllowedApps>
 <App AppUserModelId="Microsoft.ZuneMusic_8wekyb3d8bbwe!Microsoft.ZuneMusic" />
 <App AppUserModelId="Microsoft.ZuneVideo_8wekyb3d8bbwe!Microsoft.ZuneVideo" />
 <App AppUserModelId="Microsoft.Windows.Photos_8wekyb3d8bbwe!App" />
 <App AppUserModelId="Microsoft.BingWeather_8wekyb3d8bbwe!App" />
 <App AppUserModelId="Microsoft.WindowsCalculator_8wekyb3d8bbwe!App" />
 <App DesktopAppPath="%windir%\system32\mspaint.exe" />
 <App DesktopAppPath="C:\Windows\System32\notepad.exe" />
 </AllowedApps>
 </AllAppsList>
 <StartLayout>
 <![CDATA[<LayoutModificationTemplate
xmlns:defaultlayout="http://schemas.microsoft.com/Start/2014/FullDefaultLayout"
xmlns:start="http://schemas.microsoft.com/Start/2014/StartLayout" Version="1"
xmlns="http://schemas.microsoft.com/Start/2014/LayoutModification">
<LayoutOptions StartTileGroupCellWidth="6" />
<DefaultLayoutOverride>
<StartLayoutCollection>
<defaultlayout:StartLayout GroupCellWidth="6" />
<start:Group Name="Group1" />
<start:Tile Size="4x4" Column="0" Row="0" AppUserModelID="Microsoft.ZuneMusic_8wekyb3d8bbwe!Microsoft.ZuneMusic" />
<start:Tile Size="2x2" Column="4" Row="2" AppUserModelID="Microsoft.ZuneVideo_8wekyb3d8bbwe!Microsoft.ZuneVideo" />
<start:Tile Size="2x2" Column="4" Row="0" AppUserModelID="Microsoft.Windows.Photos_8wekyb3d8bbwe!App" />
<start:Tile Size="2x2" Column="4" Row="4" AppUserModelID="Microsoft.BingWeather_8wekyb3d8bbwe!App" />

```

```

AppUserModelID="Microsoft.WindowsCalculator_8wekyb3d8bbwe!App"; />
 </start:Group>
 <start:Group Name="Group2">
 <start:DesktopApplicationTile Size="2x2"; Column="2";
Row="0"; DesktopApplicationLinkPath="%ALLUSERSPROFILE%\Microsoft\Windows\Start
Menu\Programs\Accessories\Paint.lnk"; />
 <start:DesktopApplicationTile Size="2x2"; Column="0";
Row="0"; DesktopApplicationLinkPath="%APPDATA%\Microsoft\Windows\Start
Menu\Programs\Accessories\Notepad.lnk"; />
 </start:Group>
 </defaultlayout:StartLayout>
 </StartLayoutCollection>
 </DefaultLayoutOverride>
 </LayoutModificationTemplate>
]]>
 </StartLayout>
 <Taskbar ShowTaskbar="true"/>
 </Profile>
 </Profiles>
 <Configs>
 <Config>
 <Account>MultiAppKioskUser</Account>
 <DefaultProfile Id="{9A2A490F-10F6-4764-974A-43B19E722C23}"/>
 </Config>
 <Configs>
 </AssignedAccessConfiguration>
 @{
 Set-CimInstance -CimInstance $obj

```

## Validate multi-app kiosk configuration

Sign in with the assigned access user account you specified in the configuration to check out the multi-app experience.

### NOTE

The setting will take effect the next time the assigned access user signs in. If that user account is signed in when you apply the configuration, make sure the user signs out and signs back in to validate the experience.

The following sections explain what to expect on a multi-app kiosk.

### App launching and switching experience

In the multi-app mode, to maximize the user productivity and streamline the experience, an app will be always launched in full screen when the users click the tile on the Start. The users can minimize and close the app, but cannot resize the app window.

The users can switch apps just as they do today in Windows. They can use the Task View button, Alt + Tab hotkey, and the swipe in from the left gesture to view all the open apps in task view. They can click the Windows button to show Start, from which they can open apps, and they can switch to an opened app by clicking it on the taskbar.

### Start changes

When the assigned access user signs in, you should see a restricted Start experience:

- Start gets launched in full screen and prevents the end user from accessing the desktop.
- Start shows the layout aligned with what you defined in the multi-app configuration XML.
- Start prevents the end user from changing the tile layout.
  - The user cannot resize, reposition, and unpin the tiles.
  - The user cannot pin additional tiles on the start.

- Start hides **All Apps** list.
- Start hides all the folders on Start (including File Explorer, Settings, Documents, Downloads, Music, Pictures, Videos, HomeGroup, Network, and Personal folders).
- Only **User** and **Power** buttons are available. (You can control whether to show the **User/Power** buttons using [existing policies](#).)
- Start hides **Change account settings** option under **User** button.

## Taskbar changes

If the applied multi-app configuration enables taskbar, when the assigned access user signs in, you should see a restricted Taskbar experience:

- Disables context menu of Start button (Quick Link)
- Disables context menu of taskbar
- Prevents the end user from changing the taskbar
- Disables Cortana and Search Windows
- Hides notification icons and system icons, e.g. Action Center, People, Windows Ink Workspace
- Allows the end user to view the status of the network connection and power state, but disables the flyout of **Network/Power** to prevent end user from changing the settings

## Blocked hotkeys

The multi-app mode blocks the following hotkeys, which are not relevant for the lockdown experience.

HOTKEY	ACTION
Windows logo key + A	Open Action center
Windows logo key + Shift + C	Open Cortana in listening mode
Windows logo key + D	Display and hide the desktop
Windows logo key + Alt + D	Display and hide the date and time on the desktop
Windows logo key + E	Open File Explorer
Windows logo key + F	Open Feedback Hub
Windows logo key + G	Open Game bar when a game is open
Windows logo key + I	Open Settings
Windows logo key + J	Set focus to a Windows tip when one is available.
Windows logo key + O	Lock device orientation
Windows logo key + Q	Open search
Windows logo key + R	Open the Run dialog box
Windows logo key + S	Open search
Windows logo key + X	Open the Quick Link menu

HOTKEY	ACTION
Windows logo key + comma (,)	Temporarily peek at the desktop
Windows logo key + Ctrl + F	Search for PCs (if you're on a network)

## Locked-down Ctrl+Alt+Del screen

The multi-app mode removes options (e.g. **Change a password**, **Task Manager**, **Network**) in the Ctrl+Alt+Del screen to ensure the users cannot access the functionalities that are not allowed in the lockdown experience.

## Auto-trigger touch keyboard

In the multi-app mode, the touch keyboard will be automatically triggered when there is an input needed and no physical keyboard is attached on touch-enabled devices. You don't need to configure any other setting to enforce this behavior.

## Considerations for Windows Mixed Reality immersive headsets

With the advent of [mixed reality devices \(video link\)](#), you might want to create a kiosk that can run mixed reality apps.

To create a multi-app kiosk that can run mixed reality apps, you must include the following apps in the [AllowedApps list](#):

```
<App AppUserModelId="MixedRealityLearning_cw5n1h2txyewy!MixedRealityLearning" />
<App AppUserModelId="HoloShell_cw5n1h2txyewy!HoloShell" />
<App AppUserModelId="Microsoft.Windows.HolographicFirstRun_cw5n1h2txyewy!App" />
```

These are in addition to any mixed reality apps that you allow.

**Before your kiosk user signs in:** An admin user must sign in to the PC, connect a mixed reality device, and complete the guided setup for the Mixed Reality Portal. The first time that the Mixed Reality Portal is set up, some files and content are downloaded. A kiosk user would not have permissions to download and so their setup of the Mixed Reality Portal would fail.

After the admin has completed setup, the kiosk account can sign in and repeat the setup. The admin user may want to complete the kiosk user setup before providing the PC to employees or customers.

There is a difference between the mixed reality experiences for a kiosk user and other users. Typically, when a user connects a mixed reality device, they begin in the [Mixed Reality home](#). The Mixed Reality home is a shell that runs in "silent" mode when the PC is configured as a kiosk. When a kiosk user connects a mixed reality device, they will see only a blank display in the device, and will not have access to the features and functionality available in the home. To run a mixed reality app, the kiosk user must launch the app from the PC Start screen.

## Policies set by multi-app kiosk configuration

It is not recommended to set policies enforced in assigned access multi-app mode to different values using other channels, as the multi-app mode has been optimized to provide a locked-down experience.

When the multi-app assigned access configuration is applied on the device, certain policies are enforced system-wide, and will impact other users on the device.

### Group Policy

The following local policies affect all **non-administrator** users on the system, regardless whether the user is configured as an assigned access user or not. This includes local users, domain users, and Azure Active Directory users.

SETTING	VALUE
Remove access to the context menus for the task bar	Enabled
Clear history of recently opened documents on exit	Enabled
Prevent users from customizing their Start Screen	Enabled
Prevent users from uninstalling applications from Start	Enabled
Remove All Programs list from the Start menu	Enabled
Remove Run menu from Start Menu	Enabled
Disable showing balloon notifications as toast	Enabled
Do not allow pinning items in Jump Lists	Enabled
Do not allow pinning programs to the Taskbar	Enabled
Do not display or track items in Jump Lists from remote locations	Enabled
Remove Notifications and Action Center	Enabled
Lock all taskbar settings	Enabled
Lock the Taskbar	Enabled
Prevent users from adding or removing toolbars	Enabled
Prevent users from resizing the taskbar	Enabled
Remove frequent programs list from the Start Menu	Enabled
Remove Pinned programs from the taskbar	Enabled
Remove the Security and Maintenance icon	Enabled
Turn off all balloon notifications	Enabled
Turn off feature advertisement balloon notifications	Enabled
Turn off toast notifications	Enabled
Remove Task Manager	Enabled
Remove Change Password option in Security Options UI	Enabled
Remove Sign Out option in Security Options UI	Enabled
Remove All Programs list from the Start Menu	Enabled – Remove and disable setting

SETTING	VALUE
Prevent access to drives from My Computer	Enabled - Restrict all drivers

#### NOTE

When **Prevent access to drives from My Computer** is enabled, users can browse the directory structure in File Explorer, but they cannot open folders and access the contents. Also, they cannot use the **Run** dialog box or the **Map Network Drive** dialog box to view the directories on these drives. The icons representing the specified drives still appear in File Explorer, but if users double-click the icons, a message appears explaining that a setting prevents the action. This setting does not prevent users from using programs to access local and network drives. It does not prevent users from using the Disk Management snap-in to view and change drive characteristics.

## MDM policy

Some of the MDM policies based on the [Policy configuration service provider \(CSP\)](#) affect all users on the system (i.e. system-wide).

SETTING	VALUE	SYSTEM-WIDE
Experience/AllowCortana	0 - Not allowed	Yes
Start/AllowPinnedFolderSettings	0 - Shortcut is hidden and disables the setting in the Settings app	Yes
Start/HidePeopleBar	1 - True (hide)	No
Start/HideChangeAccountSettings	1 - True (hide)	Yes
WindowsInkWorkspace/AllowWindowsInkWorkspace	0 - Access to ink workspace is disabled and the feature is turned off	Yes
Start/StartLayout	Configuration dependent	No
WindowsLogon/DontDisplayNetworkSelectionUI	<Enabled/>	Yes

## Provision .lnk files using Windows Configuration Designer

First, create your desktop app's shortcut file by installing the app on a test device. Right-click the installed application, and choose **Send to > Desktop (create shortcut)**. Rename the shortcut to `<appName>.lnk`

Next, create a batch file with two commands. If the desktop app is already installed on the target device, skip the first command for MSI install.

```
msiexec /I "<appName>.msi" /qn /norestart
copy <appName>.lnk "%AllUsersProfile%\Microsoft\Windows\Start Menu\Programs\<appName>.lnk"
```

In Windows Configuration Designer, under **ProvisioningCommands > DeviceContext**:

- Under **CommandFiles**, upload your batch file, your .lnk file, and your desktop app installation file
- Under **CommandLine**, enter cmd /c *FileName.bat*

# Troubleshoot multi-app kiosk

10/17/2017 • 1 min to read • [Edit Online](#)

## Applies to

- Windows 10

## Unexpected results

For example:

- Start is not launched in full-screen
- Blocked hotkeys are allowed
- Task Manager, Cortana, or Settings can be launched
- Start layout has more apps than expected

## Troubleshooting steps

1. [Verify that the provisioning package is applied successfully.](#)
2. Verify that the account (config) is mapped to a profile in the configuration XML file.
3. Verify that the configuration XML file is authored and formatted correctly. Correct any configuration errors, then create and apply a new provisioning package. Sign out and sign in again to check the new configuration.

## Apps configured in AllowedList are blocked

1. Ensure the account is mapped to the correct profile and that the apps are specific for that profile.
2. Check the EventViewer logs for Applocker and AppxDeployment (under **Application and Services Logs\Microsoft\Windows**).

## Start layout not as expected

- Make sure the Start layout is authored correctly. Ensure that the attributes **Size**, **Row**, and **Column** are specified for each application and are valid.
- Check if the apps included in the Start layout are installed for the assigned access user.
- Check if the shortcut exists on the target device, if a desktop app is missing on Start.

# Use AppLocker to create a Windows 10 kiosk that runs multiple apps

10/17/2017 • 4 min to read • [Edit Online](#)

## Applies to

- Windows 10

Learn how to configure a device running Windows 10 Enterprise or Windows 10 Education, version 1703 and earlier, so that users can only run a few specific apps. The result is similar to a [kiosk device](#), but with multiple apps available. For example, you might set up a library computer so that users can search the catalog and browse the Internet, but can't run any other apps or change computer settings.

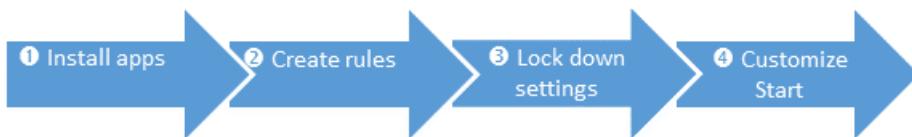
### NOTE

For devices running Windows 10, version 1709, we recommend the [multi-app kiosk method](#).

You can restrict users to a specific set of apps on a device running Windows 10 Enterprise or Windows 10 Education by using [AppLocker](#). AppLocker rules specify which apps are allowed to run on the device.

AppLocker rules are organized into collections based on file format. If no AppLocker rules for a specific rule collection exist, all files with that file format are allowed to run. However, when an AppLocker rule for a specific rule collection is created, only the files explicitly allowed in a rule are permitted to run. For more information, see [How AppLocker works](#).

This topic describes how to lock down apps on a local device. You can also use AppLocker to set rules for applications in a domain by using Group Policy.



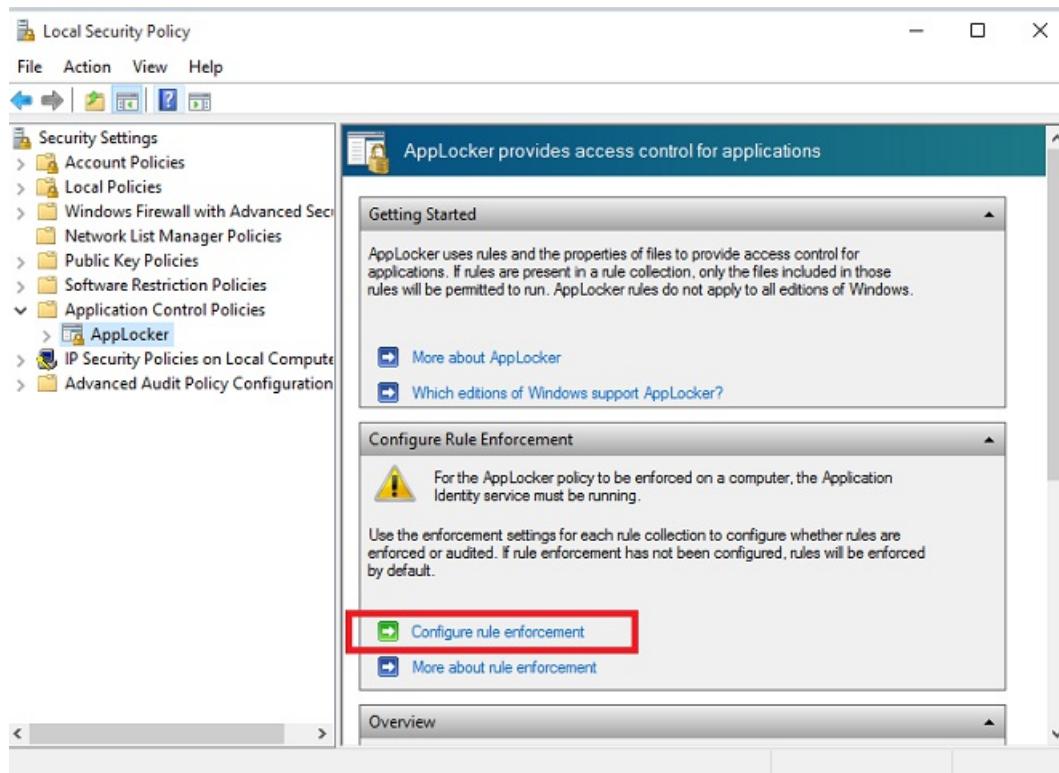
## Install apps

First, install the desired apps on the device for the target user account(s). This works for both Store and Win32. For Store apps, you must log on as that user for the app to install. For Win32 you can install an app for all users without logging on to the particular account.

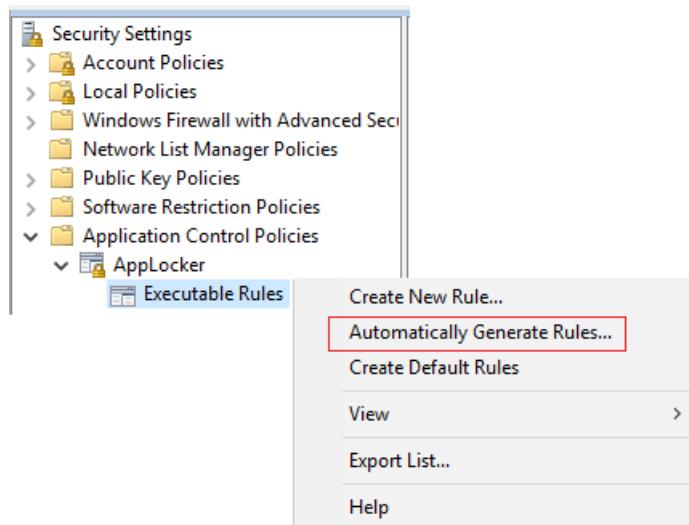
## Use AppLocker to set rules for apps

After you install the desired apps, set up AppLocker rules to only allow specific apps, and block everything else.

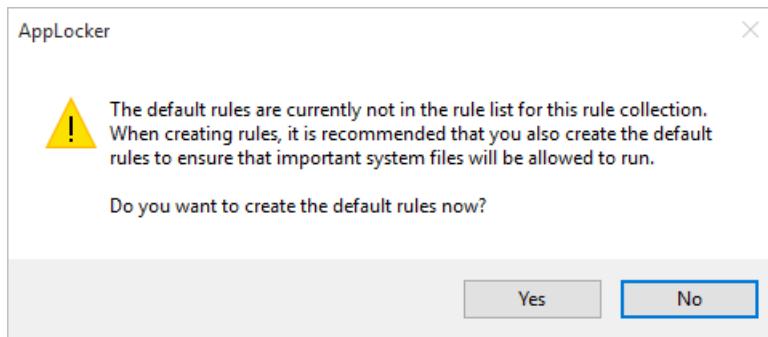
- Run Local Security Policy (secpol.msc) as an administrator.
- Go to **Security Settings > Application Control Policies > AppLocker**, and select **Configure rule enforcement**.



3. Check **Configured** under **Executable rules**, and then click **OK**.
4. Right-click **Executable Rules** and then click **Automatically generate rules**.



5. Select the folder that contains the apps that you want to permit, or select C:\ to analyze all apps.
6. Type a name to identify this set of rules, and then click **Next**.
7. On the **Rule Preferences** page, click **Next**. Be patient, it might take awhile to generate the rules.
8. On the **Review Rules** page, click **Create**. The wizard will now create a set of rules allowing the installed set of apps.
9. Read the message and click **Yes**.



10. (optional) If you want a rule to apply to a specific set of users, right-click on the rule and select **Properties**. Then use the dialog to choose a different user or group of users.
11. (optional) If rules were generated for apps that should not be run, you can delete them by right-clicking on the rule and selecting **Delete**.
12. Before AppLocker will enforce rules, the **Application Identity** service must be turned on. To force the Application Identity service to automatically start on reset, open a command prompt and run:

```
sc config appidsvc start=auto
```

13. Restart the device.

## Other settings to lock down

In addition to specifying the apps that users can run, you should also restrict some settings and functions on the device. For a more secure experience, we recommend that you make the following configuration changes to the device:

- Remove **All apps**.

Go to **Group Policy Editor** > **User Configuration** > **Administrative Templates\Start Menu and Taskbar\Remove All Programs list from the Start menu**.

- Hide **Ease of access** feature on the logon screen.

Go to **Control Panel** > **Ease of Access** > **Ease of Access Center**, and turn off all accessibility tools.

- Disable the hardware power button.

Go to **Power Options** > **Choose what the power button does**, change the setting to **Do nothing**, and then **Save changes**.

- Disable the camera.

Go to **Settings** > **Privacy** > **Camera**, and turn off **Let apps use my camera**.

- Turn off app notifications on the lock screen.

Go to **Group Policy Editor** > **Computer Configuration** > **Administrative Templates\System\Logon\Turn off app notifications on the lock screen**.

- Disable removable media.

Go to **Group Policy Editor** > **Computer Configuration** > **Administrative Templates\System\Device Installation\Device Installation Restrictions**. Review the policy settings available in **Device Installation Restrictions** for the settings applicable to your situation.

### Note

To prevent this policy from affecting a member of the Administrators group, in **Device Installation Restrictions**, enable **Allow administrators to override Device Installation Restriction policies**.

To learn more about locking down features, see [Customizations for Windows 10 Enterprise](#).

## Customize Start screen layout for the device (recommended)

Configure the Start menu on the device to only show tiles for the permitted apps. You will make the changes manually, export the layout to an .xml file, and then apply that file to devices to prevent users from making changes. For instructions, see [Manage Windows 10 Start layout options](#).

# Multi-app kiosk XML reference

10/17/2017 • 1 min to read • [Edit Online](#)

## Applies to

- Windows 10

## Full XML sample

```

<?xml version="1.0" encoding="utf-8" ?>
<AssignedAccessConfiguration xmlns="http://schemas.microsoft.com/AssignedAccess/2017/config">
 <Profiles>
 <Profile Id="{9A2A490F-10F6-4764-974A-43B19E722C23}">
 <AllAppsList>
 <AllowedApps>
 <App AppUserModelId="Microsoft.ZuneMusic_8wekyb3d8bbwe!Microsoft.ZuneMusic" />
 <App AppUserModelId="Microsoft.ZuneVideo_8wekyb3d8bbwe!Microsoft.ZuneVideo" />
 <App AppUserModelId="Microsoft.Windows.Photos_8wekyb3d8bbwe!App" />
 <App AppUserModelId="Microsoft.BingWeather_8wekyb3d8bbwe!App" />
 <App AppUserModelId="Microsoft.WindowsCalculator_8wekyb3d8bbwe!App" />
 <App DesktopAppPath="%windir%\system32\mspaint.exe" />
 <App DesktopAppPath="C:\Windows\System32\notepad.exe" />
 </AllowedApps>
 </AllAppsList>
 <StartLayout>
 <![CDATA[<LayoutModificationTemplate
xmlns:defaultlayout="http://schemas.microsoft.com/Start/2014/FullDefaultLayout"
xmlns:start="http://schemas.microsoft.com/Start/2014/StartLayout" Version="1"
xmlns="http://schemas.microsoft.com/Start/2014/LayoutModification">
 <LayoutOptions StartTileGroupCellWidth="6" />
 <DefaultLayoutOverride>
 <StartLayoutCollection>
 <defaultlayout:StartLayout GroupCellWidth="6">
 <start:Group Name="Group1">
 <start:Tile Size="4x4" Column="0" Row="0"
AppUserModelID="Microsoft.ZuneMusic_8wekyb3d8bbwe!Microsoft.ZuneMusic" />
 <start:Tile Size="2x2" Column="4" Row="2"
AppUserModelID="Microsoft.ZuneVideo_8wekyb3d8bbwe!Microsoft.ZuneVideo" />
 <start:Tile Size="2x2" Column="4" Row="0"
AppUserModelID="Microsoft.Windows.Photos_8wekyb3d8bbwe!App" />
 <start:Tile Size="2x2" Column="4" Row="4"
AppUserModelID="Microsoft.BingWeather_8wekyb3d8bbwe!App" />
 <start:Tile Size="4x2" Column="0" Row="4"
AppUserModelID="Microsoft.WindowsCalculator_8wekyb3d8bbwe!App" />
 </start:Group>
 <start:Group Name="Group2">
 <start:DesktopApplicationTile Size="2x2" Column="2" Row="0"
DesktopApplicationLinkPath="%ALLUSERSPROFILE%\Microsoft\Windows\Start Menu\Programs\Accessories\Paint.lnk" />
 <start:DesktopApplicationTile Size="2x2" Column="0" Row="0"
DesktopApplicationLinkPath="%APPDATA%\Microsoft\Windows\Start Menu\Programs\Accessories\Notepad.lnk" />
 </start:Group>
 </defaultlayout:StartLayout>
 </StartLayoutCollection>
 </DefaultLayoutOverride>
 </LayoutModificationTemplate>
]]>
 <StartLayout>
 <Taskbar ShowTaskbar="true"/>
 </StartLayout>
 </Profile>
 </Profiles>
 <Configs>
 <Config>
 <Account>MultiAppKioskUser</Account>
 <DefaultProfile Id="{9A2A490F-10F6-4764-974A-43B19E722C23}" />
 </Config>
 </Configs>
</AssignedAccessConfiguration>

```

## XSD for AssignedAccess configuration XML

```

<?xml version="1.0" encoding="utf-8"?>
<xss:schema
 elementFormDefault="qualified"
 xmlns:xss="http://www.w3.org/2001/XMLSchema">

```

```

xmlns="http://schemas.microsoft.com/AssignedAccess/2017/config"
targetNamespace="http://schemas.microsoft.com/AssignedAccess/2017/config"
>

<xs:complexType name="profile_list_t">
 <xs:sequence minOccurs="1" >
 <xs:element name="Profile" type="profile_t" minOccurs="1" maxOccurs="unbounded">
 <xs:unique name="duplicateRolesForbidden">
 <xs:selector xpath="Profile"/>
 <xs:field xpath="@Id"/>
 </xs:unique>
 </xs:element>
 </xs:sequence>
</xs:complexType>

<xs:complexType name="profile_t">
 <xs:sequence minOccurs="1" maxOccurs="1" >
 <xs:element name="AllAppsList" type="allappslist_t" minOccurs="1" maxOccurs="1">
 <xs:unique name="ForbidDupApps">
 <xs:selector xpath="App"/>
 <xs:field xpath="@AppUserId"/>
 <xs:field xpath="@DesktopAppPath"/>
 </xs:unique>
 </xs:element>
 <xs:element name="StartLayout" type="xs:string" minOccurs="1" maxOccurs="1"/>
 <xs:element name="Taskbar" type="taskbar_t" minOccurs="1" maxOccurs="1"/>
 </xs:sequence>
 <xs:attribute name="Id" type="guid_t" use="required"/>
 <xs:attribute name="Name" type="xs:string" use="optional"/>
</xs:complexType>

<xs:complexType name="allappslist_t">
 <xs:sequence minOccurs="1" >
 <xs:element name="AllowedApps" type="allowedapps_t" minOccurs="1" maxOccurs="1">
 </xs:element>
 </xs:sequence>
</xs:complexType>

<xs:complexType name="allowedapps_t">
 <xs:sequence minOccurs="1" maxOccurs="1" >
 <xs:element name="App" type="app_t" minOccurs="1" maxOccurs="unbounded">
 <xs:key name="mutexAumidOrDesktopApp">
 <xs:selector xpath=".//>
 <xs:field xpath="@AppUserId|@DesktopAppPath"/>
 </xs:key>
 </xs:element>
 </xs:sequence>
</xs:complexType>

<xs:complexType name="app_t">
 <xs:attribute name="AppUserId" type="xs:string"/>
 <xs:attribute name="DesktopAppPath" type="xs:string"/>
</xs:complexType>

<xs:complexType name="taskbar_t">
 <xs:attribute name="ShowTaskbar" type="xs:boolean" use="required"/>
</xs:complexType>

<xs:complexType name="profileId_t">
 <xs:attribute name="Id" type="guid_t" use="required"/>
</xs:complexType>

<xs:simpleType name="guid_t">
 <xs:restriction base="xs:string">
 <xs:pattern value="\{[0-9a-fA-F]\{8\}\-\{[0-9a-fA-F]\{4\}\-\{3\}[0-9a-fA-F]\{12\}\}"/>
 </xs:restriction>
</xs:simpleType>

<xs:complexType name="config_list_t">

```

```
<xs:sequence minOccurs="1" >
 <xs:element name="Config" type="config_t" minOccurs="1" maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType>

<xs:complexType name="config_t">
 <xs:sequence minOccurs="1" maxOccurs="1">
 <xs:element name="Account" type="xs:string" minOccurs="1" maxOccurs="1"/>
 <xs:element name="DefaultProfile" type="profileId_t" minOccurs="1" maxOccurs="1"/>
 </xs:sequence>
</xs:complexType>

<!--below is the definition of the config xml content-->
<xs:element name="AssignedAccessConfiguration">
 <xs:complexType>
 <xs:all minOccurs="1">
 <xs:element name="Profiles" type="profile_list_t">
 </xs:element>
 <xs:element name="Configs" type="config_list_t"/>
 </xs:all>
 </xs:complexType>
</xs:element>
</xs:schema></pre>
```

# Configure Windows 10 Mobile devices

7/28/2017 • 1 min to read • [Edit Online](#)

Windows 10 Mobile enables administrators to define what users can see and do on a device, which you might think of as "configuring" or "customizing" or "device lockdown". Your device configuration can provide a standard Start screen with pre-installed apps, or restrict various settings and features, or even limit the device to run only a single app (kiosk).

## In this section

TOPIC	DESCRIPTION
<a href="#">Set up a kiosk on Windows 10 Mobile or Windows 10 Mobile Enterprise</a>	You can configure a device running Windows 10 Mobile or Windows 10 Mobile Enterprise as a kiosk device, so that users can only interact with a single application that you select.
<a href="#">Use Windows Configuration Designer to configure Windows 10 Mobile devices</a>	Use Windows Configuration Designer to create provisioning packages. Using provisioning packages, you can easily specify desired configuration and settings required to enroll the devices into management and then apply that configuration to target devices in a matter of minutes.
<a href="#">Use the Lockdown Designer app to configure Windows 10 Mobile devices</a>	The Lockdown Designer app provides a guided wizard-like process to generate a Lockdown XML file that you can apply to devices running Windows 10 Mobile.
<a href="#">Configure Windows 10 Mobile using Lockdown XML</a>	Windows 10 Mobile allows enterprises to lock down a device, define multiple user roles, and configure custom layouts on a device.
<a href="#">Start layout XML for mobile editions of Windows 10 (reference)</a>	On Windows 10 Mobile, you can use the XML-based layout to modify the Start screen and provide the most robust and complete Start customization experience. This reference topic describes the supported elements and attributes for the LayoutModification.xml file.
<a href="#">Settings and quick actions that can be locked down in Windows 10 Mobile</a>	This topic lists the settings and quick actions that can be locked down in Windows 10 Mobile.
<a href="#">Product IDs in Windows 10 Mobile</a>	You can use the product ID and Application User Model (AUMID) in Lockdown.xml to specify apps that will be available to the user.

# Set up a kiosk on Windows 10 Mobile or Windows 10 Mobile Enterprise

7/28/2017 • 7 min to read • [Edit Online](#)

## Applies to

- Windows 10 Mobile

A device in kiosk mode runs a specified app with no access to other device functions, menus, or settings. You use the [Enterprise Assigned Access](#) configuration service provider (CSP) to configure a kiosk experience. You can also configure a device running Windows 10 Mobile or Windows 10 Mobile Enterprise, version 1607 or earlier, for kiosk mode by using the [Apps Corner](#) feature. (Apps Corner is removed in version 1703.)

## Enterprise Assigned Access

Enterprise Assigned Access allows you to put your Windows 10 Mobile or Windows 10 Mobile Enterprise device in kiosk mode by creating a user role that has only a single app, set to run automatically, in the Allow list.

### NOTE

The app can be a Universal Windows app, Universal Windows Phone 8 app, or a legacy Silverlight app.

### Set up Enterprise Assigned Access in MDM

In AssignedAccessXml, for Application, you enter the product ID for the app to run in kiosk mode. Find product IDs at [Product IDs in Windows 10 Mobile](#).

See the [technical reference for the Enterprise Assigned Access configuration service provider \(CSP\)](#).

### Set up assigned access using Windows Configuration Designer

#### IMPORTANT

When you build a provisioning package, you may include sensitive information in the project files and in the provisioning package (.ppkg) file. Although you have the option to encrypt the .ppkg file, project files are not encrypted. You should store the project files in a secure location and delete the project files when they are no longer needed.

#### Create the `AssignedAccess.xml` file

- Create an `AssignedAccess.xml` file that specifies the app the device will run. (You can name use any file name.) For instructions on AssignedAccessXml, see [EnterpriseAssignedAccess CSP](#).

### NOTE

Do not escape the xml in `AssignedAccess.xml` file as Windows Configuration Designer will do that when building the package. Providing escaped xml in Windows ICD will cause building the package fail.

#### Create the provisioning package

- [Install Windows Configuration Designer](#).
- Open Windows Configuration Designer (if you installed it from the Windows ADK,

```
%windir%\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Imaging and Configuration Designer\x86\ICD.exe
```

).

3. Choose **Advanced provisioning**.
4. Name your project, and click **Next**.
5. Choose **All Windows mobile editions** and click **Next**.
6. On **New project**, click **Finish**. The workspace for your package opens.
7. Expand **Runtime settings > EmbeddedLockdownProfiles**, and click **AssignedAccessXml**.
8. Click **Browse** to select the *AssignedAccess.xml* file.
9. On the **File** menu, select **Save**.
10. On the **Export** menu, select **Provisioning package**.
11. Change **Owner** to **IT Admin**, which will set the precedence of this provisioning package higher than provisioning packages applied to this device from other sources, and then select **Next**.
12. Optional. In the **Provisioning package security** window, you can choose to encrypt the package and enable package signing.
  - **Enable package encryption** - If you select this option, an auto-generated password will be shown on the screen.
  - **Enable package signing** - If you select this option, you must select a valid certificate to use for signing the package. You can specify the certificate by clicking **Select** and choosing the certificate you want to use to sign the package.
13. Click **Next** to specify the output location where you want the provisioning package to go when it's built. By default, Windows ICD uses the project folder as the output location.

Optionally, you can click **Browse** to change the default output location.
14. Click **Next**.
15. Click **Build** to start building the package. The provisioning package doesn't take long to build. The project information is displayed in the build page and the progress bar indicates the build status.

If you need to cancel the build, click **Cancel**. This cancels the current build process, closes the wizard, and takes you back to the **Customizations Page**.
16. If your build fails, an error message will show up that includes a link to the project folder. You can scan the logs to determine what caused the error. Once you fix the issue, try building the package again.

If your build is successful, the name of the provisioning package, output directory, and project directory will be shown.

  - If you choose, you can build the provisioning package again and pick a different path for the output package. To do this, click **Back** to change the output package name and path, and then click **Next** to start another build.
  - If you are done, click **Finish** to close the wizard and go back to the **Customizations Page**.
17. Select the **output location** link to go to the location of the package.

#### Distribute the provisioning package

You can distribute that .ppkg to mobile devices using any of the following methods:

- Removable media (USB/SD)

#### To apply a provisioning package from removable media

1. Copy the provisioning package file to the root directory on a micro SD card.
2. On the device, insert the micro SD card containing the provisioning package.
3. Go to **Settings > Accounts > Provisioning**.
4. Tap **Add a package**.
5. On the **Choose a method** screen, in the **Add from** dropdown menu, select **Removable Media**.
6. Select a package will list all available provisioning packages on the micro SD card. Tap the desired package, and then tap **Add**.
7. You will see a message that tells you what the package will do the device, such as **Adding it will: Lock down the user interface**. Tap **Yes, add it**.
8. Restart the device and verify that the runtime settings that were configured in the provisioning package were applied to the device.

- Email

#### To apply a provisioning package sent in email

1. Send the provisioning package in email to an account on the device.
2. Open the email on the device, and then double-tap the attached file.
3. You will see a message that tells you what the package will do the device, such as **Adding it will: Lock down the user interface**. Tap **Yes, add it**.
4. Restart the device and verify that the runtime settings that were configured in the provisioning package were applied to the device.

- USB tether

#### To apply a provisioning package using USB tether

1. Connect the device to your PC by USB.
2. Select the provisioning package that you want to use to provision the device, and then drag and drop the file to your device.
3. The provisioning package installation dialog will appear on the phone.
4. You will see a message that tells you what the package will do the device, such as **Adding it will: Lock down the user interface**. Tap **Yes, add it**.
5. Restart the device and verify that the runtime settings that were configured in the provisioning package were applied to the device.

## Apps Corner

### NOTE

For Windows 10, versions 1507, 1511, and 1607 only.

Apps Corner lets you set up a custom Start screen on your Windows 10 Mobile or Windows 10 Mobile Enterprise device, where you can share only the apps you choose with the people you let use your device. You configure a device for kiosk mode by selecting a single app to use in Apps Corner.

### To set up Apps Corner

1. On Start , swipe over to the App list, then tap **Settings**  > **Accounts** > **Apps Corner**.
2. Tap **Apps**, tap to select the app that you want people to use in the kiosk mode, and then tap done .
3. If your phone doesn't already have a lock screen password, you can set one now to ensure that people can't get to your Start screen from Apps Corner. Tap **Protect my phone with a password**, click **Add**, type a PIN in the **New PIN** box, type it again in the **Confirm PIN** box, and then tap **OK**. Press **Back**  to the Apps Corner settings.
4. Turn **Action center** on or off, depending on whether you want people to be able to use these features when using the device in kiosk mode.
5. Tap **advanced**, and then turn features on or off, depending on whether you want people to be able to use them.
6. Press **Back**  when you're done.

## To use Apps Corner

1. On Start , swipe over to the App list, then tap **Settings**  > **Accounts** > **Apps Corner** > launch .

### TIP

Want to get to Apps Corner with one tap? In **Settings**, tap **Apps Corner** > **pin** to pin the Apps Corner tile to your Start screen.

2. Give the device to someone else, so they can use the device and only the one app you chose.
3. When they're done and you get the device back, press and hold Power , and then swipe right to exit Apps Corner.

## Related topics

[Set up a kiosk on Windows 10 Pro, Enterprise, or Education](#)

[Configure Windows 10 Mobile using Lockdown XML](#)

[Product IDs in Windows 10 Mobile](#)

# Use Windows Configuration Designer to configure Windows 10 Mobile devices

7/28/2017 • 3 min to read • [Edit Online](#)

Windows provisioning makes it easy for IT administrators to configure end-user devices without imaging. Using provisioning packages, you can easily specify desired configuration, settings, and information required to enroll the devices into management, and then apply that configuration to target devices in a matter of minutes.

A provisioning package (.ppkg) is a container for a collection of configuration settings. Using Windows Configuration Designer, you can create provisioning packages that let you quickly and efficiently configure a device without having to install a new image.

Windows Configuration Designer can be installed from the [Windows Assessment and Deployment Kit \(ADK\) for Windows 10](#). Windows Configuration Designer is also available as an app in the Microsoft Store. [Learn more about installing Windows Configuration Designer](#).

## Create a provisioning package using the wizard

The **Provision Windows mobile devices** wizard lets you configure common settings for devices running Windows 10 Mobile in a simple, graphical workflow.

### Start a new project

1. Open Windows Configuration Designer:

- From either the Start screen or Start menu search, type 'Windows Configuration Designer' and click the Windows Configuration Designer shortcut,

or

- If you installed Windows Configuration Designer from the ADK, navigate to

```
C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Imaging and Configuration Designer\x86
```

(on an x64 computer) or

```
C:\Program Files\Windows Kits\10\Assessment and Deployment Kit\Imaging and Configuration Designer\x86\ICD.exe
```

(on an x86 computer), and then double-click **ICD.exe**.

2. On the **Start** page, choose **Provision Windows mobile devices**.

3. Enter a name for your project, and then click **Next**.

### Configure settings in the wizard

**1**

## Set up device

Enter a device name.

Optionally, you can enter a product key to upgrade the device from Windows 10 Mobile to Windows 10 Mobile Enterprise.

### Device name

Enter a unique 15-character name for the device. For help generating a unique name, you can use %SERIAL%, which includes a hardware-specific serial number, or you can use %RAND:x%, which generates random characters of x length.

Example device name values:

Contoso-%SERIAL%

Fabrikam-%RAND:5%

Contoso-%RAND:5%

### Enter product key

Optional: Enter a product key to upgrade Windows.

XXXXX-XXXXX-XXXXX-XXXXX-XXXXX

**2**

## Set up network

Toggle **On** or **Off** for wireless network connectivity.

If you select **On**, enter the SSID, network type (**Open** or **WPA2-Personal**), and (if **WPA2-Personal**) the password for the wireless network.

### Set up network

Connect devices to a Wi-Fi network

On

Network SSID \*

Required

Network type \*

Open

**3**

## Bulk Enrollment in Azure AD

Before you use a Windows Configuration Designer wizard to configure bulk Azure AD enrollment, [set up Azure AD join in your organization](#). The **maximum number of devices per user** setting in your Azure AD tenant determines how many times the bulk token that you get in the wizard can be used.

Set an expiration date for the token (maximum is 30 days from the date you get the token). Click **Get bulk token**. In the **Let's get you signed in** window, enter an account that has permissions to join a device to Azure AD, and then the password. Click **Accept** to give Windows Configuration Designer the necessary permissions.

**Warning:** You must run Windows Configuration Designer on Windows 10 to configure Azure Active Directory enrollment using any of the wizards.

### Bulk Enroll Devices in Azure Active Directory

Improve security and remote management by enrolling devices into Azure Active Directory

The default number of devices that can be joined to an Azure AD tenant is limited to 20. Contact your domain administrator to change the limit and [go here](#) for more info.

Bulk Token Expiry \*

04/8/2017

Bulk AAD Token \*

Get Bulk Token

**4**

## Finish

You can set a password to protect your provisioning package. You must enter this password when you apply the provisioning package to a device.

### Summary

#### Protect your package

Protect the contents of your package by specifying a password. The password length must be 8-16 characters.

No

After you're done, click **Create**. It only takes a few seconds. When the package is built, the location where the package is stored is displayed as a hyperlink at the bottom of the page.

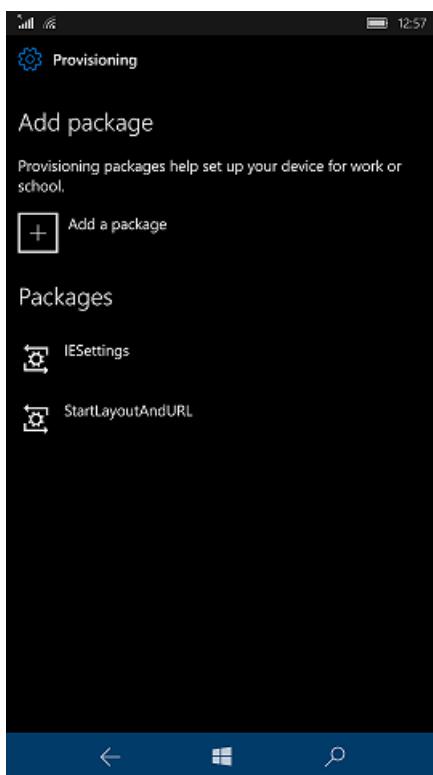
## Apply provisioning package

You can apply a provisioning package to a device running Windows 10 Mobile by using:

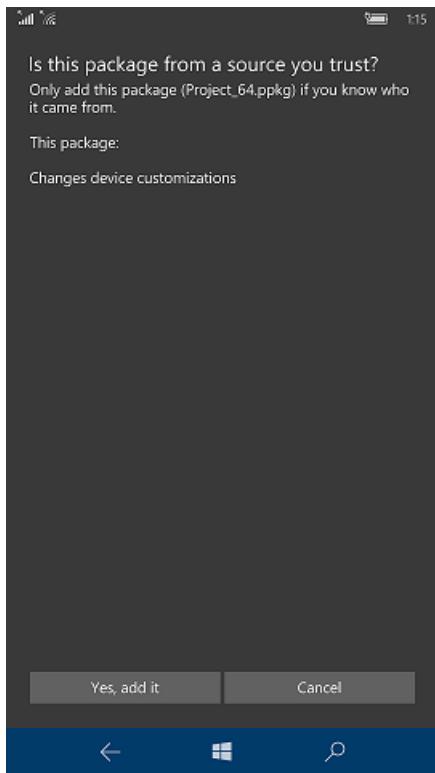
- removable media
- copying the provisioning package to the device
- [NFC tags](#)
- [barcodes](#)

### Using removable media

1. Insert an SD card containing the provisioning package into the device.
2. Navigate to **Settings > Accounts > Access work or school > Add or remove a provisioning package > Add a package**, and select the package to install.

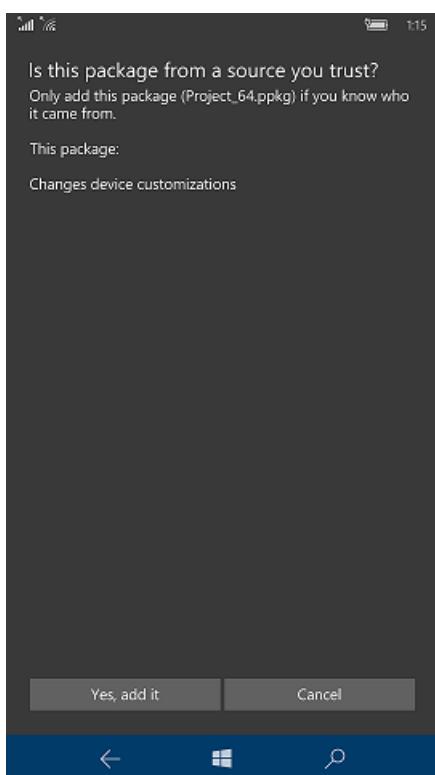


3. Click **Add**.
4. On the device, the **Is this package from a source you trust?** message will appear. Tap **Yes, add it**.



### Copying the provisioning package to the device

1. Connect the device to your PC through USB.
2. On the PC, select the provisioning package that you want to use to provision the device and then drag and drop the file to your device.
3. On the device, the **Is this package from a source you trust?** message will appear. Tap **Yes, add it**.



## Related topics

- [NFC-based device provisioning](#)
- [Use the package splitter tool](#)

# NFC-based device provisioning

7/28/2017 • 5 min to read • [Edit Online](#)

## Applies to

- Windows 10 Mobile

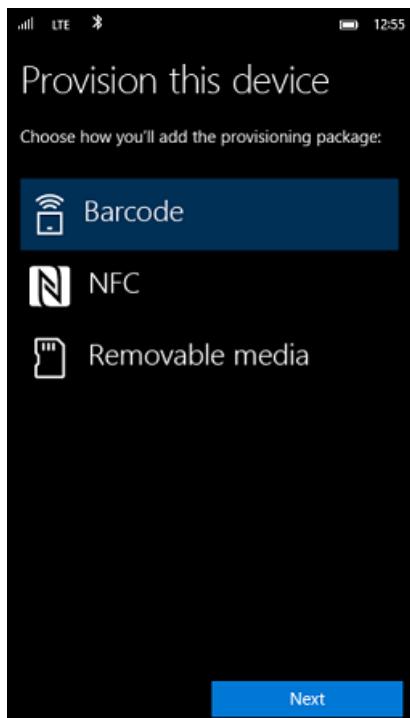
Near field communication (NFC) enables Windows 10 Mobile Enterprise and Windows 10 Mobile devices to communicate with an NFC tag or another NFC-enabled transmitting device. Enterprises that do bulk provisioning can use NFC-based device provisioning to provide a provisioning package to the device that's being provisioned. NFC provisioning is simple and convenient and it can easily store an entire provisioning package.

The NFC provisioning option enables the administrator to provide a provisioning package during initial device setup (the out-of-box experience or OOBE phase). Administrators can use the NFC provisioning option to transfer provisioning information to persistent storage by tapping an unprovisioned mobile device to an NFC tag or NFC-enabled device. To use NFC for pre-provisioning a device, you must either prepare your own NFC tags by storing your provisioning package to a tag as described in this section, or build the infrastructure needed to transmit a provisioning package between an NFC-enabled device and a mobile device during OOBE.

## Provisioning OOBE UI

All Windows 10 Mobile Enterprise and Windows 10 Mobile images have the NFC provisioning capability incorporated into the operating system. On devices that support NFC and are running Windows 10 Mobile Enterprise or Windows 10 Mobile, NFC-based device provisioning provides an additional mechanism to provision the device during OOBE.

On all Windows devices, device provisioning during OOBE can be triggered by 5 fast taps on the Windows hardware key, which shows the **Provision this device** screen. In the **Provision this device** screen, select **NFC** for NFC-based provisioning.



If there is an error during NFC provisioning, the device will show a message if any of the following errors occur:

- **NFC initialization error** - This can be caused by any error that occurs before data transfer has started. For example, if the NFC driver isn't enabled or there's an error communicating with the proximity API.
- **Interrupted download or incomplete package transfer** - This error can happen if the peer device is out of range or the transfer is aborted. This error can be caused whenever the device being provisioned fails to receive the provisioning package in time.
- **Incorrect package format** - This error can be caused by any protocol error that the operating system encounters during the data transfer between the devices.
- **NFC is disabled by policy** - Enterprises can use policies to disallow any NFC usage on the managed device. In this case, NFC functionality is not enabled.

## NFC tag

You can use an NFC tag for minimal provisioning and use an NFC-enabled device tag for larger provisioning packages.

The protocol used for NFC-based device provisioning is similar to the one used for NFC provisioning on Windows Embedded 8.1 Handheld, which supported both single-chunk and multi-chunk transfer when the total transfer didn't fit in one NDEP message size. In Windows 10, the provisioning stack contains the following changes:

- **Protocol namespace** - The protocol namespace has changed from Windows.WEH.PreStageProv.Chunk to Windows.ProvPlugins.Chunk.
- **Tag data type** - The tag data type has changed from UTF-8 into binary raw data.

### NOTE

The NFC tag doesn't go in the secondary device. You can transfer the NFC tag by using a provisioning package from device-to-device using the NFC radio or by re-reading the provisioning package from an NFC tag.

## NFC tag components

NFC tags are suitable for very light applications where minimal provisioning is required. The size of NFC tags that contain provisioning packages is typically 4 KB to 10 KB.

To write to an NFC tag, you will need to use an NFC Writer tool, or you can use the [ProximityDevice class API](#) to write your own custom tool to transfer your provisioning package file to your NFC tag. The tool must publish a binary message (write) a Chunk data type to your NFC tag.

The following table describes the information that is required when writing to an NFC tag.

REQUIRED FIELD	DESCRIPTION
Type	Windows.ProvPlugins.Chunk The receiving device uses this information to understand information in the Data field.
Data	Tag data with small header in raw binary format that contains a chunk of the provisioning package to be transferred.

## NFC provisioning helper

The NFC provisioning helper device must split the provisioning package raw content into multiple parts and publish these in order. Each part should follow the following format:

Version (1 byte)	Leading (1 byte)	Order (1 byte)	Total (1 byte)	Chunk payload (N bytes)
---------------------	---------------------	-------------------	-------------------	----------------------------

For each part:

- **Version** should always be 0x00.
- **Leading byte** should always be 0xFF.
- **Order** represents which message chunk (out of the whole message) the part belongs to. The Order begins with zero (0).
- **Total** represents the total number of chunks to be transferred for the whole message.
- **Chunk payload** represents each of the split parts.

The NFC provisioning helper device must publish the record in a type of Windows.ProvPlugins.Chunk.

### Code example

The following example shows how to write to an NFC tag. This example assumes that the tag is already in range of the writing device.

```
private async void WriteProvPkgToTag(IStorageFile provPkgFile)
{
 var buffer = await FileIO.ReadBufferAsync(provPkgFile);
 if (null == buffer)
 {
 return;
 }

 var proximityDevice = Windows.Networking.Proximity.ProximityDevice.GetDefault();
 if (null == proximityDevice)
 {
 return;
 }

 var dataWriter = new DataWriter();
 var header = new NfcProvHeader();

 header.version = NFC_PROV_MESSAGE_CURRENT_VERSION; // Currently the supported version is 0x00.
 header.leading = NFC_PROV_MESSAGE_LEADING_BYTE; // The leading byte should be always 0xFF.
 header.index = 0; // Assume we only have 1 chunk.
 header.total = 1; // Assume we only have 1 chunk.

 // Write the header first and then the raw data of the provisioning package.
 dataWriter.WriteBytes(GetBytes(header));
 dataWriter.WriteBuffer(buffer);

 var chunkPubId = proximityDevice.PublishBinaryMessage(
 "Windows:WriteTag.ProvPlugins.Chunk",
 dataWriter.DetachBuffer());
}
```

### NFC-enabled device tag components

Provisioning from an NFC-enabled source device allows for larger provisioning packages than can be transferred using an NFC tag. When provisioning from an NFC-enabled device, we recommend that the total file size not exceed 120 KB. Be aware that the larger the NFC file is, the longer it will take to transfer the provisioning file. Depending on your NFC hardware, the transfer time for a 120 KB file will vary between 2.5 seconds and 10 seconds.

To provision from an NFC-enabled source device, use [ProximityDevice class API](#) to write your own custom tool that transfers your provisioning package in chunks to your target mobile device. The tool must publish binary messages (transmit) a Header message, followed by one or more Chunk messages. The Header specifies the total amount of data that will be transferred to the target device; the Chunks must contain binary raw data formatted provisioning data, as shown in the NFC tag components section.

For detailed information and code samples on how to implement an NFC-enabled device tag, see **ConvertToNfcMessageAsync** in [this GitHub NfcProvisioner Universal Windows app example](#). The sample app shows you how to host the provisioning package on a master device so that you can transfer it to the receiving device.

## Related topics

- [Use Windows Configuration Designer to configure Windows 10 Mobile devices](#)
- [Barcode provisioning and the package splitter tool](#)

# Barcode provisioning and the package splitter tool

7/28/2017 • 2 min to read • [Edit Online](#)

## Applies to

- Windows 10 Mobile

Enterprises that do bulk provisioning can use barcode-based device provisioning to provide a provisioning package to the device that's being provisioned.

The barcode provisioning option enables the administrator to provide a provisioning package during initial device setup (the out-of-box experience or OOBE phase). To use barcodes to provision a device, your devices must have an integrated barcode scanner. You can get the barcode format that the scanner supports from your OEM or device provider, and use your existing tools and processes to convert a provisioning package into barcodes.

Enterprise IT professionals who want to use a barcode to provision mobile devices during OOBE can use the package splitter tool, **ppkgtobase64.exe**, which is a command-line tool to split the provisioning package into smaller files.

The smallest provisioning package is typically 5-6 KB, which cannot fit into one single barcode. The package splitter tool allows partners to split the original provisioning package into multiple smaller sized chunks that are encoded with Base64 so that enterprises can leverage their existing tools to convert these files into barcodes.

When you [install Windows Configuration Designer](#) from the Windows Assessment and Deployment Kit (ADK), **ppkgtobase64.exe** is installed to the same folder.

## Prerequisites

Before you can use the tool, you must have a built provisioning package. The package file is the input to the package splitter tool.

- To build a provisioning package using the Windows Configuration Designer UI, see [Use Windows Configuration Designer to configure Windows 10 Mobile devices](#).
- To build a provisioning package using the Windows Configuration Designer CLI, see [Windows Configuration Designer command-line interface](#).

## To use the package splitter tool (ppkgtobase64.exe)

- Open a command-line window with administrator privileges.
- From the command-line, navigate to the Windows Configuration Designer install directory.

On an x64 computer, type:

```
cd C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Imaging and Configuration Designer\x86
```

- or -

On an x86 computer, type:

```
cd C:\Program Files\Windows Kits\10\Assessment and Deployment Kit\Imaging and Configuration Designer\x86
```

3. Run `ppkgtobase64.exe`. The [syntax](#) and [switches and arguments](#) sections provide details for the command.

## Syntax

```
ppkgtobase64.exe -i <InputFile> -o <OutputDirectory> -s <BlockSize> [-c] [/?]
```

## Switches and arguments

SWITCH	REQUIRED?	ARGUMENTS
-i	Yes	Use to specify the path and file name of the provisioning package that you want to divide into smaller files. The tool allows you to specify the absolute path of the provisioning package file. However, if you don't specify the path, the tool will search the current folder for a package that matches the file name you specified.
-o	Yes	Use to specify the directory where the output files will be saved.
-s	Yes	Use to specify the size of the block that will be encoded in Base64.
-c	No	Use to delete any files in the output directory if the directory already exists. This parameter is optional.
/?	No	Lists the switches and their descriptions for the command-line tool or for certain commands.

## Related topics

# Use the Lockdown Designer app to create a Lockdown XML file

7/28/2017 • 7 min to read • [Edit Online](#)



Windows 10 Mobile allows enterprises to lock down a device, define multiple user roles, and configure custom layouts on a device. For example, the enterprise can lock down a device so that only applications and settings in an allow list are available. This is accomplished using Lockdown XML, an XML file that contains settings for Windows 10 Mobile.

When you deploy the lockdown XML file to a device, it is saved on the device as **wehlockdown.xml**. When the device boots, it looks for wehlockdown.xml and applies any settings configured in the file. You can deploy the lockdown XML file by [adding it to a provisioning package](#) or [by using mobile device management \(MDM\)](#).

The Lockdown Designer app helps you configure and create a lockdown XML file that you can apply to devices running Windows 10 Mobile, version 1703, and includes a remote simulation to help you determine the layout for tiles on the Start screen. Lockdown Designer also validates the XML. Using Lockdown Designer is easier than [manually creating a lockdown XML file](#).

## Overview

Lockdown Designer can be installed on a PC running Windows 10, version 1607 or later. After you install the app, you connect a mobile device running Windows 10 Mobile, version 1703, to the PC.

### NOTE

Lockdown Designer will not make any changes to the connected device, but we recommend that you use a test device.

Lockdown Designer will populate the available settings and apps to configure from the connected device. Using the different pages in the app, you select the settings, apps, and layout to be included in the lockdown XML.

When you're done, you export the configuration to a lockdown XML file. This configuration can be applied to any device running Windows 10 Mobile, version 1703.

### NOTE

You can also import an existing WEHLockdown.xml file to Lockdown Designer and modify it in the app.

## Prepare the test mobile device

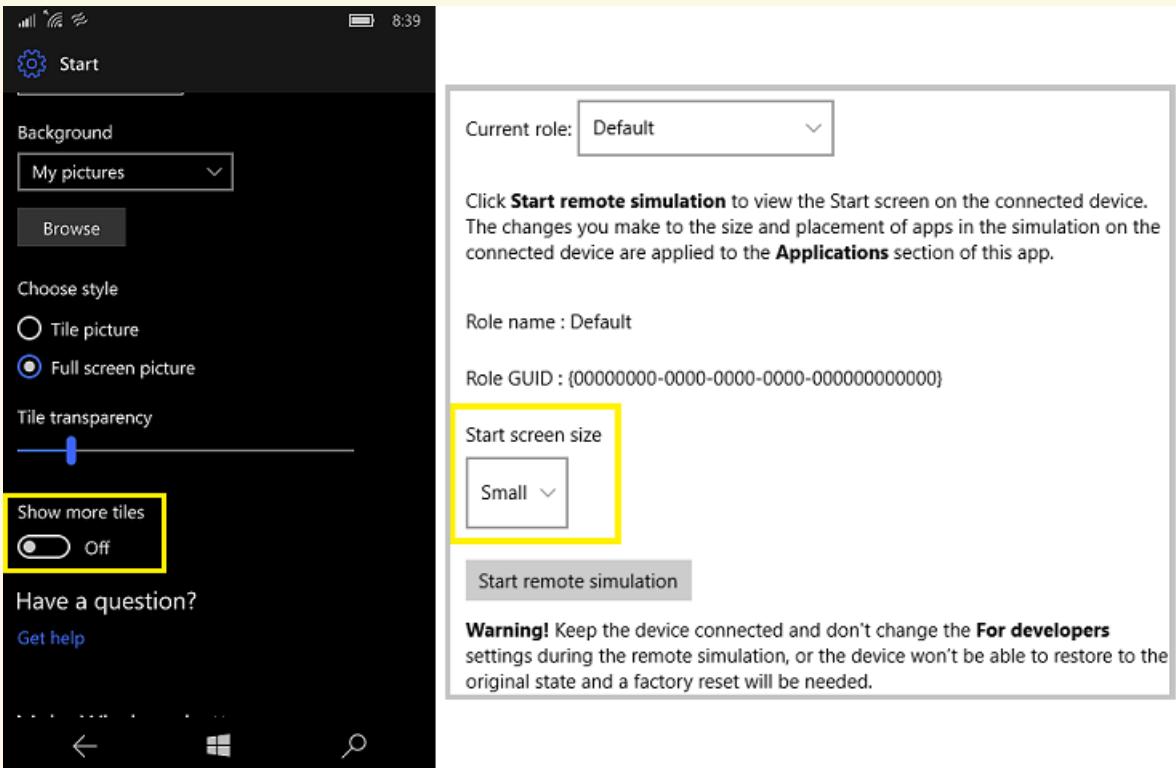
Perform these steps on the device running Windows 10 Mobile that you will use to supply the settings, apps, and layout to Lockdown Designer.

1. Install all apps on the device that you want to include in the configuration, including line-of-business apps.

2. On the mobile device, go to **Settings > Update & security > For developers**, enable **Developer mode**.
3. Read the disclaimer, then click **Yes** to accept the change.
4. Enable **Device discovery**, and then turn on **Device Portal**.

**IMPORTANT**

Check **Settings > Personalization > Start > Show more tiles** on the test mobile device. If **Show more tiles** is **On**, you must select **Large** on the **Start screen** page in Lockdown Designer. If you want to apply a **Small** layout, set **Show more tiles** on the test mobile device to **Off**.



## Prepare the PC

[Install Lockdown Designer](#) on the PC.

If the PC and the test mobile device are on the same Wi-Fi network, you can connect the devices using Wi-Fi.

If you want to connect the PC and the test mobile device using a USB cable, perform the following steps on the PC:

1. [Install the Windows 10 Software Development Kit \(SDK\)](#). This enables the **Windows Phone IP over USB Transport (IpOverUsbSvc)** service.
2. Open a command prompt as an administrator and run

```
checknetisolation LoopbackExempt -a -n=microsoft.lockdowndesigner_8wekyb3d8bbwe
```

**NOTE**

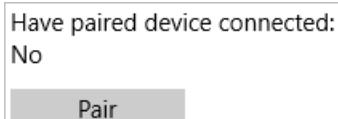
Loopback is permitted only for development purposes. To remove the loopback exemption when you're done using Lockdown Designer, run

```
checknetisolation LoopbackExempt -d -n=microsoft.lockdowndesigner_8wekyb3d8bbwe
```

## Connect the mobile device to Lockdown Designer

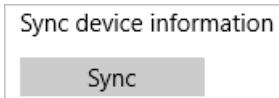
## Using Wi-Fi

1. Open Lockdown Designer.
2. Click **Create new project**.
3. On the test mobile device, go to **Settings > Update & security > For developers > Connect using:** and get the IP address listed for **Wi-Fi**.
4. On the **Project setting > General settings** page, in **Remote device IP address**, enter the IP address for the test mobile device, using `https://`.
5. Click **Pair**.



**Connect to remote device** appears.

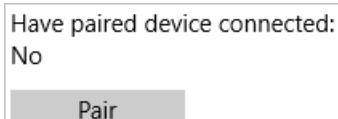
6. On the mobile device, under **Device discovery**, tap **Pair**. A case-sensitive code is displayed.
7. On the PC, in **Connect to remote device**, enter the code from the mobile device.
8. Next, click **Sync** to pull information from the device in to Lockdown Designer.



9. Click the **Save** icon and enter a name for your project.

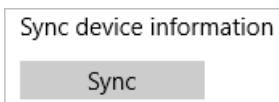
## Using a USB cable

1. Open Lockdown Designer.
2. Click **Create new project**.
3. Connect a Windows 10 Mobile device to the PC by USB and unlock the device.
4. On the **Project setting > General settings** page, click **Pair**.



**Connect to remote device** appears.

5. On the mobile device, under **Device discovery**, tap **Pair**. A case-sensitive code is displayed.
6. On the PC, in **Connect to remote device**, enter the code from the mobile device.
7. Next, click **Sync** to pull information from the device in to Lockdown Designer.



8. Click the **Save** icon and enter a name for your project.

## Configure your lockdown XML settings

The apps and settings available in the pages of Lockdown Designer should now be populated from the test mobile device. The following table describes what you can configure on each page.

PAGE	DESCRIPTION
Applications	Each app from the test mobile device is listed. Select the apps that you want visible to users. You can select an app to run automatically when a user signs in to the device. The <b>Select Auto-Run</b> menu is populated by the apps that you select to allow on the device.
CSPRunner	CSPRunner enables you to include settings and policies that are not defined in other sections of the app. To make use of CSPRunner, you must create the SyncML block that contains the settings, and then import the SyncML in Lockdown Designer. <a href="#">Learn how to use CSPRunner and author SyncML</a> .
Settings	On this page, you select the settings that you want visible to users. See the <a href="#">ms settings: URL scheme reference</a> to see which Settings page maps to a URI.
Quick actions	On this page, you select the settings that you want visible to users.
Buttons	Each hardware button on a mobile device has different actions that can be disabled. In addition, the behavior for <b>Search</b> button can be changed to open an app other than <b>Search</b> . Some devices may have additional hardware buttons provided by the OEM. These are listed as Custom1, Custom2, and Custom3. If your device has custom hardware buttons, contact your equipment provider to identify how their custom buttons are defined.
Other settings	This page contains several settings that you can configure: - The context menu is displayed when a user presses and holds an application in the All Apps list. You can enable or disable the context menu.  - Tile manipulation allows users to pin, unpin, move, and resize tiles on the Start screen. You can enable or disable tile manipulation.  - The Action Center setting controls whether the user can open the Action Center on the device. When the Action Center is disabled, notifications on the lockscreen and toasts are also disabled. You can use optional attributes with the Action Center element to change that behavior for either notifications, toasts, or both.

PAGE	DESCRIPTION
Start screen	<p>On this page, you can start a remote simulation session with the test mobile device. Click <b>Start remote simulation</b>. You will see a <b>Start screen remote simulation in progress</b> message on the PC. (If the <b>Start remote simulation</b> button is not active, <a href="#">pair the mobile device with the PC again</a>.) On the test mobile device, tiles for the apps that you allowed on the <b>Applications</b> page are displayed on the screen. You can move, resize, or unpin these tiles to achieve the desired layout.</p> <p>When you are done changing the layout on the test mobile device, click <b>Accept</b> on the PC.</p>

## Validate and export

On the **Validate and export** page, click **Validate** to make sure your lockdown XML is valid.

### WARNING

Lockdown Designer cannot validate SyncML that you imported to CSPRunner.

Click **Export** to generate the XML file for your project. You can select the location to save the file.

## Create and configure multiple roles

You can create additional roles for the device and have unique configurations for each role. For example, you could have one configuration for a **Manager** role and a different configuration for a **Salesperson** role.

### NOTE

Using multiple roles on a device requires a login application that displays the list of roles and allows users to sign in to Azure Active Directory. [Learn how to create a login application that will work with your Lockdown XML file](#).

### For each role:

1. On the **Project setting** page, click **Role management**.
2. Click **Add a role**.
3. Enter a name for the role, and then click **Save**.
4. Configure the settings for the role as above, but make sure on each page that you select the correct role.

Current role: Default

# Configure Windows 10 Mobile using Lockdown XML

7/28/2017 • 17 min to read • [Edit Online](#)

## Applies to

- Windows 10 Mobile

Windows 10 Mobile allows enterprises to lock down a device, define multiple user roles, and configure custom layouts on a device. For example, the enterprise can lock down a device so that only applications and settings in an allow list are available.

This is accomplished using Lockdown XML, an XML file that contains settings for Windows 10 Mobile. When you deploy the lockdown XML file to a device, it is saved on the device as **wehlockdown.xml**. When the device boots, it looks for wehlockdown.xml and applies any settings configured in the file.

In this topic, you'll learn how to create an XML file that contains all lockdown entries available in the AssignedAccessXml area of the [EnterpriseAssignedAccess configuration service provider \(CSP\)](#). This topic provides example XML that you can use in your own lockdown XML file that can be included in a provisioning package or when using a mobile device management (MDM) solution to push lockdown settings to enrolled devices. You can also use the [Lockdown Designer app](#) to configure and export your lockdown XML file.

### NOTE

On Windows 10 desktop editions, *assigned access* is a feature that lets you configure the device to run a single app above the lockscreen ([kiosk mode](#)). On a Windows 10 Mobile device, assigned access refers to the lockdown settings in AssignedAccessXml in the [EnterpriseAssignedAccess configuration service provider \(CSP\)](#).

If you're not familiar with CSPs, read [Introduction to configuration service providers \(CSPs\)](#) first.

## Overview of the lockdown XML file

Let's start by looking at the basic structure of the lockdown XML file. You can start your file by pasting the following XML (or any other examples in this topic) into a text or XML editor, and saving the file as *filename.xml*.

```
<?xml version="1.0" encoding="utf-8"?>
<HandheldLockdown version="1.0" >
 <Default>
 <ActionCenter/>
 <Apps/>
 <Buttons/>
 <CSPRunner/>
 <MenuItems/>
 <Settings/>
 <Tiles/>
 <StartScreenSize/>
 </Default>
</HandheldLockdown>
```

**Default** and the entries beneath it establish the default device settings that are applied for every user. The device will always boot to this Default role. You can create additional roles on the device, each with its own settings, in the same XML file. [Learn how to add roles](#).

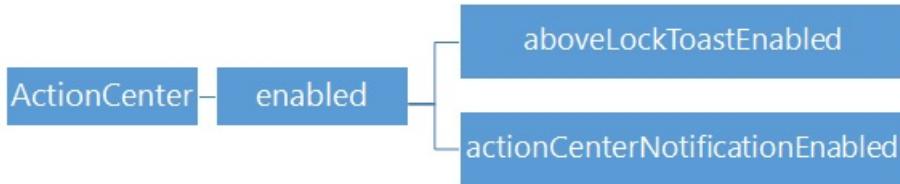
The settings for the Default role and other roles must be listed in your XML file in the order presented in this topic.

All of the entries are optional. If you don't include a setting, that aspect of the device will operate as it would for an nonconfigured device.

#### TIP

Keep your XML file easy to work with and to understand by using proper indentation and adding comments for each setting you configure.

## Action Center



The Action Center setting controls whether the user can open the Action Center on the device. When the Action Center is disabled, notifications on the lockscreen and toasts are also disabled. You can use optional attributes with the Action Center element to change that behavior for either notifications, toasts, or both.

In the following example, the Action Center is enabled and both policies are disabled.

```
<ActionCenter enabled="true" aboveLockToastEnabled="0" actionCenterNotificationEnabled="0"/>
```

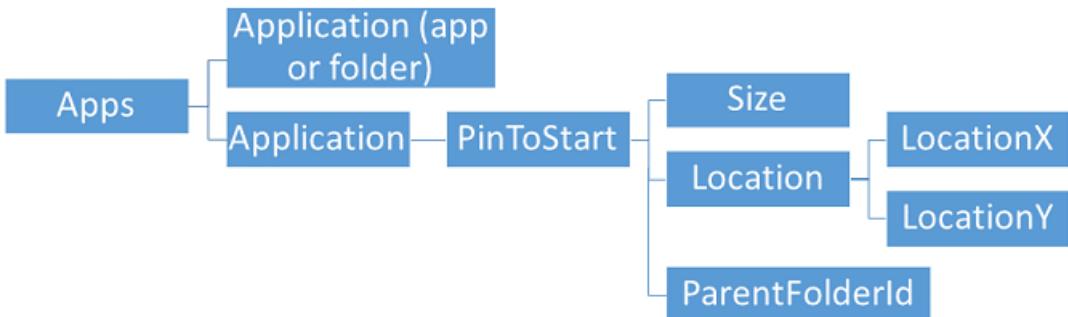
In the following example, Action Center and the toast policy are enabled, and the notifications policy is disabled.

```
<ActionCenter enabled="true" aboveLockToastEnabled="1" actionCenterNotificationEnabled="0"/>
```

The following example is a complete lockdown XML file that disables Action Center, notifications, and toasts.

```
<?xml version="1.0" encoding="utf-8"?>
<HandheldLockdown version="1.0" >
 <Default>
 <!-- disable Action Center -->
 <ActionCenter enabled="false" />
 </Default>
</HandheldLockdown>
```

## Apps



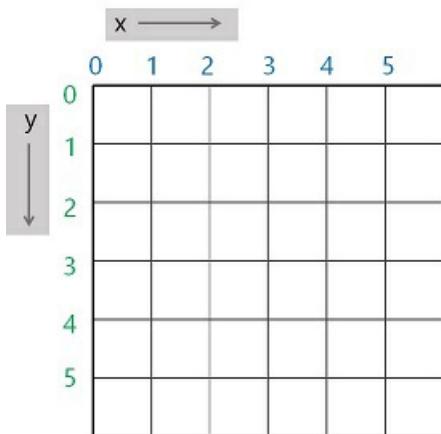
The Apps setting serves as an allow list and specifies the applications that will be available in the All apps list. Apps that are not included in this setting are hidden from the user and blocked from running.

You provide the App User Model ID (AUMID) and product ID for each app in your file. The product ID identifies an app package, and an app package can contain multiple apps, so you also provide the ADUMID to differentiate the app. Optionally, you can set an app to run automatically. [Get product ID and AUMID for apps in Windows 10 Mobile](#).

The following example makes Outlook Calendar available on the device.

```
<Apps>
 <!-- Outlook Calendar -->
 <Application productId="{A558FEBA-85D7-4665-B5D8-A2FF9C19799B}"
 aumid="microsoft.windowscommunicationsapps_8wekyb3d8bbwe!microsoft.windowslive.calendar">
 </Application>
</Apps>
```

When you list an app, you can also set the app to be pinned to the Start screen by specifying the tile size and location. Tip: draw a grid and mark your app tiles on it to make sure you get the result you want. The width (X axis) in the following example is the limit for Windows 10 Mobile, but the length (Y axis) is unlimited. The number of columns available to you depends on the value for [StartScreenSize](#).



Tile sizes are:

- Small: 1x1
- Medium: 2x2
- Large: 2x4

Based on 6 columns, you can pin six small tiles or three medium tiles on a single row. A large tile can be combined with two small tiles or one medium tile on the same row. Obviously, you cannot set a medium tile for LocationX=5, or a large tile for LocationX=3, 4, or 5.

If the tile configuration in your file exceeds the available width, such as setting a large tile to start at position 3 on the X axis, that tile is appended to the bottom of the Start screen. Also, if the tile configuration in your file would result in tiles overlapping each other, the overlapping tiles are instead appended to the bottom of the Start screen.

In the following example, Outlook Calendar and Outlook Mail are pinned to the Start screen, and the Store app is allowed but is not pinned to Start.

```

<Apps>
 <!-- Outlook Calendar -->
 <Application productId="{A558FEBA-85D7-4665-B5D8-A2FF9C19799B}"
aumid="microsoft.windowscommunicationsapps_8wekyb3d8bbwe!microsoft.windowslive.calendar">
 <PinToStart>
 <Size>Large</Size>
 <Location>
 <LocationX>0</LocationX>
 <LocationY>0</LocationY>
 </Location>
 </PinToStart>
 </Application>
 <!-- Outlook Mail-->
 <Application productId="{A558FEBA-85D7-4665-B5D8-A2FF9C19799B}"
aumid="microsoft.windowscommunicationsapps_8wekyb3d8bbwe!microsoft.windowslive.mail">
 <PinToStart>
 <Size>Medium</Size>
 <Location>
 <LocationX>4</LocationX>
 <LocationY>0</LocationY>
 </Location>
 </PinToStart>
 </Application>
 <!-- Store -->
 <Application productId="7D47D89A-7900-47C5-93F2-46EB6D94C159"
aumid="Microsoft.WindowsStore_8wekyb3d8bbwe!App" />
</Apps>

```

That layout would appear on a device like this:



You can create and pin folders to Start by using the Apps setting. Each folder requires a **folderId**, which must be a consecutive positive integer starting with **1**. You can also specify a **folderName** (optional) which will be displayed on Start.

```

<Apps>
 <!-- Management folder -->
 <Application folderId="1" folderName="Management">
 <PinToStart>
 <Size>Medium</Size>
 <Location>
 <LocationX>4</LocationX>
 <LocationY>0</LocationY>
 </Location>
 </PinToStart>
 </Application>
</Apps>

```

To add apps to the folder, include **ParentFolderId** in the application XML, as shown in the following example:

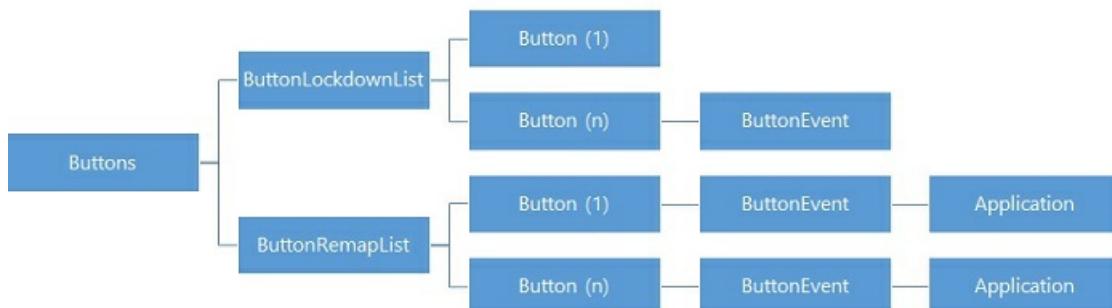
```

<Apps>
 <!-- Outlook Calendar -->
 <Application productId="{A558FEBA-85D7-4665-B5D8-A2FF9C19799B}"
 aumid="microsoft.windowscommunicationsapps_8wekyb3d8bbwe!microsoft.windowslive.calendar">
 <PinToStart>
 <Size>Large</Size>
 <Location>
 <LocationX>0</LocationX>
 <LocationY>0</LocationY>
 </Location>
 <ParentFolderId>1</ParentFolderId>
 </PinToStart>
 </Application>
 <!-- Outlook Mail-->
 <Application productId="{A558FEBA-85D7-4665-B5D8-A2FF9C19799B}"
 aumid="microsoft.windowscommunicationsapps_8wekyb3d8bbwe!microsoft.windowslive.mail">
 <PinToStart>
 <Size>Medium</Size>
 <Location>
 <LocationX>4</LocationX>
 <LocationY>0</LocationY>
 </Location>
 <ParentFolderId>1</ParentFolderId>
 </PinToStart>
 </Application>
</Apps>

```

When an app is contained in a folder, its **PinToStart** configuration (tile size and location) applies to its appearance when the folder is opened.

## Buttons



In the Buttons setting, you use ButtonLockdownList to disable hardware buttons and ButtonRemapList to change button events to open an app that you specify.

### ButtonLockdownList

When a user taps a button that is in the lockdown list, nothing will happen. The following table lists which events can be disabled for each button.

BUTTON	PRESS	PRESSANDHOLD	ALL
Start	✗	✓	✗
Back	✓	✓	✓
Search	✓	✓	✓

BUTTON	PRESS	PRESSANDHOLD	ALL
Camera	✓	✓	✓
Custom 1, 2, and 3	✓	✓	✓

#### NOTE

Custom buttons are hardware buttons that can be added to devices by OEMs.

In the following example, press-and-hold is disabled for the Back button.

```
<Buttons>
 <ButtonLockdownList>
 <Button name="Back">
 <ButtonEvent name="PressAndHold" />
 </Button>
 </ButtonLockdownList>
</Buttons>
```

If you don't specify a button event, all actions for the button are disabled. In the next example, all actions are disabled for the camera button.

```
<Buttons>
 <ButtonLockdownList>
 <Button name="Camera">
 </Button>
 </ButtonLockdownList>
</Buttons>
```

#### ButtonRemapList

ButtonRemapList lets you change the app that a button will run. You can remap the Search button and any custom buttons included by the OEM. You can't remap the Back, Start, or Camera buttons.

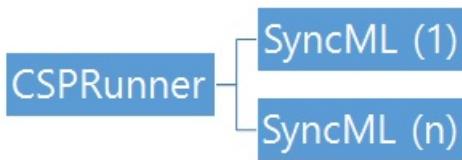
#### WARNING

Button remapping can enable a user to open an application that is not in the allow list for that user role. Use button lock down to prevent application access for a user role.

To remap a button, you specify the button, the event, and the product ID for the app that you want the event to open. In the following example, when a user presses the Search button, the phone dialer will open instead of the Search app.

```
<Buttons>
 <ButtonRemapList>
 <Button name="Search">
 <ButtonEvent name="Press">
 <!-- Phone dialer -->
 <Application productID="{F41B5D0E-EE94-4F47-9CFE-3D3934C5A2C7}" parameters="" />
 </ButtonEvent>
 </Button>
 </ButtonRemapList>
</Buttons>
```

# CSPRunner



You can use CSPRunner to include settings that are not defined in AssignedAccessXML. For example, you can include settings from other sections of EnterpriseAssignedAccess CSP, such as lockscreen, theme, and time zone. You can also include settings from other CSPs, such as [Wi-Fi CSP](#) or [Policy CSP](#).

CSPRunner is helpful when you are configuring a device to support multiple roles. It lets you apply different policies according to the role that is signed on. For example, Wi-Fi could be enabled for a supervisor role and disabled for a stocking clerk role.

In CSPRunner, you specify the CSP and settings using SyncML, a standardized markup language for device management. A SyncML section can include multiple settings, or you can use multiple SyncML sections -- it's up to you how you want to organize settings in this section.

## NOTE

This description of SyncML is just the information that you need to use SyncML in a lockdown XML file. To learn more about SyncML, see [Structure of OMA DM provisioning files](#).

Let's start with the structure of SyncML in the following example:

```
SyncML>
 <SyncBody>
 <Add>|<Replace>
 <CmdID>#</CmdID>
 <Item>
 <Target>
 <LocURI>CSP Path</LocURI>
 </Target>
 <Meta>
 <Format xmlns="syncml:metinf">Data Type</Format>
 </Meta>
 <Data>Value</Data>
 </Item>
 </Add>|<Replace>
 <Final/>
 </SyncBody>
</SyncML>
```

This table explains the parts of the SyncML structure.

SYNCML ENTRY	DESCRIPTION
<b>Add or Replace</b>	Use <b>Add</b> to apply a setting or policy that is not already configured. Use <b>Replace</b> to change an existing setting or policy.
<b>CmdID</b>	SyncBody can contain multiple commands. Each command in a lockdown XML file must have a different <b>CmdID</b> value.

SYNCML ENTRY	DESCRIPTION
<b>Item</b>	<b>Item</b> is a wrapper for a single setting. You can include multiple items for the command if they all use the same <b>Add</b> or <b>Replace</b> operation.
<b>Target &gt; LocURI</b>	<b>LocURI</b> is the path to the CSP.
<b>Meta &gt; Format</b>	The data format required by the CSP.
<b>Data</b>	The value for the setting.

## Menu items



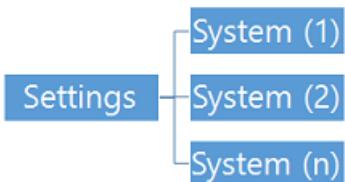
Use **DisableMenuItems** to prevent use of the context menu, which is displayed when a user presses and holds an application in the All Apps list. You can include this entry in the default profile and in any additional user role profiles that you create.

```

<MenuItems>
 <DisableMenuItems/>
</MenuItems>

```

## Settings



The **Settings** section contains an **allow** list of pages in the Settings app and quick actions. The following example allows all settings.

```

<Settings>
 <!-- Allow all settings -->
</Settings>

```

In earlier versions of Windows 10, you used the page name to define allowed settings. Starting in Windows 10, version 1703, you use the settings URI.

In the following example for Windows 10, version 1703, all system setting pages that have a settings URI are enabled.

```
<Settings>
<System name="ms-settings:screenrotation" />
<System name="ms-settings:notifications" />
<System name="ms-settings:phone" />
<System name="ms-settings:messaging" />
<System name="ms-settings:batterysaver" />
<System name="ms-settings:batterysaver-usagedetails" />
<System name="ms-settings:about" />
<System name="ms-settings:deviceencryption" />
<System name="ms-settings:maps" />
</Settings>
```

If you list a setting or quick action in **Settings**, all settings and quick actions that are not listed are blocked. To remove access to all of the settings in the system, do not include the settings application in **Apps**.

For a list of the settings and quick actions that you can allow or block, see [Settings and quick actions that can be locked down in Windows 10 Mobile](#).

## Tiles

Tiles

— EnableTileManipulation

By default, under Assigned Access, tile manipulation is turned off (blocked) and only available if enabled in the user's profile. If tile manipulation is enabled in the user's profile, they can pin/unpin, move, and resize tiles based on their preferences. When multiple people use one device and you want to enable tile manipulation for multiple users, you must enable it for each user in their user profile.

### IMPORTANT

If a device is turned off then back on, the tiles reset to their predefined layout. If a device has only one profile, the only way to reset the tiles is to turn off then turn on the device. If a device has multiple profiles, the device resets the tiles to the predefined layout based on the logged-in user's profile.

```
<Tiles>
 <EnableTileManipulation/>
</Tiles>
```

## Start screen size

Specify the size of the Start screen. In addition to 4/6 columns, you can also use 4/6/8 depending on screen resolutions. Valid values:

- Small sets the width to 4 columns on devices with short axis (less than 400epx) or 6 columns on devices with short axis (greater than or equal to 400epx).
- Large sets the width to 6 columns on devices with short axis (less than 400epx) or 8 columns on devices with short axis (greater than or equal to 400epx).

If you have existing lockdown xml, you must update start screen size if your device has  $\geq 400\text{epx}$  on its short axis so that tiles on Start can fill all 8 columns if you want to use all 8 columns instead of 6, or use 6 columns instead of 4.

[Learn about effective pixel width \(epx\) for different device size classes.](#)

## Configure additional roles

You can add custom configurations by role. In addition to the role configuration, you must also install a login application on the device. The app displays a list of available roles on the device; the user taps a role, such as "Manager"; the configuration defined for the "Manager" role is applied.

Learn how to create a login application that will work with your [Lockdown XML file](#). For reference, see the [Windows.Embedded.DeviceLockdown API](#).

In the XML file, you define each role with a GUID and name, as shown in the following example:

```
<Role guid="{7bb62e8c-81ba-463c-b691-74af68230b42}" name="Manager">
```

You can create a GUID using a GUID generator -- free tools are available online. The GUID needs to be unique within this XML file.

You can configure the same settings for each role as you did for the default role, except Start screen size which can only be configured for the default role. If you use CSPRunner with roles, be aware that the last CSP setting applied will be retained across roles unless explicitly changed in each role configuration. CSP settings applied by CSPRunner may conflict with settings applied by MDM.

```
<?xml version "1.0" encoding "utf-8"?>
<HandheldLockdown version "1.0" >
 <Default>
 <ActionCenter/>
 <Apps/>
 <Buttons/>
 <CSPRunner/>
 <MenuItems/>
 <Settings/>
 <Tiles/>
 <StartScreenSize/>
 </Default>
 <RoleList>
 <Role>
 <ActionCenter/>
 <Apps/>
 <Buttons/>
 <CSPRunner/>
 <MenuItems/>
 <Settings/>
 <Tiles/>
 </Role>
 </RoleList>
</HandheldLockdown>
```

## Validate your XML

You can validate your lockdown XML file against the [EnterpriseAssignedAccess XSD](#).

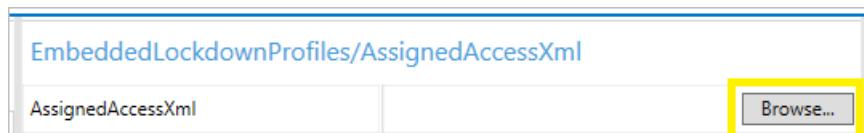
## Add lockdown XML to a provisioning package

Use the Windows ICD tool included in the Windows Assessment and Deployment Kit (ADK) for Windows 10 to create a provisioning package. [Install the ADK](#).

1. Follow the instructions at [Build and apply a provisioning package](#) to create a project, selecting **Common to all Windows mobile editions** for your project.
2. In **Available customizations**, go to **Runtime settings > EmbeddedLockdownProfiles >**

## **AssignedAccessXml**

3. In the center pane, click **Browse** to locate and select the lockdown XML file that you created.



4. On the **File** menu, select **Save**.
5. On the **Export** menu, select **Provisioning package**.
6. Change **Owner** to **IT Admin**, which will set the precedence of this provisioning package higher than provisioning packages applied to this device from other sources, and then select **Next**.
7. Optional. In the **Provisioning package security** window, you can choose to encrypt the package and enable package signing.
  - **Enable package encryption** - If you select this option, an auto-generated password will be shown on the screen.
  - **Enable package signing** - If you select this option, you must select a valid certificate to use for signing the package. You can specify the certificate by clicking **Select** and choosing the certificate you want to use to sign the package.
8. Click **Next** to specify the output location where you want the provisioning package to go when it's built. By default, Windows ICD uses the project folder as the output location.

Optionally, you can click **Browse** to change the default output location.
9. Click **Next**.
10. Click **Build** to start building the package. The provisioning package doesn't take long to build. The project information is displayed in the build page and the progress bar indicates the build status.

If you need to cancel the build, click **Cancel**. This cancels the current build process, closes the wizard, and takes you back to the **Customizations Page**.
11. If your build fails, an error message will show up that includes a link to the project folder. You can scan the logs to determine what caused the error. Once you fix the issue, try building the package again.

If your build is successful, the name of the provisioning package, output directory, and project directory will be shown.

  - If you choose, you can build the provisioning package again and pick a different path for the output package. To do this, click **Back** to change the output package name and path, and then click **Next** to start another build.
  - If you are done, click **Finish** to close the wizard and go back to the **Customizations Page**.

After you build the provisioning package, follow the instructions for [applying a provisioning package at runtime to Windows 10 Mobile](#).

## Push lockdown XML using MDM

After you deploy your devices, you can still configure lockdown settings through your MDM solution if it supports the [EnterpriseAssignedAccess CSP](#).

To push lockdown settings to enrolled devices, use the `AssignedAccessXML` setting and use the lockdown XML as the value. The lockdown XML will be in a `HandheldLockdown` section that becomes XML embedded in XML, so the

XML that you enter must use escaped characters (such as &lt; in place of <). After the MDM provider pushes your lockdown settings to the device, the CSP processes the file and updates the device.

## Full Lockdown.xml example

```
<?xml version="1.0" encoding="utf-8"?>
<HandheldLockdown version="1.0" >
 <Default>
 <ActionCenter enabled="true" />
 <Apps>
 <!-- Settings -->
 <Application productId="{2A4E62D8-8809-4787-89F8-69D0F01654FB}">
 <PinToStart>
 <Size>Large</Size>
 <Location>
 <LocationX>0</LocationX>
 <LocationY>0</LocationY>
 </Location>
 </PinToStart>
 </Application>
 <!-- Outlook Calendar -->
 <Application productId="{A558FEBA-85D7-4665-B5D8-A2FF9C19799B}">
 aumid="microsoft.windowscommunicationsapps_8wekyb3d8bbwe!microsoft.windowslive.calendar">
 <PinToStart>
 <Size>Small</Size>
 <Location>
 <LocationX>0</LocationX>
 <LocationY>2</LocationY>
 </Location>
 </PinToStart>
 </Application>
 <!-- Photos -->
 <Application productId="{FCA55E1B-B9A4-4289-882F-084EF4145005}">
 <PinToStart>
 <Size>Medium</Size>
 <Location>
 <LocationX>2</LocationX>
 <LocationY>2</LocationY>
 </Location>
 </PinToStart>
 </Application>
 <!-- Edge -->
 <Application productId="{395589FB-5884-4709-B9DF-F7D558663FFD}" />
 <!-- Login App -->
 <Application productId="{C85DC60D-30D4-4C67-A4B4-58282F1D152C}" />
 </Apps>
 <Buttons>
 <ButtonLockdownList>
 <!-- Lockdown all buttons -->
 <Button name="Search">
 </Button>
 <Button name="Camera">
 </Button>
 <Button name="Custom1">
 </Button>
 <Button name="Custom2">
 </Button>
 <Button name="Custom3">
 </Button>
 </ButtonLockdownList>
 <ButtonRemapList>
 <Button name="Search">
 <ButtonEvent name="Press">
 <!-- Edge-->
 <Application productId="{395589FB-5884-4709-B9DF-F7D558663FFD}" parameters="" />
 </ButtonEvent>
 </Button>
 </ButtonRemapList>
 </Buttons>
 </Default>
</HandheldLockdown>
```

```
 </Target>
 </Replace>
 <Item>
 <Target>

<LocURI>./Vendor/MSFT/EnterpriseAssignedAccess/Theme/ThemeAccentColorID</LocURI>
 </Target>
 <Meta>
 <Format xmlns="syncml:metinf">int</Format>
 </Meta>
 <!-- zero based index of available theme colors -->
 <Data>7</Data>
 </Item>
</Replace>
<Final/>
</SyncBody>
</SyncML>
<SyncML xmlns="SYNCML:SYNCML1.2">
 <SyncBody>
 <Replace>
 <CmdID>1</CmdID>
 <Item>
 <Target>
 <LocURI>./Vendor/MSFT/EnterpriseAssignedAccess/Theme/ThemeBackground</LocURI>
 </Target>
 <Meta>
 <Format xmlns="syncml:metinf">int</Format>
 </Meta>
 <!-- 0 for "light", 1 for "dark" -->
 <Data>1</Data>
 </Item>
 </Replace>
 <Final/>
 </SyncBody>
 </SyncML>
 <SyncML xmlns="SYNCML:SYNCML1.2">
 <SyncBody>
 <Replace>
 <CmdID>2</CmdID>
 <Item>
 <Target>

<LocURI>./Vendor/MSFT/EnterpriseAssignedAccess/LockScreenWallpaper/BGFileName</LocURI>
 </Target>
 <Meta>
 <Format xmlns="syncml:metinf">chr</Format>
 <Type xmlns="syncml:metinf">text/plain</Type>
 </Meta>
 <Data>c:\windows\system32\lockscreen\480x800\Wallpaper_05.jpg</Data>
 </Item>
 </Replace>
 <Final/>
 </SyncBody>
 </SyncML>
</CSPRunner>
<MenuItems>
 <DisableMenuItems/>
</MenuItems>
<Settings>
 <!-- Quick actions: Brightness, Rotation -->
 <System name="SystemSettings_System_Display_QuickAction_Brightness"/>
 <System name="SystemSettings_System_Display_Internal_Rotation"/>
 <!-- Rotation, About -->
 <System name="SystemSettings_System_Display_Rotation"/>
 <System name="SystemSettings_System_Display_About"/>
</Settings>
```

```

<System name="ms-settings:screenrotation"/>
<System name="ms-settings:about"/>
<!-- Ringtones, sounds -->
<System name="ms-settings:personalizationn"/>
<System name="ms-settings:sounds"/>
</Settings>
<Tiles>
 <EnableTileManipulation/>
</Tiles>
<StartScreenSize>Small</StartScreenSize>
</Default>
<RoleList>
 <Role guid="{88501844-3b51-4c9f-9da7-7ca745e7da6b}" name="Associate">
 <ActionCenter enabled="0"/>
 <Apps>
 <!-- Settings -->
 <Application productId="{2A4E62D8-8809-4787-89F8-69D0F01654FB}">
 <PinToStart>
 <Size>Small</Size>
 <Location>
 <LocationX>0</LocationX>
 <LocationY>0</LocationY>
 </Location>
 </PinToStart>
 </Application>
 <!-- Outlook Calendar -->
 <Application productId="{A558FEBA-85D7-4665-B5D8-A2FF9C19799B}">
 <PinToStart>
 <Size>Large</Size>
 <Location>
 <LocationX>0</LocationX>
 <LocationY>2</LocationY>
 </Location>
 </PinToStart>
 </Application>
 <!-- Login App -->
 <Application productId="{C85DC60D-30D4-4C67-A4B4-58282F1D152C}" />
 </Apps>
 <Buttons />
 <CSPRunner>
 <SyncML xmlns="SYNCML:SYNCML1.2">
 <SyncBody>
 <Replace>
 <CmdID>1</CmdID>
 <Item>
 <Target>
 <LocURI>./Vendor/MSFT/EnterpriseAssignedAccess/Theme/ThemeAccentColorID</LocURI>
 <Target>
 <Meta>
 <Format xmlns="syncml:metinf">int</Format>
 </Meta>
 <!-- zero based index of available theme colors -->
 <Data>10</Data>
 </Target>
 <Item>
 <Replace>
 <Final/>
 </Replace>
 </Item>
 </Target>
 </Item>
 </Replace>
 <SyncBody>
 </SyncML>
 <SyncML xmlns="SYNCML:SYNCML1.2">
 <SyncBody>
 <Replace>
 <CmdID>1</CmdID>
 <Item>
 <Target>
 <LocURI>./Vendor/MSFT/EnterpriseAssignedAccess/Theme/ThemeBackground</LocURI>
 <Target>
</pre>

```

```

 <Meta>
 <Format xmlns="syncml:metinf">int</Format>
 </Meta>
 <!-- 0 for "light", 1 for "dark" -->
 <Data>0</Data>
 </Item>
</Replace>
<Final/>
</SyncBody>
</SyncML>
<SyncML xmlns="SYNCML:SYNCML1.2">
 <SyncBody>
 <Replace>
 <CmdID>2</CmdID>
 <Item>
 <Target>

<LocURI>./Vendor/MSFT/EnterpriseAssignedAccess/LockScreenWallpaper/BGFileName</LocURI>
 </Target>
 <Meta>
 <Format xmlns="syncml:metinf">chr</Format>
 <Type xmlns="syncml:metinf">text/plain</Type>
 </Meta>
 <Data>c:\windows\system32\lockscreen\480x800\Wallpaper_08.jpg</Data>
 </Item>
 </Replace>
 <Final/>
 </SyncBody>
</SyncML>
</CSPRunner>
<MenuItems>
 <DisableMenuItems/>
</MenuItems>
<Settings>
 <!-- Rotation, Notifications, About -->
 <System name="ms-settings:screenrotation"/>
 <System name="ms-settings:notifications"/>
 <System name="ms-settings:about"/>
 <!-- Ringtones, sounds -->
 <System name="ms-settings:personalization"/>
 <System name="ms-settings:sounds"/>
 <!-- Workplace -->
 <System name="ms-settings:workplace"/>
 <System name="ms-settings:emailandaccounts"/>
</Settings>
</Role>
<Role guid="{7bb62e8c-81ba-463c-b691-74af68230b42}" name="Manager">
 <ActionCenter enabled="true" />
 <Apps>
 <!-- Alarms and Clock -->
 <Application productId="{44F7D2B4-553D-4BEC-A8B7-634CE897ED5F}">
 <PinToStart>
 <Size>Small</Size>
 <Location>
 <LocationX>0</LocationX>
 <LocationY>0</LocationY>
 </Location>
 </PinToStart>
 </Application>
 <!-- Settings -->
 <Application productId="{2A4E62D8-8809-4787-89F8-69D0F01654FB}">
 <PinToStart>
 <Size>Small</Size>
 <Location>
 <LocationX>1</LocationX>
 <LocationY>0</LocationY>
 </Location>
 </PinToStart>
 </Application>
 </Apps>
</Role>

```

```

<!-- Outlook Calendar -->
<Application productId="{A558FEBA-85D7-4665-B5D8-A2FF9C19799B}">
aumid="microsoft.windowscommunicationsapps_8wekyb3d8bbwe!microsoft.windowslive.calendar">
 <PinToStart>
 <Size>Medium</Size>
 <Location>
 <LocationX>2</LocationX>
 <LocationY>0</LocationY>
 </Location>
 </PinToStart>
</Application>
<!-- Calculator -->
<Application productId="{B58171C6-C70C-4266-A2E8-8F9C994F4456}" />
<!-- Photos -->
<Application productId="{FCA55E1B-B9A4-4289-882F-084EF4145005}">
 <PinToStart>
 <Size>Small</Size>
 <Location>
 <LocationX>0</LocationX>
 <LocationY>2</LocationY>
 </Location>
 </PinToStart>
</Application>
<!-- Store -->
<Application productId="{7D47D89A-7900-47C5-93F2-46EB6D94C159}">
 <PinToStart>
 <Size>Medium</Size>
 <Location>
 <LocationX>2</LocationX>
 <LocationY>2</LocationY>
 </Location>
 </PinToStart>
</Application>
<!-- Login App -->
<Application productId="{C85DC60D-30D4-4C67-A4B4-58282F1D152C}" />
</Apps>
<Buttons />
<CSPRunner>
 <SyncML xmlns="SYNCML:SYNCML1.2">
 <SyncBody>
 <Replace>
 <CmdID>1</CmdID>
 <Item>
 <Target>
<LocURI>./Vendor/MSFT/EnterpriseAssignedAccess/Theme/ThemeAccentColorID</LocURI>
 </Target>
 <Meta>
 <Format xmlns="syncml:metinf">int</Format>
 </Meta>
 <!-- zero based index of available theme colors -->
 <Data>2</Data>
 </Item>
 </Replace>
 <Final/>
 </SyncBody>
</SyncML>
<SyncML xmlns="SYNCML:SYNCML1.2">
 <SyncBody>
 <Replace>
 <CmdID>1</CmdID>
 <Item>
 <Target>
<LocURI>./Vendor/MSFT/EnterpriseAssignedAccess/Theme/ThemeBackground</LocURI>
 </Target>
 <Meta>
 <Format xmlns="syncml:metinf">int</Format>
 </Meta>

```

```
 <!-- 0 for "light", 1 for "dark" -->
 <Data>1</Data>
 </Item>
</Replace>
<Final/>
</SyncBody>
</SyncML>
<SyncML xmlns="SYNCML:SYNCML1.2">
 <SyncBody>
 <Replace>
 <CmdID>2</CmdID>
 <Item>
 <Target>

<LocURI>./Vendor/MSFT/EnterpriseAssignedAccess/LockScreenWallpaper/BGFileName</LocURI>
 </Target>
 <Meta>
 <Format xmlns="syncml:metinf">chr</Format>
 <Type xmlns="syncml:metinf">text/plain</Type>
 </Meta>
 <Data>c:\windows\system32\lockscreen\480x800\Wallpaper_015.jpg</Data>
 </Item>
 </Replace>
 <Final/>
 </SyncBody>
</SyncML>
</CSPRunner>
<MenuItems>
 <DisableMenuItems/>
</MenuItems>
<Settings>
 <!-- Allow all settings -->
</Settings>
<Tiles>
 <EnableTileManipulation/>
</Tiles>
</Role>
</RoleList>
</HandheldLockdown>
```

## Learn more

[Customizing Your Device Experience with Assigned Access](#)

## Related topics

[Settings and quick actions that can be locked down in Windows 10 Mobile](#)

[Product IDs in Windows 10 Mobile](#)

# Settings and quick actions that can be locked down in Windows 10 Mobile

7/28/2017 • 2 min to read • [Edit Online](#)

## Applies to

- Windows 10 Mobile

This topic lists the settings and quick actions that can be locked down in Windows 10 Mobile.

## Settings lockdown in Windows 10, version 1703

In earlier versions of Windows 10, you used the page name to define allowed settings. Starting in Windows 10, version 1703, you use the settings URI.

For example, in place of **SettingsPageDisplay**, you would use **ms-settings:display**.

See the [ms-settings: URI scheme reference](#) to find the URI for each Settings page.

## Settings lockdown in Windows 10, version 1607 and earlier

You can use Lockdown.xml to configure lockdown settings.

The following table lists the settings pages and page groups. Use the page name in the Settings section of Lockdown.xml. The Settings section contains an allow list of pages in the Settings app.

MAIN MENU	SUB-MENU	PAGE NAME
System		SettingsPageGroupPCSystem
	Display	SettingsPageDisplay
	Notifications & actions	SettingsPageAppsNotifications
	Phone	SettingsPageCalls
	Messaging	SettingsPageMessaging
	Battery	SettingsPageBatterySaver
	Apps for websites	SettingsPageAppsForWebsites
	Storage	SettingsPageStorageSenseStorageOverview
	Driving mode	SettingsPageDrivingMode
	Offline maps	SettingsPageMaps
	About	SettingsPagePCSystemInfo

MAIN MENU	SUB-MENU	PAGE NAME
Devices		SettingsPageGroupDevices
	Default camera	SettingsPagePhotos
	Bluetooth	SettingsPagePCSystemBluetooth
	NFC	SettingsPagePhoneNFC
	Mouse	SettingsPageMouseTouchpad
	USB	SettingsPageUsb
Network and wireless		SettingsPageGroupNetwork
	Cellular & SIM	SettingsPageNetworkCellular
	Wi-Fi	SettingsPageNetworkWiFi
	Airplane mode	SettingsPageNetworkAirplaneMode
	Data usage	SettingsPageDataSenseOverview
	Mobile hotspot	SettingsPageNetworkMobileHotspot
	VPN	SettingsPageNetworkVPN
Personalization		SettingsPageGroupPersonalization
	Start	SettingsPageBackGround
	Colors	SettingsPageColors
	Sounds	SettingsPageSounds
	Lock screen	SettingsPageLockscreen
	Glance screen	SettingsPageGlance
	Navigation bar	SettingsNagivationBar
Accounts		SettingsPageGroupAccounts
	Your info	SettingsPageAccountsPicture
	Sign-in options	SettingsPageAccountsSignInOptions
	Email & app accounts	SettingsPageAccountsEmailApp
	Access work or school	SettingsPageWorkAccess

MAIN MENU	SUB-MENU	PAGE NAME
	Sync your settings	SettingsPageAccountsSync
	Apps corner (disabled in Assigned Access)	SettingsPageAppsCorner
Time & language		SettingsPageGroupTimeRegion
	Date & time	SettingsPageTimeRegionDateTime
	Language	SettingsPageTimeLanguage
	Region	SettingsPageTimeRegion
	Keyboard	SettingsPageKeyboard
	Speech	SettingsPageSpeech
Ease of access		SettingsPageGroupEaseOfAccess
	Narrator	SettingsPageEaseOfAccessNarrator
	Magnifier	SettingsPageEaseOfAccessMagnifier
	High contrast	SettingsPageEaseOfAccessHighContrast
	Closed captions	SettingsPageEaseOfAccessClosedCaptioning
	More options	SettingsPageEaseOfAccessMoreOptions
Privacy		SettingsPageGroupPrivacy
	Location	SettingsPagePrivacyLocation
	Camera	SettingsPagePrivacyWebcam
	Microphone	SettingsPagePrivacyMicrophone
	Motion	SettingsPagePrivacyMotionData
	Notifications	SettingsPagePrivacyNotifications
	Speech, inking, & typing	SettingsPagePrivacyPersonalization
	Account info	SettingsPagePrivacyAccountInfo
	Contacts	SettingsPagePrivacyContacts
	Calendar	SettingsPagePrivacyCalendar

MAIN MENU	SUB-MENU	PAGE NAME
	Phone calls	SettingsPagePrivacyPhoneCall
	Call history	SettingsPagePrivacyCallHistory
	Email	SettingsPagePrivacyEmail
	Messaging	SettingsPagePrivacyMessaging
	Radios	SettingsPagePrivacyRadios
	Continue App Experiences	SettingsPagePrivacyCDP
	Background apps	SettingsPagePrivacyBackgroundApps
	Accessory apps	SettingsPageAccessories
	Advertising ID	SettingsPagePrivacyAdvertisingId
	Other devices	SettingsPagePrivacyCustomPeripherals
	Feedback and diagnostics	SettingsPagePrivacySIUFSettings
Update and security		SettingsPageGroupRestore
	Phone update	SettingsPageRestoreMusUpdate
	Windows Insider Program	SettingsPageFlights
	Device encryption	SettingsPageGroupPCSystemDeviceEncryption
	Backup	SettingsPageRestoreOneBackup
	Find my phone	SettingsPageFindMyDevice
	For developers	SettingsPageSystemDeveloperOptions
OEM		SettingsPageGroupExtensibility
	Extensibility	SettingsPageExtensibility

## Quick actions lockdown

Quick action buttons are locked down in exactly the same way as Settings pages/groups. By default they are always conditional.

You can specify the quick actions as follows:

```
<Settings>
 <System name="QuickActions_Launcher_AllSettings" />
 <System name="QuickActions_Launcher_DeviceDiscovery" />
 <System name="SystemSettings_BatterySaver_LandingPage_OverrideControl" />
 <System name="SystemSettings_Device_BluetoothQuickAction"/>
 <System name="SystemSettings_Flashlight_Toggle"/>
 <System name="SystemSettings_Launcher_QuickNote" />
 <System name="SystemSettings_Network_VPN_QuickAction"/>
 <System name="SystemSettings_Privacy_LocationEnabledUserPhone"/>
 <System name="SystemSettings_QuickAction_AirplaneMode"/>
 <System name="SystemSettings_QuickAction_Camera" />
 <System name="SystemSettings_QuickAction_CellularData"/>
 <System name="SystemSettings_QuickAction_InternetSharing"/>
 <System name="SystemSettings_QuickAction_QuietHours" />
 <System name="SystemSettings_QuickAction_WiFi"/>
 <System name="SystemSettings_System_Display_Internal_Rotation"/>
 <System name="SystemSettings_System_Display_QuickAction_Brightness"/>
</Settings>
```

## Related topics

[Configure Windows 10 Mobile using Lockdown XML](#)

[Product IDs in Windows 10 Mobile](#)

# Product IDs in Windows 10 Mobile

7/28/2017 • 1 min to read • [Edit Online](#)

## Applies to

- Windows 10 Mobile

You can use the product ID and Application User Model (AUMID) in Lockdown.xml to specify apps that will be available to the user.

## Apps included in Windows 10 Mobile

The following table lists the product ID and AUMID for each app that is included in Windows 10 Mobile.

APP	PRODUCT ID	AUMID
Alarms and clock	44F7D2B4-553D-4BEC-A8B7-634CE897ED5F	Microsoft.WindowsAlarms_8wekyb3d8bbwe!App
Calculator	B58171C6-C70C-4266-A2E8-8F9C994F4456	Microsoft.WindowsCalculator_8wekyb3d8bbwe!App
Camera	F0D8FEFD-31CD-43A1-A45A-D0276DB069F1	Microsoft.WindowsCamera_8wekyb3d8bbwe!App
Contact Support	0DB5FCFF-4544-458A-B320-E352DFD9CA2B	Windows.ContactSupport_cw5n1h2txeyewy!App
Cortana	FD68DCF4-166F-4C55-A4CA-348020F71B94	Microsoft.Windows.Cortana_cw5n1h2txeyewy!CortanaUI
Excel	EAD3E7C0-FAE6-4603-8699-6A448138F4DC	Microsoft.Office.Excel_8wekyb3d8bbwe!microsoft.excel
Facebook	82A23635-5BD9-DF11-A844-00237DE2DB9E	Microsoft.MSFacebook_8wekyb3d8bbwe!82a236355bd9df11a84400237de2db9e
File Explorer	C5E2524A-EA46-4F67-841F-6A9465D9D515	c5e2524a-ea46-4f67-841f-6a9465d9d515_cw5n1h2txeyewy!App
FM Radio	F725010E-455D-4C09-AC48-BCDEF0D4B626	N/A
Get Started	B3726308-3D74-4A14-A84C-867C8C735C3C	Microsoft.Getstarted_8wekyb3d8bbwe!App
Groove Music	D2B6A184-DA39-4C9A-9E0A-8B589B03DEC0	Microsoft.ZuneMusic_8wekyb3d8bbwe!Microsoft.ZuneMusic
Maps	ED27A07E-AF57-416B-BC0C-2596B622EF7D	Microsoft.WindowsMaps_8wekyb3d8bbwe!App

APP	PRODUCT ID	AUMID
Messaging	27E26F40-E031-48A6-B130-D1F20388991A	Microsoft.Messaging_8wekyb3d8bbwe!x27e26f40ye031y48a6yb130yd1f20388991ax
Microsoft Edge	395589FB-5884-4709-B9DF-F7D558663FFD	Microsoft.MicrosoftEdge_8wekyb3d8bbwe!MicrosoftEdge
Money	1E0440F1-7ABF-4B9A-863D-177970EEFB5E	Microsoft.BingFinance_8wekyb3d8bbwe!AppexFinance
Movies and TV	6AFFE59E-0467-4701-851F-7AC026E21665	Microsoft.ZuneVideo_8wekyb3d8bbwe!Microsoft.ZuneVideo
News	9C3E8CAD-6702-4842-8F61-B8B33CC9CAF1	Microsoft.BingNews_8wekyb3d8bbwe!AppexNews
OneDrive	AD543082-80EC-45BB-AA02-FFE7F4182BA8	Microsoft.MicrosoftSkydrive_8wekyb3d8bbwe!App
OneNote	CA05B3AB-F157-450C-8C49-A1F127F5E71D	Microsoft.Office.OneNote_8wekyb3d8bbwe!microsoft.onenoteim
Outlook Calendar	A558FEBA-85D7-4665-B5D8-A2FF9C19799B	Microsoft.WindowsCommunicationApps_8wekyb3d8bbwe!Microsoft.WindowsLive.Calendar
Outlook Mail	A558FEBA-85D7-4665-B5D8-A2FF9C19799B	Microsoft.WindowsCommunicationApps_8wekyb3d8bbwe!Microsoft.WindowsLive.Mail
People	60BE1FB8-3291-4B21-BD39-2221AB166481	Microsoft.People_8wekyb3d8bbwe!xb94d6231y84ddy49a8yace3ybc955e769e85x
Phone (dialer)	F41B5D0E-EE94-4F47-9CFE-3D3934C5A2C7	Microsoft.CommsPhone_8wekyb3d8bbwe!App
Photos	FCA55E1B-B9A4-4289-882F-084EF4145005	Microsoft.Windows.Photos_8wekyb3d8bbwe!App
Podcasts	C3215724-B279-4206-8C3E-61D1A9D63ED3	Microsoft.MSPodcast_8wekyb3d8bbwe!xc3215724yb279y4206y8c3ey61d1a9d63ed3x
Powerpoint	B50483C4-8046-4E1B-81BA-590B24935798	Microsoft.Office.PowerPoint_8wekyb3d8bbwe!microsoft.pptim
Settings	2A4E62D8-8809-4787-89F8-69D0F01654FB	2a4e62d8-8809-4787-89f8-69d0f01654fb_8wekyb3d8bbwe!App
Skype	C3F8E570-68B3-4D6A-BDBB-C0A3F4360A51	Microsoft.SkypeApp_kzf8qxf38zg5c!Skype.AppId

APP	PRODUCT ID	AUMID
Skype Video	27E26F40-E031-48A6-B130-D1F20388991A	Microsoft.Messaging_8wekyb3d8bbwe!App
Sports	0F4C8C7E-7114-4E1E-A84C-50664DB13B17	Microsoft.BingSports_8wekyb3d8bbwe!AppexSports
Storage	5B04B775-356B-4AA0-AAF8-6491FFEA564D	N/A
Store	7D47D89A-7900-47C5-93F2-46EB6D94C159	Microsoft.WindowsStore_8wekyb3d8bbwe!App
Voice recorder	7311B9C5-A4E9-4C74-BC3C-55B06BA95AD0	Microsoft.WindowsSoundRecorder_8wekyb3d8bbwe!App
Wallet	587A4577-7868-4745-A29E-F996203F1462	Microsoft.MicrosoftWallet_8wekyb3d8bbwe!App
Weather	63C2A117-8604-44E7-8CEF-DF10BE3A57C8	Microsoft.BingWeather_8wekyb3d8bbwe!App
Windows Feedback	7604089D-D13F-4A2D-9998-33FC02B63CE3	Microsoft.WindowsFeedback_8wekyb3d8bbwe!App
Word	258F115C-48F4-4ADB-9A68-1387E634459B	Microsoft.Office.Word_8wekyb3d8bbwe!microsoft.word
Xbox	B806836F-EEBE-41C9-8669-19E243B81B83	Microsoft.XboxApp_8wekyb3d8bbwe!Microsoft.XboxApp

## Related topics

[Configure Windows 10 Mobile using Lockdown XML](#)

[Settings and quick actions that can be locked down in Windows 10 Mobile](#)

# Start layout XML for mobile editions of Windows 10 (reference)

7/28/2017 • 10 min to read • [Edit Online](#)

## Applies to

- Windows 10

**Looking for consumer information?** See [Customize the Start menu](#)

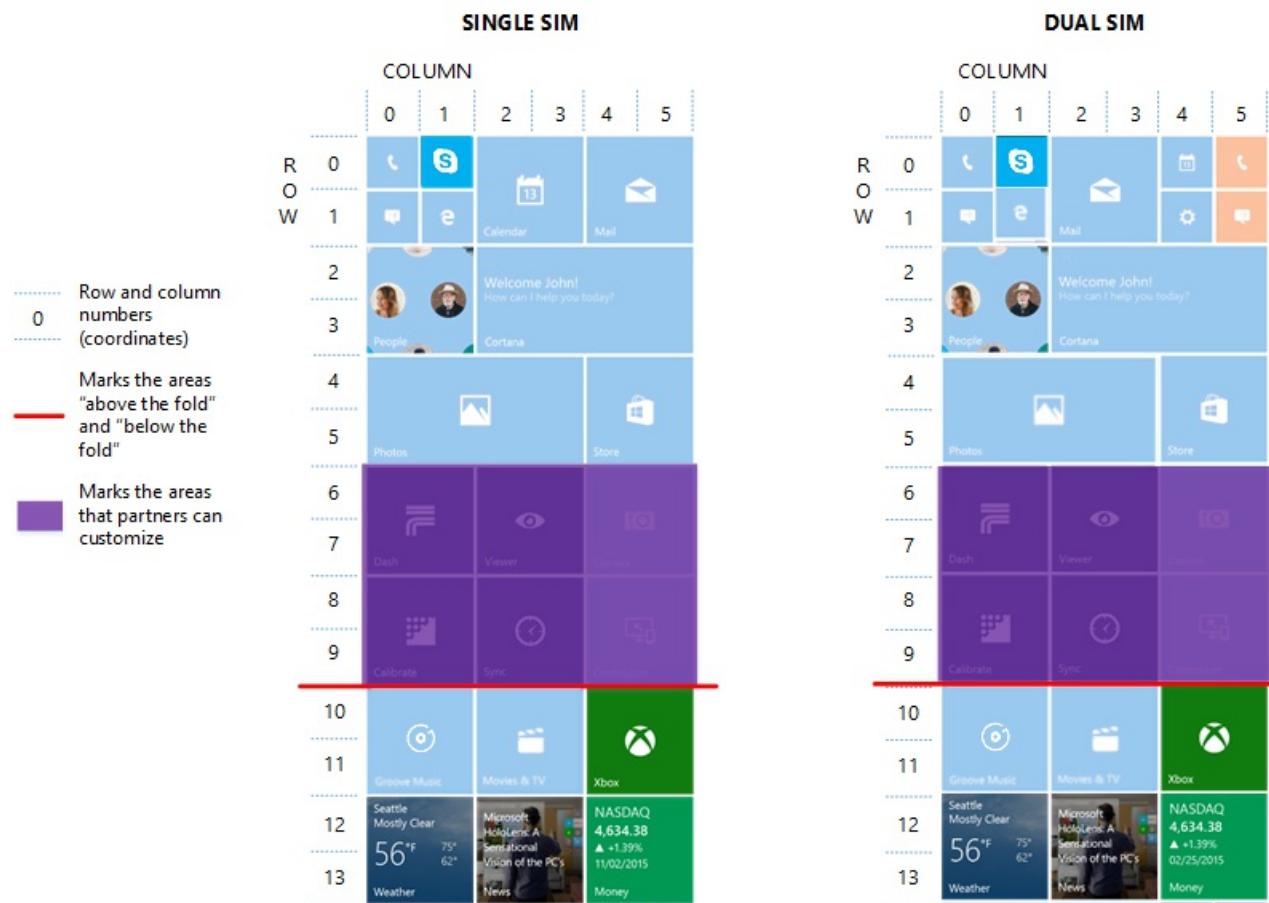
On Windows 10 Mobile, you can use the XML-based layout to modify the Start screen and provide the most robust and complete Start customization experience.

On Windows 10 Mobile, the customized Start works by:

- Windows 10 performs checks to determine the correct base default layout. The checks include the mobile edition, whether the device is dual SIM, the column width, and whether Cortana is supported for the country/region.
- Windows 10 ensures that it does not overwrite the layout that you have set and will sequence the level checks and read the file layout such that any multivariant settings that you have set is not overwritten.
- Windows 10 reads the LayoutModification.xml file and appends the group to the Start screen.

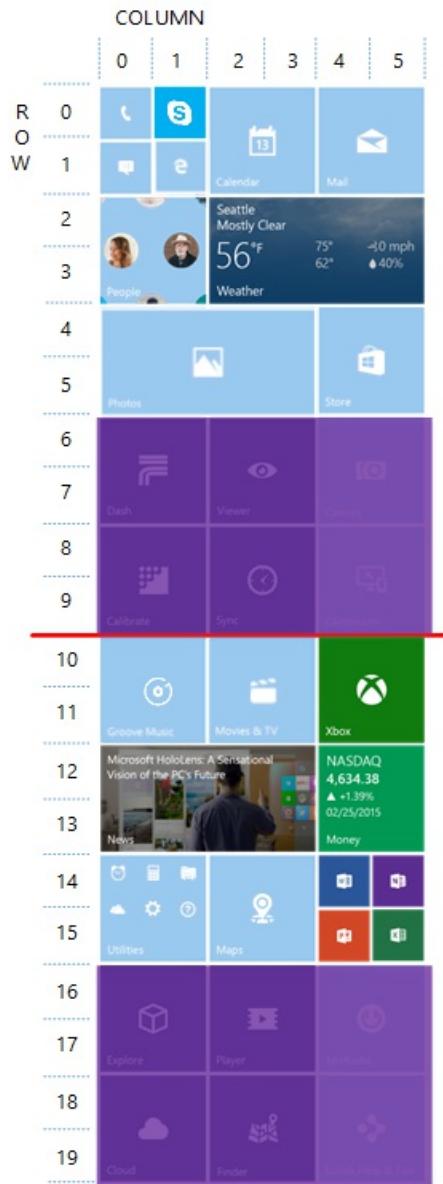
## Default Start layouts

The following diagrams show the default Windows 10, version 1607 Start layouts for single SIM and dual SIM devices with Cortana support, and single SIM and dual SIM devices with no Cortana support.





### SINGLE SIM – NO CORTANA



### DUAL SIM – NO CORTANA



The diagrams show:

- Tile coordinates - These are determined by the row number and the column number.
- Fold - Tiles "above the fold" are visible when users first navigate to the Start screen. Tiles "below the fold" are visible after users scroll up.
- Partner-customizable tiles - OEM and mobile operator partners can customize these areas of the Start screen by prepinning content. The partner configurable slots are:
  - o Rows 6-9

- o Rows 16-19

## LayoutModification XML

IT admins can provision the Start layout by creating a LayoutModification.xml file. This file supports several mechanisms to modify or replace the default Start layout and its tiles.

### NOTE

To make sure the Start layout XML parser processes your file correctly, follow these guidelines when writing your LayoutModification.xml file:

- Do not leave spaces or white lines in between each element.
- Do not add comments inside the StartLayout node or any of its children elements.
- Do not add multiple rows of comments.

The following table lists the supported elements and attributes for the LayoutModification.xml file.

ELEMENT	ATTRIBUTES	DESCRIPTION
LayoutModificationTemplate	xmlns xmlns:defaultlayout xmlns:start Version	Use to describe the changes to the default Start layout.
DefaultLayoutOverride Parent: LayoutModificationTemplate	n/a	Use to specify the customized Start layout for mobile devices.
StartLayoutCollection Parent: DefaultLayoutOverride	n/a	Use to contain a collection of Start layouts.
StartLayout Parent: StartLayoutCollection	n/a	Use to specify the tile groups that will be appended to the Start screen.
start:Group Parent: StartLayout	Name	Use to specify the tiles that need to be appended to the default Start layout.
start:Tile Parent: start:Group	AppUserModelID Size Row Column	Use to specify any Universal Windows app that has a valid <b>AppUserModelID</b> attribute.

ELEMENT	ATTRIBUTES	DESCRIPTION
start:SecondaryTile Parent: start:Group	AppUserModelID TileID Arguments DisplayName Square150x150LogoUri ShowNameOnSquare150x150Logo ShowNameOnWide310x150Logo Wide310x150LogoUri BackgroundColor ForegroundText IsSuggestedApp Size Row Column	Use to pin a Web link through a Microsoft Edge secondary tile.
start:PhoneLegacyTile Parent: start:Group	ProductID Size Row Column	Use to add a mobile app that has a valid <b>ProductID</b> attribute.
start:Folder Parent: start:Group	Name Size Row Column	Use to add a folder to the mobile device's Start screen.
RequiredStartTiles Parent: LayoutModificationTemplate	n/a	Use to specify the tiles that will be pinned to the bottom of the Start screen even if a restored Start screen does not have the tiles during backup or restore.

## start:Group

**start:Group** tags specify a group of tiles that will be appended to Start. You can set the **Name** attribute to specify a name for the Start group.

### NOTE

Windows 10 Mobile only supports one Start group.

For Windows 10 Mobile, **start:Group** tags can contain the following tags or elements:

- **start:Tile**
- **start:SecondaryTile**
- **start:PhoneLegacyTile**
- **start:Folder**

## Specify Start tiles

To pin tiles to Start, you must use the right kind of tile depending on what you want to pin.

### Tile size and coordinates

All tile types require a size (**Size**) and coordinates (**Row** and **Column**) attributes regardless of the tile type that you use when prepining items to Start.

The following table describes the attributes that you must use to specify the size and location for the tile.

ATTRIBUTE	DESCRIPTION
Size	Determines how large the tile will be. - 1x1 - small tile - 2x2 - medium tile - 4x2 - wide tile - 4x4 - large tile
Row	Specifies the row where the tile will appear.
Column	Specifies the column where the tile will appear.

For example, a tile with Size="2x2", Row="2", and Column="2" results in a tile located at (2,2) where (0,0) is the top-left corner of a group.

#### start:Tile

You can use the **start:Tile** tag to pin a Universal Windows app to Start.

To specify an app, you must set the **AppUserModelID** attribute to the application user model ID that's associated with the corresponding app.

The following example shows how to pin the Microsoft Edge Universal Windows app:

```
<start:Tile
 AppUserModelID="Microsoft.MicrosoftEdge_8wekyb3d8bbwe!MicrosoftEdge"
 Size="2x2"
 Row="0"
 Column="0"/>
```

#### start:SecondaryTile

You can use the **start:SecondaryTile** tag to pin a Web link through a Microsoft Edge secondary tile.

The following example shows how to create a tile of the Web site's URL using the Microsoft Edge secondary tile:

```
<start:SecondaryTile
 AppUserModelID="Microsoft.MicrosoftEdge_8wekyb3d8bbwe!MicrosoftEdge"
 TileID="MyWeblinkTile"
 Arguments="http://msn.com"
 DisplayName="MySite"
 Square150x150LogoUri="ms-appx:///Assets/MicrosoftEdgeSquare150x150.png"
 Wide310x150LogoUri="ms-appx:///Assets/MicrosoftEdgeWide310x150.png"
 ShowNameOnSquare150x150Logo="true"
 ShowNameOnWide310x150Logo="false"
 BackgroundColor="#FF112233"
 Size="2x2"
 Row="0"
 Column="4"/>
```

The following table describes the other attributes that you can use with the **start:SecondaryTile** tag in addition to **Size**, **Row**, and **Column**.

ATTRIBUTE	REQUIRED/OPTIONAL	DESCRIPTION
AppUserModelID	Required	Must point to Microsoft Edge.
TileID	Required	Must uniquely identify your Web site tile.

ATTRIBUTE	REQUIRED/OPTIONAL	DESCRIPTION
Arguments	Required	Must contain the URL of your Web site.
DisplayName	Required	Must specify the text that you want users to see.
Square150x150LogoUri	Required	Specifies the logo to use on the 2x2 tile.
Wide310x150LogoUri	Optional	Specifies the logo to use on the 4x2 tile.
ShowNameOnSquare150x150Logo	Optional	Specifies whether the display name is shown on the 2x2 tile. You can set the value for this attribute to true or false. By default, this is set to false.
ShowNameOnWide310x150Logo	Optional	Specifies whether the display name is shown on the 4x2 tile. You can set the value for this attribute to true or false. By default, this is set to false.
BackgroundColor	Optional	Specifies the color of the tile. You can specify the value in ARGB hexadecimal (for example, #FF112233) or specify "transparent".
ForegroundText	Optional	Specifies the color of the foreground text. Set the value to either "light" or "dark".

Secondary Microsoft Edge tiles have the same size and location behavior as a Universal Windows app.

#### **start:PhoneLegacyTile**

You can use the **start:PhoneLegacyTile** tag to add a mobile app that has a valid ProductID, which you can find in the app's manifest file. The **ProductID** attribute must be set to the GUID of the app.

The following example shows how to add a mobile app with a valid ProductID using the start:PhoneLegacyTile tag:

```
<start:PhoneLegacyTile
 ProductID="{00000000-0000-0000-0000-000000000000}"
 Size="2x2"
 Row="0"
 Column="2"/>
```

#### **start:Folder**

You can use the **start:Folder** tag to add a folder to the mobile device's Start screen.

You must set these attributes to specify the size and location of the folder: **Size**, **Row**, and **Column**.

Optionally, you can also specify a folder name by using the **Name** attribute. If you specify a name, set the value to a string.

The position of the tiles inside a folder is relative to the folder. You can add any of the following tile types to the folder:

- Tile - Use to pin a Universal Windows app to Start.
- SecondaryTile - Use to pin a Web link through a Microsoft Edge secondary tile.
- PhoneLegacyTile - Use to pin a mobile app that has a valid ProductID.

The following example shows how to add a medium folder that contains two apps inside it:

```
<start:Folder
 Name="Contoso apps"
 Size="2x2"
 Row="0"
 Column="2">
 <start:Tile
 AppUserModelID="Microsoft.BingMaps_8wekyb3d8bbwe!ApplicationID"
 Size="2x2"
 Row="0"
 Column="0"/>
 <start:PhoneLegacyTile
 ProductID="{00000000-0000-0000-0000-000000000000}"
 Size="1x1"
 Row="0"
 Column="2"/>
</start:Folder>
```

#### RequiredStartTiles

You can use the **RequiredStartTiles** tag to specify the tiles that will be pinned to the bottom of the Start screen even if a restored Start screen does not have the tiles during backup or restore.

##### NOTE

Enabling this Start customization may be disruptive to the user experience.

For Windows 10 Mobile, **RequiredStartTiles** tags can contain the following tags or elements. These are similar to the tiles supported in **start:Group**.

- Tile - Use to pin a Universal Windows app to Start.
- SecondaryTile - Use to pin a Web link through a Microsoft Edge secondary tile.
- PhoneLegacyTile - Use to pin a mobile app that has a valid ProductID.
- Folder - Use to pin a folder to the mobile device's Start screen.

Tiles specified within the **RequiredStartTiles** tag have the following behavior:

- The partner-pinned tiles will begin in a new row at the end of the user-restored Start screen.
- If there's a duplicate tile between what the user has in their Start screen layout and what the OEM has pinned to the Start screen, only the app or tile shown in the user-restored Start screen layout will be shown and the duplicate tile will be omitted from the pinned partner tiles at the bottom of the Start screen.

The lack of duplication only applies to pinned apps. Pinned Web links may be duplicated.

- If partners have prepinned folders to the Start screen, Windows 10 treats these folders in the same way as appended apps on the Start screen. Duplicate folders will be removed.
- All partner tiles that are appended to the bottom of the user-restored Start screen will be medium-sized. There will be no gaps in the appended partner Start screen layout. Windows 10 will shift tiles accordingly to prevent gaps.

## Sample LayoutModification.xml

The following sample LayoutModification.xml shows how you can configure the Start layout for devices running Windows 10 Mobile:

```

<?xml version="1.0" encoding="utf-8"?>
<LayoutModificationTemplate
 xmlns="http://schemas.microsoft.com/Start/2014/LayoutModification"
 xmlns:defaultlayout="http://schemas.microsoft.com/Start/2014/FullDefaultLayout"
 xmlns:start="http://schemas.microsoft.com/Start/2014/StartLayout"
 Version="1">
 <DefaultLayoutOverride>
 <StartLayoutCollection>
 <defaultlayout:StartLayout>
 <start:Group
 Name="First Group">
 <start:Tile
 AppUserModelID="Microsoft.BingFinance_8wekyb3d8bbwe!ApplicationID"
 Size="2x2"
 Row="0"
 Column="0"/>
 <start:Tile
 AppUserModelID="Microsoft.BingMaps_8wekyb3d8bbwe!ApplicationID"
 Size="1x1"
 Row="0"
 Column="2"/>
 </start:Group>
 </defaultlayout:StartLayout>
 </StartLayoutCollection>
 </DefaultLayoutOverride>
 <RequiredStartTiles>
 <PhoneLegacyTile ProductID="{b00d3141-1caa-43aa-b0b5-78c1acf778fd}" />
 <PhoneLegacyTile ProductID="{C3F8E570-68B3-4D6A-BDBB-C0A3F4360A51}" />
 <PhoneLegacyTile ProductID="{C60904B7-8DF4-4C2E-A417-C8E1AB2E51C7}" />
 <Tile AppUserModelID="Microsoft.MicrosoftFeedback_8wekyb3d8bbwe!ApplicationID" />
 </RequiredStartTiles>
</LayoutModificationTemplate>

```

## Use Windows Provisioning multivariant support

The Windows Provisioning multivariant capability allows you to declare target conditions that, when met, supply specific customizations for each variant condition. For Start customization, you can create specific layouts for each variant that you have. To do this, you must create a separate LayoutModification.xml file for each variant that you want to support and then include these in your provisioning package. For more information on how to do this, see [Create a provisioning package with multivariant settings](#).

The provisioning engine chooses the right customization file based on the target conditions that were met, adds the file in the location that's specified for the setting, and then uses the specific file to customize Start. To differentiate between layouts, you can add modifiers to the LayoutModification.xml filename such as "LayoutCustomization1". Regardless of the modifier that you use, the provisioning engine will always output "LayoutCustomization.xml" so that the OS has a consistent file name to query against.

For example, if you want to ensure that there's a specific layout for a certain mobile operator in a certain country/region, you can:

1. Create a specific layout customization file and then name it LayoutCustomization1.xml.
2. Include the file as part of your provisioning package.
3. Create your multivariant target and reference the XML file within the target condition in the main customization XML file.

The following example shows what the overall customization file might look like with multivariant support for Start:

```

<?xml version="1.0" encoding="utf-8"?>
<WindowsCustomizations>
 <PackageConfig xmlns="urn:schemas-Microsoft-com:Windows-ICD-Package-Config.v1.0">
 <ID>{6aaa4dfa-00d7-4aaa-8adf-73c6a7e2501e}</ID>
 <Name>My Provisioning Package</Name>
 <Version>1.0</Version>
 <OwnerType>OEM</OwnerType>
 <Rank>50</Rank>
 </PackageConfig>
 <Settings xmlns="urn:schemas-microsoft-com:windows-provisioning">
 <Customizations>
 <Targets>
 <Target Id="Operator XYZ">
 <TargetState>
 <Condition Name="MCC" Value="Range:310, 320" />
 <Condition Name="MNC" Value="!Range:400, 550" />
 </TargetState>
 </Target>
 <Target Id="Processor ABC">
 <TargetState>
 <TargetState>
 <Condition Name="ProcessorName" Value="Pattern:.*Celeron.*" />
 <Condition Name="ProcessorType" Value="Pattern:.*I|intel.*" />
 </TargetState>
 </TargetState>
 </Target>
 </Targets>
 <Common>
 <Settings>
 <Policies>
 <AllowBrowser>1</AllowBrowser>
 <AllowCamera>1</AllowCamera>
 <AllowBluetooth>1</AllowBluetooth>
 </Policies>
 <HotSpot>
 <Enabled>1</Enabled>
 </HotSpot>
 </Settings>
 </Common>
 <Variant>
 <TargetRefs>
 <TargetRef Id="Operator XYZ" />
 </TargetRefs>
 <Settings>
 <StartLayout>c:\users\</StartLayout>
 <userprofile>\appdata\local\Microsoft\Windows\Shell\LayoutCustomization1.XML</StartLayout>
 <HotSpot>
 <Enabled>1</Enabled>
 </HotSpot>
 </Settings>
 </Variant>
 </Customizations>
 </Settings>
</WindowsCustomizations>

```

When the condition is met, the provisioning engine takes the XML file and places it in the location that Windows 10 has set and then the Start subsystem reads the file and applies the specific customized layout.

You must repeat this process for all variants that you want to support so that each variant can have a distinct layout for each of the conditions and targets that need to be supported. For example, if you add a **Language** condition, you can create a Start layout that has its own localized group or folder titles.

## Add the LayoutModification.xml file to the image

Once you have created your LayoutModification.xml file to customize devices that will run Windows 10 Mobile, you can use Windows ICD to add the XML file to the device:

1. In the **Available customizations** pane, expand **Runtime settings**, select **Start** and then click the **StartLayout** setting.
2. In the middle pane, click **Browse** to open File Explorer.
3. In the File Explorer window, navigate to the location where you saved your LayoutModification.xml file.
4. Select the file and then click **Open**.

This should set the value of **StartLayout**. The setting appears in the **Selected customizations** pane.

## Related topics

- [Manage Windows 10 Start layout options](#)
- [Configure Windows 10 taskbar](#)
- [Customize Windows 10 Start and taskbar with Group Policy](#)
- [Customize Windows 10 Start and taskbar with ICD and provisioning packages](#)
- [Customize Windows 10 Start with mobile device management \(MDM\)](#)
- [Changes to Group Policy settings for Windows 10 Start](#)
- [Start layout XML for desktop editions of Windows 10 \(reference\)](#)

# Configure cellular settings for tablets and PCs

7/28/2017 • 2 min to read • [Edit Online](#)

## Applies to

- Windows 10

**Looking for consumer information?** See [Cellular settings in Windows 10](#)

Enterprises can configure cellular settings for tablets and PC that have built-in cellular modems or plug-in USB modem dongles and apply the settings in a [provisioning package](#). After the devices are configured, users are automatically connected using the access point name (APN) defined by the enterprise without needing to manually connect.

For users who work in different locations, you can configure one APN to connect when the users are at work and a different APN when the users are traveling.

## Prerequisites

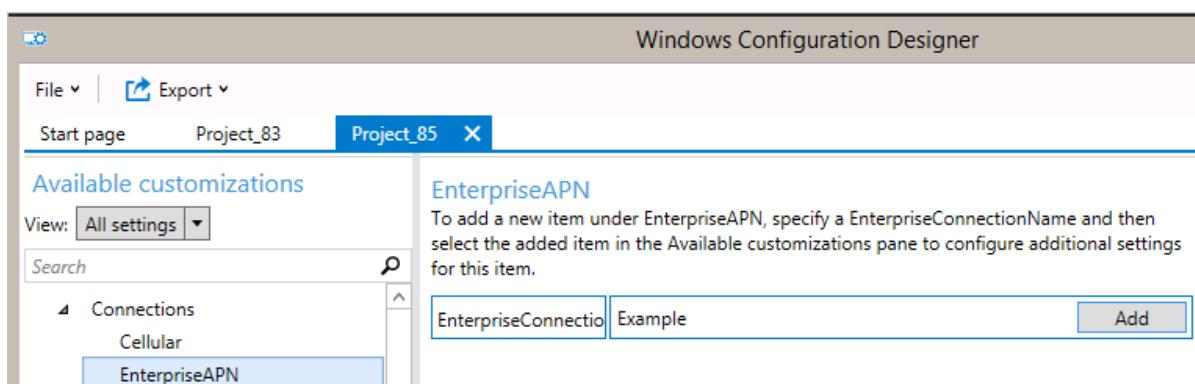
- Windows 10, version 1703, desktop editions (Home, Pro, Enterprise, Education)
- Tablet or PC with built-in cellular modem or plug-in USB modem dongle
- [Windows Configuration Designer](#)
- APN (the address that your PC uses to connect to the Internet when using the cellular data connection)

### NOTE

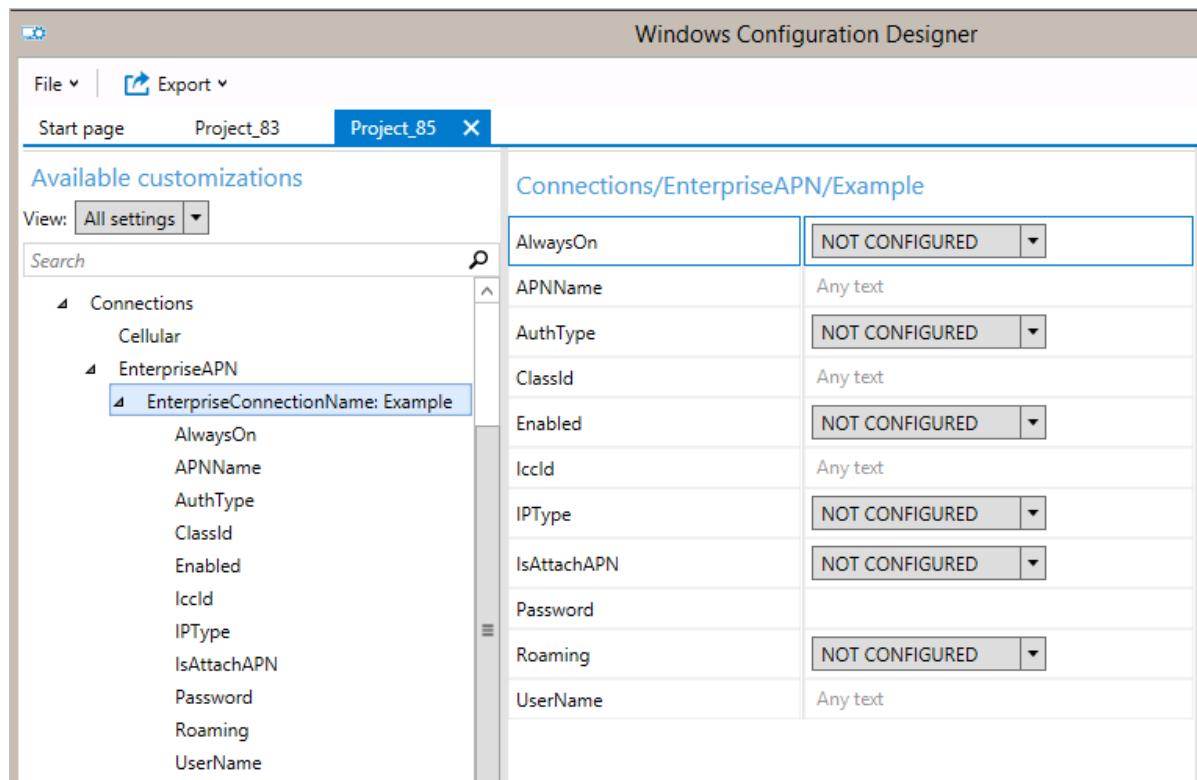
You can get the APN from your mobile operator.

## How to configure cellular settings in a provisioning package

- In Windows Configuration Designer, [start a new project](#) using the **Advanced provisioning** option.
- Enter a name for your project, and then click **Next**.
- Select **All Windows desktop editions**, click **Next**, and then click **Finish**.
- Go to **Runtime settings > Connections > EnterpriseAPN**.
- Enter a name for the connection, and then click **Add**.



6. The connection appears in the **Available customizations** pane. Select it to view the settings that you can configure for the connection.



7. The following table describes the settings available for the connection.

SETTING	DESCRIPTION
AlwaysOn	By default, the Connection Manager will automatically attempt to connect to the APN when a connection is available. You can disable this setting.
APNName	Enter the name of the APN.
AuthType	You can select <b>None</b> (the default), or specify <b>Auto</b> , <b>PAP</b> , <b>CHAP</b> , or <b>MSCHAPv2</b> authentication. If you select PAP, CHAP, or MSCHAPv2 authentication, you must also enter a user name and password.
ClassId	This is a GUID that defines the APN class to the modem. This is only required when <b>IsAttachAPN</b> is <b>true</b> and the attach APN is not only used as the Internet APN.
Enabled	By default, the connection is enabled. You can change this setting.
Icld	This is the Integrated Circuit Card ID (ICCID) associated with the cellular connection profile.
IPType	By default, the connection can use IPv4 and IPv6 concurrently. You can change this setting to only IPv4, only IPv6, or IPv6 with IPv4 provided by 46xlat.
IsAttachAPN	Specify whether this APN should be requested as part of an LTE Attach.

SETTING	DESCRIPTION
Password	If you select PAP, CHAP, or MSCHAPv2 authentication, enter a password that corresponds to the user name.
Roaming	Select the behavior that you want when the device is roaming. The options are: -Disallowed -Allowed (default) -DomesticRoaming -Use OnlyForDomesticRoaming -UseOnlyForNonDomesticRoaming -UseOnlyForRoaming
UserName	If you select PAP, CHAP, or MSCHAPv2 authentication, enter a user name.

8. After you configure the connection settings, [build the provisioning package](#).

9. [Apply the package to devices](#).

# Configure Start layout, taskbar, and lock screen for Windows 10 PCs

7/28/2017 • 1 min to read • [Edit Online](#)

## In this section

TOPIC	DESCRIPTION
<a href="#">Windows Spotlight on the lock screen</a>	Windows Spotlight is an option for the lock screen background that displays different background images and occasionally offers suggestions on the lock screen. <b>Note:</b> You can also use the <a href="#">Personalization CSP</a> settings to set lock screen and desktop background images.
<a href="#">Manage Windows 10 and Microsoft Store tips, tricks, and suggestions</a>	Options to manage the tips, tricks, and suggestions offered by Windows and Microsoft Store.
<a href="#">Manage Windows 10 Start and taskbar layout</a>	Organizations might want to deploy a customized Start screen and menu to devices running Windows 10 Pro, Enterprise, or Education. A standard Start layout can be useful on devices that are common to multiple users and devices that are locked down for specialized purposes.

## Related topics

- [Configure Windows 10 Mobile devices](#)

# Configure Windows Spotlight on the lock screen

10/17/2017 • 3 min to read • [Edit Online](#)

## Applies to

- Windows 10

Windows Spotlight is an option for the lock screen background that displays different background images and occasionally offers suggestions on the lock screen. Windows Spotlight is available in all desktop editions of Windows 10.

For managed devices running Windows 10 Enterprise and Windows 10 Education, enterprise administrators can configure a mobile device management (MDM) or Group Policy setting to prevent users from using the Windows Spotlight background. For managed devices running Windows 10 Pro, version 1607, administrators can disable suggestions for third party apps.

### NOTE

In Windows 10, version 1607, the lock screen background does not display if you disable the **Animate windows when minimizing and maximizing** setting in **This PC > Properties > Advanced system settings > Performance settings > Visual Effects**, or if you enable the Group Policy setting **Computer Configuration > Administrative Templates > Windows Components > Desktop Windows Manager > Do not allow windows animations**.

In Windows 10, version 1703, you can use the [Personalization CSP](#) settings to set lock screen and desktop background images.

## What does Windows Spotlight include?

- Background image**

The Windows Spotlight displays a new image on the lock screen each day. The initial background image is included during installation. Additional images are downloaded on ongoing basis.



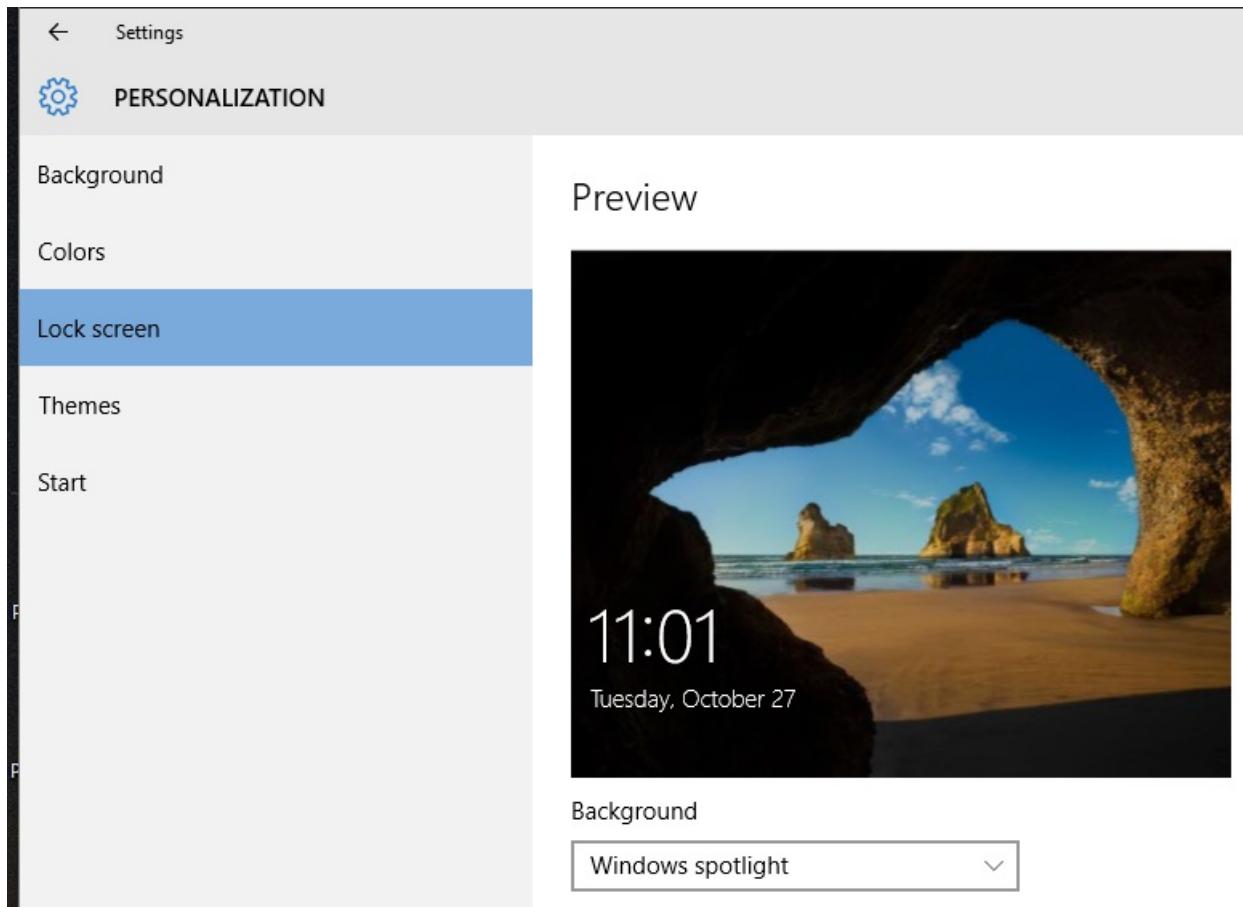
- Feature suggestions, fun facts, tips**

The lock screen background will occasionally suggest Windows 10 features that the user hasn't tried yet, such as **Snap assist**.



## How do you turn off Windows Spotlight locally?

To turn off Windows Spotlight locally, go to **Settings > Personalization > Lock screen > Background > Windows spotlight** > select a different lock screen background



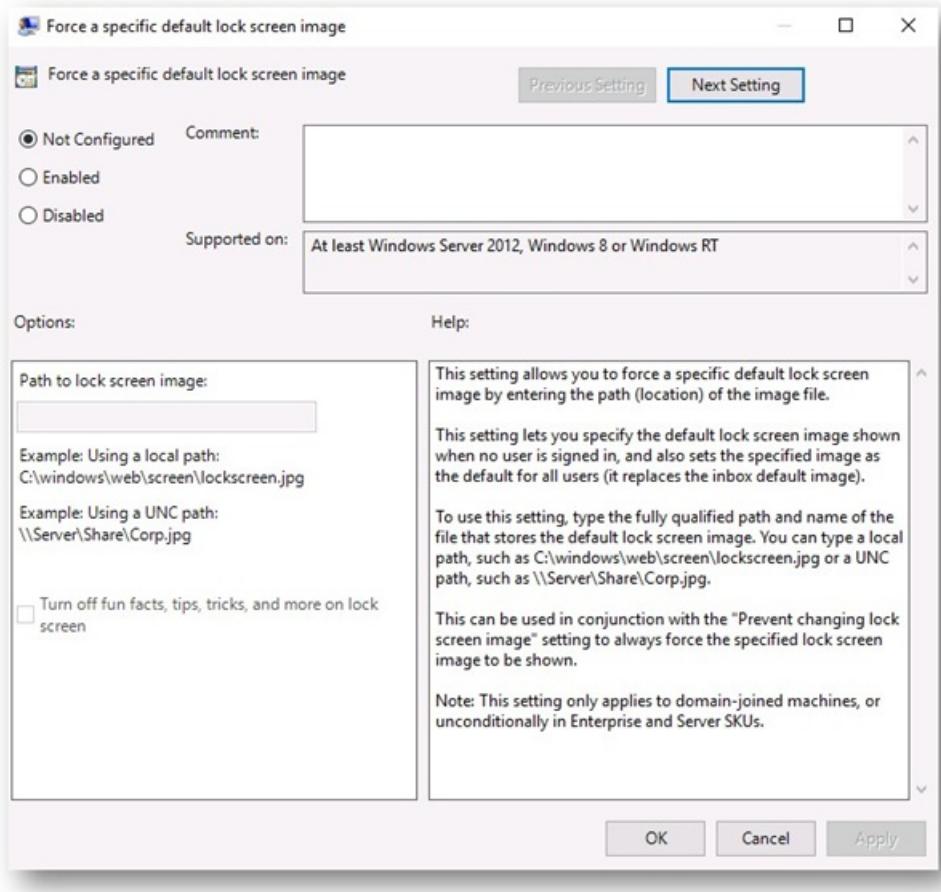
## How do you disable Windows Spotlight for managed devices?

Windows Spotlight is enabled by default. Windows 10 provides Group Policy and mobile device management (MDM) settings to help you manage Windows Spotlight on enterprise computers.

GROUP POLICY	MDM	DESCRIPTION	APPLIES TO
<b>User Configuration\Administrative Templates\Windows Components\Cloud Content\Do not suggest third-party content in Windows spotlight</b>	<b>Experience/Allow ThirdParty Suggestions In Windows Spotlight</b>	Enables enterprises to restrict suggestions to Microsoft apps and services	Windows 10 Pro, Enterprise, and Education, version 1607 and later

GROUP POLICY	MDM	DESCRIPTION	APPLIES TO
<b>User Configuration\Administrative Templates\Windows Components\Cloud Content\Turn off all Windows Spotlight features</b>	<b>Experience/Allow Windows Spotlight</b>	Enables enterprises to completely disable all Windows Spotlight features in a single setting	Windows 10 Enterprise and Education, version 1607 and later
<b>User Configuration\Administrative Templates\Windows Components\Cloud Content\Configure Spotlight on lock screen</b>	<b>Experience/Configure Windows Spotlight On Lock Screen</b>	Specifically controls the use of the dynamic Windows Spotlight image on the lock screen, and can be enabled or disabled	Windows 10 Enterprise and Education, version 1607 and later
<b>Administrative Templates \ Windows Components \ Cloud Content \ Turn off the Windows Spotlight on Action Center</b>	<b>Experience/Allow Windows Spotlight On Action Center</b>	Turn off Suggestions from Microsoft that show after each clean install, upgrade, or on an on-going basis to introduce users to what is new or changed	Windows 10 Enterprise and Education, version 1703
<b>User Configuration \ Administrative Templates \ Windows Components \ Cloud Content \ Do not use diagnostic data for tailored experiences</b>	<b>Experience/Allow Tailored Experiences With Diagnostic Data</b>	Prevent Windows from using diagnostic data to provide tailored experiences to the user	Windows 10 Pro, Enterprise, and Education, version 1703
<b>User Configuration \ Administrative Templates \ Windows Components \ Cloud Content \ Turn off the Windows Welcome Experience</b>	<b>Experience/Allow Windows Spotlight Windows Welcome Experience</b>	Turn off the Windows Spotlight Windows Welcome experience which helps introduce users to Windows, such as launching Microsoft Edge with a web page highlighting new features	Windows 10 Enterprise and Education, version 1703

In addition to the specific policy settings for Windows Spotlight, administrators can replace Windows Spotlight with a selected image using the Group Policy setting **Computer Configuration > Administrative Templates > Control Panel > Personalization > Force a specific default lock screen image.**



Pay attention to the checkbox in **Options**. In addition to providing the path to the lock screen image, administrators can choose to allow or **Turn off fun facts, tips, tricks, and more on lock screen**. If the checkbox is not selected, users will see the lock screen image that is defined in the policy setting, and will also see occasional messages.

## Related topics

[Manage Windows 10 Start layout options](#)

Not finding content you need? Windows 10 users, tell us what you want on [Feedback Hub](#).

# Manage Windows 10 and Microsoft Store tips, tricks, and suggestions

9/20/2017 • 1 min to read • [Edit Online](#)

## Applies to

- Windows 10

Since its inception, Windows 10 has included a number of user experience features that provide useful tips, tricks, and suggestions as you use Windows, as well as app suggestions from the Microsoft Store. These features are designed to help people get the most out of their Windows 10 experience by, for example, sharing new features, providing more details on the features they use, or sharing content available in the Microsoft Store. Examples of such user experiences include:

- Windows Spotlight on the lock screen.** Daily updated images on the lock screen that can include additional facts and tips in "hotspots" that are revealed on hover.
- Start menu app suggestions.** App suggestions in Start that recommend productivity tool or utilities from the Microsoft Store.
- Additional apps on Start.** Additional apps pre-installed on the Start screen which can enhance the user's experience.
- Windows tips.** Contextual tips that appear based on specific user actions to reveal related Windows features or help users complete a scenario.
- Microsoft account notifications.** For users who have a connected Microsoft account, toast notifications about their account like parental control notifications or subscription expiration.

### TIP

On all Windows desktop editions, users can directly enable and disable Windows 10 tips, tricks, and suggestions and Microsoft Store suggestions. For example, users are able to select personal photos for the lock screen as opposed to the images provided by Microsoft, or turn off tips, tricks, or suggestions as they use Windows.

Windows 10, version 1607 (also known as the Anniversary Update), provides organizations the ability to centrally manage the type of content provided by these features through Group Policy or mobile device management (MDM). The following table describes how administrators can manage suggestions and tips in Windows 10 commercial and education editions.

## Options available to manage Windows 10 tips and tricks and Microsoft Store suggestions

WINDOWS 10 EDITION	DISABLE	SHOW MICROSOFT APPS ONLY	SHOW MICROSOFT AND POPULAR THIRD-PARTY APPS
Windows 10 Pro	No	Yes	Yes (default)
Windows 10 Enterprise	Yes	Yes	Yes (default)

WINDOWS 10 EDITION	DISABLE	SHOW MICROSOFT APPS ONLY	SHOW MICROSOFT AND POPULAR THIRD-PARTY APPS
Windows 10 Pro Education	Yes (default)	Yes	No (setting cannot be changed)
Windows 10 Education	Yes (default)	Yes	No (setting cannot be changed)

[Learn more about policy settings for Windows Spotlight.](#)

## Related topics

- [Manage Windows 10 Start layout](#)
- [Cortana integration in your business or enterprise](#)
- [Windows spotlight on the lock screen](#)
- [Windows 10 editions for education customers](#)

# Manage Windows 10 Start and taskbar layout

10/31/2017 • 5 min to read • [Edit Online](#)

## Applies to

- Windows 10

**Looking for consumer information?** See [Customize the Start menu](#)

Organizations might want to deploy a customized Start and taskbar configuration to devices running Windows 10 Pro, Enterprise, or Education. A standard, customized Start layout can be useful on devices that are common to multiple users and devices that are locked down for specialized purposes. Configuring the taskbar allows the organization to pin useful apps for their employees and to remove apps that are pinned by default.

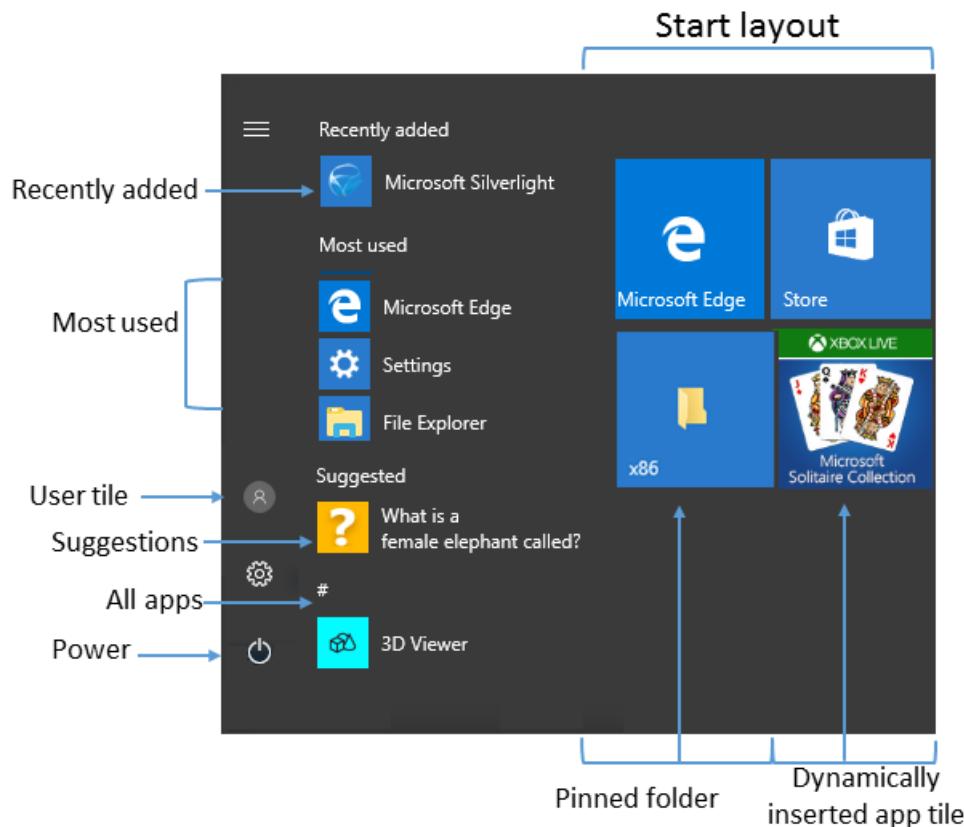
### NOTE

Taskbar configuration is available starting in Windows 10, version 1607.

Start and taskbar configuration can be applied to devices running Windows 10 Pro, version 1703.

Using the layout modification XML to configure Start is not supported with roaming user profiles. For more information, see [Deploy Roaming User Profiles](#).

## Start options



Some areas of Start can be managed using Group Policy. The layout of Start tiles can be managed using either Group Policy or Mobile Device Management (MDM) policy.

The following table lists the different parts of Start and any applicable policy settings or Settings options. Group

Policy settings are in the **User Configuration\Administrative Templates\Start Menu and Taskbar** path except where a different path is listed in the table.

START	POLICY	LOCAL SETTING
User tile	MDM: <b>Start/HideUserTile</b> <b>Start/HideSwitchAccount</b> <b>Start/HideSignOut</b> <b>Start/HideLock</b> <b>Start/HideChangeAccountSettings</b> Group Policy: <b>Remove Logoff on the Start menu</b>	none
Most used	MDM: <b>Start/HideFrequentlyUsedApps</b> Group Policy: <b>Remove frequent programs from the Start menu</b>	<b>Settings &gt; Personalization &gt; Start &gt; Show most used apps</b>
Suggestions -and- Dynamically inserted app tile	MDM: <b>Allow Windows Consumer Features</b> Group Policy: <b>Computer Configuration\Administrative Templates\Windows Components\Cloud Content\Turn off Microsoft consumer experiences</b>  <b>Note:</b> This policy also enables or disables notifications for a user's Microsoft account and app tiles from Microsoft dynamically inserted in the default Start menu.	<b>Settings &gt; Personalization &gt; Start &gt; Occasionally show suggestions in Start</b>
Recently added	MDM: <b>Start/HideRecentlyAddedApps</b>	<b>Settings &gt; Personalization &gt; Start &gt; Show recently added apps</b>
Pinned folders	MDM: <b>AllowPinnedFolder</b>	<b>Settings &gt; Personalization &gt; Start &gt; Choose which folders appear on Start</b>
Power	MDM: <b>Start/HidePowerButton</b> <b>Start/HideHibernate</b> <b>Start/HideRestart</b> <b>Start/HideShutDown</b> <b>Start/HideSleep</b> Group Policy: <b>Remove and prevent access to the Shut Down, Restart, Sleep, and Hibernate commands</b>	none

Start	Policy	Local setting
Start layout	<p>MDM: <b>Start layout</b> <b>ImportEdgeAssets</b></p> <p>Group Policy: <b>Prevent users from customizing their Start screen</b></p> <p><b>Note:</b> When a full Start screen layout is imported with Group Policy or MDM, the users cannot pin, unpin, or uninstall apps from the Start screen. Users can view and open all apps in the <b>All Apps</b> view, but they cannot pin any apps to the Start screen. When a partial Start screen layout is imported, users cannot change the tile groups applied by the partial layout, but can modify other tile groups and create their own.</p> <p><b>Start layout</b> policy can be used to pin apps to the taskbar based on an XML File that you provide. Users will be able to change the order of pinned apps, unpin apps, and pin additional apps to the taskbar.</p>	none
Jump lists	<p>MDM: <b>Start/HideRecentJumplists</b></p> <p>Group Policy: <b>Do not keep history of recently opened documents</b></p>	<b>Settings &gt; Personalization &gt; Start &gt; Show recently opened items in Jump Lists on Start or the taskbar</b>
Start size	<p>MDM: <b>Force Start size</b></p> <p>Group Policy: <b>Force Start to be either full screen size or menu size</b></p>	<b>Settings &gt; Personalization &gt; Start &gt; Use Start full screen</b>
App list	MDM: <b>Start/HideAppList</b>	<b>Settings &gt; Personalization &gt; Start &gt; Show app list in Start menu</b>
All Settings	Group Policy: <b>Prevent changes to Taskbar and Start Menu Settings</b>	none
Taskbar	MDM: <b>Start/NoPinningToTaskbar</b>	none

[Learn how to customize and export Start layout](#)

## Taskbar options

Starting in Windows 10, version 1607, you can pin additional apps to the taskbar and remove default pinned apps from the taskbar. You can specify different taskbar configurations based on device locale or region.

There are three categories of apps that might be pinned to a taskbar:

- Apps pinned by the user
- Default Windows apps, pinned during operating system installation (Microsoft Edge, File Explorer, Store)
- Apps pinned by the enterprise, such as in an unattended Windows setup

#### NOTE

We recommend using the [layoutmodification.xml](#) method to configure taskbar options, rather than the earlier method of using [TaskbarLinks](#) in an unattended Windows setup file.

The following example shows how apps will be pinned - Windows default apps to the left (blue circle), apps pinned by the user in the center (orange triangle), and apps that you pin using XML to the right (green square).



#### NOTE

In operating systems configured to use a right-to-left language, the taskbar order will be reversed.

Whether you apply the taskbar configuration to a clean install or an update, users will still be able to:

- Pin additional apps
- Change the order of pinned apps
- Unpin any app

#### NOTE

In Windows 10, version 1703, you can apply an MDM policy, [Start/NoPinningToTaskbar](#), to prevent users from pinning and unpinning apps on the taskbar.

### Taskbar configuration applied to clean install of Windows 10

In a clean install, if you apply a taskbar layout, only the apps that you specify and default apps that you do not remove will be pinned to the taskbar. Users can pin additional apps to the taskbar after the layout is applied.

### Taskbar configuration applied to Windows 10 upgrades

When a device is upgraded to Windows 10, apps will be pinned to the taskbar already. Some apps may have been pinned to the taskbar by a user, and others may have been pinned to the taskbar through a customized base image or by using Windows Unattend setup.

The new taskbar layout for upgrades to Windows 10, version 1607 or later, will apply the following behavior:

- If the user pinned the app to the taskbar, those pinned apps remain and new apps will be added to the right.
- If the user didn't pin the app (it was pinned during installation or by policy) and the app is not in updated layout file, the app will be unpinned.
- If the user didn't pin the app and the app is in the updated layout file, the app will be pinned to the right.
- New apps specified in updated layout file are pinned to right of user's pinned apps.

[Learn how to configure Windows 10 taskbar.](#)

## Related topics

- [Configure Windows 10 taskbar](#)
- [Customize and export Start layout](#)
- [Add image for secondary tiles](#)
- [Start layout XML for desktop editions of Windows 10 \(reference\)](#)
- [Customize Windows 10 Start and taskbar with Group Policy](#)

- [Customize Windows 10 Start and taskbar with provisioning packages](#)
- [Customize Windows 10 Start and tasbkar with mobile device management \(MDM\)](#)
- [Changes to Start policies in Windows 10](#)

Not finding content you need? Windows 10 users, tell us what you want on [Feedback Hub](#).

# Configure Windows 10 taskbar

10/17/2017 • 6 min to read • [Edit Online](#)

Starting in Windows 10, version 1607, administrators can pin additional apps to the taskbar and remove default pinned apps from the taskbar by adding a `<TaskbarLayout>` section to a layout modification XML file. This method never removes user-pinned apps from the taskbar.

## NOTE

The only aspect of the taskbar that can currently be configured by the layout modification XML file is the layout.

You can specify different taskbar configurations based on device locale and region. There is no limit on the number of apps that you can pin. You specify apps using the [Application User Model ID \(AUMID\)](#) or Desktop Application Link Path (the local path to the application).

If you specify an app to be pinned that is not provisioned for the user on the computer, the pinned icon won't appear on the taskbar.

The order of apps in the XML file dictates the order of pinned apps on the taskbar from left to right, to the right of any existing apps pinned by the user.

## NOTE

In operating systems configured to use a right-to-left language, the taskbar order will be reversed.

The following example shows how apps will be pinned: Windows default apps to the left (blue circle), apps pinned by the user in the center (orange triangle), and apps that you pin using the XML file to the right (green square).



## Configure taskbar (general)

To configure the taskbar:

1. Create the XML file.
  - If you are also [customizing the Start layout](#), use `Export-StartLayout` to create the XML, and then add the `<CustomTaskbarLayoutCollection>` section from the following sample to the file.
  - If you are only configuring the taskbar, use the following sample to create a layout modification XML file.
2. Edit and save the XML file. You can use [AUMID](#) or Desktop Application Link Path to identify the apps to pin to the taskbar.
  - Use `<taskbar:UWA>` and [AUMID](#) to pin Universal Windows Platform apps.
  - Use `<taskbar:DesktopApp>` and Desktop Application Link Path to pin desktop applications.
3. Apply the layout modification XML file to devices using [Group Policy](#) or a provisioning package created in [Windows Imaging and Configuration Designer \(Windows ICD\)](#).

## IMPORTANT

If you use a provisioning package to configure the taskbar, your configuration will be reapplied each time the explorer.exe process restarts. If your configuration pins an app and the user then unpins that app, the user's change will be overwritten the next time the configuration is applied. To apply a taskbar configuration that allows users to make changes that will persist, apply your configuration by using Group Policy.

If you use Group Policy and your configuration only contains a taskbar layout, the default Windows tile layout will be applied and cannot be changed by users. If you use Group Policy and your configuration includes taskbar and a full Start layout, users can only make changes to the taskbar. If you use Group Policy and your configuration includes taskbar and a [partial Start layout](#), users can make changes to the taskbar and to tile groups not defined in the partial Start layout.

## Tips for finding AUMID and Desktop Application Link Path

In the layout modification XML file, you will need to add entries for applications in the XML markup. In order to pin an application, you need either its AUMID or Desktop Application Link Path.

The easiest way to find this data for an application is to:

1. Pin the application to the Start menu on a reference or testing PC.
2. Open Windows PowerShell and run the `Export-StartLayout` cmdlet.
3. Open the generated XML file.
4. Look for an entry corresponding to the app you pinned.
5. Look for a property labeled `AppUserModelID` or `DesktopApplicationLinkPath`.

## Sample taskbar configuration XML file

```
<?xml version="1.0" encoding="utf-8"?>
<LayoutModificationTemplate
 xmlns="http://schemas.microsoft.com/Start/2014/LayoutModification"
 xmlns:defaultlayout="http://schemas.microsoft.com/Start/2014/FullDefaultLayout"
 xmlns:start="http://schemas.microsoft.com/Start/2014/StartLayout"
 xmlns:taskbar="http://schemas.microsoft.com/Start/2014/TaskbarLayout"
 Version="1">
 <CustomTaskbarLayoutCollection>
 <defaultlayout:TaskbarLayout>
 <taskbar:TaskbarPinList>
 <taskbar:UWA AppUserModelID="Microsoft.MicrosoftEdge_8wekyb3d8bbwe!MicrosoftEdge" />
 <taskbar:DesktopApp DesktopApplicationLinkPath="%APPDATA%\Microsoft\Windows\Start
Menu\Programs\System Tools\File Explorer.lnk" />
 </taskbar:TaskbarPinList>
 </defaultlayout:TaskbarLayout>
 </CustomTaskbarLayoutCollection>
</LayoutModificationTemplate>
```

## Sample taskbar configuration added to Start layout XML file

```

<?xml version="1.0" encoding="utf-8"?>
<LayoutModificationTemplate
 xmlns="http://schemas.microsoft.com/Start/2014/LayoutModification"
 xmlns:defaultlayout="http://schemas.microsoft.com/Start/2014/FullDefaultLayout"
 xmlns:start="http://schemas.microsoft.com/Start/2014/StartLayout"
 xmlns:taskbar="http://schemas.microsoft.com/Start/2014/TaskbarLayout"
 Version="1">
 <LayoutOptions StartTileGroupCellWidth="6" StartTileGroupsColumnCount="1" />
 <DefaultLayoutOverride>
 <StartLayoutCollection>
 <defaultlayout:StartLayout GroupCellWidth="6"
 xmlns:defaultlayout="http://schemas.microsoft.com/Start/2014/FullDefaultLayout">
 <start:Group Name="Life at a glance"
 xmlns:start="http://schemas.microsoft.com/Start/2014/StartLayout">
 <start:Tile Size="2x2" Column="0" Row="0"
 AppUserModelID="Microsoft.MicrosoftEdge_8wekyb3d8bbwe!MicrosoftEdge" />
 <start:Tile Size="2x2" Column="4" Row="0"
 AppUserModelID="Microsoft.Windows.Cortana_cw5n1h2txyewy!CortanaUI" />
 <start:Tile Size="2x2" Column="2" Row="0" AppUserModelID="Microsoft.BingWeather_8wekyb3d8bbwe!App"
 />
 </start:Group>
 </defaultlayout:StartLayout>
 </StartLayoutCollection>
 </DefaultLayoutOverride>
 <CustomTaskbarLayoutCollection>
 <defaultlayout:TaskbarLayout>
 <taskbar:TaskbarPinList>
 <taskbar:UWA AppUserModelID="Microsoft.MicrosoftEdge_8wekyb3d8bbwe!MicrosoftEdge" />
 <taskbar:DesktopApp DesktopApplicationLinkPath="%APPDATA%\Microsoft\Windows\Start
Menu\System Tools\File Explorer.lnk" />
 </taskbar:TaskbarPinList>
 </defaultlayout:TaskbarLayout>
 </CustomTaskbarLayoutCollection>
</LayoutModificationTemplate>

```

## Keep default apps and add your own

The `<CustomTaskbarLayoutCollection>` section will append listed apps to the taskbar by default. The following sample keeps the default apps pinned and adds pins for Paint, Microsoft Reader, and a command prompt.

```

<?xml version="1.0" encoding="utf-8"?>
<LayoutModificationTemplate
 xmlns="http://schemas.microsoft.com/Start/2014/LayoutModification"
 xmlns:defaultlayout="http://schemas.microsoft.com/Start/2014/FullDefaultLayout"
 xmlns:start="http://schemas.microsoft.com/Start/2014/StartLayout"
 xmlns:taskbar="http://schemas.microsoft.com/Start/2014/TaskbarLayout"
 Version="1">
 <CustomTaskbarLayoutCollection>
 <defaultlayout:TaskbarLayout>
 <taskbar:TaskbarPinList>
 <taskbar:DesktopApp DesktopApplicationLinkPath="%ALLUSERSPROFILE%\Microsoft\Windows\Start
Menu\Programs\Accessories\Paint.lnk" />
 <taskbar:UWA AppUserModelID="Microsoft.Reader_8wekyb3d8bbwe!Microsoft.Reader" />
 <taskbar:DesktopApp DesktopApplicationLinkPath="%appdata%\Microsoft\Windows\Start
Menu\Programs\System Tools\Command Prompt.lnk" />
 </taskbar:TaskbarPinList>
 </defaultlayout:TaskbarLayout>
 </CustomTaskbarLayoutCollection>
</LayoutModificationTemplate>

```

**Before:**



**After:**



## Remove default apps and add your own

By adding `PinListPlacement="Replace"` to `<CustomTaskbarLayoutCollection>`, you remove all default pinned apps; only the apps that you specify will be pinned to the taskbar.

If you only want to remove some of the default pinned apps, you would use this method to remove all default pinned apps and then include the default app that you want to keep in your list of pinned apps.

```
<?xml version="1.0" encoding="utf-8"?>
<LayoutModificationTemplate
 xmlns="http://schemas.microsoft.com/Start/2014/LayoutModification"
 xmlns:defaultlayout="http://schemas.microsoft.com/Start/2014/FullDefaultLayout"
 xmlns:start="http://schemas.microsoft.com/Start/2014/StartLayout"
 xmlns:taskbar="http://schemas.microsoft.com/Start/2014/TaskbarLayout"
 Version="1">
 <CustomTaskbarLayoutCollection PinListPlacement="Replace">
 <defaultlayout:TaskbarLayout>
 <taskbar:TaskbarPinList>
 <taskbar:DesktopApp DesktopApplicationLinkPath="%APPDATA%\Microsoft\Windows\Start
Menu\Programs\Accessories\Internet Explorer.lnk"/>
 <taskbar:DesktopApp DesktopApplicationLinkPath="%ALLUSERSPROFILE%\Microsoft\Windows\Start
Menu\Programs\Accessories\Paint.lnk" />
 <taskbar:UWA AppUserModelID="Microsoft.Office.Word_8wekyb3d8bbwe!microsoft.word" />
 </taskbar:TaskbarPinList>
 </defaultlayout:TaskbarLayout>
 </CustomTaskbarLayoutCollection>
</LayoutModificationTemplate>
```

**Before:**



**After:**



## Configure taskbar by country or region

The following example shows you how to configure taskbars by country or region. When the layout is applied to a computer, if there is no `<TaskbarPinList>` node with a region tag for the current region, the first `<TaskbarPinList>` node that has no specified region will be applied. When you specify one or more countries or regions in a `<TaskbarPinList>` node, the specified apps are pinned on computers configured for any of the specified countries or regions.

```

<?xml version="1.0" encoding="utf-8"?>
<LayoutModificationTemplate
 xmlns="http://schemas.microsoft.com/Start/2014/LayoutModification"
 xmlns:defaultlayout="http://schemas.microsoft.com/Start/2014/FullDefaultLayout"
 xmlns:start="http://schemas.microsoft.com/Start/2014/StartLayout"
 xmlns:taskbar="http://schemas.microsoft.com/Start/2014/TaskbarLayout"
 Version="1">

 <CustomTaskbarLayoutCollection PinListPlacement="Replace">
 <defaultlayout:TaskbarLayout region="US|UK">
 <taskbar:TaskbarPinList>
 <taskbar:UWA AppUserModelID="Microsoft.MicrosoftEdge_8wekyb3d8bbwe!MicrosoftEdge" />
 <taskbar:DesktopApp DesktopApplicationLinkPath="%APPDATA%\Microsoft\Windows\Start
Menu\Programs\System Tools\File Explorer.lnk" />
 <taskbar:UWA AppUserModelID="Microsoft.Office.Word_8wekyb3d8bbwe!microsoft.word" />
 <taskbar:DesktopApp DesktopApplicationLinkPath="%ALLUSERSPROFILE%\Microsoft\Windows\Start
Menu\Programs\Accessories\Paint.lnk"/>
 <taskbar:UWA AppUserModelID="Microsoft.Reader_8wekyb3d8bbwe!Microsoft.Reader" />
 </taskbar:TaskbarPinList>
 </defaultlayout:TaskbarLayout>
 <defaultlayout:TaskbarLayout region="DE|FR">
 <taskbar:TaskbarPinList>
 <taskbar:DesktopApp DesktopApplicationLinkPath="%APPDATA%\Microsoft\Windows\Start
Menu\Programs\System Tools\File Explorer.lnk" />
 <taskbar:UWA AppUserModelID="Microsoft.Office.Word_8wekyb3d8bbwe!microsoft.word" />
 <taskbar:UWA AppUserModelID="Microsoft.Office.Excel_8wekyb3d8bbwe!microsoft.excel" />
 <taskbar:DesktopApp DesktopApplicationLinkPath="%ALLUSERSPROFILE%\Microsoft\Windows\Start
Menu\Programs\Accessories\Paint.lnk"/>
 <taskbar:UWA AppUserModelID="Microsoft.Reader_8wekyb3d8bbwe!Microsoft.Reader" />
 </taskbar:TaskbarPinList>
 </defaultlayout:TaskbarLayout>
 <defaultlayout:TaskbarLayout>
 <taskbar:TaskbarPinList>
 <taskbar:DesktopApp DesktopApplicationLinkPath="%APPDATA%\Microsoft\Windows\Start
Menu\Programs\System Tools\File Explorer.lnk" />
 <taskbar:UWA AppUserModelID="Microsoft.Office.Word_8wekyb3d8bbwe!microsoft.word" />
 <taskbar:DesktopApp DesktopApplicationLinkPath="%ALLUSERSPROFILE%\Microsoft\Windows\Start
Menu\Programs\Accessories\Paint.lnk"/>
 <taskbar:UWA AppUserModelID="Microsoft.Reader_8wekyb3d8bbwe!Microsoft.Reader" />
 </taskbar:TaskbarPinList>
 </defaultlayout:TaskbarLayout>
 </CustomTaskbarLayoutCollection>
</LayoutModificationTemplate>

```

When the preceding example XML file is applied, the resulting taskbar for computers in the US or UK:



The resulting taskbar for computers in Germany or France:



The resulting taskbar for computers in any other country region:



#### NOTE

[Look up country and region codes \(use the ISO Short column\)](#)

# Layout Modification Template schema definition

```
<?xml version="1.0" encoding="utf-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:local="http://schemas.microsoft.com/Start/2014/TaskbarLayout"
 targetNamespace="http://schemas.microsoft.com/Start/2014/TaskbarLayout"
 elementFormDefault="qualified">

 <xsd:complexType name="ct_PinnedUWA">
 <xsd:attribute name="AppUserModelID" type="xsd:string" />
 </xsd:complexType>

 <xsd:complexType name="ct_PinnedDesktopApp">
 <xsd:attribute name="DesktopApplicationID" type="xsd:string" />
 <xsd:attribute name="DesktopApplicationLinkPath" type="xsd:string" />
 </xsd:complexType>

 <xsd:complexType name="ct_TaskbarPinList">
 <xsd:sequence>
 <xsd:choice minOccurs="1" maxOccurs="unbounded">
 <xsd:element name="UWA" type="local:ct_PinnedUWA" />
 <xsd:element name="DesktopApp" type="local:ct_PinnedDesktopApp" />
 </xsd:choice>
 </xsd:sequence>
 <xsd:attribute name="Region" type="xsd:string" use="optional" />
 </xsd:complexType>

 <xsd:simpleType name="st_TaskbarPinListPlacement">
 <xsd:restriction base="xsd:string">
 <xsd:enumeration value="Append" />
 <xsd:enumeration value="Replace" />
 </xsd:restriction>
 </xsd:simpleType>

 <xsd:attributeGroup name="ag_SelectionAttributes">
 <xsd:attribute name="SKU" type="xsd:string" use="optional"/>
 <xsd:attribute name="Region" type="xsd:string" use="optional"/>
 </xsd:attributeGroup>

 <xsd:complexType name="ct_TaskbarLayout">
 <xsd:sequence>
 <xsd:element name="TaskbarPinList" type="local:ct_TaskbarPinList" minOccurs="1" maxOccurs="1" />
 </xsd:sequence>
 <xsd:attributeGroup ref="local:ag_SelectionAttributes"/>
 </xsd:complexType>
</xsd:schema>
```

## Related topics

- [Manage Windows 10 Start and taskbar layout](#)
- [Customize and export Start layout](#)
- [Add image for secondary tiles](#)
- [Start layout XML for desktop editions of Windows 10 \(reference\)](#)
- [Customize Windows 10 Start and taskbar with Group Policy](#)
- [Customize Windows 10 Start and taskbar with provisioning packages](#)
- [Customize Windows 10 Start and taskbar with mobile device management \(MDM\)](#)
- [Changes to Start policies in Windows 10](#)

Not finding content you need? Windows 10 users, tell us what you want on [Feedback Hub](#).

# Customize and export Start layout

10/17/2017 • 5 min to read • [Edit Online](#)

## Applies to

- Windows 10

**Looking for consumer information?** See [Customize the Start menu](#)

The easiest method for creating a customized Start layout to apply to other Windows 10 devices is to set up the Start screen on a test computer and then export the layout.

After you export the layout, decide whether you want to apply a *full* Start layout or a *partial* Start layout.

When a full Start layout is applied, the users cannot pin, unpin, or uninstall apps from Start. Users can view and open all apps in the **All Apps** view, but they cannot pin any apps to Start.

When a **partial Start layout** is applied, the contents of the specified tile groups cannot be changed, but users can move those groups, and can also create and customize their own groups.

### NOTE

Partial Start layout is only supported on Windows 10, version 1511 and later.

You can deploy the resulting .xml file to devices using one of the following methods:

- [Group Policy](#)
- [Windows Configuration Designer provisioning package](#)
- [Mobile device management \(MDM\)](#)

## Customize the Start screen on your test computer

To prepare a Start layout for export, you simply customize the Start layout on a test computer.

### To prepare a test computer

- Set up a test computer on which to customize the Start layout. Your test computer should have the operating system that is installed on the users' computers (Windows 10 Pro, Enterprise, or Education). Install all apps and services that the Start layout should display.
- Create a new user account that you will use to customize the Start layout.

### To customize Start

- Sign in to your test computer with the user account that you created.
- Customize the Start layout as you want users to see it by using the following techniques:

- Pin apps to Start.** From Start, type the name of the app. When the app appears in the search results, right-click the app, and then click **Pin to Start**.

To view all apps, click **All apps** in the bottom-left corner of Start. Right-click any app, and pin or unpin it from Start.

- **Unpin apps** that you don't want to display. To unpin an app, right-click the app, and then click **Unpin from Start**.
- **Drag tiles** on Start to reorder or group apps.
- **Resize tiles**. To resize tiles, right-click the tile and then click **Resize**.
- **Create your own app groups**. Drag the apps to an empty area. To name a group, click above the group of tiles and then type the name in the **Name group** field that appears above the group.

#### IMPORTANT

In Windows 10, version 1703, if the Start layout includes tiles for apps that are not installed on the device that the layout is later applied to, the tiles for those apps will be blank. The blank tiles will persist until the next time the user signs in, at which time the blank tiles are removed. Some system events may cause the blank tiles to be removed before the next sign-in.

In earlier versions of Windows 10, no tile would be pinned.

## Export the Start layout

When you have the Start layout that you want your users to see, use the [Export-StartLayout](#) cmdlet in Windows PowerShell to export the Start layout to an .xml file.

#### IMPORTANT

If you include secondary Microsoft Edge tiles (tiles that link to specific websites in Microsoft Edge), see [Add custom images to Microsoft Edge secondary tiles](#) for instructions.

### To export the Start layout to an .xml file

1. From Start, open **Windows PowerShell**.
2. At the Windows PowerShell command prompt, enter the following command:

```
export-startlayout -path <path><file name>.xml
```

In the previous command, `-path` is a required parameter that specifies the path and file name for the export file. You can specify a local path or a UNC path (for example, `\FileServer01\StartLayouts\StartLayoutMarketing.xml`).

Use a file name of your choice—for example, `StartLayoutMarketing.xml`. Include the .xml file name extension. The [Export-StartLayout](#) cmdlet does not append the file name extension, and the policy settings require the extension.

Example of a layout file produced by `Export-StartLayout` :

## XML

```
<LayoutModificationTemplate Version="1"
 xmlns="http://schemas.microsoft.com/Start/2014/LayoutModification">
 <DefaultLayoutOverride>
 <StartLayoutCollection>
 <defaultlayout:StartLayout GroupCellWidth="6"
 xmlns:defaultlayout="http://schemas.microsoft.com/Start/2014/FullDefaultLayout">
 <start:Group Name="Life at a glance"
 xmlns:start="http://schemas.microsoft.com/Start/2014/StartLayout">
 <start:Tile Size="2x2" Column="0" Row="0"
 AppUserModelID="Microsoft.MicrosoftEdge_8wekyb3d8bbwe!MicrosoftEdge" />
 <start:Tile Size="2x2" Column="4" Row="0"
 AppUserModelID="Microsoft.Windows.Cortana_cw5n1h2txyewy!CortanaUI" />
 <start:Tile Size="2x2" Column="2" Row="0"
 AppUserModelID="Microsoft.BingWeather_8wekyb3d8bbwe!App" />
 </start:Group>
 </defaultlayout:StartLayout>
 </StartLayoutCollection>
 </DefaultLayoutOverride>
</LayoutModificationTemplate>
```

### IMPORTANT

If the Start layout that you export contains tiles for desktop (Win32) apps or .url links, **Export-StartLayout** will use **DesktopApplicationLinkPath** in the resulting file. Use a text or XML editor to change **DesktopApplicationLinkPath** to **DesktopApplicationID**. See [Specify Start tiles](#) for details on using the app ID in place of the link path.

## Configure a partial Start layout

A partial Start layout enables you to add one or more customized tile groups to users' Start screens or menus, while still allowing users to make changes to other parts of the Start layout. All groups that you add are *locked*, meaning users cannot change the contents of those tile groups, however users can change the location of those groups. Locked groups are identified with an icon, as shown in the following image.



When a partial Start layout is applied for the first time, the new groups are added to the users' existing Start layouts. If an app tile is in both an existing group and in a new locked group, the duplicate app tile is removed from the existing (unlocked) group.

When a partial Start layout is applied to a device that already has a StartLayout.xml applied, groups that were added previously are removed and the groups in the new layout are added.

If the Start layout is applied by Group Policy or MDM, and the policy is removed, the groups remain on the devices but become unlocked.

### To configure a partial Start screen layout

1. [Customize the Start layout](#).
2. [Export the Start layout](#).
3. Open the layout .xml file. There is a `<DefaultLayoutOverride>` element. Add `LayoutCustomizationRestrictionType="OnlySpecifiedGroups"` to the **DefaultLayoutOverride** element as follows:

```
<DefaultLayoutOverride LayoutCustomizationRestrictionType="OnlySpecifiedGroups">
```

4. Save the file and apply using any of the deployment methods.

## Related topics

- [Manage Windows 10 Start and taskbar layout](#)
- [Configure Windows 10 taskbar](#)
- [Add image for secondary tiles](#)
- [Start layout XML for desktop editions of Windows 10 \(reference\)](#)
- [Customize Windows 10 Start and taskbar with Group Policy](#)
- [Customize Windows 10 Start and taskbar with provisioning packages](#)
- [Customize Windows 10 Start and taskbar with mobile device management \(MDM\)](#)
- [Changes to Start policies in Windows 10](#)

Not finding content you need? Windows 10 users, tell us what you want on [Feedback Hub](#).

# Add image for secondary Microsoft Edge tiles

8/7/2017 • 7 min to read • [Edit Online](#)

## Applies to

- Windows 10
- Windows 10 Mobile

App tiles are the Start screen tiles that represent and launch an app. A tile that allows a user to go to a specific location in an app is a *secondary tile*. Some examples of secondary tiles include:

- Weather updates for a specific city in a weather app
- A summary of upcoming events in a calendar app
- Status and updates from an important contact in a social app
- A website in Microsoft Edge

In a Start layout for Windows 10, version 1703, you can include secondary tiles for Microsoft Edge that display a custom image, rather than a tile with the standard Microsoft Edge logo.

Suppose that the [Start layout that you export](#) had two secondary tiles, such as in the following image:



In prior versions of Windows 10, when you applied the Start layout to a device, the tiles would display as shown in the following image:



In Windows 10, version 1703, by using the PowerShell cmdlet `Export-StartLayoutEdgeAssets` and the policy setting `ImportEdgeAssets`, the tiles will now display the same as they did on the device from which you exported the Start layout.



## Example of secondary tiles in XML generated by Export-StartLayout

```
<start:SecondaryTile
 AppUserModelID="Microsoft.Windows.Edge_cw5n1h2txyewy!Microsoft.Edge.Edge"
 TileID="-9513911450"
 DisplayName="Bing"
 Size="2x2"
 Column="0"
 Row="0"
 Arguments="-contentTile -formatVersion 0x00000003 -pinnedTimeLow 0x36a8c2e4 -pinnedTimeHigh 0x01d0919b -securityFlags 0x00000000 -tileType 0x00000000 -url 0x00000014 http://www.bing.com/" Square150x150LogoUri="ms-appdata:///local/PinnedTiles/-9513911450/lowres.png"
 Wide310x150LogoUri="ms-appx:///"
 ShowNameOnSquare150x150Logo="true"
 ShowNameOnWide310x150Logo="true"
 BackgroundColor="#7fffffff"
/>
```

## Export Start layout and assets

1. Follow the instructions in [Customize and export Start layout](#) to customize the Start screen on your test computer.
2. Open Windows PowerShell and enter the following command:

```
Export-StartLayout -path <path><file name>.xml
```

In the previous command, `-path` is a required parameter that specifies the path and file name for the export file. You can specify a local path or a UNC path (for example, `\\\FileServer01\StartLayouts\StartLayoutMarketing.xml`).

Use a file name of your choice—for example, `StartLayoutMarketing.xml`. Include the `.xml` file name extension. The `Export-StartLayout` cmdlet does not append the file name extension, and the policy settings require the extension.

3. If you'd like to change the image for a secondary tile to your own custom image, open the `layout.xml` file, and look for the images that the tile references.
  - For example, your `layout.xml` contains

```
Square150x150LogoUri="ms-appdata:///local/PinnedTiles/21581260870/hires.png" Wide310x150LogoUri="ms-appx:///"
```
  - Open `c:\Users\<username>\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\LocalState` and replace those images with your customized images

### TIP

A quick method for getting appropriately sized images for each tile size is to upload your image at [BuildMyPinnedSite](#) and then download the resized tile images.

- a. In Windows PowerShell, enter the following command:

```
Export-StartLayoutEdgeAssets assets.xml
```

## Configure policy settings

You can apply the customized Start layout with images for secondary tiles by using [mobile device management](#) or [a provisioning package](#). However, because you are including the images for secondary tiles, you must

configure an additional policy to import the Edge assets.

## Using MDM

Follow the instructions to [create a custom policy](#). Replace the markup characters with escape characters in both the layout.xml and the assets.xml.

In addition to the `./User/Vendor/MSFT/Policy/Config/Start/StartLayout` setting, you must also add the `ImportEdgeAssets` setting.

ITEM	INFORMATION
<b>Setting name</b>	Enter a unique name for the OMA-URI setting to help you identify it in the list of settings.
<b>Setting description</b>	Provide a description that gives an overview of the setting and other relevant information to help you locate it.
<b>Data type</b>	<b>String</b>
<b>OMA-URI (case sensitive)</b>	<b>./User/Vendor/MSFT/Policy/Config/Start/ImportEdgeAssets</b>
<b>Value</b>	Paste the contents of the assets.xml file that you created.

## Example XML string value for the Start/ImportEdgeAssets policy

</SecondaryTileAsset>

## <SecondaryTileAsset>

```
<RelativeFilePath><![CDATA[PinnedTiles\30382655640\lowres.png]]></RelativeFilePath>
```

</SecondaryTileAsset>

## <SecondaryTileAsset>

```
<RelativeFilePath><![CDATA[PinnedTiles\29499280780\lowres.png]]></RelativeFilePath>
```

```
<Base64EncodedImage>iVBORw0KGgoAAAANSUhEUgAAAOEAAAdhCAMAAAjBsjIAAAAM1BMVEUAAAD///////////////+3leKCAAAAEVRSTlMAIECAR9//MFC/73CPEJ/PY0JQWV4AAAJsSURBVHgB7dyHcqswEEDRpa2KkcT/f+1zwNHYgtbevHPPtEzPNWxP8r8BAAAAAAAAAAAAAAAIhnoZFxSbnpxB1j+aMtzVoZaswjVMu2hIbBj+FRStbhw5bQ9TKVuF4y0ErW4VpHwSVvcKoldFCPaGQqgoppJBCCiUnb1rNF+pqv1c1CxTGEJZfLAz9F8bsk9wln3+hMeV3hVOSagg/W3iT3gsXJy+mnyvcPfcKqSx/URhdNJ7YRzkZP3hwmUP3Lou9HKh/EhhXF5Zey6c5dMw5sjx+wOH8EZiyd104yGGKqhqcPIQfKcxy14p2XZj1UJr9hv+Bwk3uXNG+C8dm17IkOSzfK4xHYNS+CxfZDvpNcsgvhetXpsQWtfPC30bocpqJ8sFHfVH2wJtq74W380V+Jzv3WiiuXE2J/gvHw1Ntcmj/+5S1yvuUmPUNC1PdcVbTubBdcdc6JfovvdjUzu1E1Lsyj1bnqbEou9aGK4KNXp5rJjh2XNUW4V17Z0eU0LVxqGG1J5Bvv12Gntcju40Wd+tcPzuvvR5fkAgre7n4XAxD4erQp1TnRkdFzY5i550p/xloZYt6lsVrqcNschnXJFjeFdijqeLGouRqnwy29qlGk6tFGY53HRXZ142U6iDH1YcwrzJwxDtFM5yJaudQvVyt5tVSYXTScfMYU1sA20VavTyzEc1U1iFUT4Ns9G73GX1491UeFKBQp4Y4qkvntyjkEIKKaSQtgp50+/0qn3L5FuyRVu86czb6nxggK9G80UPvt4CAAAAAAAAAAAAAAAAAAAAAAADezheimI2HJD0xgNwAAAABJRU5ErkJgg==</Base64EncodedImage>
```

</SecondaryTileAsset>

## <SecondaryTileAsset>

```
<RelativeFilePath><![CDATA[PinnedTiles\29499280780\lowres.png]]></RelativeFilePath>
```

```
<Base64EncodedImage>iVBORw0KGgoAAAANSUhEUgAAAOEAAAdCAMAAAjBsjIAAAAM1BMVEUAAAD///////////////+3leKCAAAEXRST1MAIECaR9//MFC/73CPEJ/PY0QjWV4AAAJsSURBVHgB7dyHcqswEEDRpaa2KkcT/f+1zwNHYgtbevbHPPtEzPNWxP8r8BAAAAAAAAAAAAAAAIAIBhnoZFxSbnpxB1J+aMtzoZaswjVMu2hIbBj+FRStbhwl5bq9tKVufF4y0ErW4VphwSVvcK0ldFCPaGQqoppJBCCinUb1rNF+pqv1c1CxTGEJzFLaz9F8bsk9wln3+hMeV3hVOSagg/W3iT3gsXJy+jmyvcPfCkqSx/URhdNj7YRzkZP3hwmUP3Lou9HKh/EhhXF5Zey6c5dMwSjx+WOH8EZiyd104yGGKqhqcPIQfKcxy14p2XZj1UJr9hv+Bwk3uXNG+C8dm17IkOSzFk4xHYNS+CxfZDvpNcsgvhetXpsQWtfPC30bocpqJ8sFhfVh2Wjtq74W380V+jZv3WiiuXE2J/gvHw1Ntcmj/+551yvuUmPUNC1PdcVbTubBdcdc6JfovvdjUzu1ElLsyj1bnqbEou9aGK4KNxp5rJjhX2NUW4V17Z0eU0LVxqGG1J5Bvv12Gntcju40Wd+tcPzuvvR5fkAgre7n4XAxD4erQp1TnRKdFzY5i550p/xloZYt6lsVrqcNschnxJFjeFdijqeLGouRqnWy29qLgk6tFGY53HRXZ142U6idHlYcwrzJwxDtFM5yJaudQvVyt5tVSYXTScfMYU1sA20VavTyzEc1U1iFUT4Ns9G73GKX1491UefKBoQp4Y4qkvntyjkEIKkaSQqp50+/0qn3L5FuyRVu86czb6nxxgK9G80UPvt4CAAAAAAAAAAAAAAAAAAAAAAADezhemI2HJD0xgNwAAAABJRUS5ErkJggg==</Base64EncodedImage>
```

```
</SecondaryTileAsset>
```

## <SecondaryTileAsset>

```
<RelativeFilePath><![CDATA[PinnedTiles\2819959990\lowres.png]]></RelativeFilePath>
```

```
<Base64EncodedImage>iVBORw0KGgoAAAANSUhEUgAAASwAAAEEsCAMAAABOo35HAAAAM1BMVEUAAAD///////////////+3leKCAAAEXRST1MAIFCAR9//QGCPv+8Qn88wcDAhSA0AAAK7SURBVHgB7d2JcrJIFIDR1pZGGzC8/900aHsLMOs/+8w5tWZPvupcGlKBBAAAAAAAAAAAAAAAA/DMdjvnUlT7xuf58qUNZiPWJsc/TtQSxPvB2zG05ifWZfp5qCWJ90sXD17FM8R/EMsU/05vi39eb418QqwSxxPpxnxCpf647pS2KFH9Qs0tfEiukINZ/7VAg1lhiSXWV8QSSyyxDHix/mhiSWWWNdap5zzPiJ1Yqg135yH3c882jrc3Nv0fWmm9FBLM6a7o1jxM6WhPJzTQ95dHb6IdZ03K+ktbVfSKT1cxYoaz5UzpGaM1g+Ohqs8c6Rrus1ftHqx7g7rGjk9neKtiyWau2M+58wb34tq1h3eX04TKEvi7p5Z7Hqau1cUxjL4tLK1dWku0u0abrVtmt28W+zTq3+RwmeFub9mLF6unj2HfoU6ymFANNrPVcinObc456bYa9FbGaa8yoKS2mU7zDFCPLgG/icDinxXWI08E5xdpYmx8rt3E+PrftU/zAnVhhbofD4Tmrjs9fvnR3KEGsU4tUnysst7HepZYxiNWatEY1NvX1FCNLrDCmRdfHnird1RyT3tZhV6DGnqpt3C99nCUGsXJaPMd6bOrPh7jCHMSq+1PCKa1cyopYw/5iwzWFNrIM+HBI DzGgXt5RrHBMT8eXn7QvQaztRb8cI3/3GrFeJ3yNTf3uNWK/Wu71/cTK7ztB9RhP7JsjcI5Pcz7kR+vESTM8XfoJqfd8QKQ62nfN09jPyhBLE+UmvNixIM+C+JJZZYYon1D+Nf6MQSSyyxDHixxBJLLFGsdzGzg0S3XrTTV3FciPq4Bbn/2lunu+xDB744VEyH1KEx195sJpH9gEAAAAAAA AAAAAAAAAAAAAAAAAAAAAAAAL8BVPKUzB0VBYIAAAASUVORK5CYII=</Base64EncodedImage>
```

```
</SecondaryTileAsset>
```

```
<SecondaryTileAsset>
```

```
<RelativeFilePath><![CDATA[PinnedTiles\2819959990\lowres.png]]></RelativeFilePath>
```

```
<Base64EncodedImage>iVBORw0KGgoAAAANSUhEUgAAASwAAAEEsCAMAAABOo35HAAAAM1BMVEUAAAD/////////////+3leKCAAAEXRST1MAIFCAR9//QGCPv+8Qn88wcDAhSA0AAAK7SURBVHgB7d2JcrJIFIDR1pZGGzC8/900aHsLMOs/+8w5tWZPvupcGlKBBAAAAAAAAAAAAAAAA/DMdjvnUlT7xuf58qUNZiPWJsc/TtQSxPvB2zG05ifWZfp5qCWJ90sXD17FM8R/EMsU/05vi39eb418QqwSxxPpxnxCpf647pS2KFH9Qs0tfEiukINZ/7VAg1lhiSXWV8QSSyyxDHix/mhiSWWWNdap5zzPiJ1Yqg135yH3c882jrc3Nv0fWmm9FBLM6a7o1jxM6WhPJzTQ95dHb6IdZ03K+ktbVfSKT1cxYoaz5UzpGaM1g+Ohqs8c6Rrus1ftHqx7g7rGjk9neKtiyWau2M+58wb34tq1h3eX04TKEvi7p5Z7Hqau1cUxjL4tLK1dWku0u0abrVtmt28W+zTq3+RwmeFub9mLF6unj2HfoU6ymFANNrPVcinObc456bYa9FbGaa8yoKS2mU7zDFCPLgG/icDinxXWI08E5xdpYmx8rt3E+PrftU/zAnVhhbofD4Tmrjs9fvnR3KEGsU4tUnysst7HepZYxiNWatEY1NvX1FCNLrDCmRdfHnird1RyT3tZhV6DGnqpt3C99nCUGsXJaPMd6bOrPh7jCHMSq+1PCKa1cyopYw/5iwzWFNrIM+HBI DzGgXt5RrHBMT8eXn7QvQaztRb8cI3/3GrFeJ3yNTf3uNWK/Wu71/cTK7ztB9RhP7JsjcI5Pcz7kR+vESTM8XfoJqfd8QKQ62nfN09jPyhBLE+UmvNixIM+C+JJZZYYon1D+Nf6MQSSyyxDHixxBJLLFGsdzGzg0S3XrTTV3FciPq4Bbn/2lunu+xDB744VEyH1KEx195sJpH9gEAAAAAAA AAAAAAAAAAAAAAAAL8BVPKUzB0VBYIAAAASUVORK5CYII=</Base64EncodedImage>
```

```
</SecondaryTileAsset>
```

```
<SecondaryTileAsset>
```

```
<RelativeFilePath><![CDATA[PinnedTiles\39721989420\lowres.png]]></RelativeFilePath>
```

```
<Base64EncodedImage>iVBORw0KGgoAAAANSUhEUgAAASwAAAEEsCAMAAABOo35HAAAAM1BMVEUAAAD/////////////+3leKCAAAEXRST1MAIFCAR9//QGCPv+8Qn88wcDAhSA0AAJLSURBVHgB7d3Joqo6FEDBbUOCAYT//9nTN/cgan9faka01qDjRASAgAAAAAAAAAAAAAAA/DMdjvnUlT7xuf58qUNZiPWJsc/TtQSxPvB2zG0vU0nbgtUUvyFWTPGKWKZ4BV8Aaa4wPek1x871lhipfu6fwTa8sDtcTqoppYSSyxxBJLLEuiCVWzv3U1cUSK17ty81YYnVTn/NXkmW4Gkusso9X30nm83YssYY1YhUrnxnjsUoph8Mh5zzGtcEdL85zxMaKeWgoV1zKZTPWGFuxYmk41vbgjm/rWEajxXzUB9r33qs2NFHiq71WHGsju1H2tXH6tvPlYcq2P15mKdyovDLr4sVx0IVdKbviZBWms0Vprj091YYk13Y63khmN18Wm4l0CsFJ9KXYK+PpZYk1gvuroEQ8OxyoMDfp8ajrXeh7kuQWk41ja+dse3pHzjdXPNveH6uVeT94an/tZThzEujF1qMtbadPGrb06V+a2VWhlzKwzrSb1Y43BrkfXXxXuxxvPN5fv5e3VRr01F+bRBrGVIVbHEGpcupduxxNrnnPeH0z1tEWv73vB/EUssscQS6waxxBJLLANerAeJJZZYYq105XTIc1UssSr46yCWWGLZUC6WNGKJJZZYYonvieWAREdvoTRVLAdR/ybm3hHnjxnz4XgW69HPMoj1d37wg/Gv+ZSM2R/Uz/6gfVYHv+8n+wAAAAAAAAAAAAAAAAGGdtQJGPPIrElgAAAABJRU5ErkJgg==</Base64EncodedImage>
```

```
</SecondaryTileAsset>
```

```
<SecondaryTileAsset>
```

```
<RelativeFilePath><![CDATA[PinnedTiles\39721989420\lowres.png]]></RelativeFilePath>
```

```
<Base64EncodedImage>iVBORw0KGgoAAAANSUhEUgAAASwAAAEEsCAMAAABOo35HAAAAM1BMVEUAAAD/////////////+3leKCAAAEXRST1MAIFCAR9//QGCPv+8Qn88wcDAhSA0AAJLSURBVHgB7d3Joqo6FEDBbUOCAYT//9nTN/cgan9faka01qDjRASAgAAAAAAAAAAAAAAA/DMdjvnUlT7xuf58qUNZiPWJsc/TtQSxPvB2zG0vU0nbgtUUvyFWTPGKWKZ4BV8Aaa4wPek1x871lhipfu6fwTa8sDtcTqoppYSSyxxBJLLEuiCVWzv3U1cUSK17ty81YYnVTn/NXkmW4Gkusso9X30nm83YssYY1YhUrnxnjsUoph8Mh5zzGtcEdL85zxMaKeWgoV1zKZTPWGFuxYmk41vbgjm/rWEajxXzUB9r33qs2NFHiq71WHGsju1H2tXH6tvPlYcq2P15mKdyovDLr4sVx0IVdKbviZBWms0Vprj091YYk13Y63khmN18Wm4l0CsFJ9KXYK+PpZYk1gvuroEQ8OxyoMDfp8ajrXeh7kuQWk41ja+dse3pHzjdXPNveH6uVeT94an/tZThzEujF1qMtbadPGrb06V+a2VWhlzKwzrSb1Y43BrkfXXxXuxxvPN5fv5e3VRr01F+bRBrGVIVbHEGpcupduxxNrnnPeH0z1tEWv73vB/EUssscQS6waxxBJLLANerAeJJZZYYq105XTIc1UssSr46yCWWGLZUC6WNGKJJZZYYonvieWAREdvoTRVLAdR/ybm3hHnjxnz4XgW69HPMoj1d37wg/Gv+ZSM2R/Uz/6gfVYHv+8n+wAAAAAAAAAAAAAAAAGGdtQJGPPIrElgAAAABJRU5ErkJgg==</Base64EncodedImage>
```

```

</SecondaryTileAsset>

<SecondaryTileAsset>

<RelativeFilePath><![CDATA[PinnedTiles\25495229280\lowres.png]]></RelativeFilePath>

<Base64EncodedImage>iVBORw0KGgoAAAANSUhEUgAAASwAAAECAMAAABo35HAAAAM1BMVEUAAAD///////////////+3leKCAAAEXRST1MAIFCAR9//QGCPv+8Qn88wcDAhSA0AAAJSURBVHgB7d1HcuMwEEBRKBg9/WUc6tQBZmhe2832V08XGEwl/kwAAAAAAAAAAAAAAAALDbH45dTl+g9MM45WeJprkcljV/kGo47Q/j1KNEU06WMdc1wha/IBG2uFg3bPEb20JifSVHYon1m2P1r3X71CZwdEMtsbrUJlaU/mJiiSWWWA611SpiiSWWWGKJ9U0JJZZDqUPpNHaNWGJN+c1Y5vTkLJZYx/LcZj671dmfJRBrmtozJdzpnKfzBBZ8n56V809DfidWvGe+bpNWHSyxNv2nHXWoDZZY9SU11wdLrE3/YcUvcbDEqo7WKT/ahcESK+rFvwYzhFitUarz7mkJ7sc0ZTG0ZqnrgJYYgXrFmmIgyVWRU1P+hIHS6yKMw3CYFnwrdHaHHMgVn00Sg7Eao/W8fpYYi1iXf/fcCfWDQt+Eevqo008ihYr2qUntfszDqXRVimfwmiJ1Rysvn3rT6w4WGue4k1lsaItUX1/XHFnwbdibYXGD8/FOrFeNAbrVHkuJ1ZjsJb8ZG2NVnrnifTu81G+F+vCuw5DfnH89DRfrMbZ6M2uc1tLrMZLM0t6MYoVY3U1LvRp/jRaDqXRuuZ3hzaY102jaMXcL3aHfmjAbHEEkssscQSSyyHUrHEEkssxZGzgcSfxpTrB9NLB+i9onz3ifObz0Xw3EV69afZDrZ/7gB/OP+SkZuz9dz+5PXL/709/LT/YBAAAAAAAAAAAAAAAAPAAPdXyNW8w51ZgAAAAASUVORK5CYII=</Base64EncodedImage>

</SecondaryTileAsset>

<SecondaryTileAsset>

<RelativeFilePath><![CDATA[PinnedTiles\25495229280\lowres.png]]></RelativeFilePath>

<Base64EncodedImage>iVBORw0KGgoAAAANSUhEUgAAASwAAAECAMAAABo35HAAAAM1BMVEUAAAD///////////////+3leKCAAAEXRST1MAIFCAR9//QGCPv+8Qn88wcDAhSA0AAAJSURBVHgB7d1HcuMwEEBRKBg9/WUc6tQBZmhe2832V08XGEwl/kwAAAAAAAAAAAAAAAALDbH45dTl+g9MM45WeJprkcljV/kGo47Q/j1KNEU06WMdc1wha/IBG2uFg3bPEb20JifSVHYon1m2P1r3X71CZwdEMtsbrUJlaU/mJiiSWWWA611SpiiSWWWGKJ9U0JJZZDqUPpNHaNWGJN+c1Y5vTkLJZYx/LcZj671dmfJRBrmtozJdzpnKfzBBZ8n56V809DfidWvGe+bpNWHSyxNv2nHXWoDZZY9SU11wdLrE3/YcUvcbDEqo7WKT/ahcESK+rFvwYzhFitUarz7mkJ7sc0ZTG0ZqnrgJYYgXrFmmIgyVWRU1P+hIHS6yKMw3CYFnwrdHaHHMgVn00Sg7Eao/W8fpYYi1iXf/fcCfWDQt+Eevqo008ihYr2qUntfszDqXRVimfwmiJ1Rysvn3rT6w4WGue4k1lsaItUX1/XHFnwbdibYXGD8/FOrFeNAbrVHkuJ1ZjsJb8ZG2NVnrnifTu81G+F+vCuw5DfnH89DRfrMbZ6M2uc1tLrMZLM0t6MYoVY3U1LvRp/jRaDqXRuuZ3hzaY102jaMXcL3aHfmjAbHEEkssscQSSyyHUrHEEkssxZGzgcSfxpTrB9NLB+i9onz3ifObz0Xw3EV69afZDrZ/7gB/OP+SkZuz9dz+5PXL/709/LT/YBAAAAAAAAAAAAAAAAPAAPdXyNW8w51ZgAAAAASUVORK5CYII=</Base64EncodedImage>

</SecondaryTileAsset>

</SecondaryTileAssets>

```

## Using a provisioning package

### Prepare the Start layout and Edge assets XML files

The **export-StartLayout** and **export-StartLayoutEdgeAssets** cmdlets produce XML files. Because Windows Configuration Designer produces a customizations.xml file that contains the configuration settings, adding the Start layout and Edge assets sections to the customizations.xml file directly would result in an XML file embedded in an XML file. Before you add the Start layout and Edge assets sections to the customizations.xml file, you must replace the markup characters in your layout.xml with escape characters.

1. Copy the contents of layout.xml into an online tool that escapes characters.
2. Copy the contents of assets.xml into an online tool that escapes characters.
3. During the procedure to create a provisioning package, you will copy the text with the escape characters and paste it in the customizations.xml file for your project.

### Create a provisioning package that contains a customized Start layout

Use the Windows Configuration Designer tool to create a provisioning package. [Learn how to install Windows Configuration Designer](#).

## IMPORTANT

When you build a provisioning package, you may include sensitive information in the project files and in the provisioning package (.ppkg) file. Although you have the option to encrypt the .ppkg file, project files are not encrypted. You should store the project files in a secure location and delete the project files when they are no longer needed.

1. Open Windows Configuration Designer (by default, %systemdrive%\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Imaging and Configuration Designer\x86\ICD.exe).
2. Choose **Advanced provisioning**.
3. Name your project, and click **Next**.
4. Choose **All Windows desktop editions** and click **Next**.
5. On **New project**, click **Finish**. The workspace for your package opens.
6. Expand **Runtime settings > Policies > Start**, and click **StartLayout**.

### TIP

If **Start** is not listed, check the type of settings you selected in step 4. You must create the project using settings for **All Windows desktop editions**.

7. Enter **layout.xml**. This value creates a placeholder in the customizations.xml file that you will replace with the contents of the layout.xml file in a later step.
8. In the **Available customizations** pane, select **ImportEdgeAssets**.
9. Enter **assets.xml**. This value creates a placeholder in the customizations.xml file that you will replace with the contents of the assets.xml file in a later step.
10. Save your project and close Windows Configuration Designer.
11. In File Explorer, open the project's directory. (The default location is C:\Users\user name\Documents\Windows Imaging and Configuration Designer (WICD)\project name)
12. Open the customizations.xml file in a text editor. The **<Customizations>** section will look like this:

```
<Customizations>
 <Common>
 <Policies>
 <Start>
 <ImportEdgeAssets>assets.xml</ImportEdgeAssets>
 <StartLayout>layout.xml</StartLayout>
 </Start>
 </Policies>
 </Common>
</Customizations>
```

13. Replace **layout.xml** with the text from the layout.xml file, [with markup characters replaced with escape characters](#).
14. Replace **assets.xml** with the text from the assets.xml file, [with markup characters replaced with escape characters](#).
15. Save and close the customizations.xml file.
16. Open Windows Configuration Designer and open your project.
17. On the **File** menu, select **Save**.

18. On the **Export** menu, select **Provisioning package**.
19. Change **Owner** to **IT Admin**, which will set the precedence of this provisioning package higher than provisioning packages applied to this device from other sources, and then select **Next**.
20. Optional. In the **Provisioning package security** window, you can choose to encrypt the package and enable package signing.
  - **Enable package encryption** - If you select this option, an auto-generated password will be shown on the screen.
  - **Enable package signing** - If you select this option, you must select a valid certificate to use for signing the package. You can specify the certificate by clicking **Select...** and choosing the certificate you want to use to sign the package.
21. Click **Next** to specify the output location where you want the provisioning package to go when it's built. By default, Windows Imaging and Configuration Designer (ICD) uses the project folder as the output location.

Optionally, you can click **Browse** to change the default output location.
22. Click **Next**.
23. Click **Build** to start building the package. The provisioning package doesn't take long to build. The project information is displayed in the build page and the progress bar indicates the build status.

If you need to cancel the build, click **Cancel**. This cancels the current build process, closes the wizard, and takes you back to the **Customizations Page**.
24. If your build fails, an error message will show up that includes a link to the project folder. You can scan the logs to determine what caused the error. Once you fix the issue, try building the package again.

If your build is successful, the name of the provisioning package, output directory, and project directory will be shown.

  - If you choose, you can build the provisioning package again and pick a different path for the output package. To do this, click **Back** to change the output package name and path, and then click **Next** to start another build.
  - If you are done, click **Finish** to close the wizard and go back to the **Customizations Page**.
25. Copy the provisioning package to the target device.
26. Double-click the ppkg file and allow it to install.

## Related topics

- [Manage Windows 10 Start and taskbar layout](#)
- [Configure Windows 10 taskbar](#)
- [Customize and export Start layout](#)
- [Start layout XML for desktop editions of Windows 10 \(reference\)](#)
- [Customize Windows 10 Start and taskbar with Group Policy](#)
- [Customize Windows 10 Start and taskbar with provisioning packages](#)
- [Customize Windows 10 Start and tasbkar with mobile device management \(MDM\)](#)
- [Changes to Start policies in Windows 10](#)

# Start layout XML for desktop editions of Windows 10 (reference)

9/19/2017 • 14 min to read • [Edit Online](#)

## Applies to

- Windows 10

**Looking for consumer information?** See [Customize the Start menu](#)

On Windows 10 for desktop editions, the customized Start works by:

- Windows 10 checks the chosen base default layout, such as the desktop edition and whether Cortana is supported for the country/region.
- Windows 10 reads the LayoutModification.xml file and allows groups to be appended to Start. The groups have the following constraints:
  - 2 groups that are 6 columns wide, or equivalent to the width of 3 medium tiles.
  - 2 medium-sized tile rows in height. Windows 10 ignores any tiles that are pinned beyond the second row.
  - No limit to the number of apps that can be pinned. There is a theoretical limit of 24 tiles per group (4 small tiles per medium square x 3 columns x 2 rows).

### NOTE

Using the layout modification XML to configure Start is not supported with roaming user profiles. For more information, see [Deploy Roaming User Profiles](#).

## LayoutModification XML

IT admins can provision the Start layout using a LayoutModification.xml file. This file supports several mechanisms to modify or replace the default Start layout and its tiles. The easiest method for creating a LayoutModification.xml file is by using the Export-StartLayout cmdlet; see [Customize and export Start layout](#) for instructions.

### NOTE

To make sure the Start layout XML parser processes your file correctly, follow these guidelines when working with your LayoutModification.xml file:

- Do not leave spaces or white lines in between each element.
- Do not add comments inside the StartLayout node or any of its children elements.
- Do not add multiple rows of comments.

The following table lists the supported elements and attributes for the LayoutModification.xml file.

ELEMENT	ATTRIBUTES	DESCRIPTION
LayoutModificationTemplate	xmlns xmlns:defaultlayout xmlns:start Version	Use to describe the changes to the default Start layout
<b>LayoutOptions</b> Parent: LayoutModificationTemplate	StartTileGroupsColumnCount FullScreenStart	Use to specify: - Whether to use full screen Start on the desktop - The number of tile columns in the Start menu
RequiredStartGroupsCollection Parent: LayoutModificationTemplate	n/a	Use to contain collection of RequiredStartGroups
<b>RequiredStartGroups</b> Parent: RequiredStartGroupsCollection	Region	Use to contain the AppendGroup tags, which represent groups that can be appended to the default Start layout
<b>AppendGroup</b> Parent: RequiredStartGroups	Name	Use to specify the tiles that need to be appended to the default Start layout
<b>start:Tile</b> Parent: AppendGroup	AppUserModelID Size Row Column	Use to specify any of the following: - A Universal Windows app - A Windows 8 or Windows 8.1 app Note that AppUserModelID is case-sensitive.
start:DesktopApplicationTile Parent: AppendGroup	DesktopApplicationID DesktopApplicationLinkPath Size Row Column	Use to specify any of the following: - A Windows desktop application with a known AppUserModelID - An application in a known folder with a link in a legacy Start Menu folder - A Windows desktop application link in a legacy Start Menu folder - A Web link tile with an associated .url file that is in a legacy Start Menu folder
start:SecondaryTile Parent: AppendGroup	AppUserModelID TileID Arguments DisplayName Square150x150LogoUri ShowNameOnSquare150x150Logo ShowNameOnWide310x150Logo Wide310x150LogoUri BackgroundColor ForegroundText IsSuggestedApp Size Row Column	Use to pin a Web link through a Microsoft Edge secondary tile. Note that AppUserModelID is case-sensitive.
TopMFUApps Parent: LayoutModificationTemplate	n/a	Use to add up to 3 default apps to the frequently used apps section in the system area

ELEMENT	ATTRIBUTES	DESCRIPTION
Tile Parent: TopMFUApps	AppUserModelID	Use with the TopMFUApps tags to specify an app with a known AppUserModelID
DesktopApplicationTile Parent: TopMFUApps	LinkFilePath	Use with the TopMFUApps tags to specify an app without a known AppUserModelID
AppendOfficeSuite Parent: LayoutModificationTemplate	n/a	Use to add the in-box installed Office suite to Start Do not use this tag with AppendDownloadOfficeTile
AppendDownloadOfficeTile Parent: LayoutModificationTemplate	n/a	Use to add a specific <b>Download Office</b> tile to a specific location in Start Do not use this tag with AppendOfficeSuite

## LayoutOptions

New devices running Windows 10 for desktop editions will default to a Start menu with 2 columns of tiles unless boot to tablet mode is enabled. Devices with screens that are under 10" have boot to tablet mode enabled by default. For these devices, users see the full screen Start on the desktop. You can adjust the following features:

- Boot to tablet mode can be set on or off.
- Set full screen Start on desktop to on or off. To do this, add the LayoutOptions element in your LayoutModification.xml file and set the FullScreenStart attribute to true or false.
- Specify the number of columns in the Start menu to 1 or 2. To do this, add the LayoutOptions element in your LayoutModification.xml file and set the StartTileGroupsColumnCount attribute to 1 or 2.

The following example shows how to use the LayoutOptions element to specify full screen Start on the desktop and to use 1 column in the Start menu:

```
<LayoutModificationTemplate
 xmlns="http://schemas.microsoft.com/Start/2014/LayoutModification"
 xmlns:defaultlayout="http://schemas.microsoft.com/Start/2014/FullDefaultLayout"
 xmlns:start="http://schemas.microsoft.com/Start/2014/StartLayout"
 Version="1">
 <LayoutOptions
 StartTileGroupsColumnCount="1"
 FullScreenStart="true"
 />
</LayoutModificationTemplate>
```

For devices being upgraded to Windows 10 for desktop editions:

- Devices being upgraded from Windows 7 will default to a Start menu with 1 column.
- Devices being upgraded from Windows 8.1 or Windows 8.1 Upgrade will default to a Start menu with 2 columns.

## RequiredStartGroups

The **RequiredStartGroups** tag contains **AppendGroup** tags that represent groups that you can append to the default Start layout.

## IMPORTANT

For Windows 10 for desktop editions, you can add a maximum of two (2) **AppendGroup** tags per **RequiredStartGroups** tag.

You can also assign regions to the append groups in the **RequiredStartGroups** tag's using the optional **Region** attribute or you can use the multivariant capabilities in Windows provisioning. If you are using the **Region** attribute, you must use a two-letter country code to specify the country/region that the append group(s) apply to. To specify more than one country/region, use a pipe ("|") delimiter as shown in the following example:

```
<RequiredStartGroups
Region="DE|ES|FR|GB|IT|US">
```

If the country/region setting for the Windows device matches a **RequiredStartGroups**, then the tiles laid out within the **RequiredStartGroups** is applied to Start.

If you specify a region-agnostic **RequiredStartGroups** (or one without the optional Region attribute) then the region-agnostic **RequiredStartGroups** is applied to Start.

## AppendGroup

**AppendGroup** tags specify a group of tiles that will be appended to Start. There is a maximum of two **AppendGroup** tags allowed per **RequiredStartGroups** tag.

For Windows 10 for desktop editions, AppendGroup tags contain start:Tile, start:DesktopApplicationTile, or start:SecondaryTile tags.

You can specify any number of tiles in an **AppendGroup**, but you cannot specify a tile with a **Row** attribute greater than 4. The Start layout does not support overlapping tiles.

## Specify Start tiles

To pin tiles to Start, partners must use the right kind of tile depending on what you want to pin.

### Tile size and coordinates

All tile types require a size (**Size**) and coordinates (**Row** and **Column**) attributes regardless of the tile type that you use when prepinning items to Start.

The following table describes the attributes that you must use to specify the size and location for the tile.

ATTRIBUTE	DESCRIPTION
Size	Determines how large the tile will be. - 1x1 - small tile - 2x2 - medium tile - 4x2 - wide tile - 4x4 - large tile
Row	Specifies the row where the tile will appear.
Column	Specifies the column where the tile will appear.

For example, a tile with Size="2x2", Row="2", and Column="2" results in a tile located at (2,2) where (0,0) is the top-left corner of a group.

### start:Tile

You can use the **start:Tile** tag to pin any of the following apps to Start:

- A Universal Windows app
- A Windows 8 app or Windows 8.1 app

To specify any one of these apps, you must set the **AppUserModelID** attribute to the application user model ID that's associated with the corresponding app.

#### IMPORTANT

**AppUserModelID** (AUMID) is case-sensitive.

The following example shows how to pin the Microsoft Edge Universal Windows app:

```
<start:Tile
 AppUserModelID="Microsoft.MicrosoftEdge_8wekyb3d8bbwe!MicrosoftEdge"
 Size="2x2"
 Row="0"
 Column="0"/>
```

#### start:DesktopApplicationTile

You can use the **start:DesktopApplicationTile** tag to pin a Windows desktop application to Start. There are two ways you can specify a Windows desktop application:

- By using a path to a shortcut link (.lnk file) to a Windows desktop application.

#### NOTE

In Start layouts for Windows 10, version 1703, you should use **DesktopApplicationID** rather than **DesktopApplicationLinkPath** if you are using Group Policy or MDM to apply the start layout and the application was installed after the user's first sign-in.

To pin a Windows desktop application through this method, you must first add the .lnk file in the specified location when the device first boots.

The following example shows how to pin the Command Prompt:

```
<start:DesktopApplicationTile
 DesktopApplicationLinkPath="%appdata%\Microsoft\Windows\Start Menu\Programs\System
Tools\Command Prompt.lnk"
 Size="2x2"
 Row="0"
 Column="4"/>
```

You must set the **DesktopApplicationLinkPath** attribute to the .lnk file that points to the Windows desktop application. The path also supports environment variables.

If you are pointing to a third-party Windows desktop application and the layout is being applied before the first boot, you must put the .lnk file in a legacy Start Menu directory before first boot; for example, "%APPDATA%\Microsoft\Windows\Start Menu\Programs" or the all users profile "%ALLUSERSPROFILE%\Microsoft\Windows\Start Menu\Programs".

- By using the application's application user model ID, if this is known. If the Windows desktop application doesn't have one, use the shortcut link option.

You can use the [Get-StartApps cmdlet](#) on a PC that has the application pinned to Start to obtain the app ID.

To pin a Windows desktop application through this method, you must set the **DesktopApplicationID** attribute to

the application user model ID that's associated with the corresponding app.

The following example shows how to pin the File Explorer Windows desktop application:

```
<start:DesktopApplicationTile
 DesktopApplicationID="Microsoft.Windows.Explorer"
 Size="2x2"
 Row="0"
 Column="2"/>
```

You can also use the **start:DesktopApplicationTile** tag as one of the methods for pinning a Web link to Start. The other method is to use a Microsoft Edge secondary tile.

To pin a legacy .url shortcut to Start, you must create .url file (right-click on the desktop, select **New > Shortcut**, and then type a Web URL). You must add this .url file in a legacy Start Menu directory before first boot; for example, `%APPDATA%\Microsoft\Windows\Start Menu\Programs\` or the all users profile `%ALLUSERSPROFILE%\Microsoft\Windows\Start Menu\Programs\`.

The following example shows how to create a tile of the Web site's URL, which you can treat similarly to a Windows desktop application tile:

```
<start:DesktopApplicationTile
 DesktopApplicationID="http://www.contoso.com/"
 Size="2x2"
 Row="0"
 Column="2"/>
```

#### NOTE

In Windows 10, version 1703, **Export-StartLayout** will use **DesktopApplicationLinkPath** for the .url shortcut. You must change **DesktopApplicationLinkPath** to **DesktopApplicationID** and provide the URL.

#### start:SecondaryTile

You can use the **start:SecondaryTile** tag to pin a Web link through a Microsoft Edge secondary tile. This method doesn't require any additional action compared to the method of using legacy .url shortcuts (through the **start:DesktopApplicationTile** tag).

The following example shows how to create a tile of the Web site's URL using the Microsoft Edge secondary tile:

```
<start:SecondaryTile
 AppUserModelID="Microsoft.MicrosoftEdge_8wekyb3d8bbwe!MicrosoftEdge"
 TileID="MyWeblinkTile"
 Arguments="http://msn.com"
 DisplayName="MySite"
 Square150x150LogoUri="ms-appx:///Assets/MicrosoftEdgeSquare150x150.png"
 Wide310x150LogoUri="ms-appx:///Assets/MicrosoftEdgeWide310x150.png"
 ShowNameOnSquare150x150Logo="true"
 ShowNameOnWide310x150Logo="false"
 BackgroundColor="#FF112233"
 Size="2x2"
 Row="0"
 Column="4"/>
```

The following table describes the other attributes that you can use with the **start:SecondaryTile** tag in addition to **Size**, **Row**, and **Column**.

ATTRIBUTE	REQUIRED/OPTIONAL	DESCRIPTION
AppUserModelID	Required	Must point to Microsoft Edge. Note that AppUserModelID is case-sensitive.
TileID	Required	Must uniquely identify your Web site tile.
Arguments	Required	Must contain the URL of your Web site.
DisplayName	Required	Must specify the text that you want users to see.
Square150x150LogoUri	Required	Specifies the logo to use on the 2x2 tile.
Wide310x150LogoUri	Optional	Specifies the logo to use on the 4x2 tile.
ShowNameOnSquare150x150Logo	Optional	Specifies whether the display name is shown on the 2x2 tile. The values you can use for this attribute are true or false.
ShowNameOnWide310x150Logo	Optional	Specifies whether the display name is shown on the 4x2 tile. The values you can use for this attribute are true or false.
BackgroundColor	Optional	Specifies the color of the tile. You can specify the value in ARGB hexadecimal (for example, #FF112233) or specify "transparent".
ForegroundText	Optional	Specifies the color of the foreground text. Set the value to either "light" or "dark".

Secondary Microsoft Edge tiles have the same size and location behavior as a Universal Windows app, Windows 8 app, or Windows 8.1 app.

#### TopMFUApps

You can use the **TopMFUApps** tag to add up to 3 default apps to the frequently used apps section in the system area, which delivers system-driven lists to the user including important or frequently accessed system locations and recently installed apps.

You can use this tag to add:

- Apps with an **AppUserModelID** attribute - This includes Windows desktop applications that have a known application user model ID. Use a **Tile** tag with the **AppUserModelID** attribute set to the app's application user model ID.
- Apps without a **AppUserModelID** attribute - For these apps, you must create a .lnk file that points to the installed app and place the .lnk file in the `%ALLUSERSPROFILE%\Microsoft\Windows\Start Menu\Programs` directory. Use a **DesktopApplicationTile** tag with the **LinkFilePath** attribute set to the .lnk file name and path.

The following example shows how to modify your LayoutModification.xml file to add both kinds of apps to the system area in Start:

```

<LayoutModificationTemplate
 xmlns="http://schemas.microsoft.com/Start/2014/LayoutModification"
 xmlns:defaultlayout="http://schemas.microsoft.com/Start/2014/FullDefaultLayout"
 xmlns:start="http://schemas.microsoft.com/Start/2014/StartLayout"
 Version="1">
 <TopMFUApps>
 <Tile AppUserModelID="Microsoft.WindowsCalculator_8wekyb3d8bbwe!App" />
 <Tile AppUserModelID="Microsoft.Getstarted_8wekyb3d8bbwe!App" />
 <DesktopApplicationTile LinkFilePath="%ALLUSERSPROFILE%\Microsoft\Windows\Start
Menu\Programs\Win32App.lnk" />
 </TopMFUApps>
</LayoutModificationTemplate>

```

#### AppendOfficeSuite

You can use the **AppendOfficeSuite** tag to add the in-box installed Office suite of apps to Start.

##### NOTE

The OEM must have installed Office for this tag to work.

The following example shows how to add the **AppendOfficeSuite** tag to your LayoutModification.xml file to append the full Universal Office suite to Start:

```

<LayoutModificationTemplate
 xmlns="http://schemas.microsoft.com/Start/2014/LayoutModification"
 xmlns:defaultlayout="http://schemas.microsoft.com/Start/2014/FullDefaultLayout"
 xmlns:start="http://schemas.microsoft.com/Start/2014/StartLayout"
 Version="1">
 <AppendOfficeSuite/>
</LayoutModificationTemplate>

```

#### AppendDownloadOfficeTile

You can use the **AppendDownloadOfficeTile** tag to append the Office trial installer to Start. This tag adds the Download Office tile to Start and the download tile will appear at the bottom right-hand side of the second group.

##### NOTE

The OEM must have installed the Office trial installer for this tag to work.

The following example shows how to add the **AppendDownloadOfficeTile** tag to your LayoutModification.xml file:

```

<LayoutModificationTemplate
 xmlns="http://schemas.microsoft.com/Start/2014/LayoutModification"
 xmlns:defaultlayout="http://schemas.microsoft.com/Start/2014/FullDefaultLayout"
 xmlns:start="http://schemas.microsoft.com/Start/2014/StartLayout"
 Version="1">
 <AppendDownloadOfficeTile/>
</LayoutModificationTemplate>

```

## Sample LayoutModification.xml

The following sample LayoutModification.xml shows how you can configure the Start layout for devices running Windows 10 for desktop editions:

```
</LayoutModificationTemplate>
```

```
\LayoutModificationTemplate
 xmlns="http://schemas.microsoft.com/Start/2014/LayoutModification"
 xmlns:defaultlayout="http://schemas.microsoft.com/Start/2014/FullDefaultLayout"
 xmlns:start="http://schemas.microsoft.com/Start/2014/StartLayout"
 Version="1">
<RequiredStartGroupsCollection>
 <RequiredStartGroups>
 <Region="DE|ES|FR|GB|IT|US">
 <AppendGroup
 Name="Fabrikam Group 1">
 <start:Tile
 AppUserModelID="Microsoft.Office.Word_8wekyb3d8bbwe!microsoft.word"
 Size="2x2"
 Row="0"
 Column="0"/>
 <start:DesktopApplicationTile
 DesktopApplicationID="Microsoft.Windows.Explorer"
 Size="2x2"
 Row="0"
 Column="2"/>
 <start:Tile
 AppUserModelID="Microsoft.Office.Excel_8wekyb3d8bbwe!microsoft.excel"
 Size="2x2"
 Row="0"
 Column="4"/>
 </AppendGroup>
 <AppendGroup
 Name="Fabrikam Group 2">
 <start:Tile
 AppUserModelID="Microsoft.Reader_8wekyb3d8bbwe!Microsoft.Reader"
 Size="2x2"
 Row="0"
 Column="0"/>
 <start:DesktopApplicationTile
 DesktopApplicationID="http://www.bing.com/"
 Size="2x2"
 Row="0"
 Column="2"/>
 <start:DesktopApplicationTile
 DesktopApplicationLinkPath="%ALLUSERSPROFILE%\Microsoft\Windows\Start
Menu\Programs\Accessories\Paint.lnk"
 Size="2x2"
 Row="0"
 Column="4"/>
 </AppendGroup>
 </RequiredStartGroups>
 <RequiredStartGroups>
 <AppendGroup
 Name="Fabrikam Group 1">
 <start:Tile
 AppUserModelID="Microsoft.Office.Word_8wekyb3d8bbwe!microsoft.word"
 Size="2x2"
 Row="0"
 Column="0"/>
 <start:SecondaryTile
 AppUserModelID="Microsoft.MicrosoftEdge_8wekyb3d8bbwe!MicrosoftEdge"
 TileID="FabrikamWeblinkTile"
 Arguments="http://www.fabrikam.com"
 DisplayName="Fabrikam"
 Square150x150LogoUri="ms-appx:///Assets/MicrosoftEdgeSquare150x150.png"
 ShowNameOnSquare150x150Logo="true"
 BackgroundColor="#FF112233"
 Size="2x2"
 Row="0"
 Column="2"/>
 </AppendGroup>
 </RequiredStartGroups>
 </RequiredStartGroupsCollection>
 <TopMFUApps>
```

```
<file AppUserModelID="MICROSOFT.WindowsCalculator_8wekyb3ddve!App" />
</TopMFUApps>
</LayoutModificationTemplate>
```

## Use Windows Provisioning multivariant support

The Windows Provisioning multivariant capability allows you to declare target conditions that, when met, supply specific customizations for each variant condition. For Start customization, you can create specific layouts for each variant that you have. To do this, you must create a separate LayoutModification.xml file for each variant that you want to support and then include these in your provisioning package. For more information on how to do this, see [Create a provisioning package with multivariant settings](#).

The provisioning engine chooses the right customization file based on the target conditions that were met, adds the file in the location that's specified for the setting, and then uses the specific file to customize Start. To differentiate between layouts, you can add modifiers to the LayoutModification.xml filename such as "LayoutCustomization1". Regardless of the modifier that you use, the provisioning engine will always output "LayoutCustomization.xml" so that the operating system has a consistent file name to query against.

For example, if you want to ensure that there's a specific layout for a certain condition, you can:

1. Create a specific layout customization file and then name it LayoutCustomization1.xml.
2. Include the file as part of your provisioning package.
3. Create your multivariant target and reference the XML file within the target condition in the main customization XML file.

The following example shows what the overall customization file might look like with multivariant support for Start:

```

<?xml version="1.0" encoding="utf-8"?>
<WindowsCustomizations>
 <PackageConfig xmlns="urn:schemas-Microsoft-com:Windows-ICD-Package-Config.v1.0">
 <ID>{6aaa4dfa-00d7-4aaa-8adf-73c6a7e2501e}</ID>
 <Name>My Provisioning Package</Name>
 <Version>1.0</Version>
 <OwnerType>OEM</OwnerType>
 <Rank>50</Rank>
 </PackageConfig>
 <Settings xmlns="urn:schemas-microsoft-com:windows-provisioning">
 <Customizations>
 <Targets>
 <Target Id="Processor ABC">
 <TargetState>
 <TargetState>
 <Condition Name="ProcessorName" Value="Pattern:.*Celeron.*" />
 <Condition Name="ProcessorType" Value="Pattern:.*I|intel.*" />
 </TargetState>
 </TargetState>
 </Target>
 </Targets>
 <Common>
 <Settings>
 <Policies>
 <AllowBrowser>1</AllowBrowser>
 <AllowCamera>1</AllowCamera>
 <AllowBluetooth>1</AllowBluetooth>
 </Policies>
 <HotSpot>
 <Enabled>1</Enabled>
 </HotSpot>
 </Settings>
 </Common>
 <Variant>
 <TargetRefs>
 <TargetRef Id="Processor ABC" />
 </TargetRefs>
 <Settings>
 <StartLayout>c:\users\</StartLayout>
<userprofile>\appdata\local\Microsoft\Windows\Shell\LayoutCustomization1.XML</StartLayout>
 <HotSpot>
 <Enabled>1</Enabled>
 </HotSpot>
 </Settings>
 </Variant>
 </Customizations>
 </Settings>
</WindowsCustomizations>

```

When the condition is met, the provisioning engine takes the XML file and places it in the location that the operating system has set and then the Start subsystem reads the file and applies the specific customized layout.

You must repeat this process for all variants that you want to support so that each variant can have a distinct layout for each of the conditions and targets that need to be supported. For example, if you add a **Language** condition, you can create a Start layout that has its own localized group.

## Add the LayoutModification.xml file to the device

Once you have created your LayoutModification.xml file to customize devices that will run Windows 10 for desktop editions, you can use Windows ICD methods to add the XML file to the device.

1. In the **Available customizations** pane, expand **Runtime settings**, select **Start** and then click the **StartLayout** setting.

2. In the middle pane, click **Browse** to open File Explorer.
3. In the File Explorer window, navigate to the location where you saved your LayoutModification.xml file.
4. Select the file and then click **Open**.

This should set the value of **StartLayout**. The setting appears in the **Selected customizations** pane.

**NOTE**

There is currently no way to add the .url and .lnk files through Windows ICD.

Once you have created the LayoutModification.xml file and it is present in the device, the system overrides the base default layout and any Unattend settings used to customize Start.

## Related topics

- [Manage Windows 10 Start and taskbar layout](#)
- [Configure Windows 10 taskbar](#)
- [Customize and export Start layout](#)
- [Add image for secondary tiles](#)
- [Customize Windows 10 Start and taskbar with Group Policy](#)
- [Customize Windows 10 Start and taskbar with provisioning packages](#)
- [Customize Windows 10 Start and tasbkar with mobile device management \(MDM\)](#)
- [Changes to Start policies in Windows 10](#)
- [Start layout XML for mobile editions of Windows 10 \(reference\)](#)

# Customize Windows 10 Start and taskbar with Group Policy

10/17/2017 • 6 min to read • [Edit Online](#)

## Applies to

- Windows 10

**Looking for consumer information?** See [Customize the Start menu](#)

In Windows 10 Pro, Enterprise, and Education, you can use a Group Policy Object (GPO) to deploy a customized Start and taskbar layout to users in a domain. No reimaging is required, and the layout can be updated simply by overwriting the .xml file that contains the layout. This enables you to customize Start and taskbar layouts for different departments or organizations, with minimal management overhead.

This topic describes how to update Group Policy settings to display a customized Start and taskbar layout when the users sign in. By creating a domain-based GPO with these settings, you can deploy a customized Start and taskbar layout to users in a domain.

### WARNING

When a full Start layout is applied with this method, the users cannot pin, unpin, or uninstall apps from Start. Users can view and open all apps in the **All Apps** view, but they cannot pin any apps to Start. When a partial Start layout is applied, the contents of the specified tile groups cannot be changed, but users can move those groups, and can also create and customize their own groups. When you apply a taskbar layout, users will still be able to pin and unpin apps, and change the order of pinned apps.

**Before you begin:** [Customize and export Start layout](#)

## Operating system requirements

Start and taskbar layout control using Group Policy is supported in Windows 10 Enterprise and Windows 10 Education, version 1607. Start and taskbar layout control is supported in Windows 10 Pro in Windows 10, version 1703.

The GPO can be configured from any computer on which the necessary ADMX and ADML files (StartMenu.admx and StartMenu.adml) for Windows 10 are installed. In Group Policy, ADMX files are used to define Registry-based policy settings in the Administrative Templates category. To find out how to create a central store for Administrative Templates files, see [article 929841, written for Windows Vista and still applicable](#) in the Microsoft Knowledge Base.

## How Start layout control works

Three features enable Start and taskbar layout control:

- The **Export-StartLayout** cmdlet in Windows PowerShell exports a description of the current Start layout in .xml file format.

**NOTE**

To import the layout of Start to a mounted Windows image, use the [Import-StartLayout](#) cmdlet.

- You can modify the **Start.xml** file to include `<CustomTaskbarLayoutCollection>` or create an .xml file just for the taskbar configuration.
- In Group Policy, you use the **Start Layout** settings for the **Start Menu and Taskbar** administrative template to set a Start and taskbar layout from an .xml file when the policy is applied. The Group Policy object doesn't support an empty tile layout, so the default tile layout for Windows is loaded in that case.

**NOTE**

To learn how customize Start to include your line-of-business apps when you deploy Windows 10, see [Customize the Windows 10 Start layout](#).

## Use Group Policy to apply a customized Start layout in a domain

To apply the Start and taskbar layout to users in a domain, use the Group Policy Management Console (GPMC) to configure a domain-based Group Policy Object (GPO) that sets **Start Layout** policy settings in the **Start Menu and Taskbar** administrative template for users in a domain.

The GPO applies the Start and taskbar layout at the next user sign-in. Each time the user signs in, the timestamp of the .xml file with the Start and taskbar layout is checked and if a newer version of the file is available, the settings in the latest version of the file are applied.

The GPO can be configured from any computer on which the necessary ADMX and ADML files (StartMenu.admx and StartMenu.adml) for Windows 10 are installed.

The .xml file with the Start and taskbar layout must be located on shared network storage that is available to the users' computers when they sign in and the users must have Read-only access to the file. If the file is not available when the first user signs in, Start and the taskbar are not customized during the session, but the user will be prevented from making changes to Start. On subsequent sign-ins, if the file is available at sign-in, the layout it contains will be applied to the user's Start and taskbar.

For information about deploying GPOs in a domain, see [Working with Group Policy Objects](#).

## Use Group Policy to apply a customized Start layout on the local computer

You can use the Local Group Policy Editor to provide a customized Start and taskbar layout for any user who signs in on the local computer. To display the customized Start and taskbar layout for any user who signs in, configure **Start Layout** policy settings for the **Start Menu and Taskbar** administrative template. You can use the **Start Menu and Taskbar** administrative template in **User Configuration** or **Computer Configuration**.

**NOTE**

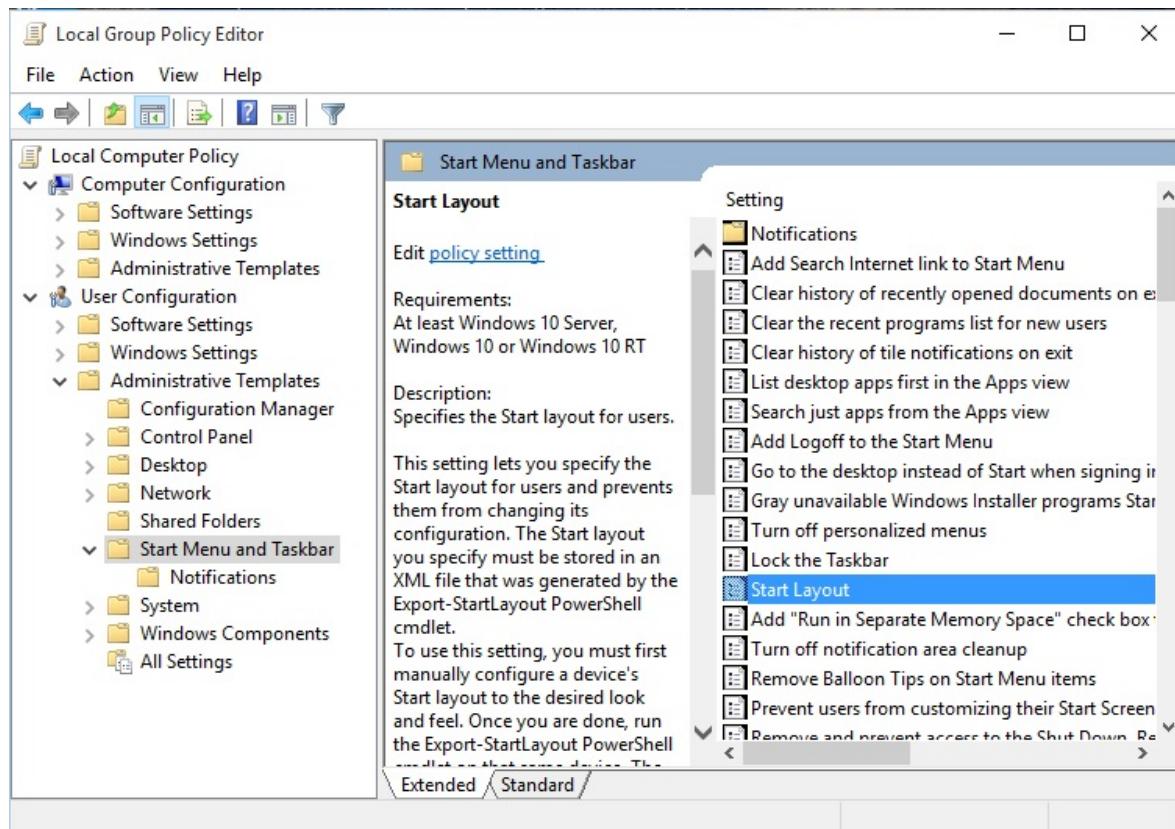
This procedure applies the policy settings on the local computer only. For information about deploying the Start and taskbar layout to users in a domain, see [Use Group Policy to deploy a customized Start layout in a domain](#).

This procedure creates a Local Group Policy that applies to all users on the computer. To configure Local Group Policy that applies to a specific user or group on the computer, see [Step-by-Step Guide to Managing Multiple Local Group Policy Objects](#). The guide was written for Windows Vista and the procedures still apply to Windows 10.

This procedure adds the customized Start and taskbar layout to the user configuration, which overrides any Start layout settings in the local computer configuration when a user signs in on the computer.

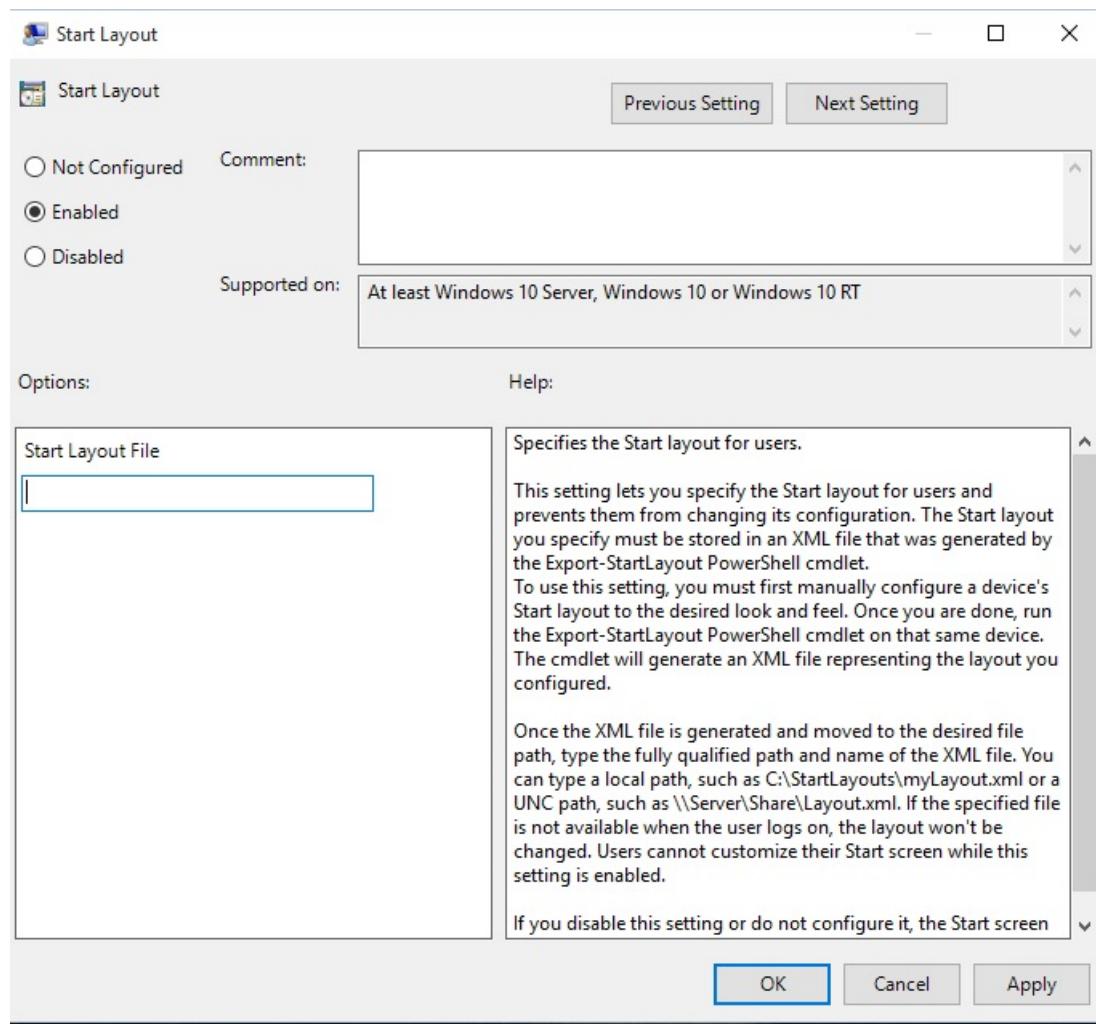
### To configure Start Layout policy settings in Local Group Policy Editor

1. On the test computer, press the Windows key, type **gpedit**, and then select **Edit group policy (Control panel)**.
2. Go to **User Configuration** or **Computer Configuration** > **Administrative Templates** > **Start Menu and Taskbar**.



3. Right-click **Start Layout** in the right pane, and click **Edit**.

This opens the **Start Layout** policy settings.



4. Enter the following settings, and then click **OK**:

- Select **Enabled**.
- Under **Options**, specify the path to the .xml file that contains the Start and taskbar layout. For example, type **C:\Users\Test01\StartScreenMarketing.xml**.
- Optionally, enter a comment to identify the Start and taskbar layout.

#### IMPORTANT

If you disable Start Layout policy settings that have been in effect and then re-enable the policy, users will not be able to make changes to Start, however the layout in the .xml file will not be reapplied unless the file has been updated. In Windows PowerShell, you can update the timestamp on a file by running the following command:

```
(ls <path>).LastWriteTime = Get-Date
```

## Update a customized Start layout

After you use Group Policy to apply a customized Start and taskbar layout on a computer or in a domain, you can update the layout simply by replacing the .xml file that is specified in the Start Layout policy settings with a file with a newer timestamp.

## Related topics

- [Manage Windows 10 Start and taskbar layout](#)
- [Configure Windows 10 taskbar](#)

- [Customize and export Start layout](#)
- [Add image for secondary tiles](#)
- [Start layout XML for desktop editions of Windows 10 \(reference\)](#)
- [Customize Windows 10 Start and taskbar with provisioning packages](#)
- [Customize Windows 10 Start and tasbkar with mobile device management \(MDM\)](#)
- [Changes to Start policies in Windows 10](#)

Not finding content you need? Windows 10 users, tell us what you want on [Feedback Hub](#).

# Customize Windows 10 Start and taskbar with provisioning packages

10/17/2017 • 5 min to read • [Edit Online](#)

## Applies to

- Windows 10

**Looking for consumer information? [Customize the Start menu](#)**

In Windows 10 Pro, Windows 10 Enterprise, and Windows 10 Education, version 1703, you can use a provisioning package that you create with Windows Configuration Designer to deploy a customized Start and taskbar layout to users. No reimaging is required, and the Start and taskbar layout can be updated simply by overwriting the .xml file that contains the layout. The provisioning package can be applied to a running device. This enables you to customize Start and taskbar layouts for different departments or organizations, with minimal management overhead.

### IMPORTANT

If you use a provisioning package to configure the taskbar, your configuration will be reapplied each time the explorer.exe process restarts. If your configuration pins an app and the user unpins that app, the user's change will be overwritten the next time the configuration is applied. To apply a taskbar configuration and allow users to make changes that will persist, apply your configuration by using Group Policy.

**Before you begin:** [Customize and export Start layout](#) for desktop editions.

## How Start layout control works

Three features enable Start and taskbar layout control:

- The **Export-StartLayout** cmdlet in Windows PowerShell exports a description of the current Start layout in .xml file format.

### NOTE

To import the layout of Start to a mounted Windows image, use the **Import-StartLayout** cmdlet.

- You can modify the Start .xml file to include `<CustomTaskbarLayoutCollection>` or create an .xml file just for the taskbar configuration.
- In Windows Configuration Designer, you use the **Policies/Start/StartLayout** setting to provide the contents of the .xml file that defines the Start and taskbar layout.

## Prepare the Start layout XML file

The **Export-StartLayout** cmdlet produces an XML file. Because Windows Configuration Designer produces a customizations.xml file that contains the configuration settings, adding the Start layout section to the customizations.xml file directly would result in an XML file embedded in an XML file. Before you add the Start layout section to the customizations.xml file, you must replace the markup characters in your layout.xml with

escape characters.

1. Copy the contents of layout.xml into an online tool that escapes characters.
2. During the procedure to create a provisioning package, you will copy the text with the escape characters and paste it in the customizations.xml file for your project.

## Create a provisioning package that contains a customized Start layout

Use the Windows Configuration Designer tool to create a provisioning package. [Learn how to install Windows Configuration Designer](#).

### IMPORTANT

When you build a provisioning package, you may include sensitive information in the project files and in the provisioning package (.ppkg) file. Although you have the option to encrypt the .ppkg file, project files are not encrypted. You should store the project files in a secure location and delete the project files when they are no longer needed.

1. Open Windows Configuration Designer (by default, %systemdrive%\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Imaging and Configuration Designer\x86\ICD.exe).
2. Choose **Advanced provisioning**.
3. Name your project, and click **Next**.
4. Choose **All Windows desktop editions** and click **Next**.
5. On **New project**, click **Finish**. The workspace for your package opens.
6. Expand **Runtime settings > Policies > Start**, and click **StartLayout**.

### TIP

If **Start** is not listed, check the type of settings you selected in step 4. You must create the project using settings for **All Windows desktop editions**.

7. Enter **layout.xml**. This value creates a placeholder in the customizations.xml file that you will replace with the contents of the layout.xml file in a later step.
8. Save your project and close Windows Configuration Designer.
9. In File Explorer, open the project's directory. (The default location is C:\Users\user name\Documents\Windows Imaging and Configuration Designer (WICD)\project name)
10. Open the customizations.xml file in a text editor. The **<Customizations>** section will look like this:

```
<Customizations>
 <Common>
 <Policies>
 <Start>
 <StartLayout>layout.xml</StartLayout>
 </Start>
 </Policies>
 </Common>
</Customizations>
```

11. Replace **layout.xml** with the text from the layout.xml file, [with markup characters replaced with escape characters](#).
12. Save and close the customizations.xml file.

13. Open Windows Configuration Designer and open your project.
14. On the **File** menu, select **Save**.
15. On the **Export** menu, select **Provisioning package**.
16. Change **Owner** to **IT Admin**, which will set the precedence of this provisioning package higher than provisioning packages applied to this device from other sources, and then select **Next**.
17. Optional. In the **Provisioning package security** window, you can choose to encrypt the package and enable package signing.
  - **Enable package encryption** - If you select this option, an auto-generated password will be shown on the screen.
  - **Enable package signing** - If you select this option, you must select a valid certificate to use for signing the package. You can specify the certificate by clicking **Browse** and choosing the certificate you want to use to sign the package.
18. Click **Next** to specify the output location where you want the provisioning package to go when it's built. By default, Windows Imaging and Configuration Designer (ICD) uses the project folder as the output location.

Optionally, you can click **Browse** to change the default output location.
19. Click **Next**.
20. Click **Build** to start building the package. The provisioning package doesn't take long to build. The project information is displayed in the build page and the progress bar indicates the build status.

If you need to cancel the build, click **Cancel**. This cancels the current build process, closes the wizard, and takes you back to the **Customizations Page**.
21. If your build fails, an error message will show up that includes a link to the project folder. You can scan the logs to determine what caused the error. Once you fix the issue, try building the package again.

If your build is successful, the name of the provisioning package, output directory, and project directory will be shown.

  - If you choose, you can build the provisioning package again and pick a different path for the output package. To do this, click **Back** to change the output package name and path, and then click **Next** to start another build.
  - If you are done, click **Finish** to close the wizard and go back to the **Customizations Page**.
22. Copy the provisioning package to the target device.
23. Double-click the ppkg file and allow it to install.

## Related topics

- [Manage Windows 10 Start and taskbar layout](#)
- [Configure Windows 10 taskbar](#)
- [Customize and export Start layout](#)
- [Add image for secondary tiles](#)
- [Start layout XML for desktop editions of Windows 10 \(reference\)](#)
- [Customize Windows 10 Start and taskbar with Group Policy](#)
- [Customize Windows 10 Start and tasbkar with mobile device management \(MDM\)](#)
- [Changes to Start policies in Windows 10](#)

# Customize Windows 10 Start and taskbar with mobile device management (MDM)

7/28/2017 • 3 min to read • [Edit Online](#)

## Applies to

- Windows 10

**Looking for consumer information?** [Customize the Start menu](#)

In Windows 10 Pro, Windows 10 Enterprise, and Windows 10 Education, you can use a mobile device management (MDM) policy to deploy a customized Start and taskbar layout to users. No reimaging is required, and the layout can be updated simply by overwriting the .xml file that contains the layout. This enables you to customize Start layouts for different departments or organizations, with minimal management overhead.

### NOTE

Support for applying a customized taskbar using MDM is added in Windows 10, version 1703.

**Before you begin:** [Customize and export Start layout](#) for desktop editions.

### WARNING

When a full Start layout is applied with this method, the users cannot pin, unpin, or uninstall apps from Start. Users can view and open all apps in the **All Apps** view, but they cannot pin any apps to Start. When a partial Start layout is applied, the contents of the specified tile groups cannot be changed, but users can move those groups, and can also create and customize their own groups.

## How Start layout control works

Two features enable Start layout control:

- The **Export-StartLayout** cmdlet in Windows PowerShell exports a description of the current Start layout in .xml file format.

### NOTE

To import the layout of Start to a mounted Windows image, use the **Import-StartLayout** cmdlet.

- In MDM, you set the path to the .xml file that defines the Start layout using an OMA-URI setting, which is based on the [Policy configuration service provider \(CSP\)](#).

## Create a policy for your customized Start layout

This example uses Microsoft Intune to configure an MDM policy that applies a customized Start layout. See the documentation for your MDM solution for help in applying the policy.

- In the Start layout file created when you ran **Export-StartLayout**, replace markup characters with escape characters, and save the file. (You can replace the characters manually or use an online tool.)

Example of a layout file produced by Export-StartLayout:

**XML**

```
<LayoutModificationTemplate Version="1"
 xmlns="http://schemas.microsoft.com/Start/2014/LayoutModification">
 <DefaultLayoutOverride>
 <StartLayoutCollection>
 <defaultlayout:StartLayout GroupCellWidth="6"
 xmlns:defaultlayout="http://schemas.microsoft.com/Start/2014/FullDefaultLayout">
 <start:Group Name="Life at a glance"
 xmlns:start="http://schemas.microsoft.com/Start/2014/StartLayout">
 <start:Tile Size="2x2" Column="0" Row="0"
 AppUserModelID="Microsoft.MicrosoftEdge_8wekyb3d8bbwe!MicrosoftEdge" />
 <start:Tile Size="2x2" Column="4" Row="0"
 AppUserModelID="Microsoft.Windows.Cortana_cw5n1h2txyewy!CortanaUI" />
 <start:Tile Size="2x2" Column="2" Row="0"
 AppUserModelID="Microsoft.BingWeather_8wekyb3d8bbwe!App" />
 </start:Group>
 </defaultlayout:StartLayout>
 </StartLayoutCollection>
 </DefaultLayoutOverride>
</LayoutModificationTemplate>
```

Example of the same layout file with escape characters replacing the markup characters:

```
<wdcml:p xmlns:wdcml="http://microsoft.com/wdcml">Example of a layout
file produced by Export-StartLayout:</wdcml:p><wdcml:snippet
xmlns:wdcml="http://microsoft.com/wdcml"><!
[CDATA[<LayoutModificationTemplate Version="1"
xmlns="http://schemas.microsoft.com/Start/2014/LayoutModification">
 <DefaultLayoutOverride>
 <StartLayoutCollection>
 <defaultlayout:StartLayout GroupCellWidth="6"
 xmlns:defaultlayout="http://schemas.microsoft.com/Start/2014/FullDefaultLayout">
 <start:Group Name="Life at a glance"
 xmlns:start="http://schemas.microsoft.com/Start/2014/StartLayout">
 <start:Tile Size="2x2" Column="0" Row="0"
 AppUserModelID="Microsoft.MicrosoftEdge_8wekyb3d8bbwe!MicrosoftEdge" />
 <start:Tile Size="2x2" Column="4" Row="0"
 AppUserModelID="Microsoft.Windows.Cortana_cw5n1h2txyewy!CortanaUI" />
 <start:Tile Size="2x2" Column="2" Row="0"
 AppUserModelID="Microsoft.BingWeather_8wekyb3d8bbwe!App" />
 </start:Group>
 </defaultlayout:StartLayout>
 </StartLayoutCollection>
 </DefaultLayoutOverride>
</LayoutModificationTemplate>]]></wdcml:snippet>
```

1. In the Microsoft Intune administration console, click **Policy** > **Add Policy**.
2. Under **Windows**, choose a **Custom Configuration (Windows 10 Desktop and Mobile and later)** policy.
3. Enter a name (mandatory) and description (optional) for the policy.
4. In the **OMA-URI Settings** section, click **Add**.
5. In **Add or Edit OMA-URI Setting**, enter the following information.

ITEM	INFORMATION
<b>Setting name</b>	Enter a unique name for the OMA-URI setting to help you identify it in the list of settings.
<b>Setting description</b>	Provide a description that gives an overview of the setting and other relevant information to help you locate it.
<b>Data type</b>	<b>String</b>
<b>OMA-URI (case sensitive)</b>	<code>./User/Vendor/MSFT/Policy/Config/Start/StartLayout</code>
<b>Value</b>	Paste the contents of the Start layout .xml file that you created.

6. Click **OK** to save the setting and return to the **Create Policy** page.

7. Click **Save Policy**.

## Related topics

- [Manage Windows 10 Start and taskbar layout](#)
- [Configure Windows 10 taskbar](#)
- [Customize and export Start layout](#)
- [Add image for secondary tiles](#)
- [Start layout XML for desktop editions of Windows 10 \(reference\)](#)
- [Customize Windows 10 Start and taskbar with Group Policy](#)
- [Customize Windows 10 Start and taskbar with provisioning packages](#)
- [Changes to Start policies in Windows 10](#)

# Changes to Group Policy settings for Windows 10 Start

7/28/2017 • 4 min to read • [Edit Online](#)

## Applies to

- Windows 10

Windows 10 has a brand new Start experience. As a result, there are changes to the Group Policy settings that you can use to manage Start. Some policy settings are new or changed, and some old Start policy settings still apply. Other Start policy settings no longer apply and are deprecated.

## Start policy settings supported for Windows 10 Pro, Windows 10 Enterprise, and Windows 10 Education

These policy settings are available in **Administrative Templates\Start Menu and Taskbar** under **User Configuration**.

POLICY	NOTES
Clear history of recently opened documents on exit	Documents that the user opens are tracked during the session. When the user signs off, the history of opened documents is deleted.
Do not allow pinning items in Jump Lists	Jump Lists are lists of recently opened items, such as files, folders, or websites, organized by the program that you use to open them. This policy prevents users from pinning items to any Jump List.
Do not display or track items in Jump Lists from remote locations	When this policy is applied, only items local on the computer are shown in Jump Lists.
Do not keep history of recently opened documents	Documents that the user opens are not tracked during the session.
Prevent changes to Taskbar and Start Menu Settings	In Windows 10, this disables all of the settings in <b>Settings &gt; Personalization &gt; Start</b> as well as the options in dialog available via right-click Taskbar > <b>Properties</b>
Prevent users from customizing their Start Screen	Use this policy in conjunction with a <a href="#">customized Start layout</a> to prevent users from changing it
Prevent users from uninstalling applications from Start	In Windows 10, this removes the uninstall button in the context menu. It does not prevent users from uninstalling the app through other entry points (e.g. PowerShell)
Remove All Programs list from the Start menu	In Windows 10, this removes the <b>All apps</b> button.

POLICY	NOTES
Remove and prevent access to the Shut Down, Restart, Sleep, and Hibernate commands	This removes the Shut Down, Restart, Sleep, and Hibernate commands from the Start Menu, Start Menu power button, CTRL+ALT+DEL screen, and Alt+F4 Shut Down Windows menu.
Remove common program groups from Start Menu	As in earlier versions of Windows, this removes apps specified in the All Users profile from Start
Remove frequent programs list from the Start Menu	In Windows 10, this removes the top left <b>Most used</b> group of apps.
Remove Logoff on the Start Menu	<b>Logoff</b> has been changed to <b>Sign Out</b> in the user interface, however the functionality is the same.
Remove pinned programs list from the Start Menu	In Windows 10, this removes the bottom left group of apps (by default, only File Explorer and Settings are pinned).
Show "Run as different user" command on Start	This enables the <b>Run as different user</b> option in the right-click menu for apps.
Start Layout	This applies a specific Start layout, and it also prevents users from changing the layout. This policy can be configured in <b>User Configuration</b> or <b>Computer Configuration</b> . <div style="border: 1px solid black; padding: 10px; margin-top: 10px;"> <b>Note</b>            Start Layout policy setting applies only to Windows 10 Enterprise and Windows 10 Education.         </div>
Force Start to be either full screen size or menu size	This applies a specific size for Start.

## Deprecated Group Policy settings for Start

The Start policy settings listed below do not work on Windows 10. Most of them were deprecated in Windows 8 however a few more were deprecated in Windows 10. Deprecation in this case means that the policy setting will not work on Windows 10. The "Supported on" text for a policy setting will not list Windows 10. The policy settings are still in the Group Policy Management Console and can be used on the operating systems that they apply to.

POLICY	WHEN DEPRECATED
Go to the desktop instead of Start when signing in	Windows 10
List desktop apps first in the Apps view	Windows 10
Pin Apps to Start when installed (User or Computer)	Windows 10
Remove Default Programs link from the Start menu.	Windows 10
Remove Documents icon from Start Menu	Windows 10

POLICY	WHEN DEPRECATED
Remove programs on Settings menu	Windows 10
Remove Run menu from Start Menu	Windows 10
Remove the "Undock PC" button from the Start Menu	Windows 10
Search just apps from the Apps view	Windows 10
Show Start on the display the user is using when they press the Windows logo key	Windows 10
Show the Apps view automatically when the user goes to Start	Windows 10
Add the Run command to the Start Menu	Windows 8
Change Start Menu power button	Windows 8
Gray unavailable Windows Installer programs Start Menu shortcuts	Windows 8
Remove Downloads link from Start Menu	Windows 8
Remove Favorites menu from Start Menu	Windows 8
Remove Games link from Start Menu	Windows 8
Remove Help menu from Start Menu	Windows 8
Remove Homegroup link from Start Menu	Windows 8
Remove Music icon from Start Menu	Windows 8
Remove Network icon from Start Menu	Windows 8
Remove Pictures icon from Start Menu	Windows 8
Remove Recent Items menu from Start Menu	Windows 8
Remove Recorded TV link from Start Menu	Windows 8
Remove user folder link from Start Menu	Windows 8
Remove Videos link from Start Menu	Windows 8

## Related topics

- [Manage Windows 10 Start and taskbar layout](#)
- [Configure Windows 10 taskbar](#)
- [Customize and export Start layout](#)

- [Add image for secondary tiles](#)
- [Start layout XML for desktop editions of Windows 10 \(reference\)](#)
- [Customize Windows 10 Start and taskbar with Group Policy](#)
- [Customize Windows 10 Start and taskbar with provisioning packages](#)
- [Customize Windows 10 Start and tasbkar with mobile device management \(MDM\)](#)

# Cortana integration in your business or enterprise

10/5/2017 • 3 min to read • [Edit Online](#)

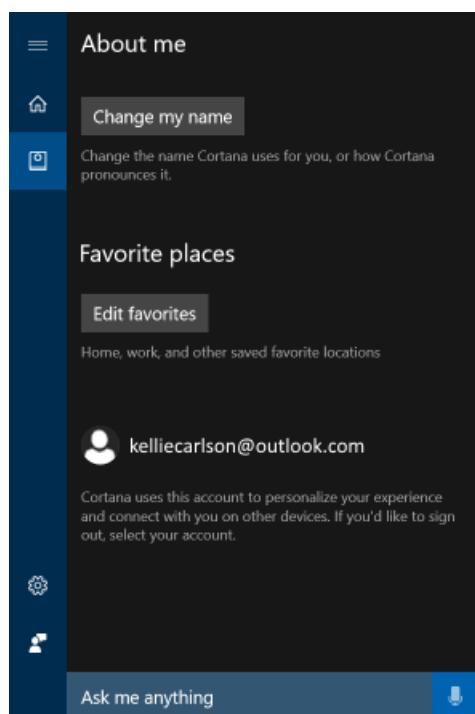
## Applies to:

- Windows 10, version 1703
- Windows 10 Mobile, version 1703

## Who is Cortana?

Cortana is Microsoft's personal digital assistant, who helps busy people get things done, even while at work. Cortana has powerful configuration options, specifically optimized for your business. By signing in with an Azure Active Directory (Azure AD) account, your employees can give Cortana access to their enterprise/work identity, while getting all the functionality Cortana provides to them outside of work.

Using Azure AD also means that you can remove an employee's profile (for example, when an employee leaves your organization) while respecting Windows Information Protection (WIP) policies and ignoring enterprise content, such as emails, calendar items, and people lists that are marked as enterprise data.



## Where is Cortana available for use in my organization?

You can use Cortana at work in all countries/regions where Cortana is supported for consumers. This includes the United States, United Kingdom, Canada, France, Italy, Germany, Spain, China, Japan, India, and Australia. As Cortana comes to more countries, she will also become available to enterprise customers.

Cortana is available on Windows 10, version 1703 and with limited functionality on Windows 10 Mobile, version 1703.

## Required hardware and software

Cortana requires the following hardware and software to successfully run the included scenario in your organization.

HARDWARE	DESCRIPTION
Microphone	For speech interaction with Cortana. If you don't have a microphone, you can still interact with Cortana by typing in the Cortana Search Box in the taskbar.
Windows Phone	For location-specific reminders. You can also use a desktop device to run through this scenario, but location accuracy is usually better on phones.
Desktop devices	For non-phone-related scenarios.
SOFTWARE	MINIMUM VERSION
Client operating system	<ul style="list-style-type: none"> <li>• <b>Desktop:</b> Windows 10, version 1703</li> <li>• <b>Mobile:</b> Windows 10 Mobile, version 1703 (with limited functionality)</li> </ul>
Azure Active Directory (Azure AD)	While all employees signing into Cortana need an Azure AD account; an Azure AD premium tenant isn't required.
Additional policies (Group Policy and Mobile Device Management (MDM))	<p>There is a rich set of policies that can be used to manage various aspects of Cortana. Most of these policies will limit the abilities of Cortana, but won't turn Cortana off.</p> <p>For example:</p> <p>If you turn <b>Location</b> off, Cortana won't be able to provide location-based reminders, such as reminding you to visit the mail room when you get to work.</p> <p>If you turn <b>Speech</b> off, your employees won't be able to use "Hello Cortana" for hands free usage or voice commands to easily ask for help.</p>
Windows Information Protection (WIP) (optional)	<p>If you want to secure the calendar, email, and contact info provided to Cortana on a device, you can use WIP. For more info about WIP, see <a href="#">Protect your enterprise data using Windows Information Protection (WIP)</a></p> <p>If you decide to use WIP, you must also have a management solution. This can be Microsoft Intune, Microsoft System Center Configuration Manager (version 1606 or later), or your current company-wide 3rd party mobile device management (MDM) solution.</p>

## Signing in using Azure AD

Your organization must have an Azure AD tenant and your employees' devices must all be Azure AD-joined for Cortana to work properly. For info about what an Azure AD tenant is, how to get your devices joined, and other Azure AD maintenance info, see [What is an Azure AD directory?](#)

## Cortana and privacy

We understand that there are some questions about Cortana and your organization's privacy, including concerns about what info is collected by Cortana, where the info is saved, how to manage what data is collected, how to turn Cortana off, how to opt completely out of data collection, and what info is shared with other Microsoft apps and

services. For more details about these concerns, see the [Cortana, Search, and privacy: FAQ](#) topic.

Cortana is covered under the [Microsoft Privacy Statement](#) and [Microsoft Services Agreement](#).

## See also

- [What is Cortana?](#)
- [Cortana and Windows](#)
- [Known issues for Windows Desktop Search and Cortana in Windows 10](#)
- [Cortana for developers](#)

# Testing scenarios using Cortana in your business or organization

10/5/2017 • 1 min to read • [Edit Online](#)

## Applies to:

- Windows 10, version 1703
- Windows 10 Mobile, version 1703

We've come up with a list of suggested testing scenarios that you can use to test Cortana in your organization.

After you complete all the scenarios, you should be able to:

- Sign-in to Cortana using Azure AD, manage entries in the notebook, and search for content across your device, Bing, and the cloud, using Cortana
- Perform a quick search with Cortana at work
- Set a reminder and have it remind you when you've reached a specific location
- Search for your upcoming meetings on your work calendar
- Send an email to a co-worker from your work email app
- Review a reminder suggested by Cortana based on what you've promised in email
- Use Windows Information Protection (WIP) to secure content on a device and then try to manage your organization's entries in the notebook

## IMPORTANT

The data created as part of these scenarios will be uploaded to Microsoft's Cloud to help Cortana learn and help your employees. This is the same info that Cortana uses in the consumer offering.

# Test scenario 1 - Sign-in to Azure AD and use Cortana to manage the notebook

10/5/2017 • 1 min to read • [Edit Online](#)

- Windows 10, version 1703
- Windows 10 Mobile, version 1703

## IMPORTANT

The data created as part of these scenarios will be uploaded to Microsoft's Cloud to help Cortana learn and help your employees. This is the same info that Cortana uses in the consumer offering.

This scenario turns on Azure AD and let's your employee use Cortana to manage an entry in the notebook.

## Turn on Azure AD

This process helps you to sign out of a Microsoft Account and to sign into an Azure AD account.

1. Click on the **Cortana** icon in the taskbar, click the **Notebook**, and then click **About Me**.
2. Click your email address.

A dialog box appears, showing the associated account info.

3. Click your email address again, and then click **Sign out**.

This signs out the Microsoft account, letting you continue to add and use the Azure AD account.

4. Click the **Search** box and then the **Notebook** icon in the left rail. This will start the sign-in request.
5. Click **Sign-In** and follow the instructions.
6. When you're asked to sign in, you'll need to choose an Azure AD account, which will look like `kelliecarlson@contoso.com`.

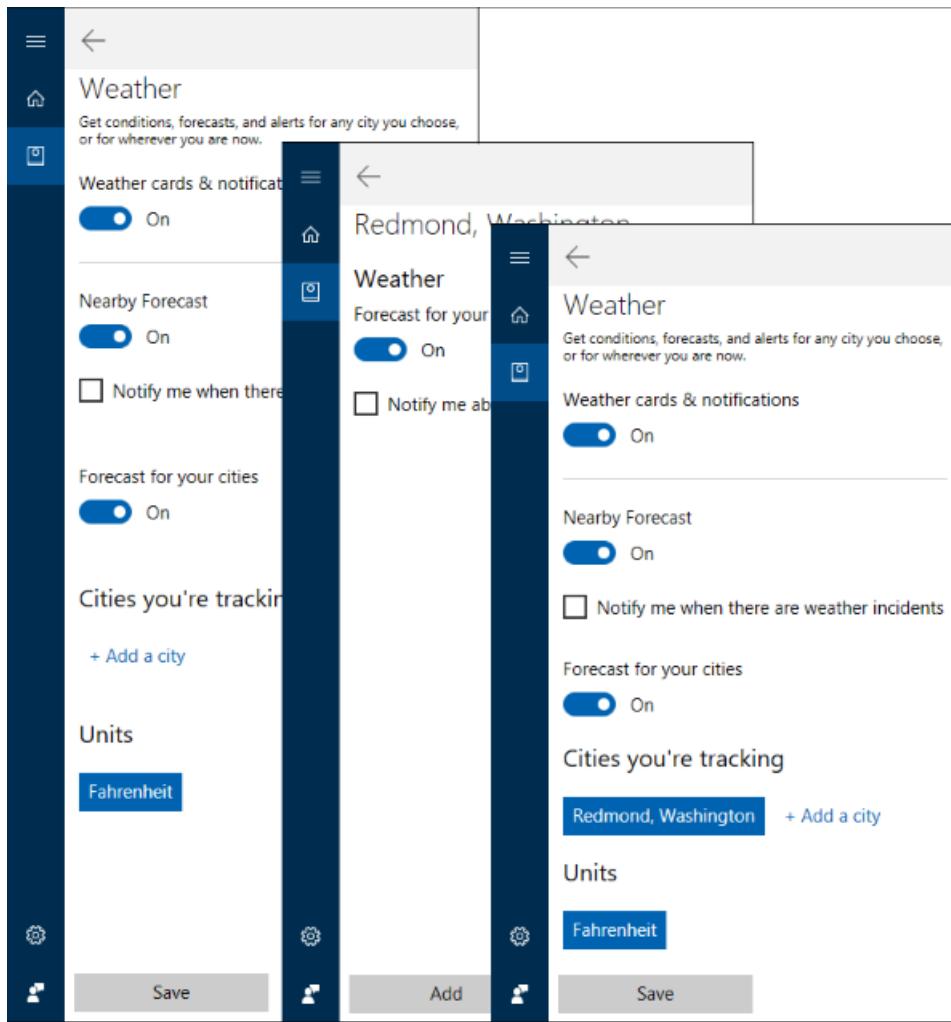
## IMPORTANT

If there's no Azure AD account listed, you'll need to go to **Windows Settings > Accounts > Email & app accounts**, and then click **Add a work or school account** to add it.

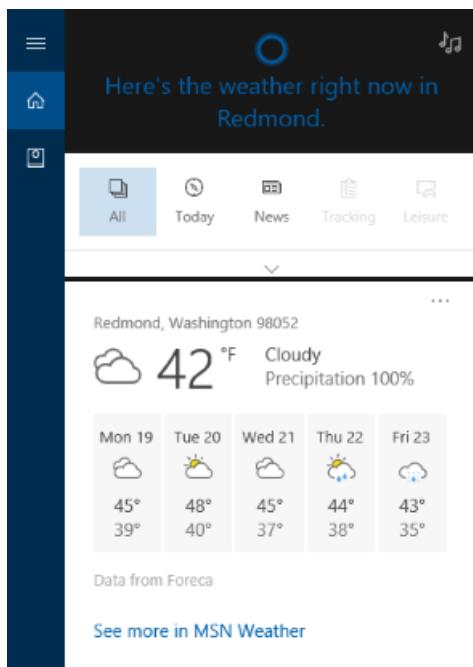
## Use Cortana to manage the notebook content

This process helps you to manage the content Cortana shows in your Notebook.

1. Click on the **Cortana** icon in the taskbar, click the **Notebook**, scroll down and click **Weather**.
2. In the **Weather** settings, scroll down to the **Cities your tracking** area, and then click **Add a city**.
3. Add *Redmond, Washington*, double-click the search result, click **Add**, and then click **Save**.



4. Click on the **Home** icon and scroll to the weather forecast for Redmond, Washington.



# Test scenario 2 - Perform a quick search with Cortana at work

10/5/2017 • 1 min to read • [Edit Online](#)

- Windows 10, version 1703
- Windows 10 Mobile, version 1703

## IMPORTANT

The data created as part of these scenarios will be uploaded to Microsoft's Cloud to help Cortana learn and help your employees. This is the same info that Cortana uses in the consumer offering.

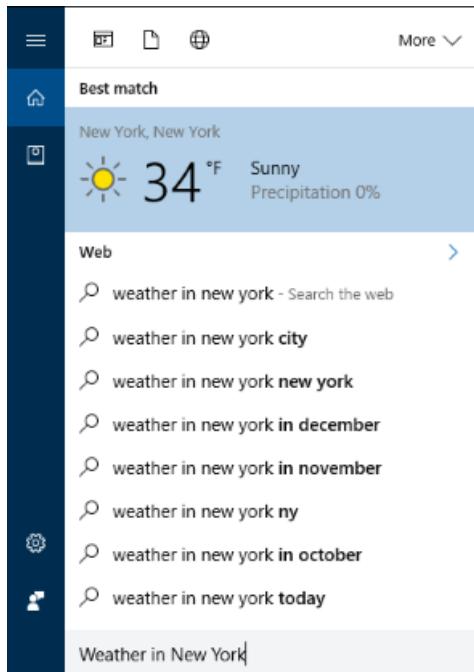
This scenario helps you perform a quick search using Cortana, both by typing and through voice commands.

## Search using Cortana

This process helps you use Cortana at work to perform a quick search.

1. Click on the **Cortana** icon in the taskbar, and then click in the **Search** bar.
2. Type *Weather in New York*.

You should see the weather in New York, New York at the top of the search results.



## Search with Cortana, by using voice commands

This process helps you to use Cortana at work and voice commands to perform a quick search.

1. Click on the **Cortana** icon in the taskbar, and then click the **Microphone** icon (to the right of the **Search** box).
2. Say *What's the weather in Chicago?* Cortana tells you and shows you the current weather in Chicago.

≡

Here's the current weather.

Chicago, IL  
Updated January 4, 10:00 AM

 13 °F 22° Precipitation: 10%  
C 10° Wind: 11 MPH

Mostly sunny · Wed 4, 10:00 AM

Wed 4	Thu 5	Fri 6	Sat 7	Sun
22° 10° ↳ 10%	14° 5° ↳ 30%	13° 11° ↳ 10%	21° 5° ↳ 0%	17° 13° ↳ 0%

15° 18° 13° 10° 10°  
11 AM 4 PM 9 PM 2 AM 7 AM  
↳ 10% ↳ 10% ↳ 20%

Data from Foreca

What's the weather in Chicago? 

# Test scenario 3 - Set a reminder for a specific location using Cortana at work

10/5/2017 • 2 min to read • [Edit Online](#)

- Windows 10, version 1703
- Windows 10 Mobile, version 1703

## IMPORTANT

The data created as part of these scenarios will be uploaded to Microsoft's Cloud to help Cortana learn and help your employees. This is the same info that Cortana uses in the consumer offering.

This scenario helps you set up, review, and edit a reminder based on a location. For example, reminding yourself to grab your expense report receipts before you leave the house.

## NOTE

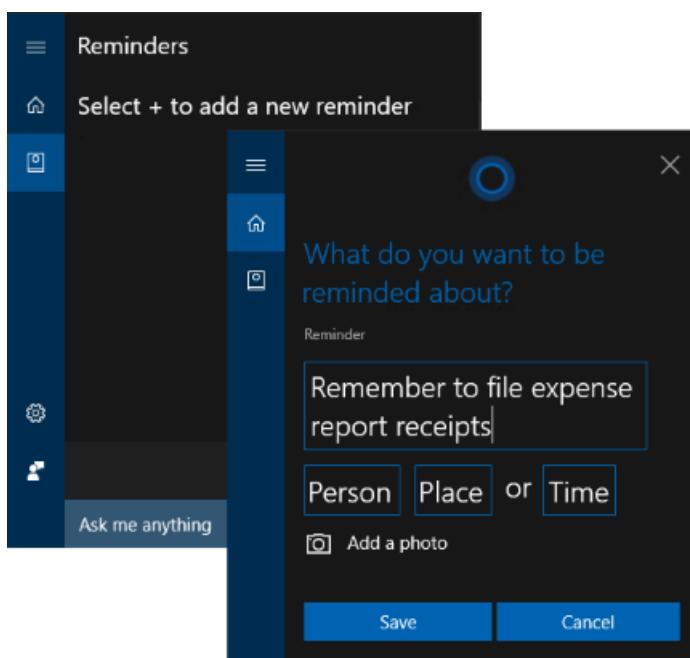
You can set each reminder location individually as you create the reminders, or you can go into the **About me** screen and add both **Work** and **Home** addresses as favorites. Make sure that you use real addresses since you'll need to go to these locations to complete your testing scenario.

Additionally, if you've turned on the **Meeting & reminder cards & notifications** option (in the **Meetings & reminders** option of your Notebook), you'll also see your pending reminders on the Cortana **Home** page.

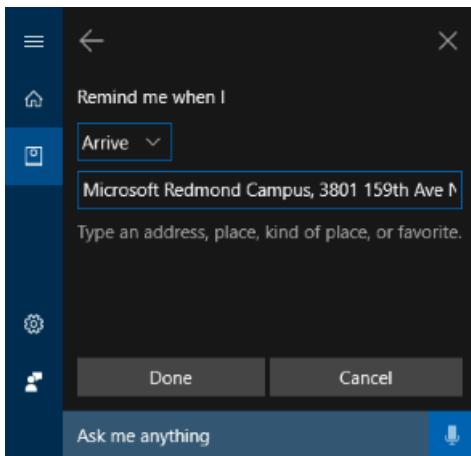
## Create a reminder for a specific location

This process helps you to create a reminder based on a specific location.

1. Click on the **Cortana** icon in the taskbar, click on the **Notebook** icon, and then click **Reminders**.
2. Click the + sign, add a subject for your reminder, such as *Remember to file expense report receipts*, and then click **Place**.



3. Choose **Arrive** from the drop-down box, and then type a location to associate with your reminder. For example, you can use the physical address of where you work. Just make sure you can physically get to your location, so you can test the reminder.



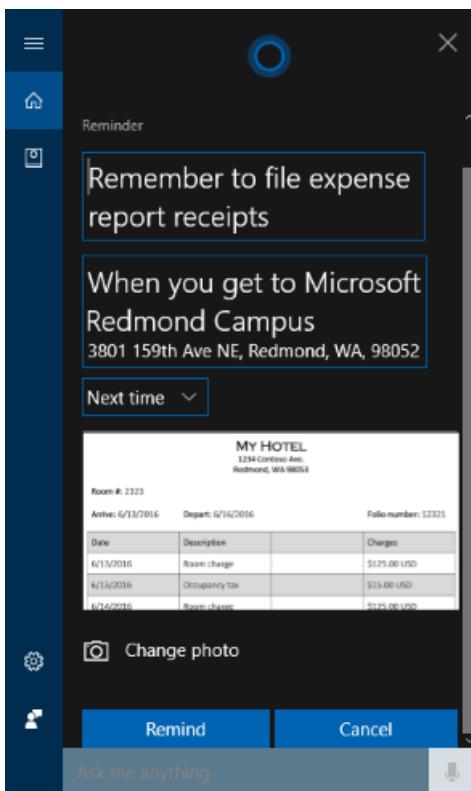
4. Click **Done**.

**NOTE**

If you've never used this location before, you'll be asked to add a name for it so it can be added to the **Favorites list** in Windows Maps.

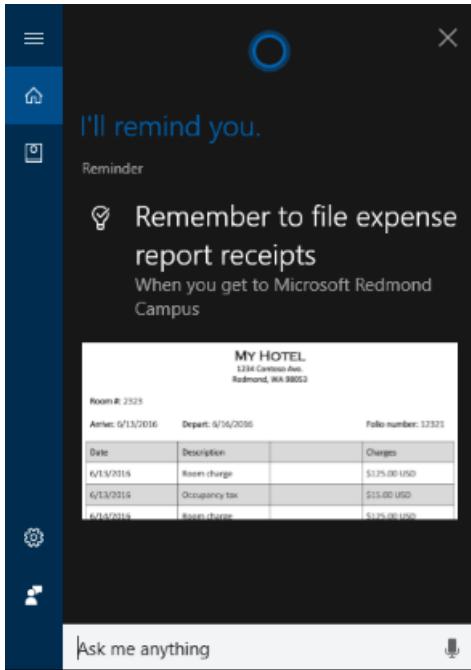
5. Choose to be reminded the **Next time you arrive at the location** or on a specific day of the week from the drop-down box.
6. Take a picture of your receipts and store them locally on your device.
7. Click **Add Photo**, click **Library**, browse to your picture, and then click **OK**.

The photo is stored with the reminder.



8. Review the reminder info, and then click **Remind**.

The reminder is saved and ready to be triggered.

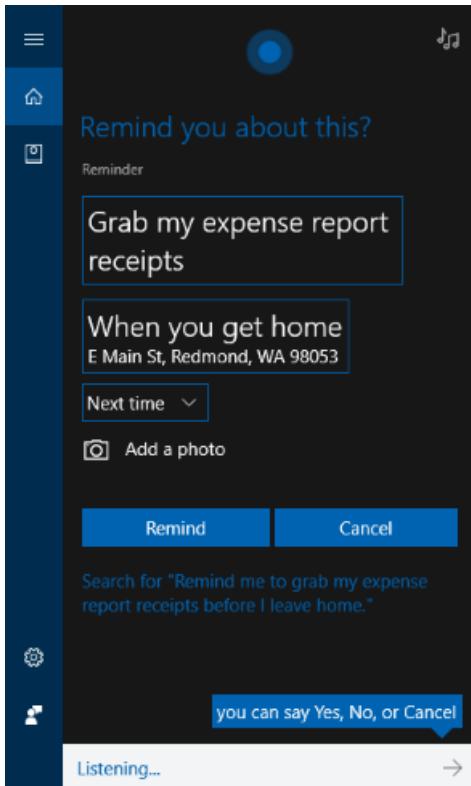


## Create a reminder for a specific location by using voice commands

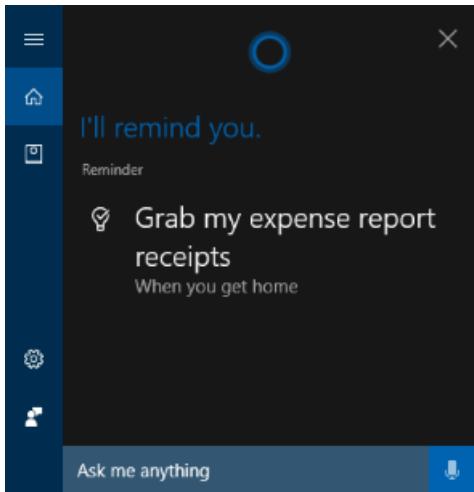
This process helps you to use Cortana at work and voice commands to create a reminder for a specific location.

1. Click on the **Cortana** icon in the taskbar, and then click the **Microphone** icon (to the right of the **Search** box).
2. Say *Remind me to grab my expense report receipts before I leave home.*

Cortana opens a new reminder task and asks if it sounds good.



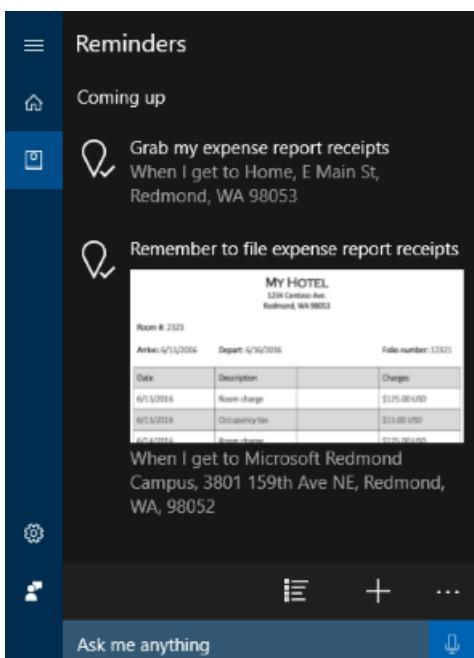
3. Say *Yes* so Cortana can save the reminder.



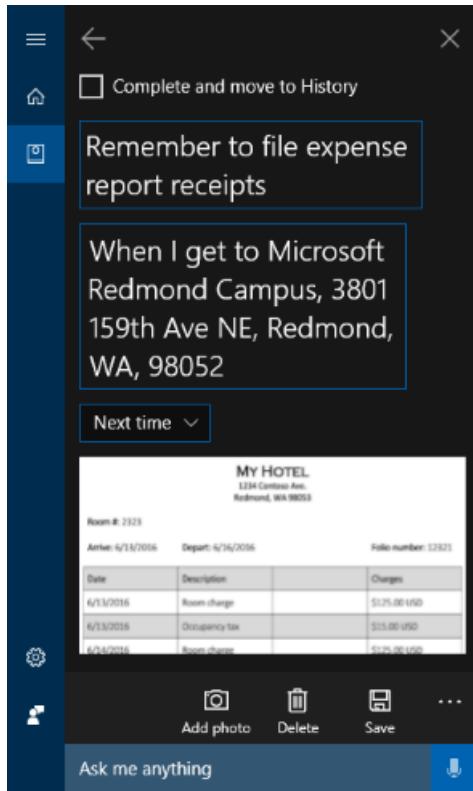
## Edit or archive an existing reminder

This process helps you to edit or archive an existing or completed reminder.

1. Click on the **Cortana** icon in the taskbar, click on the **Notebook** icon, and then click **Reminders**.



2. Click the pending reminder you want to edit.



3. Change any text that you want to change, click **Add photo** if you want to add or replace an image, click **Delete** if you want to delete the entire reminder, click **Save** to save your changes, and click **Complete and move to History** if you want to save a completed reminder in your **Reminder History**.

# Test scenario 4 - Use Cortana at work to find your upcoming meetings

10/5/2017 • 1 min to read • [Edit Online](#)

- Windows 10, version 1703
- Windows 10 Mobile, version 1703

## IMPORTANT

The data created as part of these scenarios will be uploaded to Microsoft's Cloud to help Cortana learn and help your employees. This is the same info that Cortana uses in the consumer offering.

This scenario helps you search for both general upcoming meetings, and specific meetings, both manually and verbally.

## NOTE

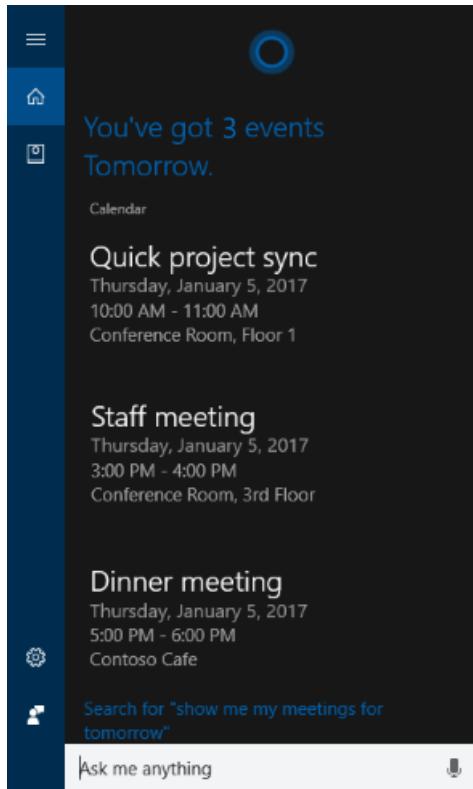
If you've turned on the **Meeting & reminder cards & notifications** option (in the **Meetings & reminders** option of your Notebook), you'll also see your pending reminders on the Cortana **Home** page.

## Find out about upcoming meetings

This process helps you find your upcoming meetings.

1. Check to make sure your work calendar is connected and synchronized with your Azure AD account.
2. Click on the **Cortana** icon in the taskbar, and then click in the **Search** bar.
3. Type *Show me my meetings for tomorrow.*

You'll see all your meetings scheduled for the next day.



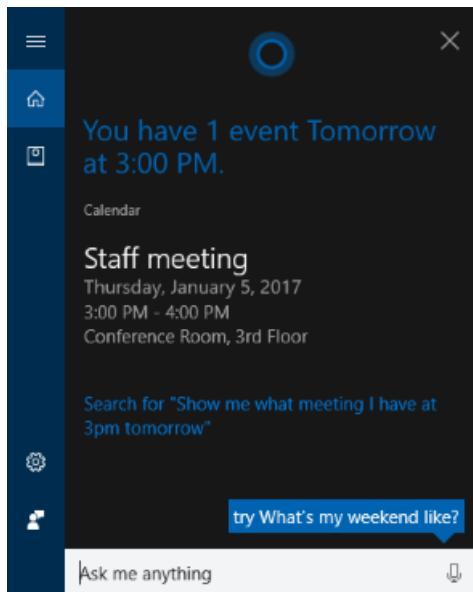
## Find out about upcoming meetings by using voice commands

This process helps you to use Cortana at work and voice commands to find your upcoming meetings.

1. Click on the **Cortana** icon in the taskbar, and then click the **Microphone** icon (to the right of the **Search** box).
2. Say *Show me what meeting I have at 3pm tomorrow.*

### IMPORTANT

Make sure that you have a meeting scheduled for the time you specify here.



# Test scenario 5 - Use Cortana to send email to a co-worker

10/5/2017 • 1 min to read • [Edit Online](#)

- Windows 10, version 1703
- Windows 10 Mobile, version 1703

## IMPORTANT

The data created as part of these scenarios will be uploaded to Microsoft's Cloud to help Cortana learn and help your employees. This is the same info that Cortana uses in the consumer offering.

This scenario helps you to send an email to a co-worker listed in your work address book, both manually and verbally.

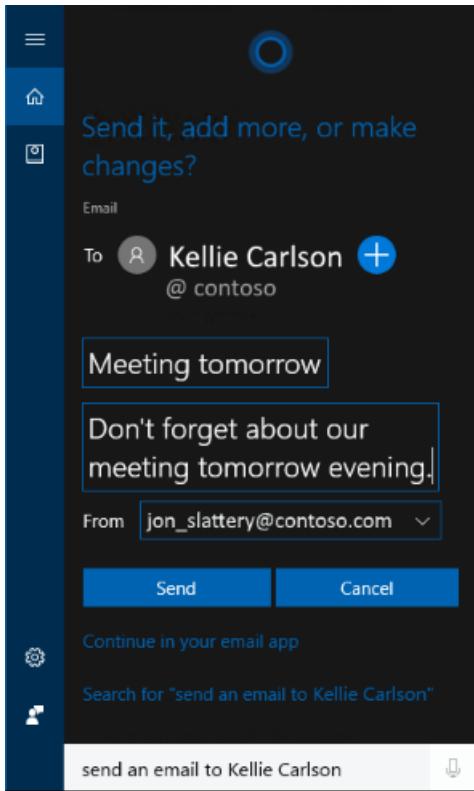
## Send an email to a co-worker

This process helps you to send a quick message to a co-worker from the work address book.

1. Check to make sure your Microsoft Outlook or mail app is connected and synchronized with your Azure AD account.
2. Click on the **Cortana** icon in the taskbar, and then click in the **Search** bar.
3. Type *Send an email to <contact\_name>*.

Where *<contact\_name>* is the name of someone in your work address book.

4. Type your email message subject into the **Quick message** (255 characters or less) box and your message into the **Message** (unlimited characters) box, and then click **Send**.



## Send an email to a co-worker by using voice commands

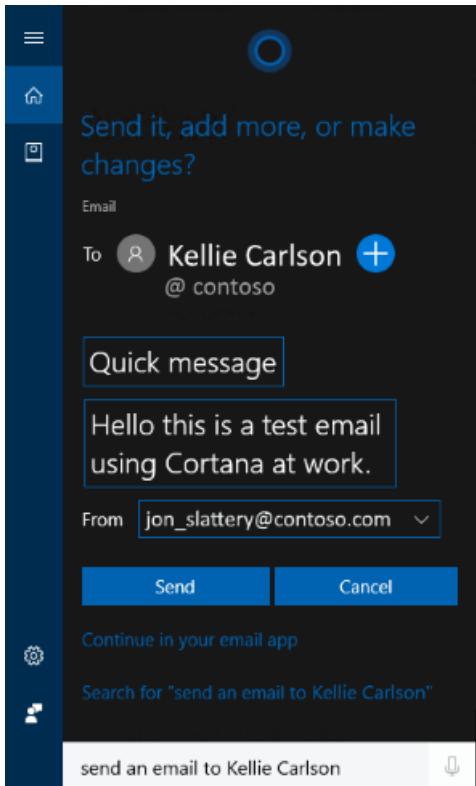
This process helps you to use Cortana at work and voice commands to send a quick message to a co-worker from the work address book.

1. Click on the **Cortana** icon in the taskbar, and then click the **Microphone** icon (to the right of the **Search** box).
2. Say *Send an email to <contact\_name>*.

Where *<contact\_name>* is the name of someone in your work address book.

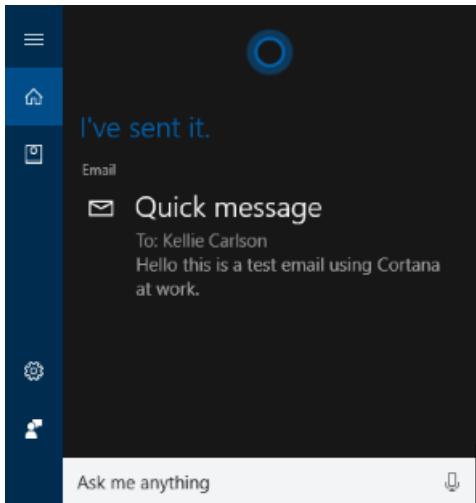
3. Add your email message by saying, *Hello this is a test email using Cortana at work.*

The message is added and you're asked if you want to **Send it**, **Add more**, or **Make changes**.



4. Say *Send it*.

The email is sent.



# Test scenario 6 - Review a reminder suggested by Cortana based on what you've promised in email

10/5/2017 • 1 min to read • [Edit Online](#)

- Windows 10, version 1703
- Windows 10 Mobile, version 1703

## IMPORTANT

The data created as part of these scenarios will be uploaded to Microsoft's Cloud to help Cortana learn and help your employees. This is the same info that Cortana uses in the consumer offering. For more info, see the [Microsoft Privacy Statement](#) and the [Microsoft Services Agreement](#).

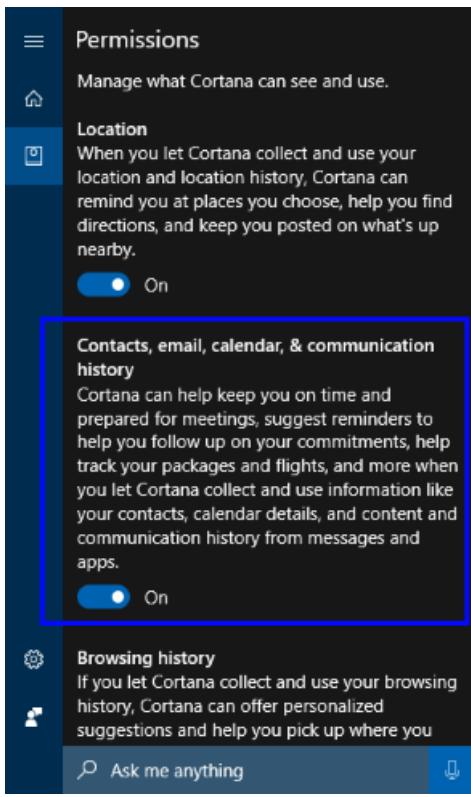
Cortana automatically finds patterns in your email, suggesting reminders based things that you said you would do so you don't forget about them. For example, Cortana recognizes that if you include the text, *I'll get this to you by the end of the week* in an email, you're making a commitment to provide something by a specific date. Cortana can now suggest that you be reminded about this event, letting you decide whether to keep it or to cancel it.

## NOTE

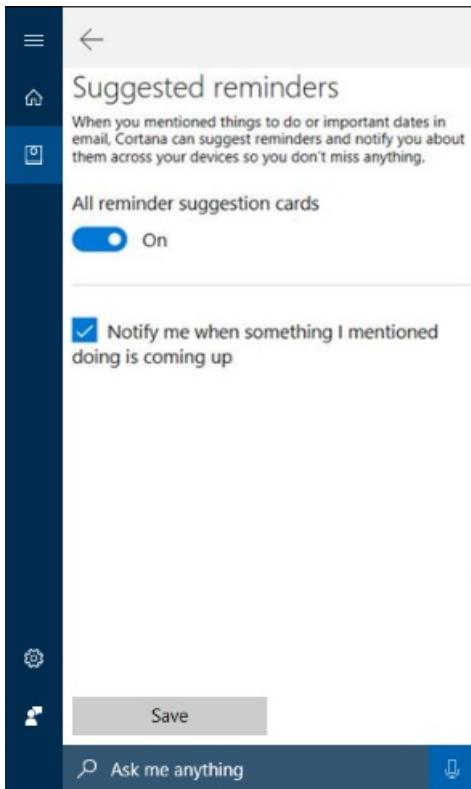
The Suggested reminders feature is currently only available in English (en-us).

## To use Cortana to create Suggested reminders for you

1. Make sure that you've connected Cortana to Office 365. For the steps to connect, see [Set up and test Cortana with Office 365 in your organization](#).
2. Click on the **Cortana** search box in the taskbar, click the **Notebook** icon, and then click **Permissions**.
3. Make sure the **Contacts, email, calendar, and communication history** option is turned on.



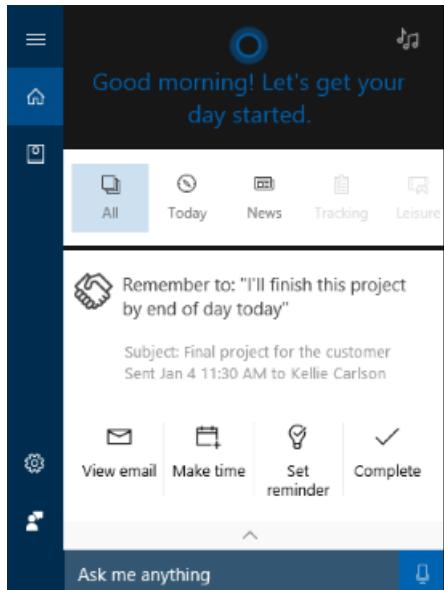
4. Click the **Notebook** icon again, click the **Suggested reminders** option, click to turn on the **All reminder suggestions cards** option, click the **Notify me when something I mentioned doing is coming up** box, and then click **Save**.



5. Create and send an email to yourself (so you can see the Suggested reminder), including the text, *I'll finish this project by end of day today.*
6. After you get the email, click on the Cortana **Home** icon, and scroll to today's events.

If the reminder has a specific date or time associated with it, like end of day, Cortana notifies you at the appropriate time and puts the reminder into the Action Center. Also from the Home screen, you can view the email where you made the promise, set aside time on your calendar, officially set the reminder, or mark the

reminder as completed.



# Test scenario 7 - Use Cortana and Windows Information Protection (WIP) to help protect your organization's data on a device

10/5/2017 • 1 min to read • [Edit Online](#)

- Windows 10, version 1703
- Windows 10 Mobile, version 1703

## IMPORTANT

The data created as part of these scenarios will be uploaded to Microsoft's Cloud to help Cortana learn and help your employees. This is the same info that Cortana uses in the consumer offering.

This optional scenario helps you to protect your organization's data on a device, based on an inspection by Cortana.

## Use Cortana and WIP to protect your organization's data

1. Create and deploy an WIP policy to your organization. For info about how to do this, see [Protect your enterprise data using Windows Information Protection \(WIP\)](#).
2. Create a new email from a non-protected or personal mailbox, including the text *I'll send you that presentation tomorrow.*
3. Wait up to 2 hours to make sure everything has updated, click the **Cortana** icon in the taskbar, and then click in the **Search** bar.  
Cortana automatically pulls your commitment to sending the presentation out of your email, showing it to you.
4. Create a new email from a protected mailbox, including the same text as above, *I'll send you that presentation tomorrow.*
5. Wait until everything has updated again, click the **Cortana** icon in the taskbar, and then click in the **Search** bar.

Because it was in an WIP-protected email, the presentation info isn't pulled out and it isn't shown to you.

# Set up and test Cortana with Office 365 in your organization

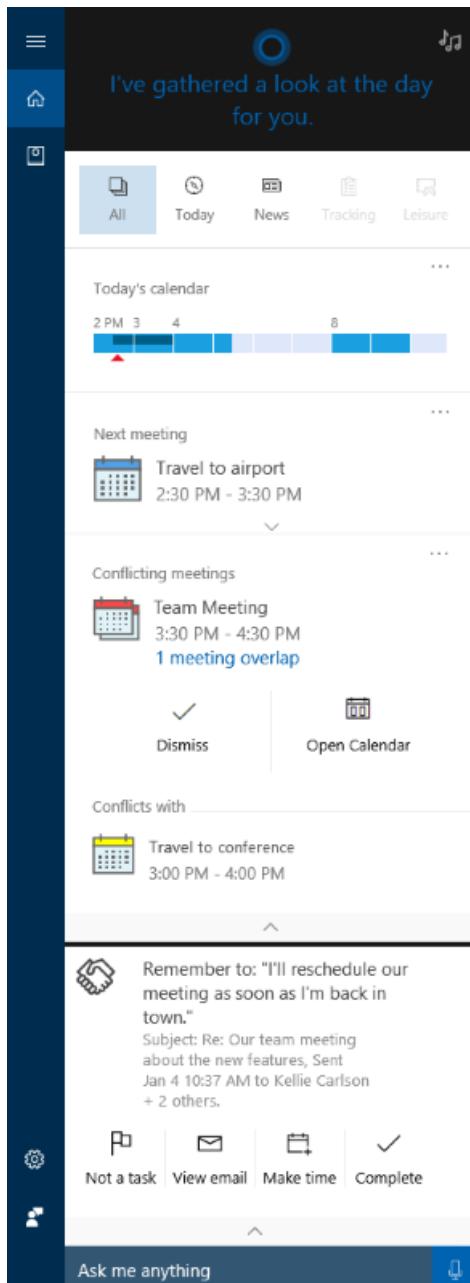
10/5/2017 • 2 min to read • [Edit Online](#)

## Applies to:

- Windows 10, version 1703
- Windows 10 Mobile, version 1703

Cortana in Windows 10 is already great at letting your employees quickly see what the day is going to look like, do meeting prep work like researching people in LinkedIn or getting documents ready, see where and when their meetings are going to be, get a sense of travel times to and from work, and even get updates from a calendar for upcoming trips.

But Cortana works even harder when she connects to Office 365, helping employees to be notified about unusual events, such as meetings over lunch or during a typical commute time, and about early meetings, even setting an alarm so the employee isn't late.



We're continuing to add more and more capabilities to Cortana so she can become even more helpful with your productivity-related tasks, such as emailing, scheduling, and other tasks that are important to help you be successful.

**NOTE**

For a quick review of the frequently asked questions about Cortana and Office 365 integration, see the blog post, [An early look at Cortana integration with Office 365](#).

## Before you begin

There are a few things to be aware of before you start using Cortana with Office 365 in your organization.

- **Software requirements.** O365 integration with Cortana is available in all countries/regions where Cortana is supported for consumers today. This includes the United States, United Kingdom, Canada, France, Italy, Germany, Spain, China, Japan, India, and Australia. As Cortana comes to more countries, it will also become available to organizations.
- **Azure Active Directory (Azure AD) account.** Before your employees can use Cortana in your org, they must be logged in using their Azure AD account through Cortana's notebook. They must also authorize

Cortana to access Office 365 on their behalf.

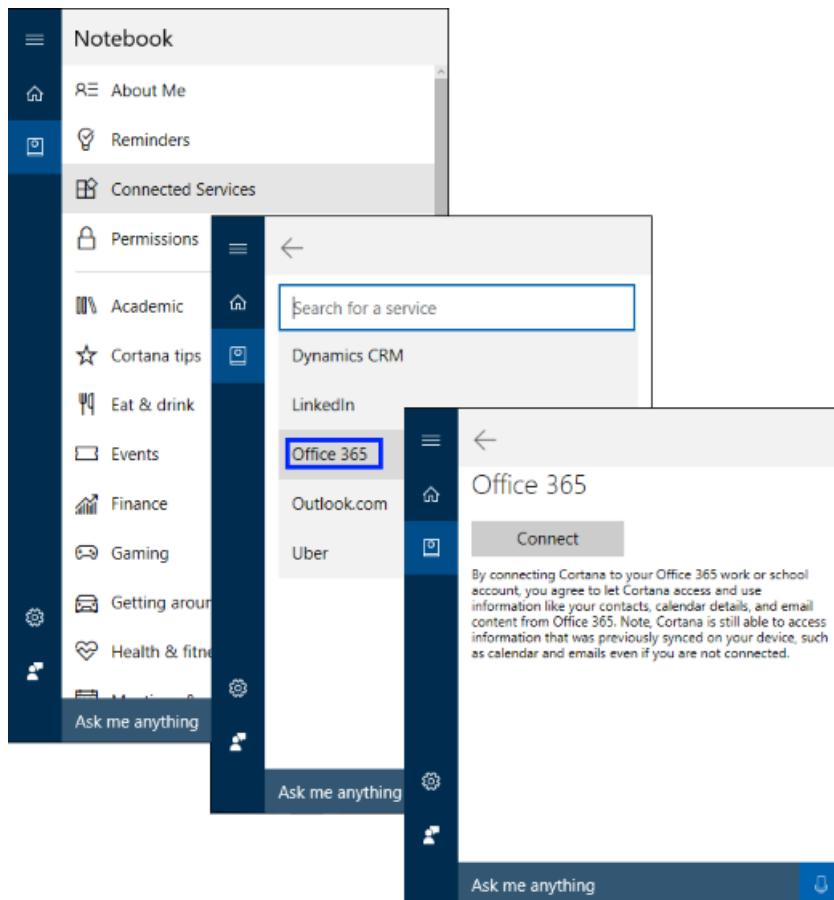
- **Office 365 Trust Center.** Cortana isn't a service covered by the Office 365 Trust Center. [Learn more about how Cortana treats your data.](#)
- **Troubleshooting tips.** If you run into issues, check out these [troubleshooting tips](#).

## Turn on Cortana with Office 365 on employees' devices

You must tell your employees to turn on Cortana before they'll be able to use it with Office 365.

### To turn on local Cortana with Office 365

1. Click on the **Cortana** search box in the taskbar, and then click the **Notebook** icon.
2. Click on **Connected Services**, click **Office 365**, and then click **Connect**.



The employee can also disconnect by clicking **Disconnect** from the **Office 365** screen.

## Turn off Cortana with Office 365

Cortana can only access data in your Office 365 org when it's turned on. If you don't want Cortana to access your corporate data, you can turn it off in the Office 365 admin center.

### To turn off Cortana with Office 365

1. [Sign in to Office 365](#) using your Azure AD account.
2. Go to the [Office 365 admin center](#).
3. Expand **Service Settings**, and select **Cortana**.
4. Click **Cortana** to toggle Cortana off.

All Office 365 functionality related to Cortana is turned off in your organization and your employees are unable to use her at work.

# Set up and test Cortana with Microsoft Dynamics CRM (Preview feature) in your organization

10/5/2017 • 1 min to read • [Edit Online](#)

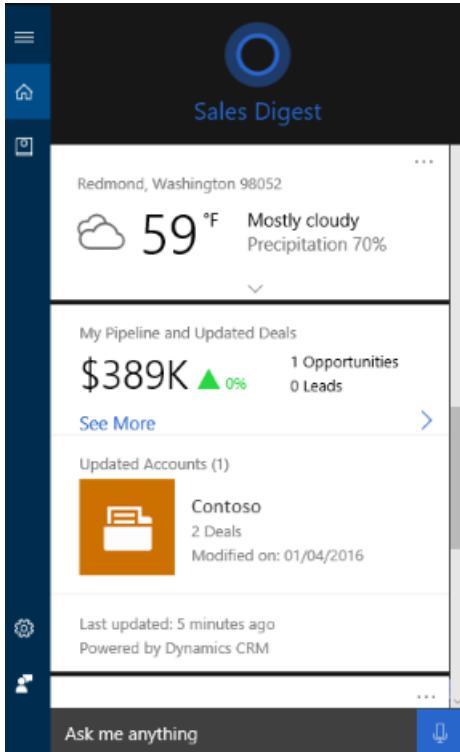
## Applies to:

- Windows 10, version 1703
- Windows 10 Mobile, version 1703

Cortana integration is a Preview feature that's available for your test or dev environment, starting with the CRM Online 2016 Update. If you decide to use this Preview feature, you'll need to turn it on and accept the license terms. After that, your salespeople will get proactive insights from Cortana on important CRM activities, including sales leads, accounts, and opportunities; presenting the most relevant info at any given time. This can even include getting company-specific news that surfaces when the person is meeting with a representative from another company.

### NOTE

For more info about Dynamics CRM integration, how to turn on Cortana, and how to provide feedback, see [Preview feature: Set up Cortana integration](#).



## Turn on Cortana with Dynamics CRM in your organization

You must be a CRM administrator to turn on and use Preview features. For more info about what Preview features are and how to use them, see [What are Preview features and how do I enable them?](#)

### To turn on Cortana with Dynamics CRM

1. Go to **Settings**, and then click **Administration**.

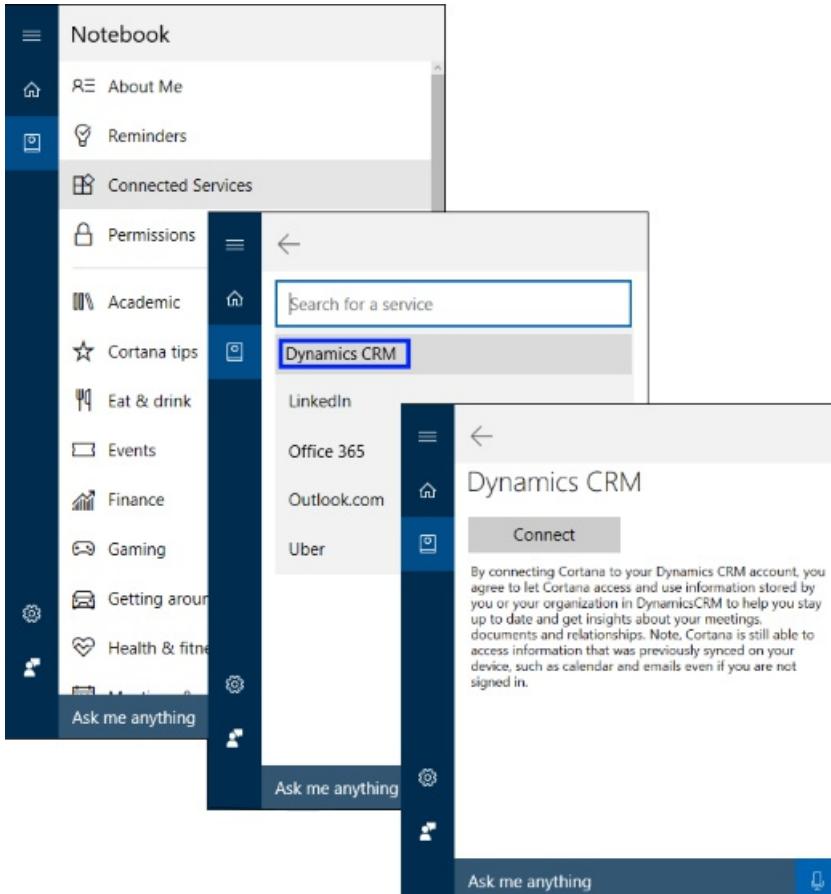
2. Choose **System Settings**, and then click the **Previews** tab.
3. Read the license terms, and if you agree, select the **I've read and agree to the license terms** check box.
4. For each preview feature you want to enable, click **Yes**.

## Turn on Cortana with Dynamics CRM on your employees' devices

You must tell your employees to turn on Cortana, before they'll be able to use it with Dynamics CRM.

### To turn on local Cortana with Dynamics CRM

1. Click on the **Cortana** search box in the taskbar, and then click the **Notebook** icon.
2. Click on **Connected Services**, click **Dynamics CRM**, and then click **Connect**.



The employee can also disconnect by clicking **Disconnect** from the **Dynamics CRM** screen.

## Turn off Cortana with Dynamics CRM

Cortana can only access data in Dynamics CRM when it's turned on. If you don't want Cortana to access your corporate data, you can turn it off.

### To turn off Cortana with Dynamics CRM

1. Go to **Settings**, and then click **Administration**.
2. Choose **System Settings**, and then click the **Previews** tab.
3. Click **No** for **Cortana**.

All Dynamics CRM functionality related to Cortana is turned off in your organization.

# Set up and test Cortana for Power BI in your organization

10/5/2017 • 5 min to read • [Edit Online](#)

## Applies to:

- Windows 10, version 1703
- Windows 10 Mobile, version 1703

Integration between Cortana and Power BI shows how Cortana can work with custom business analytics solutions to enable you to get answers directly from your key business data, including introducing new features that let you create custom Cortana “answers” using the full capabilities of Power BI Desktop.

### NOTE

Cortana for Power BI is currently only available in English. For more info about Cortana and Power BI, see [Use Power BI to create a custom Answer Page for Cortana](#).

## Before you begin

To use this walkthrough, you'll need:

- **Windows 10.** You'll need to be running at least Windows 10, version 1703.
- **Cortana.** You need to have Cortana turned on and be logged into your account.
- **Power BI account with data.** You can use an existing Power BI account, or else you can get a trial account by signing up at <http://powerbi.com>. Just make sure that either way, you enter some data that you can use.
- **Azure Active Directory (Azure AD)/Work or School account.** You can use the account that you created for Office 365, or you can create a new one while you're establishing your Power BI account. If you choose to use Azure AD, you must connect your Azure AD account to your Windows account.

**To connect your account to Windows** a. Open **Windows Settings**, click **Accounts**, click **Access work or school**, and then in the **Connect to work or school** section, click **Connect**.

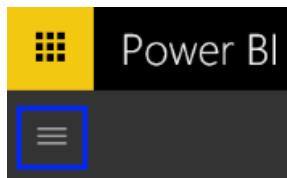
b. Follow the instructions to add your Azure Active Directory (Azure AD) account to Windows.

## Set up your test environment for Cortana for Power BI

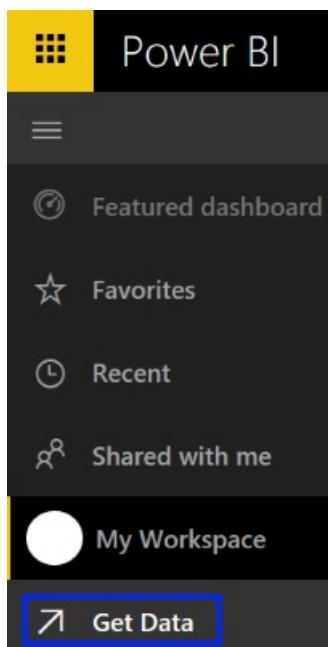
Before you can start this testing scenario, you must first set up your test environment and data, and then you must turn on and set up Cortana to connect and work with Power BI.

### To set up your test environment with Cortana and Power BI

1. Go to <http://powerbi.com> and sign-in with the same O365 credentials you used in the Set up and use Cortana with Office 365 topic.
2. Expand the left rail by clicking the **Show the navigation pane** icon.



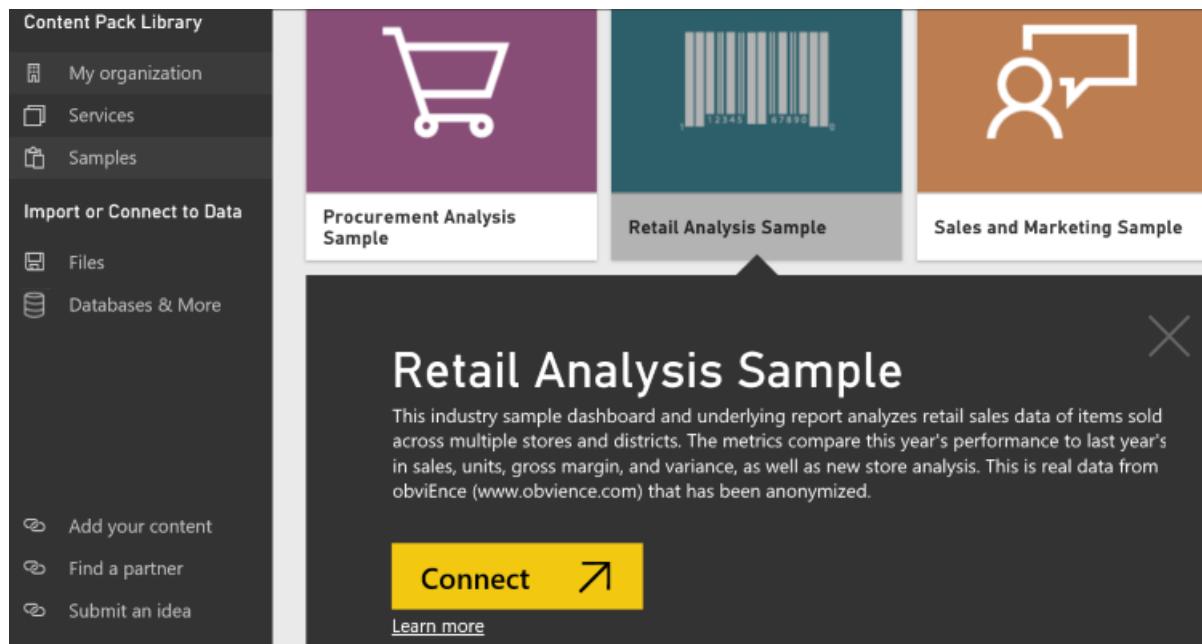
3. Click **Get Data** from the left-hand navigation in Power BI.



4. Click **Samples** from the **Content Pack Library** area of the **Get Data** screen.

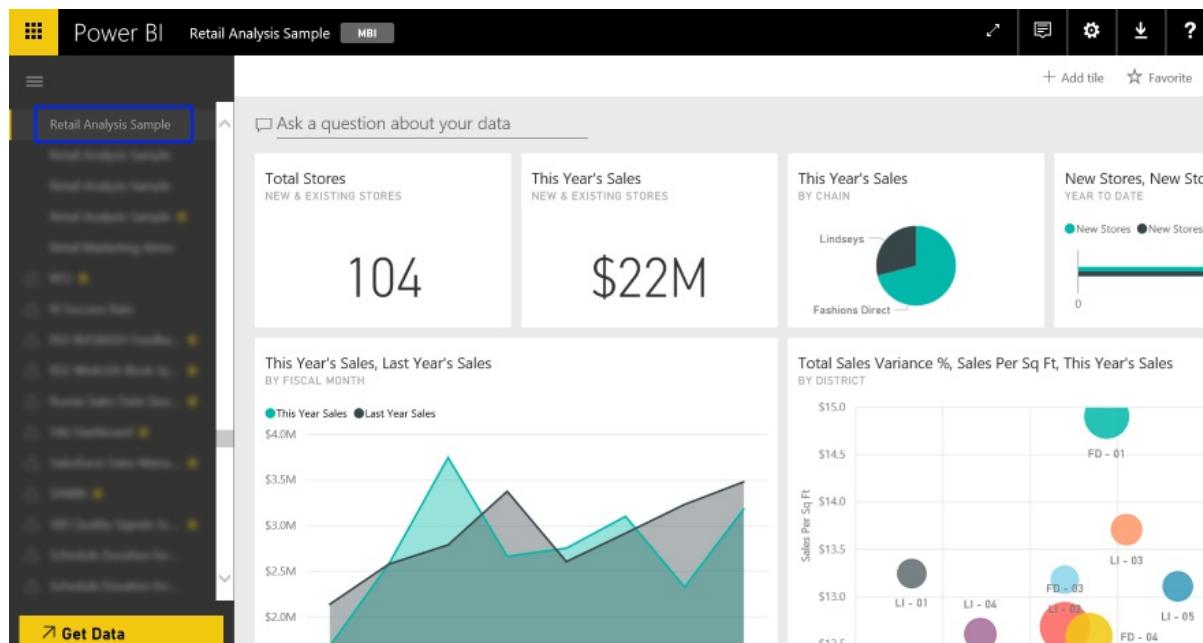
A screenshot of the "Get Data" screen in Power BI. The title "Get Data" is at the top center. Below it is a sub-header "Content Pack Library". There are four main sections: "My organization", "Services", "Files", and "Databases", each with a "Get" button. At the bottom left, there is a "Samples" button. The "Samples" button is highlighted with a blue border. The top right of the screen has a toolbar with icons for message, gear, download, help, and user.

5. Click **Retail Analysis Sample**, and then click **Connect**.

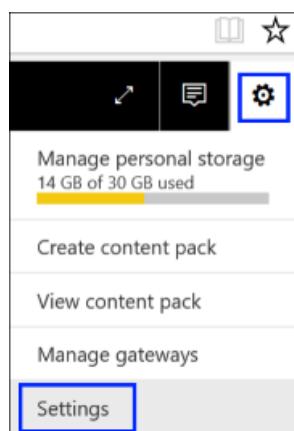


The sample data is imported and you're returned to the **Power BI** screen.

- Click **Dashboards** from the left pane of the **Power BI** screen, and then click **Retail Analysis Sample**.



- In the upper right-hand menu, click the **Settings** icon, and then click **Settings**.



- Click the **Datasets** tab, and then pick the **Retail Analysis Sample** dataset from the list.

- Click **Q&A and Cortana**, check the **Allow Cortana to access this dataset** box, and then click **Apply**.

General Dashboards Datasets Workbooks Alerts Subscriptions

### Settings for Retail Analysis Sample

Opportunity Tracking Sample  
Procurement Analysis Sample  
**Retail Analysis Sample**

① Refresh can't be scheduled because the data set doesn't contain any data model connections, or is a worksheet or linked table. To schedule refresh, the data must be loaded into the data model.

[Refresh history](#)

▲ Q&A and Cortana

Allow Cortana to access this dataset

Cortana will only share this information with Power BI users who have access to it.

**Apply** Discard

**NOTE**

It can take up to 30 minutes for a new dataset to appear for Power BI and Cortana. Logging in and out of Windows 10, or otherwise restarting Cortana, causes the new content to appear immediately.

If you enable a dataset for Cortana, and that dataset is part of a content pack you own, you'll need to re-publish for your colleagues to also use it with Cortana.

## Create a custom Answer Page for Cortana

You must create special reports, known as *Answer Pages*, to display the most commonly asked answers in Cortana. For example, if you want Cortana to quickly show sales data to your employees, you can create a 2016 sales data Answer Page that shows sales data, with various pivots, in Cortana.

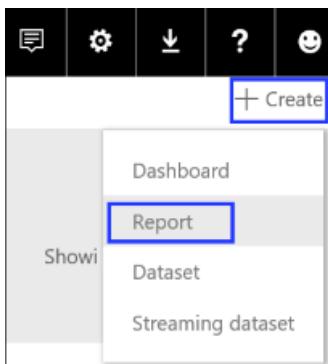
After you've finished creating your Answer Page, you can continue to the included testing scenarios.

**NOTE**

It can take up to 30 minutes for a custom Answer Page to appear for Power BI and Cortana. Logging in and out of Windows 10, or otherwise restarting Cortana, causes the new content to appear immediately.

### To create a custom sales data Answer Page for Cortana

1. In Power BI, click **My Workspace**, click **Create**, and then click **Report**.

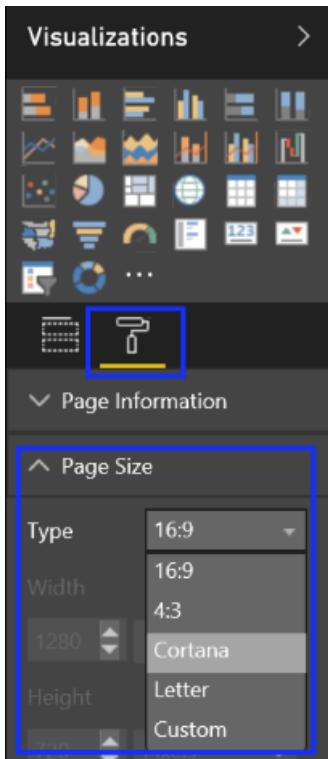


2. In the **Create Report** screen, click the **Retail Analysis Sample**, and then click **Create**.

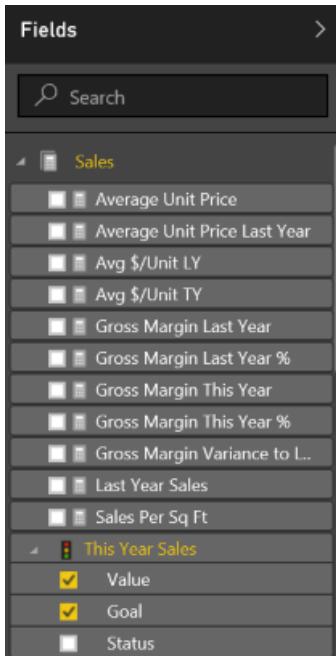
A blank report page appears.

3. In the **Visualizations** pane, click the paint roller icon, expand **Page Size**, and then pick **Cortana** from the

Type drop-down list.



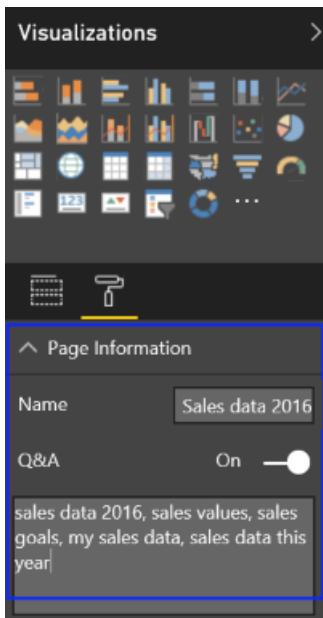
4. In the **Fields** pane, click to expand **Sales**, expand **This year sales**, and then add both **Value** and **Goal**.



The automatically generated graph is added to your blank report. You have the option to change colors, add borders, add additional visualizations, and modify this page so that it answers the question about sales data as precisely, and in as custom a way, as you want. You just need to make sure that it all stays within the page borders.

5. In the **Visualizations** pane, click the paint roller icon again, expand **Page Information**, type *Sales data 2016* into the **Name** box, turn on **Q&A**, and then add alternate report names (separated by commas) into the text box.

The alternate names help Cortana to know what questions to look for and when to show this report. To also improve your results, you should avoid using the names of your report columns.



6. Click **File**, click **Save as**, and save the report as *Sales data 2016*.

Because this is part of the Retail Analysis Sample, it will automatically be included as part of the dataset you included for Cortana. However, you will still need to log in and out of Windows 10, or otherwise restart Cortana, before the new content appears.

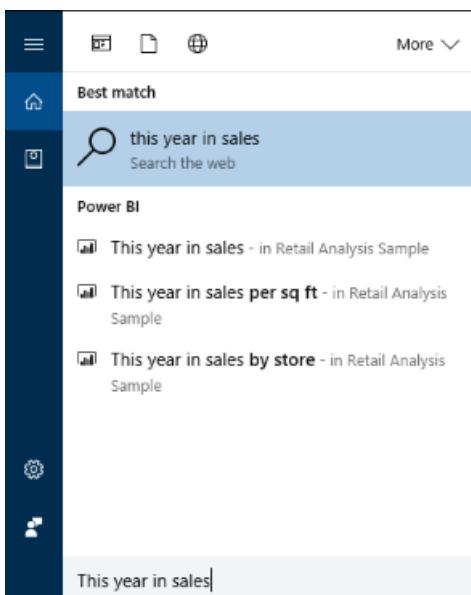
## Test Scenario: Use Cortana to show info from Power BI in your organization

Now that you've set up your device, you can use Cortana to show your info from within Power BI.

### To use Cortana with Power BI

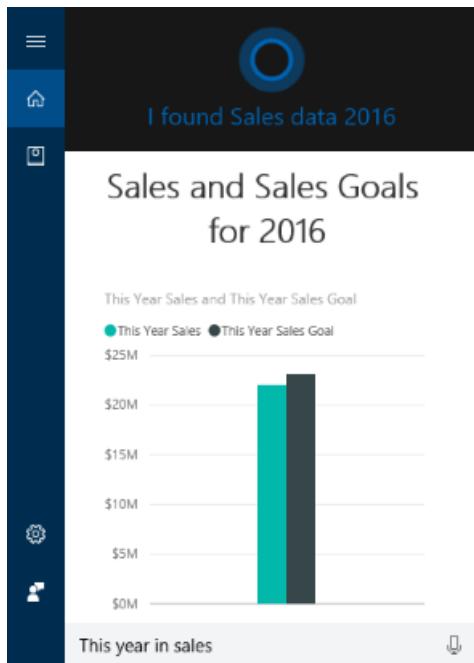
1. Click on the **Cortana** icon in the taskbar, and then click in the **Search** bar.
2. Type *This year in sales*.

Cortana shows you the available results.



3. In the **Power BI** area, click **This year in sales – in Retail Analysis Sample**.

Cortana returns your custom report.



**NOTE**

For more info about how to connect your own data, build your own custom Power BI cards and Answer Pages for Cortana, and how to share the cards with everyone in your organization, see [Use Power BI to create a custom Answer Page for Cortana](#).

# Set up and test custom voice commands in Cortana for your organization

10/5/2017 • 2 min to read • [Edit Online](#)

## Applies to:

- Windows 10, version 1703
- Windows 10 Mobile, version 1703

Working with a developer, you can create voice commands that use Cortana to perform voice-enabled actions in your line-of-business (LOB) Universal Windows Platform (UWP) apps. These voice-enabled actions can reduce the time necessary to access your apps and to complete simple actions.

### NOTE

For more info about how your developer can extend your current apps to work directly with Cortana, see [The Cortana Skills Kit](#).

## High-level process

Cortana uses a Voice Command Definition (VCD) file, aimed at an installed app, to define the actions that are to happen during certain vocal commands. A VCD file can be very simple to very complex, supporting anything from a single sound to a collection of more flexible, natural language sounds, all with the same intent.

To enable voice commands in Cortana

1. **Extend your LOB app.** Add a custom VCD file to your app package. This file defines what capabilities are available to Cortana from the app, letting you tell Cortana what vocal commands should be understood and handled by your app and how the app should start when the command is vocalized.

Cortana can perform actions on apps in the foreground (taking focus from Cortana) or in the background (allowing Cortana to keep focus). We recommend that you decide where an action should happen, based on what your voice command is intended to do. For example, if your voice command requires employee input, it's best for that to happen in the foreground. However, if the app only uses basic commands and doesn't require interaction, it can happen in the background.

- **Start Cortana with focus on your app, using specific voice-enabled statements.** [Activate a foreground app with voice commands through Cortana](#).
  - **Start Cortana removing focus from your app, using specific voice-enabled statements.** [Activate a background app in Cortana using voice commands](#).
2. **Install the VCD file on employees' devices.** You can use System Center Configuration Manager or Microsoft Intune to deploy and install the VCD file on your employees' devices, the same way you deploy and install any other package in your organization.

## Test scenario: Use voice commands in a Microsoft Store app

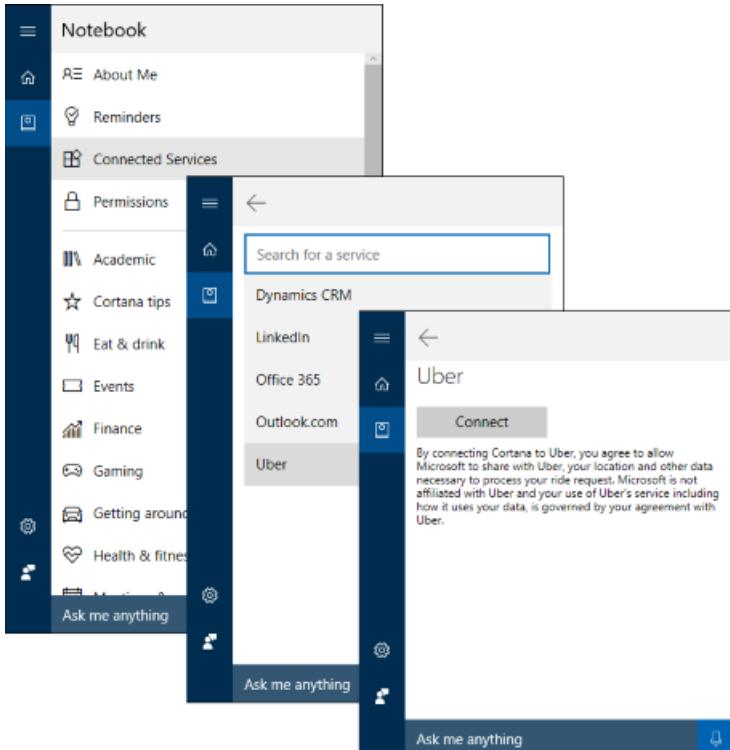
While these aren't line-of-business apps, we've worked to make sure to implement a VCD file, allowing you to test how the functionality works with Cortana in your organization.

## To get a Microsoft Store app

1. Go to the Microsoft Store, scroll down to the **Collections** area, click **Show All**, and then click **Better with Cortana**.
2. Click **Uber**, and then click **Install**.
3. Open Uber, create an account or sign in, and then close the app.

## To set up the app with Cortana

1. Click on the **Cortana** search box in the taskbar, and then click the **Notebook** icon.
2. Click on **Connected Services**, click **Uber**, and then click **Connect**.



## To use the voice-enabled commands with Cortana

1. Click on the **Cortana** icon in the taskbar, and then click the **Microphone** icon (to the right of the **Search** box).
2. Say *Uber get me a taxi*.

Cortana changes, letting you provide your trip details for Uber.

## See also

- [Cortana for developers](#)

# Use Group Policy and mobile device management (MDM) settings to configure Cortana in your organization

10/5/2017 • 2 min to read • [Edit Online](#)

## Applies to:

- Windows 10
- Windows 10 Mobile

### NOTE

For specific info about how to set, manage, and use each of these MDM policies to configure Cortana in your enterprise, see the [Policy CSP](#) topic, located in the configuration service provider reference topics. For specific info about how to set, manage, and use each of these Group Policies to configure Cortana in your enterprise, see the [Group Policy TechCenter](#).

GROUP POLICY	MDM POLICY	DESCRIPTION
Computer Configuration\Administrative Templates\Windows Components\Search\AllowCortanaAboveLock	AboveLock/AllowCortanaAboveLock	<p>Specifies whether an employee can interact with Cortana using voice commands when the system is locked.</p> <p><b>Note</b> This setting only applies to Windows 10 for desktop devices.</p>
Computer Configuration\Administrative Templates\Control Panel\Regional and Language Options\Allow input personalization	Privacy/AllowInputPersonalization	<p>Specifies whether an employee can use voice commands with Cortana in your organization.</p> <p><b>In Windows 10, version 1511</b> Cortana won't work if this setting is turned off (disabled).</p> <p><b>In Windows 10, version 1607 and later</b> Cortana still works if this setting is turned off (disabled).</p>
None	System/AllowLocation	<p>Specifies whether to allow app access to the Location service.</p> <p><b>In Windows 10, version 1511</b> Cortana won't work if this setting is turned off (disabled).</p> <p><b>In Windows 10, version 1607 and later</b> Cortana still works if this setting is turned off (disabled).</p>

GROUP POLICY	MDM POLICY	DESCRIPTION
None	Accounts/AllowMicrosoftAccountConnection	<p>Specifies whether to allow employees to sign in using a Microsoft account (MSA) from Windows apps.</p> <p>Use this setting if you only want to support Azure AD in your organization.</p>
Computer Configuration\Administrative Templates\Windows Components\Search\Allow search and Cortana to use location	Search/AllowSearchToUseLocation	Specifies whether Cortana can use your current location during searches and for location reminders.
Computer Configuration\Administrative Templates\Windows Components\Search\Set the SafeSearch setting for Search	Search/SafeSearchPermissions	<p>Specifies what level of safe search (filtering adult content) is required.</p> <p><b>Note</b> This setting only applies to Windows 10 Mobile.</p>
User Configuration\Administrative Templates\Windows Components\File Explorer\Turn off display of recent search entries in the File Explorer search box	None	Specifies whether the search box can suggest recent queries and prevent entries from being stored in the registry for future reference.
Computer Configuration\Administrative Templates\Windows Components\Search\Don't search the web or display web results	None	<p>Specifies whether search can perform queries on the web and if the web results are displayed in search.</p> <p><b>In Windows 10 Pro edition</b> This setting can't be managed.</p> <p><b>In Windows 10 Enterprise edition</b> Cortana won't work if this setting is turned off (disabled).</p>
Computer Configuration\Administrative Templates\Windows Components\Search\Allow Cortana	Experience/AllowCortana	<p>Specifies whether employees can use Cortana.</p> <p><b>Important</b> Cortana won't work if this setting is turned off (disabled). However, employees can still perform local searches even with Cortana turned off.</p>

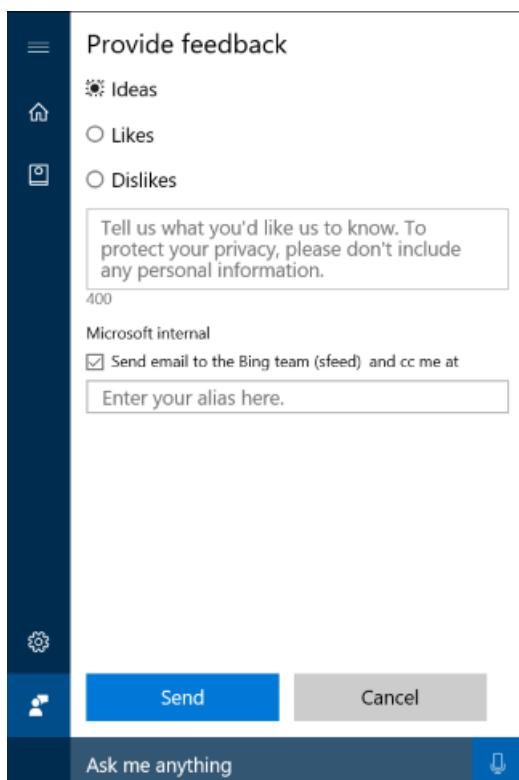
# Send feedback about Cortana at work back to Microsoft

10/5/2017 • 1 min to read • [Edit Online](#)

## Applies to:

- Windows 10, version 1703
- Windows 10 Mobile, version 1703

We ask that you report bugs and issues. To provide feedback, you can click the **Feedback** icon in the Cortana window. When you send this form to Microsoft it also includes troubleshooting info, in case you run into problems.



If you don't want to use the feedback tool in Cortana, you can add feedback through the general Windows Insider Program feedback app. For info about the feedback app, see [How to use Windows Insider Preview – Updates and feedback](#).

# Configure access to Microsoft Store

10/17/2017 • 3 min to read • [Edit Online](#)

## Applies to

- Windows 10
- Windows 10 Mobile

For more info about the features and functionality that are supported in each edition of Windows, see [Compare Windows 10 Editions](#).

IT pros can configure access to Microsoft Store for client computers in their organization. For some organizations, business policies require blocking access to Microsoft Store.

## Options to configure access to Microsoft Store

You can use these tools to configure access to Microsoft Store: AppLocker or Group Policy. For Windows 10, this is only supported on Windows 10 Enterprise edition.

### Block Microsoft Store using AppLocker

Applies to: Windows 10 Enterprise, Windows 10 Education, Windows 10 Mobile

AppLocker provides policy-based access control management for applications. You can block access to Microsoft Store app with AppLocker by creating a rule for packaged apps. You'll give the name of the Microsoft Store app as the packaged app that you want to block from client computers.

For more information on AppLocker, see [What is AppLocker?](#) For more information on creating an AppLocker rule for app packages, see [Create a rule for packaged apps](#).

#### To block Microsoft Store using AppLocker

1. Type secpol in the search bar to find and start AppLocker.
2. In the console tree of the snap-in, click **Application Control Policies**, click **AppLocker**, and then click **Packaged app Rules**.
3. On the **Action** menu, or by right-clicking on **Packaged app Rules**, click **Create New Rule**.
4. On **Before You Begin**, click **Next**.
5. On **Permissions**, select the action (allow or deny) and the user or group that the rule should apply to, and then click **Next**.
6. On **Publisher**, you can select **Use an installed app package as a reference**, and then click **Select**.
7. On **Select applications**, find and click **Store** under **Applications** column, and then click **OK**. Click **Next**.

[Create a rule for packaged apps](#) has more information on reference options and setting the scope on packaged app rules.

8. Optional: On **Exceptions**, specify conditions by which to exclude files from being affected by the rule. This allows you to add exceptions based on the same rule reference and rule scope as you set before. Click **Next**.

# Block Microsoft Store using Group Policy

Applies to: Windows 10 Enterprise, Windows 10 Education

## NOTE

Not supported on Windows 10 Pro, starting with version 1511. For more info, see [Knowledge Base article #3135657](#).

You can also use Group Policy to manage access to Microsoft Store.

## To block Microsoft Store using Group Policy

1. Type gpedit in the search bar to find and start Group Policy Editor.
2. In the console tree of the snap-in, click **Computer Configuration**, click **Administrative Templates**, click **Windows Components**, and then click **Store**.
3. In the Setting pane, click **Turn off Store application**, and then click **Edit policy setting**.
4. On the **Turn off Store application** setting page, click **Enabled**, and then click **OK**.

# Block Microsoft Store using management tool

Applies to: Windows 10 Mobile

If you have mobile devices in your organization that you upgraded from earlier versions of Windows Phone 8 to Windows 10 Mobile, existing policies created using the Windows Phone 8.1 configuration service providers (CSP) with your MDM tool will continue to work on Windows 10 Mobile. If you are starting with Windows 10 Mobile, we recommend using [AppLocker](#) to manage access to Microsoft Store app.

When your MDM tool supports Microsoft Store for Business, the MDM can use these CSPs to block Microsoft Store app:

- [Policy](#)
- [EnterpriseAssignedAccess](#) (Windows 10 Mobile, only)

For more information, see [Configure an MDM provider](#).

# Show private store only using Group Policy

Applies to Windows 10 Enterprise, version 1607, Windows 10 Education

If you're using Microsoft Store for Business and you want employees to only see apps you're managing in your private store, you can use Group Policy to show only the private store. Microsoft Store app will still be available, but employees can't view or purchase apps. Employees can view and install apps that the admin has added to your organization's private store.

## To show private store only in Microsoft Store app

1. Type **gpedit** in the search bar, and then select **Edit group policy (Control panel)** to find and start Group Policy Editor.
2. In the console tree of the snap-in, go to **User Configuration** or **Computer Configuration** > **Administrative Templates** > **Windows Components**, and then click **Store**.
3. Right-click **Only display the private store within the Microsoft Store app** in the right pane, and click **Edit**.

This opens the **Only display the private store within the Microsoft Store app** policy settings.

4. On the **Only display the private store within the Microsoft Store app** setting page, click **Enabled**, and then click **OK**.

## Related topics

[Distribute apps using your private store](#)

[Manage access to private store](#)

Not finding content you need? Windows 10 users, tell us what you want on [Feedback Hub](#).

# Provisioning packages for Windows 10

7/28/2017 • 6 min to read • [Edit Online](#)

## Applies to

- Windows 10
- Windows 10 Mobile

Windows provisioning makes it easy for IT administrators to configure end-user devices without imaging. Using Windows provisioning, an IT administrator can easily specify desired configuration and settings required to enroll the devices into management and then apply that configuration to target devices in a matter of minutes. It is best suited for small- to medium-sized businesses with deployments that range from tens to a few hundred computers.

A provisioning package (.ppkg) is a container for a collection of configuration settings. With Windows 10, you can create provisioning packages that let you quickly and efficiently configure a device without having to install a new image.

Provisioning packages are simple enough that with a short set of written instructions, a student or non-technical employee can use them to configure their device. This can result in a significant reduction in the time required to configure multiple devices in your organization.

The [Windows Assessment and Deployment Kit \(ADK\) for Windows 10](#) includes the Windows Configuration Designer, a tool for configuring provisioning packages. Windows Configuration Designer is also available as an [app in the Microsoft Store](#).

## New in Windows 10, version 1703

- The tool for creating provisioning packages is renamed Windows Configuration Designer, replacing the Windows Imaging and Configuration Designer (ICD) tool. The components for creating images have been removed from Windows Configuration Designer, which now provides access to runtime settings only.
- Windows Configuration Designer can still be installed from the Windows ADK. You can also install it from the Microsoft Store.
- Windows Configuration Designer adds more wizards to make it easier to create provisioning packages for specific scenarios. See [What you can configure](#) for wizard descriptions.
- The wizard **Provision desktop devices** (previously called **Simple provisioning**) now enables joining Azure Active Directory (Azure AD) domains and also allows you to remove non-Microsoft software from Windows desktop devices during provisioning.
- When provisioning packages are applied to a device, a status screen indicates successful or failed provisioning.
- Windows 10 includes PowerShell cmdlets that simplify scripted provisioning. Using these cmdlets, you can add provisioning packages, remove provisioning packages and generate log files to investigate provisioning errors.
- The **Provision school devices** wizard is removed from Windows Configuration Designer. Instead, use the [Setup School PCs app](#) from the Microsoft Store.

## Benefits of provisioning packages

Provisioning packages let you:

- Quickly configure a new device without going through the process of installing a new image.

- Save time by configuring multiple devices using one provisioning package.
- Quickly configure employee-owned devices in an organization without a mobile device management (MDM) infrastructure.
- Set up a device without the device having network connectivity.

Provisioning packages can be:

- Installed using removable media such as an SD card or USB flash drive.
- Attached to an email.
- Downloaded from a network share.
- Deployed in NFC tags or barcodes.

## What you can configure

### Configuration Designer wizards

The following table describes settings that you can configure using the wizards in Windows Configuration Designer to create provisioning packages.

Step	Description	Desktop wizard	Mobile wizard	Kiosk wizard
Set up device	Assign device name, enter product key to upgrade Windows, configure shared used, remove pre-installed software	✓	✓ (Only device name and upgrade key)	✓
Set up network	Connect to a Wi-Fi network	✓	✓	✓
Account management	Enroll device in Active Directory, enroll device in Azure Active Directory, or create a local administrator account	✓	✗	✓
Bulk Enrollment in Azure AD	Enroll device in Azure Active Directory Before you use a Windows Configuration Designer wizard to configure bulk Azure AD enrollment, <a href="#">set up Azure AD join in your organization</a> .	✗	✓	✗
Add applications	Install applications using the provisioning package.	✓	✗	✓

Add certificates	Include a certificate file in the provisioning package.	✓	✗	✓
Configure kiosk account and app	Create local account to run the kiosk mode app, specify the app to run in kiosk mode	✗	✗	✓
Configure kiosk common settings	Set tablet mode, configure welcome and shutdown screens, turn off timeout settings	✗	✗	✓

- [Instructions for the desktop wizard](#)
- [Instructions for the mobile wizard](#)
- [Instructions for the kiosk wizard](#)

**NOTE**

After you start a project using a Windows Configuration Designer wizard, you can switch to the advanced editor to configure additional settings in the provisioning package.

## Configuration Designer advanced editor

The following table provides some examples of settings that you can configure using the Windows Configuration Designer advanced editor to create provisioning packages.

CUSTOMIZATION OPTIONS	EXAMPLES
Bulk Active Directory join and device name	Join devices to Active Directory domain and assign device names using hardware-specific serial numbers or random characters
Applications	Windows apps, line-of-business applications
Bulk enrollment into MDM	Automatic enrollment into a third-party MDM service*
Certificates	Root certification authority (CA), client certificates
Connectivity profiles	Wi-Fi, proxy settings, Email
Enterprise policies	Security restrictions (password, device lock, camera, and so on), encryption, update settings
Data assets	Documents, music, videos, pictures
Start menu customization	Start menu layout, application pinning
Other	Home and lock screen wallpaper, computer name, domain join, DNS settings, and so on

\* Using a provisioning package for auto-enrollment to System Center Configuration Manager or Configuration

Manager/Intune hybrid is not supported. Use the Configuration Manager console to enroll devices.

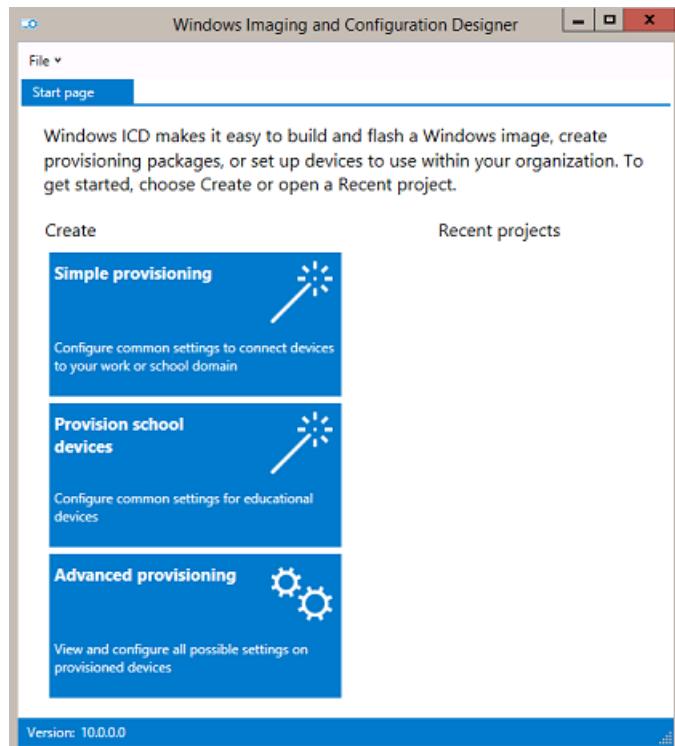
For details about the settings you can customize in provisioning packages, see [Windows Provisioning settings reference](#).

## Changes to provisioning in Windows 10, version 1607

### NOTE

This section is retained for customers using Windows 10, version 1607, on the Current Branch for Business. Some of this information is not applicable in Windows 10, version 1703.

Windows ICD for Windows 10, version 1607, simplified common provisioning scenarios.



Windows ICD in Windows 10, version 1607, supported the following scenarios for IT administrators:

- **Simple provisioning** – Enables IT administrators to define a desired configuration in Windows ICD and then apply that configuration on target devices. The simple provisioning wizard makes the entire process quick and easy by guiding an IT administrator through common configuration settings in a step-by-step manner.  

[Learn how to use simple provisioning to configure Windows 10 computers.](#)
- **Advanced provisioning (deployment of classic (Win32) and Universal Windows Platform (UWP) apps, and certificates)** – Allows an IT administrator to use Windows ICD to open provisioning packages in the advanced settings editor and include apps for deployment on end-user devices.
- **Mobile device enrollment into management** - Enables IT administrators to purchase off-the-shelf retail Windows 10 Mobile devices and enroll them into mobile device management (MDM) before handing them to end-users in the organization. IT administrators can use Windows ICD to specify the management end-point and apply the configuration on target devices by connecting them to a Windows PC (tethered deployment) or through an SD card. Supported management end-points include:
  - System Center Configuration Manager and Microsoft Intune hybrid (certificate-based enrollment)

- AirWatch (password-string based enrollment)
- Mobile Iron (password-string based enrollment)
- Other MDMS (cert-based enrollment)

**NOTE**

Windows ICD in Windows 10, version 1607, also provided a wizard to create provisioning packages for school PCs. To learn more, see [Set up students' PCs to join domain](#).

## Learn more

- Watch the video: [Provisioning Windows 10 Devices with New Tools](#)
- Watch the video: [Windows 10 for Mobile Devices: Provisioning Is Not Imaging](#)

## Related topics

- [How provisioning works in Windows 10](#)
- [Install Windows Configuration Designer](#)
- [Create a provisioning package](#)
- [Apply a provisioning package](#)
- [Settings changed when you uninstall a provisioning package](#)
- [Provision PCs with common settings for initial deployment \(simple provisioning\)](#)
- [Use a script to install a desktop app in provisioning packages](#)
- [PowerShell cmdlets for provisioning Windows 10 \(reference\)](#)
- [Windows Configuration Designer command-line interface \(reference\)](#)
- [Create a provisioning package with multivariant settings](#)
- [Use Windows Configuration Designer to configure Windows 10 Mobile devices](#)

# How provisioning works in Windows 10

7/28/2017 • 11 min to read • [Edit Online](#)

## Applies to

- Windows 10
- Windows 10 Mobile

Provisioning packages in Windows 10 provide IT administrators with a simplified way to apply configuration settings to Windows 10 devices. Windows Configuration Designer is a tool that makes it easy to create a provisioning package. Windows Configuration Designer can be installed from the [Windows Assessment and Deployment Kit \(ADK\)](#) or through the Microsoft Store.

## Provisioning packages

A provisioning package contains specific configurations/settings and assets that can be provided through a removable media or simply downloaded to the device.

To enable adding multiple sets of settings or configurations, the configuration data used by the provisioning engine is built out of multiple configuration sources that consist of separate provisioning packages. Each provisioning package contains the provisioning data from a different source.

A provisioning package (.ppkg) is a container for a collection of configuration settings. The package has the following format:

- Package metadata – The metadata contains basic information about the package such as package name, description, version, ranking, and so on.
- XML descriptors – Each descriptor defines a customization asset or configuration setting included in the package.
- Asset payloads – The payloads of a customization asset or a configuration setting associated with an app or data asset.

You can use provisioning packages for runtime device provisioning by accessing the package on a removable media attached to the device, through near field communication (NFC), or by downloading from a remote source location.

## Precedence for provisioning packages

When multiple provisioning packages are available for device provisioning, the combination of package owner type and package rank level defined in the package manifest is used to resolve setting conflicts. The pre-defined package owner types are listed below in the order of lowest to highest owner type precedence:

1. Microsoft
2. Silicon Vendor
3. OEM
4. System Integrator
5. Mobile Operator
6. IT Admin

The valid value range of package rank level is 0 to 99.

When setting conflicts are encountered, the final values provisioned on the device are determined by the owner type precedence and the rank level of the packages containing the settings. For example, the value of a setting in a package with owner **System Integrator** and rank level **3** takes precedence over the same setting in a package with owner **OEM** and rank level **4**. This is because the System Integrator owner type has the higher precedence over the OEM owner type. For packages with the same owner type, the package rank level determines the package from which the setting values get provisioned on the device.

## Windows provisioning XML

Windows provisioning XML is the framework that allows Microsoft and OEM components to declare end-user configurable settings and the on-device infrastructure for applying the settings with minimal work by the component owner.

Settings for each component can be declared within that component's package manifest file. These declarations are turned into settings schema that are used by Windows Configuration Designer to expose the potential settings to users to create customizations in the image or in provisioning packages. Windows Configuration Designer translates the user configuration, which is declared through Windows provisioning answer file(s), into the on-device provisioning format.

When the provisioning engine selects a configuration, the Windows provisioning XML is contained within the selected provisioning data and is passed through the configuration manager and then to the [Windows provisioning CSP](#). The Windows provisioning CSP then takes and applies the provisioning to the proper location for the actual component to use.

## Provisioning engine

The provisioning engine is the core component for managing provisioning and configuration at runtime in a device running Windows 10.

The provisioning engine provides the following functionality:

- Provisioning configuration at any time when the device is running including first boot and setup or OOBE. It is also extensible to other points during the run-time of the device.
- Reading and combining settings from multiple sources of configuration that may be added to an image by Microsoft, the OEM, or system integrator, or added by IT/education administrators or users to the device at run-time. Configuration sources may be built into the image or from provisioning packages added to the device.
- Responding to triggers or events and initiating a provisioning stage.
- Authenticating the provisioning packages.
- Selecting a set of configuration based on the stage and a set of keys—such as the SIM, MCC/MNC, IMSI range, and so on—that map to a specific configuration then passing this configuration to the configuration management infrastructure to be applied.
- Working with OOBE and the control panel UI to allow user selection of configuration when a specific match cannot be determined.

## Configuration manager

The configuration manager provides the unified way of managing Windows 10 devices. Configuration is mainly done through the Open Mobile Alliance (OMA) Device Management (DM) and Client Provisioning (CP) protocols. The configuration manager handles and parses these protocol requests from different channels and passes them down to [Configuration Service Providers \(CSPs\)](#) to perform the specific management requests and settings.

The provisioning engine relies on configuration manager for all of the actual processing and application of a

chosen configuration. The provisioning engine determines the stage of provisioning and, based on a set of keys, determines the set of configuration to send to the configuration manager. The configuration manager in turn parses and calls into the CSPs for the setting to be applied.

Underneath the configuration manager are the CSPs. Each section of configuration translates to a particular CSP to handle interpreting into an action on the device. Each CSP translates the instructions in the configuration and calls into the appropriate APIs and components to perform the requested provisioning actions.

## Policy and resource manager

The policy, resource, and context manager components manage the enrollment and unenrollment of devices into enterprise environments. The enrollment process into an enterprise is essentially the provisioning of configuration and device management policies that the enterprise wants to enforce on the device. This is usually done through the explicit signing up of the device to an enterprise's device management server over a network connection. This provides the user with the ability to access the enterprise's resources through the device and the enterprise with a means to manage and control access and manage and control the device itself.

The key differences between enterprise enrollment and the configuration performed by the provisioning engine are:

- Enrollment enforces a limited and controlled set of policies on the device that the user may not have full control over. The provisioning engine exposes a larger set of settings that configure more aspects of the device and are generally user adjustable.
- The policy manager manages policy settings from multiple entities and performs a selection of the setting based on priority of the entities. The provisioning engine applies the settings and does not offer a means of prioritizing settings from different sources. The more specific provisioning is the last one applied and the one that is used.
- Individual policy settings applied from different enrollment entities are stored so they can be removed later during unenrollment. This enables the user to remove enterprise policy and return the device to a state without the enterprise restrictions and any sensitive data. The provisioning engine does not maintain individual provisioning settings or a means to roll back all applied settings.

In Windows 10, the application of policy and enrollment through provisioning is required to support cases where an enterprise or educational institution does not have a DM server for full device management. The provisioning engine supports provisioning enrollment and policy through its configuration and integrates with the existing policy and resource manager components directly or through the configuration manager.

## Triggers and stages

Triggers are events during the lifetime of the system that start a provisioning stage. Some examples of triggers are: boot, OOBE, SIM change, user added, administrator added, user login, device update, and various manual triggers (such as deployment over USB or launched from an email attachment or USB flash drive).

When a trigger occurs, provisioning is initiated for a particular provisioning stage. The stages are grouped into sets based on the scope of the settings:

- **Static:** First stage run for provisioning to apply configuration settings to the system to set up OOBE or apply device-wide settings that cannot be done when the image is being created.
- **System:** Run during OOBE and configure system-wide settings.
- **UICC:** UICC stages run for each new UICC in a device to handle configuration and branding based on the identity of the UICC or SIM card. This enables the runtime configuration scenarios where an OEM can maintain one image that can be configured for multiple operators.
- **Update:** Runs after an update to apply potential updated settings changes.
- **User:** runs during a user account first run to configure per-user settings.

## Device provisioning during OOBE

The provisioning engine always applies provisioning packages persisted in the `C:\Recovery\Customizations` folder on the OS partition. When the provisioning engine applies provisioning packages in the `%ProgramData%\Microsoft\Provisioning` folder, certain runtime setting applications, such as the setting to install and configure Windows apps, may be extended past the OOBE pass and continually be processed in the background when the device gets to the desktop. Settings for configuring policies and certain crucial system configurations are always completed before the first point at which they must take effect.

Device users can apply a provisioning package from a remote source when the device first boots to OOBE. The device provisioning during OOBE is only triggered after the language, locale, time zone, and other settings on the first OOBE UI page are configured. When device provisioning is triggered, the provisioning UI is displayed in the OOBE page. The provisioning UI allows users to select a provisioning package acquired from a remote source, such as through NFC or a removable media.

The following table shows how device provisioning can be initiated when a user first boots to OOBE.

PACKAGE DELIVERY	INITIATION METHOD	SUPPORTED DEVICE
Removable media - USB drive or SD card (Packages must be placed at media root)	5 fast taps on the Windows key to launch the provisioning UI	All Windows devices
From an administrator device through machine-to-machine NFC or NFC tag (The administrator device must run an app that can transfer the package over NFC)	5 fast taps on the Windows key to launch the provisioning UI	Windows 10 Mobile devices and IoT Core devices

The provisioning engine always copies the acquired provisioning packages to the `%ProgramData%\Microsoft\Provisioning` folder before processing them during OOBE. The provisioning engine always applies provisioning packages embedded in the installed Windows image during Windows Setup OOBE pass regardless of whether the package is signed and trusted. When the provisioning engine applies an encrypted provisioning package on an end-user device during OOBE, users must first provide a valid password to decrypt the package. The provisioning engine also checks whether a provisioning package is signed and trusted; if it's not, the user must provide consent before the package is applied to the device.

When the provisioning engine applies provisioning packages during OOBE, it applies only the runtime settings from the package to the device. Runtime settings can be system-wide configuration settings, including security policy, Windows app install/uninstall, network configuration, bootstrapping MDM enrollment, provisioning of file assets, account and domain configuration, Windows edition upgrade, and more. The provisioning engine also checks for the configuration settings on the device, such as region/locale or SIM card, and applies the multivariant settings with matching condition(s).

## Device provisioning at runtime

At device runtime, stand-alone provisioning packages can be applied by user initiation. The following table shows when provisioning at device runtime can be initiated.

PACKAGE DELIVERY	INITIATION METHOD	SUPPORTED DEVICE
Removable media - USB drive or SD card (Packages must be placed at media root)	<b>Settings &gt; Accounts &gt; Access work or school &gt; Add or remove a provisioning package</b>	All Windows devices
Downloaded from a network connection and copied to a local folder	Double-click the package file	Windows 10 for desktop editions devices
From an administrator device connected to the target device through USB tethering	Drag and drop the package file onto the target device	Windows 10 Mobile devices and IoT Core devices

When applying provisioning packages from a removable media attached to the device, the Settings UI allows viewing contents of a package before selecting the package for provisioning. To minimize the risk of the device being spammed by applying provisioning packages from unknown sources, a provisioning package can be signed and encrypted. Partners can also set policies to limit the application of provisioning packages at device runtime. Applying provisioning packages at device runtime requires administrator privilege. If the package is not signed or trusted, a user must provide consent before the package is applied to the device. If the package is encrypted, a valid password is needed to decrypt the package before it can be applied to the device.

When applying multiple provisioning packages to a device, the provisioning engine resolves settings with conflicting configuration values from different packages by evaluating the package ranking using the combination of package owner type and package rank level defined in the package metadata. A configuration setting applied from a provisioning package with the highest package ranking will be the final value applied to the device.

After a stand-alone provisioning package is applied to the device, the package is persisted in the `%ProgramData%\Microsoft\Provisioning` folder on the device. Provisioning packages can be removed by an administrator by using the **Add or remove a provisioning package** available under **Settings > Accounts > Access work or school**.

## Learn more

- Watch the video: [Provisioning Windows 10 Devices with New Tools](#)
- Watch the video: [Windows 10 for Mobile Devices: Provisioning Is Not Imaging](#)

## Related topics

- [Provisioning packages for Windows 10](#)
- [Install Windows Configuration Designer](#)
- [Create a provisioning package](#)
- [Apply a provisioning package](#)
- [Settings changed when you uninstall a provisioning package](#)
- [Provision PCs with common settings for initial deployment \(simple provisioning\)](#)
- [Use a script to install a desktop app in provisioning packages](#)
- [PowerShell cmdlets for provisioning Windows 10 \(reference\)](#)
- [Windows Configuration Designer command-line interface \(reference\)](#)
- [Create a provisioning package with multivariant settings](#)

# Introduction to configuration service providers (CSPs) for IT pros

7/28/2017 • 9 min to read • [Edit Online](#)

## Applies to

- Windows 10
- Windows 10 Mobile

Configuration service providers (CSPs) expose device configuration settings in Windows 10. This topic is written for people who have no experience with CSPs.

The CSPs are documented on the [Hardware Dev Center](#) because CSPs are used by mobile device management (MDM) service providers. This topic explains how IT pros and system administrators can take advantage of many settings available through CSPs to configure devices running Windows 10 and Windows 10 Mobile in their organizations.

### NOTE

The explanation of CSPs and CSP documentation also apply to Windows Mobile 5, Windows Mobile 6, Windows Phone 7, and Windows Phone 8, but links to current CSPs are for Windows 10 and Windows 10 Mobile.

See what's new for CSPs in Windows 10, version 1607.

## What is a CSP?

A CSP is an interface in the client operating system between configuration settings specified in a provisioning document and configuration settings on the device. Their function is similar to that of Group Policy client-side extensions in that they provide an interface to read, set, modify, or delete configuration settings for a given feature. Typically, these settings map to registry keys, files or permissions. Some of these settings are configurable and some are read-only.

Starting in Windows Mobile 5.0, CSPs were used to manage Windows mobile devices. In the Windows 10 platform, the management approach for both desktop and mobile devices converges, taking advantage of the same CSPs to configure and manage all devices running Windows 10.

Each CSP provides access to specific settings. For example, the [Wi-Fi CSP](#) contains the settings to create a Wi-Fi profile.

CSPs are behind many of the management tasks and policies for Windows 10 in Microsoft Intune and non-Microsoft MDM service providers. For example, in Intune, the policy to allow search suggestions in the Microsoft Edge address bar uses **Browser/AllowSearchSuggestionsinAddressBar** in the [Policy CSP](#).

**Intune**

**CSP**

**Browser/AllowSearchSuggestionsinAddressBar**

Specifies whether search suggestions are allowed in the address bar.

The following list shows the supported values:

- 0 – Not allowed.
- 1 (default) – Allowed.

Most restricted value is 0.

CSPs receive configuration policies in the XML-based SyncML format pushed to it from an MDM-compliant management server such as Microsoft Intune. Traditional enterprise management systems, such as System Center Configuration Manager, can also target CSPs by using a client-side WMI-to-CSP bridge.

### Synchronization Markup Language (SyncML)

The Open Mobile Alliance Device Management (OMA-DM) protocol uses the XML-based Synchronization Markup Language (SyncML) for data exchange between compliant servers and clients. SyncML offers an open standard to use as an alternative to vendor-specific management solutions (such as WMI). The value for enterprises adopting industry standard management protocols is that it allows the management of a broader set of vendor devices using a single platform (such as Microsoft Intune). Device policies, including VPN connection profiles, are delivered to client devices formatted as in SyncML. The target CSP reads this information and applies the necessary configurations.

### The WMI-to-CSP Bridge

The WMI-to-CSP Bridge is a component allowing configuration of Windows 10 CSPs via scripts and traditional enterprise management software such as Configuration Manager using Windows Management Instrumentation (WMI). The bridge is responsible for reading WMI commands and through a component called the common device configurator pass them to a CSP for application on the device.

[Learn how to use the WMI Bridge Provider with PowerShell.](#)

## Why should you learn about CSPs?

Generally, enterprises rely on Group Policy or MDM to configure and manage devices. For devices running Windows, MDM services use CSPs to configure your devices.

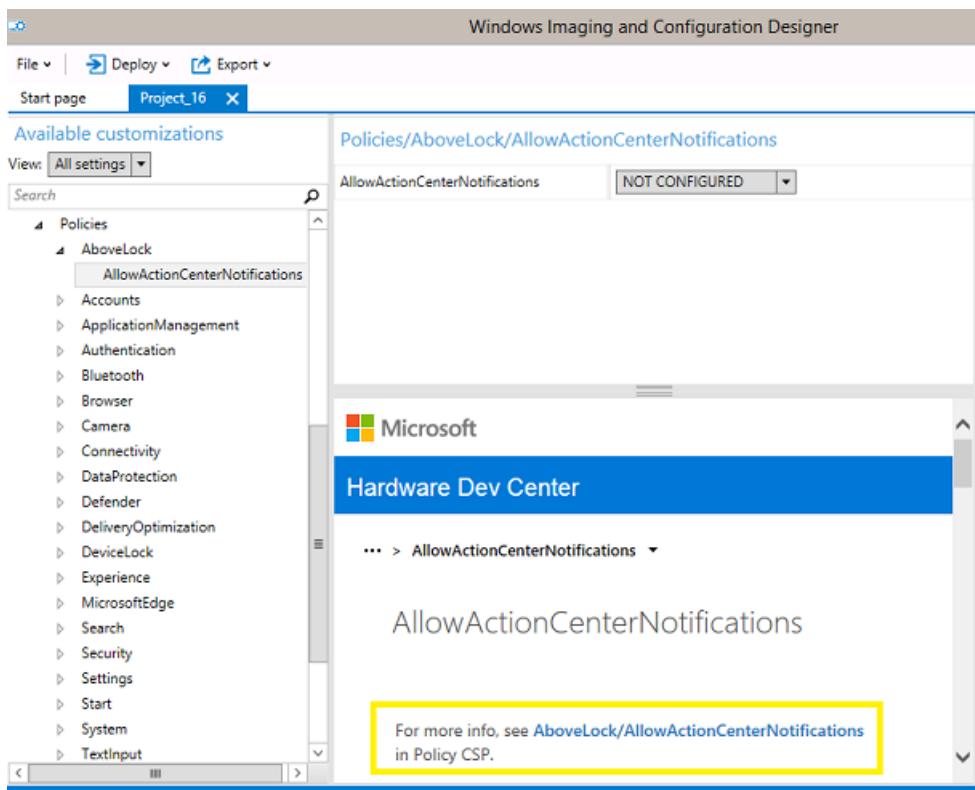
In addition, you may have unmanaged devices, or a large number of devices that you want to configure before enrolling them in management, or you want to apply custom settings that aren't available through your MDM service. The [CSP documentation](#) can help you understand the settings that can be configured or queried.

In addition, some of the topics in the [Windows 10 and Windows 10 Mobile](#) library on Technet include links to applicable CSP reference topics, such as [Cortana integration in your business or enterprise](#) which links to the [Policy CSP](#). In the CSP topics, you can learn about all of the available configuration settings.

### CSPs in Windows Configuration Designer

You can use Windows Configuration Designer to create [provisioning packages](#) to apply settings to devices during the out-of-box-experience (OOBE) and after devices are set up. You can use provisioning packages to configure a device's connectivity and enroll the device in MDM. Many of the runtime settings in Windows Configuration Designer are based on CSPs.

Many settings in Windows Configuration Designer will display documentation for that setting in the center pane, and will include a reference to the CSP if the setting uses one, as shown in the following image.



[Provisioning packages in Windows 10](#) explains how to use the Windows Configuration Designer tool to create a runtime provisioning package.

### CSPs in MDM

Most, if not all, CSPs are surfaced through your MDM service. If you see a CSP that provides a capability that you want to make use of and cannot find that capability in your MDM service, contact your MDM provider for assistance. It might simply be named differently than you expected. You can see the CSPs supported by MDM in the [Configuration service provider reference](#).

When a CSP is available but is not explicitly included in your MDM solution, you may be able to make use of the CSP by using OMA-URI settings. In Intune, for example, you can use [custom policy settings](#) to deploy settings.

Intune documents [a partial list of settings](#) that you can enter in the **OMA-URI Settings** section of a custom policy, if your MDM service provides that extension. You'll notice that the list doesn't explain the meanings of the allowed and default values, so use the [CSP reference documentation](#) to locate that information.

### CSPs in Lockdown XML

Lockdown XML can be used to configure devices running Windows 10 Mobile. You can manually author a [Lockdown XML file](#) to make use of the configuration settings available through the [EnterpriseAssignedAccess configuration service provider \(CSP\)](#). In Windows 10, version 1703, you can also use the new [Lockdown Designer app](#) to configure your Lockdown XML.

## How do you use the CSP documentation?

All CSPs in Windows 10 are documented in the [Configuration service provider reference](#).

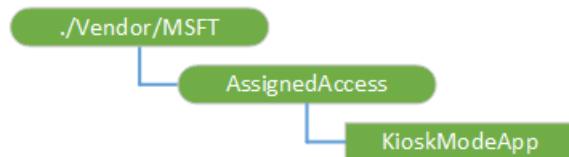
The [main CSP topic](#) tells you which CSPs are supported on each edition of Windows 10, and links to the documentation for each individual CSP.

Configuration service provider	Supported in Windows 10 Pro	Supported in Windows 10 Enterprise	Supported in Windows 10 Education	Supported in Windows 10 Home	Supported in Windows 10 Mobile	Supported in Windows 10 IoT Core (IoT Core)
ActiveSync CSP	✓	✓	✓	✓	✓	✗

The documentation for each CSP follows the same structure. After an introduction that explains the purpose of the CSP, a diagram shows the parts of the CSP in tree format.

The full path to a specific configuration setting is represented by its Open Mobile Alliance - Uniform Resource Identifier (OMA-URI). The URI is relative to the devices' root node (MSFT, for example). Features supported by a particular CSP can be set by addressing the complete OMA-URI path.

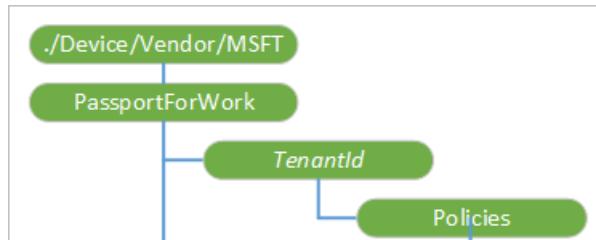
The following example shows the diagram for the [AssignedAccess CSP](#). The diagram maps to the XML for that CSP. Notice the different shapes in the diagram: rounded elements are nodes and rectangular elements are settings or policies for which a value must be supplied.



The element in the tree diagram after the root node tells you the name of the CSP. Knowing this structure, you would recognize in XML the parts of the URI path for that CSP and, if you saw it in XML, you would know which CSP reference to look up. For example, in the following OMS-URI path for the kiosk mode app settings, you can see it uses the [AssignedAccess CSP](#).

```
./Vendor/MSFT/AssignedAccess/KioskModeApp
```

When an element in the diagram uses italic font, it indicates a placeholder for specific information, such as the tenant ID in the following example.



After the diagram, the documentation describes each element. For each policy or setting, the valid values are listed.

For example, in the [AssignedAccess CSP](#), the setting is **KioskModeApp**. The documentation tells you that the value for **KioskModeApp** is a JSON string that contains the user account name and Application User Model ID (AUMID) of the Kiosk mode app.

The documentation for most CSPs will also include an XML example.

## CSP examples

CSPs provide access to a number of settings useful to enterprises. This section introduces two CSPs that an enterprise might find particularly useful.

- [EnterpriseAssignedAccess CSP](#)

The EnterpriseAssignedAccess configuration service provider allows IT administrators to configure settings on a Windows 10 Mobile device. An enterprise can make use of this CSP to create single-use or limited-use mobile devices, such as a handheld device that only runs a price-checking app.

In addition to lockscreen wallpaper, theme, time zone, and language, the EnterpriseAssignedAccess CSP includes AssignedAccessXml which can be used to lock down the device through the following settings:

- Enabling or disabling the Action Center.

- Configuring the number of tile columns in the Start layout.
- Restricting the apps that will be available on the device.
- Restricting the settings that the user can access.
- Restricting the hardware buttons that will be operable.
- Restricting access to the context menu.
- Enabling or disabling tile manipulation.
- Creating role-specific configurations.

- [Policy CSP](#)

The Policy configuration service provider enables the enterprise to configure policies on Windows 10 and Windows 10 Mobile. Some of these policy settings can also be applied using Group Policy, and the CSP documentation lists the equivalent Group Policy settings.

Some of the settings available in the Policy CSP include the following:

- **Accounts**, such as whether a non-Microsoft account can be added to the device
- **Application management**, such as whether only Microsoft Store apps are allowed
- **Bluetooth**, such as the services allowed to use it
- **Browser**, such as restricting InPrivate browsing
- **Connectivity**, such as whether the device can be connected to a computer by USB
- **Defender** (for desktop only), such as day and time to scan
- **Device lock**, such as the type of PIN or password required to unlock the device
- **Experience**, such as allowing Cortana
- **Security**, such as whether provisioning packages are allowed
- **Settings**, such as allowing the user to change VPN settings
- **Start**, such as applying a standard Start layout
- **System**, such as allowing the user to reset the device
- **Text input**, such as allowing the device to send anonymized user text input data samples to Microsoft
- **Update**, such as specifying whether the device could use Microsoft Update, Windows Server Update Services (WSUS), or Microsoft Store
- **WiFi**, such as whether to enable Internet sharing

Here is a list of CSPs supported on Windows 10 Enterprise, Windows 10 Mobile Enterprise, or both:

- [ActiveSync CSP](#)
- [Application CSP](#)
- [AppLocker CSP](#)
- [AssignedAccess CSP](#)
- [Bootstrap CSP](#)
- [BrowserFavorite CSP](#)
- [CellularSettings CSP](#)
- [CertificateStore CSP](#)
- [ClientCertificateInstall CSP](#)
- [CM\\_CellularEntries CSP](#)
- [CM\\_ProxyEntries CSP](#)
- [CMPolicy CSP](#)
- [Defender CSP](#)
- [DevDetail CSP](#)
- [DeviceInstanceService CSP](#)
- [DeviceLock CSP](#)

- DeviceStatus CSP
- DevInfo CSP
- DiagnosticLog CSP
- DMAcc CSP
- DMClient CSP
- Email2 CSP
- EnterpriseAPN CSP
- EnterpriseAppManagement CSP
- EnterpriseAssignedAccess CSP
- EnterpriseDesktopAppManagement CSP
- EnterpriseExt CSP
- EnterpriseExtFileSystem CSP
- EnterpriseModernAppManagement CSP
- FileSystem CSP
- HealthAttestation CSP
- HotSpot CSP
- Maps CSP
- NAP CSP
- NAPDEF CSP
- NodeCache CSP
- PassportForWork CSP
- Policy CSP
- PolicyManager CSP
- Provisioning CSP
- Proxy CSP
- PXLOGICAL CSP
- Registry CSP
- RemoteFind CSP
- RemoteWipe CSP
- Reporting CSP
- RootCATrustedCertificates CSP
- SecurityPolicy CSP
- Storage CSP
- SUPL CSP
- UnifiedWriteFilter CSP
- Update CSP
- VPN CSP
- VPnv2 CSP
- Wi-Fi CSP
- WindowsLicensing CSP
- WindowsSecurityAuditing CSP

# Install Windows Configuration Designer

10/17/2017 • 4 min to read • [Edit Online](#)

## Applies to

- Windows 10
- Windows 10 Mobile

Use the Windows Configuration Designer tool to create provisioning packages to easily configure devices running Windows 10. Windows Configuration Designer is primarily designed for use by IT departments for business and educational institutions who need to provision bring-your-own-device (BYOD) and business-supplied devices.

## Supported platforms

Windows Configuration Designer can create provisioning packages for Windows 10 desktop and mobile editions, including Windows 10 IoT Core, as well as Microsoft Surface Hub and Microsoft HoloLens. You can run Windows Configuration Designer on the following operating systems:

- Windows 10 - x86 and amd64
- Windows 8.1 Update - x86 and amd64
- Windows 8.1 - x86 and amd64
- Windows 8 - x86 and amd64
- Windows 7 - x86 and amd64
- Windows Server 2016
- Windows Server 2012 R2 Update
- Windows Server 2012 R2
- Windows Server 2012
- Windows Server 2008 R2

### WARNING

You must run Windows Configuration Designer on Windows 10 to configure Azure Active Directory enrollment using any of the wizards.

## Install Windows Configuration Designer

On devices running Windows 10, you can install [the Windows Configuration Designer app from the Microsoft Store](#). To run Windows Configuration Designer on other operating systems or in languages other than English, install it from the [Windows Assessment and Deployment Kit \(ADK\) for Windows 10](#).

### NOTE

If you install Windows Configuration Designer from both the ADK and Microsoft Store, the Store app will not open.

The Windows Configuration Designer App from Microsoft Store currently supports only English. For a localized version of the Windows Configuration Designer, install it from the Windows ADK.

1. Go to [Download the Windows ADK](#) and select **Get Windows ADK** for the version of Windows 10 that

you want to create provisioning packages for (version 1511, 1607, or 1703).

**NOTE**

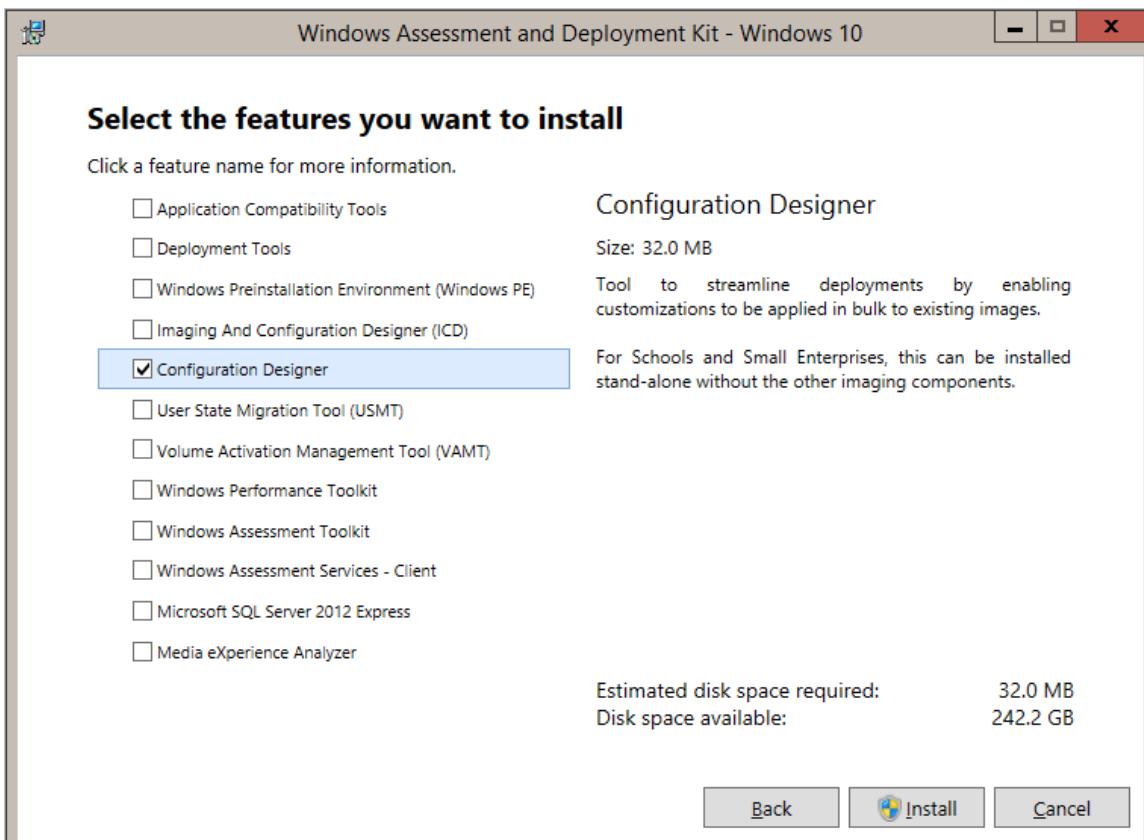
The rest of this procedure uses Windows ADK for Windows 10, version 1703 as an example.

2. Save **adksetup.exe** and then run it.
3. On the **Specify Location** page, select an installation path and then click **Next**.

**NOTE**

The estimated disk space listed on this page applies to the full Windows ADK. If you only install Windows Configuration Designer, the space requirement is approximately 32 MB.

4. Make a selection on the **Windows Kits Privacy** page, and then click **Next**.
5. Accept the **License Agreement**, and then click **Next**.
6. On the **Select the features you want to install** page, clear all selections except **Configuration Designer**, and then click **Install**.



## Current Windows Configuration Designer limitations

- You can only run one instance of Windows Configuration Designer on your computer at a time.
- Be aware that when adding apps and drivers, all files stored in the same folder will be imported and may cause errors during the build process.
- The Windows Configuration Designer UI does not support multivariant configurations. Instead, you must use the Windows Configuration Designer command-line interface to configure multivariant settings. For more information, see [Create a provisioning package with multivariant settings](#).

- While you can open multiple projects at the same time within Windows Configuration Designer, you can only build one project at a time.
- In order to enable the simplified authoring jscripts to work on a server SKU running Windows Configuration Designer, you need to explicitly enable **Allow websites to prompt for information using scripted windows**. Do this by opening Internet Explorer and then navigating to **Settings > Internet Options > Security -> Custom level > Allow websites to prompt for information using scripted windows**, and then choose **Enable**.
- If you copy a Windows Configuration Designer project from one PC to another PC, make sure that all the associated files for the deployment assets, such as apps and drivers, are copied along with the project to the same path as it was on the original PC.

For example, when you add a driver to a provisioned package, you must copy the .INF file to a local directory on the PC that is running Windows Configuration Designer. If you don't do this, and attempt to use a copied version of this project on a different PC, Windows Configuration Designer might attempt to resolve the path to the files that point to the original PC.

- **Recommended:** Before starting, copy all source files to the PC running Windows Configuration Designer, rather than using external sources like network shares or removable drives. This reduces the risk of interrupting the build process from a temporary network issue or from disconnecting the USB device.

**Next step:** [How to create a provisioning package](#)

## Learn more

- Watch the video: [Provisioning Windows 10 Devices with New Tools](#)
- Watch the video: [Windows 10 for Mobile Devices: Provisioning Is Not Imaging](#)

## Related topics

- [Provisioning packages for Windows 10](#)
- [How provisioning works in Windows 10](#)
- [Create a provisioning package](#)
- [Apply a provisioning package](#)
- [Settings changed when you uninstall a provisioning package](#)
- [Provision PCs with common settings for initial deployment \(simple provisioning\)](#)
- [Use a script to install a desktop app in provisioning packages](#)
- [PowerShell cmdlets for provisioning Windows 10 \(reference\)](#)
- [Windows Configuration Designer command-line interface \(reference\)](#)
- [Create a provisioning package with multivariant settings](#)

Not finding content you need? Windows 10 users, tell us what you want on [Feedback Hub](#).

# Create a provisioning package for Windows 10

7/28/2017 • 7 min to read • [Edit Online](#)

## Applies to

- Windows 10
- Windows 10 Mobile

You use Windows Configuration Designer to create a provisioning package (.ppkg) that contains customization settings. You can apply the provisioning package to a device running Windows 10 or Windows 10 Mobile.

[Learn how to install Windows Configuration Designer.](#)

### TIP

We recommend creating a local admin account when developing and testing your provisioning package. We also recommend using a "least privileged" domain user account to join devices to the Active Directory domain.

## Start a new project

### 1. Open Windows Configuration Designer:

- From either the Start screen or Start menu search, type 'Windows Configuration Designer' and click on the Windows Configuration Designer shortcut,

or

- If you installed Windows Configuration Designer from the ADK, navigate to

C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Imaging and Configuration Designer\x86

(on an x64 computer) or

C:\Program Files\Windows Kits\10\Assessment and Deployment Kit\Imaging and Configuration Designer\x86\ICD.exe

(on an x86 computer), and then double-click **ICD.exe**.

### 2. Select your desired option on the **Start** page, which offers multiple options for creating a provisioning package, as shown in the following image:

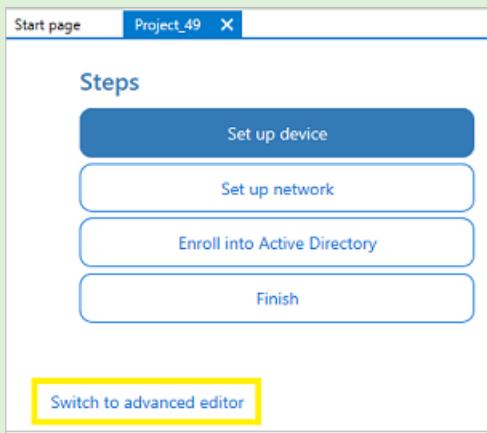
## Create

<b>Provision desktop devices</b>  Configure common settings for Windows desktop devices	<b>Provision Windows mobile devices</b>  Configure common settings for Windows mobile devices
<b>Provision HoloLens devices</b>  Configure common settings for HoloLens devices	<b>Provision Surface Hub devices</b>  Configure common settings for Surface Hub devices
<b>Provision kiosk devices</b>  Configure common settings for a device that will run a single app in kiosk mode	<b>Advanced provisioning</b>  View and configure all possible settings on provisioned devices

- The wizard options provide a simple interface for configuring common settings for desktop, mobile, and kiosk devices. Wizards are also available for creating provisioning packages for Microsoft Surface Hub and Microsoft HoloLens devices. For a summary of the settings available in the desktop, mobile, and kiosk devices, see [What you can configure using Configuration Designer wizardS](#).
  - [Instructions for the desktop wizard](#)
  - [Instructions for the mobile wizard](#)
  - [Instructions for the kiosk wizard](#)
  - [Instructions for HoloLens wizard](#)
  - [Instructions for Surface Hub wizard](#)
- The **Advanced provisioning** option opens a new project with all **Runtime settings** available. *The rest of this procedure uses advanced provisioning.*

### TIP

You can start a project in the simple wizard editor and then switch the project to the advanced editor.



- Enter a name for your project, and then click **Next**.
- Select the settings you want to configure, based on the type of device, and then click **Next**. The following table describes the options.

WINDOWS EDITION	SETTINGS AVAILABLE FOR CUSTOMIZATION	PROVISIONING PACKAGE CAN APPLY TO
All Windows editions	Common settings	All Windows 10 devices
All Windows desktop editions	Common settings and settings specific to desktop devices	All Windows 10 desktop editions (Home, Pro, Enterprise, Pro Education, Enterprise Education)
All Windows mobile editions	Common settings and settings specific to mobile devices	All Windows 10 Mobile devices
Windows 10 IoT Core	Common settings and settings specific to Windows 10 IoT Core	All Windows 10 IoT Core devices
Windows 10 Holographic	Common settings and settings specific to Windows 10 Holographic	<a href="#">Microsoft HoloLens</a>
Common to Windows 10 Team edition	Common settings and settings specific to Windows 10 Team	<a href="#">Microsoft Surface Hub</a>

5. On the **Import a provisioning package (optional)** page, you can click **Finish** to create your project, or browse to and select an existing provisioning packge to import to your project, and then click **Finish**.

**TIP**

**Import a provisioning package** can make it easier to create different provisioning packages that all have certain settings in common. For example, you could create a provisioning package that contains the settings for your organization's network, and then import it into other packages you create so you don't have to reconfigure those common settings repeatedly.

After you click **Finish**, Windows Configuration Designer will open the **Available customizations** pane and you can then configure settings for the package.

## Configure settings

For an advanced provisioning project, Windows Configuration Designer opens the **Available customizations** pane. The example in the following image is based on **All Windows desktop editions** settings.

The screenshot shows the Windows Configuration Designer interface. At the top, there are 'File' and 'Export' dropdowns, followed by a 'Start page' button and the title 'Project\_63'. Below the title is a 'Available customizations' section with a 'View: All settings' dropdown and a search bar. The main pane displays a hierarchical list under the 'Runtime settings' category, which is currently expanded. The listed items include: Accounts, AssignedAccess, Browser, Certificates, Connections, ConnectivityProfiles, CountryAndRegion, DesktopBackgroundAndColors, DeviceFormFactor, DeviceManagement, DMClient, EditionUpgrade, Folders, Maps, OOBE, Policies, ProvisioningCommands, SharedPC, SMISettings, Start, TabletMode, TakeATest, UnifiedWriteFilter, UniversalAppInstall, UniversalAppUninstall, UsbErrorsOEMOverride, WeakCharger, and Workplace.

- Runtime settings
  - Accounts
  - AssignedAccess
  - Browser
  - Certificates
  - Connections
  - ConnectivityProfiles
  - CountryAndRegion
  - DesktopBackgroundAndColors
  - DeviceFormFactor
  - DeviceManagement
  - DMClient
  - EditionUpgrade
  - Folders
  - Maps
  - OOBE
  - Policies
  - ProvisioningCommands
  - SharedPC
  - SMISettings
  - Start
  - TabletMode
  - TakeATest
  - UnifiedWriteFilter
  - UniversalAppInstall
  - UniversalAppUninstall
  - UsbErrorsOEMOverride
  - WeakCharger
  - Workplace

The settings in Windows Configuration Designer are based on Windows 10 configuration service providers (CSPs). To learn more about CSPs, see [Introduction to configuration service providers \(CSPs\) for IT pros](#).

The process for configuring settings is similar for all settings. The following table shows an example.

<b>1</b> Expand a category.	<p>The screenshot shows the Windows Configuration Designer interface. On the left, a numbered step 1 with the text 'Expand a category.' is shown. On the right, a detailed view of the 'Available customizations' screen. The 'Certificates' category under 'Runtime settings' is expanded, revealing its sub-categories: CACertificates, ClientCertificates, RootCertificates, TrustedPeopleCertificates, and TrustedProvisioners.</p> <ul style="list-style-type: none"><li>Runtime settings<ul style="list-style-type: none"><li>Certificates<ul style="list-style-type: none"><li>CACertificates</li><li>ClientCertificates</li><li>RootCertificates</li><li>TrustedPeopleCertificates</li><li>TrustedProvisioners</li></ul></li></ul></li></ul>
--------------------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

2

Select a setting.

The screenshot shows the Windows Configuration Designer interface. On the left, the 'Available customizations' pane lists several categories: Certificates, CACertificates, ClientCertificates (which is selected and highlighted in blue), RootCertificates, TrustedPeopleCertificates, and TrustedProvisioners. On the right, the 'ClientCertificates' pane contains instructions: 'To add a new item under ClientCertificates, specify a CertificateName and then select the added item in the Available customizations pane to configure additional settings for this item.' Below this is a text input field labeled 'CertificateName' with the value 'Example' and a 'Add' button.

3

Enter a value for the setting. Click **Add** if the button is displayed.

This screenshot shows the same interface as the previous one, but with a new item added. The 'ClientCertificates' pane now includes the 'Example' entry in its list. The 'Add' button is visible next to the 'CertificateName' input field.

4

Some settings, such as this example, require additional information. In **Available customizations**, select the value you just created, and additional settings are displayed.

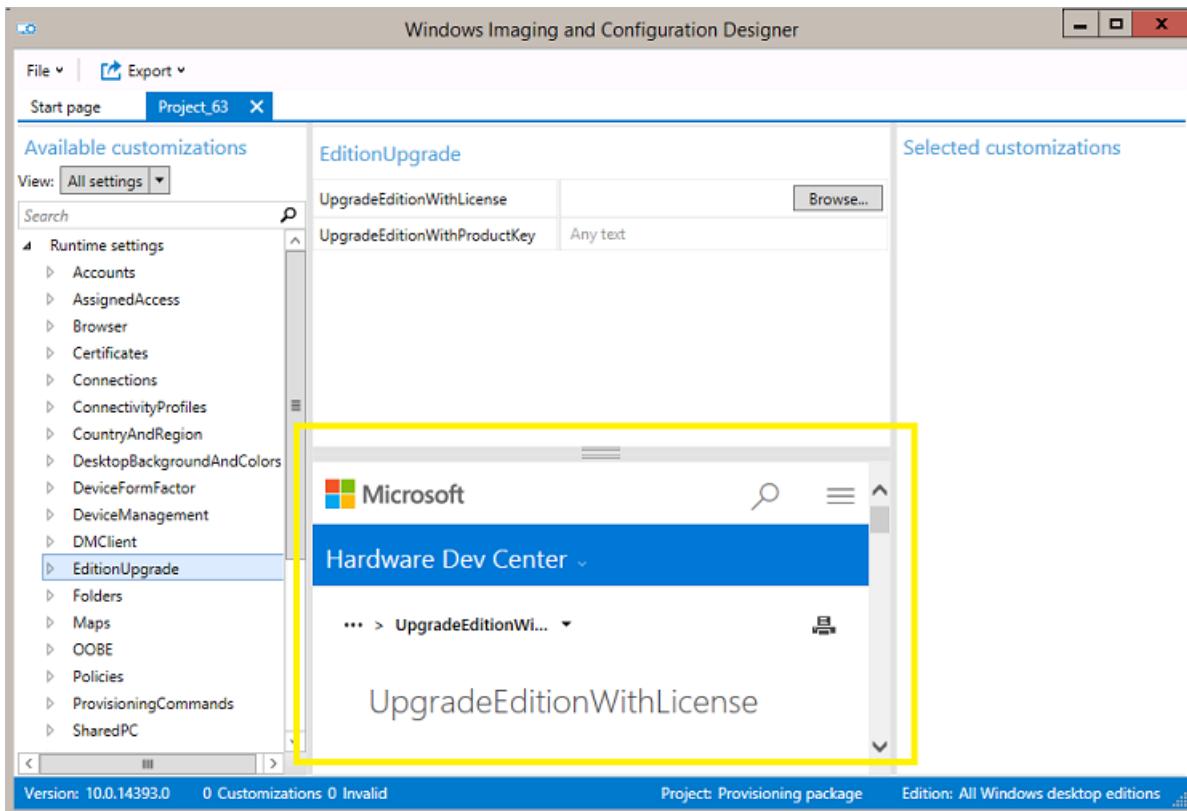
The 'Available customizations' pane now shows the 'ClientCertificates' node expanded, revealing the 'Example' item. When selected, it displays its properties in the main pane: 'CertificatePassword' (input field), 'CertificatePath' (dropdown set to 'NOT CONFIGURED'), 'ExportCertificate' (dropdown set to 'NOT CONFIGURED'), and 'KeyLocation' (dropdown set to 'NOT CONFIGURED').

5

When the setting is configured, it is displayed in the **Selected customizations** pane.

The 'Selected customizations' pane is shown on the right, listing the configured settings for the 'Example' certificate: 'Runtime settings', 'Certificates', 'ClientCertificates', and 'CertificateName: Example'. Under 'CertificateName: Example', the specific settings 'CertificatePassword', 'CertificatePath', 'ExportCertificate', and 'KeyLocation' are listed.

For details on each specific setting, see [Windows Provisioning settings reference](#). The reference topic for a setting is also displayed in Windows Configuration Designer when you select the setting, as shown in the following image.



## Build package

1. After you're done configuring your customizations, click **Export** and select **Provisioning Package**.



2. In the **Describe the provisioning package** window, enter the following information, and then click **Next:**

- **Name** - This field is pre-populated with the project name. You can change this value by entering a different name in the **Name** field.
- **Version (in Major.Minor format)** - Optional. You can change the default package version by specifying a new value in the **Version** field.
- **Owner** - Select **IT Admin**. For more information, see [Precedence for provisioning packages](#).
- **Rank (between 0-99)** - Optional. You can select a value between 0 and 99, inclusive. The default package rank is 0.

3. In the **Select security details for the provisioning package** window, you can select to encrypt and/or sign a provisioning package with a selected certificate. Both selections are optional. Click **Next** after you make your selections.

- **Encrypt package** - If you select this option, an auto-generated password will be shown on the screen.
- **Sign package** - If you select this option, you must select a valid certificate to use for signing the package. You can specify the certificate by clicking **Select** and choosing the certificate you want to use to sign the package.

#### **NOTE**

You should only configure provisioning package security when the package is used for device provisioning and the package has contents with sensitive security data such as certificates or credentials that should be prevented from being compromised. When applying an encrypted and/or signed provisioning package, either during OOOE or through the setting UI, the package can be decrypted, and if signed, be trusted without explicit user consent. An IT administrator can set policy on a user device to restrict the removal of required packages from the device, or the provisioning of potentially harmful packages on the device.

If a provisioning package is signed by a trusted provisioner, it can be installed on a device without a prompt for user consent. In order to enable trusted provider certificates, you must set the

**TrustedProvisioners** setting prior to installing the trusted provisioning package. This is the only way to install a package without user consent. To provide additional security, you can also set

**RequireProvisioningPackageSignature**, which prevents users from installing provisioning packages that are not signed by a trusted provisioner.

4. In the **Select where to save the provisioning package** window, specify the output location where you want the provisioning package to go once it's built, and then click **Next**. By default, Windows Configuration Designer uses the project folder as the output location.
5. In the **Build the provisioning package** window, click **Build**. The provisioning package doesn't take long to build. The project information is displayed in the build page and the progress bar indicates the build status.  
If you need to cancel the build, click **Cancel**. This cancels the current build process, closes the wizard, and takes you back to the Customizations Page.
6. If your build fails, an error message will show up that includes a link to the project folder. You can scan the logs to determine what caused the error. Once you fix the issue, try building the package again.  
If your build is successful, the name of the provisioning package, output directory, and project directory will be shown.  
If you choose, you can build the provisioning package again and pick a different path for the output package. To do this, click **Back** to change the output package name and path, and then click **Next** to start another build.
7. When you are done, click **Finish** to close the wizard and go back to the Customizations page.

**Next step:** [How to apply a provisioning package](#)

## Learn more

- Watch the video: [Provisioning Windows 10 Devices with New Tools](#)
- Watch the video: [Windows 10 for Mobile Devices: Provisioning Is Not Imaging](#)
- [How to bulk-enroll devices with On-premises Mobile Device Management in System Center Configuration Manager](#)

## Related topics

- [Provisioning packages for Windows 10](#)
- [How provisioning works in Windows 10](#)
- [Install Windows Configuration Designer](#)
- [Apply a provisioning package](#)

- [Settings changed when you uninstall a provisioning package](#)
- [Provision PCs with common settings for initial deployment \(simple provisioning\)](#)
- [Use a script to install a desktop app in provisioning packages](#)
- [PowerShell cmdlets for provisioning Windows 10 \(reference\)](#)
- [Windows Configuration Designer command-line interface \(reference\)](#)
- [Create a provisioning package with multivariant settings](#)

# Apply a provisioning package

10/17/2017 • 2 min to read • [Edit Online](#)

## Applies to

- Windows 10
- Windows 10 Mobile

Provisioning packages can be applied to a device during the first-run experience (out-of-box experience or "OOBE") and after ("runtime").

### NOTE

Applying a provisioning package to a desktop device requires administrator privileges on the device.

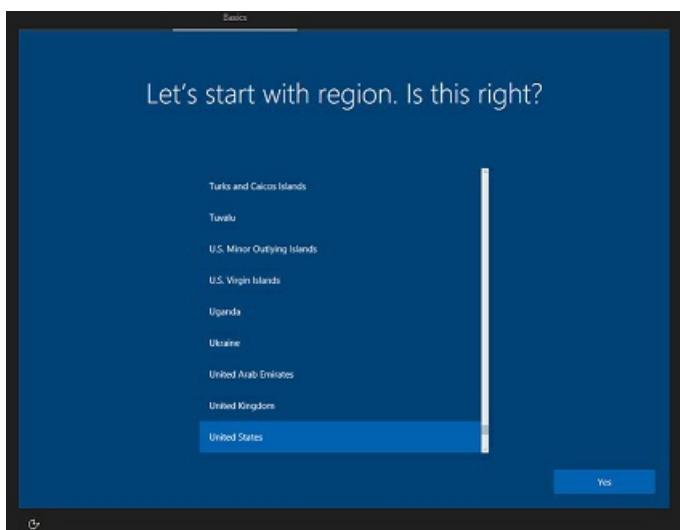
## Desktop editions

### NOTE

In Windows 10, version 1709, you can interrupt a long-running provisioning process by pressing ESC.

### During initial setup, from a USB drive

1. Start with a computer on the first-run setup screen. If the PC has gone past this screen, reset the PC to start over. To reset the PC, go to **Settings > Update & security > Recovery > Reset this PC**.



2. Insert the USB drive. Windows Setup will recognize the drive and ask if you want to set up the device. Select **Set up**.

## Set up PC?

You'll need to install a provisioning package to get things from your work or school.

Set up

Cancel

3. The next screen asks you to select a provisioning source. Select **Removable Media** and tap **Next**.

## Provision this device

Provisioning helps set up your device for work or school. If you're provisioning from removable media, make sure it's connected.

Provision from

Removable Media

Cancel

Next

4. Select the provisioning package (\*.ppkg) that you want to apply, and tap **Next**.

## Choose a package

You can add more later in Settings.

Choose one

\* .ppkg

Back

Next

5. Select **Yes, add it**.

## Is this package from a source you trust?

Only add this package (SetupSchoolPCs.ppkg) if you know who it came from.

This package:

Changes device customizations  
Adds or changes applications  
Adds or changes file content (documents etc.)  
Runs scripts at the system level  
Applies policies

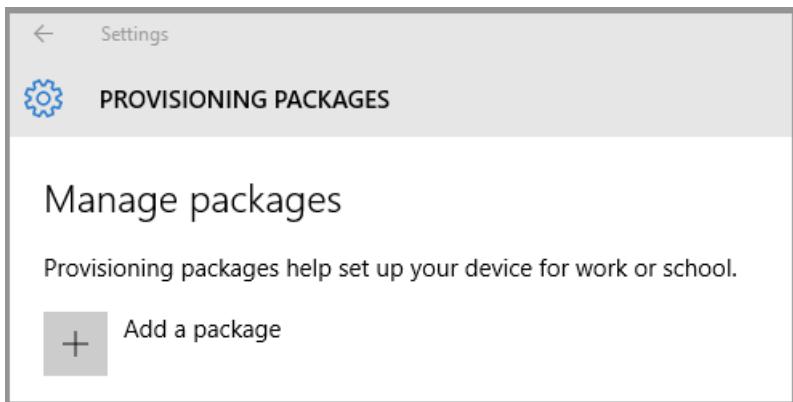
Yes, add it

Cancel

## After setup, from a USB drive, network folder, or SharePoint site

Insert the USB drive to a desktop computer, navigate to **Settings > Accounts > Access work or school > Add or remove a provisioning package > Add a package**, and select the package to install. For a provisioning package stored on a network folder or on a SharePoint site, navigate to the provisioning package and double-

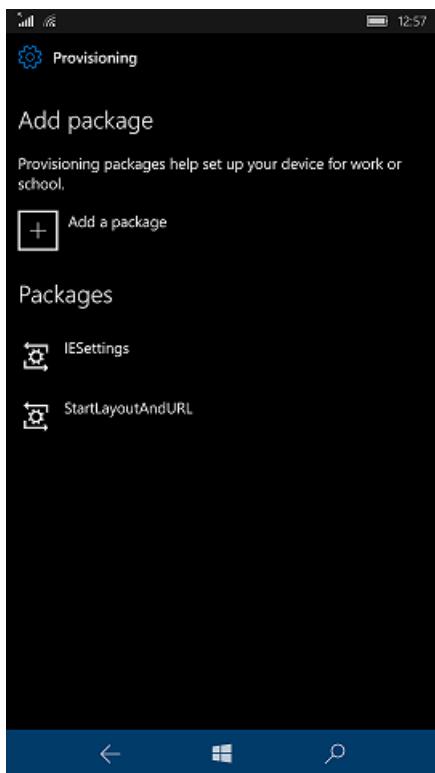
click it to begin installation.



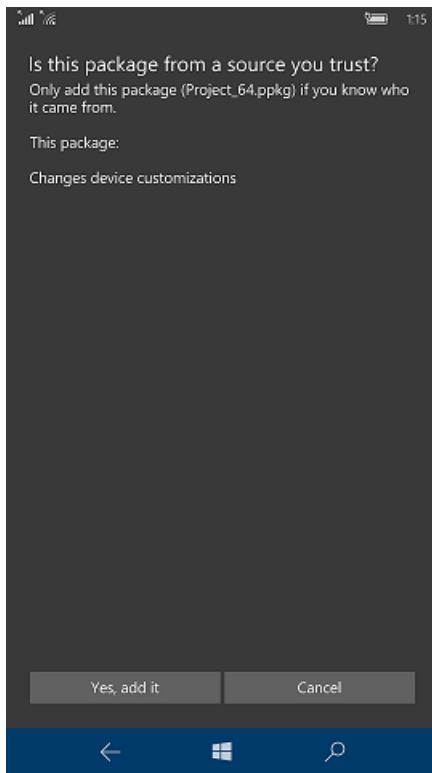
## Mobile editions

### Using removable media

1. Insert an SD card containing the provisioning package into the device.
2. Navigate to **Settings > Accounts > Access work or school > Add or remove a provisioning package > Add a package**, and select the package to install.

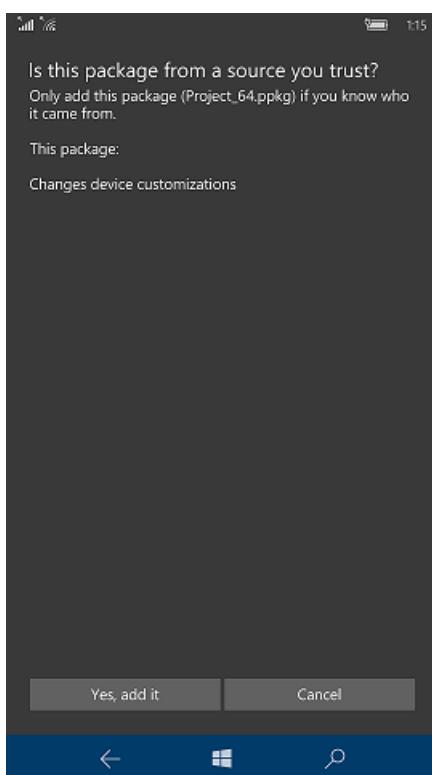


3. Click **Add**.
4. On the device, the **Is this package from a source you trust?** message will appear. Tap **Yes, add it**.



### Copying the provisioning package to the device

1. Connect the device to your PC through USB.
2. On the PC, select the provisioning package that you want to use to provision the device and then drag and drop the file to your device.
3. On the device, the **Is this package from a source you trust?** message will appear. Tap **Yes, add it**.



## Related topics

- [Provisioning packages for Windows 10](#)
- [How provisioning works in Windows 10](#)
- [Install Windows Configuration Designer](#)

- [Create a provisioning package](#)
- [Settings changed when you uninstall a provisioning package](#)
- [Provision PCs with common settings for initial deployment \(simple provisioning\)](#)
- [Use a script to install a desktop app in provisioning packages](#)
- [PowerShell cmdlets for provisioning Windows 10 \(reference\)](#)
- [Windows Configuration Designer command-line interface \(reference\)](#)
- [Create a provisioning package with multivariant settings](#)

# Settings changed when you uninstall a provisioning package

7/28/2017 • 1 min to read • [Edit Online](#)

## Applies to

- Windows 10
- Windows 10 Mobile

When you uninstall a provisioning package, only certain settings are revertible. This topic lists the settings that are reverted when you uninstall a provisioning package.

As an administrator, you can uninstall by using the **Add or remove a package for work or school** option available under **Settings > Accounts > Access work or school**.

When a provisioning package is uninstalled, some of its settings are reverted, which means the value for the setting is changed to the next available or default value. Not all settings, however, are revertible.

Only settings in the following lists are revertible.

## Registry-based settings

The registry-based settings that are revertible when a provisioning package is uninstalled all fall under these categories, which you can find in the Windows Configuration Designer.

- [Wi-Fi Sense](#)
- [CountryAndRegion](#)
- DeviceManagement / PGList/ LogicalProxyName
- UniversalAppInstall / LaunchAppAtLogin
- [Power](#)
- [TabletMode](#)
- [Maps](#)
- [Browser](#)
- [DeviceFormFactor](#)
- [USBErrorsOEMOverride](#)
- [WeakCharger](#)

## CSP-based settings

Here is the list of revertible settings based on configuration service providers (CSPs).

[ActiveSync CSP](#) [AppLocker CSP](#)

[BrowserFavorite CSP](#)

[CertificateStore CSP](#) [ClientCertificateInstall CSP](#)

[RootCATrustedCertificates CSP](#)

[CM\\_CellularEntries CSP](#)

[CM\\_ProxyEntries CSP](#)

[CMPolicy CSP](#)

[CMPolicyEnterprise CSP](#)

[EMAIL2 CSP](#)

[EnterpriseAPN CSP](#)

[EnterpriseAppManagement CSP](#)

[EnterpriseDesktopAppManagement CSP](#)

[EnterpriseModernAppManagement CSP](#)

[NAP CSP](#)

[PassportForWork CSP](#)

[Provisioning CSP](#)

[PROXY CSP](#)

[SecureAssessment CSP](#)

[VPN CSP](#)

[VPNv2 CSP](#)

[WiFi CSP](#)

## Related topics

- [Provisioning packages for Windows 10](#)
- [How provisioning works in Windows 10](#)
- [Install Windows Configuration Designer](#)
- [Create a provisioning package](#)
- [Apply a provisioning package](#)
- [Provision PCs with common settings for initial deployment \(simple provisioning\)](#)
- [Use a script to install a desktop app in provisioning packages](#)
- [PowerShell cmdlets for provisioning Windows 10 \(reference\)](#)
- [Windows Configuration Designer command-line interface \(reference\)](#)
- [Create a provisioning package with multivariant settings](#)

# Provision PCs with common settings for initial deployment (desktop wizard)

7/28/2017 • 4 min to read • [Edit Online](#)

## Applies to

- Windows 10

This topic explains how to create and apply a provisioning package that contains common enterprise settings to a device running all desktop editions of Windows 10 except Windows 10 Home.

You can apply a provisioning package on a USB drive to off-the-shelf devices during setup, making it fast and easy to configure new devices.

## Advantages

- You can configure new devices without reimaging.
- Works on both mobile and desktop devices.
- No network connectivity required.
- Simple to apply.

[Learn more about the benefits and uses of provisioning packages.](#)

## What does the desktop wizard do?

The desktop wizard helps you configure the following settings in a provisioning package:

- Set device name
- Upgrade product edition
- Configure the device for shared use
- Remove pre-installed software
- Configure Wi-Fi network
- Enroll device in Active Directory or Azure Active Directory
- Create local administrator account
- Add applications and certificates

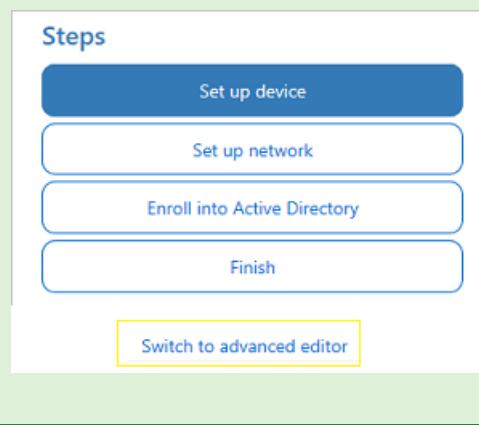
### WARNING

You must run Windows Configuration Designer on Windows 10 to configure Azure Active Directory enrollment using any of the wizards.

Provisioning packages can include management instructions and policies, installation of specific apps, customization of network connections and policies, and more.

**TIP**

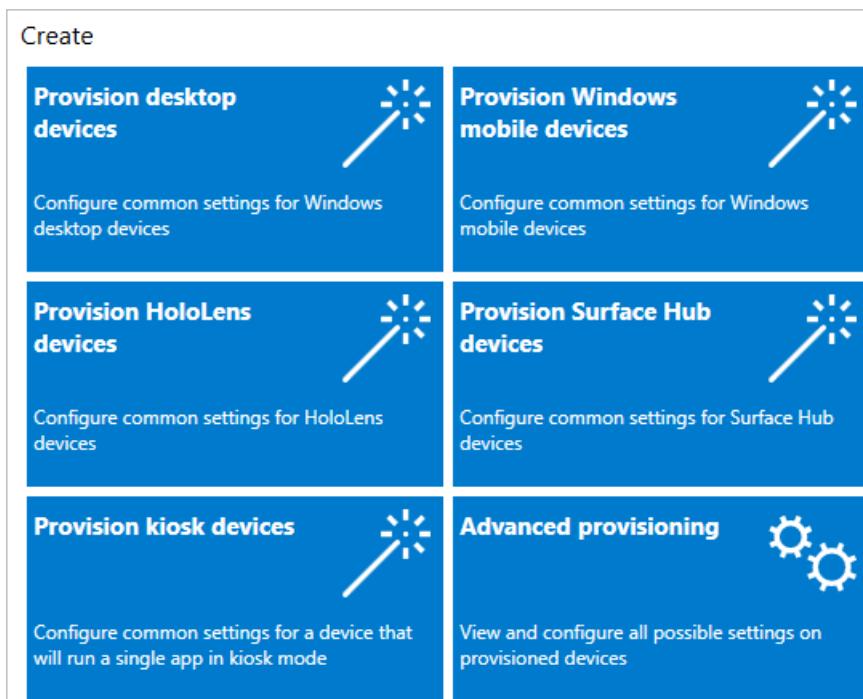
Use the desktop wizard to create a package with the common settings, then switch to the advanced editor to add other settings, apps, policies, etc.



## Create the provisioning package

Use the Windows Configuration Designer tool to create a provisioning package. [Learn how to install Windows Configuration Designer](#).

1. Open Windows Configuration Designer (by default, %windir%\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Imaging and Configuration Designer\x86\ICD.exe).
2. Click **Provision desktop devices**.



3. Name your project and click **Finish**. The pages for desktop provisioning will walk you through the following steps.

## Steps

Set up device

Set up network

Account Management

Add applications

Add certificates

Finish

### IMPORTANT

When you build a provisioning package, you may include sensitive information in the project files and in the provisioning package (.ppkg) file. Although you have the option to encrypt the .ppkg file, project files are not encrypted. You should store the project files in a secure location and delete the project files when they are no longer needed.

## Configure settings

1

Set up device

Enter a name for the device.

(Optional) Select a license file to upgrade Windows 10 to a different edition. [See the permitted upgrades](#).

Toggle **Yes** or **No** to **Configure devices for shared use**. This setting optimizes Windows 10 for shared use scenarios. [Learn more about shared PC configuration](#).

You can also select to remove pre-installed software from the device.

### Device name

Enter a unique 15-character name for the device. For help generating a unique name, you can use %SERIAL%, which includes a hardware-specific serial number, or you can use %RANDx%, which generates random characters of x length.

Example device name values:

Contoso-%SERIAL%

Fabrikam-%RAND:5%

### Enter product key

Optional: Enter a product key to upgrade Windows.

XXXXXX-XXXXXX-XXXXXX-XXXXXX-XXXXXX

### Configure devices for shared use

Allow students to quickly login with their credentials or as an anonymous guest, and store all their work in the cloud

No

### Remove pre-installed software

Optional: remove pre-installed software without keeping any user data

No

**2**

## Set up network

Toggle **On** or **Off** for wireless network connectivity. If you select **On**, enter the SSID, the network type (**Open** or **WPA2-Personal**), and (if **WPA2-Personal**) the password for the wireless network.

## Set up network

Connect devices to a Wi-Fi network

On

Network SSID \*

|

Required

Network type \*

Open

**3**

## Account Management

Enable account management if you want to configure settings on this page.

You can enroll the device in Active Directory, enroll in Azure Active Directory, or create a local administrator account on the device.

To enroll the device in Active Directory, enter the credentials for a least-privileged user account to join the device to the domain.

Before you use a Windows Configuration Designer wizard to configure bulk Azure AD enrollment, [set up Azure AD join in your organization](#). The **maximum number of devices per user** setting in your Azure AD tenant determines how many times the bulk token that you get in the wizard can be used. To enroll the device in Azure AD, select that option and enter a friendly name for the bulk token you will get using the wizard. Set an expiration date for the token (maximum is 30 days from the date you get the token). Click **Get bulk token**. In the **Let's get you signed in** window, enter an account that has permissions to join a device to Azure AD, and then the password. Click **Accept** to give Windows Configuration Designer the necessary permissions.

To create a local administrator account, select that option and enter a user name and password.

**Important:** If you create a local account in the provisioning package, you must change the password using the **Settings** app every 42 days. If the password is not changed during that period, the account might be locked out and unable to sign in.

Enabled

### Manage Organization/School Accounts

Improve security and remote management by enrolling devices into Active Directory

- [Enroll into Active Directory](#)
- [Enroll in Azure AD](#)
- [Local Admin](#)

### Create a local administrator account

User name \*

|

Required

Password \*

|

Required

<p><b>4</b></p> <p><b>Add applications</b></p> <p>You can install multiple applications, both Classic Windows (Win32) apps and Universal Windows Platform (UWP) apps, in a provisioning package. The settings in this step vary according to the application that you select. For help with the settings, see <a href="#">Provision PCs with apps</a>.</p>	<h3>Add an Application</h3> <p><b>Application name</b> Specify the name of the application <input type="text"/></p> <p><b>Required</b></p> <p><b>Installer Path</b> Specify the installer file <input type="text"/> <b>Browse</b></p> <p><b>Required</b></p>
<p><b>5</b></p> <p><b>Add certificates</b></p> <p>To provision the device with a certificate, click <b>Add a certificate</b>. Enter a name for the certificate, and then browse to and select the certificate to be used.</p>	<h3>Add a certificate</h3> <p><b>Certificate name</b> Specify a name for the certificate <input type="text"/></p> <p><b>Required</b></p> <p><b>Certificate path</b> Choose the certificate file <input type="text"/> <b>Browse</b></p> <p><b>Required</b></p>
<p><b>Finish</b></p> <p>You can set a password to protect your provisioning package. You must enter this password when you apply the provisioning package to a device.</p>	<h3>Summary</h3> <p><b>Protect your package</b> Protect the contents of your package by specifying a password. The password length must be 8-16 characters. <input checked="" type="checkbox"/> No</p>

After you're done, click **Create**. It only takes a few seconds. When the package is built, the location where the package is stored is displayed as a hyperlink at the bottom of the page.

**Next step:** [How to apply a provisioning package](#)

## Learn more

- Watch the video: [Provisioning Windows 10 Devices with New Tools](#)
- Watch the video: [Windows 10 for Mobile Devices: Provisioning Is Not Imaging](#)

## Related topics

- [Provisioning packages for Windows 10](#)
- [How provisioning works in Windows 10](#)
- [Install Windows Configuration Designer](#)
- [Create a provisioning package](#)
- [Apply a provisioning package](#)
- [Settings changed when you uninstall a provisioning package](#)
- [Use a script to install a desktop app in provisioning packages](#)

- [PowerShell cmdlets for provisioning Windows 10 \(reference\)](#)
- [NFC-based device provisioning](#)
- [Use the package splitter tool](#)
- [Windows Configuration Designer command-line interface \(reference\)](#)
- [Create a provisioning package with multivariant settings](#)

# Provision PCs with apps

10/17/2017 • 7 min to read • [Edit Online](#)

## Applies to

- Windows 10

In Windows 10, version 1703, you can install multiple Universal Windows Platform (UWP) apps and Classic Windows (Win32) applications in a provisioning package. This topic explains the various settings in [Windows Configuration Designer](#) for app install.

When you add an app in a Windows Configuration Designer wizard, the appropriate settings are displayed based on the app that you select. For instructions on adding an app using the advanced editor in Windows Configuration Designer, see [Add an app using advanced editor](#).

### IMPORTANT

If you plan to use Intune to manage your devices, we recommend using Intune to install Office 365 ProPlus 2016 apps (Access, Excel, OneDrive for Business, OneNote, Outlook, PowerPoint, Publisher, Skype for Business, Word, Project Online Desktop Client, and Visio Pro for Office 365 ProPlus). Apps that are installed using a provisioning package cannot be managed or modified using Intune. [Learn how to assign Office 365 ProPlus 2016 apps using Microsoft Intune](#).

## Settings for UWP apps

- License Path:** Specify the license file if it is an app from the Microsoft Store. This is optional if you have a certificate for the app.
- Package family name:** Specify the package family name if you don't specify a license. This field will be auto-populated after you specify a license.
- Required appx dependencies:** Specify the appx dependency packages that are required for the installation of the app

## Settings for Classic Windows apps

### MSI installer

- Command line arguments:** Optionally, append additional command arguments. The silent flag is appended for you. Example: PROPERTY=VALUE
- Continue installations after failure:** Optionally, specify if you want to continue installing additional apps if this app fails to install
- Restart required:** Optionally, specify if you want to initiate a reboot after a successful install of this app
- Required win32 app dependencies:** Optionally, specify additional files that are required for the installation of the app. For installers that have multiple file dependencies or have directory structures, [create a cab file of the assets](#). The installation script should [include expansion of the .cab file](#).

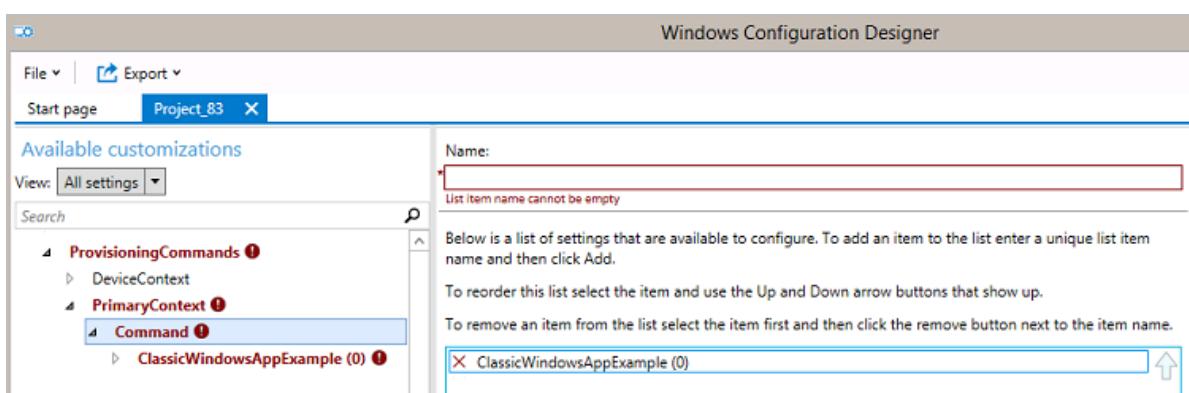
### Exe or other installer

- Command line arguments:** Append the command line arguments with a silent flag (required). Optionally, append additional flags

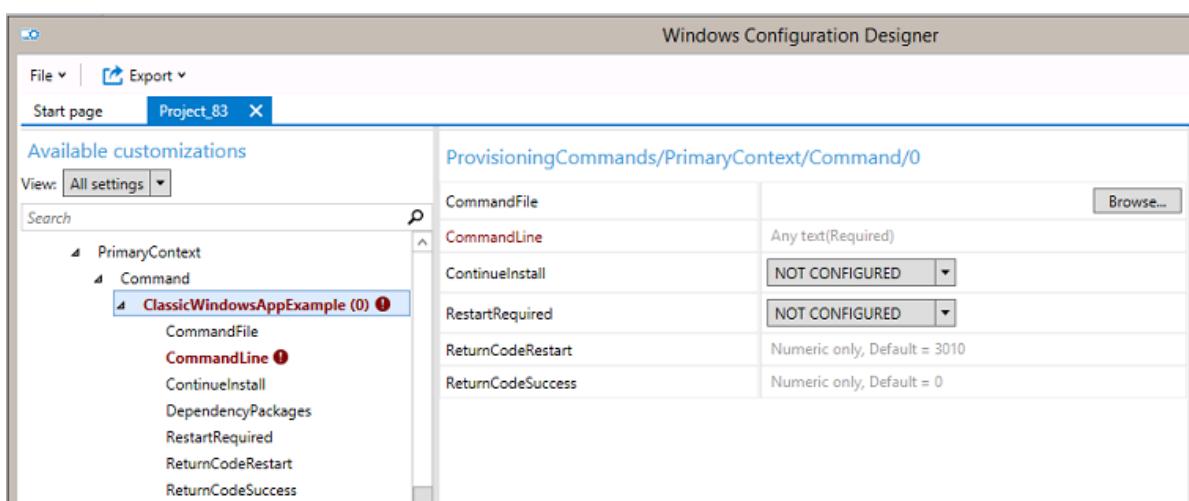
- **Return Codes:** Specify the return codes for success and success with restart (0 and 3010 by default respectively) Any return code that is not listed will be interpreted as failure. The text boxes are space delimited.
- **Continue installations after failure:** Optionally, specify if you want to continue installing additional apps if this app fails to install
- **Restart required:** Optionally, specify if you want to initiate a reboot after a successful install of this app
- **Required win32 app dependencies:** Optionally, specify additional files that are required for the installation of the app. For installers that have multiple file dependencies or have directory structures, [create a cab file of the assets](#). The installation script should [include expansion of the .cab file](#).

## Add a Classic Windows app using advanced editor in Windows Configuration Designer

1. In the **Available customizations** pane, go to **Runtime settings > ProvisioningCommands > PrimaryContext > Command**.
2. Enter a name for the first app, and then click **Add**.



3. Configure the settings for the appropriate installer type.



### Add a universal app to your package

Universal apps that you can distribute in the provisioning package can be line-of-business (LOB) apps developed by your organization, Microsoft Store for Business apps that you acquire with [offline licensing](#), or third-party apps. This procedure will assume you are distributing apps from the Microsoft Store for Business. For other apps, obtain the necessary information (such as the package family name) from the app developer.

1. In the **Available customizations** pane, go to **Runtime settings > UniversalAppInstall**.

2. For **DeviceContextApp**, specify the **PackageFamilyName** for the app. In Microsoft Store for Business, the package family name is listed in the **Package details** section of the download page.

Package details
Download the package for offline use
Package identity name: Microsoft.MicrosoftPowerBIForWindows
Package family name: Microsoft.MicrosoftPowerBIForWindows_8wekyb3d8bbwe

3. For **ApplicationFile**, click **Browse** to find and select the target app (either an \*.appx or \*.appxbundle).
4. For **DependencyAppxFiles**, click **Browse** to find and add any dependencies for the app. In Microsoft Store for Business, any dependencies for the app are listed in the **Required frameworks** section of the download page.

Required frameworks
Microsoft.VCLibs.140.00_14.0.24123.0_x86_8wekyb3d8bbwe
File size: 607410 bytes
Architecture X86
<a href="#">Download</a>

5. For **DeviceContextAppLicense**, enter the **LicenseProductId**.

- In Microsoft Store for Business, generate the unencoded license for the app on the app's download page.

Get app license
<input type="radio"/> Encoded license
<input checked="" type="radio"/> Unencoded license
<a href="#">Generate license</a>

- Open the license file and search for **LicenseID=** to get the GUID, enter the GUID in the **LicenseProductId** field and click **Add**.
6. In the **Available customizations** pane, click the **LicenseProductId** that you just added.
7. For **LicensesInstall**, click **Browse**, navigate to the license file that you renamed **.ms-windows-store-license**, and select the license file.

[Learn more about distributing offline apps from the Microsoft Store for Business.](#)

**NOTE**

Removing a provisioning package will not remove any apps installed by device context in that provisioning package.

### Add a certificate to your package

- In the **Available customizations** pane, go to **Runtime settings > Certificates > ClientCertificates**.
- Enter a **CertificateName** and then click **Add**.
- Enter the **CertificatePassword**.
- For **CertificatePath**, browse and select the certificate to be used.

5. Set **ExportCertificate** to **False**.
6. For **KeyLocation**, select **Software only**.

#### Add other settings to your package

For details about the settings you can customize in provisioning packages, see [Windows Provisioning settings reference](#).

#### Build your package

1. When you are done configuring the provisioning package, on the **File** menu, click **Save**.
2. Read the warning that project files may contain sensitive information, and click **OK**.

**Important** When you build a provisioning package, you may include sensitive information in the project files and in the provisioning package (.ppkg) file. Although you have the option to encrypt the .ppkg file, project files are not encrypted. You should store the project files in a secure location and delete the project files when they are no longer needed.

3. On the **Export** menu, click **Provisioning package**.
4. Change **Owner** to **IT Admin**, which will set the precedence of this provisioning package higher than provisioning packages applied to this device from other sources, and then select **Next**.
5. Set a value for **Package Version**.

#### TIP

You can make changes to existing packages and change the version number to update previously applied packages.

6. Optional. In the **Provisioning package security** window, you can choose to encrypt the package and enable package signing.
  - **Enable package encryption** - If you select this option, an auto-generated password will be shown on the screen.
  - **Enable package signing** - If you select this option, you must select a valid certificate to use for signing the package. You can specify the certificate by clicking **Select...** and choosing the certificate you want to use to sign the package.

#### Important

We recommend that you include a trusted provisioning certificate in your provisioning package. When the package is applied to a device, the certificate is added to the system store and any package signed with that certificate thereafter can be applied silently.

7. Click **Next** to specify the output location where you want the provisioning package to go once it's built. By default, Windows ICD uses the project folder as the output location.

Optionally, you can click **Browse** to change the default output location.

8. Click **Next**.
9. Click **Build** to start building the package. The project information is displayed in the build page and the progress bar indicates the build status.

If you need to cancel the build, click **Cancel**. This cancels the current build process, closes the wizard, and takes you back to the **Customizations Page**.

10. If your build fails, an error message will show up that includes a link to the project folder. You can scan the

logs to determine what caused the error. Once you fix the issue, try building the package again.

If your build is successful, the name of the provisioning package, output directory, and project directory will be shown.

- If you choose, you can build the provisioning package again and pick a different path for the output package. To do this, click **Back** to change the output package name and path, and then click **Next** to start another build.

- If you are done, click **Finish** to close the wizard and go back to the **Customizations Page**.

11. Select the **output location** link to go to the location of the package. You can provide that .ppkg to others through any of the following methods:

- Shared network folder
- SharePoint site
- Removable media (USB/SD)
- Email
- USB tether (mobile only)
- NFC (mobile only)

**Next step:** [How to apply a provisioning package](#)

## Learn more

- Watch the video: [Provisioning Windows 10 Devices with New Tools](#)
- Watch the video: [Windows 10 for Mobile Devices: Provisioning Is Not Imaging](#)

## Related topics

- [Provisioning packages for Windows 10](#)
- [How provisioning works in Windows 10](#)
- [Install Windows Configuration Designer](#)
- [Create a provisioning package](#)
- [Apply a provisioning package](#)
- [Settings changed when you uninstall a provisioning package](#)
- [Provision PCs with common settings for initial deployment \(simple provisioning\)](#)
- [Use a script to install a desktop app in provisioning packages](#)
- [PowerShell cmdlets for provisioning Windows 10 \(reference\)](#)
- [NFC-based device provisioning](#)
- [Use the package splitter tool](#)
- [Windows Configuration Designer command-line interface \(reference\)](#)
- [Create a provisioning package with multivariant settings](#)

# Use a script to install a desktop app in provisioning packages

7/28/2017 • 8 min to read • [Edit Online](#)

## Applies to

- Windows 10
- Windows 10 Mobile

This walkthrough describes how to leverage the ability to include scripts in a Windows 10 provisioning package to install Win32 applications. Scripted operations other than installing apps can also be performed, however, some care is needed in order to avoid unintended behavior during script execution (see [Remarks](#) below).

**Prerequisite:** [Windows Assessment and Deployment Kit \(ADK\) for Windows 10](#), version 1511 or higher

### NOTE

This scenario is only supported for installing applications on Windows 10 for desktop, version 1511 or higher.

## Assemble the application assets

1. On the device where you're authoring the package, place all of your assets in a known location. Each asset must have a unique filename, because all files will be copied to the same temp directory on the device. It's common for many apps to have an installer called 'install.exe' or similar, and there may be name overlap because of that. To fix this, you can use the technique described in the next step to include a complete directory structure that is then expanded into the temp directory on the device. The most common use for this would be to include a subdirectory for each application.
2. If you need to include a directory structure of files, you will need to cab the assets for easy inclusion in the provisioning packages.

## Cab the application assets

1. Create a .DDF file as below, replacing *file1* and *file2* with the files you want to package, and adding the name of file/directory.

```

;*** MSDN Sample Source Code MakeCAB Directive file example

;

.OPTION EXPLICIT ; Generate errors on variable typos

.set DiskDirectoryTemplate=CDROM ; All cabinets go in a single directory

.set MaxDiskFileCount=1000; Limit file count per cabinet, so that

; scanning is not too slow

.set FolderSizeThreshold=200000 ; Aim for ~200K per folder

.set CompressionType=MSZIP

;** All files are compressed in cabinet files

.set Cabinet=on

.set Compress=on

;-----

;** CabinetNameTemplate = name of cab

;** DiskDirectory1 = output directory where cab will be created

;-----

.set CabinetNameTemplate=tt.cab

.set DiskDirectory1=.

;-----

; Replace <file> with actual files you want to package

;-----

<file1>

<file2>

;*** <the end>

```

## 2. Use makecab to create the cab files.

```
Makecab -f <path to DDF file>
```

## Create the script to install the application

In Windows 10, version 1607 and earlier, create a script to perform whatever work is needed to install the application(s). The following examples are provided to help get started authoring the orchestrator script that will execute the required installers. In practice, the orchestrator script may reference many more assets than those in these examples.

In Windows 10, version 1703, you don't need to create an orchestrator script. You can have one command line per app. If necessary, you can create a script that logs the output per app, as mentioned below (rather than one orchestrator script for the entire provisioning package).

## NOTE

All actions performed by the script must happen silently, showing no UI and requiring no user interaction.

The scripts will be run on the device in system context.

## Debugging example

Granular logging is not built in, so the logging must be built into the script itself. Here is an example script that logs 'Hello World' to a logfile. When run on the device, the logfile will be available after provisioning is completed. As you will see in the following examples, it's recommended that you log each action that your script performs.

```
set LOGFILE=%SystemDrive%\HelloWorld.log
echo Hello, World >> %LOGFILE%
```

## .exe example

This example script shows how to create a log output file on the system drive, install an app from a .exe installer, and echo the results to the log file.

```
set LOGFILE=%SystemDrive%\Fiddler_install.log
echo Installing Fiddler.exe >> %LOGFILE%
fiddler4setup.exe /S >> %LOGFILE%
echo result: %ERRORLEVEL% >> %LOGFILE%
```

## .msi example

This is the same as the previous installer, but installs the app from an MSI installer. Notice that msieexec is called with the /quiet flag in order to meet the silent requirement of scripts run from within a provisioning package.

```
set LOGFILE=%SystemDrive%\IpOverUsb_install.log
echo Installing IpOverUsbInstaller.msi >> %LOGFILE%
msiexec /i IpOverUsbInstaller.msi /quiet >> %LOGFILE%
echo result: %ERRORLEVEL% >> %LOGFILE%
```

## PowerShell example

This is an example script with logging that shows how to run a powershell script from the provisioning commands setting. Note that the PowerShell script referenced from this example must also be included in the package, and obey the same requirements as all scripts run from within the provisioning package: it must execute silently, with no user interaction.

```
set LOGFILE=%SystemDrive%\my_powershell_script.log
echo Running my_powershell_script.ps1 in system context >> %LOGFILE%
echo Executing "PsExec.exe -accepteula -i -s cmd.exe /c powershell.exe my_powershell_script.ps1" >>
%LOGFILE%
PsExec.exe -accepteula -i -s cmd.exe /c powershell.exe my_powershell_script.ps1' >> %LOGFILE%
echo result: %ERRORLEVEL% >> %LOGFILE%
```

## Extract from a .CAB example

This example script shows expansion of a .cab from the provisioning commands script, as well as installation of the expanded setup.exe

```

set LOGFILE=%SystemDrive%\install_my_app.log
echo Expanding installer_assets.cab >> %LOGFILE%
expand -r installer_assets.cab -F:* . >> %LOGFILE%
echo result: %ERRORLEVEL% >> %LOGFILE%
echo Installing MyApp >> %LOGFILE%
setup.exe >> %LOGFILE%
echo result: %ERRORLEVEL% >> %LOGFILE%

```

## Calling multiple scripts in the package

In Windows 10, version 1703, your provisioning package can include multiple CommandLine.

In Windows 10, version 1607 and earlier, you are allowed one CommandLine per provisioning package. The batch files shown above are orchestrator scripts that manage the installation and call any other scripts included in the provisioning package. The orchestrator script is what should be invoked from the CommandLine specified in the package.

Here's a table describing this relationship, using the PowerShell example from above:

ICD SETTING	VALUE	DESCRIPTION
ProvisioningCommands/DeviceContext /CommandLine	cmd /c PowerShell_Example.bat	The command line needed to invoke the orchestrator script.
ProvisioningCommands/DeviceContext /CommandFiles	PowerShell_Example.bat	The single orchestrator script referenced by the command line that handles calling into the required installers or performing any other actions such as expanding cab files. This script must do the required logging.
ProvisioningCommands/DeviceContext /CommandFiles	my_powershell_script.ps1	Other assets referenced by the orchestrator script. In this example there is only one, but there could be many assets referenced here. One common use case is using the orchestrator to call a series of install.exe or setup.exe installers to install several applications. Each of those installers must be included as an asset here.

## Add script to provisioning package (Windows 10, version 1607)

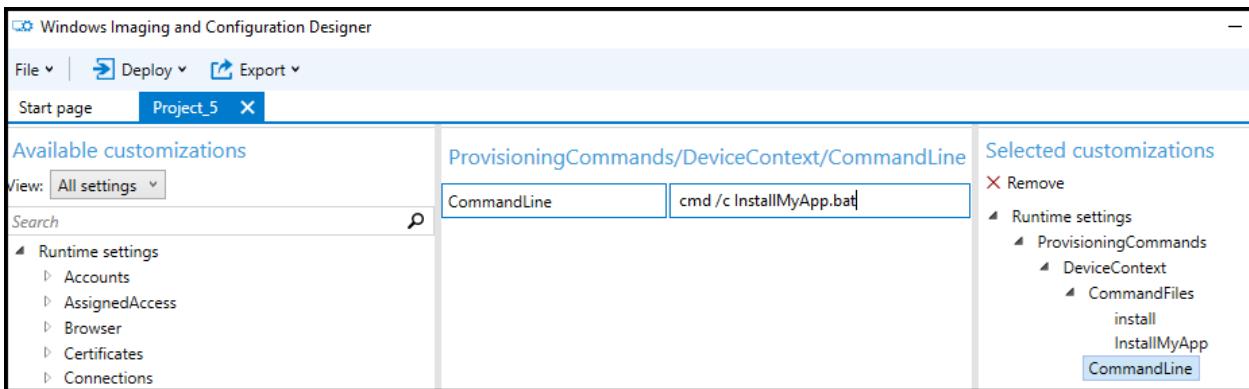
When you have the batch file written and the referenced assets ready to include, you can add them to a provisioning package in the Window Configuration Designer.

Using Windows Configuration Designer, specify the full details of how the script should be run in the CommandLine setting in the provisioning package. This includes flags or any other parameters that you would normally type on the command line. So for example if the package contained an app installer called install.exe and a script used to automate the install called InstallMyApp.bat, the

`ProvisioningCommands/DeviceContext/CommandLine` setting should be configured to:

```
cmd /c InstallMyApp.bat
```

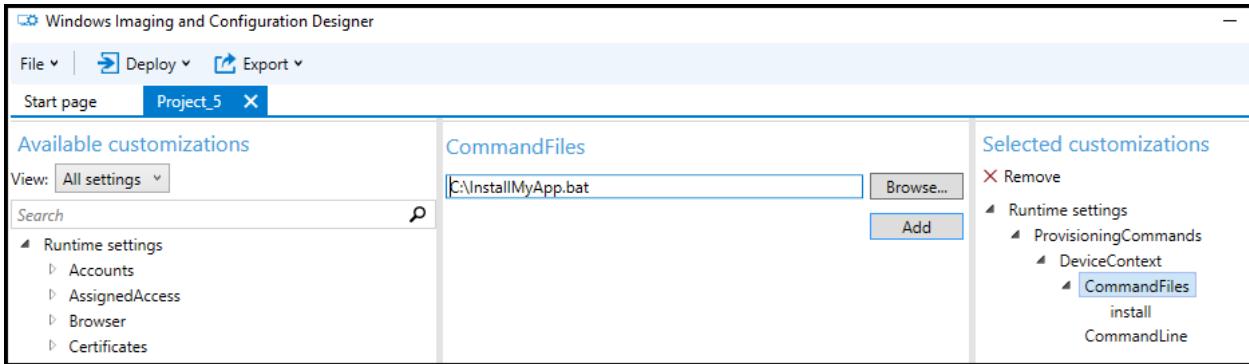
In Windows Configuration Designer, this looks like:



You also need to add the relevant assets for that command line including the orchestrator script and any other assets it references such as installers or .cab files.

In Windows Configuration Designer, that is done by adding files under the

`ProvisioningCommands/DeviceContext/CommandFiles` setting.



When you are done, [build the package](#).

## Remarks

- No user interaction or console output is supported via ProvisioningCommands. All work needs to be silent. If your script attempts to do any of the following it will cause undefined behavior, and could put the device in an unrecoverable state if executed during setup or the Out of Box Experience: a. Echo to console b. Display anything on the screen c. Prompt the user with a dialog or install wizard
  - When applied at first boot, provisioning runs early in the boot sequence and before a user context has been established; care must be taken to only include installers that can run at this time. Other installers can be provisioned via a management tool.
  - If the device is put into an unrecoverable state because of a bad script, you can reset it using [recovery options in Windows 10](#).
  - The CommandFile assets are deployed on the device to a temporary folder unique to each package.
    - For Windows 10, version 1607 and earlier: a. For packages added during the out of box experience, this is usually in  
`%WINDIR%\system32\config\systemprofile\appdata\local\Temp\ProvisioningPkgTmp\<{PackageIdGuid}>\Commands`
    - b. For packages added by double-clicking on an already deployed device, this will be in the temp folder for the user executing the PPKG: `%TMP%\ProvisioningPkgTmp\<{PackageIdGuid}>\Commands`
  - For Windows 10, version 1703: a. For packages added during the out of box experience, this is usually in  
`%WINDIR%\system32\config\systemprofile\appdata\local\Temp\ProvisioningPkgTmp\<{PackageIdGuid}>\Commands\0`
- The `\0` after `Commands\` refers to the installation order and indicates the first app to be installed. The number will increment for each app in the package. b. For packages added by double-clicking on an already deployed device, this will be in the temp folder for the user executing the provisioning package:  
`%TMP%\ProvisioningPkgTmp\<{PackageIdGuid}>\Commands\0`

5. The command line will be executed with the directory the CommandFiles were deployed to as the working directory. This means you do not need to specific the full path to assets in the command line or from within any script.
6. The runtime provisioning component will attempt to run the scripts from the provisioning package at the earliest point possible, depending on the stage when the PPKG was added. For example, if the package was added during the Out-of-Box Experience, it will be run immediately after the package is applied, while the out of box experience is still happening. This is before the user account configuration options are presented to the user. A spinning progress dialog will appear and "please wait" will be displayed on the screen.

**NOTE**

There is a timeout of 30 minutes for the provisioning process at this point. All scripts and installs need to complete within this time.

7. The scripts are executed in the background as the rest of provisioning continues to run. For packages added on existing systems using the double-click to install, there is no notification that provisioning or script execution has completed

## Related topics

- [Provisioning packages for Windows 10](#)
- [How provisioning works in Windows 10](#)
- [Install Windows Configuration Designer](#)
- [Create a provisioning package](#)
- [Apply a provisioning package](#)
- [Settings changed when you uninstall a provisioning package](#)
- [Provision PCs with common settings for initial deployment \(simple provisioning\)](#)
- [Windows Configuration Designer command-line interface \(reference\)](#)
- [PowerShell cmdlets for provisioning Windows 10 \(reference\)](#)
- [Create a provisioning package with multivariant settings](#)

# Create a provisioning package with multivariant settings

11/8/2017 • 8 min to read • [Edit Online](#)

## Applies to

- Windows 10
- Windows 10 Mobile

In your organization, you might have different configuration requirements for devices that you manage. You can create separate provisioning packages for each group of devices in your organization that have different requirements. Or, you can create a multivariant provisioning package, a single provisioning package that can work for multiple conditions. For example, in a single provisioning package, you can define one set of customization settings that will apply to devices set up for French and a different set of customization settings for devices set up for Japanese.

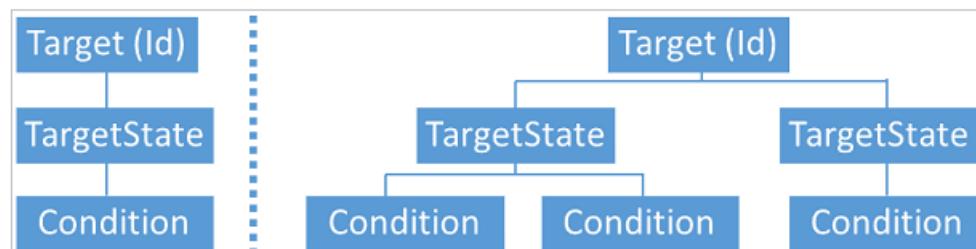
To provision multivariant settings, you use Windows Configuration Designer to create a provisioning package that contains all of the customization settings that you want to apply to any of your devices. Next, you manually edit the .XML file for that project to define each set of devices (a **Target**). For each **Target**, you specify at least one **Condition** with a value, which identifies the devices to receive the configuration. Finally, for each **Target**, you provide the customization settings to be applied to those devices.

Let's begin by learning how to define a **Target**.

## Define a target

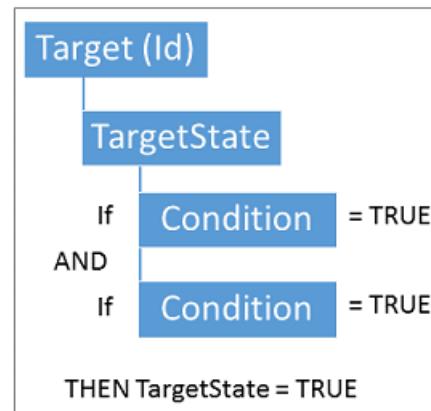
In the XML file, you provide an **Id**, or friendly name, for each **Target**. Each **Target** is defined by at least one **TargetState** which contains at least one **Condition**. A **Condition** element defines the matching type between the condition and the specified value.

A **Target** can have more than one **TargetState**, and a **TargetState** can have more than one **Condition**.

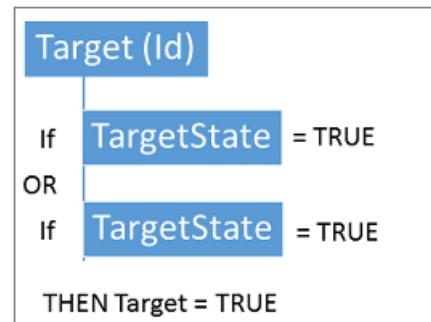


The following table describes the logic for the target definition.

When all **Condition** elements are TRUE, **TargetState** is TRUE.



If any of the **TargetState** elements is TRUE, **Target** is TRUE, and the **Id** can be used for setting customizations.



## Conditions

The following table shows the conditions supported in Windows 10 provisioning for a **TargetState**:

CONDITION NAME	CONDITION PRIORITY	WINDOWS 10 MOBILE	WINDOWS 10 FOR DESKTOP EDITIONS	VALUE TYPE	VALUE DESCRIPTION
MNC	P0	Supported	Supported	Digit string	Use to target settings based on the Mobile Network Code (MNC) value.
MCC	P0	Supported	Supported	Digit string	Use to target settings based on the Mobile Country Code (MCC) value.
SPN	P0	Supported	Supported	String	Use to target settings based on the Service Provider Name (SPN) value.
PNN	P0	Supported	Supported	String	Use to target settings based on public land mobile network (PLMN) Network Name value.

CONDITION NAME	CONDITION PRIORITY	WINDOWS 10 MOBILE	WINDOWS 10 FOR DESKTOP EDITIONS	VALUE TYPE	VALUE DESCRIPTION
GID1	P0	Supported	Supported	Digit string	Use to target settings based on the Group Identifier (level 1) value.
ICCID	P0	Supported	Supported	Digit string	Use to target settings based on the Integrated Circuit Card Identifier (ICCID) value.
Roaming	P0	Supported	N/A	Boolean	Use to specify roaming. Set the value to <b>1</b> (roaming) or <b>0</b> (non-roaming).
UICC	P0	Supported	N/A	Enumeration	Use to specify the Universal Integrated Circuit Card (UICC) state. Set the value to one of the following: - 0 - Empty - 1 - Ready - 2 - Locked
UICCSLOT	P0	Supported	N/A	Digit string	Use to specify the UICC slot. Set the value one of the following: - 0 - Slot 0 - 1 - Slot 1
ProcessorType	P1	Supported	Supported	String	Use to target settings based on the processor type.
ProcessorName	P1	Supported	Supported	String	Use to target settings based on the processor name.
AoAc ("Always On, Always Connected")	P1	Supported	Supported	Boolean	Set the value to <b>0</b> (false) or <b>1</b> (true). If this condition is TRUE, the system supports the S0 low power idle model.

Condition Name	Condition Priority	Windows 10 Mobile	Windows 10 for Desktop Editions	Value Type	Value Description
PowerPlatformRole	P1	Supported	Supported	Enumeration	Indicates the preferred power management profile. Set the value based on the <a href="#">POWER_PLATFORM_ROLE enumeration</a> .
Architecture	P1	Supported	Supported	String	Matches the PROCESSOR_ARCHITECTURE environment variable.
Server	P1	Supported	Supported	Boolean	Set the value to <b>0</b> (false) or <b>1</b> (true) to identify a server.
Region	P1	Supported	Supported	Enumeration	Use to target settings based on country/region, using the 2-digit alpha ISO code per <a href="#">ISO 3166-1 alpha-2</a> .
Lang	P1	Supported	Supported	Enumeration	Use to target settings based on language code, using the 2-digit <a href="#">ISO 639 alpha-2 code</a> .

The matching types supported in Windows 10 are:

Matching Type	Syntax	Example
Straight match	Matching type is specified as-is	<Condition Name="ProcessorName" Value="Barton" />
Regular expression (Regex) match	Matching type is prefixed by "Pattern:"	<Condition Name="ProcessorName" Value="Pattern:.Celeron." />
Numeric range match	Matching type is prefixed by "!Range:"	<Condition Name="MNC" Value="!Range:400, 550" />

### TargetState priorities

You can define more than one **TargetState** within a provisioning package to apply settings to devices that match device conditions. When the provisioning engine evaluates each **TargetState**, more than one **TargetState** may fit current device conditions. To determine the order in which the settings are applied, the system assigns a priority to every **TargetState**.

A setting that matches a **TargetState** with a lower priority is applied before the setting that matches a **TargetState** with a higher priority. This means that a setting for the **TargetState** with the higher priority can overwrite a setting for the **TargetState** with the lower priority.

Settings that match more than one **TargetState** with equal priority are applied according to the order that each **TargetState** is defined in the provisioning package.

The **TargetState** priority is assigned based on the condition's priority (see the [Conditions table](#) for priorities). The priority evaluation rules are as followed:

1. A **TargetState** with P0 conditions is higher than a **TargetState** without P0 conditions.
2. A **TargetState** with both P0 and P1 conditions is higher than a **TargetState** with only P0 conditions.
3. A **TargetState** with a greater number of matched P0 conditions is higher than **TargetState** with fewer matched P0 conditions, regardless of the number of P1 conditions matched.
4. If the number of P0 conditions matched are equivalent, then the **TargetState** with the most matched P1 conditions has higher priority.
5. If both P0 and P1 conditions are equally matched, then the **TargetState** with the greatest total number of matched conditions has highest priority.

## Create a provisioning package with multivariant settings

Follow these steps to create a provisioning package with multivariant capabilities.

1. Build a provisioning package and configure the customizations you want to apply during certain conditions. For more information, see [Create a provisioning package](#).
2. After you've [configured the settings](#), save the project.
3. Open the project folder and copy the customizations.xml file to any local location.
4. Use an XML or text editor to open the customizations.xml file.

The customizations.xml file holds the package metadata (including the package owner and rank) and the settings that you configured when you created your provisioning package. The **Customizations** node of the file contains a **Common** section, which contains the customization settings.

The following example shows the contents of a sample customizations.xml file.

```
<?xml version="1.0" encoding="utf-8"?>
<WindowsCustomizations>
<PackageConfig xmlns="urn:schemas-Microsoft-com:Windows-ICD-Package-Config.v1.0">
<ID>{6aaa4dfa-00d7-4aaa-8adf-73c6a7e2501e}</ID>
<Name>My Provisioning Package</Name>
<Version>1.0</Version>
<OwnerType>OEM</OwnerType>
<Rank>50</Rank>
</PackageConfig>
<Settings xmlns="urn:schemas-microsoft-com:windows-provisioning">
<Customizations>
<Common>
<Policies>
<AllowBrowser>0</AllowBrowser>
<AllowCamera>0</AllowCamera>
<AllowBluetooth>0</AllowBluetooth>
</Policies>
<HotSpot>
<Enabled>0</Enabled>
</HotSpot>
</Common>
</Customizations>
</Settings>
</WindowsCustomizations>
```

5. Edit the customizations.xml file to create a **Targets** section to describe the conditions that will handle your multivariant settings.

The following example shows the customizations.xml, which has been modified to include several conditions including **ProcessorName**, **ProcessorType**, **MCC**, and **MNC**.

```

<?xml version="1.0" encoding="utf-8"?>
<WindowsCustomizations>
<PackageConfig xmlns="urn:schemas-Microsoft-com:Windows-ICD-Package-Config.v1.0">
<ID>{6aaa4dfa-00d7-4aaa-8adf-73c6a7e2501e}</ID>
<Name>My Provisioning Package</Name>
<Version>1.0</Version>
<OwnerType>OEM</OwnerType>
<Rank>50</Rank>
</PackageConfig>
<Settings xmlns="urn:schemas-microsoft-com:windows-provisioning">
<Customizations>
<Common>
<Policies>
<AllowBrowser>0</AllowBrowser>
<AllowCamera>0</AllowCamera>
<AllowBluetooth>0</AllowBluetooth>
</Policies>
<HotSpot>
<Enabled>0</Enabled>
</HotSpot>
</Common>
<Targets>
<Target Id="Unique target identifier for desktop">
<TargetState>
<Condition Name="ProcessorName" Value="Pattern:. *Celeron.*" />
<Condition Name="ProcessorType" Value="Pattern:. *(I|i)ntel.*" />
</TargetState>
<TargetState>
<Condition Name="ProcessorName" Value="Barton" />
<Condition Name="ProcessorType" Value="Athlon MP" />
</TargetState>
</Target>
<Target Id="Mobile target">
<TargetState>
<Condition Name="MCC" Value="Range:310, 320" />
<Condition Name="MNC" Value="!Range:400, 550" />
</TargetState>
</Target>
</Targets>
</Customizations>
</Settings>
</WindowsCustomizations>

```

6. In the customizations.xml file, create a **Variant** section for the settings you need to customize. To do this:

- Define a child **TargetRefs** element.
- Within the **TargetRefs** element, define a **TargetRef** element. You can define multiple **TargetRef** elements for each **Id** that you need to apply to customized settings.
- Move compliant settings from the **Common** section to the **Variant** section.

If any of the **TargetRef** elements matches the **Target**, all settings in the **Variant** are applied.

#### **NOTE**

You can define multiple **Variant** sections. Settings that reside in the **Common** section are applied unconditionally on every triggering event.

The following example shows the customizations.xml updated to include a **Variant** section and the moved settings that will be applied if the conditions for the variant are met.

```

<?xml version="1.0" encoding="utf-8"?>
<WindowsCustomizations>
<PackageConfig xmlns="urn:schemas-Microsoft-com:Windows-ICD-Package-Config.v1.0">
<ID>{6aaa4dfa-00d7-4aaa-8adf-73c6a7e2501e}</ID>
<Name>My Provisioning Package</Name>
<Version>1.0</Version>
<OwnerType>OEM</OwnerType>
<Rank>50</Rank>
</PackageConfig>
<Settings xmlns="urn:schemas-microsoft-com:windows-provisioning">
<Customizations>
<Common>
</Common>
<Targets>
<Target Id="Unique target identifier for desktop">
<TargetState>
<Condition Name="ProcessorName" Value="Pattern:. *Celeron.*" />
<Condition Name="ProcessorType" Value="Pattern:. *(I|i)ntel.*" />
</TargetState>
<TargetState>
<Condition Name="ProcessorName" Value="Barton" />
<Condition Name="ProcessorType" Value="Athlon MP" />
</TargetState>
</Target>
<Target Id="Mobile target">
<TargetState>
<Condition Name="MCC" Value="Range:310, 320" />
<Condition Name="MNC" Value="!Range:400, 550" />
</TargetState>
</Target>
</Targets>
<Variant>
<TargetRefs>
<TargetRef Id="Unique target identifier for desktop" />
<TargetRef Id="Mobile target" />
</TargetRefs>
<Settings>
<Policies>
<AllowBrowser>1</AllowBrowser>
<AllowCamera>1</AllowCamera>
<AllowBluetooth>1</AllowBluetooth>
</Policies>
<HotSpot>
<Enabled>1</Enabled>
</HotSpot>
</Settings>
</Variant>
</Customizations>
</Settings>
</WindowsCustomizations>

```

7. Save the updated customizations.xml file and note the path to this updated file. You will need the path as one of the values for the next step.
8. Use the [Windows Configuration Designer command-line interface](#) to create a provisioning package using the updated customizations.xml.

For example:

```

icd.exe /Build-ProvisioningPackage /CustomizationXML:"C:\CustomProject\customizations.xml"
/PackagePath:"C:\CustomProject\output.ppkg" /StoreFile:C:\Program Files (x86)\Windows
Kits\10\Assessment and Deployment Kit\Imaging and Configuration Designer\x86\Microsoft-Common-
Provisioning.dat"

```

In this example, the **StoreFile** corresponds to the location of the settings store that will be used to create the package for the required Windows edition.

**NOTE**

The provisioning package created during this step will contain the multivariant settings. You can use this package either as a standalone package that you can apply to a Windows device or use it as the base when starting another project.

## Events that trigger provisioning

When you install the multivariant provisioning package on a Windows 10 device, the provisioning engine applies the matching condition settings at every event and triggers provisioning.

The following events trigger provisioning on Windows 10 devices:

EVENT	WINDOWS 10 MOBILE	WINDOWS 10 FOR DESKTOP EDITIONS
System boot	Supported	Supported
Operating system update	Supported	Planned
Package installation during device first run experience	Supported	Supported
Detection of SIM presence or update	Supported	Supported
Package installation at runtime	Supported	Supported
Roaming detected	Supported	Not supported

## Related topics

- [Provisioning packages for Windows 10](#)
- [How provisioning works in Windows 10](#)
- [Install Windows Configuration Designer](#)
- [Create a provisioning package](#)
- [Apply a provisioning package](#)
- [Settings changed when you uninstall a provisioning package](#)
- [Provision PCs with common settings for initial deployment \(simple provisioning\)](#)
- [Use a script to install a desktop app in provisioning packages](#)
- [PowerShell cmdlets for provisioning Windows 10 \(reference\)](#)
- [Windows Configuration Designer command-line interface \(reference\)](#)

# PowerShell cmdlets for provisioning Windows 10 (reference)

7/28/2017 • 1 min to read • [Edit Online](#)

## Applies to

- Windows 10
- Windows 10 Mobile

Windows 10, version 1703, ships with Windows Provisioning PowerShell cmdlets. These cmdlets make it easy to script the following functions.

CMDLET	USE THIS CMDLET TO	SYNTAX
Add-ProvisioningPackage	Apply a provisioning package	<pre>Add-ProvisioningPackage [-Path &lt;string&gt; [-ForceInstall] [-LogsFolder &lt;string&gt;] [-WprpFile &lt;string&gt;] [&lt;CommonParameters&gt;]]</pre>
Remove-ProvisioningPackage	Remove a provisioning package	<pre>Remove-ProvisioningPackage -PackageName &lt;string&gt; [-LogsFolder &lt;string&gt;] [-WprpFile &lt;string&gt;] [&lt;CommonParameters&gt;]</pre> <pre>Remove-ProvisioningPackage -Path &lt;string&gt; [-LogsFolder &lt;string&gt;] [-WprpFile &lt;string&gt;] [&lt;CommonParameters&gt;]</pre> <pre>Remove-ProvisioningPackage -AllInstalledPackages [-LogsFolder &lt;string&gt;] [-WprpFile &lt;string&gt;] [&lt;CommonParameters&gt;]</pre>
Get-ProvisioningPackage	Get information about an installed provisioning package	<pre>Get-ProvisioningPackage -PackageName &lt;string&gt; [-LogsFolder &lt;string&gt;] [-WprpFile &lt;string&gt;] [&lt;CommonParameters&gt;]</pre> <pre>Get-ProvisioningPackage -Path &lt;string&gt; [-LogsFolder &lt;string&gt;] [-WprpFile &lt;string&gt;] [&lt;CommonParameters&gt;]</pre> <pre>Get-ProvisioningPackage -AllInstalledPackages [-LogsFolder &lt;string&gt;] [-WprpFile &lt;string&gt;] [&lt;CommonParameters&gt;]</pre>
Export-ProvisioningPackage	Extract the contents of a provisioning package	<pre>Export-ProvisioningPackage -PackageName &lt;string&gt; -OutputFolder &lt;string&gt; [-Overwrite] [-AnswerFileOnly] [-LogsFolder &lt;string&gt;] [-WprpFile &lt;string&gt;] [&lt;CommonParameters&gt;]</pre>

		<pre>Export-ProvisioningPackage -Path &lt;string&gt; -OutputFolder &lt;string&gt; [-Overwrite] [-AnswerFileOnly] [-LogsFolder &lt;string&gt;] [-WpprFile &lt;string&gt;] [&lt;CommonParameters&gt;]</pre>
Install-TrustedProvisioningCertificate	Adds a certificate to the Trusted Certificate store	<pre>Install-TrustedProvisioningCertificate &lt;path to local certificate file on disk&gt;</pre>
Get-TrustedProvisioningCertificate	List all installed trusted provisioning certificates; use this cmdlet to get the certificate thumbprint to use with the <b>Uninstall-TrustedProvisioningCertificate</b> cmdlet	<pre>Get-TrustedProvisioningCertificate</pre>
Uninstall-TrustedProvisioningCertificate	Remove a previously installed provisioning certificate	<pre>Uninstall-TrustedProvisioningCertificate &lt;thumbprint&gt;</pre>

#### NOTE

You can use Get-Help to get usage help on any command. For example: `Get-Help Add-ProvisioningPackage`

Trace logs are captured when using cmdlets. The following logs are available in the logs folder after the cmdlet completes:

- ProvTrace.<timestamp>.ETL - ETL trace file, unfiltered
- ProvTrace.<timestamp>.XML - ETL trace file converted into raw trace events, unfiltered
- ProvTrace.<timestamp>.TXT - TEXT file containing trace output formatted for easy reading, filtered to only show events logged by providers in the WPPR file
- ProvLogReport.<timestamp>.XLS - Excel file containing trace output, filtered to only show events logged by providers in WPPR file

#### NOTE

When applying provisioning packages using Powershell cmdlets, the default behavior is to suppress the prompt that appears when applying an unsigned provisioning package. This is by design so that provisioning packages can be applied as part of existing scripts.

## Related topics

- [How provisioning works in Windows 10](#)
- [Install Windows Configuration Designer](#)
- [Create a provisioning package](#)
- [Apply a provisioning package](#)
- [Settings changed when you uninstall a provisioning package](#)
- [Provision PCs with common settings for initial deployment \(simple provisioning\)](#)
- [Use a script to install a desktop app in provisioning packages](#)
- [Windows Configuration Designer command-line interface \(reference\)](#)
- [Create a provisioning package with multivariant settings](#)

# Windows Configuration Designer command-line interface (reference)

7/28/2017 • 2 min to read • [Edit Online](#)

## Applies to

- Windows 10
- Windows 10 Mobile

You can use the Windows Configuration Designer command-line interface (CLI) to automate the building of provisioning packages.

- IT pros can use the Windows Configuration Designer CLI to require less re-tooling of existing processes. You must run the Windows Configuration Designer CLI from a command window with administrator privileges.
- You must use the Windows Configuration Designer CLI and edit the customizations.xml sources to create a provisioning package with multivariant support. You need the customizations.xml file as one of the inputs to the Windows Configuration Designer CLI to build a provisioning package. For more information, see [Create a provisioning package with multivariant settings](#).

## Syntax

```
icd.exe /Build-ProvisioningPackage /CustomizationXML:<path_to_xml> /PackagePath:<path_to_ppkg>
[/StoreFile:<path_to_storefile>] [/MSPackageRoot:<path_to_mspackage_directory>] [/OEMInputXML:
<path_to_xml>]
[/ProductName:<product_name>] [/Variables:<name>:<value>] [[+|-]Encrypted] [[+|-]Overwrite] [/?]
```

## Switches and arguments

SWITCH	REQUIRED?	ARGUMENTS
/CustomizationXML	No	Specifies the path to a Windows provisioning XML file that contains the customization assets and settings. For more information, see Windows provisioning answer file.
/PackagePath	Yes	Specifies the path and the package name where the built provisioning package will be saved.

SWITCH	REQUIRED?	ARGUMENTS
/StoreFile	No See Important note.	For partners using a settings store other than the default store(s) used by Windows Configuration Designer, use this parameter to specify the path to one or more comma-separated Windows settings store file. By default, if you don't specify a settings store file, the settings store that's common to all Windows editions will be loaded by Windows Configuration Designer. <b>Important</b> If you use this parameter, you must not use /MSPackageRoot or /OEMInputXML.
/Variables	No	Specifies a semicolon separated and macro pair. The format for the argument must be =.
Encrypted	No	Denotes whether the provisioning package should be built with encryption. Windows Configuration Designer auto-generates the decryption password and includes this information in the output. Precede with + for encryption or - for no encryption. The default is no encryption.
Overwrite	No	Denotes whether to overwrite an existing provisioning package. Precede with + to overwrite an existing package or - if you don't want to overwrite an existing package. The default is false (don't overwrite).
/?	No	Lists the switches and their descriptions for the command-line tool or for certain commands.

## Related topics

- [Provisioning packages for Windows 10](#)
- [How provisioning works in Windows 10](#)
- [Install Windows Configuration Designer](#)
- [Create a provisioning package](#)
- [Apply a provisioning package](#)
- [Settings changed when you uninstall a provisioning package](#)
- [Provision PCs with common settings for initial deployment \(simple provisioning\)](#)
- [Use a script to install a desktop app in provisioning packages](#)
- [PowerShell cmdlets for provisioning Windows 10 \(reference\)](#)
- [Create a provisioning package with multivariant settings](#)

# Windows Configuration Designer provisioning settings (reference)

10/17/2017 • 1 min to read • [Edit Online](#)

This section describes the settings that you can configure in [provisioning packages](#) for Windows 10 using Windows Configuration Designer.

## Edition that each group of settings applies to

SETTING GROUP	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
Accounts	X	X	X	X	X
ADMXIngestion	X				
ApplicationManagement					X
AssignedAccess	X			X	
AutomaticTime		X			
Browser	X	X	X	X	
CallAndMessagingEnhancement		X			
Calling		X			
CellCore	X	X			
Cellular	X				
Certificates	X	X	X	X	X
CleanPC	X				
Connections	X	X	X	X	
ConnectivityProfiles	X	X	X	X	X
CountryAndRegion	X	X	X	X	
DesktopBackgroundAndColors	X				

SETTING GROUP	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
DeveloperSetup				X	
DeviceFormFactor	X	X	X	X	
DeviceInfo		X			
DeviceManagement	X	X	X	X	
DMClient	X	X	X	X	X
EditionUpgrade	X	X	X	X	
EmbeddedLockdownProfiles		X			
FirewallConfiguration					X
FirstExperience				X	
Folders	X	X	X	X	
HotSpot	X	X	X	X	X
InitialSetup		X			
InternetExplorer		X			
Licensing	X				
Maps	X	X	X	X	
Messaging		X			
ModemConfigurations		X			
Multivariant		X			
NetworkProxy			X		
NetworkQOSPolicy			X		
NFC		X			
OOBE	X	X			
OtherAssets		X			

SETTING GROUP	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
Personalization	X				
Policies	X	X	X	X	X
ProvisioningCommands	X				
SharedPC	X				
Shell		X			
SMISettings	X				
Start	X	X			
StartupApp					X
StartupBackgroundTasks					X
SurfaceHubManagement			X		
TabletMode	X	X	X	X	
TakeATest	X				
TextInput		X			
Theme		X			
UnifiedWriteFilter	X				
UniversalAppInstall	X	X	X	X	X
UniversalAppUninstall	X	X	X	X	X
WeakCharger	X	X	X	X	
WindowsTeamSettings			X		
WLAN				X	
Workplace	X	X	X	X	X

# Accounts (Windows Configuration Designer reference)

9/6/2017 • 2 min to read • [Edit Online](#)

Use these settings to join a device to an Active Directory domain or an Azure Active Directory tenant, or to add local user accounts to the device.

## Applies to

SETTING GROUPS	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
Azure	X	X	X		
ComputerAccount	X		X		X
Users	X		X	X	

## Azure

The **Azure > Authority** and **Azure > BPRT** settings for bulk Azure Active Directory (Azure AD) enrollment can only be configured using one of the provisioning wizards. After you get a bulk token for Azure AD enrollment in a wizard, you can switch to the advanced editor to configure additional provisioning settings. For information about using the wizards, see:

- [Instructions for desktop wizard](#)
- [Instructions for the mobile wizard](#)
- [Instructions for the kiosk wizard](#)

## ComputerAccount

Specifies the settings you can configure when joining a device to a domain, including the computer name and the account to use for joining the computer to the domain.

### NOTE

If you want to create a provisioning package that joins a device to Active Directory AND sets `HideOobe`, and you want to apply that package during OOBE, we also recommend setting the `ComputerName` and creating a local admin account in the provisioning package.

SETTING	VALUE	DESCRIPTION
Account	string	Account to use to join computer to domain
AccountOU	string	Name of organizational unit for the computer account

Setting	Value	Description
ComputerName	<p>Specify a unique name for the domain-joined computers using %RAND:x%, where x is an integer less than 15 digits long, or using %SERIALNUMBER% characters in the name.</p> <p>ComputerName is a string with a maximum length of 15 bytes of content:</p> <ul style="list-style-type: none"> <li>- ComputerName can use ASCII characters (1 byte each) and/or multi-byte characters such as Kanji, so long as you do not exceed 15 bytes of content.</li> <li>- ComputerName cannot use spaces or any of the following characters: {} ~ [ \ ] ^ ' ; &lt; = &gt; ? @ ! " # \$ % ` ( ) + / . , * &amp;, or contain any spaces.</li> <li>- ComputerName cannot use some non-standard characters, such as emoji.</li> </ul> <p>Computer names that cannot be validated through the DnsValidateName function cannot be used, for example, computer names that only contain numbers (0-9). For more information, see the <a href="#">DnsValidateName function</a>.</p>	Specifies the name of the Windows device (computer name on PCs)
DomainName	string (cannot be empty)	Specify the name of the domain that the device will join
Password	string (cannot be empty)	Corresponds to the password of the user account that's authorized to join the computer account to the domain.

## Users

Use these settings to add local user accounts to the device.

Setting	Value	Description
UserName	string (cannot be empty)	Specify a name for the local user account
HomeDir	string (cannot be empty)	Specify the path of the home directory for the user
Password	string (cannot be empty)	Specify the password for the user account
UserGroup	string (cannot be empty)	Specify the local user group for the user

# ADMXIngestion (Windows Configuration Designer reference)

9/6/2017 • 2 min to read • [Edit Online](#)

Starting in Windows 10, version 1703, you can import (*ingest*) select Group Policy administrative templates (ADMX files) and configure values for ADMX-backed policies in a provisioning package. To see which types of ADMX-backed policies can be applied, see [Win32 and Desktop Bridge app policy configuration overview](#).

- The settings under [ConfigADMXInstalledPolicy](#) allow you to set values for policies in the imported ADMX file.
- The settings under [ConfigOperations](#) specify the ADMX file to be imported.

## IMPORTANT

Only per-device policies can be set using a provisioning package.

## Applies to

SETTING GROUPS	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
<a href="#">ConfigADMXInstalledPolicy</a>	X				
<a href="#">ConfigOperations</a>	X				

## ConfigADMXInstalledPolicy

### IMPORTANT

Configure the settings to import the ADMX file in [ConfigOperations](#) first.

In **ConfigADMXInstalledPolicy**, you provide a policy setting and value for that policy from the imported ADMX. You will need information from the ADMX that you import in **ConfigOperations** to complete **ConfigADMXInstalledPolicy**.

1. Enter an area name, and then click **Add**. The structure of the area name is the following:

AppName (from ConfigOperations) ~ SettingType ~ category name from ADMX

See [Category and policy in ADMX](#) for more information. A setting may have multiple levels of category names, as in the following example.

Example: Office16~Policy~L\_MicrosoftOfficeMachine~L\_Updates

2. Select the area name in the Customization pane, enter a policy name from the ADMX, and then click **Add**.

For example, L\_HideEnableDisableUpdates .

3. Select the policy name in the Customization pane, and then enter a value from the ADMX in the text field. For example, <disabled/> .

# ConfigOperations

Use **ConfigOperations** to import an ADMX file or policies from an ADMX file.

1. Enter an app name, and then click **Add**.

This can be any name you assign, so choose something descriptive to help you identify its purpose. For example, if you are importing ADMX for Office 16, enter an app name of **Office 16**.

2. Select the app name in the Customizations pane, select a setting type, and then click **Add**.

The choices, **Policy** and **Preference**, have no impact on the behavior of the settings, and are only provided for your convenience should you want to categorize the settings you add.

3. Select the setting type in the Customizations pane. In the **AdmxFileUid** field, enter the name of the ADMX file or a unique ID for the file, and then click **Add**.

The **AdmxFileUid** can be any string, but must be unique in the provisioning package. Using the name of the ADMX file will help you identify the file in the future.

4. Select the AdmxFileUid in the Customizations pane, and paste the contents of the ADMX file in the text field. Before copying the contents of the ADMX file, you must convert it to a single-line. See [Convert multi-line to single line](#) for instructions.

## NOTE

When you have a large ADMX file, you may want to only include specific settings. Instead of pasting in the entire ADMX file, you can paste just one or more specific policies (after converting them to single-line).

5. Repeat for each ADMX, or set of ADMX policies, that you want to add, and then configure [ConfigADMXInstalledPolicy](#) for each one.

## Convert multi-line to single line

Use the following PowerShell cmdlet to remove carriage returns and line feeds from a multi-line file to create a single-line file that you can paste in **AdmxFileUid**.

```
$path="file path"
Get-Content $path -Raw).Replace("'r'n","");
| Set-Content $path -Force
```

## Category and policy in ADMX

The following images show snippets of the ADMX file for Office 16 that are used in the examples in the procedures above. The first image highlights the category names.

```
<categories>
...
<category name="L_MicrosoftOfficeMachine"
displayName="$(string.L_MicrosoftOfficeMachine)" />
<category name="L_Updates" displayName="$(string.L_Updates)">
 <parentCategory ref="L_MicrosoftOfficeMachine" />
</category>
...
</categories>
....
```

The next image highlights the specific policy.

```
| <policy name="L_HideEnableDisableUpdates" class="Machine"
| displayName="$(string.L_HideEnableDisableUpdates)"
| explainText="$(string.L_HideEnableDisableUpdatesExplain)"
| key="software\policies\microsoft\office\16.0\common\officeupdate"
| valueName="hideenabledisableupdates">
| <parentCategory ref="L_Updates" />
| <supportedOn ref="windows:SUPPORTED_Windows7" />
| <enabledValue>
| <decimal value="1" />
| </enabledValue>
| <disabledValue>
| <decimal value="0" />
| </disabledValue>
</policy>
```

## Related topics

- [Policy configuration service provider \(CSP\): ADMX-backed policies](#)
- [Understanding ADMX-backed policies](#)

# ApplicationManagement (Windows Configuration Designer reference)

10/17/2017 • 1 min to read • [Edit Online](#)

Use these settings to manage app installation and management.

## NOTE

ApplicationManagement settings are not available in Windows 10, version 1709.

## Applies to

SETTINGS	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
AllowAllTrustedApps					X
AllowAppStoreAutoUpdate					X
RestrictAppDataToSystemVolume					X
RestrictAppToSystemVolume					X

## AllowAllTrustedApps

Specifies whether non-Microsoft Store apps are allowed.

VALUE	DESCRIPTION
No	Only Microsoft Store apps are allowed
Yes	Non-Microsoft Store apps are allowed

## AllowAppStoreAutoUpdate

Specifies whether automatic update of apps from Microsoft Store are allowed

VALUE	DESCRIPTION
Disallowed	Automatic update of apps is not allowed
Allowed	Automatic update of apps is allowed

## RestrictAppDataToSystemVolume

Specifies whether application data is restricted to the system drive.

VALUE	DESCRIPTION
0	Not restricted
1	Restricted

## RestrictAppToSystemVolume

Specifies whether the installation of applications is restricted to the system drive.

VALUE	DESCRIPTION
0	Not restricted
1	Restricted

## Related topics

- [Policy configuration service provider \(CSP\): ApplicationManagement/AllowAllTrustedApps](#)
- [Policy CSP: ApplicationManagement/AllowAppStoreAutoUpdate](#)
- [Policy CSP: ApplicationManagement/RestrictAppDataToSystemVolume](#)
- [Policy CSP: ApplicationManagement/RestrictAppToSystemVolume](#)

# AssignedAccess (Windows Configuration Designer reference)

10/17/2017 • 1 min to read • [Edit Online](#)

Use this setting to configure single use (kiosk) devices.

## Applies to

Setting	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
<a href="#">AssignedAccessSettings</a>	X			X	
<a href="#">MultiAppAssignedAccessSettings</a>	X				

## AssignedAccessSettings

Enter the account and the application you want to use for Assigned access, using [the AUMID](#). When that user account signs in on the device, only the specified app will run.

### Example:

```
"Account": "domain\user", "AUMID": "Microsoft.WindowsCalculator_8wekyb3d8bbwe!App"
```

## MultiAppAssignedAccessSettings

### NOTE

MultiAppAssignedAccessSettings is supported on Windows 10, version 1709 only.

Use this setting to configure a kiosk device that runs more than one app.

1. [Create an assigned access configuration XML file for multiple apps](#).
2. In Windows Configuration Designer, select **MultiAppAssignedAccessSettings**.
3. Browse to and select the assigned access configuration XML file.

## Related topics

- [AssignedAccess configuration service provider \(CSP\)](#)

# AutomaticTime (Windows Configuration Designer reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Use these settings to configure automatic time updates.

## Applies to

SETTINGS	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
NTPRegularSyncInterval		X			
NTPRetryInterval		X			
NTPServer		X			

## NTPRegularSyncInterval

Set the regular sync interval for phones that are set to use Network Time Protocol (NTP) time servers. Select a value between  and  hours, inclusive. The default sync interval is  hours.

## NTPRetryInterval

Set the retry interval if the regular sync fails. Select a value between  and  hours, inclusive.

## NTPServer

Change the default NTP server for phones that are set to use NTP. To enumerate the NTP source server(s) used by the NTP client, set the value for NTPServer to a list of server names, delimited by semi-colons.

### Example:

```
ntpserver1.contoso.com;ntpserver2.fabrikam.com;ntpserver3.contoso.com
```

The list should contain one or more server names. The default NTP source server value is .

# Browser (Windows Configuration Designer reference)

9/6/2017 • 2 min to read • [Edit Online](#)

Use to configure browser settings that should only be set by OEMs who are part of the Partner Search Code program.

## Applies to

Setting Groups	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
Favorites		X			
PartnerSearchCode	X	X	X	X	
SearchProviders		X			

## Favorites

Use to configure the default list of Favorites that show up in the browser.

To add a new item under the browser's **Favorites** list:

1. In the **Name** field, enter a friendly name for the item, and then click **Add**.
2. In the **Available customizations** pane, select the friendly name that you just created, and in the text field, enter the URL for the item.

For example, to include the corporate Web site to the list of browser favorites, a company called Contoso can specify **Contoso** as the value for the name and "<http://www.contoso.com>" for the URL.

## PartnerSearchCode

### IMPORTANT

This setting should only be set by OEMs who are part of the Partner Search Code program.

Set the value to a character string that corresponds to the OEM's Partner Search Code. This identification code must match the one assigned to you by Microsoft.

OEMs who are part of the program only have one PartnerSearchCode and this should be used for all Windows 10 for desktop editions images.

## SearchProviders

Contains the settings you can use to configure the default and additional search providers.

Microsoft Bing is the default search provider for Windows 10 Mobile. The default search provider must be set to Bing, except for devices shipping to certain countries where a different default search provider is required as specified in the [Specific region guidance](#) section of [Default](#).

## Default

Use **Default** to specify a name that matches one of the search providers you enter in [SearchProviderList](#). If you don't specify a default search provider, this will default to Microsoft Bing.

### Specific region guidance

Some countries require specific, default search providers. The following table lists the applicable countries and information for configuring the necessary search provider.

#### NOTE

For Russia + Commonwealth of Independent States (CIS), the independent states consist of Russia, Ukraine, Georgia, The Republic of Azerbaijan, Republic Of Belarus, The Republic of Kazakhstan, The Kyrgyz Republic, The Republic of Moldova, The Republic of Tajikistan, The Republic of Armenia, Turkmenistan, The Republic of Uzbekistan, and Turkey.

## SearchProviderList

Use to specify a list of additional search providers.

1. In the **Name** field, enter a name for the item, and then click **Add**.
2. In the **Available customizations** pane, select the name that you just created, and in the text field, enter the URL for the additional search provider.

For example, to specify Yandex in Russia and Commonwealth of Independent States (CIS), set the value of URL to "https://yandex.ru/search/touch/?text={searchTerm}&clid=2234144".

When configured with multiple search providers, the browser can display up to ten search providers.

#### IMPORTANT

Microsoft Bing is the default search provider for Windows 10 Mobile. The default search provider must be set to Bing, except for devices shipping to certain countries where a different default search provider is required as specified in the [Specific region guidance](#) section of [Default](#).

# CallAndMessagingEnhancement (Windows Configuration Designer reference)

10/17/2017 • 1 min to read • [Edit Online](#)

Use to configure call origin and blocking apps.

## IMPORTANT

These settings are intended to be used only by manufacturers, mobile operators, and solution providers when configuring devices, and are not intended for use by administrators in the enterprise.

## Applies to

SETTING GROUPS	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
BlockingApp		X			
CallOriginApp		X			

## BlockingApp

SETTING	VALUE	DESCRIPTION
ActiveBlockingAppUserModelId	AUMID	The AUMID of the application that will be set as the active blocking app by default.
DefaultBlockingAppUserModelId	AUMID	The AUMID of the application that the OS will select as the active blocking app if the user uninstalls the current active blocking app. This app should be uninstallable.

## CallOriginApp

SETTING	VALUE	DESCRIPTION
ActiveCallOriginAppUserModelId	AUMID	The AUMID of the application to be set as the active call origin provider app by default.
DefaultCallOriginAppUserModelId	AUMID	The AUMID of the application that the OS will select as the active call origin provider app if the user uninstalls the current active call origin app. This app should be uninstallable.

# Calling (Windows Configuration Designer reference)

10/17/2017 • 5 min to read • [Edit Online](#)

Use to configure settings for Calling.

## IMPORTANT

These settings are intended to be used only by manufacturers, mobile operators, and solution providers when configuring devices, and are not intended for use by administrators in the enterprise.

## Applies to

SETTING GROUPS	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
All settings		X			

## Branding

See [Branding for phone calls](#).

## PartnerAppSupport

See [Dialer codes to launch diagnostic applications](#).

## PerSimSettings

Use to configure settings for each subscriber identification module (SIM) card. Enter the Integrated Circuit Card Identifier (ICCID) for the SIM card, click Add, and then configure the following settings.

### Critical

SETTING	DESCRIPTION
MOSimFallbackVoicemailNumber	Partners who do not have the voicemail numbers on the device SIM can configure the voicemail number for their devices. If the voicemail number is not on the SIM and the registry key is not set, the default voicemail will not be set and the user will need to set the number. Set MOSimFallbackVoicemailNumber to the voicemail number that you want to use for the phone.
SimOverrideVoicemailNumber	Mobile operators can override the voicemail number on the UICC with a different voicemail number that is configured in the registry. Set SimOverrideVoicemailNumber to a string that contains the digits of the voicemail number to use instead of the voicemail number on the UICC.

### General

SETTING	DESCRIPTION
AllowVideoConferencing	Set as <b>True</b> to enable the ability to conference video calls.
DefaultCallerIdSetting	Configure the default setting for caller ID. Select between <b>No one</b> , <b>Only contacts</b> , <b>Every one</b> , and <b>Network default</b> . If set to <b>Network default</b> , set <b>ShowCallerIdNetworkDefaultSetting</b> to <b>True</b> .
DefaultEnableVideoCalling	Set as <b>True</b> to enable LTE video calling as the default setting.
IgnoreMWINotifications	Set as <b>True</b> to configure the voicemail system so the phone ignores message waiting indicator (MWI) notifications.
IgnoreUssdExclusions	Set as <b>True</b> to ignore Unstructured Supplementary Service Data (USSD) exclusions.
ResetCallForwarding	When set to <b>True</b> , user is provided with an option to retry call forwarding settings query.
ShowCallerIdNetworkDefaultSetting	Indicates whether the network default setting can be allowed for outgoing caller ID.
ShowVideoCallingSwitch	Use to specify whether to show the video capability sharing switch on the mobile device's Settings screen.
SupressVideoCallingChargesDialog	Configure the phone settings CPL to supress the video calling charges dialog.
UssdExclusionList	List used to exclude predefined USSD entries, allowing the number to be sent as standard DTMF tones instead. Set <b>UssdExclusionList</b> to the list of desired exclusions, separated by semicolons. For example, setting the value to 66;330 will override 66 and 330. Leading zeros are specified by using F. For example, to override code 079, set the value to F79. If you set <b>UssdExclusionList</b> , you must set <b>IgnoreUssdExclusions</b> as well. Otherwise, the list will be ignored. See <a href="#">List of USSD codes</a> for values.
WiFiCallingOperatorName	Enter the operator name to be shown when the phone is using WiFi calling. If you don't set a value for <b>WiFiCallingOperatorName</b> , the device will always display <b>SIMServiceProviderName Wi-Fi</b> , where <b>SIMServiceProviderName</b> is a string that corresponds to the SPN for the SIM on the device. If the service provider name in the SIM is not set, only <b>Wi-Fi</b> will be displayed.

## PhoneSettings

SETTING	DESCRIPTION
AssistedDialSetting	Turn off the international assist feature that helps users with the country codes needed for dialing international phone numbers.

SETTING	DESCRIPTION
CallIDMatch	Sets the number of digits that the OS will try to match against contacts for Caller ID. For any country/region that doesn't exist in the default mapping table, mobile operators can use this legacy CallIDMatch setting to specify the minimum number of digits to use for matching caller ID.
ContinuousDTMFEEnabled	Enable DTMF tone duration for as long as the user presses a dialpad key.
DisableVoicemailPhoneNumberDisplay	Disable the display of the voicemail phone number below the Voicemail label in call progress dialog.
HideCallForwarding	Partners can hide the user option to turn on call forwarding. By default, users can decide whether to turn on call forwarding. Partners can hide this user option so that call forwarding is permanently disabled.
ShowLongTones	Partners can make a user option visible that makes it possible to toggle between short and long DTMF tones, instead of the default continuous tones. By default, the phone supports Dual-Tone Multi-frequency (DTMF) with continuous tones. Partners can make a user option visible that makes it possible to toggle between short and long tones instead.
UseOKForUssdDialogs	OEMs can change the button label in USSD dialogs from <b>Close</b> (the default) to <b>OK</b> .
VoLTEAudioQualityString	Partners can add a string to the call progress screen to indicate if the active call is a high quality voice over LTE (VoLTE). Set the value of VoLTEAudioQualityString to the string that you want to display in the call progress screen to indicate that the call is a VoLTE call. This string is combined with the PLMN so if the string is "VoLTE", the resulting string is "PLMN_String VoLTE". For example, the string displayed in the call progress screen can be "Litware VoLTE" if the PLMN_String is "Litware". The value you specify for VoLTEAudioQualityString must exceed 10 characters.

## SupplementaryServiceCodeOverrides

See [Dialer codes for supplementary services](#).

## VoicemailRegistrationTable

Configure these settings to customize visual voicemail in the Windows 10 Mobile UI. For settings and values, see [Visual voicemail](#).

## List of USSD codes

CODES	DESCRIPTION	DWORD VALUE
04	CHANGEPIN	000000F4
042	CHANGEPIN2	00000F42

<b>CODES</b>	<b>DESCRIPTION</b>	<b>DWORD VALUE</b>
05	UNBLOCKPIN	000000F5
052	UNBLOCKPIN2	00000F52
03	SSCHANGEPASSWORD	000000F3
75	EMLPPBASE	00000075
750	EMLPPLVEL0	00000750
751	EMLPPLVEL1	00000751
752	EMLPPLVEL2	00000752
753	EMLPPLVEL3	00000753
754	EMLPPLVEL4	00000754
66	CALLDEFLECT	00000066
30	CALLIDCLIP	00000030
31	CALLIDCLIR	00000031
76	CALLIDCOLP	00000076
77	CALLIDCOLR	00000077
21	FWDUNCONDITIONAL	00000021
67	FWDBUSY	00000067
61	FWDNOREPLY	00000061
62	FWDNOTREACHABLE	00000062
002	FWDALL	00000FF2
004	FWDALLCONDITIONAL	00000FF4
43	CALLWAITING	00000043
360	UUSALL	00000360
361	UUSSERVICE1	00000361
362	UUSSERVICE2	00000362
363	UUSSERVICE3	00000363

<b>CODES</b>	<b>DESCRIPTION</b>	<b>DWORD VALUE</b>
33	BARROUT	00000033
331	BARROUTINTL	00000331
332	BARROUTINTLEXTOHOME	00000332
35	BARRIN	00000035
351	BARRINROAM	00000351
330	BARRALL	00000330
333	BARRALLOUT	00000333
353	BARRALLIN	00000353
354	BARRINCOMINGINTERMEDIATE	00000354
96	CALLTRANSFER	00000096
37	CALLCOMPLETEBUSY	00000037
070	PNP0	00000F70
071	PNP1	00000F71
072	PNP2	00000F72
073	PNP3	00000F73
074	PNP4	00000F74
075	PNP5	00000F75
076	PNP6	00000F76
077	PNP7	00000F77
078	PNP8	00000F78
079	PNP9	00000F79
300	CALLCNAP	00000300
591	MSP1	00000591
592	MSP2	00000592
593	MSP3	00000593

<b>CODES</b>	<b>DESCRIPTION</b>	<b>DWORD VALUE</b>
594	MSP4	00000594

# CellCore (Windows Configuration Designer reference)

10/17/2017 • 51 min to read • [Edit Online](#)

Use to configure settings for cellular data.

## IMPORTANT

These settings are intended to be used only by manufacturers, mobile operators, and solution providers when configuring devices, and are not intended for use by administrators in the enterprise.

## Applies to

SETTING GROUPS	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
PerDevice: <a href="#">CellConfiguration</a> s		X			
PerDevice: <a href="#">CellData</a> CellularFailover	X	X			
PerDevice: <a href="#">CellData</a> MaxNumberOfPD PContexts		X			
PerDevice: <a href="#">CellData</a> ModemProfiles		X			
PerDevice: <a href="#">CellData</a> PersistAtImaging		X			
PerDevice: <a href="#">CellUX</a>		X			
PerDevice: <a href="#">CGDual</a>		X			
PerDevice: <a href="#">eSim</a>	X	X			
PerDevice: <a href="#">External</a>		X			
PerDevice: <a href="#">General</a>		X			
PerDevice: <a href="#">RCS</a>		X			

SETTING GROUPS	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
PerDevice: <a href="#">SMS</a>	X	X			
PerDevice: <a href="#">UIX</a>		X			
PerDevice: <a href="#">UTK</a>		X			
PerIMSI: <a href="#">CellData</a>		X			
PerIMSI: <a href="#">CellUX</a>		X			
PerIMSI: <a href="#">General</a>		X			
PerIMSI: <a href="#">RCS</a>		X			
PerIMSI: <a href="#">SMS</a>	X	X			
PerIMSI: <a href="#">UTK</a>		X			
PerIMSI: <a href="#">VoLTE</a>		X			

## PerDevice

### CellConfigurations

1. In **CellConfiguration > PropertyGroups**, enter a name for the property group.
2. Select the **PropertyGroups** you just created in the **Available customizations** pane and then enter a **PropertyName**.
3. Select the **PropertyName** you just created in the **Available customizations** pane, and then select one of the following data types for the property:
  - Binary
  - Boolean
  - Integer
  - String
4. The data type that you selected is added in **Available customizations**. Select it to enter a value for the property.

### CellData

SETTING	DESCRIPTION
CellularFailover	Allow or disallow cellular data failover when in limited Wi-Fi connectivity. By default, if the phone is connected to a Wi-Fi network and the data connection to a site is unsuccessful due to limited Wi-Fi connectivity, the phone will complete the connection to the site using available cellular data networks (when possible) to provide an optimal user experience. When the customization is enabled, a user option to use or not use cellular data for limited Wi-Fi connectivity becomes visible in the <b>Settings &gt; cellular+SIM</b> screen. This option is automatically set to <b>don't use cellular data</b> when the customization is enabled.

SETTING	DESCRIPTION
MaxNumberOfPDPContexts	Set a maximum value (1 through 4, inclusive, or 0x1 through 0x4 hexadecimal) for the number of simultaneous packet data protocol (PDP) contexts for 3GPP connections. By default, the OS enforces a maximum of four (4) simultaneous packet data protocol (PDP) contexts for 3GPP connections, and one (1) PDP context for 3GPP2 connections. You can set a different maximum value if required by their mobile operator. The same maximums apply for both roaming and non-roaming scenarios. This maximum does not include packet contexts used internally by the modem.
ModemProfiles > LTEAttachGuids	Set the value for LTEAttachGuid to the OemConnectionId GUID used for the LTE attach profile in the modem. The value is a GUID in the string format XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXX.
PersistAtImaging > DisableAoAc	Enable or disable Always-on/Always-connected (AoAc) on the WWAN adapter.

## CellUX

SETTING	DESCRIPTION
APNAuthTypeDefault	Select between <b>Pap</b> and <b>Chap</b> for default APN authentication type.
APNIPTypelfHidden	Select between <b>IPV4</b> , <b>IPV6</b> , <b>IPV4V6</b> , and <b>IPV4V6XLAT</b> for default APN IP type.
Critical > ShowVoLTEToggle	Select <b>Yes</b> to show the VoLTE toggle in the <b>Settings &gt; Cellular+SIM &gt; SIM</b> settings page. Select <b>No</b> to hide the toggle.
Disable2GByDefault	Select <b>Yes</b> to disable 2G by default. Select <b>No</b> to enable 2G.
Disabled2GNoticeDescription	Enter text to customize the notification for disabled 2G.
GenericWifiCallingErrorMessage	Enter text to customize the generic error message when a Wi-Fi calling error occurs.
Hide3GPP2ModeSelection	Select <b>Yes</b> to hide the <b>CDMA</b> option in the network <b>Mode</b> selection drop-down menu. Select <b>No</b> to show the <b>CDMA</b> option.
Hide3GPP2Selection	For 3GPP2 or CDMA phones, select <b>Yes</b> to hide the <b>Network Type</b> drop-down menu in the <b>SIM</b> settings screen. Select <b>No</b> to show <b>Network Type</b> .
Hide3GPPNetworks	For 3GPP or GSM phones, select <b>Yes</b> to hide the <b>Network Type</b> drop-down menu in the <b>SIM settings</b> screen. Select <b>No</b> to show <b>Network Type</b> .
HideAPN	Select <b>Yes</b> to hide the <b>add internet APN</b> button in the <b>SIM settings</b> screen. Select <b>No</b> to show <b>add internet APN</b> .

SETTING	DESCRIPTION
HideAPNAuthType	Select <b>Yes</b> to hide the APN authentication selector. Select <b>No</b> to show the APN authentication selector.
HideAPNIPType	Select <b>Yes</b> to hide the <b>IP type</b> list in the <b>internet APN</b> settings screen. Select <b>No</b> to show <b>IP type</b> .
HideDisabled2GNotice	Select <b>Yes</b> to hide the notification for disabled 2G. Select <b>No</b> to show the notification for disabled 2G.
HideHighestSpeed	Select <b>Yes</b> to hide the <b>Highest connection speed</b> drop-down menu on the <b>Settings &gt; Cellular+SIM &gt; SIM</b> settings page. Select <b>No</b> to show <b>Highest connection speed</b> .
HideHighestSpeed2G	Select <b>Yes</b> to hide the 2G option on the <b>Highest connection speed</b> drop-down menu on the <b>Settings &gt; Cellular+SIM &gt; SIM</b> settings page. Select <b>No</b> to show the 2G option.
HideHighestSpeed3GOnly	Select <b>Yes</b> to hide the 3G option on the <b>Highest connection speed</b> drop-down menu on the <b>Settings &gt; Cellular+SIM &gt; SIM</b> settings page. Select <b>No</b> to show the 3G option.
HideHighestSpeed4G	Select <b>Yes</b> to hide the 4G option on the <b>Highest connection speed</b> drop-down menu on the <b>Settings &gt; Cellular+SIM &gt; SIM</b> settings page. Select <b>No</b> to show the 4G option.
HideHighestSpeed4G3GOnly	Select <b>Yes</b> to hide the 4G or 3G Only option on the <b>Highest connection speed</b> drop-down menu on the <b>Settings &gt; Cellular+SIM &gt; SIM</b> settings page. Select <b>No</b> to show the 4G or 3G Only option.
HideHighestSpeed4GOnly	Select <b>Yes</b> to hide the 4G Only option on the <b>Highest connection speed</b> drop-down menu on the <b>Settings &gt; Cellular+SIM &gt; SIM</b> settings page. Select <b>No</b> to show the 4G Only option.
HideLTEAttachAPN	Select <b>Yes</b> to hide the <b>LTE attach APN</b> button on the <b>Settings &gt; Cellular+SIM &gt; SIM</b> settings page. Select <b>No</b> to show the <b>LTE attach APN</b> button.
HideMMSAPN	Select <b>Yes</b> to hide the <b>add mms apn</b> button on the <b>Settings &gt; Cellular+SIM &gt; SIM</b> settings page. Select <b>No</b> to show the <b>add mms apn</b> button.
HideMMSAPNAuthType	Select <b>Yes</b> to hide the APN authentication type selector on the MMS APN page. Select <b>No</b> to show APN authentication selector.
HideMMSAPNIPType	Select <b>Yes</b> to hide the APN IP type selector on the MMS APN page. Select <b>No</b> to show the APN IP type selector.
HideModeSelection	Select <b>Yes</b> to hide the <b>Network Mode selection</b> drop-down menu on the <b>Settings &gt; Cellular+SIM &gt; SIM</b> settings page. Select <b>No</b> to show the <b>Network Mode selection</b> .

Setting	Description
HidePersoUnlock	Select <b>Yes</b> to hide the Perso unlock UI. Select <b>No</b> to show the Perso unlock UI.
HighestSpeed2G	<p>You can customize the listed names of the connection speeds with their own character codes. To modify "2G" to another character code, change the value of HighestSpeed2G.</p> <p>Although there is no limit to the number of characters you can use, if the character code is too long, it will be truncated in the UI.</p>
HighestSpeed3G	<p>You can customize the listed names of the connection speeds with their own character codes. To modify "3G" to another character code, change the value of HighestSpeed3G.</p> <p>Although there is no limit to the number of characters you can use, if the character code is too long, it will be truncated in the UI.</p>
HighestSpeed3GOnly	<p>You can customize the listed names of the connection speeds with their own character codes. To modify "3G Only" to another character code, change the value of HighestSpeed3GOnly. Although there is no limit to the number of characters you can use, if the character code is too long, it will be truncated in the UI.</p>
HighestSpeed3GPreferred	<p>You can customize the listed names of the connection speeds with their own character codes. To modify "3G Preferred" to another character code, change the value of HighestSpeed3GPreferred. Although there is no limit to the number of characters you can use, if the character code is too long, it will be truncated in the UI.</p>
HighestSpeed4G	<p>You can customize the listed names of the connection speeds with their own character codes. To modify "4G" to another character code, change the value of HighestSpeed4G.</p> <p>Although there is no limit to the number of characters you can use, if the character code is too long, it will be truncated in the UI.</p>
HighestSpeed4G3GOnly	<p>You can customize the listed names of the connection speeds with their own character codes. To modify "4G or 3G Only" to another character code, change the value of HighestSpeed4G3GOnly. Although there is no limit to the number of characters you can use, if the character code is too long, it will be truncated in the UI.</p>
HighestSpeed4GOnly	<p>You can customize the listed names of the connection speeds with their own character codes. To modify "4G Only" to another character code, change the value of HighestSpeed4GOnly. Although there is no limit to the number of characters you can use, if the character code is too long, it will be truncated in the UI.</p>
HighestSpeedTitle	<p>You can customize the <b>Highest connection speed</b> drop-down label in the <b>Settings &gt; Cellular+SIM &gt; SIM</b> settings page. To change the Highest connection speed drop-down label, set HighestSpeedTitle to another string. For example, you can set this to "Preferred connection speed".</p>

Setting	Description
IsATTSpecific	<p>Control the roaming text for AT&amp;T devices. AT&amp;T requires the phone to show a particular roaming text to meet their legal and marketing guidelines. By default, if the user chooses <b>roam</b> under <b>Data roaming options</b> in the <b>Settings &gt; Cellular+SIM</b> screen, they will see the following text:</p> <p><i>Depending on your service agreement, you might pay more when using data roaming.</i> If you set IsATTSpecific to <b>Yes</b>, the following roaming text will be displayed instead: <i>International data roaming charges apply for data usage outside the United States, Puerto Rico, and United States Virgin Islands. Don't allow roaming to avoid international data roaming charges.</i></p>
LTEAttachGUID	<p>Set the value for LTEAttachGuid to the OemConnectionId GUID used for the LTE attach profile in the modem. The value is a GUID in the string format XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXX.</p>
MMSAPNAuthTypeDefault	<p>Select between <b>Pap</b> and <b>Chap</b> for default MMS APN authentication type.</p>
MMSAPNITypeIfHidden	<p>Select between <b>IPV4</b>, <b>IPV6</b>, <b>IPV4V6</b>, and <b>IPV4V6XLAT</b> for default MMS APN IP type.</p>
ShowExtendedRejectCodes	<p>When a reject code is sent by the network, partners can specify that extended error messages should be displayed instead of the standard simple error messages. This customization is only intended for use when required by the mobile operator's network. The short versions of the extended reject message are shown in the following screens:</p> <ul style="list-style-type: none"> <li>- Phone tile in Start</li> <li>- Call History screen</li> <li>- Dialer</li> <li>- Call Progress screen</li> <li>- Incoming Call screen</li> <li>- As the status string under Settings &gt; cellular+SIM</li> </ul> <p>The long version of the extended reject message is shown under the Active Network label in <b>Settings &gt; cellular+SIM</b>. Select <b>Yes</b> to show the extended error message. Select <b>No</b> to hide the extended error message. See <a href="#">Error messages for reject codes</a> to see the versions of the message.</p>
ShowHighestSpeed3GPreferred	<p>Select <b>Yes</b> to show the <b>3G Preferred</b> option in the <b>Highest connection speed</b> drop-down menu. Select <b>No</b> to hide <b>3G Preferred</b>.</p>
ShowManualAvoidance	<p>Select <b>Yes</b> to show the <b>Switch to next network manually</b> button in SIM settings when Mode Selection is CDMA on a C+G dual SIM phone. Select <b>No</b> to hide the <b>Switch to next network manually</b> button</p>

SETTING	DESCRIPTION
ShowPreferredPLMNPage	Select <b>Yes</b> to show the preferred public land mobile network (PLMN) page in SIM settings.
ShowSpecificWifiCallingError	Select <b>Yes</b> to show a specific error message based on operator requirements.
ShowViewAPN	Select <b>Yes</b> to show the <b>View Internet APN</b> button in <b>Settings &gt; cellular+SIM</b> .
ShowWifiCallingEmergencyCallWarning	Select <b>Yes</b> to show Wi-Fi emergency call warning.
ShowWifiCallingError	Select <b>Yes</b> to show Wi-Fi calling error message.
SuppressDePersoUI	Select <b>Yes</b> to hide the perso unlock UI.

## CGDual

Use **CGDual > RestrictToGlobalMode** to configure settings for global mode on C+G Dual SIM phones. When the device registration changes, if the value for this setting is set, the OS changes the preferred system type to the default preferred system type for world mode. If the phone is not camped on any network, the OS assumes the phone is on the home network and changes the network registration preference to default mode.

Select from the following:

- **RestrictToGlobalMode\_Disabled**: the phone is not restricted to global mode.
- **RestrictToGlobalMode\_Home**: when a slot is registered at home and supports global mode, the mode selection is restricted to global mode.
- **RestrictToGlobalMode\_Always**: if a slot supports global mode and this value is selected, the mode selection is restricted to global mode.

## eSim

Configure **FwUpdate > AllowedAppIdList** to whitelist apps that are allowed to update the firmware. Obtain the app IDs from the card vendor.

## External

SETTING	DESCRIPTION
CallSupplementaryService > OTASPNonStandardDialString	Enter a list of all desired non-standard OTASP dial strings.
CarrierSpecific > FallBackMode	Select between <b>GWCSFB</b> and <b>1xCSFB</b> for fallback mode.
CarrierSpecific > VZW > ActSeq	Enables activation for 4G VZW card. Do not configure this setting for non-VZW devices.
EnableLTEsnrReporting	Select between <b>Use only RSRP</b> and <b>Use both RSRP and ECNO</b> to check if SNR needs to be used for LTE Signal Quality calculations.
EnableUMTSEcnoReporting	Select between <b>Use only RSSI</b> and <b>Use both RSSI and SNR</b> to check if SNR needs to be used for UMTS Signal Quality calculations.

Setting	Description
ImageOnly > ERI > AlgorithmMBB0	Select between <b>Sprint</b> and <b>Verizon</b> to specify the ERI algorithm in MBB for subscription 0.
ImageOnly > ERI > AlgorithmMBB1	Select between <b>Sprint</b> and <b>Verizon</b> to specify the ERI algorithm in MBB for subscription 1.
ImageOnly > ERI > AlgorithmWmRil	Select between <b>Sprint</b> and <b>Verizon</b> to specify the ERI-based notification algorithm.
ImageOnly > ERI > DataFileNameWmRil	Specify the location of the ERI file on the device; for example, C:\Windows\System32\SPCS_en.eri . SPCS_en.eri is a placeholder. Obtain the ERI file name from the mobile operator and replace this filename with it.
ImageOnly > ERI > EnabledWmRil	Enable or disable ERI-based notifications.
ImageOnly > ERI > ERIDataFileNameMBB0	Specify the ERI data file name with international roaming list for Verizon in MBB for subscription 0.
ImageOnly > ERI > ERIDataFileNameMBB1	Specify the ERI data file name with international roaming list for Verizon in MBB for subscription 1.
ImageOnly > ERI > ERISprintIntlRoamDataFileNameMBB0	Specify the ERI data file name with international roaming list for Sprint in MBB for subscription 0.
ImageOnly > ERI > ERISprintIntlRoamDataFileNameMBB1	Specify the ERI data file name with international roaming list for Sprint in MBB for subscription 1.
ImageOnly > ERI > SprintInternationalERIVValuesWmRil	Specify the international ERI values for Sprint as to 4A,7C,7D,7E,9D,9E,9F,C1,C2,C3,C4,C5,C6,E4,E5,E6,E7,E8. .
ImageOnly > MTU > DormancyTimeout0	Enter the number of milliseconds to wait after dormancy hint before telling the modem to make the air interface dormant for subscription 0. Minimum value is 1703, and maximum value is 5000.
ImageOnly > MTU > DormancyTimeout1	Enter the number of milliseconds to wait after dormancy hint before telling the modem to make the air interface dormant for subscription 1. Minimum value is 1703, and maximum value is 5000.
ImageOnly > MTU > MTUDataSize	Customize the TCP maximum segment size (MSS) by setting the maximum transmission unit (MTU) data size if the MSS does not meet the requirements of the mobile operator network. For TCP, the default maximum transmission unit (MTU) is set to 1500 bytes, which makes the maximum segment size (MSS) 1460 bytes. In general, this value should not be changed, as the user experience will degrade if low values are set. However, if the MSS does not meet the requirements of the mobile operator network, OEMs can customize it by setting the MTU data size. This customization configures the MTU, so the size should be set to the required MSS size plus 40 bytes.

SETTING	DESCRIPTION
ImageOnly > MTU > RoamingMTUDataSize	Customize the TCP maximum segment size (MSS) for roaming by setting the maximum transmission unit (MTU) data size if the MSS does not meet the requirements of the mobile operator network. For TCP, the default maximum transmission unit (MTU) is set to 1500 bytes, which makes the maximum segment size (MSS) 1460 bytes. In general, this value should not be changed, as the user experience will degrade if low values are set. However, if the MSS does not meet the requirements of the mobile operator network, OEMs can customize it for roaming by setting the MTU data size. This customization configures the MTU, so the size should be set to the required MSS size plus 40 bytes.
ImageOnly > SuppressNwPSDetach	Configure whether to suppress reporting of network-initiated PS detach (appear attached to OS) until deregistered.
SignalBarMapping Table	You can modify the percentage values used for the signal strength in the status bar per filter. For details, see <a href="#">Custom percentages for signal strength bars</a> .
SRVCCAutoToggleWmRil	Configure whether to link SRVCC to VOLTE on/off.

## General

SETTING	DESCRIPTION
atomicRoamingTableSettings3GPP	If you enable 3GPP roaming, configure the following settings: <ul style="list-style-type: none"> <li>- <b>Exceptions</b> maps the SerialNumber key to the Exceptions value. The wildcard, \$(SerialNumber), is a 3-digit decimal serial number (000 through 999) represented as a string. The wildcard is used as a regvalue under the "Exceptions" subkey. Multiple reg values in this form may be configured or customized by the OEM, all placed under the same subkey (Exceptions). The data in the regvalue is a string representing an MCC-MNC pair, such as "410510" where 410 is the MCC and 510 is the MNC.</li> <li>- <b>HomePLMN</b> maps the SerialNumber key to the HomePLMN value. The wildcard, \$(SerialNumber), is a 3-digit decimal serial number (000 through 999) represented as a string. The wildcard is used as a regvalue under the "HomePLMN" subkey. Multiple reg values in this form may be configured or customized by the OEM, all placed under the same subkey (HomePLMN). The data in the regvalue is a string representing an MCC-MNC pair, such as "410510" where 410 is the MCC and 510 is the MNC.</li> <li>- <b>TargetImsi</b> maps the SerialNumber key to the TargetImsi value. The wildcard, \$(SerialNumber), is a 3-digit decimal serial number (000 through 999) represented as a string. The wildcard is used as a regvalue under the "TargetImsi" subkey. Multiple reg values in this form may be configured or customized by the OEM, all placed under the same subkey (TargetImsi). The data in the regvalue is a string representing an MCC-MNC pair, such as "410510" where 410 is the MCC and 510 is the MNC.</li> </ul>

Setting	Description
atomicRoamingTableSettings3GPP2	<p>If you enable 3GPP2 roaming, configure the following settings:</p> <ul style="list-style-type: none"> <li>- <b>Home</b> maps the SerialNumber key to the Home value. The wildcard, \$(SerialNumber), is a 3-digit decimal serial number (000 through 999) represented as a string. The wildcard is used as a regvalue under the "Home" subkey. Multiple reg values in this form may be configured or customized by the OEM, all placed under the same subkey (Home). The data in the regvalue is a DWORD representing the Roaming Indicator.</li> <li>- <b>Roaming</b> maps the SerialNumber key to the Roaming value. The wildcard, \$(SerialNumber), is a 3-digit decimal serial number (000 through 999) represented as a string. The wildcard is used as a regvalue under the "Roaming" subkey. Multiple reg values in this form may be configured or customized by the OEM, all placed under the same subkey (Roaming). The data in the regvalue is a DWORD representing the Roaming Indicator.</li> </ul>
AvoidStayingInManualSelection	<p>You can enable permanent automatic mode for mobile networks that require the cellular settings to revert to automatic network selection after the user has manually selected another network when roaming or out of range of the home network.</p>
CardAllowList	<p>Define the list of SIM cards allowed in the first slot of a C+G dual SIM phone. This setting is used only if <b>CardLock</b> is set to allow it. If <b>CardLock</b> is not set, this list is ignored. To configure the list of SIM cards allowed in the first slot, set the value for CardAllowList to a comma-separated MCC:MNC list. You can also use wild cards, represented by an asterisk, to accept any value. For example, you can set the value to</p> <div style="border: 1px solid black; padding: 2px; display: inline-block;">310:410,311:*,404:012,310:70</div>
CardBlockList	<p>Define the list of SIM cards that are not allowed in the first slot of a C+G dual SIM phone. This setting is used only if <b>CardLock</b> is set to allow it. If <b>CardLock</b> is not set, this list is ignored. To configure the list of SIM cards that are not allowed in the first slot, set the value for CardBlockList to a comma separated MCC:MNC list. You can also use wild cards, represented by an asterisk, to accept any value. For example, you can set the value to</p> <div style="border: 1px solid black; padding: 2px; display: inline-block;">310:410,311:*,404:012,310:70</div>
CardLock	<p>Used to enforce either the card allow list or both the card allow and block lists on a C+G dual SIM phone.</p>
DefaultSlotAffinity	<p>Set the data connection preference for:</p> <ul style="list-style-type: none"> <li>- <b>SlotAffinityForInternetData_Automatic</b>: data connection preference is automatically set</li> <li>- <b>SlotAffinityForInternetData_Slot0</b>: sets the data connection preference to Slot 0. The data connection cannot be edited by the user.</li> <li>- <b>SlotAffinityForInternetData_Slot1</b>: Sets the data connection preference to Slot 1. The data connection cannot be edited by the user.</li> </ul>
DisableLTESupportWhenRoaming	<p>Set to <b>Yes</b> to disable LTE support when roaming.</p>
DisableSystemTypeSupport	<p>Enter the system types to be removed.</p>

SETTING	DESCRIPTION
DTMFOffTime	Sets the length of time, in milliseconds (between 64 and 1000 inclusive), of the pause between DTMF digits. For example, a value of 120 specifies 0.12 seconds.
DTMFOnTime	Sets the length of time, in milliseconds (between 64 and 1000 inclusive), to generate the DTMF tone when a key is pressed. For example, a value of 120 specifies 0.12 seconds.
ExcludedSystemTypesByDefault	Set the default value for <b>Highest connection speed</b> in the <b>Settings &gt; Cellular &amp; SIM &gt; SIM</b> screen by specifying the bitmask for any combination of radio technology to be excluded from the default value. The connection speed that has not been excluded will show up as the highest connection speed. On dual SIM phones that only support up to 3G connection speeds, the <b>Highest connection speed</b> option is replaced by a 3G on/off toggle based on the per-device setting. Enter the binary setting to exclude 4G ( <code>10000</code> ) or 3G ( <code>01000</code> ).
ExcludedSystemTypesPerOperator	Exclude specified system types from SIM cards that match the MCC:MNC pairs listed in <b>OperatorListForExcludedSystemTypes</b> . This setting is used only for China. Set the value to match the system type to be excluded. For more information about the RIL system types, see <a href="#">RILSYSTEMTYPE</a> . For example, a value of 0x8 specifies RIL_SYSTEMTYPE_UMTS (3G) while 0x10 specifies RIL_SYSTEMTYPE_LTE (4G). To exclude more than one system type, perform a bitwise OR operation on the radio technologies you want to exclude. For example, a bitwise OR operation on RIL_SYSTEMTYPE_LTE (4G) and RIL_SYSTEMTYPE_UMTS (3G) results in the value 11000 (binary) or 0x18 (hexadecimal). In this case, the ExcludedSystemTypesPerOperator value must be set to 0x18 to limit the matching MCC:MNC pairs to 2G.
LTEEnabled	Select <b>Yes</b> to enable LTE, and <b>No</b> to disable LTE.
LTEForced	Select <b>Yes</b> to force LTE.
ManualNetworkSelectionTimeout	Set the default network selection timeout value, in a range of 1-600 seconds. By default, the OS allows the phone to attempt registration on the manually selected network for 60 seconds (or 1 minute) before it switches back to automatic mode. This value is the amount of time that the OS will wait for the modem to register on the manually selected network. If the time lapses and the modem was not able to register on the network that was manually selected by the user, the OS will either switch back to the automatic network selection mode if Permanent automatic mode is enabled, and the user has manually selected a network or the modem was turned on, or display a dialog that notifies the user that the phone was unable to connect to the manually selected network after the phone was turned on or after airplane mode was turned off.

SETTING	DESCRIPTION
NetworkSuffix	<p>To meet branding requirements for some mobile operators, you can add a suffix to the network name that is displayed on the phone. For example, you can change from ABC to ABC 3G when under 3G coverage. This feature can be applied for any radio access technology (RAT). For TD-SCDMA RAT, a 3G suffix is always appended by default, but partners can also customize this the same way as with any other RAT. In the setting name, set SYSTEMTYPE to the network type that you want to append the network name to and click <b>Add</b>:</p> <ul style="list-style-type: none"> <li>- system type 4: 2G (GSM)</li> <li>- system type 8: 3G (UMTS)</li> <li>- system type 16: LTE</li> <li>- system type 32: 3G (TS-SCDMA)</li> </ul> <p>Select the system type that you added, and enter the network name and suffix that you want displayed.</p>
NitzFiltering	<p>For mobile networks that can receive Network Identity and Time Zone (NITZ) information from multiple sources, partners can set the phone to ignore the time received from an LTE network. Time received from a CDMA network is not affected. Set the value of NitzFiltering to <code>0x10</code>.</p>
OperatorListForExcludedSystemTypes	<p>Enter a comma-separated list of MCC and MNC (MCC:MNC) for which system types should be restricted. For mobile operators that require more control over the system types that their phones use to connect to the mobile operators' networks, OEMs can specify the MCC and MNC of other specific operators that the main mobile operator wishes to limit. If the UICC's MCC and MNC matches any of the pairs that OEMs can specify for the operator, a specified RIL system type will be removed from the UICC regardless of its app types, slot position, or executor mapping. This setting is used only for China. OEMs should not use this setting unless required by the mobile operator. Set the value of the OperatorListForExcludedSystemTypes setting a comma separated list of MCC:MNC pairs for which the system types should be restricted. For example, the value can be set to 310:026,310:030 to restrict operators with an MCC:MNC of 310:026 and 310:030.</p>
OperatorPreferredForFasterRadio	<p>Set Issuer Identification Number (IIN) or partial ICCID of preferred operator for the faster radio. For mobile operators that require more control over the system types that their phones use to connect to the mobile operators' networks, OEMs can map a partial ICCID or an Industry Identification Number (IIN) to the faster radio regardless of which SIM card is chosen for data connectivity. This setting is used only for China. OEMs should not use this setting unless required by the mobile operator. To map a partial ICCID or an IIN to the faster radio regardless of which SIM card is chosen for data connectivity, set the value of OperatorPreferredForFasterRadio to match the IIN or the ICCID, up to 7 digits, of the preferred operator.</p>

Setting	Description
PreferredDataProviderList	<p>OEMs can set a list of MCC/MNC pairs for the purchase order (PO) carrier or primary operator. For mobile operators that require it, OEMs can set a list of MCC/MNC pairs for the purchase order (PO) carrier or primary operator so that it can be set as the default data line for phones that have a dual SIM. When the PO SIM is inserted into the phone, the OS picks the PO SIM as the data line and shows a notification to the user that the SIM has been selected for Internet data. If two PO SIMs are inserted, the OS will choose the first PO SIM that was detected as the default data line and the mobile operator action required dialogue (ARD) is shown. If two non-PO SIMs are inserted, the user is prompted to choose the SIM to use as the default data line. Note OEMs should not set this customization unless required by the mobile operator. To enumerate the MCC/MNC value pairs to use for data connections, set the value for <b>PreferredDataProviderList</b>. The value must be a comma-separated list of preferred MCC:MNC values. For example, the value can be 301:026,310:030 and so on.</p>
Slot2DisableAppsList	<p>Disable specified apps from slot 2 on a C+G dual SIM phone. To disable a list of specified apps from Slot 2, set Slot2DisableAppsList to a comma-separated list of values representing the apps. For example, <code>4,6</code>.</p>
Slot2ExcludedSystemTypes	<p>Exclude specified system types from SIM cards inserted in Slot 2. For mobile operators that require more control over the system types that their phones use to connect to the mobile operators' networks, OEMs can restrict the second slot in a dual-SIM phone regardless of what apps or executor mapping the second slot is associated with. Note This setting is used only for China. OEMs should not use this setting unless required by the mobile operator. To allow an operator to simply restrict the second slot in a dual SIM phone regardless of what apps or executor mapping the second slot is associated with, set the value of Slot2ExcludedSystemTypes to the system types to be excluded from the SIM cards inserted in Slot 2. For example, a value of 0x8 specifies RIL_SYSTEMTYPE_UMTS (3G) while 0x10 specifies RIL_SYSTEMTYPE_LTE (4G). To exclude more than one system type, perform a bitwise OR operation on the radio technologies you want to exclude. For example, a bitwise OR operation on RIL_SYSTEMTYPE_LTE (4G) and RIL_SYSTEMTYPE_UMTS (3G) results in the value 11000 (binary) or 0x18 (hexadecimal). In this case, any SIM inserted in Slot 2 will be limited to 2G. For more information about the RIL system types, see <a href="#">RILSYSTEMTYPE</a>.</p>
SuggestDataRoamingARD	<p>Use to show the data roaming suggestion dialog when roaming and the data roaming setting is set to no roaming.</p>
SuggestGlobalModeARD	<p>Define whether Global Mode is suggested on a C+G dual SIM phone.</p>
SuggestGlobalModeTimeout	<p>To specify the number of seconds to wait for network registration before suggesting global mode, set SuggestGlobalModeTimeout to a value between 1 and 600, inclusive. For example, to set the timeout to 60 seconds, set the value to 60 (decimal) or 0x3C (hexadecimal).</p>

## RCS

SETTING	DESCRIPTION
SystemEnabled	Select <b>Yes</b> to specify that the system is RCS-enabled.
UserEnabled	Select <b>Yes</b> to show the user setting if RCS is enabled on the device.

## SMS

SETTING	DESCRIPTION
AckExpirySeconds	Set the value, in seconds, for how long to wait for a client ACK before trying to deliver.
DefaultMCC	Set the default mobile country code (MCC).
Encodings > GSM7BitEncodingPage	Enter the code page value for the 7-bit GSM default alphabet encoding. Values: - Code page value: 55000 (Setting value: 0xD6D8)(Code page: default alphabet) - Code page value: 55001 (Setting value: 0xD6D9)(Code page: GSM with single shift for Spanish)- Code page value: 55002 (Setting value: 0xD6DA)(Code page: GSM with single shift for Portuguese)- Code page value: 55003 (Setting value: 0xD6DB) (Code page: GSM with single shift for Turkish)- Code page value: 55004 (Setting value: 0xD6DC)(Code page: SMS Greek Reduction)
Encodings > GSM8BitEncodingPage	Enter the code page value for GSM 8-bit encoding (OEM set). OEM-created code page IDs should be in the range 55050–55099. For more information, see [Add encoding extension tables for SMS] <a href="https://docs.microsoft.com/windows-hardware/customize/mobile/mcsf/add-encoding-extension-tables-for-sms">https://docs.microsoft.com/windows-hardware/customize/mobile/mcsf/add-encoding-extension-tables-for-sms</a> .
Encodings > OctetEncodingPage	Set the octet (binary) encoding.
Encodings > SendUDHNLSS	Set the 7 bit GSM shift table encoding.
Encodings > UseASCII	Set the 7 bit ASCII encoding. Used only for CDMA carriers that use 7-bit ASCII encoding instead of GSM 7-bit encoding.
Encodings > UseKeyboardLangague	Set whether to use the keyboard language (Portuguese, Spanish, or Turkish) based encoding (set shift table based on keyboard language).
IncompleteMsgDeliverySeconds	Set the value, in seconds, for long to wait for all parts of multisegment Sprint messages for concatenation.

Setting	Description
MessageExpirySeconds	<p>Partners can set the expiration time before the phone deletes the received parts of a long SMS message. For example, if the phone is waiting for a three-part SMS message and the first part has been received, the first part will be deleted when the time expires and the other part of the message has not arrived. If the second part of the message arrives before the time expires, the first and second parts of the message will be deleted if the last part does not arrive after the time expires. The expiration time is reset whenever the next part of the long message is received. Set MessageExpirySeconds to the number seconds that the phone should wait before deleting the received parts of a long SMS messages. This value should be in hexadecimal and must be prefixed with 0x. The default value is 0x15180, which is equivalent to 1 day or 86,400 seconds.</p>
SmsFragmentLimit	<p>Partners can specify a maximum length for SMS messages. This requires setting both the maximum number of SMS fragments per SMS message, from 1 to 255, and the maximum size in bytes of each SMS fragment, from 16 to 140 bytes. Use SmsFragmentLimit to set the maximum number of bytes in the user data body of an SMS message. You must set the value between 16 (0x10) and 140 (0x8C). You must also use SmsPageLimit to set the maximum number of segments in a concatenated SMS message.</p>
SmsPageLimit	<p>Partners can specify a maximum length for SMS messages. This requires setting both the maximum number of SMS fragments per SMS message, from 1 to 255, and the maximum size in bytes of each SMS fragment, from 16 to 140 bytes. Use SmsPageLimit to set the maximum number of segments in a concatenated SMS message. You must set the value to 255 (0xFF) or smaller. You must also use SmsFragmentLimit to set the maximum number of bytes in the body of the SMS message.</p>
SprintFragmentInfoInBody	<p>Partners can enable the messaging client to allow users to enter more than 160 characters per message. Messages longer than 160 characters are sent as multiple SMS messages that contain a tag at the beginning of the message in the form "(1/2)", where the first number represents the segment or part number and the second number represents the total number of segments or parts. Multiple messages are limited to 6 total segments. When enabled, the user cannot enter more characters after the 6 total segments limit is reached. Any message received with tags at the beginning is recombined with its corresponding segments and shown as one composite message.</p>
Type3GPP > ErrorHandling > ErrorType	<p>Enter a name for ERRORCODE3GPP, and click <b>Add</b>. Configure the error type that you added as <b>Transient Failure</b> or <b>Permanent Failure</b>.</p>
Type3GPP > ErrorHandling > FriendlyErrorClass	<p>Enter a name for ERRORCODE3GPP, and click <b>Add</b>. Configure the error class that you added as <b>generic error</b>, <b>invalid recipient address</b>, or <b>network connectivity trouble</b>.</p>
Type3GPP > IMS > SmsUse16BitReferenceNumbers	<p>Configure whether to use 8-bit or 16-bit message ID (reference number) in the UDH.</p>

SETTING	DESCRIPTION
Type3GPP2 > ErrorHandling > FriendlyErrorClass	Enter a name for ERRORCODE3GPP2, and click <b>Add</b> . Configure the error class that you added as <b>generic error</b> , <b>invalid recipient address</b> , or <b>network connectivity trouble</b> .
Type3GPP2 > ErrorHandling > UseReservedAsPermanent	Set the 3GPP2 permanent error type.

## UIX

SETTING	DESCRIPTION
SIM1ToUIM1	Used to show UIM1 as an alternate string instead of SIM1 for the first SIM on C+G dual SIM phones.
SIMToSIMUIM	Partners can change the string "SIM" to "SIM/UIM" to accommodate scenarios such as Dual Mode cards of SIM cards on the phone. This can provide a better user experience for users in some markets. Enabling this customization changes all "SIM" strings to "SIM/UIM".

## UTK

SETTING	DESCRIPTION
UIDefaultDuration	Specifies the default time, in milliseconds, that the DISPLAY TEXT, GET INKEY, PLAY TONE, or SELECT ITEM dialog should be displayed. The default value is 60000 milliseconds (60 seconds). The valid value range is 1-120000.
UIGetInputDuration	Specifies the default time, in milliseconds, that the GET INPUT dialog should be displayed. The default value is 120000 milliseconds (120 seconds). The valid value range is 1-120000.

## PerIMSI

Enter an IMSI, click **Add**, and then select the IMSI that you added to configure the following settings.

### CellData

SETTING	DESCRIPTION
MaxNumberOfPDPContexts	OEMs can set a maximum value for the number of simultaneous packet data protocol (PDP) contexts for 3GPP connections. By default, the OS enforces a maximum of four (4) simultaneous packet data protocol (PDP) contexts for 3GPP connections, and one (1) PDP context for 3GPP2 connections. OEMs can set a different maximum value if required by their mobile operator. The same maximums apply for both roaming and non-roaming scenarios. This maximum does not include packet contexts used internally by the modem.

### CellUX

SETTING	DESCRIPTION
APNIPTypeIfHidden	Used to set the default IP type shown in the <b>IP type</b> listbox on the <b>internet APN</b> settings screen.
Critical > ShowVoLTERoaming	Use to show the IMS roaming control in the cellular settings page
Critical > ShowVoLTEToggle	Show or hide VoLTE toggle.
Critical > SwitchIMS	Switch IMS on or off with a toggle. OEMs can configure the default settings and toggle for IMS services to meet mobile operator requirements. Users can later manually change the default values for these settings if they choose to do so.
Critical > SwitchSMSOverIMS	Switch SMS over IMS on or off when VoLTE is toggled.
Critical > SwitchVideoOverIMS	Use to switch video over IMS when VoLTE is switched.
Critical > SwitchVoiceOverIMS	Switch voice over IMS when VoLTE is toggled.
Critical > SwitchXCAP	Use to switch the XML Configuration Access Protocol (XCAP) when VoLTE is enabled.
Critical > VoLTERoamingOffDescription	Use to customize the description string that appears under IMS roaming control when IMS roaming is turned off. The string must not be longer than 127 characters.
Critical > VoLTERoamingOnDescription	Use to customize the description string that appears under IMS roaming control when IMS roaming is turned on. The string must not be longer than 127 characters.
Critical > VoLTERoamingSettingDisableDuringCall	Use to specify whether to grey out VoLTE roaming settings during an active VoLTE call.
Critical > VoLTERoamingTitle	Use to customize the description string for the IMS roaming control. The string must not be longer than 127 characters.
Critical > VoLTESectionTitle	Use to customize the section title for the IMS settings. The string must not be longer than 127 characters.
Critical > VoLTESettingDisableDuringCall	Use to specify whether to grey out VoLTE-related settings during an active VoLTE call.
Critical > VoLTEToggleDescription	Use to customize the VoLTE toggle description. To customize the VoLTE toggle description, set VoLTEToggleDescription to the name of the resource-only .dll file, specifying the string offset. For example: @DisplayStrings.dll,-101.
Critical > VoLTEToggleSettingDisableDuringCall	Use to specify whether to grey out the VoLTE toggle during an active VoLTE call.
Critical > VoLTEToggleTitle	Use to customize the VoLTE toggle label. To customize the VoLTE toggle label, set VoLTEToggleTitle to the name of the resource-only .dll file, specifying the string offset. For example: @DisplayStrings.dll,-102.

SETTING	DESCRIPTION
Critical > WFCSettingDisableDuringCall	Use to specify whether to grey out the Wi-Fi calling settings during an active VoLTE call.
Disable2GByDefault	Select <b>Yes</b> to disable 2G by default. Select <b>No</b> to enable 2G.
Disabled2GNoticeDescription	Enter text to customize the notification for disabled 2G.
GenericWifiCallingErrorMessage	Enter text to customize the generic error message when a Wi-Fi calling error occurs.
Hide3GPP2ModeSelection	Select <b>Yes</b> to hide the <b>CDMA</b> option in the network <b>Mode</b> selection drop-down menu. Select <b>No</b> to show the <b>CDMA</b> option.
Hide3GPP2Selection	For 3GPP2 or CDMA phones, select <b>Yes</b> to hide the <b>Network Type</b> drop-down menu in the <b>SIM</b> settings screen. Select <b>No</b> to show <b>Network Type</b> .
Hide3GPPNetworks	For 3GPP or GSM phones, select <b>Yes</b> to hide the <b>Network Type</b> drop-down menu in the <b>SIM settings</b> screen. Select <b>No</b> to show <b>Network Type</b> .
HideAPN	Select <b>Yes</b> to hide the <b>add internet APN</b> button in the <b>SIM settings</b> screen. Select <b>No</b> to show <b>add internet APN</b> .
HideAPNIType	Select <b>Yes</b> to hide the <b>IP type</b> list in the <b>internet APN</b> settings screen. Select <b>No</b> to show <b>IP type</b> .
HideDisabled2GNotice	Select <b>Yes</b> to hide the notification for disabled 2G. Select <b>No</b> to show the notification for disabled 2G.
HideHighestSpeed	Select <b>Yes</b> to hide the <b>Highest connection speed</b> drop-down menu on the <b>Settings &gt; Cellular+SIM &gt; SIM</b> settings page. Select <b>No</b> to show <b>Highest connection speed</b> .
HideHighestSpeed2G	Select <b>Yes</b> to hide the 2G option on the <b>Highest connection speed</b> drop-down menu on the <b>Settings &gt; Cellular+SIM &gt; SIM</b> settings page. Select <b>No</b> to show the 2G option.
HideHighestSpeed3GOnly	Select <b>Yes</b> to hide the 3G option on the <b>Highest connection speed</b> drop-down menu on the <b>Settings &gt; Cellular+SIM &gt; SIM</b> settings page. Select <b>No</b> to show the 3G option.
HideHighestSpeed4G	Select <b>Yes</b> to hide the 4G option on the <b>Highest connection speed</b> drop-down menu on the <b>Settings &gt; Cellular+SIM &gt; SIM</b> settings page. Select <b>No</b> to show the 4G option.
HideHighestSpeed4G3GOnly	Select <b>Yes</b> to hide the 4G or 3G Only option on the <b>Highest connection speed</b> drop-down menu on the <b>Settings &gt; Cellular+SIM &gt; SIM</b> settings page. Select <b>No</b> to show the 4G or 3G Only option.

Setting	Description
HideHighestSpeed4GOnly	Select <b>Yes</b> to hide the 4G Only option on the <b>Highest connection speed</b> drop-down menu on the <b>Settings &gt; Cellular+SIM &gt; SIM</b> settings page. Select <b>No</b> to show the 4G Only option.
HideLTEAttachAPN	Select <b>Yes</b> to hide the <b>LTE attach APN</b> button on the <b>Settings &gt; Cellular+SIM &gt; SIM</b> settings page. Select <b>No</b> to show the <b>LTE attach APN</b> button.
HideMMSAPN	Select <b>Yes</b> to hide the <b>add mms apn</b> button on the <b>Settings &gt; Cellular+SIM &gt; SIM</b> settings page. Select <b>No</b> to show the <b>add mms apn</b> button.
HideMMSAPNIPType	Select <b>Yes</b> to hide the APN IP type selector on the MMS APN page. Select <b>No</b> to show the APN IP type selector.
HideModeSelection	Select <b>Yes</b> to hide the <b>Network Mode selection</b> drop-down menu on the <b>Settings &gt; Cellular+SIM &gt; SIM</b> settings page. Select <b>No</b> to show the <b>Network Mode selection</b> .
HidePersoUnlock	Select <b>Yes</b> to hide the Perso unlock UI. Select <b>No</b> to show the Perso unlock UI.
HighestSpeed2G	You can customize the listed names of the connection speeds with their own character codes. To modify "2G" to another character code, change the value of HighestSpeed2G. Although there is no limit to the number of characters you can use, if the character code is too long, it will be truncated in the UI.
HighestSpeed3G	You can customize the listed names of the connection speeds with their own character codes. To modify "3G" to another character code, change the value of HighestSpeed3G. Although there is no limit to the number of characters you can use, if the character code is too long, it will be truncated in the UI.
HighestSpeed3GOnly	You can customize the listed names of the connection speeds with their own character codes. To modify "3G Only" to another character code, change the value of HighestSpeed3GOnly. Although there is no limit to the number of characters you can use, if the character code is too long, it will be truncated in the UI.
HighestSpeed3GPreferred	You can customize the listed names of the connection speeds with their own character codes. To modify "3G Preferred" to another character code, change the value of HighestSpeed3GPreferred. Although there is no limit to the number of characters you can use, if the character code is too long, it will be truncated in the UI.
HighestSpeed4G	You can customize the listed names of the connection speeds with their own character codes. To modify "4G" to another character code, change the value of HighestSpeed4G. Although there is no limit to the number of characters you can use, if the character code is too long, it will be truncated in the UI.

Setting	Description
HighestSpeed4G3GOnly	You can customize the listed names of the connection speeds with their own character codes. To modify "4G or 3G Only" to another character code, change the value of HighestSpeed4G3GOnly. Although there is no limit to the number of characters you can use, if the character code is too long, it will be truncated in the UI.
HighestSpeed4GOnly	You can customize the listed names of the connection speeds with their own character codes. To modify "4G Only" to another character code, change the value of HighestSpeed4GOnly. Although there is no limit to the number of characters you can use, if the character code is too long, it will be truncated in the UI.
HighestSpeedTitle	You can customize the <b>Highest connection speed</b> drop-down label in the <b>Settings &gt; Cellular+SIM &gt; SIM</b> settings page. To change the Highest connection speed drop-down label, set HighestSpeedTitle to another string. For example, you can set this to "Preferred connection speed".
IsATTSpecific	Control the roaming text for AT&T devices. AT&T requires the phone to show a particular roaming text to meet their legal and marketing guidelines. By default, if the user chooses <b>roam</b> under <b>Data roaming options</b> in the <b>Settings &gt; Cellular+SIM</b> screen, they will see the following text: <i>Depending on your service agreement, you might pay more when using data roaming.</i> If you set IsATTSpecific to <b>Yes</b> , the following roaming text will be displayed instead: <i>International data roaming charges apply for data usage outside the United States, Puerto Rico, and United States Virgin Islands. Don't allow roaming to avoid international data roaming charges.</i>
LTEAttachGUID	Set the value for LTEAttachGuid to the OemConnectionId GUID used for the LTE attach profile in the modem. The value is a GUID in the string format XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXX.
MMSAPNIPTypeIfHidden	Select between <b>IPV4</b> , <b>IPV6</b> , <b>IPV4V6</b> , and <b>IPV4V6XLAT</b> for default MMS APN IP type.

SETTING	DESCRIPTION
ShowExtendedRejectCodes	<p>When a reject code is sent by the network, partners can specify that extended error messages should be displayed instead of the standard simple error messages. This customization is only intended for use when required by the mobile operator's network. The short versions of the extended reject message are shown in the following screens:</p> <ul style="list-style-type: none"> <li>- Phone tile in Start</li> <li>- Call History screen</li> <li>- Dialer</li> <li>- Call Progress screen</li> <li>- Incoming Call screen</li> <li>- As the status string under Settings &gt; cellular+SIM</li> </ul> <p>The long version of the extended reject message is shown under the Active Network label in <b>Settings &gt; cellular+SIM</b>. Select <b>Yes</b> to show the extended error message. Select <b>No</b> to hide the extended error message. See <a href="#">Error messages for reject codes</a> to see the versions of the message.</p>
ShowHighestSpeed3GPreferred	Select <b>Yes</b> to show the <b>3G Preferred</b> option in the <b>Highest connection speed</b> drop-down menu. Select <b>No</b> to hide <b>3G Preferred</b> .
ShowManualAvoidance	Select <b>Yes</b> to show the <b>Switch to next network manually</b> button in SIM settings when Mode Selection is CDMA on a C+G dual SIM phone. Select <b>No</b> to hide the <b>Switch to next network manually</b> button
ShowPreferredPLMNPage	Select <b>Yes</b> to show the preferred public land mobile network (PLMN) page in SIM settings.
ShowSpecificWifiCallingError	Select <b>Yes</b> to show a specific error message based on operator requirements.
ShowViewAPN	Select <b>Yes</b> to show the <b>View Internet APN</b> button in <b>Settings &gt; cellular+SIM</b> .
ShowWifiCallingEmergencyCallWarning	Select <b>Yes</b> to show Wi-Fi emergency call warning.
ShowWifiCallingError	Select <b>Yes</b> to show Wi-Fi calling error message.

## General

SETTING	DESCRIPTION
---------	-------------

Setting	Description
atomicRoamingTableSettings3GPP	<p>If you enable 3GPP roaming, configure the following settings:</p> <ul style="list-style-type: none"> <li>- <b>Exceptions</b> maps the SerialNumber key to the Exceptions value. The wildcard, \$(SerialNumber), is a 3-digit decimal serial number (000 through 999) represented as a string. The wildcard is used as a regvalue under the "Exceptions" subkey. Multiple reg values in this form may be configured or customized by the OEM, all placed under the same subkey (Exceptions). The data in the regvalue is a string representing an MCC-MNC pair, such as "410510" where 410 is the MCC and 510 is the MNC.</li> <li>- <b>HomePLMN</b> maps the SerialNumber key to the HomePLMN value. The wildcard, \$(SerialNumber), is a 3-digit decimal serial number (000 through 999) represented as a string. The wildcard is used as a regvalue under the "HomePLMN" subkey. Multiple reg values in this form may be configured or customized by the OEM, all placed under the same subkey (HomePLMN). The data in the regvalue is a string representing an MCC-MNC pair, such as "410510" where 410 is the MCC and 510 is the MNC.</li> <li>- <b>TargetImsi</b> maps the SerialNubmer key to the TargetImsi value. The wildcard, \$(SerialNumber), is a 3-digit decimal serial number (000 through 999) represented as a string. The wildcard is used as a regvalue under the "TargetImsi" subkey. Multiple reg values in this form may be configured or customized by the OEM, all placed under the same subkey (TargetImsi). The data in the regvalue is a string representing an MCC-MNC pair, such as "410510" where 410 is the MCC and 510 is the MNC.</li> </ul>
atomicRoamingTableSettings3GPP2	<p>If you enable 3GPP2 roaming, configure the following settings:</p> <ul style="list-style-type: none"> <li>- <b>Home</b> maps the SerialNumber key to the Home value. The wildcard, \$(SerialNumber), is a 3-digit decimal serial number (000 through 999) represented as a string. The wildcard is used as a regvalue under the "Home" subkey. Multiple reg values in this form may be configured or customized by the OEM, all placed under the same subkey (Home). The data in the regvalue is a DWORD representing the Roaming Indicator.</li> <li>- <b>Roaming</b> maps the SerialNumber key to the Roaming value. The wildcard, \$(SerialNumber), is a 3-digit decimal serial number (000 through 999) represented as a string. The wildcard is used as a regvalue under the "Roaming" subkey. Multiple reg values in this form may be configured or customized by the OEM, all placed under the same subkey (Roaming). The data in the regvalue is a DWORD representing the Roaming Indicator.</li> </ul>
AvoidStayingInManualSelection	<p>You can enable permanent automatic mode for mobile networks that require the cellular settings to revert to automatic network selection after the user has manually selected another network when roaming or out of range of the home network.</p>
CardAllowList	<p>Define the list of SIM cards allowed in the first slot of a C+G dual SIM phone. This setting is used only if <b>CardLock</b> is set to allow it. If <b>CardLock</b> is not set, this list is ignored. To configure the list of SIM cards allowed in the first slot, set the value for CardAllowList to a comma-separated MCC:MNC list. You can also use wild cards, represented by an asterisk (*), to accept any value. For example, you can set the value to `310:410,311:,404:012,310:70`.</p>

SETTING	DESCRIPTION
CardBlockList	Define the list of SIM cards that are not allowed in the first slot of a C+G dual SIM phone. This setting is used only if <b>CardLock</b> is set to allow it. If <b>CardLock</b> is not set, this list is ignored. To configure the list of SIM cards that are not allowed in the first slot, set the value for CardBlockList to a comma separated MCC:MNC list. You can also use wild cards, represented by an asterisk (*), to accept any value. For example, you can set the value to '310:410,311;404:012,310:70'.
CardLock	Used to enforce either the card allow list or both the card allow and block lists on a C+G dual SIM phone.
Critical > MultivariantProvisionedSPN	Used to change the default friendly SIM names in dual SIM phones. By default, the OS displays SIM 1 or SIM 2 as the default friendly name for the SIM in slot 1 or slot 2 if the service provider name (SPN) or mobile operator name has not been set. Partners can use this setting to change the default name read from the SIM to define the SPN for SIM cards that do not contain this information or to generate the default friendly name for the SIM. The OS uses the default value as the display name for the SIM or SPN in the Start screen and other parts of the UI including the SIM settings screen. For dual SIM phones that contain SIMs from the same mobile operator, the names that appear in the UI may be similar. See <a href="#">Values for MultivariantProvisionedSPN</a> .
Critical > SimNameWithoutMSISDNEnabled	Use this setting to remove the trailing MSISDN digits from the service provider name (SPN) in the phone UI. By default, the OS appends the trailing MSISDN digits to the service provider name (SPN) in the phone UI, including on the phone and messaging apps. If required by mobile operators, OEMs can use the SimNameWithoutMSISDNEnabled setting to remove the trailing MSISDN digits. However, you must use this setting together with <b>MultivariantProvisionedSPN</b> to suppress the MSISDN digits.
DisableLTSupportWhenRoaming	Set to <b>Yes</b> to disable LTE support when roaming.
ExcludedSystemTypesByDefault	Set the default value for <b>Highest connection speed</b> in the <b>Settings &gt; Cellular &amp; SIM &gt; SIM</b> screen by specifying the bitmask for any combination of radio technology to be excluded from the default value. The connection speed that has not been excluded will show up as the highest connection speed. On dual SIM phones that only support up to 3G connection speeds, the <b>Highest connection speed</b> option is replaced by a 3G on/off toggle based on the per-device setting. Enter the binary setting to exclude 4G ( <code>10000</code> ) or 3G ( <code>01000</code> ).
LTEEnabled	Select <b>Yes</b> to enable LTE, and <b>No</b> to disable LTE.
LTEForced	Select <b>Yes</b> to force LTE.

Setting	Description
NetworkSuffix	<p>To meet branding requirements for some mobile operators, you can add a suffix to the network name that is displayed on the phone. For example, you can change from ABC to ABC 3G when under 3G coverage. This feature can be applied for any radio access technology (RAT). For TD-SCDMA RAT, a 3G suffix is always appended by default, but partners can also customize this the same way as with any other RAT. In the setting name, set SYSTEMTYPE to the network type that you want to append the network name to and click <b>Add</b>:</p> <ul style="list-style-type: none"> <li>- system type 4: 2G (GSM)</li> <li>- system type 8: 3G (UMTS)</li> <li>- system type 16: LTE</li> <li>- system type 32: 3G (TS-SCDMA)</li> </ul> <p>Select the system type that you added, and enter the network name and suffix that you want displayed.</p>
NitzFiltering	<p>For mobile networks that can receive Network Identity and Time Zone (NITZ) information from multiple sources, partners can set the phone to ignore the time received from an LTE network. Time received from a CDMA network is not affected. Set the value of NitzFiltering to <code>0x10</code>.</p>
OperatorListForExcludedSystemTypes	<p>Enter a comma-separated list of MCC and MNC (MCC:MNC) for which system types should be restricted. For mobile operators that require more control over the system types that their phones use to connect to the mobile operators' networks, OEMs can specify the MCC and MNC of other specific operators that the main mobile operator wishes to limit. If the UICC's MCC and MNC matches any of the pairs that OEMs can specify for the operator, a specified RIL system type will be removed from the UICC regardless of its app types, slot position, or executor mapping. This setting is used only for China. OEMs should not use this setting unless required by the mobile operator. Set the value of the OperatorListForExcludedSystemTypes setting a comma separated list of MCC:MNC pairs for which the system types should be restricted. For example, the value can be set to 310:026,310:030 to restrict operators with an MCC:MNC of 310:026 and 310:030.</p>
OperatorPreferredForFasterRadio	<p>Set Issuer Identification Number (IIN) or partial ICCID of preferred operator for the faster radio. For mobile operators that require more control over the system types that their phones use to connect to the mobile operators' networks, OEMs can map a partial ICCID or an Industry Identification Number (IIN) to the faster radio regardless of which SIM card is chosen for data connectivity. This setting is used only for China. OEMs should not use this setting unless required by the mobile operator. To map a partial ICCID or an IIN to the faster radio regardless of which SIM card is chosen for data connectivity, set the value of OperatorPreferredForFasterRadio to match the IIN or the ICCID, up to 7 digits, of the preferred operator.</p>
SuggestDataRoamingARD	<p>Use to show the data roaming suggestion dialog when roaming and the data roaming setting is set to no roaming.</p>

## RCS

See descriptions in Windows Configuration Designer.

## SMS

SETTING	DESCRIPTION
AckExpirySeconds	Set the value, in seconds, for how long to wait for a client ACK before trying to deliver.
DefaultMCC	Set the default mobile country code (MCC).
Encodings > GSM7BitEncodingPage	Enter the code page value for the 7-bit GSM default alphabet encoding. Values: - Code page value: 55000 (Setting value: 0xD6D8)(Code page: default alphabet) - Code page value: 55001 (Setting value: 0xD6D9)(Code page: GSM with single shift for Spanish)- Code page value: 55002 (Setting value: 0xD6DA)(Code page: GSM with single shift for Portuguese)- Code page value: 55003 (Setting value: 0xD6DB) (Code page: GSM with single shift for Turkish)- Code page value: 55004 (Setting value: 0xD6DC)(Code page: SMS Greek Reduction)
Encodings > GSM8BitEncodingPage	Enter the code page value for GSM 8-bit encoding (OEM set). OEM-created code page IDs should be in the range 55050–55099. For more information, see [Add encoding extension tables for SMS] <a href="https://docs.microsoft.com/windows-hardware/customize/mobile/mcsf/add-encoding-extension-tables-for-sms">https://docs.microsoft.com/windows-hardware/customize/mobile/mcsf/add-encoding-extension-tables-for-sms</a> .
Encodings > OctetEncodingPage	Set the octet (binary) encoding.
Encodings > SendUDHNLSS	Set the 7 bit GSM shift table encoding.
Encodings > UseASCII	Set the 7 bit ASCII encoding. Used only for CDMA carriers that use 7-bit ASCII encoding instead of GSM 7-bit encoding.
Encodings > UseKeyboardLangague	Set whether to use the keyboard language (Portuguese, Spanish, or Turkish) based encoding (set shift table based on keyboard language).
IncompleteMsgDeliverySeconds	Set the value, in seconds, for long to wait for all parts of multisegment Sprint messages for concatenation.
MessageExpirySeconds	Partners can set the expiration time before the phone deletes the received parts of a long SMS message. For example, if the phone is waiting for a three-part SMS message and the first part has been received, the first part will be deleted when the time expires and the other part of the message has not arrived. If the second part of the message arrives before the time expires, the first and second parts of the message will be deleted if the last part does not arrive after the time expires. The expiration time is reset whenever the next part of the long message is received. Set MessageExpirySeconds to the number seconds that the phone should wait before deleting the received parts of a long SMS messages. This value should be in hexadecimal and must be prefixed with 0x. The default value is 0x15180, which is equivalent to 1 day or 86,400 seconds.

SETTING	DESCRIPTION
SmsFragmentLimit	Partners can specify a maximum length for SMS messages. This requires setting both the maximum number of SMS fragments per SMS message, from 1 to 255, and the maximum size in bytes of each SMS fragment, from 16 to 140 bytes. Use SmsFragmentLimit to set the maximum number of bytes in the user data body of an SMS message. You must set the value between 16 (0x10) and 140 (0x8C). You must also use SmsPageLimit to set the maximum number of segments in a concatenated SMS message.
SmsPageLimit	Partners can specify a maximum length for SMS messages. This requires setting both the maximum number of SMS fragments per SMS message, from 1 to 255, and the maximum size in bytes of each SMS fragment, from 16 to 140 bytes. Use SmsPageLimit to set the maximum number of segments in a concatenated SMS message. You must set the value to 255 (0xFF) or smaller. You must also use SmsFragmentLimit to set the maximum number of bytes in the body of the SMS message.
SprintFragmentInfoInBody	Partners can enable the messaging client to allow users to enter more than 160 characters per message. Messages longer than 160 characters are sent as multiple SMS messages that contain a tag at the beginning of the message in the form "(1/2)", where the first number represents the segment or part number and the second number represents the total number of segments or parts. Multiple messages are limited to 6 total segments. When enabled, the user cannot enter more characters after the 6 total segments limit is reached. Any message received with tags at the beginning is recombined with its corresponding segments and shown as one composite message.
Type3GPP > ErrorHandling > ErrorType	Enter a name for ERRORCODE3GPP, and click <b>Add</b> . Configure the error type that you added as <b>Transient Failure</b> or <b>Permanent Failure</b> .
Type3GPP > ErrorHandling > FriendlyErrorClass	Enter a name for ERRORCODE3GPP, and click <b>Add</b> . Configure the error class that you added as <b>generic error</b> , <b>invalid recipient address</b> , or <b>network connectivity trouble</b> .
Type3GPP > IMS > SmsUse16BitReferenceNumbers	Configure whether to use 8-bit or 16-bit message ID (reference number) in the UDH.
Type3GPP2 > ErrorHandling > FriendlyErrorClass	Enter a name for ERRORCODE3GPP2, and click <b>Add</b> . Configure the error class that you added as <b>generic error</b> , <b>invalid recipient address</b> , or <b>network connectivity trouble</b> .
Type3GPP2 > ErrorHandling > UseReservedAsPermanent	Set the 3GPP2 permanent error type.

## UTK

SETTING	DESCRIPTION
---------	-------------

SETTING	DESCRIPTION
UIDefaultDuration	Specifies the default time, in milliseconds, that the DISPLAY TEXT, GET INKEY, PLAY TONE, or SELECT ITEM dialog should be displayed. The default value is 60000 milliseconds (60 seconds). The valid value range is 1-120000.
UIGetInputDuration	Specifies the default time, in milliseconds, that the GET INPUT dialog should be displayed. The default value is 120000 milliseconds (120 seconds). The valid value range is 1-120000.

## VoLTE

SETTING	DESCRIPTION
IMSOMADMServices	Allows configuration of OMA DM Services Mask. The value is mapped directly to RIL_IMS_NW_ENABLED_FLAGS on the modem side. To configure the OMA DM services mask, set the IMSOMADMServices setting to one of the following values: <ul style="list-style-type: none"> <li>- None, Flag: 0, Bitmask: 00000</li> <li>- OMA DM, Flag: 1, Bitmask: 00001</li> <li>- Voice, Flag: 2, Bitmask: 00010</li> <li>- Video, Flag: 4, Bitmask: 00100</li> <li>- EAB presence, Flag: 8, Bitmask: 01000</li> <li>- Enable all services, Flag: 15, Bitmask: 10000</li> </ul>
IMSServices	Identifies which IMS services are enabled (if any). The value is any combination of flags 1 (IMS), 2 (SMS over IMS), 4 (Voice over IMS) and 8 (Video Over IMS). Set the value for the IMSServices setting to any combination of the following flags or bitmasks: <ul style="list-style-type: none"> <li>- IMS, Flag: 1, Bitmask: 0001</li> <li>- SMS over IMS, Flag: 2, Bitmask: 0010</li> <li>- Voice over IMS, Flag: 4, Bitmask: 0100</li> <li>- Video over IMS, Flag: 8, Bitmask: 1000</li> </ul>

## Error messages for reject codes

REJECT CODE	EXTENDED ERROR MESSAGE	SHORT ERROR MESSAGE
2 (The SIM card hasn't been activated or has been deactivated)	SIM not set up MM#2	Invalid SIM
3 (The SIM card fails authentication or one of the identity check procedures. This can also happen due to a duplication of the TMSI across different MSCs.)	Can't verify SIM MM#3	Invalid SIM
6 (The device has been put on a block list, such as when the phone has been stolen or the IMEI is restricted.)	Phone not allowed MM#6	No service

## Values for MultivariantProvisionedSPN

Set the MultivariantProvisionedSPN value to the name of the SPN or mobile operator.

The following table shows the scenarios supported by this customization:

#### **NOTE**

In the Default SIM name column:

- The " " in MultivariantProvisionedSPN" "1234 means that there is a space between the mobile operator name or SPN and the last 4 digits of the MSISDN.
- MultivariantProvisionedSPN means the value that you set for the MultivariantProvisionedSPN setting.
- SIM 1 or SIM 2 is the default friendly name for the SIM in slot 1 or slot 2.

Multivariant setting set?|SPN provisioned?|MSISDN (last 4 digits: 1234, for example) provisioned?|Default SIM

name Yes|Yes|Yes|*MultivariantProvisionedSPN*1234 or *MultivariantProvisionedSPN*" "1234

Yes|No|No|*MultivariantProvisionedSPN* (up to 16 characters) Yes|Yes|No|*MultivariantProvisionedSPN* (up to 16 characters) Yes|No|Yes|*MultivariantProvisionedSPN*1234 or *MultivariantProvisionedSPN*" "1234 No|Yes|Yes|If SPN string >= 12: *SPN*1234

If SPN string < 12: *SPN*" "1234 No|No|No|SIM 1 or SIM 2 No|Yes|No|SPN (up to 16 characters) No|No|Yes|SIM 1 or SIM 2

# Cellular (Windows Configuration Designer reference)

10/17/2017 • 1 min to read • [Edit Online](#)

Use to configure settings for cellular connections.

## IMPORTANT

These settings are intended to be used only by manufacturers, mobile operators, and solution providers when configuring devices, and are not intended for use by administrators in the enterprise.

## Applies to

SETTING GROUPS	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
All settings	X				

To begin, enter a SIM integrated circuit card identifier (**SimIccid**), and click **Add**. In the **Customizations** pane, select the SimIccid that you just entered and configure the following settings for it.

## AccountExperienceURL

Enter the URL for the mobile operator's web page.

## AppID

Enter the AppID for the mobile operator's app in Microsoft Store.

## BrandingIcon

Browse to and select an .ico file.

## BrandingIconPath

Enter the destination path for the BrandingIcon .ico file.

## BrandingName

Enter the service provider name for the mobile operator.

## NetworkBlockList

Enter a comma-separated list of mobile country code (MCC) and mobile network code (MCC) pairs (MCC:MNC).

## SIMBlockList

Enter a comma-separated list of mobile country code (MCC) and mobile network code (MCC) pairs (MCC:MNC).

## UseBrandingNameOnRoaming

Select an option for displaying the BrandingName when the device is roaming.

# Certificates (Windows Configuration Designer reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Use to deploy Root Certificate Authority (CA) certificates to devices. The following list describes the purpose of each setting group.

- In [CACertificates](#), you specify a certificate that will be added to the Intermediate CA store on the target device.
- In [ClientCertificates](#), you specify a certificate that will be added to the Personal store on the target device, and provide (password, keylocation), (and configure whether the certificate can be exported).
- In [RootCertificates](#), you specify a certificate that will be added to the Trusted Root CA store on the target device.
- In [TrustedPeopleCertificates](#), you specify a certificate that will be added to the Trusted People store on the target device.
- In [TrustedProvisioners](#), you specify a certificate which allows devices to automatically trust packages from the specified publisher.

## Applies to

SETTING GROUPS	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
All setting groups	X	X	X	X	X

## CACertificates

1. In **Available customizations**, select **CACertificates**, enter a friendly name for the certificate, and then click **Add**.
2. In **Available customizations**, select the name that you just created.
3. In **CertificatePath**, browse to or enter the path to the certificate.

## ClientCertificates

1. In **Available customizations**, select **ClientCertificates**, enter a friendly name for the certificate, and then click **Add**.
2. In **Available customizations**, select the name that you just created. The following table describes the settings you can configure. Settings in **bold** are required.

SETTING	VALUE	DESCRIPTION
<b>CertificatePassword</b>		
<b>CertificatePath</b>		Adds the selected certificate to the Personal store on the target device.
ExportCertificate	True or false	Set to <b>True</b> to allow certificate export.

SETTING	VALUE	DESCRIPTION
<b>KeyLocation</b>	- TPM only - TPM with software fallback - Software only	

## RootCertificates

1. In **Available customizations**, select **RootCertificates**, enter a friendly name for the certificate, and then click **Add**.
2. In **Available customizations**, select the name that you just created.
3. In **CertificatePath**, browse to or enter the path to the certificate.

## TrustedPeopleCertificates

1. In **Available customizations**, select **TrustedPeopleCertificates**, enter a friendly name for the certificate, and then click **Add**.
2. In **Available customizations**, select the name that you just created.
3. In **TrustedCertificate**, browse to or enter the path to the certificate.

## TrustedProvisioners

1. In **Available customizations**, select **TrustedProvisioners**, enter a CertificateHash, and then click **Add**.
2. In **Available customizations**, select the name that you just created.
3. In **TrustedProvisioner**, browse to or enter the path to the certificate.

## Related topics

- [RootCATrustedCertificates configuration service provider \(CSP\)](#)

# CleanPC (Windows Configuration Designer reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Use to remove user-installed and pre-installed applications, with the option to persist user data.

## Applies to

SETTINGS	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
CleanPCRetainingUserData	X				
CleanPCWithoutR etainingUserData	X				

For each setting, the options are **Enable** and **Not configured**.

## Related topics

- [CleanPC configuration service provider \(CSP\)](#)

# Connections (Windows Configuration Designer reference)

10/17/2017 • 1 min to read • [Edit Online](#)

Use to configure settings related to various types of phone connections.

## Applies to

SETTING GROUPS	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
All settings	X	X	X	X	

For each setting group:

1. In **Available customizations**, select the setting group (such as **Cellular**), enter a friendly name for the connection, and then click **Add**.
2. In **Available customizations**, select the name that you just created.

## Cellular

See [CM\\_CellularEntries configuration service provider \(CSP\)](#) for settings and values.

## EnterpriseAPN

See [Configure cellular settings for tablets and PCs](#) and [EnterpriseAPN CSP](#) for settings and values.

## General

Use **General > DataRoam** to set the default value for the **Default roaming options** option in the **Settings > cellular + SIM** area on the device. Select between **DoNotRoam**, **DomesticRoaming**, or **InternationalRoaming**.

## Policies

See [CMPolicy CSP](#) for settings and values.

## Proxies

See [CM\\_ProxyEntries CSP](#) for settings and values.

# ConnectivityProfiles (Windows Configuration Designer reference)

9/6/2017 • 7 min to read • [Edit Online](#)

Use to configure profiles that a user will connect with, such as an email account or VPN profile.

## Applies to

SETTING GROUPS	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
Email	X	X	X	X	X
Exchange	X	X	X	X	X
KnownAccounts	X	X	X	X	X
VPN	X	X	X	X	X
WiFiSense	X	X	X	X	X
WLAN	X	X	X	X	X

## Email

Specify an email account to be automatically set up on the device.

1. In **Available customizations**, select **Email**, enter a friendly name for the account, and then click **Add**.
2. In **Available customizations**, select the name that you just created. The following table describes the settings you can configure for each account. Settings in **bold** are required.

SETTING	DESCRIPTION
<b>AccountType</b>	Select between <b>Normal email</b> and <b>Visual voice mail</b>
AuthForOutgoingMail	Set to <b>True</b> if the outgoing server requires authentication
Domain	Enter the domain for the account
HaveAlternateCredentialsForSMTP	Specify whether the user's alternate SMTP account is enabled. If enabled, configure the <b>SMTPDomain</b> , <b>SMTPName</b> , and <b>SMTPPassword</b> settings
InboxUpdateFrequency	Specify the time between email send/receive updates, in minutes. Available values are: <ul style="list-style-type: none"><li>- Manual update</li><li>- Every 2 hours</li><li>- Every 15 minutes</li><li>- Every 30 minutes</li><li>- Every hour</li></ul>
<b>IncomingMailServerName</b>	Enter the name of the messaging service's incoming email server
<b>OutgoingServerName</b>	Enter the name of the messaging service's outgoing mail server

SETTING	DESCRIPTION
Password	Enter the password for the account
ReplyAddress	Enter the reply address for the account
SenderName	Enter the name of the sender for the account
<b>ServiceName</b>	Enter the name of the email service
<b>ServiceType</b>	Select <b>IMAP4</b> or <b>POP3</b> for service type
SMTPDomain	Enter the domain name for the user's alternate SMTP account, if <b>HaveAlternateCredentialsForSMTP</b> is enabled
SMTPName	Enter the display name associated with the user's alternate SMTP account, if <b>HaveAlternateCredentialsForSMTP</b> is enabled
SMTPPassword	Enter the password for the user's alternate SMTP account, if <b>HaveAlternateCredentialsForSMTP</b> is enabled
SSLIncoming	Specify whether the incoming email server uses SSL
SSLOutgoing	Specify whether the outgoing email server uses SSL
SyncOptions	Specify how many days' worth of emails should be downloaded from the server. Available values are: - All mail - Two weeks - One month - One week
<b>UserName</b>	Enter the user name for the account

## Exchange

Configure settings related to Exchange email server. These settings are related to the [ActiveSync configuration service provider \(CSP\)](#).

- In **Available customizations**, select **Exchange**, enter a name for the account, and then click **Add**. A globally unique identifier (GUID) is generated for the account.
- In **Available customizations**, select the GUID that you just created. The following table describes the settings you can configure. Settings in **bold** are required.

SETTING	DESCRIPTION
AccountIcon	Specify the location of the icon associated with the account. The account icon can be used as a tile in the Start list or as an icon in the applications list under <b>Settings &gt; Email &amp; accounts</b> . Some icons are already provided on the device. The suggested icon for POP/IMAP or generic ActiveSync accounts is at <code>res://AccountSettingsSharedRes{ScreenResolution}!%s.genericmail.png</code> . The suggested icon for Exchange Accounts is at <code>res://AccountSettingsSharedRes{ScreenResolution}!%s.office.outlook.png</code> . Custom icons can be added if desired.
<b>AccountName</b>	Enter the name that refers to the account on the device
<b>AccountType</b>	Select <b>Exchange</b>

SETTING	DESCRIPTION
<b>DiagnosticLogging</b>	Select whether to disable logging, enable basic logging, or enable advanced logging
Domain	Enter the domain name of the Exchange server
<b>EmailAddress</b>	Enter the email address associated with the Exchange ActiveSync account.
<b>MailAgeFilter</b>	Specify the time window used for syncing email items to the device. Available values are: - All email is synced - Only email up to three days old is synced - Email up to a week old is synced (default) - Email up to two weeks old is synced - Email up to a month old is synced
<b>Password</b>	Enter the password for the account
<b>Schedule</b>	Specify the time until the next sync is performed, in minutes. Available values are: - As items are received (default) - Sync manually - Every 15 minutes - Every 30 minutes - Every 60 minutes
<b>ServerName</b>	Enter the server name used by the account
SyncCalendar_Enable	Enable or disable calendar sync
SyncCalendar_Name	If you enable calendar sync, enter <b>Calendar</b>
SyncContacts_Enable	Enable or disable contacts sync
SyncContacts_Name	If you enable contacts sync, enter <b>Contacts</b>
SyncEmail_Enable	Enable or disable email sync
SyncEmail_Name	If you enable email sync, enter <b>Email</b>
SyncTasks_Enable	Enable or disable tasks sync
SyncTasks_Name	If you enable tasks sync, enter <b>Tasks</b>
<b>UserName</b>	Enter the user name for the account
UseSSL	Specify whether to use Secure Sockets Layer (SSL)

## KnownAccounts

Configure the settings to add additional email accounts.

SETTING	DESCRIPTION
KnownAccountsOEM	Enter the source or file location of the KnownAccountsOEM.xml file on your development workstation.

SETTING	DESCRIPTION
OemFilePath	Enter the name of the XML file that defines the new account to be added. The name must be KnownAccountsOEM.xml.

## VPN

Configure settings to change the default maximum transmission unit ([MTU](#)) size settings for Point-to-Point Protocol (PPP) connections or for virtual private network (VPN) connections, or to create a [VPN profile](#).

### MTU

SETTING	DESCRIPTION
PPPProtocolType	Select <b>VPNPPPProtocolType</b>
ProtocolType	Select <b>VPNProtocolType</b>
TunnelMTU	Enter the desired MTU size, between <b>1</b> and <b>1500</b>

### VPN

1. In **Available customizations**, select **VPNSetting**, enter a friendly name for the account, and then click **Add**.
2. In **Available customizations**, select the name that you just created. The following table describes the settings you can configure. Settings in **bold** are required.

SETTING	DESCRIPTION
<b>ProfileType</b>	Choose between <b>Native</b> and <b>Third Party</b>
RememberCredentials	Select whether credentials should be cached
AlwaysOn	Set to <b>True</b> to automatically connect the VPN at sign-in
LockDown	When set to <b>True</b> : - Profile automatically becomes an "always on" profile - VPN cannot be disconnected - If the profile is not connected, the user has no network connectivity - No other profiles can be connected or modified
ByPassForLocal	When set to <b>True</b> , requests to local resources on the same Wi-Fi network as the VPN client can bypass VPN
DnsSuffix	Enter one or more comma-separated DNS suffixes. The first suffix listed is used as the primary connection-specific DNS suffix for the VPN interface. The list is added to the SuffixSearchList.
TrustedNetworkDetection	Enter a comma-separated string to identify the trusted network. VPN will not connect automatically when the user is on their corporate wireless network where protected resources are directly accessible to the device.
Proxy	Configure to <b>Automatic</b> or <b>Manual</b>
ProxyAutoConfigUrl	When <b>Proxy</b> is set to <b>Automatic</b> , enter the URL to automatically retrieve the proxy settings
ProxyServer	When <b>Proxy</b> is set to <b>Manual</b> , enter the proxy server address as a fully qualified hostname or enter <b>IP address:Port</b>

# WiFiSense

Configure settings related to Wi-Fi Sense.

## Config

The **Config** settings are initial settings that can be overwritten when settings are pushed to the device by the cloud.

SETTING	DESCRIPTION
WiFiSharingFacebookInitial	Enable or disable sharing of Wi-Fi networks with Facebook contacts
WiFiSharingOutlookInitial	Enable or disable sharing of Wi-Fi networks with Outlook contacts
WiFiSharingSkypeInitial	Enable or disable sharing of Wi-Fi networks with Skype contacts

## FirstBoot

SETTING	DESCRIPTION
DefaultAutoConnectOpenState	When enabled, the OOBW Wi-Fi Sense checkbox to automatically connect to open networks will be checked.
DefaultAutoConnectSharedState	When enabled, the OOBW Wi-Fi Sense checkbox to share networks with contacts will be checked.
WiFiSenseAllowed	Enable or disable Wi-Fi Sense. Wi-Fi Sense features include auto-connect to Wi-Fi hotspots and credential sharing.

## SystemCapabilities

You can use these settings to configure system capabilities for Wi-Fi adapters, which is a new functionality in Windows 10. These system capabilities are added at image time to ensure that the information is at its most accurate. The capabilities allow the OS to have a better understanding of the underlying hardware that it's running on. Telemetry data is generated by the system to provide data that can be used to diagnose both software and hardware issues.

SETTING	DESCRIPTION
CoexistenceSupport	Specify the type of co-existence that's supported on the device: - <b>Both</b> : Both Wi-Fi and Bluetooth work at the same performance level during co-existence - <b>Wi-Fi reduced</b> : On a 2X2 system, Wi-Fi performance is reduced to 1X1 level - <b>Bluetooth centered</b> : When co-existing, Bluetooth has priority and restricts Wi-Fi performance - <b>One</b> : Either Wi-Fi or Bluetooth will stop working
NumAntennaConnected	Enter the number of antennas that are connected to the WLAN radio
SimultaneousMultiChannelSupported	Enter the maximum number of channels that the Wi-Fi device can simultaneously operate on. For example, you can use this to specify support for Station mode and Wi-Fi Direct GO on separate channels simultaneously.
WLANFunctionLevelDeviceResetSupported	Select whether the device supports functional level device reset (FLDR). The FLDR feature in the OS checks this system capability exclusively to determine if it can run.
WLANPlatformLevelDeviceResetSupported	Select whether the device supports platform level device reset (PLDR). The PLDR feature in the OS checks this system capability exclusively to determine if it can run.

# WLAN

Configure settings for wireless connectivity.

## Profiles

### To add a profile

1. Create [the wireless profile XML](#).
2. In **WLAN > Profiles**, browse to and select the profile XML file.
3. Click **Add**.

## WLANXmlSettings

Enter a SSID, click **Add**, and then configure the following settings for the SSID.

SETTINGS	DESCRIPTION
ProxyServerPort	(Optional) Specify the configuration of the network proxy as <b>host:port</b> . A proxy server host and port can be specified per connection for Windows 10 for mobile devices. The host can be server name, FQDN, or SLN or IPv4 or IPv6 address. This proxy configuration is only supported in Windows 10 for mobile devices. Using this configuration in Windows 10 for desktop editions will result in failure.
AutoConnect	(Optional) Select <b>True</b> or <b>false</b> to specify whether to automatically connect to WLAN.
HiddenNetwork	(Optional) Select <b>True</b> or <b>false</b> to specify whether the network is hidden.
SecurityType	Choose between <b>Open</b> , <b>WEP</b> , and <b>WPA2-Personal</b> . If you select <b>WEP</b> or <b>WPA2-Personal</b> , enter the <b>SecurityKey</b> required by the WLAN.

# CountryAndRegion (Windows Configuration Designer reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Use to configure a setting that partners must customize to ship Windows devices to specific countries/regions.

## Applies to

Setting Groups	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
CountryCodeForExtendedCapabilityPrompts	X	X	X	X	

You can set the **CountryCodeForExtendedCapabilityPrompts** setting for **China** to enable additional capability prompts when apps use privacy-sensitive features (such as Contacts or Microphone).

# DesktopBackgroundAndColors (Windows Configuration Designer reference)

10/17/2017 • 1 min to read • [Edit Online](#)

Do not use. Instead, use the [Personalization settings](#).

## Applies to

SETTING GROUPS	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
All settings	X				

# DeveloperSetup (Windows Configuration Designer reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Use to unlock developer mode on HoloLens devices and configure authentication to Windows Device Portal.

## Applies to

Setting Groups	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	Iot Core
<a href="#">EnableDeveloperMode</a>				X	
<a href="#">AuthenticationMode</a>				X	

## DeveloperSetupSettings: EnableDeveloperMode

When this setting is configured as **True**, the device is unlocked for developer functionality.

## WindowsDevicePortalSettings: Authentication Mode

When AuthenticationMode is set to **Basic Auth**, enter a user name and password to enable the device to connect to and authenticate with the Windows Device Portal.

## Related topics

- [Device Portal for HoloLens](#)

# DeviceFormFactor (Windows Configuration Designer reference)

9/6/2017 • 3 min to read • [Edit Online](#)

Use to identify the form factor of the device.

## Applies to

SETTING	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
DeviceForm	X	X	X	X	

Specifies the device form factor running Windows 10. Generally, the device form is set by the original equipment manufacturer (OEM), however you might want to change the device form based on its usage in your organization.

DeviceForm supports the following features or components:

- Cortana and Bing use the DeviceForm value to determine the accuracy of specific signals, such as location (GPS versus Wi-Fi versus reverse IP address lookup).
- Windows 10 features, such as Bluetooth and camera, may require DeviceForm to be accurately configured for full functionality.

Select the appropriate form from the dropdown menu.

DEVICE FORM	DESCRIPTION
Phone	A typical smartphone combines cellular connectivity, a touch screen, rechargeable power source, and other components into a single chassis.
LargeScreen	Microsoft Surface Hub
HMD	(Head-mounted display) A holographic computer that is completely untethered - no wires, phones, or connection to a PC needed.
IndustryHandheld	A device screen less than 7" diagonal designed for industrial solutions. May or may not have a cellular stack.
IndustryTablet	A device with an integrated screen greater than 7" diagonal and no attached keyboard designed for industrial solutions as opposed to consumer personal computer. May or may not have a cellular stack.
Banking	A machine at a bank branch or another location that enables customers to perform basic banking activities including withdrawing money and checking one's bank balance.

DEVICE FORM	DESCRIPTION
BuildingAutomation	A controller for industrial environments that can include the scheduling and automatic operation of certain systems such as conferencing, heating and air conditioning, and lighting.
DigitalSignage	A computer or playback device that's connected to a large digital screen and displays video or multimedia content for informational or advertising purposes.
Gaming	A device that's used for playing a game. It can be mechanical, electronic, or electromechanical equipment.
HomeAutomation	A controller that can include the scheduling and automatic operation of certain systems including heating and air conditioning, security, and lighting.
Industrial Automation	Computers that are used to automate manufacturing systems such as controlling an assembly line where each station is occupied by industrial robots.
Tablet	A device with an integrated screen that's less than 18". It combines a touch screen, rechargeable power source, and other components into a single chassis with an optional attachable keyboard.
Kiosk	An unattended structure that can include a keyboard and touch screen and provides a user interface to display interactive information and allow users to get more information.
MakerBoard	A low-cost and compact development board that's used for prototyping any number IoT-related things.
Medical	Devices built specifically to provide medical staff with information about the health and well-being of a patient.
Networking	A device or software that determines where messages, packets, and other signals will go next.
POS	(Point of Service) An electronic cash register or self-service checkout.
Printing	A printer, copy machine, or a combination of both.
ThinClient	A device that connects to a server to perform computing tasks as opposed to running apps locally.
Toy	A device used solely for enjoyment or entertainment.
Vending	A machine that dispenses items in exchange for payment in the form of coin, currency, or credit/debit card.
IndustryOther	A device that doesn't fit into any of the previous categories.

DEVICE FORM	DESCRIPTION
Desktop	A desktop PC form factor traditional comes in an upright tower or small desktop chassis and does not have an integrated screen.
Notebook	A notebook is a portable clamshell device with an attached keyboard that cannot be removed.
Convertible	A convertible device is an evolution of the traditional notebook where the keyboard can be swiveled, rotated or flipped, but not completely removed. It is a blend between a traditional notebook and tablet, also called a 2-in-1.
Detachable	A detachable device is an evolution of the traditional notebook where the keyboard can be completely removed. It is a blend between a traditional notebook and tablet, also called a 2-in-1.
AIO	An All-in-One (AIO) device is an evolution of the traditional desktop with an attached display.
Stick	A device that turns your TV into a Windows computer. Plug the stick into the HDMI slot on the TV and connect a USB or Bluetooth keyboard or mouse.
Puck	A small-size PC that users can use to plug in a monitor and keyboard.

# DeviceInfo (Windows Configuration Designer reference)

10/17/2017 • 1 min to read • [Edit Online](#)

Use to configure settings for DeviceInfo.

## IMPORTANT

These settings are intended to be used only by manufacturers, mobile operators, and solution providers when configuring devices, and are not intended for use by administrators in the enterprise.

## Applies to

SETTING GROUPS	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
All settings		X			

## PhoneMobileOperatorDisplayName

Enter a friendly name for the mobile operator. This string is displayed in the support section of the **Settings > About** screen and in the ringtone list.

## PhoneMobileOperatorName

This setting is used for targeting phone updates. It must contain a code specified by Microsoft that corresponds to the mobile operator. These codes are provided in [Registry values for mobile operator IDs](#). For open market phones, in which the mobile operator is not known, use the codes in [Registry values for carrier-unlocked phones](#) instead.

This string is not visible to the user.

This setting must not be changed over time even if the user switches SIMs or mobile operators, as updates are always targeted based on the first mobile operator associated with the phone.

The [PhoneManufacturer](#), [PhoneManufacturerModelName](#), and [PhoneMobileOperatorName](#) should create a unique Phone-Operator-Pairing (POP).

## PhoneOEMSupportLink

This should be a functional link that starts with `http://`. The link should be a URL that redirects to the mobile version of the web page. The content in the webpage should reflow to the screen width. This can be achieved by adding the CSS Tag `@-ms-viewport { width: device-width; }`.

The default is an empty string (""), which means that a support link will not be displayed to the user.

This setting varies by OEM.

## PhoneSupportLink

This should be a functional link that starts with `http://`. The link should be a URL that redirects to the mobile version

of the web page. The content in the webpage should reflow to the screen width. This can be achieved by adding the CSS Tag `[@-ms-viewport { width: device-width; }]`.

The default is an empty string (""), which means that a support link will not be displayed to the user.

This setting varies by OEM.

## PhoneSupportPhoneNumber

Use to specify the OEM or mobile operator's support contact phone number. The country code is not required. This string is displayed in the About screen in Settings. This setting also corresponds to the Genuine Windows Phone Certificates (GWPC) support number.

# DeviceManagement (Windows Configuration Designer reference)

10/17/2017 • 6 min to read • [Edit Online](#)

Use to configure device management settings.

## Applies to

Setting	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
Accounts	X	X	X	X	
PGList	X	X	X	X	
Policies	X	X	X	X	
TrustedProvisioningSource	X	X	X	X	

## Accounts

1. In **Available customizations**, select **Accounts**, enter a friendly name for the account, and then click **Add**.
2. In **Available customizations**, select the account that you just created. The following table describes the settings you can configure. Settings in **bold** are required.

Setting	Description
<b>Address</b>	Enter the OMA DM server address
<b>AddressType</b>	Choose between <b>IPv4</b> and <b>URI</b> for the type of OMA DM server address. The default value of <b>URI</b> specifies that the OMA DM account address is a URI address. A value of <b>IPv4</b> specifies that the OMA DM account address is an IP address.
<b>AppID</b>	Select <b>w7</b>

SETTING	DESCRIPTION
Authentication > Credentials	<p>1. Select a credentials level (CLCRED or SRVCRED). A value of <b>CLCRED</b> indicates that the credentials client will authenticate itself to the OMA DM server at the OMA DM protocol level. A value of <b>SRVCRED</b> indicates that the credentials server will authenticate itself to the OMA DM Client at the OMA DM protocol level.</p> <p>2. In <b>Available customizations</b>, select the level.</p> <p>3. For <b>Data</b>, enter the authentication nonce as a Base64 encoded string.</p> <p>4. For <b>Level</b>, select <b>CLCRED</b> or <b>SRVCRED</b>.</p> <p>5. For <b>Name</b>, enter the authentication name.</p> <p>6. For <b>Secret</b>, enter the password or secret used for authentication.</p> <p>7. For <b>Type</b>, select between <b>Basic</b>, <b>Digest</b>, and <b>HMAC</b>. For <b>CLCRED</b>, the supported values are <b>BASIC</b> and <b>DIGEST</b>. For <b>SRVCRED</b>, the supported value is <b>DIGEST</b>.</p>
AuthenticationPreference	Select between <b>Basic</b> , <b>Digest</b> , and <b>HMAC</b>
BackCompatRetryDisabled	Specify whether to retry resending a package with an older protocol version (for example, 1.1) in the SyncHdr on subsequent attempts (not including the first time). The default value of "FALSE" indicates that backward-compatible retries are enabled. A value of "TRUE" indicates that backward-compatible retries are disabled.
ConnectionRetries	Enter a number to specify how many retries the DM client performs when there are Connection Manager-level or wininet-level errors. The default value is <input type="text" value="3"/> .
CRLCheck	Specify whether a CRL Check should be performed. Allows connection to the DM server to check the Certificate Revocation List (CRL). Set to <b>True</b> to enable SSL revocation.
DefaultEncoding	Select whether the OMA DM client will use <b>WBXML</b> or <b>XML</b> for the DM package when communicating with the server
DisableOnRoaming	Specify whether the client will connect while cellular roaming
InitialBackOffTime	Specify the initial amount of time (in milliseconds) that the DM client waits before attempting a connection retry
InitiateSession	Specify whether a session should be started with the MDM server when the account is provisioned
MaxBackOffTime	Specify the maximum number of milliseconds to wait before attempting a connection retry
Name	Enter a display name for the management server
Port	Enter the OMA DM server port
PrefConRef	Enter a URI to NAP management object or a connection GUID used by the device Connection Manager

SETTING	DESCRIPTION
ProtocolVersion	Select between <b>1.1</b> and <b>1.2</b> for the OMA DM protocol version that the server supports
<b>Role</b>	Select between <b>Enterprise</b> and <b>Mobile Operator</b> for the role mask that the DM session runs with when it communicates with the server
<b>ServerID</b>	Enter the OMA DM server's unique identifier for the current OMA DM account
SSLClientCertSearchCriteria	Specify the client certificate search criteria, by subject attribute and certificate stores. For details, see <a href="#">DMAcc configuration service provider (CSP)</a> .
UseHardwareDeviceID	Specify whether to use the hardware ID for the ./DevInfo/DevID parameter in the DM account to identify the device
UseNonceResync	Specify whether the OMA DM client should use the nonce resynchronization procedure if the server trigger notification fails authentication

## PGList

1. In **Available customizations**, select **PGList**, enter a LogicalProxyName, and then click **Add**.
2. In **Available customizations**, select the LogicalProxyName that you just created, and then select **PhysicalProxies**.
3. Enter a PhysicalProxyName, and then click **Add**. The following table describes the settings you can configure for the physical proxy and for **Trust**.

SETTING	DESCRIPTION
Address	Enter the address of the physical proxy
AddressType	Select between <b>E164</b> , <b>IPV4</b> , and <b>IPV^</b> for the format and protocol of the PXADDR element for a physical proxy
MatchedNapID	Enter a string that defines the SMS bearer. This string must match the NAPID exactly. The value must contains MVID macro if it is an IPv4 PXADDRTYPE.
PushEnabled	Select whether push operations are enabled
Trust	Specify whether or not the physical proxies in this logical proxy are privileged

## Policies

The following table describes the settings you can configure for **Policies**.

Setting	Description
MMS > MMSMessageRoles	<p>Select between <b>SECROLE_KNOWN_PPG</b>, <b>SECROLE_ANY_PUSH_SOURCE</b>, and <b>SECROLE_KNOWN_PPG_OR_SECROLE_ANY_PUSH_SOURCE</b>. If a message contains at least one of the roles in the selected role mask, then the message is processed.</p>
OMACP > NetwpinRoles	<p>Select a policy role to specify whether OMA network PIN-signed messages will be accepted. OMA Client Provisioning Network PIN policy determines whether the OMA network PIN signed message will be accepted. The message's role mask and the policy's role mask are combined using the AND operator. If the result is non-zero, then the message is accepted.</p> <p>Available roles are: <b>SECROLE_OPERATOR_TIPS</b>, <b>SECROLE_KNOWN_PPG</b>, <b>SECROLE_OPERATOR_TPS_OR_SECROLE_KNOWN_PPG</b>, <b>SECROLE_ANY_PUSH_SOURCE</b>, <b>SECROLE_OPERATOR_TPS_OR_SECROLE_ANY_PUSH_SOURCE</b>, <b>SECROLE_KNOWN_PPG_OR_SECROLE_ANY_PUSH_SOURCE</b>, and <b>SECROLE_OPERATOR_TPS_OR_SECROLE_KNOWN_PPG_OR_SECROLE_ANY_PUSH_SOURCE</b>.</p> <p><b>Note</b> IMSI-based NETWPIN and USERNETWPIN may not work for dual SIM phones. The OMA-CP authentication provider only uses the IMSI from executor 0 (the current, active data SIM) when hashing these messages. OMA-CP payloads targeting executor 1 are rejected by the phone. For more information about executors, see Dual SIM.</p>
OMACP > UsernetwpinRoles	<p>Select a policy role to specify whether the OMA user network PIN-signed message will be accepted. The message's role mask and the policy's role mask are combined using the AND operator. If the result is non-zero, then the message is accepted.</p> <p>Available roles are: <b>SECROLE_OPERATOR_TIPS</b>, <b>SECROLE_KNOWN_PPG</b>, <b>SECROLE_OPERATOR_TPS_OR_SECROLE_KNOWN_PPG</b>, <b>SECROLE_ANY_PUSH_SOURCE</b>, <b>SECROLE_OPERATOR_TPS_OR_SECROLE_ANY_PUSH_SOURCE</b>, <b>SECROLE_KNOWN_PPG_OR_SECROLE_ANY_PUSH_SOURCE</b>, and <b>SECROLE_OPERATOR_TPS_OR_SECROLE_KNOWN_PPG_OR_SECROLE_ANY_PUSH_SOURCE</b>.</p> <p><b>Note</b> IMSI-based NETWPIN and USERNETWPIN may not work for dual SIM phones. The OMA-CP authentication provider only uses the IMSI from executor 0 (the current, active data SIM) when hashing these messages. OMA-CP payloads targeting executor 1 are rejected by the phone. For more information about executors, see Dual SIM.</p>

SETTING	DESCRIPTION
OMACP > UserpinRoles	<p>Select a policy role to specify whether the OMA user PIN or user MAC signed message will be accepted. OMA Client Provisioning User PIN policy determines whether the OMA user PIN or user MAC signed message will be accepted. The message's role mask and the policy's role mask are combined using the AND operator. If the result is non-zero, then the message is accepted.</p> <p>Available roles are: <b>SECROLE_OPERATOR_TIPS</b>, <b>SECROLE_KNOWN_PPG</b>, <b>SECROLE_OPERATOR_TPS_OR_SECROLE_KNOWN_PPG</b>, <b>SECROLE_ANY_PUSH_SOURCE</b>, <b>SECROLE_OPERATOR_TPS_OR_SECROLE_ANY_PUSH_SOURCE</b>, <b>SECROLE_KNOWN_PPG_OR_SECROLE_ANY_PUSH_SOURCE</b>, and <b>SECROLE_OPERATOR_TPS_OR_SECROLE_KNOWN_PPG_OR_SECROLE_ANY_PUSH_SOURCE</b>.</p>
SISL > ServiceIndicationRoles	<p>Specify the security roles that can accept SI messages. Service Indication (SI) Message policy indicates whether SI messages are accepted by specifying the security roles that can accept SI messages. An SI message is sent to the phone to notify users of new services, service updates, and provisioning services.</p> <p>Available roles are: <b>SECROLE_KNOWN_PPG</b>, <b>SECROLE_ANY_PUSH_SOURCE</b>, and <b>SECROLE_KNOWN_PPG_OR_SECROLE_ANY_PUSH_SOURCE</b>.</p>
SISL > ServiceLoadingRoles	<p>Specify the security roles that can accept SL messages. Service Loading (SL) Message policy indicates whether SL messages are accepted by specifying the security roles that can accept SL messages. An SL message downloads new services or provisioning XML to the phone.</p> <p>Available roles are: <b>SECROLE_KNOWN_PPG</b>, <b>SECROLE_ANY_PUSH_SOURCE</b>, and <b>SECROLE_KNOWN_PPG_OR_SECROLE_ANY_PUSH_SOURCE</b>.</p>

## TrustedProvisioningSource

In **PROVURL**, enter the URL for a Trusted Provisioning Server (TPS).

### Related topics

- [DMAcc configuration service provider \(CSP\)](#)
- [PXLOGICAL CSP](#)

# DMClient (Windows Configuration Designer reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Use to specify enterprise-specific mobile device management configuration setting.

## Applies to

Setting	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
UpdateManagementServiceAddresses	X	X	X	X	X

For the **UpdateManagementServiceAddress** setting, enter a list of servers. The first server in the semi-colon delimited list is the server that will be used to instantiate MDM sessions.

## Related topics

- [DMClient configuration service provider \(CSP\)](#)

# EditionUpgrade (Windows Configuration Designer reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Use to upgrade the edition of Windows 10 on the device. [Learn about Windows 10 edition upgrades.](#)

## Applies to

Setting	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
<a href="#">ChangeProductKey</a>	X	X		X	
<a href="#">UpgradeEditionWithLicense</a>	X	X		X	
<a href="#">UpgradeEditionWithProductKey</a>	X	X		X	

## ChangeProductKey

Enter a product key, which will be used to update the existing product key on the device.

## UpgradeEditionWithLicense

Browse to and select a license XML file for the edition upgrade.

## UpgradeEditionWithProductKey

Enter a product key for an edition upgrade of Windows 10 devices.

If a product key is entered in a provisioning package and the user begins installation of the package, a notification is shown to the user that their system will restart to complete the package installation. Upon explicit consent from the user to proceed, the package continues installation and changepk.exe runs using the product key. The user will receive a reminder notification 30 seconds before the automatic restart.

After the device restarts, the edition upgrade process completes. The user will receive a notification of the successful upgrade.

## Related topics

- [WindowsLicensing configuration service provider \(CSP\)](#)

# EmbeddedLockdownProfiles (Windows Configuration Designer reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Use to apply an XML configuration to a mobile device that locks down the device, configures custom layouts, and define multiple roles.

## Applies to

SETTING	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
AssignedAccessXml		X			

1. Create a lockdown XML file, either by using [the Lockdown Designer app](#) or [manually](#).
2. In the **AssignedAccessXml** setting, browse to and select the lockdown XML file that you created.

## Related topics

- [EnterpriseAssignedAccess configuration service provider \(CSP\)](#)

# FirewallConfiguration (Windows Configuration Designer reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Use to enable AllJoyn router to work on public networks.

## Applies to

Setting	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
EnableAllJoynOnPublicNetwork					X

Set to **True** or **False**.

## Related topics

- [AllJoyn](#)

# FirstExperience (Windows Configuration Designer reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Do not configure **FirstExperience** in provisioning packages at this time. These settings will be available to configure the out-of-box experience (OOBE) to set up HoloLens in a future release.

# Folders (Windows Configuration Designer reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Use to add files to the device.

## Applies to

SETTING	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
PublicDocuments	X	X	X	X	

Browse to and select a file or files that will be included in the provisioning package and added to the public profile documents folder on the target device. You can use the **Relative path to directory on target device** field to create a new folder within the public profile documents folder.

# HotSpot (Windows Configuration Designer reference)

10/17/2017 • 4 min to read • [Edit Online](#)

Use HotSpot settings to configure Internet sharing.

## Applies to

SETTING GROUPS	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
All settings		X			

### NOTE

Although the HotSpot settings are available in advanced editing for multiple editions, the settings are only supported on devices running Windows 10 Mobile.

## DedicatedConnections

(Optional) Set DedicatedConnections to a semicolon-separated list of connections.

Specifies the list of Connection Manager cellular connections that Internet sharing will use as public connections.

By default, any available connection will be used as a public connection. However, this node allows a mobile operator to specify one or more connection names to use as public connections.

Specified connections will be mapped, by policy, to the Internet sharing service. All attempts to enumerate Connection Manager connections for the Internet sharing service will return only the mapped connections.

The mapping policy will also include the connection specified in the TetheringNAIConnection value as well.

If the specified connections do not exist, Internet sharing will not start because it will not have any cellular connections available to share.

## Enabled

Specify **True** to enable Internet sharing on the device or **False** to disable Internet sharing.

If Enabled is initially set to **True**, the feature is turned off and the internet sharing screen is removed from Settings so that the user cannot access it. Configuration changes or connection sharing state changes will not be possible.

When Enabled is set to **False**, the internet sharing screen is added to Settings, although sharing is turned off by default until the user turns it on.

## MaxBluetoothUsers

(Optional) Specify the maximum number of simultaneous Bluetooth users that can be connected to a device while sharing over Bluetooth. Set MaxBluetoothUsers to an integer value between 1 and 7 inclusive. The default value is 7.

## MaxUsers

(Optional) Specify the maximum number of simultaneous users that can be connected to a device while sharing. Set MaxUsers to an integer value between 1 and 8 inclusive. The default value is 5.

## MOAppLink

(Optional) Enter an application link that points to a pre-installed application, provided by the mobile operator, that will help a user to subscribe to the mobile operator's Internet sharing service when Internet sharing is not provisioned or entitlement fails.

Set MOAppLink to a valid app ID. The general format for the link is *app://MOappGUID*. For example, if your app ID is `12345678-9012-3456-7890-123456789012`, you must set the value to `app://12345678-9012-3456-7890-123456789012`.

## MOHelpMessage

(Optional) Enter a reference to a localized string, provided by the mobile operator, that is displayed when Internet sharing is not enabled due to entitlement failure. The node takes a language-neutral registry value string, which has the following form:

```
@<res_dll>,-<str_id>
```

Where `<res_dll>` is the resource dll that contains the string and `<str_id>` is the string identifier. For more information on language-neutral string resource registry values, see [Using Registry String Redirection](#).

## MOHelpNumber

(Optional) Enter a mobile operator–specified phone number that is displayed to the user when the Internet sharing service fails to start. The user interface displays a message informing the user that they can call the specified number for help.

## MOInfoLink

(Optional) Enter a mobile operator–specified HTTP link that is displayed to the user when Internet sharing is disabled or the device is not entitled. The user interface displays a message informing the user that they can visit the specified link for more information about how to enable the feature.

## PeerlessTimeout

(Optional) Enter the time-out period, in minutes, after which Internet sharing should automatically turn off if there are no active clients.

Set PeerlessTimeout to any value between 1 and 120 inclusive. A value of 0 is not supported. The default value is 5 minutes.

## PublicConnectionTimeout

(Optional) Enter the time-out value, in minutes, after which Internet sharing is automatically turned off if a cellular connection is not available.

Set PublicConnectionTimeout to any value between 1 and 60 inclusive. The default value is 20 minutes. A value of 0 is not supported.

## TetheringNAIConnection

(Optional) Specify the CDMA TetheringNAI Connection Manager cellular connection that Internet sharing will use

as a public connection. Set TetheringNAIConnection to the CDMA TetheringNAI Connection Manager cellular connection.

If a CDMA mobile operator requires using a Tethering NAI during Internet sharing, they must configure a TetheringNAI connection and then specify the connection in this node.

Specified connections will be mapped, by policy, to the Internet sharing service. All attempts to enumerate Connection Manager connections for the Internet sharing service will return only the mapped connections. The mapping policy will also include the connection specified in the TetheringNAIConnection value as well.

If the specified connections do not exist, Internet sharing will not start because it will not have any cellular connections available to share.

**NOTE**

CDMA phones are limited to one active data connection at a time. This means any application or service (such as e-mail or MMS) that is bound to another connection may not work while Internet sharing is turned on.

# InitialSetup (Windows Configuration Designer reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Use to set the name of the Windows mobile device.

## Applies to

SETTING	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
DeviceName		X			

In **DeviceName**, enter a name for the device. If **DeviceName** is set to an asterisk (\*) or is an empty string, a random device name will be generated.

**DeviceName** is a string with a maximum length of 15 bytes of content:

- **DeviceName** can use ASCII characters (1 byte each) and/or multi-byte characters such as Kanji, so long as you do not exceed 15 bytes of content.
- **DeviceName** cannot use spaces or any of the following characters: { | } ~ [ \ ] ^ ' : ; < = > ? @ ! " # \$ % ` ( ) + / . , \* &, or contain any spaces.
- **DeviceName** cannot use some non-standard characters, such as emoji.

# InternetExplorer (Windows Configuration Designer reference)

9/6/2017 • 2 min to read • [Edit Online](#)

Use to configure settings related to Internet Explorer.

## Applies to

Setting	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
<a href="#">CustomHTTPHeaders</a>		X			
<a href="#">CustomUserAgentString</a>		X			
<a href="#">DataSaving &gt; BrowseDataSaver</a>		X			
<a href="#">DataSaving &gt; ShowPicturesAutomatically</a>		X			
<a href="#">FirstRunURL</a>		X			

## CustomHTTPHeaders

Configure Microsoft Edge to send custom HTTP headers. These will be sent in addition to the default HTTP headers with all HTTP and HTTPS requests. The header is the portion of the HTTP request that defines the form of the message.

- A maximum of 16 custom headers can be defined.
- Custom headers cannot be used to modify the user agent string.
- Each header must be no more than 1 KB in length.

The following header names are reserved and must not be overwritten:

- Accept
- Accept-Charset
- Accept-Encoding
- Authorization
- Expect
- Host
- If-Match
- If-Modified-Since
- If-None-Match
- If-Range
- If-Unmodified-Since

- Max-Forwards
- Proxy-Authorization
- Range
- Referer
- TE
- USER-AGENT
- X-WAP-PROFILE

1. In **Available customizations**, select **CustomHTTPHeaders**, enter a name, and then click **Add**.
2. In **Available customizations**, select the name that you just created.
3. Enter the custom header.

## CustomUserAgentString

The user agent string indicates which browser you are using, its version number, and details about your system, such as operating system and version. A web server can use this information to provide content that is tailored for your specific browser and phone.

The user agent string for the browser cannot be modified. By default, the string has the following format:

```
Mozilla/5.0 (Windows Phone 10.0; Android 4.2.1; <Manufacturer>; <Device>) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/42.0.2311.135 Mobile Safari/537.36 Edge/12.10166
```

- is automatically replaced with the OEM name. This is the same as the PhoneManufacturer setting value that is set as part of the customization Phone metadata in DeviceTargetingInfo.
- is replaced with the device name or phone name. This is the same as the PhonemodelName setting value that is set as part of the customization Phone metadata in DeviceTargetingInfo.

### **Limitations and restrictions:**

- The user agent string for the browser cannot be modified outside of the customizations listed above.
- The user agent type registry setting cannot be modified or used to change the default browser view from Mobile to Desktop.

## BrowseDataSaver

Use to set the browser data saver default setting. **True** turns on the browser data saver feature.

Partners can configure the default setting for the browser data saver feature by turning the browser optimization service (through the BrowserDataSaver setting) on or off.

## ShowPicturesAutomatically

Use to enable or disable whether the **Show pictures automatically** setting is available in Internet Explorer **advanced settings**.

## FirstRunURL

Use to set the home page that appears the first time that Microsoft Edge is opened. This page is only shown the first time the browser is opened. After that, the browser displays either the most recently viewed page or an empty page if the user has closed all tabs or opens a new tab.

Specify the **FirstRunURL** value with a valid link that starts with http://. It is recommended you use a forward link that redirects the user to a localized page.

# Licensing (Windows Configuration Designer reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Use for settings related to Microsoft licensing programs.

## Applies to

Setting	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
<a href="#">AllowWindowsEntitlementReactivation</a>	X				
<a href="#">DisallowKMSClientOnlineAVSValidation</a>	X				

## AllowWindowsEntitlementReactivation

Enable or disable Windows license reactivation.

## DisallowKMSClientOnlineAVSValidation

Enable this setting to prevent the device from sending data to Microsoft regarding its activation state.

# Maps (Windows Configuration Designer reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Use for settings related to Maps.

## Applies to

Setting	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
ChinaVariantWin10	X	X	X	X	
UseExternalStorage	X	X	X	X	
UseSmallerCache	X	X	X	X	

## ChinaVariantWin10

Use **ChinaVariantWin10** to specify that the Windows device is intended to ship in China. When set to **True**, maps approved by the State Bureau of Surveying and Mapping in China are used, which are obtained from a server located in China.

This customization may result in different maps, servers, or other configuration changes on the device.

## UseExternalStorage

Use to store map data on an SD card.

Map data is used by the Maps application and the map control for third-party applications. This data can be stored on an SD card, which provides the advantage of saving internal memory space for user data and allows the user to download more offline map data. Microsoft recommends enabling the **UseExternalStorage** setting on devices that have less than 8 GB of user storage and an SD card slot.

You can use **UseExternalStorage** whether or not you include an SD card with preloaded map data on the phone. If set to **True**, the OS only allows the user to download offline maps when an SD card is present. If an SD card is not present, users can still view and cache maps, but they will not be able to download a region of offline maps until an SD card is inserted.

If set to **False**, map data will always be stored on the internal data partition of the device.

### NOTE

SD card performance can affect the quality of the Maps experience when maps are stored on the SD card. When an SD card is used, Microsoft recommends that you test the Maps experience and the speed of map downloads with the specific SD card part that will be used on retail phones to determine if performance is satisfactory.

## UseSmallerCache

Do not use.

# Messaging (Windows Configuration Designer reference)

10/17/2017 • 17 min to read • [Edit Online](#)

Use for settings related to Messaging and Commercial Mobile Alert System (CMAS).

## IMPORTANT

These settings are intended to be used only by manufacturers, mobile operators, and solution providers when configuring devices, and are not intended for use by administrators in the enterprise.

## NOTE

CMAS is now known as Wireless Emergency Alerts (WEA).

## Applies to

SETTING	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
All settings		X			

## GlobalSettings

### DisplayCmasLifo

Use this setting to change the order in which CMAS alert messages are displayed, from the default first in/first out (FIFO) message order to last in/first out (LIFO) message order.

If the phone receives at least one CMAS alert message which has not been acknowledged by the user, and another CMAS alert message arrives on the phone, partners can configure the order in which the newly received alert messages are displayed on the phone regardless of the service category of the alert. Users will not be able to change the message order once it has been set.

If partners do not specify a value for this customization, the default FIFO display order is used. Users will be able to acknowledge the messages in the reverse order they were received.

When configured as **True**, you set a LIFO message order. When configured as **False**, you set a FIFO message order.

### EnableCustomLineSetupDialog

Enable this setting to allow custom line setup dialogs in the Messaging app.

### ShowSendingStatus

#### NOTE

This setting is removed in Windows 10, version 1709.

Set **ShowSendingStatus** to **True** to display the sending status for SMS/MMS messages.

## VoiceMailIntercept

Partners can define a filter that intercepts an incoming SMS message and triggers visual voicemail synchronization. The filtered message does not appear in the user's conversation list.

A visual voicemail sync is triggered by an incoming SMS message if the following conditions are met:

- The message sender value starts with the string specified in the SyncSender setting. The length of the specified values must be greater than 3 characters but less than 75 characters.
- The body of the message starts with the string specified in the SyncPrefix setting. The length of the specified values must be greater than 3 characters but less than 75 characters.
- Visual voicemail is configured and enabled. For more information, see [Visual voicemail](#).

### NOTE

These settings are atomic, so both SyncSender and SyncPrefix must be set.

The SyncSender and SyncPrefix values vary for each mobile operator, so you must work with your mobile operators to obtain the correct or required values.

SETTING	DESCRIPTION
SyncPrefix	Specify a value for SyncPrefix that is greater than 3 characters but less than 75 characters in length. For networks that support it, this value can be the keyword for the SMS notification.
SyncSender	Specify a value for SyncSender that is greater than 3 characters but less than 75 characters in length. For networks that support it, this value can be a short code of the mailbox server that sends a standard SMS notification.

## PerSimSettings

Use to configure settings for each subscriber identification module (SIM) card. Enter the Integrated Circuit Card Identifier (ICCID) for the SIM card, click **Add**, and then configure the following settings.

### AllowMmsIfDataIsOff

SETTING	DESCRIPTION
AllowMmsIfDataIsOff	<b>True</b> allows MMS if data is off
AllowMmsIfDataIsOffSupported	<b>True</b> shows the toggle for allowing MMS if data is turned off
AllowMmsIfDataIsOffWhileRoaming	<b>True</b> allows MMS if data is off while roaming

### AllowSelectAllContacts

### NOTE

This setting is removed in Windows 10, version 1709.

Set to **True** to show the select all contacts/unselect all menu option to allow users to easily select multiple recipients for an SMS or MMS message. This menu option provides users with an easier way to add multiple

recipients and may also meet a mandatory requirement for some mobile operator networks.

Windows 10 Mobile supports the following select multiple recipients features:

- A multi-select chooser, which enables users to choose multiple contacts.
- A **Select all contacts/unselect all** menu option, which enables users to select or unselect all their contacts. This option is not shown by default and must be enabled by the OEM.

### AllowSendingDeliveryReport

Specify whether the phone automatically sends a receipt acknowledgment for MMS messages. Partners can specify whether the phone automatically sends a receipt acknowledgment for MMS messages when they arrive, and they can determine whether users can control the receipt acknowledgments by using the **Send MMS acknowledgment** toggle in **Messaging > settings**. By default, this user setting is visible and turned on.

SETTING	DESCRIPTION
AllowSendingDeliveryReport	<b>True</b> sets the <b>Send MMS acknowledgment</b> toggle to <b>On</b>
AllowSendingDeliveryReportIsSupported	<b>True</b> shows the <b>Send MMS acknowledgment</b> toggle, and <b>False</b> hides the toggle

### AutomaticallyDownload

Specify whether MMS messages are automatically downloaded.

SETTING	DESCRIPTION
AutomaticallyDownload	<b>True</b> sets the <b>Automatically download MMS</b> toggle to <b>On</b>
ShowAutomaticallyDownloadMMSToggle	<b>True</b> shows the <b>Automatically download MMS</b> toggle, and <b>False</b> hides the toggle

### DefaultContentLocationUrl

#### NOTE

This setting is removed in Windows 10, version 1709.

For networks that require it, you can specify the default GET path within the MMSC to use when the GET URL is missing from the WAP push MMS notification.

Set **DefaultContentLocationUrl** to specify the default GET path within the MMSC.

### ErrorCodeEnabled

#### NOTE

This setting is removed in Windows 10, version 1709.

You can choose to display additional content in the conversation view when an SMS or MMS message fails to send. This content includes a specific error code in decimal format that the user can report to technical support. Common errors also include a friendly string to help the user self-diagnose and fix the problem.

Set to **True** to display the error message with an explanation of the problem and the decimal-format error codes. When set to **False**, the full error message is not displayed.

### EmergencyAlertOptions

Configure settings for CMAS alerts.

SETTING	DESCRIPTION
CmasAMBERAlertEnabled	<b>True</b> enables the device to receive AMBER alerts
CmasExtremeAlertEnabled	<b>True</b> enables the device to receive extreme alerts
CmasSevereAlertEnabled	<b>True</b> enables the device to receive severe alerts
EmOperatorEnabled	Select which Emergency Alerts Settings page is displayed from dropdown menu
SevereAlertDependentOnExtremeAlert	When set as <b>True</b> , the CMAS-Extreme alert option must be on to modify CMAS-Severe alert option

## General

SETTING	DESCRIPTION
AllowSelectAllContacts	Set to <b>True</b> to show the <b>select all contacts/unselect all</b> menu option to allow users to easily select multiple recipients for an SMS or MMS message. This menu option provides users with an easier way to add multiple recipients and may also meet a mandatory requirement for some mobile operator networks. Windows 10 Mobile supports the following select multiple recipients features: <ul style="list-style-type: none"> <li>- A multi-select chooser, which enables users to choose multiple contacts.</li> <li>- A <b>select all contacts/unselect all</b> menu option, which enables users to select or unselect all their contacts. This option is not shown by default and must be enabled by the OEM.</li> </ul>
AllowSMSToSMTPAddress	Allow SMS to SMTP address.
AssistedDialingMcc	By setting AssistedDialingMcc and AssistedDialingMnc, international assisted dialing will be enabled for SMS if the user setting for international assisted dialing is enabled. Enter the Mobile Country Code (MCC) to use for sending SMS.
AssistedDialingMnc	By setting AssistedDialingMcc and AssistedDialingMnc, international assisted dialing will be enabled for SMS if the user setting for international assisted dialing is enabled. Enter the Mobile Network Code (MNC) to use for sending SMS.
AssistedDialingPlusCodeSupportOverride	For devices that support IMS over SMS, you can override support for the assisted dialing plus (+) code for SMS by setting AssistedDialingPlusCodeSupportOverride. If enabled, the OS will not convert the plus (+) code to the proper assisted number when the user turns on the dialing assist option.
AutoRetryDownload	You can configure the messaging app to automatically retry downloading an MMS message if the initial download attempt fails. When this customization is enabled, the download is retried 3 times at 20-, 40-, and 60-second intervals.

Setting	Description
BroadcastChannels	You can specify one or more ports from which the device will accept cellular broadcast messages. Set the BroadcastChannels value to the port number(s) that can accept cellular broadcast messages. If you specify the same port that Windows 10 Mobile already recognizes as an Emergency Alert port (a CMAS or ETWS port number) and a cell broadcast message is received on that port, the user will only receive the message once. The message that is received will be displayed as an Emergency Alert message.
ConvertLongSMSstoMMS	For networks that do support MMS and do not support segmentation of SMS messages, you can specify an automatic switch from SMS to MMS for long messages.
DefaultContentLocationUrl	For networks that require it, you can specify the default GET path within the MMSC to use when the GET URL is missing from the WAP push MMS notification. Set DefaultContentLocationUrl to specify the default GET path within the MMSC.
ErrorCodeEnabled	You can choose to display additional content in the conversation view when an SMS or MMS message fails to send. This content includes a specific error code in decimal format that the user can report to technical support. Common errors also include a friendly string to help the user self-diagnose and fix the problem. Set to <b>True</b> to display the error message with an explanation of the problem and the decimal-format error codes. When set to <b>False</b> , the full error message is not displayed.
HideMediumSIPopups	By default, when a service indication message is received with a signal-medium or signal-high setting, the phone interrupts and shows the user prompt for these messages. However, you can hide the user prompts for signal-medium messages.
ImsiAuthenticationToken	Configure whether MMS messages include the IMSI in the GET and POST header. Set ImsiAuthenticationToken to the token used as the header for authentication. The string value should match the IMSI provided by the UICC.
LimitRecipients	Set the maximum number of recipients to which a single SMS or MMS message can be sent. Enter a number between 1 and 500 to limit the maximum number of recipients.
MaxRetryCount	You can specify the number of times that the phone can retry sending the failed MMS message and photo before the user receives a notification that the photo could not be sent. Specify MaxRetryCount to specify the number of times the MMS transport will attempt resending the MMS message. This value has a maximum limit of 3.
MMSLimitAttachments	You can specify the maximum number of attachments for MMS messages, from 1 to 20. The default is 5.

SETTING	DESCRIPTION
RetrySize	For MMS messages that have photo attachments and that fail to send, you can choose to automatically resize the photo and attempt to resend the message. Specify the maximum size to use to resize the photo in KB. Minimum is 0xA (10 KB).
SetCacheControlNoTransform	When set, proxies and transcoders are instructed not to change the HTTP header and the content should not be modified. A value of 1 or 0x1 adds support for the HTTP header Cache-Control No-Transform directive. When the SetCacheControlNoTransform``Value is set to 0 or 0x0 or when the setting is not set, the default HTTP header Cache-Control No-Cache directive is used.
ShowRequiredMonthlyTest	<b>True</b> enables devices to receive CMAS Required Monthly Test (RMT) messages and have these show up on the device. <b>False</b> disables devices from receiving CMAS RMT messages.
SmscPanelDisabled	<b>True</b> disables the short message service center (SMSC) panel.
SMSToSMTPShortCode	Use to configure SMS messages to be sent to email addresses and phone numbers. <input type="checkbox"/> 0 disables sending SMS messages to SMTP addresses. <input checked="" type="checkbox"/> 1 enables sending SMS messages to SMTP addresses.
TargetVideoFormat	You can specify the transcoding to use for video files sent as attachments in MMS messages. Set TargetVideoFormat to one of the following values to configure the default transcoding for video files sent as attachments in MMS messages: <ul style="list-style-type: none"> <li>- 0 or 0x0 Sets the transcoding to H.264 + AAC + MP4. This is the default set by the OS.</li> <li>- 1 or 0x1 Sets the transcoding to H.264 + AAC + 3GP.</li> <li>- 2 or 0x2 Sets the transcoding to H.263 + AMR.NB + 3GP.</li> <li>- 3 or 0x3 Sets the transcoding to MPEG4 + AMR.NB + 3GP.</li> </ul>
UAProf	You can specify a user agent profile to use on the phone for MMS messages. The user agent profile XML file details a phone's hardware specifications and media capabilities so that an MMS application server (MMSC) can return supported optimized media content to the phone. The user agent profile XML file is generally stored on the MMSC. There are two ways to correlate a user agent profile with a given phone: <ul style="list-style-type: none"> <li>- You can take the user agent string of the phone that is sent with MMS requests and use it as a hash to map to the user agent profile on the MMSC. The user agent string cannot be modified.</li> <li>- Alternatively, you can directly set the URI of the user agent profile on the phone.</li> </ul> <p>Set UAProf to the full URI of your user agent profile file. Optionally, you can also specify the custom user agent property name for MMS that is sent in the header by setting UAProfToken to either <code>x-wap-profile</code> or <code>profile</code>.</p>
UAProfToken	You can specify a user agent profile to use on the phone for MMS messages. The user agent profile XML file details a phone's hardware specifications and media capabilities so that an MMS application server (MMSC) can return supported optimized media content to the phone. The user agent profile XML file is generally stored on the MMSC.

SETTING	DESCRIPTION
UseDefaultAddress	By default, the MMS transport sends an acknowledgement to the provisioned MMS application server (MMSC). However, on some networks, the correct server to use is sent as a URL in the MMS message. In that case, a registry key must be set, or else the acknowledgement will not be received and the server will continue to send duplicate messages. <b>True</b> enables some networks to correctly acknowledge MMS messages. <b>False</b> disables the feature.
UserAgentString	Set UserAgentString to the new user agent string for MMS in its entirety. By default, this string has the format WindowsPhoneMMS/MicrosoftMMSVersionNumber WindowsPhoneOS/OSVersion-buildNumber OEM-deviceName, in which the italicized text is replaced with the appropriate values for the phone.
UseUTF8ForUnspecifiedCharset	Some incoming MMS messages may not specify a character encoding. To properly decode MMS messages that do not specify a character encoding, you can set UTF-8 to decode the message.
WapPushTechnology	For networks that require non-standard handling of single-segment incoming MMS WAP Push notifications, you can specify that MMS messages may have some of their content truncated and that they may require special handling to reconstruct truncated field values. <b>1</b> or <b>0x1</b> enables MMS messages to have some of their content truncated. <b>0</b> or <b>0x0</b> disables MMS messages from being truncated

## ImsiAuthenticationToken

### NOTE

This setting is removed in Windows 10, version 1709.

Configure whether MMS messages include the IMSI in the GET and POST header.

Set **ImsiAuthenticationToken** to the token used as the header for authentication. The string value should match the IMSI provided by the UICC.

### LatAlertOptions

Enable **LatLocalAlertEnabled** to enable support for LAT-Alert Local Alerts for devices sold in Chile. For more information, see [Emergency notifications](#).

### MaxRetryCount

### NOTE

This setting is removed in Windows 10, version 1709.

You can specify the number of times that the phone can retry sending the failed MMS message and photo before the user receives a notification that the photo could not be sent.

Specify MaxRetryCount to specify the number of times the MMS transport will attempt resending the MMS message. This value has a maximum limit of 3.

## MMSGroupText

Set options for group messages sent to multiple people.

SETTING	DESCRIPTION
MMSGroupText	<b>True</b> enables group messages to multiple people sent as MMS.
ShowMMSGroupTextUI	<b>True</b> shows the toggle for group text in messaging settings.
ShowMmsGroupTextWarning	<b>True</b> shows the warning that alerts users of possible additional charges before sending a group text as MMS.

## NIAAlertOptions

Enable `NI2AlertEnabled` to enable support for the Netherlands Announcements for devices sold in the Netherlands. For more information, see [Emergency notifications](#).

## RcsOptions

Set options for Rich Communications Services (RCS).

SETTING	DESCRIPTION
RcsEnabled	Toggle to enable/disable RCS service. Set to <b>True</b> to enable.
RcsFileTransferAutoAccept	Set to <b>True</b> to auto-accept RCS incoming file transfer if the file size is less than warning file size.
RcsSendReadReceipt	Set to <b>True</b> to send read receipt to the sender when a message is read.
ShowRcsEnabled	Set to <b>True</b> to show the toggle for RCS activation.

## RequestDeliveryReport

Set options related to MMS message notifications. You can specify whether users receive notification that MMS messages could not be delivered, and determine whether users can control this by using the MMS delivery confirmation toggle in **Messaging > settings**. By default, this user setting is visible but turned off.

SETTING	DESCRIPTION
RequestDeliveryReport	Set to <b>True</b> to set the default value to on.
RequestDeliveryReportIsSupported	<b>True</b> shows the toggle for MMS delivery confirmation, and <b>False</b> hides the toggle.

## SMSDeliveryNotify

SETTING	DESCRIPTION
DeliveryNotifySupported	Set to <b>True</b> to enable SMS delivery confirmation.
SMSDeliveryNotify	Set to <b>True</b> to toggle SMS delivery confirmation.

## TargetVideoFormat

#### NOTE

This setting is removed in Windows 10, version 1709.

You can specify the transcoding to use for video files sent as attachments in MMS messages.

Set TargetVideoFormat to one of the following values to configure the default transcoding for video files sent as attachments in MMS messages:

VALUE	DESCRIPTION
0 or 0x0	Sets the transcoding to H.264 + AAC + MP4. This is the default set by the OS.
1 or 0x1	Sets the transcoding to H.264 + AAC + 3GP.
2 or 0x2	Sets the transcoding to H.263 + AMR.NB + 3GP.
3 or 0x3	Sets the transcoding to MPEG4 + AMR.NB + 3GP.

## UAProf

#### NOTE

This setting is removed in Windows 10, version 1709.

You can specify a user agent profile to use on the phone for MMS messages. The user agent profile XML file details a phone's hardware specifications and media capabilities so that an MMS application server (MMSC) can return supported optimized media content to the phone. The user agent profile XML file is generally stored on the MMSC.

There are two ways to correlate a user agent profile with a given phone:

- You can take the user agent string of the phone that is sent with MMS requests and use it as a hash to map to the user agent profile on the MMSC. The user agent string cannot be modified.
- Alternatively, you can directly set the URI of the user agent profile on the phone.

Set **UAProf** to the full URI of your user agent profile file. Optionally, you can also specify the custom user agent property name for MMS that is sent in the header by setting **UAProfToken** to either `x-wap-profile` or `profile`.

## UAProfToken

#### NOTE

This setting is removed in Windows 10, version 1709.

You can specify a user agent profile to use on the phone for MMS messages. The user agent profile XML file details a phone's hardware specifications and media capabilities so that an MMS application server (MMSC) can return supported optimized media content to the phone. The user agent profile XML file is generally stored on the MMSC.

Optional, in addition to specifying **UAProf**, you can also specify the custom user agent property name for MMS that is sent in the header by setting **UAProfToken** to either `x-wap-profile` or `profile`.

## UserAgentString

**NOTE**

This setting is removed in Windows 10, version 1709.

Set **UserAgentString** to the new user agent string for MMS in its entirely.

By default, this string has the format WindowsPhoneMMS/MicrosoftMMSVersionNumber

WindowsPhoneOS/OSVersion-buildNumber OEM-deviceName, in which the italicized text is replaced with the appropriate values for the phone.

w4

SETTING	DESCRIPTION
ADDR	Specify the absolute MMSC URL. The possible values to configure the ADDR parameter are: <ul style="list-style-type: none"><li>- A Uniform Resource Identifier (URI)</li><li>- An IPv4 address represented in decimal format with dots as delimiters</li><li>- A fully qualified Internet domain name</li></ul>
APPID	Set to w4 .
MS	(optional) Specify the maximum size of MMS, in KB. If the value is not a number, or is less than or equal to 10, it will be ignored and outgoing MMS will not be resized.
NAME	(optional) Enter user-readable application identity. This parameter is also used to define part of the registry path for the APPLICATION parameters. The possible values to configure the NAME parameter are: <ul style="list-style-type: none"><li>- Character string containing the name</li><li>- no value specified</li></ul> If no value is specified, the registry location will default to . If NAME is greater than 40 characters, it will be truncated to 40 characters.
TONAPID	Specify the network access point identification name (NAPID) defined in the provisioning file. This parameter takes a string value. It is only possible to refer to network access points defined within the same provisioning file (except if the INTERNET attribute is set in the NAPDEF characteristic). For more information about the NAPDEF characteristic, see <a href="#">NAPDEF configuration service provider</a> .
TOPROXY	Specify one logical proxy with a matching PROXY-ID. It is only possible to refer to proxies defined within the same provisioning file. Only one proxy can be listed. The TO-PROXY value must be set to the value of the PROXY ID in PXLOGICAL that defines the MMS specific-proxy.

## WapPushTechnology

**NOTE**

These settings are removed in Windows 10, version 1709.

For networks that require non-standard handling of single-segment incoming MMS WAP Push notifications, you

can specify that MMS messages may have some of their content truncated and that they may require special handling to reconstruct truncated field values.

VALUE	DESCRIPTION
1 or 0x1	Enables MMS messages to have some of their content truncated.
0 or 0x0	Disables MMS messages from being truncated.

## Related topics

- [Customizations for SMS and MMS](#)

# ModemConfiguration (Windows Configuration Designer reference)

10/17/2017 • 1 min to read • [Edit Online](#)

ModemConfiguration settings are removed in Windows 10, version 1709.

## Applies to

Setting	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
All settings		X			

# Multivariant (Windows Configuration Designer reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Use to select a default profile for mobile devices that have multivariant configurations.

## Applies to

SETTING	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
DefaultProfile		X			

If you will be adding [multivariant settings](#) to your provisioning package, you can use the **DefaultProfile** setting to specify which variant should be applied by default if OOBEM is skipped. In the **DefaultProfile** field, enter the UINAME from your customizations.xml that you want to use as default.

# NetworkProxy (Windows Configuration Designer reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Use for settings related to NetworkProxy.

## Applies to

Setting	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
All settings			X		

## AutoDetect

Automatically detect network proxy settings.

Value	Description
0	Disabled. Do not automatically detect settings.
1	Enabled. Automatically detect settings.

## ProxyServer

Node for configuring a static proxy for Ethernet and Wi-Fi connections. The same proxy server is used for all protocols - including HTTP, HTTPS, FTP, and SOCKS. These settings do not apply to VPN connections.

Setting	Description
ProxyAddress	Address to the proxy server. Specify an address in the format <code>server:port</code> .
ProxyExceptions	Addresses that should not use the proxy server. The system will not use the proxy server for addresses that begin with the values specified in this node. Use semicolons (;) to separate entries.
UseProxyForLocalAddresses	Whether the proxy server should be used for local (intranet) addresses. <ul style="list-style-type: none"><li>- 0 = Disabled. Do not use the proxy server for local addresses.</li><li>- 1 = Enabled. Use the proxy server for local addresses.</li></ul>

## SetupScriptUrl

Address to the PAC script you want to use.

## Related topics

- NetworkProxy configuration service provider (CSP)

# NetworkQoSPolicy (Windows Configuration Designer reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Use to create network Quality of Service (QoS) policies. A QoS policy performs a set of actions on network traffic based on a set of matching conditions.

## Applies to

SETTING	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
All settings			X		

1. In **Available customizations**, select **NetworkQoSPolicy**, enter a friendly name for the account, and then click **Add**.
2. In **Available customizations**, select the name that you just created. The following table describes the settings you can configure.

SETTING	DESCRIPTION
AppPathNameMatchCondition	Enter the name of an application to be used to match the network traffic, such as application.exe or %ProgramFiles%\application.exe.
DestinationPortMatchCondition	Specify a port or a range of ports to be used to match the network traffic. Valid values are [first port number]-[last port number], or [port number].
DSCPAction	Enter the differentiated services code point (DSCP) value to apply to match with network traffic. Valid values are 0-63.
IPProtocolMatchCondition	Select between <b>Both TCP and UDP</b> , <b>TCP</b> , and <b>UDP</b> to specify the IP protocol used to match the network traffic.
PriorityValue8021Action	Specify the IEEE 802.1p value. Valid values are 0 through 7.
SourcePortMatchCondition	Specify a single port or range of ports. Valid values are [first port number]-[last port number], or [port number].

## Related topics

- [NetworkQoSPolicy configuration service provider \(CSP\)](#)

# NFC (Windows Configuration Designer reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Use to configure settings related to near field communications (NFC) subsystem.

## Applies to

SETTING	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
All settings		X			

Expand **NFC > SEMgr > UI**. The following table describes the settings you can configure.

SETTING	DESCRIPTION
CardEmulationState	Configure the default state of <b>Tap to pay</b> . Select between <b>OFF</b> , <b>When Phone Unlocked</b> , <b>When Screen On</b> , and <b>Anytime</b> .
DefaultFastCardSetting	Configure the default fast card usage for NFC payments. Select between <b>When Phone Unlocked</b> , <b>When Screen On</b> , and <b>Anytime</b> .
HideFastCardsOption	Show or hide the fast cards options drop-down menu in the <b>NFC &gt; Tap to pay</b> control panel.

# OOBE (Windows Configuration Designer reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Use to configure settings for the Out Of Box Experience (OOBE).

## Applies to

SETTING	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
<a href="#">Mobile &gt; EnforceEnterpriseProvisioning</a>		X			
<a href="#">Mobile &gt; HideOobe</a>		X			
<a href="#">Desktop &gt; HideOobe</a>	X				

## EnforceEnterpriseProvisioning

When set to **True**, it forces the OOBE flow into using the enterprise provisioning page without making the user interact with the Windows button. This is the default setting.

When set to **False**, it does not force the OOBE flow to the enterprise provisioning page.

## HideOobe for mobile

When set to **True**, it hides the interactive OOBE flow for Windows 10 Mobile.

When set to **False**, the OOBE screens are displayed.

## HideOobe for desktop

When set to **True**, it hides the interactive OOBE flow for Windows 10.

### NOTE

You must create a user account if you set the value to true or the device will not be usable.

When set to **False**, the OOBE screens are displayed.

# OtherAssets (Windows Configuration Designer reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Use to configure settings for Map data.

## Applies to

Setting	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
MapData		X			

Use **MapData** to specify the source directory location of the map region you want to include.

For example, if C:\Path\Maps\Europe contains the downloaded map data that you want to preload, set the value to that directory.

To add additional maps, add a new MapData setting and set the source to the directory location of the map region you want to include.

# Personalization (Windows Configuration Designer reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Use to configure settings to personalize a PC.

## Applies to

Setting	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
<a href="#">DeployDesktopImage</a>	X				
<a href="#">DeployLockScreenImage</a>	X				
<a href="#">DesktopImageUrl</a>	X				
<a href="#">LockScreenImageUrl</a>	X				

## DeployDesktopImage

Deploy a jpg, jpeg or png image to the device to be used as desktop image. If you have a local file and want to embed it into the package being deployed, you configure this setting and [DesktopImageUrl](#).

When using **DeployDesktopImage** and [DeployLockScreenImageFile](#deploylockscreenimage, the file names need to be different.

## DeployLockScreenImage

Deploy a jpg, jpeg or png image to the device to be used as lock screen image. If you have a local file and want to embed it into the package being deployed, you configure this setting and [LockScreenImageUrl](#).

When using [DeployDesktopImage](#) and **DeployLockScreenImageFile**, the file names need to be different.

## DesktopImageUrl

Specify a jpg, jpeg or png image to be used as desktop image. This setting can take a http or https url to a remote image to be downloaded or a file url to a local image. If you have a local file and want to embed it into the package being deployed, you also set [DeployDesktopImage](#).

## LockScreenImageUrl

Specify a jpg, jpeg or png image to be used as Lock Screen Image. This setting can take a http or https Url to a remote image to be downloaded or a file Url to an existing local image. If you have a local file and want to embed it into the package being deployed, you also set [DeployLockScreenImage](#).

# Policies (Windows Configuration Designer reference)

10/17/2017 • 24 min to read • [Edit Online](#)

This section describes the **Policies** settings that you can configure in [provisioning packages](#) for Windows 10 using Windows Configuration Designer. Each setting below links to its supported values, as documented in the [Policy configuration service provider \(CSP\)](#).

## AboveLock

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	Iot Core
<a href="#">AllowActionCenterNotifications</a>	Allow Action Center notifications above the device lock screen.		X			
<a href="#">AllowToasts</a>	Allow toast notifications above the device lock screen.	X	X			

## Accounts

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	Iot Core
<a href="#">AllowAddingNonMicrosoftAccountManually</a>	Whether users can add non-Microsoft email accounts	X	X			
<a href="#">AllowMicrosoftAccountConnection</a>	Whether users can use a Microsoft account for non-email-related connection authentication and services	X	X			
<a href="#">AllowMicrosoftAccountSignInAssistant</a>	Disable the <b>Microsoft Account Sign-In Assistant</b> (wlidsvc) NT service	X	X			

SETTING	DESCRIPTION	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
DomainsForEmailSync	List of domains that are allowed to sync email on the devices	X	X			

## ApplicationDefaults

SETTING	DESCRIPTION	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
DefaultAssociationsConfiguration	Set default file type and protocol associations	X				

## ApplicationManagement

SETTING	DESCRIPTION	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
AllowAllTrustedApps	Whether non-Microsoft Store apps are allowed	X	X			
AllowAppStoreAutoUpdate	Whether automatic update of apps from Microsoft Store is allowed	X	X			
AllowDeveloperUnlock	Whether developer unlock of device is allowed	X	X	X	X	X
AllowGameDVR	Whether DVR and broadcasting is allowed	X				
AllowSharedUserAppData	Whether multiple users of the same app can share data	X	X			

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	Iot Core
AllowStore	Whether app store is allowed at device (?)		X			
ApplicationRestrictions	An XML blob that specifies app restrictions, such as an allow list, disallow list, etc.		X			
RestrictAppDataToSystemVolume	Whether app data is restricted to the system drive	X	X			
RestrictAppToSystemVolume	Whether the installation of apps is restricted to the system drive	X	X			

## Authentication

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	Iot Core
AllowFastReconnect	Allows EAP Fast Reconnect from being attempted for EAP Method TLS.	X	X	X	X	X

## BitLocker

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	Iot Core
EncryptionMethod	Specify BitLocker drive encryption method and cipher strength	X	X			

## Bluetooth

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IoT Core
AllowAdvertising	Whether the device can send out Bluetooth advertisements	X	X	X	X	X
AllowDiscoverableMode	Whether other Bluetooth-enabled devices can discover the device	X	X	X	X	X
AllowPrepairing	Whether to allow specific bundled Bluetooth peripherals to automatically pair with the host device	X	X	X		X
LocalDeviceName	Set the local Bluetooth device name	X	X	X		X
ServicesAllowedList	Set a list of allowable services and profiles	X	X		X	

## Browser

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IoT Core

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IoT Core
AllowAddressBarDropdown	Specify whether to allow the address bar drop-down functionality in Microsoft Edge. If you want to minimize network connections from Microsoft Edge to Microsoft services, we recommend disabling this functionality.	X				
AllowAutofill	Specify whether autofill on websites is allowed.	X	X	X		
AllowBrowser	Specify whether the browser is allowed on the device.		X			
AllowCookies	Specify whether cookies are allowed.	X	X	X		
AllowDeveloperTools	Specify whether employees can use F12 Developer Tools on Microsoft Edge.	X				
AllowDoNotTrack	Specify whether Do Not Track headers are allowed.	X	X	X		

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
AllowExtensions	Specify whether Microsoft Edge extensions are allowed.	X				
AllowFlash	Specify whether Adobe Flash can run in Microsoft Edge.	X				
AllowFlashClickToRun	Specify whether users must take an action, such as clicking the content or a Click-to-Run button, before seeing content in Adobe Flash.	X				
AllowInPrivate	Specify whether InPrivate browsing is allowed on corporate networks.	X	X	X		
AllowMicrosoftCompatibilityList	Specify whether to use the Microsoft compatibility list in Microsoft Edge.	X	X	X		
AllowPasswordManager	Specify whether saving and managing passwords locally on the device is allowed.	X	X	X		
AllowPopups	Specify whether pop-up blocker is allowed or enabled.	X				

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IoT Core
AllowSearchEngineCustomization	Allow search engine customization for MDM-enrolled devices.	X				
AllowSearchSuggestionsinAddressBar	Specify whether search suggestions are allowed in the address bar.	X	X	X		
AllowSmartScreen	Specify whether Windows Defender SmartScreen is allowed.	X	X	X		
ClearBrowsingDataOnExit	Specify whether to clear browsing data when exiting Microsoft Edge.	X				
ConfigureAdditionalSearchEngines	Allows you to add up to 5 additional search engines for MDM-enrolled devices.	X	X	X		
DisableLockdownOfStartPages	Specify whether the lockdown on the Start pages is disabled.	X				
EnterpriseModeSiteList	Allow the user to specify a URL of an enterprise site list.	X				

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
EnterpriseSiteListServiceUrl	This policy (introduced in Windows 10, version 1507) was deprecated in Windows 10, version 1511 by <a href="#">Browser/EnterpriseModeSiteList</a> .	X				
FirstRunURL	Specify the URL that Microsoft Edge will use when it is opened for the first time.		X			
HomePages	Specify your Start pages for MDM-enrolled devices.	X				
PreventAccessToAboutFlagsInMicrosoftEdge	Specify whether users can access the <b>about:flags</b> page, which is used to change developer settings and to enable experimental features.	X	X	X		
PreventFirstRunPage	Specify whether to enable or disable the First Run webpage.	X				
PreventLiveTileDataCollection	Specify whether Microsoft can collect information to create a Live Tile when pinning a site to Start from Microsoft Edge.	X	X	X		

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
PreventSmartScreenPromptOverride	Specify whether users can override the Windows Defender SmartScreen Filter warnings about potentially malicious websites.	X	X	X		
PreventSmartScreenPromptOverrideForFiles	Specify whether users can override the Windows Defender SmartScreen Filter warnings about downloading unverified files.	X	X	X		
PreventUsingLocalHostIPAddressForWebRTC	Specify whether a user's localhost IP address is displayed while making phone calls using the WebRTC protocol.	X	X	X		
SendIntranetTraffictoInternetExplorer	Specify whether to send intranet traffic to Internet Explorer.	X				
SetDefaultSearchEngine	Configure the default search engine for your employees.	X	X	X		

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
howMessageWhenOpeningSitesInInternetExplorer	Specify whether users should see a full interstitial page in Microsoft Edge when opening sites that are configured to open in Internet Explorer using the Enterprise Site list.	X				
SyncFavoritesBetweenIEAndMicrosoftEdge	Specify whether favorites are kept in sync between Internet Explorer and Microsoft Edge.	X				

## Camera

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
AllowCamera	Disable or enable the camera.	X	X	X		

## Connectivity

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
AllowBluetooth	Allow the user to enable Bluetooth or restrict access.	X	X	X	X	
AllowCellularData	Allow the cellular data channel on the device.	X	X	X		
AllowCellularDataRoaming	Allow or disallow cellular data roaming on the device.	X	X	X		

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
AllowConnectedDevices	Allows IT admins the ability to disable the Connected Devices Platform component.	X	X	X		
AllowNFC	Allow or disallow near field communication (NFC) on the device.		X			
AllowUSBConnection	Enable USB connection between the device and a computer to sync files with the device or to use developer tools or to deploy or debug applications.		X			
AllowVPNOverCellular	Specify what type of underlying connections VPN is allowed to use.	X	X	X		
AllowVPNRoamingOverCellular	Prevent the device from connecting to VPN when the device roams over cellular networks.	X	X	X		
HideCellularConnectionMode	Hide the checkbox that lets the user change the connection mode.	X	X	X		

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	Iot Core
HideCellularRoamingOption	Hide the dropdown menu that lets the user change the roaming preferences.	X	X	X		

## CredentialProviders

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	Iot Core
DisableAutomaticReDeploymentCredentials	This setting disables the visibility of the credential provider that triggers the PC refresh on a device. This policy does not actually trigger the refresh. The admin user is required to authenticate to trigger the refresh on the target device. The Windows 10 Automatic ReDeployment feature allows admin to reset devices to a known good managed state while preserving the management enrollment. After the automatic redeployment is triggered the devices are ready for use by information workers or students.	X				

## Cryptography

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IoT Core
AllowFipsAlgorithmPolicy	Allow or disallow the Federal Information Processing Standard (FIPS) policy.	X	X			
TLSCipherSuites	List the Cryptographic Cipher Algorithms allowed for SSL connections. Format is a semicolon delimited list. Last write win.	X	X			

## Defender

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IoT Core
AllowArchiveScanning	Allow or disallow scanning of archives.	X				
AllowBehaviorMonitoring	Allow or disallow Windows Defender Behavior Monitoring functionality.	X				

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IoT Core
AllowCloudProtection	To best protect your PC, Windows Defender will send information to Microsoft about any problems it finds. Microsoft will analyze that information, learn more about problems affecting you and other customers, and offer improved solutions.	X				
AllowEmailScanning	Allow or disallow scanning of email.	X				
AllowFullScanOnMappedNetworkDrives	Allow or disallow a full scan of mapped network drives.	X				
AllowFullScanRemovableDriveScanning	Allow or disallow a full scan of removable drives.	X				
AllowIntrusionPreventionSystem	Allow or disallow Windows Defender Intrusion Prevention functionality.	X				
AllowIOAVProtection	Allow or disallow Windows Defender IOAVP Protection functionality.	X				

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IoT Core
AllowOnAccessProtection	Allow or disallow Windows Defender On Access Protection functionality.	X				
AllowRealtimeMonitoring	Allow or disallow Windows Defender Realtime Monitoring functionality.	X				
AllowScanningNetworkFiles	Allow or disallow scanning of network files.	X				
AllowScriptScanning	Allow or disallow Windows Defender Script Scanning functionality.	X				
AllowUserUIAccess	Allow or disallow user access to the Windows Defender UI.	X				
AvgCPULoadFactor	Represents the average CPU load factor for the Windows Defeder scan (in percent).	X				
DaysToRetainCleanedMalware	Specify time period (in days) that quarantine items will be stored on the system.	X				

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IoT Core
ExcludedExtensions	Specify a list of file type extensions to ignore during a scan. Separate each file type in the list by using  .	X				
ExcludedPaths	Specify a list of directory paths to ignore during a scan. Separate each path in the list by using  .	X				
ExcludedProcesses	Specify a list of files opened by processes to ignore during a scan. Separate each file type in the list by using  . The process itself is not excluded from the scan, but can be excluded by using the <a href="#">Defender/ExcludedPaths</a> policy to exclude its path.	X				
RealTimeScanDirection	Control which sets of files should be monitored.	X				
ScanParameter	Select whether to perform a quick scan or full scan.	X				
ScheduleQuickScanTime	Specify the time of day that Windows Defender quick scan should run.	X				

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	Iot Core
ScheduleScanDay	Select the day that Windows Defender scan should run.	X				
ScheduleScanTime	Select the time of day that the Windows Defender scan should run.	X				
SignatureUpdateInterval	Specify the interval (in hours) that will be used to check for signatures, so instead of using the ScheduleDay and ScheduleTime the check for new signatures will be set according to the interval.	X				
SubmitSamplesConsent	Checks for the user consent level in Windows Defender to send data.	X				
ThreatSeverityDefaultAction	Specify any valid threat severity levels and the corresponding default action ID to take.	X				

## Delivery Optimization

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	Iot Core
DOAbsoluteMaxCacheSize	Specify the maximum size in GB of Delivery Optimization cache.	X				

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
DOAllowVPNPeerCaching	Specify whether the device is allowed to participate in Peer Caching while connected via VPN to the domain network.	X				
DODownloadMode	Specify the download method that Delivery Optimization can use in downloads of Windows Updates, apps, and app updates.	X				
DOGroupId	Specify an arbitrary group ID that the device belongs to.	X				
DOMaxCacheAge	Specify the maximum time in seconds that each file is held in the Delivery Optimization cache after downloading successfully.	X				
DOMaxCacheSize	Specify the maximum cache size that Delivery Optimization can utilize, as a percentage of disk size (1-100).	X				

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
DOMaxDownloadBandwidth	Specify the maximum download bandwidth in kilobytes/second that the device can use across all concurrent download activities using Delivery Optimization.	X				
DOMaxUploadBandwidth	Specify the maximum upload bandwidth in kilobytes/second that a device will use across all concurrent upload activity usinng Delivery Optimization.	X				
DOMinBackgroundQos	Specify the minimum download QoS (Quality of Service or speed) in kilobytes/second for background downloads.	X				
DOMinBatteryPercentageAllowedToUpload	Specify any value between 1 and 100 (in percentage) to allow the device to upload data to LAN and group peers while on battery power.	X				
DOMinDiskSizeAllowedToPeer	Specify the required minimum disk size (capability in GB) for the device to use Peer Caching.	X				

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
<a href="#">DOMinFileSizeToCache</a>	Specify the minimum content file size in MB enabled to use Peer Caching.	X				
<a href="#">DOMinRAMAllowedToPeer</a>	Specify the minimum RAM size in GB required to use Peer Caching.	X				
<a href="#">DOModifyCacheDrive</a>	Specify the drive that Delivery Optimization should use for its cache.	X				
<a href="#">DOMonthlyUploadDataCap</a>	Specify the maximum total bytes in GB that Delivery Optimization is allowed to upload to Internet peers in each calendar month.	X				
<a href="#">DOPercentageMaxDownloadBandwidth</a>	Specify the maximum download bandwidth that Delivery Optimization uses across all concurrent download activities as a percentage of available download bandwidth.	X				

## DeviceGuard

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	Iot Core
EnableVirtualizationBasedSecurity	Turns on virtualization based security(VBS) at the next reboot. virtualization based security uses the Windows Hypervisor to provide support for security services.	X				

## DeviceLock

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	Iot Core
AllowIdleReturnWithoutPassword	Specify whether the user must input a PIN or password when the device resumes from an idle state.		X			
AllowScreenTimeoutWhileLockedUserConfig	Specify whether to show a user-configurable setting to control the screen timeout while on the lock screen.		X			
AllowSimpleDevicePassword	Specify whether PINs or passwords such as "1111" or "1234" are allowed. For the desktop, it also controls the use of picture passwords.	X	X			

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
AlphanumericDevicePasswordRequired	Select the type of PIN or password required.	X	X			
DevicePasswordEnabled	Specify whether device password is enabled.	X	X			
DevicePasswordExpiration	Specify when the password expires (in days).	X	X			
DevicePasswordHistory	Specify how many passwords can be stored in the history that can't be reused.	X	X			
MaxDevicePasswordFailedAttempts	Specify the number of authentication failures allowed before the device will be wiped.	X	X			
MaxInactivityTimeDeviceLock	Specify the maximum amount of time (in minutes) allowed after the device is idle that will cause the device to become PIN or password locked.	X	X			
MinDevicePasswordComplexCharacters	Specify the number of complex element types (uppercase and lowercase letters, numbers, and punctuation) required for a strong PIN or password.	X	X			

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	Iot Core
MinDevicePasswordLength	Specify the minimum number or characters required in the PIN or password.	X	X			
ScreenTimeoutWhileLocked	Specify the duration in seconds for the screen timeout while on the lock screen.		X			

## DeviceManagement

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	Iot Core
DisableMDMEnrollment	Use this setting to prevent the device from enrolling in MDM.	X				

## Experience

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	Iot Core
AllowCopyPaste	Specify whether copy and paste is allowed.		X			
AllowCortana	Specify whether Cortana is allowed on the device.	X	X			
AllowDeviceDiscovery	Allow users to turn device discovery on or off in the UI.	X	X			
AllowFindMyDevice	Turn on <b>Find my device</b> feature.	X	X			

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
AllowManualMDMUnenrollment	Specify whether the user is allowed to delete the workplace account.	X	X			
AllowScreenCapture	Specify whether screen capture is allowed.		X			
AllowSIMErrorDialogPromptWhenNoSIM	Specify whether to display a dialog prompt when no SIM card is detected.		X			
AllowSyncMySettings	Allow or disallow all Windows sync settings on the device.	X	X			
AllowTailoredExperiencesWithDiagnosticData	Prevent Windows from using diagnostic data to provide customized experiences to the user.	X				
AllowTaskSwitcher	Allow or disallow task switching on the device.		X			
AllowThirdPartySuggestionsInWindowsSpotlight	Specify whether to allow app and content suggestions from third-party software publishers in Windows Spotlight.	X				
AllowVoiceRecording	Specify whether voice recording is allowed for apps.		X			

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	Iot Core
AllowWindowsConsumerFeatures	Turn on experiences that are typically for consumers only, such as Start suggestions, membership notifications, post-OOBE app install, and redirect tiles.	X				
AllowWindowsSpotlight	Specify whether to turn off all Windows Spotlight features at once.	X				
AllowWindowsSpotlightOnActionCenter	Prevent Windows Spotlight notifications from being displayed in the Action Center.	X				
AllowWindowsSpotlightWindowsWelcomeExperience	Turn off the Windows Spotlight Windows welcome experience feature.	X				
AllowWindowsTips	Enable or disable Windows Tips.	X				
ConfigureWindowsSpotlightOnLockScreen	Specify whether Spotlight should be used on the user's lock screen.	X				

## ExploitGuard

SETTING	DESCRIPTION	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
ExploitProtectionSettings	See the <a href="#">explanation of ExploitProtectionSettings</a> in the Policy CSP for instructions. In the <b>ExploitProtectionSettings</b> field, you can enter a path (local, UNC, or URI) to the mitigation options config, or you can enter the XML for the config.	X	X			

## Games

SETTING	DESCRIPTION	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
AllowAdvancedGamingServices	Currently not supported.	X				

## Location

SETTING	DESCRIPTION	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
EnableLocation	Configure whether the Location Service's Device Switch is enabled or disabled for the device.	X	X			

## Privacy

SETTING	DESCRIPTION	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	Iot Core
AllowAutoAcceptPairingAndPrivacyConsentPrompts	Allow or disallow the automatic acceptance of the pairing and privacy user consent dialog boxes when launching apps.		X			
AllowInputPersonalization	Allow the use of cloud-based speech services for Cortana, dictation, or Store apps.	X	X			

## Search

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	Iot Core
AllowIndexingEncryptedStoresOrItems	Allow or disallow the indexing of items.	X	X			
AllowSearchToUseLocation	Specify whether search can use location information.	X	X			
AllowUsingDiacritics	Allow the use of diacritics.	X	X			

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
AllowWindowsIndexer	<p>The indexer provides fast file, email, and web history search for apps and system components including Cortana, Outlook, file explorer, and Edge. To do this, it requires access to the file system and app data stores such as Outlook OST files.</p> <ul style="list-style-type: none"> <li>- <b>Off</b> setting disables Windows indexer</li> <li>-</li> <li><b>EnterpriseSecure</b> setting stops the indexer from indexing encrypted files or stores, and is recommended for enterprises using Windows Information Protection (WIP)</li> <li>- <b>Enterprise</b> setting reduces potential network loads for enterprises</li> <li>- <b>Standard</b> setting is appropriate for consumers</li> </ul>	X	X			

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
AlwaysUseAutomaticLangDetection	Specify whether to always use automatic language detection when indexing content and properties.	X	X			
DisableBackoff	If enabled, the search indexer backoff feature will be disabled.	X	X			
DisableRemovableDriveIndexing	Configure whether locations on removable drives can be added to libraries.	X	X			
PreventingLowDiskSpaceMB	Prevent indexing from continuing after less than the specified amount of hard drive space is left on the same drive as the index location.	X	X			
PreventRemoteQueries	If enabled, clients will be unable to query this device's index remotely.	X	X			
SafeSearchPermissions	Specify the level of safe search (filtering adult content) required.		X			

## Security

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IoT Core
AllowAddProvisioningPackage	Specify whether to allow installation of provisioning packages.	X	X	X	X	X
AllowManualRootCertificateInstallation	Specify whether the user is allowed to manually install root and intermediate CA certificates.		X			
AllowRemoveProvisioningPackage	Specify whether removal of provisioning packages is allowed.	X	X	X	X	X
AntiTheftMode	Allow or disallow Anti Theft Mode on the device.		X			
RequireDeviceEncryption	Specify whether encryption is required.	X	X	X	X	X
RequireProvisioningPackageSignature	Specify whether provisioning packages must have a certificate signed by a device-trusted authority.	X	X	X	X	X

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	Iot Core
RequireRetrieveHealthCertificateOnBoot	Specify whether to retrieve and post TCG Boot logs, and get or cache an encrypted or signed Health Attestation Report from the Microsoft Health Attestation Service when a device boots or reboots.	X	X			

## Settings

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	Iot Core
AllowAutoPlay	Allow the user to change AutoPlay settings.		X			
AllowDataSense	Allow the user to change Data Sense settings.		X			
AllowVPN	Allow the user to change VPN settings.		X			
ConfigureTaskbarCalendar	Configure the default setting for showing additional calendars (besides the default calendar for the locale) in the taskbar clock and calendar flyout.	X				

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	Iot Core
PageVisibilityList	Allows IT admins to prevent specific pages in the System Settings app from being visible or accessible. Pages are identified by a shortened version of their already published URIs, which is the URI minus the "ms-settings:" prefix. For example, if the URI for a settings page is "ms-settings:foo", the page identifier used in the policy will be just "foo". Multiple page identifiers are separated by semicolons.	X				

## Start

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	Iot Core
AllowPinnedOlderDocuments	Control the visibility of the Documents shortcut on the Start menu.	X				
AllowPinnedOlderDownloads	Control the visibility of the Downloads shortcut on the Start menu.	X				

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IoT Core
AllowPinnedFolderFileExplorer	Control the visibility of the File Explorer shortcut on the Start menu.	X				
AllowPinnedFolderHomeGroup	Control the visibility of the Home Group shortcut on the Start menu.	X				
AllowPinnedFolderMusic	Control the visibility of the Music shortcut on the Start menu.	X				
AllowPinnedFolderNetwork	Control the visibility of the Network shortcut on the Start menu.	X				
AllowPinnedFolderPersonalFolder	Control the visibility of the Personal Folder shortcut on the Start menu.	X				
AllowPinnedFolderPictures	Control the visibility of the Pictures shortcut on the Start menu.	X				
AllowPinnedFolderSettings	Control the visibility of the Settings shortcut on the Start menu.	X				
AllowPinnedFolderVideos	Control the visibility of the Videos shortcut on the Start menu.	X				

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
ForceStartSize	Force the size of the Start screen.	X				
HideAppList	Collapse or remove the all apps list.	X				
HideChangeAccountSettings	Hide <b>Change account settings</b> from appearing in the user tile.	X				
HideFrequentlyUsedApps	Hide <b>Most used</b> section of Start.	X				
HideHibernate	Prevent <b>Hibernate</b> option from appearing in the Power button.	X				
HideLock	Prevent <b>Lock</b> from appearing in the user tile.	X				
HidePeopleBar	Remove the people icon from the taskbar, as well as the corresponding settings toggle. It also prevents users from pinning people to the taskbar.	X				
HidePowerButton	Hide the <b>Power</b> button.	X				
HideRecentJumplists	Hide jumplists of recently opened items.	X				
HideRecentlyAddedApps	Hide <b>Recently added</b> section of Start.	X				

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
HideRestart	Prevent <b>Restart</b> and <b>Update and restart</b> from appearing in the Power button.	X				
HideShutdown	Prevent <b>Shutdown</b> and <b>Update and shut down</b> from appearing in the Power button.	X				
HideSignOut	Prevent <b>Sign out</b> from appearing in the user tile.	X				
HideSleep	Prevent <b>Sleep</b> from appearing in the Power button.	X				
HideSwitchAccount	Prevent <b>Switch account</b> from appearing in the user tile.	X				
HideUserTile	Hide the user tile.	X				
ImportEdgeAssets	Import Edge assets for secondary tiles. For more information, see <a href="#">Add image for secondary Microsoft Edge tiles</a> .	X				
NoPinningToTaskbar	Prevent users from pinning and unpinning apps on the taskbar.	X				

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	Iot Core
StartLayout	Apply a custom Start layout. For more information, see <a href="#">Customize Windows 10 Start and taskbar with provisioning packages</a>	X				

## System

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	Iot Core
AllowBuildPreview	Specify whether users can access the Insider build controls in the <b>Advanced Options</b> for Windows Update.	X	X			
AllowEmbeddedMode	Specify whether to set general purpose device to be in embedded mode.	X	X	X	X	X
AllowExperimentation	Determine the level that Microsoft can experiment with the product to study user preferences or device behavior.	X	X			
AllowLocation	Specify whether to allow app access to the Location service.	X	X	X	X	X

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IoT Core
AllowStorageCard	Specify whether the user is allowed to use the storage card for device storage.	X	X	X	X	X
AllowTelemetry	Allow the device to send diagnostic and usage telemetry data.	X	X			
AllowUserToResetPhone	Allow the user to factory reset the phone.	X	X			
DisableOneDriveFileSync	Prevent apps and features from working with files on OneDrive.	X				
LimitEnhancedDiagnosticDataWindowsAnalytics	This policy setting, in combination with the System/AllowTelemetry policy setting, enables organizations to send Microsoft a specific set of diagnostic data for IT insights via Windows Analytics services. To enable this behavior you must enable this policy setting, and set Allow Telemetry to level 2 (Enhanced). When you configure these policy settings, a basic level of diagnostic	X	X			

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IoT Core
	<p>data plus additional events that are required for Windows Analytics are sent to Microsoft. These events are documented in <a href="#">Windows 10, version 1703 basic level Windows diagnostic events and fields</a>. Enabling enhanced diagnostic data in the System/AllowTelemetry policy in combination with not configuring this policy will also send the required events for Windows Analytics, plus additional enhanced level telemetry data. This setting has no effect on computers configured to send full, basic or security level diagnostic data to Microsoft. If you disable or do not configure this policy setting, then the level of diagnostic data sent to Microsoft is determined by the System/AllowTelemetry policy.</p>					

## TextInput

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
AllowIMELogging	Allow the user to turn on and off the logging for incorrect conversion and saving auto-tuning result to a file and history-based predictive input.	X				
AllowIMENetworkAccess	Allow the user to turn on Open Extended Dictionary, Internet search integration, or cloud candidate features to provide input suggestions that do not exist in the device's local dictionary.	X				
AllowInputPanel	Disable the touch/handwriting keyboard.	X				
AllowJapaneseIMESurrogatePairCharacters	Allow the Japanese IME surrogate pair characters.	X				
AllowJapaneseIVSCharacters	Allow Japanese Ideographic Variation Sequence (IVS) characters.	X				
AllJapaneseNonPublishingStandardGlyph	All the Japanese non-publishing standard glyph.	X				
AllowJapaneseUserDictionary	Allow the Japanese user dictionary.	X				

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
AllowKeyboardTextSuggestions	Specify whether text prediction is enabled or disabled for the on-screen keyboard, touch keyboard, and handwriting recognition tool.	X				
AllowLanguageFeaturesUninstall	All language features to be uninstalled.	X				
AllowUserInputsFromMiracastReceiver	Do not use. Instead, use <a href="#">WirelessDisplay/AllowUserInputFromWirelessDisplayReceiver</a>					
ExcludeJapaneseIMEExceptISO208	Allow users to restrict character code range of conversion by setting the character filter.	X				
ExcludeJapaneseIMEExceptISO208andEUDC	Allow users to restrict character code range of conversion by setting the character filter.	X				
ExcludeJapaneseIMEExceptShiftJIS	Allow users to restrict character code range of conversion by setting the character filter.	X				

## TimeLanguageSettings

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	Iot Core
AllowSet24HourClock	Configure the default clock setting to be the 24 hour format.		X			

## Update

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	Iot Core
ActiveHoursEnd	Use with <b>Update/ActiveHoursStart</b> to manage the range of active hours where update reboots are not scheduled.	X	X	X	X	X
ActiveHoursMaxRange	Specify the maximum active hours range.	X	X	X	X	X
ActiveHoursStart	Use with <b>Update/ActiveHoursEnd</b> to manage the range of active hours where update reboots are not scheduled.	X	X	X	X	X
AllowautoUpdate	Configure automatic update behavior to scan, download, and install updates.	X	X	X	X	X
AllowAutoWindowsUpdateDownloadOverMeteredNetwork	Option to download updates automatically over metered connections (off by default). Enter <input type="checkbox"/> for not allowed, or <input checked="" type="checkbox"/> for allowed.	X	X	X	X	X

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IoT Core
AllowMUUpdateService	Manage whether to scan for app updates from Microsoft Update.	X	X	X	X	X
AllowNonMicrosoftSignedUpdate	Manage whether Automatic Updates accepts updates signed by entities other than Microsoft when the update is found at the UpdateService Url location.	X	X	X	X	X
AllowUpdateService	Specify whether the device can use Microsoft Update, Windows Server Update Services (WSUS), or Microsoft Store.	X	X	X	X	X
AutoRestartDeadlinePeriodInDays	Specify number of days (between 2 and 30) after which a forced restart will occur outside of active hours when restart is pending.	X	X	X	X	X
AutoRestartNotificationSchedule	Specify the period for auto-restart reminder notifications.	X	X	X	X	X

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IoT Core
AutoRestartRequiredNotificationDismissal	Specify the method by which the auto-restart required notification is dismissed.	X	X	X	X	X
BranchReadinessLevel	Select which branch a device receives their updates from.	X	X	X	X	X
DeferFeatureUpdatesPeriodInDays	Defer Feature Updates for the specified number of days.	X	X	X	X	X
DeferQualityUpdatesPeriodInDays	Defer Quality Updates for the specified number of days.	X	X	X	X	X
DeferUpdatePeriod	Specify update delays for up to 4 weeks.	X	X	X	X	X
DeferUpgradePeriod	Specify upgrade delays for up to 8 months.	X	X	X	X	X
DetectionFrequency	Specify the frequency to scan for updates, from every 1-22 hours.	X	X	X	X	X
DisableDualScan	Do not allow update deferral policies to cause scans against Windows Update.	X	X	X	X	X

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IoT Core
EngagedRestartDeadline	Specify the deadline in days before automatically scheduling and executing a pending restart outside of active hours.	X	X	X	X	X
EngagedRestartSnoozeSchedule	Specify the number of days a user can snooze Engaged restart reminder notifications.	X	X	X	X	X
EngagedRestartTransitionSchedule	Specify the timing before transitioning from Auto restarts scheduled outside of active hours to Engaged restart, which requires the user to schedule.	X	X	X	X	X
FillEmptyContentUrls	Allow Windows Update Agent to determine the download URL when it is missing from the metadata.	X	X	X	X	X
ManagePreviewBuilds	Use to enable or disable preview builds.	X	X	X	X	X
PhoneUpdateRestrictions	Deprecated		X			

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IoT Core
RequireDeferUpgrade	Configure device to receive updates from Current Branch for Business (CBB).	X	X	X	X	X
ScheduledInstallAllDay	Schedule the day for update installation.	X	X	X	X	X
ScheduledInstallEveryWeek	To schedule update installation every week, set the value as <a href="#">1</a> .	X	X	X	X	X
ScheduledInstallFirstWeek	To schedule update installation the first week of the month, see the value as <a href="#">1</a> .	X	X	X	X	X
ScheduledInstallFourthWeek	To schedule update installation the fourth week of the month, see the value as <a href="#">1</a> .	X	X	X	X	X
ScheduledInstallSecondWeek	To schedule update installation the second week of the month, see the value as <a href="#">1</a> .	X	X	X	X	X
ScheduledInstallThirdWeek	To schedule update installation the third week of the month, see the value as <a href="#">1</a> .	X	X	X	X	X

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IoT Core
ScheduledInstallTime	Schedule the time for update installation.	X	X	X	X	X
ScheduleImminentRestartWarning	Specify the period for auto-restart imminent warning notifications.	X	X	X	X	X
ScheduleRestartWarning	Specify the period for auto-restart warning reminder notifications.	X	X	X	X	X
SetAutoRestartNotificationDisable	Disable auto-restart notifications for update installations.	X	X	X	X	X
SetEDURestart	Skip the check for battery level to ensure that the reboot will happen at ScheduledInstallTime.	X	X	X	X	X
UpdateServiceUrl	Configure the device to check for updates from a WSUS server instead of Microsoft Update.	X	X	X	X	X
UpdateServiceUrlAlternate	Specify an alternate intranet server to host updates from Microsoft Update.	X	X	X	X	X

## WiFi

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	Iot Core
AllowAutoConnectToWiFiSenseHotspots	Allow the device to connect automatically to Wi-Fi hotspots.	X	X			
AllowInternetSharing	Allow Internet sharing.	X	X			
AllowManualWiFiConfiguration	Allow connecting to Wi-Fi outside of MDM server-installed networks.		X			
AllowWiFi	Allow Wi-Fi connections.		X			
WLANScanMode	Configure the WLAN scanning behavior and how aggressively devices should be actively scanning for Wi-Fi networks to get devices connected.	X	X	X	X	X

## WindowsInkWorkspace

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	Iot Core
AllowSuggestedAppsInWindowsInkWorkspace	Show recommended app suggestions in the ink workspace.	X				
AllowWindowsInkWorkspace	Specify whether to allow the user to access the ink workspace.	X				

## WindowsLogon

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
HideFastUserSwitching	Hide the <b>Switch account</b> button on the sign-in screen, Start, and the Task Manager.	X				

## WirelessDisplay

Setting	Description	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
AllowUserInputFromWirelessDisplayReceiver	This policy controls whether or not the wireless display can send input (keyboard, mouse, pen, and touch, dependent upon display support) back to the source device. For example, a Surface Laptop is projecting wirelessly to a Surface Hub. If input from the wireless display receiver is allowed, users can draw with a pen on the Surface Hub.	X	X			

# ProvisioningCommands (Windows Configuration Designer reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Use ProvisioningCommands settings to install Classic Windows apps using a provisioning package.

## Applies to

Setting	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
All settings	X				

For instructions on adding apps to provisioning packages, see [Provision PCs with apps](#).

# SharedPC (Windows Configuration Designer reference)

10/17/2017 • 4 min to read • [Edit Online](#)

Use SharedPC settings to optimize Windows 10 for shared use scenarios, such as touchdown spaces in an enterprise and temporary customer use in retail.

## Applies to

Setting	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IoT Core
All settings	X				

## Account Management

Use these settings to configure settings for accounts allowed on the shared PC.

Setting	Value	Description
AccountModel	<ul style="list-style-type: none"><li>- Only guest</li><li>- Domain-joined only</li><li>- Domain-joined and guest</li></ul>	This option controls how users can sign-in on the PC. Choosing domain-joined will enable any user in the domain to sign-in. Specifying the guest option will add the Guest option to the sign-in screen and enable anonymous guest access to the PC. <ul style="list-style-type: none"><li>- Only guest allows anyone to use the PC as a local standard (non-admin) account.</li><li>- Domain-joined only allows users to sign in with an Active Directory or Azure AD account.</li><li>- Domain-joined and guest allows users to sign in with an Active Directory, Azure AD, or local standard account.</li></ul>
DeletionPolicy	<ul style="list-style-type: none"><li>- Delete immediately</li><li>- Delete at disk space threshold</li><li>- Delete at disk space threshold and inactive threshold</li></ul>	<ul style="list-style-type: none"><li>- Delete immediately will delete the account on sign-out.</li><li>- Delete at disk space threshold will start deleting accounts when available disk space falls below the threshold you set for DiskLevelDeletion, and it will stop deleting accounts when the available disk space reaches the threshold you set for DiskLevelCaching. Accounts are deleted in order of oldest accessed to most recently accessed.</li><li>- Delete at disk space threshold and inactive threshold will apply the same disk space checks as noted above, but also delete accounts if they have not signed in within the number of days specified by InactiveThreshold</li></ul>

SETTING	VALUE	DESCRIPTION
DiskLevelCaching	A number between 0 and 100	If you set <b>DeletionPolicy</b> to <b>Delete at disk space threshold</b> , set the percent of total disk space to be used as the disk space threshold for account caching.
DiskLevelDeletion	A number between 0 and 100	If you set <b>DeletionPolicy</b> to <b>Delete at disk space threshold</b> , set the percent of total disk space to be used as the disk space threshold for account deletion.
EnableAccountManager	True or false	Set as <b>True</b> to enable automatic account management. If this is not set to true, no automatic account management will be done.
InactiveThreshold	Number	If you set <b>DeletionPolicy</b> to <b>Delete at disk space threshold and inactive threshold</b> , set the number of days after which an account that has not signed in will be deleted.
KioskModeAUMID	String	Set an Application User Model ID (AUMID) to enable the kiosk account on the sign-in screen. A new account will be created and will use assigned access to only run the app specified by the AUMID. Note that the app must be installed on the PC. Set the name of the account using <b>KioskModeUserTileDisplayText</b> , or a default name will be used. <a href="#">Find the Application User Model ID of an installed app</a>
KioskModeUserTileDisplayText	String	Sets the display text on the kiosk account if <b>KioskModeAUMID</b> has been set.

## EnableSharedPCMode

Set as **True**. If this is not set to **True**, shared PC mode is not turned on and none of the other settings apply. This setting controls this API: [IsEnabled](#).

Some of the remaining settings in SharedPC are optional, but we strongly recommend that you also set **EnableAccountManager** to **True**.

## PolicyCustomization

Use these settings to configure policies for shared PC mode.

SETTING	VALUE	DESCRIPTION
MaintenanceStartTime	A number between 0 and 1440	By default, the maintenance start time (which is when automatic maintenance tasks run, such as Windows Update) is midnight. You can adjust the start time in this setting by entering a new start time in minutes from midnight. For example, if you want maintenance to begin at 2 AM, enter <code>120</code> as the value.
MaxPageFileSizeMB	A number between 1024 and 2048	Adjusts the maximum page file size in MB. This can be used to fine-tune page file behavior, especially on low end PCs.
RestrictLocalStorage	True or false	Set as <b>True</b> to restrict the user from saving or viewing local storage when using File Explorer. This setting controls this API: <a href="#">ShouldAvoidLocalStorage</a>
SetEduPolicies	True or false	Set to <b>True</b> for PCs that will be used in a school. For more information, see <a href="#">Windows 10 configuration recommendations for education customers</a> . This setting controls this API: <a href="#">IsEducationEnvironment</a>
SetPowerPolicies	True or false	When set as <b>True</b> : - Prevents users from changing power settings - Turns off hibernate - Overrides all power state transitions to sleep (e.g. lid close)
SignInOnResume	True or false	This setting specifies if the user is required to sign in with a password when the PC wakes from sleep.
SleepTimeout	Number	Specifies all timeouts for when the PC should sleep. Enter the amount of idle time in seconds. If you don't set sleep timeout, the default of 1 hour applies.

## Related topics

- [Set up shared or guest PC](#)

Not finding content you need? Windows 10 users, tell us what you want on [Feedback Hub](#).

# Shell (Windows Configuration Designer reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Do not use. Use [Start > StartLayout](#)

## Applies to

SETTING	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
All settings		X			

# SMISettings (Windows Configuration Designer reference)

9/6/2017 • 6 min to read • [Edit Online](#)

Use SMISettings settings to customize the device with custom shell, suppress Windows UI during boot and sign-in, and block or allow specific keys.

## Applies to

Setting	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
All settings	X				

## All settings in SMISettings

The following table describes the settings in SMISettings. Some settings have additional details in sections after the table.

Setting	Value	Description
AutoLogon	Enable Domain name Password UserName	Allows automatic sign-in at startup so that the user does not need to enter a user name and password.
BrandingNeutral	See <a href="#">BrandingNeutral values</a>	Specifies which UI elements display on the Welcome screen.
CrashDumpEnabled	See <a href="#">CrashDumpEnabled values</a>	Specifies the type of information to be saved in the event of a crash.
DisableBootMenu	True or false	Disables the F8 and F10 keys during startup to prevent access to the <b>Advanced Startup Options</b> menu.
DisplayDisabled	True or false	Configures the device to display a blank screen when the OS encounters an error that it cannot recover from.
HideAllBootUI	True or false	Suppresses all Windows UI elements (logo, status indicator, and status message) during startup.
HideAutologonUI	True or false	Hides the Welcome screen when automatic sign-in (AutoLogon) is enabled.
HideBootLogo	True or false	Suppresses the default Windows logo that displays during the OS loading phase.

SETTING	VALUE	DESCRIPTION
HideBootStatusIndicator	True or false	Suppresses the status indicator that displays during the OS loading phase.
HideBootStatusMessages	True or false	Suppresses the startup status text that displays during the OS loading phase.
HideFirstLogonAnimation	True or false	Disable the animation during the first sign-in.
KeyboardFilter	See <a href="#">KeyboardFilter settings</a>	Use these settings to configure devices to suppress key presses or key combinations.
NoLockScreen	True or false	Disables the lock screen functionality and UI elements
ShellLauncher	See <a href="#">ShellLauncher settings</a>	Settings used to specify the application or executable to use as the default custom shell.
UIVerbosityLevel	Suppress or do not suppress	Disables the Windows status messages during device startup, sign-in, and shut down.

## BrandingNeutral values

The following table shows the possible values. You can combine these values using bitwise exclusive-OR logic to disable multiple Welcome screen UI elements.

The default value is **17**, which disables all Welcome screen UI elements and the Switch user button.

VALUE	DESCRIPTION
1	Disables all Welcome screen UI elements
2	Disables the Power button
4	Disables the Language button
8	Disables the Ease of access button
16	Disables the Switch user button
32	Disables the blocked shutdown resolver (BSDR) screen so that restarting or shutting down the system causes the OS to immediately force close any applications that are blocking system shut down. No UI is displayed and users are not given a chance to cancel the shutdown process. This can result in a loss of data if any open applications have unsaved data.

## CrashDumpEnabled values

Contains an integer that specifies the type of information to capture in a dump (.dmp) file that is generated when

the system stops unexpectedly.

The .dmp file is typically saved in %SystemRoot% as Memory.dmp.

Set CrashDumpEnabled to one of the following values:

VALUE	DESCRIPTION
1	Records all the contents of system memory. This dump file may contain data from processes that were running when the information was collected.
2	Records only the kernel memory. This dump file includes only memory that is allocated to the kernel, kernel-mode drivers, and other kernel-mode programs. It does not include unallocated memory or any memory that is allocated to user-mode programs. For most purposes, this kind of dump file is the most useful because it is significantly smaller than the complete memory dump file, but it contains information that is most likely to have been involved in the issue.  If a second problem occurs, the dump file is overwritten with new information.
3	Records the smallest amount of useful information that may help identify why the device stopped unexpectedly. This type of dump file includes the following information: <ul style="list-style-type: none"><li>- A list of loaded drivers</li><li>- The processor context (PRCB) for the processor that stopped</li><li>- The process information and kernel context (EPROCESS) for the process that stopped</li><li>- The process information and kernel context (ETHREAD) for the thread that stopped</li><li>- The kernel-mode call stack for the thread that stopped</li></ul> This kind of dump file can be useful when space is limited. However, because of the limited information included, errors that were not directly caused by the thread that was running at the time of the problem may not be discovered by analyzing this file.
4	The date is encoded in the file name. If a second problem occurs, the previous file is preserved and the new file is given a distinct name. A list of all small memory dump files is kept in the %SystemRoot%\Minidump folder.
7	Records only the kernel memory. This value produces the same results as entering a value of 2. This is the default value.
Any other value	Disables crash dump and does not record anything.

## KeyboardFilter settings

You can use KeyboardFilter to suppress undesirable key presses or key combinations. KeyboardFilter works with physical keyboards, the Windows on-screen keyboard, and the touch keyboard.

When you **enable** KeyboardFilter, a number of other settings become available for configuration.

SETTING	VALUE	DESCRIPTION
CustomKeyFilters	Allow or block	Add your own key filters to meet any special requirements that you may have that are not included in the predefined key filters. Enter a custom key combination in <b>CustomKeyFilter</b> , and then select it to allow or block it. The format to add custom filter combinations is "Alt+F9." This also appears as the CustomKey name, which is specified without "+". For more information, see <a href="#">WEKF_CustomKey</a> .
CustomScancodeFilters	Allow or block	Blocks the list of custom scan codes. When a key is pressed on a physical keyboard, the keyboard sends a scan code to the keyboard driver. The driver then sends the scan code to the OS and the OS converts the scan code into a virtual key based on the current active layout. Enter a custom scancode in <b>CustomScancodeFilter</b> , and then select it to allow or block it. For more information, see <a href="#">WEKF_Scancode</a> .
DisableKeyboardFilterForAdministrators	True or false	Disables the keyboard filter for administrators.
ForceOffAccessibility	True or false	Disables all Ease of Access features and prevents users from enabling them.
PredefinedKeyFilters	Allow or block	Specifies the list of predefined keys. For each key, the value will default to <b>Allow</b> . Specifying <b>Block</b> will suppress the key combination.

[Learn more about using keyboard filters.](#)

## ShellLauncher settings

Use ShellLauncher to specify the application or executable to use as the default custom shell. One use of ShellLauncher is to [create a kiosk \(fixed-purpose\) device running a Classic Windows application](#).

You can also configure ShellLauncher to launch different shell applications for different users or user groups.

**IMPORTANT**

You may specify any executable file to be the default shell except C:\Windows\System32\Eshell.exe. Using Eshell.exe as the default shell will result in a blank screen after a user signs in.

You cannot use ShellLauncher to launch a Windows app as a custom shell. However, you can use Windows 10 application launcher to launch a Windows app at startup.

ShellLauncher processes the Run and RunOnce registry keys before starting the custom shell, so your custom shell doesn't need to handle the automatic startup of other applications or services. ShellLauncher also handles the behavior of the system when your custom shell exits. You can configure the shell exit behavior if the default behavior does not meet your needs.

**IMPORTANT**

A custom shell is launched with the same level of user rights as the account that is signed in. This means that a user with administrator rights can perform any system action that requires administrator rights, including launching other applications with administrator rights, while a user without administrator rights cannot. If your shell application requires administrator rights and needs to be elevated, and User Account Control (UAC) is present on your device, you must disable UAC in order for ShellLauncher to launch the shell application.

# Start (Windows Configuration Designer reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Use Start settings to apply a customized Start screen to devices.

## Applies to

Setting	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
StartLayout	X	X			
StartLayoutFilePath		X			

### IMPORTANT

The StartLayout setting is available in the advanced provisioning for Windows 10 desktop editions, but should only be used to apply a layout to Windows 10 Mobile devices. For desktop editions, use [Policies > StartLayout](#).

## StartLayout

Use StartLayout to select the LayoutModification.xml file that applies a customized Start screen to a device.

For more information, see [Start layout XML for mobile editions of Windows 10](#) ).

## StartLayoutFilePath

Do not use.

# StartupApp (Windows Configuration Designer reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Use StartupApp settings to configure the default app that will run on start for Windows 10 IoT Core (IoT Core) devices.

## Applies to

SETTING	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
Default					X

Enter the [Application User Model ID \(AUMID\)](#) for the default app.

# StartupBackgroundTasks (Windows Configuration Designer reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Documentation not available at this time.

## Applies to

Setting	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
All settings					X

# SurfaceHubManagement (Windows Configuration Designer reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Use SurfaceHubManagement settings to set the administrator group that will manage a Surface Hub that is joined to the domain.

## IMPORTANT

These settings should be used only in provisioning packages that are applied during OOBE.

## Applies to

SETTING	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
All settings			X		

## GroupName

Enter the group name for the administrators group in Active Directory.

## GroupSid

Enter the SID or the administrators group in Active Directory.

# TabletMode (Windows Configuration Designer reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Use TabletMode to configure settings related to tablet mode.

## Applies to

Setting	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
All settings	X	X	X	X	X

## ConvertibleSlateModePromptPreference

Set the default for hardware-based prompts.

## SignInMode

Specify whether users switch to table mode by default after signing in.

# TakeATest (Windows Configuration Designer reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Use TakeATest to configure the Take A Test app, a secure browser for test-taking. Many schools use online testing for formative and summative assessments. It's critical that students use a secure browser that prevents them from using other computer or Internet resources during the test. For more information, see [Take tests in Windows 10](#).

## Applies to

Setting	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
All settings	X				

## AllowScreenMonitoring

When set to True, students are able to record and take screen captures in the Take A Test app.

## AllowTextSuggestions

When set to True, students can see autofill suggestions from onscreen keyboards when typing in the Take A Test app.

## LaunchURI

Enter a link to an assessment that will be automatically loaded when the Take A Test app is opened.

## RequirePrinting

When set to True, students can print in the Take A Test app.

## TesterAccount

Enter the account to use when taking a test.

To specify a domain account, enter **domain\user**. To specify an AAD account, enter **username@tenant.com**. To specify a local account, enter the username.

## Related topics

- [SecureAssessment configuration service provider \(CSP\)](#)

# TextInput (Windows Configuration Designer reference)

10/17/2017 • 4 min to read • [Edit Online](#)

Use TextInput settings to configure text intelligence and keyboard for mobile devices.

## Applies to

Setting Groups	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
Intelligence > DisablePredictions		X			
PreEnabledKeyboard		X			

## Intelligence

Set **DisablePredictions** to the locale or alternative input language that must have the text intelligence features disabled. For example, to disable text correction and suggestions for English (UK), set the value of **DisablePredictions** to `en-gb`.

## PreEnabledKeyboard

In addition to the automatically-enabled default keyboard, OEMs may choose to pre-enable more keyboards for a particular market.

During phone bring-up, OEMs must set the boot locale, or default locale, for the phone. During first boot, Windows Phone reads the locale setting and automatically enables a default keyboard based on the locale to keyboard mapping table in Set languages and locales.

The mapping works for almost all regions and additional customizations are not needed unless specified in the pre-enabled keyboard column in Set languages and locales. If an OEM chooses to pre-enable more keyboards for a particular market, they can do so by specifying the setting. Pre-enabled keyboards will automatically be enabled during boot. Microsoft recommends that partners limit the number of pre-enabled keyboards to those languages that correspond to the languages spoken within the market.

PreEnabledKeyboard must be entered once for each keyboard you want to pre-enable. As shown below, the format to specify a particular keyboard must be: Locale code.Locale value. See the following table for more information on the locale codes and values that you can use. The setting Value must be set to 1 to enable the keyboard.

The following table shows the values that you can use for the Locale code.Locale value part of the setting name.

### NOTE

The keyboards for some locales require additional language model files: am-ET, bn-IN, gu-IN, hi-IN, ja-JP, kn-IN, ko-KR, ml-IN, mr-IN, my-MM, or-IN, pa-IN, si-LK, ta-IN, te-IN, zh-TW, zh-CN, and zh-HK.

NAME	LOCALE CODE	KEYBOARD LAYOUT VALUE
Afrikaans (South Africa)	af-ZA	1
Albanian	sq-AL	1
Amharic	am-ET	1
Arabic	ar-SA	1
Armenian	hy-AM	1
Assamese - INSCRIPT	as-IN	1
Azerbaijani (Cyrillic)	az-Cyrl-AZ	1
Azerbaijani (Latin)	az-Latn-AZ	1
Bangla (Bangladesh) - 49 key	bn-BD	1
Bangla (India) - INSCRIPT	bn-IN	1
Bangla (India) - Phonetic	bn-IN	2
Bashkir	ba-RU	1
Basque	eu-ES	1
Belarusian	be-BY	1
Bosnian (Cyrillic)	bs-Cyrl-BA	1
Bosnian (Latin)	bs-Latn-BA	1
Bulgarian	bg-BG	1
Catalan	ca-ES	1
Central Kurdish	ku-Arab-IQ	1
Cherokee	chr-Cher-US	1
Chinese Simplified QWERTY	zh-CN	1
Chinese Simplified - 12-key	zh-CN	2
Chinese Simplified - Handwriting	zh-CN	3
Chinese Simplified - Stroke	zh-CN	4
Chinese Traditional (Hong Kong SAR) - Cangjie	zh-HK	1

NAME	LOCALE CODE	KEYBOARD LAYOUT VALUE
Chinese Traditional (Hong Kong SAR) - Quick	zh-HK	2
Chinese Traditional (Hong Kong SAR) - Stroke	zh-HK	3
Chinese Traditional (Taiwan) - BoPoMoFo	zh-TW	1
Chinese Traditional (Taiwan) - Handwriting	zh-TW	2
Croatian	hr-HR	1
Czech	cs-CZ	1
Danish	da-DK	1
Divehi	dv-MV	1
Dutch (Belgium)	nl-BE	1
Dutch (Netherlands)	nl-NL	1
Dzongkha	dz-BT	1
English (Australia)	en-AU	1
English (Canada)	en-CA	1
English (India)	en-IN	1
English (Ireland)	en-IE	1
English (United Kingdom)	en-GB	1
English (United States)	en-US	1
Estonian	et-EE	1
Faroese	fo-FO	1
Filipino	fil-PH	1
Finnish	fi-FI	1
French (Belgium)	fr-BE	1
French (Canada)	fr-CA	1
French (France)	fr-FR	1

NAME	LOCALE CODE	KEYBOARD LAYOUT VALUE
French (Switzerland)	fr-CH	1
Galician	gl-ES	1
Georgian	ka-GE	1
German (Germany)	de-DE	1
German (Switzerland)	de-CH	1
Greek	el-GR	1
Greenlandic	kl-GL	1
Guarani	gn-PY	1
Gujarati - INSCRIPT	gu-IN	1
Gujarati - Phonetic	gu-IN	2
Hausa	ha-Latn-NG	1
Hebrew	he-IL	1
Hindi - 37-key	hi-IN	1
Hindi - INSCRIPT	hi-IN	3
Hindi - Phonetic	hi-IN	2
Hinglish	hi-Latn	1
Hungarian	hu-HU	1
Icelandic	is-IS	1
Igbo	ig-NG	1
Indonesian	id-ID	1
Inuktitut - Latin	iu-Latn-CA	1
Irish	ga-IE	1
Italian	it-IT	1
Japanese - 12-key	ja-JP	1
Japanese - QWERTY	ja-JP	2

NAME	LOCALE CODE	KEYBOARD LAYOUT VALUE
Kannada - INSCRIPT	kn-IN	1
Kannada - Phonetic	kn-IN	2
Kazakh	kk-KZ	1
Khmer	km-KH	1
Kinyarwanda	rw-RW	1
Kiswahili	sw-KE	1
Konkani	kok-IN	1
Korean - 12-key Chunjiin	ko-KR	2
Korean - 12-key Naratgeul	ko-KR	3
Korean - 12-key Sky	ko-KR	4
Korean - QWERTY	ko-KR	1
Kyrgyz	ky-KG	1
Lao	lo-LA	1
Latvian	lv-LV	1
Lithuanian	lt-LT	1
Luxembourgish	lb-LU	1
Macedonian	mk-MK	1
Malay (Brunei Darussalam)	ms-BN	1
Malay (Malaysia)	ms-MY	1
Malayalam - INSCRIPT	ml-IN	1
Malayalam - Phonetic	ml-IN	2
Maltese	mt-MT	1
Maori	mi-NZ	1
Marathi - INSCRIPT	mr-IN	1
Marathi - Phonetic	mr-IN	2

NAME	LOCALE CODE	KEYBOARD LAYOUT VALUE
Mongolian - Cyrillic	mn-MN	1
Mongolian - Traditional Mongolian	mn-Mong-CN	1
Myanmar	my-MM	1
Nepali	ne-NP	1
Norwegian - Bokmal	nb-NO	1
Norwegian - Nynorsk	ny-NO	1
Odia - INSCRIPT	or-IN	1
Odia - Phonetic	or-IN	2
Pashto	ps-AF	1
Persian	fa-IR	1
Polish	pl-PL	1
Portuguese (Brazil)	pt-BR	1
Portuguese (Portugal)	pt-PT	1
Punjabi - INSCRIPT	pa-IN	1
Punjabi - Phonetic	pa-IN	2
Romanian	ro-RO	1
Romansh	rm-CH	1
Russian	ru-RU	1
Sakha	sah-RU	1
Sami, Northern (Norway)	se-NO	1
Sami, Northern (Sweden)	se-NO	1
Scottish Gaelic	gd-GB	1
Serbian - Cyrillic	sr-Cyrl-RS	1
Serbian - Latin	sr-Latn-RS	1
Sesotho sa Leboa	nso-ZA	1

NAME	LOCALE CODE	KEYBOARD LAYOUT VALUE
Setswana	tn-ZA	1
Sinhala	si-LK	1
Slovak	sk-SK	1
Slovenian	sl-SI	1
Sorbian, Upper	hsb-DE	1
Spanish (Mexico)	es-MX	1
Spanish (Spain)	es-ES	1
Swedish	sv-SE	1
Syriac	syr-SY	1
Tajik	tg-Cyrl-TJ	1
Tamazight (Central Atlas) - Tifinagh	tzm-Tfng-MA	1
Tamazight (Central Atlas) - Latin	tzm-Latn-DZ	1
Tamil - INSCRIPT	ta-IN	1
Tamil - Phonetic	ta-IN	2
Tatar	tt-RU	1
Telugu - INSCRIPT	te-IN	1
Telugu - Phonetic	te-IN	2
Thai	th-TH	1
Tibetan	bo-CN	1
Turkish	tr-TR	1
Turkmen	tk-TM	1
Ukrainian	uk-UA	1
Urdu	ur-PK	1
Uyghur	ug-CN	1
Uzbek - Cyrillic	uz-Cyrl-UZ	1

NAME	LOCALE CODE	KEYBOARD LAYOUT VALUE
Uzbek - Latin	uz-Latn-UZ	1
Valencian	ca-ES-valencia	1
Vietnamese - QWERTY	vi-VN	1
Vietnamese - TELEX	vi-VN	2
Vietnamese - VNI	vi-VN	3
Welsh	cy-GB	1
Wolof	N/A	1
Xhosa	xh-ZA	1
Yoruba	yo-NG	1
Zulu	zu-ZA	1

# Theme (reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Use Theme to configure accent and background colors on Windows 10 Mobile.

## Applies to

SETTING	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
All settings		X			

## DefaultAccentColor

In the dropdown menu for DefaultAccentColor, select from the list of colors. The accent color is used for the background of the start tiles, some text, the progress indicator, the user's My Phone web site, and so on.

## DefaultBackgroundColor

Select between **Light** and **Dark** for theme.

## Related topics

- [Themes and accent colors](#)

# UnifiedWriteFilter (reference)

9/6/2017 • 2 min to read • [Edit Online](#)

Use UnifiedWriteFilter to configure settings for the Unified Write Filter (UWF) in your device to help protect your physical storage media, including most standard writable storage types that are supported by the OS, such as physical hard disks, solid-state drives, internal USB devices, external SATA devices, and so on. You can also use UWF to make read-only media appear to the OS as a writeable volume.

## IMPORTANT

You cannot use UWF to protect external USB devices or flash drives.

UWF intercepts all write attempts to a protected volume and redirects those write attempts to a virtual overlay. This improves the reliability and stability of your device and reduces the wear on write-sensitive media, such as flash memory media like solid-state drives.

The overlay does not mirror the entire volume, but dynamically grows to keep track of redirected writes. Generally the overlay is stored in system memory, although you can cache a portion of the overlay on a physical volume.

## NOTE

UWF fully supports the NTFS system; however, during device startup, NTFS file system journal files can write to a protected volume before UWF has loaded and started protecting the volume.

[Learn more about the Unified Write Filter feature.](#)

## Applies to

SETTING	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
All settings	X				X

## FilterEnabled

Set to **True** to enable UWF.

## OverlaySize

Enter the maximum overlay size, in megabytes (MB), for the UWF overlay. The minimum value for maximum overlay size is 1024.

## NOTE

UnifiedWriteFilter must be enabled for this setting to work.

## OverlayType

OverlayType specifies where the overlay is stored. Select between **RAM** (default) and **Disk** (pre-allocated file on the

system volume).

## Registry Exclusions

You can add or remove registry entries that will be excluded from UWF filtering. When a registry key is in the exclusion list, all writes to that registry key bypass UWF filtering and are written directly to the registry and persist after the device restarts.

Use **Add** to add a registry entry to the exclusion list after you restart the device.

Use **Remove** to remove a registry entry from the exclusion list after you restart the device.

## Volumes

Enter a drive letter for a volume to be protected by UWF.

### NOTE

In the current OS release, Windows Configuration Designer contains a validation bug. To work around this issue, you must include a ":" after the drive letter when specifying the value for the setting. For example, if you are specifying the C drive, you must set DriveLetter to "C:" instead of just "C".

# UniversalApplInstall (reference)

10/17/2017 • 3 min to read • [Edit Online](#)

Use UniversalApplInstall settings to install Windows apps from the Microsoft Store or a hosted location.

## NOTE

You can only use the Windows provisioning settings and provisioning packages for apps where you have the available installation files, namely with sideloaded apps that have an offline license. [Learn more about offline app distribution](#).

## Applies to

SETTING	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
DeviceContextApp	X		X		
DeviceContextAppLicense	X		X		
StoreInstall	X	X	X	X	X
UserContextApp	X	X	X	X	X
UserContextAppLicense	X	X	X	X	X

## DeviceContextApp

Enter an app package family name to install an app for all users of the device. You can use the [Get-AppxPackage cmdlet](#) to get the package family name for an installed app.

## NOTE

For XAP files, enter the product ID.

For each app that you add to the package, configure the settings in the following table.

SETTING	VALUE	DESCRIPTION
ApplicationFile	.appx or .appxbundle	Set the value to the app file that you want to install on the device. In addition, you must also enable the <a href="#">AllowAllTrustedApps setting</a> and add a root certificate or license file.

Setting	Value	Description
DependencyAppxFiles	any required frameworks	In Microsoft Store for Business, any dependencies for the app are listed in the <b>Required frameworks</b> section of the download page.
DeploymentOptions	<ul style="list-style-type: none"> <li>- None</li> <li>-Force application shutdown: If this package, or any package that depends on this package, is currently in use, the processes associated with the package are shut down forcibly so that registration can continue</li> <li>- Development mode: do not use</li> <li>- Install all resources: When you set this option, the app is instructed to skip resource applicability checks.</li> <li>- Force target application shutdown: If this package is currently in use, the processes associated with the package are shut down forcibly so that registration can continue</li> </ul>	Select a deployment option.
LaunchAppAtLogin	<ul style="list-style-type: none"> <li>- Do not launch app</li> <li>- Launch app</li> </ul>	Set the value for app behavior when a user signs in.
OptionalPackageFiles	additional files required by the package	Browse to, select, and add the optional package files.

For more information on deployment options, see [DeploymentOptions Enum](#).

## DeviceContextAppLicense

Use to specify the license file for the provisioned app.

- Specify a **LicenseProductId** for the app. You can find the license ID in the root header of the license file. Here is an example, `LicenseID="aaaaaaaa-dddd-8848-f8d0-7d6a93dfcccc"`. Enter it in the LicenseProductId field, and click **Add**.
- Select the LicenseProductId in the Available Customizations pane, and then browse to and select the app license file.

## StoreInstall

Use to install an app from the Microsoft Store for Business.

- Enter a package family name, and then click **Add**.
- Configure the following required settings for the app package.

Setting	Description
Flags	Description not available at this time.
ProductID	Enter the product ID. <a href="#">Learn how to find the product ID</a> .
Skuid	Enter the SKU ID. <a href="#">Learn how to find the SKU ID</a> .

## UserContextApp

Use to add a new user context app.

1. Specify a **PackageFamilyName** for the app, and then click **Add**.
2. Select the PackageFamilyName in the Available Customizations pane, and then configure the following settings.

SETTING	VALUE	DESCRIPTION
ApplicationFile	app file	Browse to, select, and add the application file,
DependencyAppxFiles	additional files required by the app	Browse to, select, and add dependency files.
DeploymentOptions	<ul style="list-style-type: none"><li>- None</li><li>- Force application shutdown</li><li>- Development mode</li><li>- Install all resources</li><li>- Force target application shutdown</li></ul>	Select a deployment option.
LaunchAppAtLogin	<ul style="list-style-type: none"><li>- Do not launch app</li><li>- Launch app</li></ul>	Select whether the app should be started when a user signs in.

## UserContextAppLicense

Use to specify the license file for the user context app.

1. Specify a **LicenseProductId** for the app. You can find the license ID in the root header of the license file. Here is an example, `LicenseID="aaaaaaaa-dddd-8848-f8d0-7d6a93dfcccc"`. Enter it in the LicenseProductId field, and click **Add**.
2. Select the LicenseProductId in the Available Customizations pane, and then browse to and select the app license file.

# UniversalAppUninstall (reference)

10/17/2017 • 1 min to read • [Edit Online](#)

Use UniversalAppUninstall settings to uninstall or remove Windows apps.

## Applies to

Setting	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
RemoveProvisionedApp	X				
Uninstall	X	X	X	X	X

## RemoveProvisionedApp

Universal apps can be *provisioned*, which means that they are available on the device for installation in user context. When a user runs the provisioned app, the app is then installed for that user.

Use **RemoveProvisionedApp** to remove app packages that are available on the device. Any instances of the app that have already been installed by a user are not uninstalled. To uninstall provisioned apps that have been installed by a user, use the **Uninstall** setting.

1. Enter the PackageFamilyName for the app package, and then click **Add**.
2. Select the PackageFamilyName in the Available Customizations pane, and then select **RemoveProvisionedApp**.

## Uninstall

Use **Uninstall** to remove provisioned apps that have been installed by a user.

1. Enter the PackageFamilyName for the app package, and then click **Add**.
2. Select the PackageFamilyName in the Available Customizations pane, and then select **Uninstall**.

# UsbErrorsOEMOverride (reference)

9/14/2017 • 1 min to read • [Edit Online](#)

Allows an OEM to hide the USB option UI in Settings and all USB device errors.

## Applies to

Setting	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
HideUsbErrorNotifyOptionUI	X	X	X	X	

## HideUsbErrorNotifyOptionUI

Configure to **Show** or **Hide** the USB error notification.

# WeakCharger (reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Use WeakCharger settings to configure the charger notification UI.

## Applies to

Setting	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
HideWeakChargerNotifyOptionUI	X	X	X	X	
NotifyOnWeakCharger	X	X	X	X	

## HideWeakChargerNotifyOptionUI

This setting determines whether the user sees the dialog that's displayed when the user connects the device to an incompatible charging source. By default, the OS shows the weak charger notification option UI.

Select between **Show Weak Charger Notifications UI** and **Hide Weak Charger Notifications UI**.

## NotifyOnWeakCharger

This setting displays a warning when the user connects the device to an incompatible charging source. This warning is intended to notify users that their device may take longer to charge or may not charge at all with the current charging source.

An incompatible charging source is one that does not behave like one of the following port types as defined by the USB Battery Charging Specification, Revision 1.2, available on the USB.org website:

- Charging downstream port
- Standard downstream port
- Dedicated charging port

Select between **Disable Weak Charger Notifications UI** and **Enable Weak Charger Notifications UI**.

# WindowsTeamSettings (reference)

9/6/2017 • 4 min to read • [Edit Online](#)

Use WindowsTeamSettings settings to configure Surface Hub.

## Applies to

Setting	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	Iot Core
All settings			X		

## Connect

Setting	Value	Description
AutoLaunch	True or false	Open the Connect app automatically when someone projects.
Channel	- 1, 3, 4, 5, 6, 7, 8, 9, 10, 11 (works with all Miracast senders in all regions) - 36, 40, 44, 48 (works with all 5ghz band Miracast senders in all regions) - 149, 153, 157, 161, 165 (works with all 5ghz band Miracast senders in all regions except Japan)	Wireless channel to use for Miracast operation. The supported channels are defined by the Wi-Fi Alliance Wi-Fi Direct specification. Integer specifying the channel. The default value is 255. Outside of regulatory concerns, if the channel is configured incorrectly the driver will either not boot, or will broadcast on the wrong channel (which senders won't be looking for).
Enabled	True or false	Enables wireless projection to the device.
PINRequired	True or false	Requires presenters to enter a PIN to connect wirelessly to the device.

## DeviceAccount

A device account is a Microsoft Exchange account that is connected with Skype for Business, which allows people to join scheduled meetings, make Skype for Business calls, and share content from the device.

Setting	Value	Description
CalendarSyncEnabled	True or false	Specifies whether calendar sync and other Exchange Server services are enabled.
DomainName	Domain of the device account when you are using Active Directory	To use a device account from Active Directory, you should specify both <b>DomainName</b> and <b>UserName</b> for the device account.

Setting	Value	Description
Email	Email address	Email address of the device account.
ExchangeServer	Exchange Server	Normally, the device will try to automatically discover the Exchange server. This field is only required if automatic discovery fails.
Password	Password	Password for the device account.
PasswordRotationEnabled	0 = enabled 1 = disabled	Specifies whether automatic password rotation is enabled. If you enforce a password expiration policy on the device account, use this setting to allow the device to manage its own password by changing it frequently, without requiring you to manually update the account information when the password expires. You can reset the password at any time using Active Directory or Azure AD.
SipAddress	Session Initiation Protocol (SIP) address	Normally, the device will try to automatically discover the SIP. This field is only required if automatic discovery fails.
UserName	User name	Username of the device account when you are using Active Directory.
UserPrincipalName	User principal name (UPN)	To use a device account from Azure Active Directory or a hybrid deployment, you should specify the UPN of the device account.
ValidateAndCommit	Any text	Validates the data provided and then commits the changes. This process occurs automatically after the other DeviceAccount settings are applied. The text you enter for the ValidateAndCommit setting doesn't matter.

## FriendlyName

Enter the name that users will see when they want to project wirelessly to the device.

## MaintenanceHours

Maintenance hours are the period of time during which automatic maintenance tasks are performed.

Setting	Value	Description
Duration	Duration in minutes. For example, to set a 3-hour duration, set this value to 180.	The amount of time the device will be in maintenance, when the device will continue to download or install updates.

Setting	Value	Description
StartTime	Start time in minutes from midnight. For example, to set a 2:00 am start time, set this value to 120	Start time for when device is allowed to start downloading and installing updates.

## OMSAgent

Configures the Operations Management Suite workspace.

Setting	Value	Description
WorkspaceID	GUID	GUID identifying the Operations Management Suite workspace ID to collect the data. Set this to an empty string to disable the MOM agent.
WorkspaceKey	Key	Primary key for authenticating with the workspace.

## Properties

Setting	Value	Description
AllowAutoProxyAuth	True or false	Specifies if the Surface Hub can use the device account to authenticate into proxy servers requiring authentication.
AllowSessionResume	True or false	Specifies if users are allowed to resume their session after session timeout.
DefaultVolume	Numeric value between 0 and 100	Default speaker volume. Speaker volume will be set to this value at every session startup.
DisableSigninSuggestions	True or false	Specifies if the Surface Hub will not show suggestions when users try to sign in to see their meetings and files.
DoNotShowMyMeetingsAndFiles	True or false	Specifies if users can sign in and have full access to personal meetings and most recently used documents.
ScreenTimeout	Select minutes from dropdown menu	The time (in minutes) of inactivity after which the Surface Hub will turn off its screen.
SessionTimeout	Select minutes from dropdown menu	The time (in minutes) of inactivity after which the Surface Hub will time out the current session and return to the welcome screen.
SleepTimeout	Select minutes from dropdown menu	The time (in minutes) of inactivity after which the Surface Hub will go into a sleep state.

## SkypeForBusiness

SETTING	VALUE	DESCRIPTION
DomainName	Domain name	Specifies the domain name of the target server when the Skype for Business server is in a domain that's different from the device account.

## Welcome

SETTING	VALUE	DESCRIPTION
AutoWakeScreen	True or false	Specifies whether to automatically turn on the screen using motion sensors.
CurrentBackgroundPath	Https URL to a PNG file	Background image for the welcome screen.
MeetingInfoOption	0 = organizer and time only 1 = organizer, time, and subject (subject is hidden for private meetings)	Specifies whether meeting information is displayed on the welcome screen.

## Related topics

- [SurfaceHub configuration service provider \(CSP\)](#)

# WLAN (reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Do not use at this time. Instead, use [ConnectivityProfiles > WLAN](#)

## Applies to

SETTING	DESKTOP EDITIONS	MOBILE EDITIONS	SURFACE HUB	HOLOLENS	IOT CORE
All settings				X	

# Workplace (reference)

9/6/2017 • 1 min to read • [Edit Online](#)

Use Workplace settings to configure bulk user enrollment to a mobile device management (MDM) service. For more information, see [Bulk enrollment step-by-step](#).

## Applies to

Setting	Desktop Editions	Mobile Editions	Surface Hub	HoloLens	IOT Core
Enrollments	X	X	X	X	X

## Enrollments

Select **Enrollments**, enter a UPN, and then click **Add** to configure the settings for the enrollment. The UPN is a unique identifier for enrollment. For bulk enrollment, this must a service account that is allowed to enroll multiple users. Example, "generic-device@contoso.com"

Settings	Value	Description
AuthPolicy	- OnPremise - Certificate	The authentication policy used by the MDM service
DiscoveryServiceFullUrl	URL	The full URL for the discovery service
EnrollmentServiceFullUrl	URL	The full URL for the enrollment service
PolicyServiceFullUrl	URL	The full URL for the policy service
Secret	- Password string for on-premise authentication enrollment - Federated security token for federated enrollment - Certificate thumb print for certificate-based enrollment	Enter the appropriate value for the selected AuthPolicy

## Related topics

- [Provisioning configuration service provider \(CSP\)](#)

# Lockdown features from Windows Embedded 8.1 Industry

7/28/2017 • 4 min to read • [Edit Online](#)

## Applies to

- Windows 10

Many of the lockdown features available in Windows Embedded 8.1 Industry have been modified in some form for Windows 10. This table maps Windows Embedded Industry 8.1 features to Windows 10 Enterprise features, along with links to documentation.

WINDOWS EMBEDDED 8.1 INDUSTRY LOCKDOWN FEATURE	WINDOWS 10 FEATURE	CHANGES
<a href="#">Hibernate Once/Resume Many (HORM):</a> Quick boot to device	N/A	HORM is supported in Windows 10, version 1607 and later.
<a href="#">Unified Write Filter:</a> protect a device's physical storage media	<a href="#">Unified Write Filter</a>	The Unified Write Filter is continued in Windows 10.
<a href="#">Keyboard Filter:</a> block hotkeys and other key combinations	<a href="#">Keyboard Filter</a>	Keyboard filter is added in Windows 10, version 1511. As in Windows Embedded Industry 8.1, Keyboard Filter is an optional component that can be turned on via <a href="#">Turn Windows Features On/Off</a> . Keyboard Filter (in addition to the WMI configuration previously available) will be configurable through Windows Imaging and Configuration Designer (ICD) in the SMISettings path.
<a href="#">Shell Launcher:</a> launch a Classic Windows application on sign-on	<a href="#">Shell Launcher</a>	Shell Launcher continues in Windows 10. It is now configurable in Windows ICD under the <b>SMISettings</b> category.  Learn <a href="#">how to use Shell Launcher to create a kiosk device</a> that runs a Classic Windows application.
<a href="#">Application Launcher:</a> launch a Universal Windows Platform (UWP) app on sign-on	<a href="#">Assigned Access</a>	The Windows 8 Application Launcher has been consolidated into Assigned Access. Application Launcher enabled launching a Windows 8 app and holding focus on that app. Assigned Access offers a more robust solution for ensuring that apps retain focus.

WINDOWS EMBEDDED 8.1 INDUSTRY LOCKDOWN FEATURE	WINDOWS 10 FEATURE	CHANGES
<a href="#">Dialog Filter</a> : suppress system dialogs and control which processes can run	<a href="#">AppLocker</a>	<p>Dialog Filter has been deprecated for Windows 10. Dialog Filter provided two capabilities; the ability to control which processes were able to run, and the ability to prevent dialogs (in practice, system dialogs) from appearing.</p> <ul style="list-style-type: none"> <li>Control over which processes are able to run will now be provided by AppLocker.</li> <li>System dialogs in Windows 10 have been replaced with system toasts. To see more on blocking system toasts, see <a href="#">Toast Notification Filter</a> below.</li> </ul>
<a href="#">Toast Notification Filter</a> : suppress toast notifications	Mobile device management (MDM) and Group Policy	<p>Toast Notification Filter has been replaced by MDM and Group Policy settings for blocking the individual components of non-critical system toasts that may appear. For example, to prevent a toast from appearing when a USB drive is connected, ensure that USB connections have been blocked using the USB-related policies, and turn off notifications from apps.</p> <p>Group Policy: <b>User Configuration &gt; Administrative Templates &gt; Start Menu and Taskbar &gt; Notifications</b></p> <p>MDM policy name may vary depending on your MDM service. In Microsoft Intune, use <b>Allow action center notifications</b> and a <a href="#">custom OMA-URI setting</a> for <b>AboveLock/AllowActionCenterNotifications</b>.</p>
<a href="#">Embedded Lockdown Manager</a> : configure lockdown features	<a href="#">Windows Imaging and Configuration Designer (ICD)</a>	The Embedded Lockdown Manager has been deprecated for Windows 10 and replaced by the Windows ICD. Windows ICD is the consolidated tool for Windows imaging and provisioning scenarios and enables configuration of all Windows settings, including the lockdown features previously configurable through Embedded Lockdown Manager.

WINDOWS EMBEDDED 8.1 INDUSTRY LOCKDOWN FEATURE	WINDOWS 10 FEATURE	CHANGES
<a href="#">USB Filter</a> : restrict USB devices and peripherals on system	MDM and Group Policy	<p>The USB Filter driver has been replaced by MDM and Group Policy settings for blocking the connection of USB devices.</p> <p>Group Policy: <b>Computer Configuration &gt; Administrative Templates &gt; System &gt; Device Installation &gt; Device Installation Restrictions</b></p> <p>MDM policy name may vary depending on your MDM service. In Microsoft Intune, use <b>Allow removable storage</b> or <b>Allow USB connection (Windows 10 Mobile only)</b>.</p>
<a href="#">Assigned Access</a> : launch a UWP app on sign-in and lock access to system	<a href="#">Assigned Access</a>	<p>Assigned Access has undergone significant improvement for Windows 10. In Windows 8.1, Assigned Access blocked system hotkeys and edge gestures, and non-critical system notifications, but it also applied some of these limitations to other accounts on the device.</p> <p>In Windows 10, Assigned Access no longer affects accounts other than the one being locked down. Assigned Access now restricts access to other apps or system components by locking the device when the selected user account logs in and launching the designated app above the lock screen, ensuring that no unintended functionality can be accessed.</p> <p>Learn <a href="#">how to use Assigned Access to create a kiosk device</a> that runs a Universal Windows app.</p>
<a href="#">Gesture Filter</a> : block swipes from top, left, and right edges of screen	MDM and Group Policy	<p>In Windows 8.1, gestures provided the ability to close an app, to switch apps, and to reach the Charms. In Windows 10, Charms have been removed. In Windows 10, version 1607, you can block swipes using the <a href="#">Allow edge swipe</a> policy.</p>
<a href="#">Custom Logon</a> : suppress Windows UI elements during Windows sign-on, sign-off, and shutdown	<a href="#">Embedded Logon</a>	No changes. Applies only to Windows 10 Enterprise and Windows 10 Education.

WINDOWS EMBEDDED 8.1 INDUSTRY LOCKDOWN FEATURE	WINDOWS 10 FEATURE	CHANGES
<a href="#">Unbranded Boot</a> : custom brand a device by removing or replacing Windows boot UI elements	<a href="#">Unbranded Boot</a>	No changes. Applies only to Windows 10 Enterprise and Windows 10 Education.

# User Experience Virtualization (UE-V) for Windows 10 overview

6/20/2017 • 4 min to read • [Edit Online](#)

## Applies to

- Windows 10, version 1607

Many users customize their settings for Windows and for specific applications. Customizable Windows settings include Microsoft Store appearance, language, background picture, font size, and accent colors. Customizable application settings include language, appearance, behavior, and user interface options.

With User Experience Virtualization (UE-V), you can capture user-customized Windows and application settings and store them on a centrally managed network file share. When users log on, their personalized settings are applied to their work session, regardless of which device or virtual desktop infrastructure (VDI) sessions they log on to.

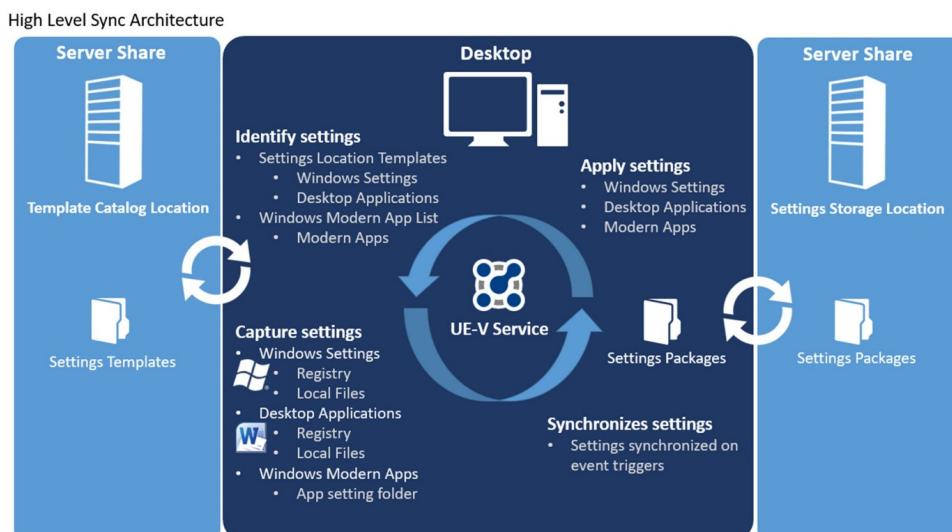
## With UE-V you can...

- Specify which application and Windows settings synchronize across user devices
- Deliver the settings anytime and anywhere users work throughout the enterprise
- Create custom templates for your third-party or line-of-business applications
- Recover settings after hardware replacement or upgrade, or after re-imaging a virtual machine to its initial state

With the release of Windows 10, version 1607, UE-V is included with the Windows 10 for Enterprise edition. If you are new to Windows 10 and UE-V or upgrading from a previous version of UE-V, you'll need to download, activate, and install server- and client-side components to start synchronizing user-customized settings across devices.

## Components of UE-V

The diagram below illustrates how UE-V components work together to synchronize user settings.



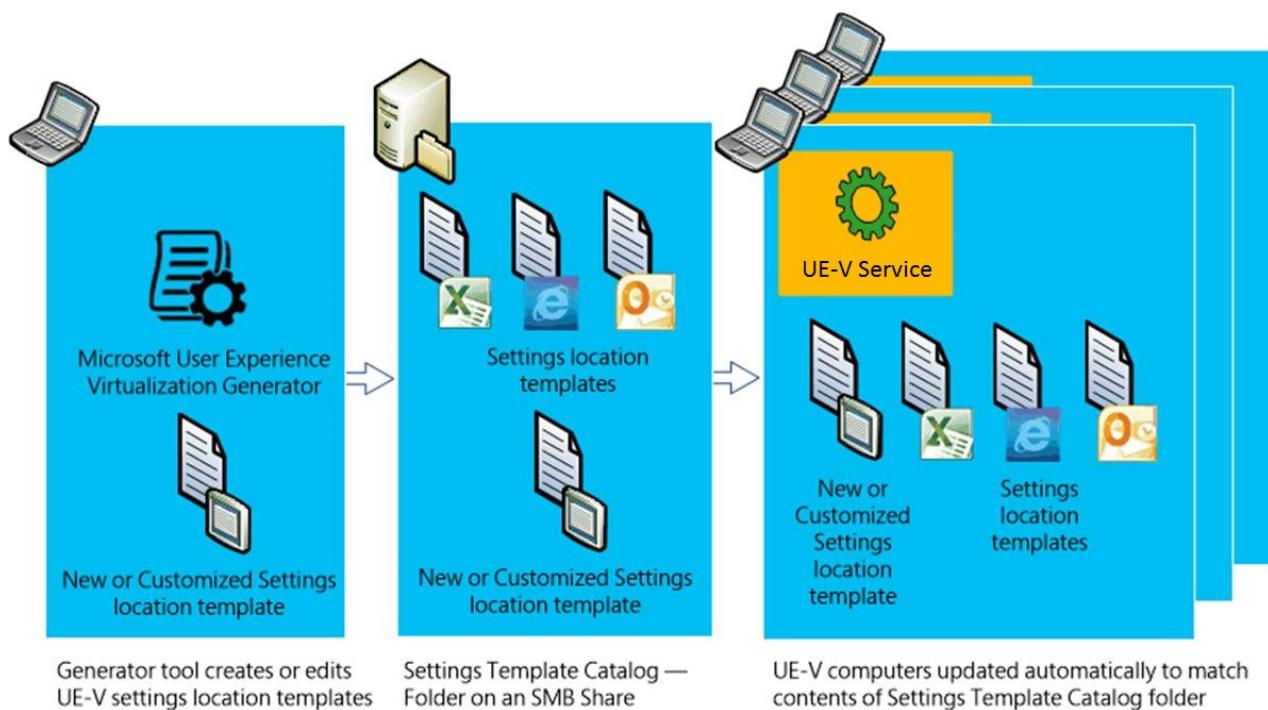
COMPONENT	FUNCTION
<b>UE-V service</b>	Enabled on every device that needs to synchronize settings, the <b>UE-V service</b> monitors registered applications and Windows for any settings changes, then synchronizes those settings between devices.
<b>Settings packages</b>	<p>Application settings and Windows settings are stored in <b>settings packages</b> created by the UE-V service. Settings packages are built, locally stored, and copied to the settings storage location.</p> <p>The setting values for <b>desktop applications</b> are stored when the user closes the application.</p> <p>Values for <b>Windows settings</b> are stored when the user logs off, when the computer is locked, or when the user disconnects remotely from a computer.</p> <p>The sync provider determines when the application or operating system settings are read from the <b>Settings Packages</b> and synchronized.</p>
<b>Settings storage location</b>	This is a standard network share that your users can access. The UE-V service verifies the location and creates a hidden system folder in which to store and retrieve user settings.
<b>Settings location templates</b>	<p>UE-V uses XML files as settings location templates to monitor and synchronize desktop application settings and Windows desktop settings between user computers. By default, some settings location templates are included in UE-V. You can also create, edit, or validate custom settings location templates by <a href="#">managing settings synchronization for custom applications</a>.</p> <p><b>Note</b> Settings location templates are not required for Windows applications.</p>
<b>Universal Windows applications list</b>	<p>Settings for Windows applications are captured and applied dynamically. The app developer specifies the settings that are synchronized for each app. UE-V determines which Windows applications are enabled for settings synchronization using a managed list of applications. By default, this list includes most Windows applications.</p> <p>You can add or remove applications in the Windows app list by following the procedures in <a href="#">Managing UE-V Settings Location Templates Using Windows PowerShell and WMI</a>.</p>

## Manage settings synchronization for custom applications

Use these UE-V components to create and manage custom templates for your third-party or line-of-business applications.

COMPONENT	DESCRIPTION

COMPONENT	DESCRIPTION
<b>UE-V template generator</b>	<p>Use the <b>UE-V template generator</b> to create custom settings location templates that you can then distribute to user computers. The UE-V template generator also lets you edit an existing template or validate a template that was created with a different XML editor.</p> <p>With the Windows 10, version 1607 release, the UE-V template generator is installed with the <a href="#">Windows Assessment and Deployment kit for Windows 10, version 1607</a> (Windows ADK).</p> <p>If you are upgrading from an existing UE-V installation, you'll need to use the new generator to create new settings location templates. Application templates created with previous versions of the UE-V template generator are still supported, however.</p>
<b>Settings template catalog</b>	<p>The <b>settings template catalog</b> is a folder path on UE-V computers or a Server Message Block (SMB) network share that stores the custom settings location templates. The UE-V service checks this location once a day, retrieves new or updated templates, and updates its synchronization behavior. If you use only the UE-V default settings location templates, then a settings template catalog is unnecessary. For more information about settings deployment catalogs, see <a href="#">Deploy a UE-V settings template catalog</a>.</p>



## Settings synchronized by default

UE-V synchronizes settings for these applications by default. For a complete list and more detailed information, see [Settings that are automatically synchronized in a UE-V deployment](#).

- Microsoft Office 2016, 2013, and 2010
- Internet Explorer 11 and 10
- Many Windows applications, such as Xbox
- Many Windows desktop applications, such as Notepad

- Many Windows settings, such as desktop background or wallpaper

**Note** You can also [customize UE-V to synchronize settings](#) for applications other than those synchronized by default.

## Other resources for this feature

- [Get Started with UE-V for Windows 10](#)
- [UE-V for Windows 10 Release Notes](#)
- [Prepare to deploy UE-V for Windows 10](#)
- [Upgrade to UE-V for Windows 10](#)
- [Administer UE-V for Windows 10](#)
- [Technical Reference for UE-V for Windows 10](#)

## Have a suggestion for UE-V?

Add or vote on suggestions on the [User Experience Virtualization feedback site](#).

For UE-V issues, use the [UE-V TechNet Forum](#).

# Get Started with UE-V

6/20/2017 • 5 min to read • [Edit Online](#)

## Applies to

- Windows 10, version 1607

Follow the steps in this topic to deploy User Experience Virtualization (UE-V) for the first time in a test environment. Evaluate UE-V to determine whether it's the right solution to manage user settings across multiple devices within your enterprise.

**Note** The information in this section is explained in greater detail throughout the rest of the documentation. If you've already determined that UE-V is the right solution and you don't need to further evaluate it, see [Prepare a UE-V deployment](#).

The standard installation of UE-V synchronizes the default Microsoft Windows and Office settings and many Windows applications settings. For best results, ensure that your test environment includes two or more user computers that share network access.

- [Step 1: Confirm prerequisites](#). Review the supported configurations in this section to verify that your environment is able to run UE-V.
- [Step 2: Deploy the settings storage location](#). Explains how to deploy a settings storage location. All UE-V deployments require a location to store settings packages that contain the synchronized setting values.
- [Step 3: Enable the UE-V service](#). Explains how to enable the UE-V service on user devices. To synchronize settings using UE-V, devices must have the UE-V service enabled and running.
- [Step 4: Test Your UE-V evaluation deployment](#). Run a few tests on two computers with the UE-V service enabled to see how UE-V works and if it meets your organization's needs.
- Step 5: Deploy UE-V for custom applications (optional). If you want to evaluate how your third-party and line-of-business applications work with UE-V, follow the steps in [Use UE-V with custom applications](#). Following this link takes you to another topic. Use your browser's **Back** button to return to this topic.

## Step 1: Confirm prerequisites

Before you proceed, ensure that your environment meets the following requirements for running UE-V.

OPERATING SYSTEM	EDITION	SERVICE PACK	SYSTEM ARCHITECTURE	WINDOWS POWERSHELL	MICROSOFT .NET FRAMEWORK
Windows 10, version 1607	Windows 10 Enterprise	NA	32-bit or 64-bit	Windows PowerShell 3.0 or higher	.NET Framework 4 or higher
Windows 8 and Windows 8.1	Enterprise or Pro	None	32-bit or 64-bit	Windows PowerShell 3.0 or higher	.NET Framework 4.5

OPERATING SYSTEM	EDITION	SERVICE PACK	SYSTEM ARCHITECTURE	WINDOWS POWERSHELL	MICROSOFT .NET FRAMEWORK
Windows Server 2012 or Windows Server 2012 R2	Standard or Datacenter	None	64-bit	Windows PowerShell 3.0 or higher	.NET Framework 4.5

## Step 2: Deploy the settings storage location

You'll need to deploy a settings storage location, a standard network share where user settings are stored in a settings package file. When you create the settings storage share, you should limit access to users that require it. For more information, see [Deploy a UE-V Settings Storage Location](#).

### Create a network share

1. Create a new security group and add UE-V users to it.
2. Create a new folder on the centrally located computer that stores the UE-V settings packages, and then grant the UE-V users access with group permissions to the folder. The administrator who supports UE-V must have permissions to this shared folder.
3. Assign UE-V users permission to create a directory when they connect. Grant full permission to all subdirectories of that directory, but block access to anything above.
4. Set the following share-level Server Message Block (SMB) permissions for the settings storage location folder.

USER ACCOUNT	RECOMMENDED PERMISSIONS
Everyone	No permissions
Security group of UE-V users	Full control

5. Set the following NTFS file system permissions for the settings storage location folder.

USER ACCOUNT	RECOMMENDED PERMISSIONS	FOLDER
Creator/owner	Full control	Subfolders and files only
Security group of UE-V users	List folder/read data, create folders/append data	This folder only

**Security Note** If you create the settings storage share on a computer running a Windows Server operating system, configure UE-V to verify that either the local Administrators group or the current user is the owner of the folder where settings packages are stored. To enable this additional security, specify this setting in the Windows Server Registry Editor:

1. Add a **REG\_DWORD** registry key named "**RepositoryOwnerCheckEnabled**" to **HKEY\_LOCAL\_MACHINE\Software\Microsoft\UEV\Agent\Configuration**.
2. Set the registry key value to 1.

## Step 3: Enable the UE-V service on user devices

For evaluation purposes, enable the service on at least two devices that belong to the same user in your test

environment.

The UE-V service is the client-side component that captures user-personalized application and Windows settings and saves them in settings packages. Settings packages are built, locally stored, and copied to the settings storage location.

Before enabling the UE-V service, you'll need to register the UE-V templates for first use. In a PowerShell window, type `register-TemplateName` where **TemplateName** is the name of the UE-V template you want to register, and press ENTER.

With Windows 10, version 1607 and later, the UE-V service is installed on user devices when the operating system is installed. Enable the service to start using UE-V. You can enable the service with the Group Policy editor or with Windows PowerShell.

### To enable the UE-V service with Group Policy

1. Open the device's **Group Policy Editor**.
2. Navigate to **Computer Configuration > Administrative Templates > Windows Components > Microsoft User Experience Virtualization**.
3. Run **Enable UEV**.
4. Restart the device.

### To enable the UE-V service with Windows PowerShell

1. In a PowerShell window, type **Enable-UEV** and press ENTER.
2. Restart the device.
3. In a PowerShell window, type **Get-UEVStatus** and press ENTER to verify that the UE-V service was successfully enabled.

## Step 4: Test your UE-V evaluation deployment

You're ready to run a few tests on your UE-V evaluation deployment to see how UE-V works.

1. On the first device (Computer A), make one or more of these changes:
  - Open Windows Desktop and move the taskbar to a different location in the window.
  - Change the default fonts.
  - Open Notepad and set format -> word wrap **on**.
  - Change the behavior of any Windows application, as detailed in [Managing UE-V settings location templates using Windows PowerShell and WMI](#).
  - Disable Microsoft Account settings synchronization and roaming profiles.
2. Log off Computer A. Settings are saved in a UE-V settings package when users lock, logoff, exit an application, or when the sync provider runs (every 30 minutes by default).
3. Log in to the second device (Computer B) as the same user as Computer A.
4. Open Windows Desktop and verify that the taskbar location matches that of Computer A. Verify that the default fonts match and that NotePad is set to **word wrap on**. Also verify the change you made to any Windows applications.
5. You can change the settings in Computer B back to the original Computer A settings. Then log off Computer B and log in to Computer A to verify the changes.

## Have a suggestion for UE-V?

Add or vote on suggestions on the [User Experience Virtualization feedback site](#).

For UE-V issues, use the [UE-V TechNet Forum](#).

## Other resources for this feature

- [User Experience Virtualization overview](#)
- [Prepare a UE-V Deployment](#)
- [Upgrade to UE-V for Windows 10](#)
- [Administering UE-V](#)
- [Troubleshooting UE-V](#)
- [Technical Reference for UE-V](#)

# What's New in UE-V

6/20/2017 • 5 min to read • [Edit Online](#)

## Applies to

- Windows 10, version 1607

User Experience Virtualization (UE-V) for Windows 10, version 1607, includes these new features and capabilities compared to UE-V 2.1. See [UE-V Release notes](#) for more information about the UE-V for Windows 10, version 1607 release.

## UE-V is now a feature in Windows 10

With Windows 10, version 1607 and later releases, UE-V is included with [Windows 10 for Enterprise](#) and is no longer part of the Microsoft Desktop Optimization Pack.

The changes in UE-V for Windows 10, version 1607 impact already existing implementations of UE-V in the following ways:

- The UE-V Agent is replaced by the UE-V service. The UE-V service is installed with Windows 10, version 1607 and no longer has to be deployed separately. Performing an in-place upgrade to Windows 10, version 1607, on user devices automatically installs the UE-V service, migrates users' UE-V configurations, and updates the settings storage path.
- The UE-V template generator is available from the Windows 10 ADK. In previous releases of UE-V, the template generator was included in the Microsoft Desktop Optimization Pack. Although you'll need to use the new template generator to create new settings location templates, existing settings location templates will continue to work.
- The Company Settings Center was removed and is no longer available on user devices. Users can no longer manage their synchronized settings.
- The inbox templates such as Office 2016 and IE 10 are included as a part of Windows 10 and need to be manually registered with Powershell or Group policy before use.

For more information about how to configure an existing UE-V installation after upgrading user devices to Windows 10, see [Upgrade to UE-V for Windows 10](#).

**Important** You can upgrade your existing UE-V installation to Windows 10 from UE-V versions 2.1 or 2.0 only. If you are using a previous version of UE-V, you'll need to upgrade from that version to UE-V 2.x before you upgrade to Windows 10.

## New UE-V template generator is available from the Windows 10 ADK

UE-V for Windows 10 includes a new template generator, available from a new location. If you are upgrading from an existing UE-V installation, you'll need to use the new generator to create settings location templates. The UE-V for Windows 10 template generator is now available in the [Windows 10 Assessment and Deployment Kit](#) (Windows ADK).

## Company Settings Center removed in UE-V for Windows 10, version 1607

In previous versions of UE-V, users could select which of their customized application settings to synchronize with the Company Settings Center, a user interface that was available on user devices. Additionally, administrators could configure the Company Settings Center to include a link to support resources so that users could easily get support on virtualized settings-related issues.

With the release of Windows 10, version 1607, the Company Settings Center was removed and users can no longer manage their synchronized settings.

Administrators can still define which user-customized application settings can synchronize (roam) with Group Policy or Windows PowerShell.

**Note** With the removal of the Company Settings Center, the following group policies are no longer applicable:

- Contact IT Link Text
- Contact IT URL
- Tray Icon

## Compatibility with Microsoft Enterprise State Roaming

With Windows 10, version 1607, users can synchronize Windows application settings and Windows operating system settings to Azure instead of to OneDrive. You can use the Windows 10 enterprise sync functionality together with UE-V on on-premises domain-joined devices only.

In hybrid cloud environments, UE-V can roam Win32 applications on-premises while [Enterprise State Roaming](#) (ESR) can roam the rest, e.g., Windows and desktop settings, themes, colors, etc., to an Azure cloud installation.

To configure UE-V to roam Windows desktop and application data only, change the following group policies:

- Disable "Roam Windows settings" group policy
- Enable "Do not synchronize Windows Apps" group policy

For more information about using UE-V with Enterprise State Roaming, see [Settings and data roaming FAQ](#).

Additionally, to enable Windows 10 and UE-V to work together, configure these policy settings in the Microsoft User Experience Virtualization node:

- Enable "Do Not Synchronize Windows Apps"
- Disable "Sync Windows Settings"

## Settings Synchronization Behavior Changed in UE-V for Windows 10

While earlier versions of UE-V roamed taskbar settings between Windows 10 devices, UE-V for Windows 10, version 1607 does not synchronize taskbar settings between devices running Windows 10 and devices running previous versions of Windows.

In addition, UE-V for Windows has removed support for the Windows calculator application.

The Windows modern apps settings (DontSyncWindows8AppSettings) group policy is enabled by default and therefore, modern apps will not roam unless this policy is changed to disabled.

Please note, UE-V will roam any AppX apps that use the WinRT settings roaming API, provided that they have been opted in to roam at the time of development by the developer so there is no definitive list.

## Support Added for Roaming Network Printers

Users can now print to their saved network printers from any network device, including their default network printer.

Printer roaming in UE-V requires one of these scenarios:

- The print server can download the required driver when it roams to a new device.
- The driver for the roaming network printer is pre-installed on any device that needs to access that network printer.
- The printer driver can be imported from Windows Update.

**Note** The UE-V printer roaming feature does not roam printer settings or preferences, such as printing double-sided.

## Office 2016 Settings Location Template

UE-V for Windows 10, version 1607 includes the Microsoft Office 2016 settings location template with improved Outlook signature support. We've added synchronization of default signature settings for new, reply, and forwarded emails. Users no longer have to choose the default signature settings.

**Note** An Outlook profile must be created on any device on which a user wants to synchronize their Outlook signature. If the profile is not already created, the user can create one and then restart Outlook on that device to enable signature synchronization.

UE-V works with Office 365 to determine whether Office 2016 settings are roamed by Office 365. If settings are roamed by Office 365, they are not roamed by UE-V. See [Overview of user and roaming settings for Microsoft Office](#) for more information.

To enable settings synchronization using UE-V, do one of the following:

- Use Group Policy to disable Office 365 synchronization
- Do not enable the Office 365 synchronization experience during Office 2013 installation

UE-V includes Office 2016, Office 2013, and Office 2010 templates. Office 2007 templates are no longer supported. Users can still use Office 2007 templates from UE-V 2.0 or earlier and can still get templates from the [User Experience Virtualization Template Gallery](#).

## Have a suggestion for UE-V?

Add or vote on suggestions on the [User Experience Virtualization feedback site](#).

For UE-V issues, use the [UE-V TechNet Forum](#).

## Related topics

- [Microsoft User Experience Virtualization](#)
- [Get Started with UE-V](#)
- [Prepare a UE-V Deployment](#)
- [User Experience Virtualization \(UE-V\) Release Notes](#) for Windows 10, version 1607
- [Upgrade to UE-V for Windows 10](#)

# User Experience Virtualization (UE-V) Release Notes

6/20/2017 • 5 min to read • [Edit Online](#)

## Applies to

- Windows 10, version 1607

This topic includes information required to successfully install and use UE-V that is not included in the User Experience Virtualization (UE-V) documentation. If there are differences between the information in this topic and other UE-V topics, the latest change should be considered authoritative.

### **Company Settings Center removed in UE-V for Windows 10, version 1607**

In previous versions of UE-V, users could select which of their customized application settings to synchronize with the Company Settings Center, a user interface that was available on user devices. Additionally, administrators could configure the Company Settings Center to include a link to support resources so that users could easily get support on virtualized settings-related issues.

With the release of Windows 10, version 1607, the Company Settings Center was removed and users can no longer manage their synchronized settings.

Administrators can still define which user-customized application settings can synchronize (roam) with Group Policy or Windows PowerShell.

**Note** With the removal of the Company Settings Center, the following group policies are no longer applicable:

- Contact IT Link Text
- Contact IT URL
- Tray Icon

### **Upgrading from UE-V 1.0 to the in-box version of UE-V is blocked**

Version 1.0 of UE-V used Offline Files (Client Side Caching) for settings synchronization and pinned the UE-V sync folder to be available when the network was offline, however, this technology was removed in UE-V 2.x. As a result, UE-V 1.0 users are blocked from upgrading to UE-V for Windows 10, version 1607.

WORKAROUND: Remove the UE-V 1.0 sync folder from the Offline Files configuration and then upgrade to the in-box version of UE-V for Windows, version 1607 release.

### **UE-V settings location templates for Skype cause Skype to crash**

When a user generates a valid settings location template for the Skype desktop application, registers it, and then launches the Skype desktop application, Skype crashes. An ACCESS\_VIOLATION is recorded in the Application Event Log.

WORKAROUND: Remove or unregister the Skype template to allow Skype to work again.

### **Registry settings do not synchronize between App-V and native applications on the same device**

When a device has an application that is installed through both Application Virtualization (App-V) and locally with a Windows Installer (.msi) file, the registry-based settings do not synchronize between the technologies.

WORKAROUND: To resolve this problem, run the application by selecting one of the two technologies, but not both.

### **Unpredictable results when both Office 2010 and Office 2013 are installed on the same device**

When a user has both Office 2010 and Office 2013 installed, any common settings between the two versions of

Office are roamed by UE-V. This could cause the Office 2010 package size to be quite large or result in unpredictable conflicts with 2013, particularly if Office 365 is used.

WORKAROUND: Install only one version of Office or limit which settings are synchronized by UE-V.

#### **Uninstall and re-install of Windows 8 applications reverts settings to initial state**

While using UE-V settings synchronization for a Windows 8 application, if the user uninstalls the application and then reinstalls the application, the application's settings revert to their default values. This happens because the uninstall removes the local (cached) copy of the application's settings but does not remove the local UE-V settings package. When the application is reinstalled and launched, UE-V gather the application settings that were reset to the application defaults and then uploads the default settings to the central storage location. Other computers running the application then download the default settings. This behavior is identical to the behavior of desktop applications.

WORKAROUND: None.

#### **UE-V does not support roaming settings between 32-bit and 64-bit versions of Microsoft Office**

We recommend that you install the 32-bit version of Microsoft Office for both 32-bit and 64-bit operating systems. To choose the Microsoft Office version that you need, click [here](#). UE-V supports roaming settings between identical architecture versions of Office. For example, 32-bit Office settings will roam between all 32-bit Office instances. UE-V does not support roaming settings between 32-bit and 64-bit versions of Office.

WORKAROUND: None

#### **Favicons that are associated with Internet Explorer 9 favorites do not roam**

The favicons that are associated with Internet Explorer 9 favorites are not roamed by User Experience Virtualization and do not appear when the favorites first appear on a new computer.

WORKAROUND: Favicons will appear with their associated favorites once the bookmark is used and cached in the Internet Explorer 9 browser.

#### **File settings paths are stored in registry**

Some application settings store the paths of their configuration and settings files as values in the registry. The files that are referenced as paths in the registry must be synchronized when settings are roamed between computers.

WORKAROUND: Use folder redirection or some other technology to ensure that any files that are referenced as file settings paths are present and placed in the same location on all computers where settings roam.

#### **Long Settings Storage Paths could cause an error**

Keep settings storage paths as short as possible. Long paths could prevent resolution or synchronization. UE-V uses the Settings storage path as part of the calculated path to store settings. That path is calculated in the following way: settings storage path + "settingspackages" + package dir (template ID) + package name (template ID) + .pkgx. If that calculated path exceeds 260 characters, package storage will fail and generate the following error message in the UE-V operational event log:

[boost::filesystem::copy\_file: The system cannot find the path specified]

To check the operational log events, open the Event Viewer and navigate to Applications and Services Logs / Microsoft / User Experience Virtualization / Logging / Operational.

WORKAROUND: None.

#### **Some operating system settings only roam between like operating system versions**

Operating system settings for Narrator and currency characters specific to the locale (i.e. language and regional settings) will only roam across like operating system versions of Windows. For example, currency characters will not roam between Windows 7 and Windows 8.

WORKAROUND: None

## Hotfixes and Knowledge Base articles for UE-V

This section contains hotfixes and KB articles for UE-V.

KB ARTICLE	TITLE	LINK
3018608	UE-V - TemplateConsole.exe crashes when UE-V WMI classes are missing	<a href="http://support.microsoft.com/kb/3018608">support.microsoft.com/kb/3018608</a>
2903501	UE-V: User Experience Virtualization (UE-V) compatibility with user profiles	<a href="http://support.microsoft.com/kb/2903501">support.microsoft.com/kb/2903501</a>
2770042	UE-V Registry Settings	<a href="http://support.microsoft.com/kb/2770042">support.microsoft.com/kb/2770042</a>
2847017	Internet Explorer settings replicated by UE-V	<a href="http://support.microsoft.com/kb/2847017">support.microsoft.com/kb/2847017</a>
2769631	How to repair a corrupted UE-V install	<a href="http://support.microsoft.com/kb/2769631">support.microsoft.com/kb/2769631</a>
2850989	Migrating MAPI profiles with Microsoft UE-V is not supported	<a href="http://support.microsoft.com/kb/2850989">support.microsoft.com/kb/2850989</a>
2769586	UE-V roams empty folders and registry keys	<a href="http://support.microsoft.com/kb/2769586">support.microsoft.com/kb/2769586</a>
2782997	How To Enable Debug Logging in Microsoft User Experience Virtualization (UE-V)	<a href="http://support.microsoft.com/kb/2782997">support.microsoft.com/kb/2782997</a>
2769570	UE-V does not update the theme on RDS or VDI sessions	<a href="http://support.microsoft.com/kb/2769570">support.microsoft.com/kb/2769570</a>
2850582	How To Use Microsoft User Experience Virtualization With App-V Applications	<a href="http://support.microsoft.com/kb/2850582">support.microsoft.com/kb/2850582</a>
3041879	Current file versions for Microsoft User Experience Virtualization	<a href="http://support.microsoft.com/kb/3041879">support.microsoft.com/kb/3041879</a>
2843592	Information on User Experience Virtualization and High Availability	<a href="http://support.microsoft.com/kb/2843592">support.microsoft.com/kb/2843592</a>

## Have a suggestion for UE-V?

Add or vote on suggestions on the [User Experience Virtualization feedback site](#).

For UE-V issues, use the [UE-V TechNet Forum](#).

### Additional resources for this feature

- [User Experience Virtualization](#)
- [Prepare a UE-V Deployment](#)
- [Upgrade to UE-V for Windows 10](#)
- [Administering UE-V](#)

- Troubleshooting UE-V
- Technical Reference for UE-V

# Upgrade to UE-V for Windows 10

6/20/2017 • 3 min to read • [Edit Online](#)

## Applies to

- Windows 10, version 1607

If you're already using UE-V 2.x and you're planning to upgrade user devices to Windows 10, version 1607 or later releases, you need to make only a few adjustments to your existing environment. These steps are explained in more detail below.

1. Upgrade user devices to Windows 10, version 1607 or later release.
2. Verify that UE-V settings were migrated correctly.
3. Set the template storage path to your current template store.
4. Enable the UE-V service on user devices.
5. Install the UE-V template generator if you want to synchronize application settings for custom applications.

**Important** You can upgrade your existing UE-V installation to Windows 10, version 1607 from UE-V versions 2.1 or 2.0 only. If you are using a previous version of UE-V, you'll need to upgrade from that version to UE-V 2.x before you upgrade to Windows 10, version 1607..

## Upgrade user devices to Windows 10, version 1607

Performing an in-place upgrade on user devices automatically installs the UE-V service, updates the settings location path, and migrates users' UE-V settings. See the [Windows 10 documentation for IT Pros](#) for information about upgrading user devices to Windows 10.

## Verify that UE-V settings were migrated correctly

After upgrading a user device to Windows 10, version 1607, it's important to verify that UE-V settings and template registrations were migrated correctly during the upgrade. You can verify UE-V settings using Windows Powershell or the device's registry.

### To verify UE-V settings using Windows PowerShell

1. Run PowerShell as Administrator, type **Get-UEVConfiguration**, and press ENTER to view current configurations.
2. Check that the settings were successfully updated.
3. Type **Get-UEVTemplate** and press ENTER to check that your templates are still registered.

**Note** You'll need to register the NotePad template again after you upgrade the device to Windows 10.

### To verify UE-V settings using the device's registry

1. In a command prompt, run **Regedit** as Administrator.
2. Navigate to **HKEY\_LOCAL\_MACHINE\Software\Microsoft\UEV\Agent\Configuration**.
3. Verify that the settings storage path and the settings template catalog path are pointing to the same

locations as before you upgraded the device to Windows 10.

## Set the template storage path to your current template store

Template Settings Storage Path will not automatically migrate. Run Set-UEVConfiguration in PowerShell or use the settings storage path Group Policy to configure and point to your current settings storage folder.

## Enable the UE-V service on user devices

The UE-V service is the client-side component that captures user-personalized application and Windows settings and saves them in settings packages. Settings packages are built, locally stored, and copied to the settings storage location.

With Windows 10, version 1607 and later, the UE-V service replaces the UE-V Agent and no longer requires a separate download and installation. Enable the service on user devices to start using UE-V. You can enable the service with the Group Policy editor or with Windows PowerShell.

**Important** The UE-V Agent used in prior releases of UE-V is replaced with the UE service. The UE-V service included with Windows 10, version 1607 and later releases, does not include the agent user interface and is configurable through cmdlets or registry settings only.

### To enable the UE-V service with Group Policy

1. Open the device's **Group Policy Editor**.
2. Navigate to **Computer Configuration > Administrative Templates > Windows Components > Microsoft User Experience Virtualization**.
3. Run **Enable UEV**
4. Restart the device.

### To enable the UE-V service with Windows PowerShell

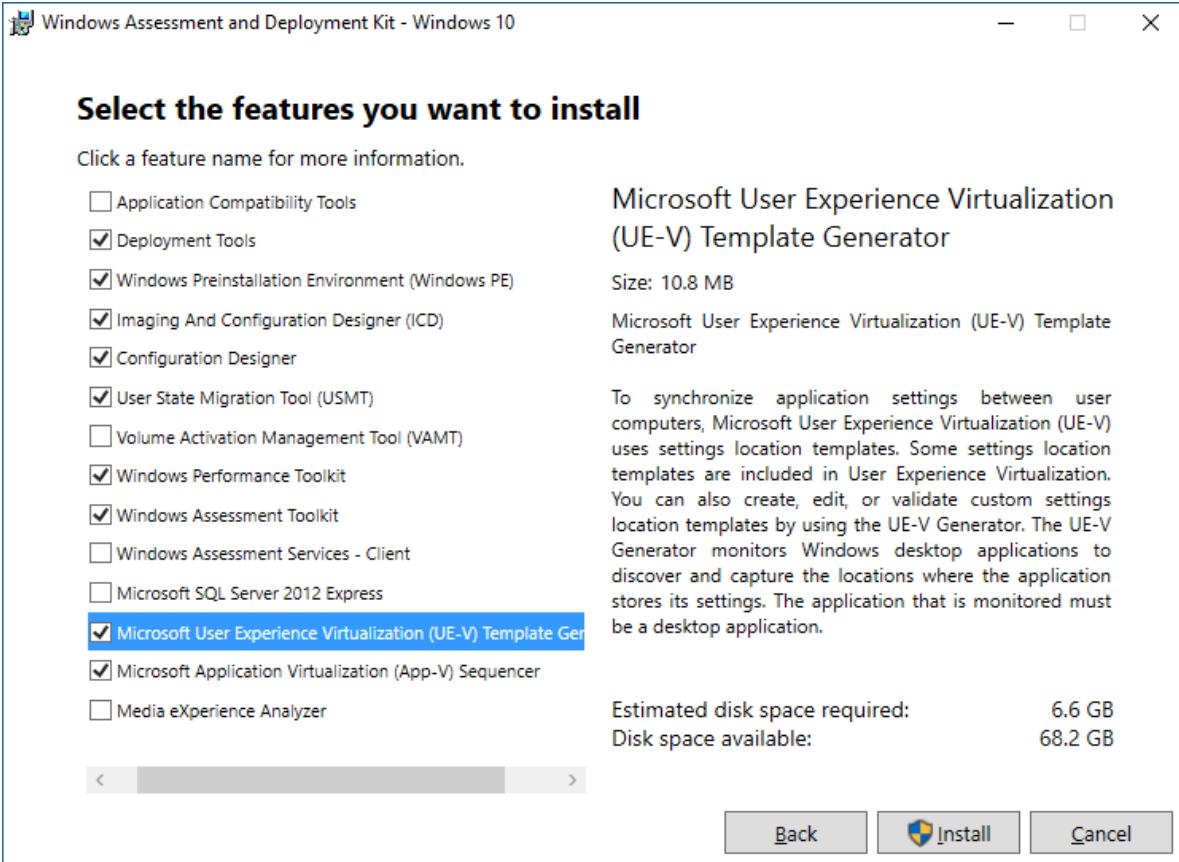
1. Run PowerShell as Administrator, type **Enable-UEV**, and press ENTER.
2. Restart the device.
3. Type **Get-UEVStatus** and press ENTER to verify that the service was successfully enabled.

## Install the UE-V template generator

The UE-V template generator is included in the Windows Assessment and Deployment Kit (ADK) for Windows 10.

### To install the UE-V template generator

1. Go to [Download the Windows ADK](#) to access the ADK.
2. Select the **Get Windows ADK for Windows 10** button on this page to start the ADK installer. On the screen pictured below, select **Microsoft User Experience Virtualization (UE-V) Template Generator** and then select **Install**.



3. To open the generator, open the **Start** menu and navigate to **Windows Kits > Microsoft User Experience Virtualization (UE-V) Template Generator**.

## Have a suggestion for UE-V?

Add or vote on suggestions on the [User Experience Virtualization feedback site](#).

For UE-V issues, use the [UE-V TechNet Forum](#).

## Other resources for this feature

- [UE-V Release Notes](#)
- [Prepare a UE-V Deployment](#)
- [Administer UE-V](#)
- [Migrating settings packages](#)
- [Technical Reference for UE-V](#)

# Prepare a UE-V Deployment

6/20/2017 • 18 min to read • [Edit Online](#)

## Applies to

- Windows 10, version 1607

Before you deploy User Experience Virtualization (UE-V), review this topic for important information about the type of deployment you're planning and for preparations you can make beforehand so that your deployment is successful. If you leave this page, be sure to come back and read through the planning information in this topic.

## Plan your UE-V deployment

With UE-V, you can synchronize user-defined application and operating system settings across all the devices that a user works from. Use UE-V to synchronize settings for Windows applications and custom applications, such as third-party and line of business applications.

Whether you want to synchronize settings for only default Windows applications or for both Windows and custom applications, you'll need to first deploy the features required to use UE-V.

### [Deploy required UE-V features](#)

- [Define a settings storage location](#)
- [Enable the UE-V service](#) on user computers

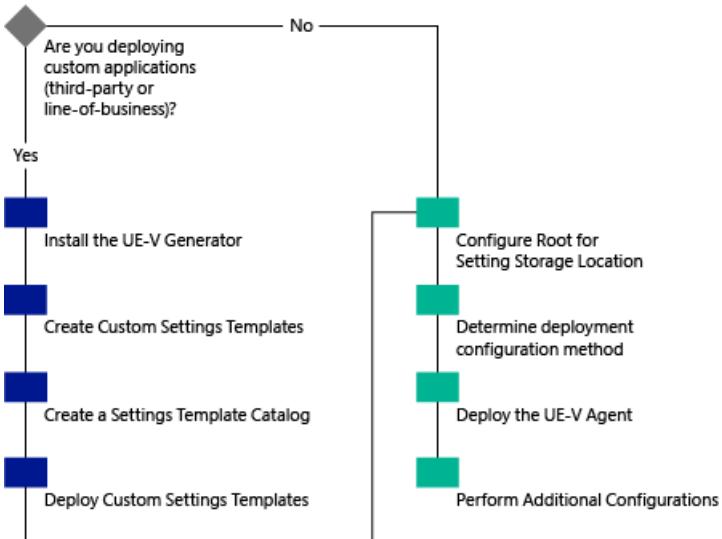
If you want to use UE-V to synchronize user-defined settings for custom applications (third-party or line-of-business), you'll need to install and configure these optional additional UE-V features:

### [Deploy UE-V for custom applications](#)

- [Install the UE-V template generator](#) so you can create, edit, and validate the custom settings location templates required to synchronize custom application settings
- [Create custom settings location templates](#) with the UE-V template generator
- [Deploy a UE-V settings template catalog](#) to store your custom settings location templates

The workflow diagram below illustrates a typical UE-V deployment and the decisions you need to be prepared to make.

## UE-V Deployment Preparation



## Planning a UE-V deployment

Review the following topics to determine which UE-V components you'll be deploying.

- [Decide whether to synchronize settings for custom applications](#)

If you want to synchronize settings for custom applications, you'll need to install the UE-V template generator. Use the generator to create custom settings location templates, which involves the following tasks:

- Review the [settings that are synchronized automatically in a UE-V deployment](#).
- [Determine whether you need settings synchronized for other applications](#).
- Review [other considerations for deploying UE-V](#), including high availability and capacity planning.
- [Confirm prerequisites and supported configurations for UE-V](#)

## Decide whether to synchronize settings for custom applications

In a UE-V deployment, many settings are automatically synchronized. You can also customize UE-V to synchronize settings for other applications, such as line-of-business and third-party apps.

Deciding if you want UE-V to synchronize settings for custom applications is an essential part of planning your UE-V deployment. The topics in this section will help you make that decision.

### Settings automatically synchronized in a UE-V deployment

This section explains which settings are synchronized by default in UE-V, including:

- Desktop applications that are synchronized by default
- Windows desktop settings that are synchronized by default
- A statement of support for Windows applications setting synchronization

For downloadable UE-V templates, see:

- [Microsoft Authored Office 2016 UE-V Templates](#)
- [User Experience Virtualization \(UE-V\) settings templates for Microsoft Office](#) (for Office 2013 and Office 2010)

### Desktop applications synchronized by default in UE-V

When you enable the UE-V service on user devices, it registers a default group of settings location templates that

capture settings values for these common Microsoft applications.

APPLICATION CATEGORY	DESCRIPTION
Microsoft Office 2016 applications <a href="#">Download a list of all settings synced</a>	Microsoft Access 2016 Microsoft Lync 2016 Microsoft Excel 2016 Microsoft OneNote 2016 Microsoft Outlook 2016 Microsoft PowerPoint 2016 Microsoft Project 2016 Microsoft Publisher 2016 Microsoft SharePoint Designer 2013 (not updated for 2016) Microsoft Visio 2016 Microsoft Word 2016 Microsoft Office Upload Manager Microsoft Infopath has been removed (deprecated) from the Office 2016 suite
Microsoft Office 2013 applications <a href="#">Download a list of all settings synced</a>	Microsoft Word 2013 Microsoft Excel 2013 Microsoft Outlook 2013 Microsoft Access 2013 Microsoft Project 2013 Microsoft PowerPoint 2013 Microsoft Publisher 2013 Microsoft Visio 2013 Microsoft InfoPath 2013 Microsoft Lync 2013 Microsoft OneNote 2013 Microsoft SharePoint Designer 2013 Microsoft Office 2013 Upload Center Microsoft OneDrive for Business 2013
Microsoft Office 2010 applications <a href="#">Download a list of all settings synced</a>	Microsoft Word 2010 Microsoft Excel 2010 Microsoft Outlook 2010 Microsoft Access 2010 Microsoft Project 2010 Microsoft PowerPoint 2010 Microsoft Publisher 2010 Microsoft Visio 2010 Microsoft SharePoint Workspace 2010 Microsoft InfoPath 2010 Microsoft Lync 2010 Microsoft OneNote 2010 Microsoft SharePoint Designer 2010
Browser options: Internet Explorer 11 and 10	Synchronize favorites, home page, tabs, and toolbars. <b>Note</b> UE-V does not roam settings for Internet Explorer cookies.
Windows accessories	Microsoft NotePad, WordPad

**Notes** An Outlook profile must be created for any device on which a user wants to sync their Outlook signature. If the profile is not already created, the user can create one and then restart Outlook on that device to enable signature synchronization.

UE-V does not synchronize settings between the Microsoft Calculator in Windows 10 and the Microsoft Calculator in previous operating systems.

## Windows settings synchronized by default

UE-V includes settings location templates that capture settings values for these Windows settings.

WINDOWS SETTINGS	DESCRIPTION	APPLY ON	EXPORT ON	DEFAULT STATE
Desktop background	Currently active desktop background or wallpaper	Log on, unlock, remote connect, Scheduled Task events	Log off, lock, remote disconnect, or scheduled task interval	Enabled
Ease of Access	Accessibility and input settings, Microsoft Magnifier, Narrator, and on-Screen Keyboard	Log on only	Log off or scheduled task interval	Enabled
Desktop settings	Start menu and Taskbar settings, folder options, default desktop icons, additional clocks, and region and language settings	Log on only	Log off or scheduled task	Enabled

**Important** UE-V roams taskbar settings between Windows 10 devices. However, UE-V does not synchronize taskbar settings between Windows 10 devices and devices running previous operating systems versions.

SETTINGS GROUP	CATEGORY	CAPTURE	APPLY
<b>Application Settings</b>	Windows applications	Close application Windows application settings change event	Start the UE-V App Monitor at startup Open app Windows application settings change event Arrival of a settings package
	Desktop applications	Application closes	Application opens and closes
<b>Desktop settings</b>	Desktop background	Lock or log off	Log on, unlock, remote connect, notification of new package arrival, or scheduled task runs
	Ease of Access (Common – Accessibility, Narrator, Magnifier, On-Screen-Keyboard)	Lock or Log off	Log on
	Ease of Access (Shell - Audio, Accessibility, Keyboard, Mouse)	Lock or log off	Log on, unlock, remote connect, notification of new package arrival, or scheduled task runs
	Desktop settings	Lock or log off	Log on

## UE-V-support for Windows applications

For Windows applications, the application developer specifies which user settings are synchronized. You can specify which Windows apps are enabled for settings synchronization.

To display a list of Windows applications that can synchronize settings with their package family name, enabled status, and enabled source, open a Windows PowerShell window, type Get-UevAppxPackage, and press ENTER.

**Note** Starting in Windows 10, version 1607, you can configure UE-V to not synchronize Windows applications settings if the device is configured to use Enterprise State Roaming.

### **UE-V-support for roaming printers**

Users can print to their saved network printers, including their default network printer, from any network device.

Printer roaming in UE-V requires one of these scenarios:

- The print server can download the required driver when it roams to a new device.
- The driver for the roaming network printer is pre-installed on any device that needs to access that network printer.
- The printer driver can be imported from Windows Update.

**Note** The UE-V printer roaming feature does not roam printer settings or preferences, such as printing double-sided.

### **Determine whether you need settings synchronized for other applications**

After you have reviewed the settings that are synchronized automatically in a UE-V deployment, you'll need to decide whether to synchronize settings for other applications as your decision will determine how you deploy UE-V throughout your enterprise.

As an administrator, when you consider which desktop applications to include in your UE-V solution, consider which settings can be customized by users, and how and where the application stores its settings. Not all desktop applications have settings that can be customized or that are routinely customized by users. In addition, not all desktop applications settings can be synchronized safely across multiple devices or environments.

In general, you can synchronize settings that meet the following criteria:

- Settings that are stored in user-accessible locations. For example, do not synchronize settings that are stored in System32 or outside the HKEY\_CURRENT\_USER (HKCU) section of the registry.
- Settings that are not specific to the particular device. For example, exclude network shortcuts or hardware configurations.
- Settings that can be synchronized between computers without risk of corrupted data. For example, do not use settings that are stored in a database file.

### **Checklist for evaluating custom applications**

If you've decided that you need to synchronize settings for custom applications, use this checklist to determine which applications you'll include.

	<b>DESCRIPTION</b>
<input type="checkbox"/>	Does this application contain settings that the user can customize?
<input type="checkbox"/>	Is it important for the user that these settings are synchronized?

	DESCRIPTION
<input type="checkbox"/>	Are these user settings already managed by an application management or settings policy solution? UE-V applies application settings at application startup and Windows settings at logon, unlock, or remote connect events. If you use UE-V with other settings sharing solutions, users might experience inconsistency across synchronized settings.
<input type="checkbox"/>	Are the application settings specific to the computer? Application preferences and customizations that are associated with hardware or specific computer configurations do not consistently synchronize across sessions and can cause a poor application experience.
<input type="checkbox"/>	Does the application store settings in the Program Files directory or in the file directory that is located in the <b>Users\ [User name] \AppData\LocalLow</b> directory? Application data that is stored in either of these locations usually should not synchronize with the user, because this data is specific to the computer or because the data is too large to synchronize.
<input type="checkbox"/>	Does the application store any settings in a file that contains other application data that should not synchronize? UE-V synchronizes files as a single unit. If settings are stored in files that include application data other than settings, then synchronizing this additional data can cause a poor application experience.
<input type="checkbox"/>	How large are the files that contain the settings? The performance of the settings synchronization can be affected by large files. Including large files can affect the performance of settings synchronization.

## Other considerations when preparing a UE-V deployment

You should also consider these things when you are preparing to deploy UE-V:

- [Managing credentials synchronization](#)
- [Windows applications settings synchronization](#)
- [Custom UE-V settings location templates](#)
- [Unintentional user settings configurations](#)
- [Performance and capacity](#)
- [High availability](#)
- [Computer clock synchronization](#)

### Managing credentials synchronization in UE-V

Many enterprise applications, including Microsoft Outlook, Lync, and Skype for Business prompt users for their domain credentials when they log in. Users have the option of saving their credentials to disk to prevent having to enter them every time they open these applications. Enabling roaming credentials synchronization lets users save their credentials on one computer and avoid re-entering them on every computer they use in their environment. Users can synchronize some domain credentials with UE-V.

**Important** Credentials synchronization is disabled by default. You must explicitly enable credentials synchronization after you enable the UE-V service to implement this feature.

UE-V can synchronize enterprise credentials, but does not roam credentials intended only for use on the local device.

Credentials are synchronous settings, meaning that they are applied to users' profiles the first time they log on to their devices after UE-V synchronizes.

Credentials synchronization is managed by its own settings location template, which is disabled by default. You can enable or disable this template through the same methods used for other templates. The template identifier for this feature is RoamingCredentialSettings.

**Important** If you are using Active Directory Credential Roaming in your environment, we recommend that you do not enable the UE-V credential roaming template. Instead, use PowerShell or Group Policy to enable credentials synchronization. Note that credentials are encrypted during synchronization.

**PowerShell:** Enter this PowerShell cmdlet to enable credential synchronization:

```
Enable-UevTemplate RoamingCredentialSettings
```

**Copy**

Use this PowerShell cmdlet to disable credential synchronization:

```
Disable-UevTemplate RoamingCredentialSettings
```

**Copy**

**Group Policy:** You must edit the Group Policy administrative template for UE-V, which is included in Windows 10, version 1607, to enable credential synchronization through group policy. Credentials synchronization is managed in Windows settings. To manage this feature with Group Policy, enable the **Synchronize Windows** settings policy.

1. Open Group Policy Editor and navigate to **User Configuration > Administrative Templates > Windows Components > Microsoft User Experience Virtualization**.
2. Double-click **Synchronize Windows settings**.
3. If this policy is enabled, you can enable credentials synchronization by checking the **Roaming Credentials** check box, or disable credentials synchronization by unchecking it.
4. Click **OK**.

### **Credential locations synchronized by UE-V**

Credential files saved by applications into the following locations are synchronized:

- %UserProfile%\AppData\Roaming\Microsoft\Credentials\
- %UserProfile%\AppData\Roaming\Microsoft\Crypto\
- %UserProfile%\AppData\Roaming\Microsoft\Protect\
- %UserProfile%\AppData\Roaming\Microsoft\SystemCertificates\

Credentials saved to other locations are not synchronized by UE-V.

### **Windows applications settings synchronization**

UE-V manages Windows application settings synchronization in three ways:

- **Sync Windows applications:** Allow or deny any Windows application synchronization
- **Windows applications list:** Synchronize a list of Windows applications
- **Unlisted default sync behavior:** Determine the synchronization behavior of Windows applications that are not in the Windows applications list.

For more information, see the [Windows Application List](#).

### Custom UE-V settings location templates

If you are deploying UE-V to synchronize settings for custom applications, you'll use the UE-V template generator to create custom settings location templates for those desktop applications. After you create and test a custom settings location template in a test environment, you can deploy the settings location templates to user devices.

Custom settings location templates must be deployed with an existing deployment infrastructure, such as an enterprise software distribution method, including System Center Configuration Manager, with preferences, or by configuring a UE-V settings template catalog. Templates that are deployed with Configuration Manager or Group Policy must be registered using UE-V WMI or Windows PowerShell.

For more information about custom settings location templates, see [Deploy UE-V with custom applications](#). For more information about using UE-V with Configuration Manager, see [Configuring UE-V with System Center Configuration Manager](#).

### Prevent unintentional user settings configuration

UE-V downloads new user settings information from a settings storage location and applies the settings to the local device in these instances:

- Each time an application is started that has a registered UE-V template
- When a user logs on to a device
- When a user unlocks a device
- When a connection is made to a remote desktop device running UE-V
- When the Sync Controller Application scheduled task is run

If UE-V is installed on computer A and computer B, and the settings that you want for the application are on computer A, then computer A should open and close the application first. If the application is opened and closed on computer B first, then the application settings on computer A are configured to the application settings on computer B. Settings are synchronized between computers on per-application basis. Over time, settings become consistent between computers as they are opened and closed with preferred settings.

This scenario also applies to Windows settings. If the Windows settings on computer B should be the same as the Windows settings on computer A, then the user should log on and log off computer A first.

If the user settings that the user wants are applied in the wrong order, they can be recovered by performing a restore operation for the specific application or Windows configuration on the computer on which the settings were overwritten. For more information, see [Manage Administrative Backup and Restore in UE-V](#).

### Performance and capacity planning

Specify your requirements for UE-V with standard disk capacity and network health monitoring.

UE-V uses a Server Message Block (SMB) share for the storage of settings packages. The size of settings packages varies depending on the settings information for each application. While most settings packages are small, the synchronization of potentially large files, such as desktop images, can result in poor performance, particularly on slower networks.

To reduce problems with network latency, create settings storage locations on the same local networks where the users' computers reside. We recommend 20 MB of disk space per user for the settings storage location.

By default, UE-V synchronization times out after 2 seconds to prevent excessive lag due to a large settings package. You can configure the SyncMethod=SyncProvider setting by using [Group Policy objects](#).

### High availability for UE-V

The UE-V settings storage location and settings template catalog support storing user data on any writable share. To ensure high availability, follow these criteria:

- Format the storage volume with an NTFS file system.
- The share can use Distributed File System (DFS) replication, but Distributed File System Replication (DFSR) is specifically not supported. Distributed File System Namespaces (DFSN) are supported. For detailed information, see:
  - [Information about roaming profiles from the Directory Services team](#)
  - [Information about Microsoft support policy for a DFS-R and DFS-N deployment scenario](#)

In addition, because SYSVOL uses DFSR for replication, SYSVOL cannot be used for UE-V data file replication.

- Configure the share permissions and NTFS access control lists (ACLs) as specified in [Deploying the settings storage location for UE-V](#).
- Use file server clustering along with the UE-V service to provide access to copies of user state data in the event of communications failures.
- You can store the settings storage path data (user data) and settings template catalog templates on clustered shares, on DFSN shares, or on both.

### Synchronize computer clocks for UE-V settings synchronization

Computers that run the UE-V service must use a time server to maintain a consistent settings experience. UE-V uses time stamps to determine if settings must be synchronized from the settings storage location. If the computer clock is inaccurate, older settings can overwrite newer settings, or the new settings might not be saved to the settings storage location.

## Confirm prerequisites and supported configurations for UE-V

Before you proceed, ensure that your environment meets these requirements for using UE-V.

OPERATING SYSTEM	EDITION	SERVICE PACK	SYSTEM ARCHITECTURE	WINDOWS POWERSHELL	MICROSOFT .NET FRAMEWORK
Windows 10, version 1607	Windows 10 for Enterprise	NA	32-bit or 64-bit	Windows PowerShell 3.0 or higher	.NET Framework 4.5 or higher
Windows 8 and Windows 8.1	Enterprise or Pro	None	32-bit or 64-bit	Windows PowerShell 3.0 or higher	.NET Framework 4.5 or higher
Windows Server 2012 and Windows Server 2012 R2	Standard or Datacenter	None	64-bit	Windows PowerShell 3.0 or higher	.NET Framework 4.5 or higher

### Note

- Windows Server 2012 operating systems come with .NET Framework 4.5 installed. The Windows 10 operating system comes with .NET Framework 4.6 installed.
- The “Delete Roaming Cache” policy for mandatory profiles is not supported with UE-V and should not be used.

There are no special random access memory (RAM) requirements specific to UE-V.

### **Synchronization of settings through the Sync Provider**

Sync Provider is the default setting for users and synchronizes a local cache with the settings storage location in these instances:

- Log on/log off
- Lock/unlock
- Remote desktop connect/disconnect
- Application open/close

A scheduled task manages this synchronization of settings every 30 minutes or through trigger events for certain applications. For more information, see [Changing the frequency of UE-V scheduled tasks](#).

The UE-V service synchronizes user settings for devices that are not always connected to the enterprise network (remote devices and laptops) and devices that are always connected to the network (devices that run Windows Server and host virtual desktop interface (VDI) sessions).

**Synchronization for computers with always-available connections** When you use UE-V on devices that are always connected to the network, you must configure the UE-V service to synchronize settings by using the *SyncMethod=None* parameter, which treats the settings storage server as a standard network share. In this configuration, the UE-V service can be configured to notify if the import of the application settings is delayed.

Enable this configuration using one of these methods:

- After you enable the UE-V service, use the Settings Management feature in System Center Configuration Manager or the UE-V ADMX templates (installed with Windows 10, version 1607) to push the *SyncMethod = None* configuration.
- Use Windows PowerShell or Windows Management Instrumentation (WMI) to set the *SyncMethod = None* configuration.

Restart the device to allow the settings to synchronize.

- >**Note** These methods do not work for pooled virtual desktop infrastructure (VDI) environments.

**Note** If you set *SyncMethod = None*, any settings changes are saved directly to the server. If the network connection to the settings storage path is not found, then the settings changes are cached on the device and are synchronized the next time that the sync provider runs. If the settings storage path is not found and the user profile is removed from a pooled VDI environment on log off, settings changes are lost and the user must reapply the change when the computer is reconnected to the settings storage path.

**Synchronization for external sync engines** The *SyncMethod=External* parameter specifies that if UE-V settings are written to a local folder on the user device, then any external sync engine (such as OneDrive for Business, Work Folders, Sharepoint, or Dropbox) can be used to apply these settings to the different devices that users access.

**Support for shared VDI sessions** UE-V supports VDI sessions that are shared among end users. You can register and configure a special VDI template, which ensures that UE-V keeps all of its functionality intact for

non-persistent VDI sessions.

**Note** If you do not enable VDI mode for non-persistent VDI sessions, certain features do not work, such as [back-up/restore and last known good \(LKG\)](#).

The VDI template is provided with UE-V and is typically available here after installation:

C:\ProgramData\Microsoft\UEV\InboxTemplates

### Prerequisites for UE-V template generator support

Install the UE-V template generator on the device that is used to create custom settings location templates. This device should be able to run the applications that you want to synchronize settings for. You must be a member of the Administrators group on the device that runs the UE-V template generator software.

The UE-V template generator must be installed on a device that uses an NTFS file system. The UE-V template generator software requires .NET Framework 4. For more information, see [Use UE-V with custom applications](#).

## Have a suggestion for UE-V?

Add or vote on suggestions on the [User Experience Virtualization feedback site](#).

For UE-V issues, use the [UE-V TechNet Forum](#).

## Other resources for this feature

- [User Experience Virtualization overview](#)
- [Get started with UE-V](#)
- [Upgrade to UE-V for Windows 10](#)
- [Administering UE-V](#)
- [Troubleshooting UE-V](#)
- [Technical Reference for UE-V](#)

# Deploy required UE-V features

6/20/2017 • 6 min to read • [Edit Online](#)

## Applies to

- Windows 10, version 1607

To get up and running with User Experience Virtualization (UE-V), install and configure the following features.

- Deploy a settings storage location** that is accessible to end users.

This is a standard network share that stores and retrieves user settings.

- Choose the configuration method for UE-V**

You can deploy and configure UE-V with common management tools including group policy, Configuration Manager, or Windows Management Infrastructure and PowerShell.

- Enable the UE-V service** on user devices.

With Windows 10, version 1607, UE-V is installed automatically. You need to enable the UE-V service on each user device you want to include in your UE-V environment.

The topics in this section describe how to deploy these features.

## Deploy a UE-V Settings Storage Location

UE-V requires a location in which to store user settings in settings package files. You can configure this settings storage location in one of these ways:

- Create your own settings storage location
- Use existing Active Directory for your settings storage location

**Note** As a matter of [performance and capacity planning](#) and to reduce problems with network latency, create settings storage locations on the same local networks where the users' devices reside. We recommend 20 MB of disk space per user for the settings storage location.

### Create a UE-V Settings Storage Location

Before you define the settings storage location, you must create a root directory with read/write permissions for users who store settings on the share. The UE-V service creates user-specific folders under this root directory.

The settings storage location is defined by setting the `SettingsStoragePath` configuration option, which you can configure by using one of these methods:

- Through [Group Policy](#) settings
- With the [System Center Configuration Pack for UE-V](#)
- With [Windows PowerShell](#) or [Windows Management Instrumentation \(WMI\)](#)

The path must be in a universal naming convention (UNC) path of the server and share. For example, `\Server\Settingsshare\`. This configuration option supports the use of variables to enable specific synchronization scenarios. For example, you can use the `%username%\%computername%` variables to preserve the end user settings experience in these scenarios:

- End users that use multiple physical devices in your enterprise
- Enterprise computers that are used by multiple end users

The UE-V service dynamically creates a user-specific settings storage path, with a hidden system folder named **SettingsPackages**, based on the configuration setting of **SettingsStoragePath**. The service reads and writes settings to this location as defined by the registered UE-V settings location templates.

**UE-V settings are determined by a "Last write wins" rule:** If the settings storage location is the same for a user with multiple managed computers, one UE-V service reads and writes to the settings location independently of services running on other computers. The last written settings and values are the ones applied when the service next reads from the settings storage location.

**Deploy the settings storage location:** Follow these steps to define the settings storage location rather than using your existing Active Directory agent. You should limit access to the settings storage share to those users that require it, as shown in the tables below.

#### To deploy the UE-V network share

1. Create a new security group for UE-V users.
2. Create a new folder on the centrally located computer that stores the UE-V settings packages, and then grant UE-V users access with group permissions to the folder. The administrator who supports UE-V must have permissions to this shared folder.
3. Set the following share-level Server Message Block (SMB) permissions for the settings storage location folder.

USER ACCOUNT	RECOMMENDED PERMISSIONS
Everyone	No permissions
Security group of UE-V users	Full control

4. Set the following NTFS file system permissions for the settings storage location folder.

USER ACCOUNT	RECOMMENDED PERMISSIONS	FOLDER
Creator/owner	Full control	Subfolders and files only
Security group of UE-V users	List folder/read data, create folders/append data	This folder only

With this configuration, the UE-V service creates and secures a Settingspackage folder while it runs in the context of the user, and grants each user permission to create folders for settings storage. Users receive full control to their Settingspackage folder while other users cannot access it.

**Note** If you create the settings storage share on a computer running a Windows Server operating system, configure UE-V to verify that either the local Administrators group or the current user is the owner of the folder where settings packages are stored. To enable this additional security, specify this setting in the Windows Server Registry Editor:

1. Add a **REG\_DWORD** registry key named "**RepositoryOwnerCheckEnabled**" to **HKEY\_LOCAL\_MACHINE\Software\Microsoft\UEV\Agent\Configuration**.
2. Set the registry key value to 1.

#### Use Active Directory with UE-V

The UE-V service uses Active Directory (AD) by default if you don't define a settings storage location. In these cases, the UE-V service dynamically creates the settings storage folder under the root of the AD home directory of each user. However, if a custom directory setting is configured in AD, then that directory is used instead.

## Choose the Configuration Method for UE-V

You'll need to decide which configuration method you'll use to manage UE-V after deployment since this will be the configuration method you use to deploy the UE-V Agent. Typically, this is the configuration method that you already use in your environment, such as Windows PowerShell or Configuration Manager.

You can configure UE-V before, during, or after you enable the UE-V service on user devices, depending on the configuration method that you use.

- **Group Policy** You can use your existing Group Policy infrastructure to configure UE-V before or after you enable the UE-V service. The UE-V Group Policy ADMX template enables the central management of common UE-V service configuration options and includes settings to configure UE-V synchronization.

**Note** Starting with Windows 10, version 1607, UE-V ADMX templates are installed automatically.

Group Policy ADMX templates configure the synchronization settings for the UE-V service and enable the central management of common UE-V service configuration settings by using an existing Group Policy infrastructure.

Supported operating systems for the domain controller that deploys the Group Policy Objects include:

Windows Server 2012 and Windows Server 2012 R2

- **Configuration Manager** The UE-V Configuration Pack lets you use the Compliance Settings feature of System Center Configuration Manager to apply consistent configurations across sites where UE-V and Configuration Manager are installed.
- **Windows PowerShell and WMI** You can use scripted commands for Windows PowerShell and Windows Management Instrumentation (WMI) to modify the configuration of the UE-V service.

**Note** Registry modification can result in data loss, or the computer becomes unresponsive. We recommend that you use other configuration methods.

## Enable the UE-V service

The UE-V service is the client-side component that captures user-personalized application and Windows settings and saves them in settings packages. Settings packages are built, locally stored, and copied to the settings storage location.

Before enabling the UE-V service, you need to register the UE-V templates for first time use. In a PowerShell window, type **register-<TemplateName>** where **TemplateName** is the name of the UE-V template you want to register, and press ENTER.

**Note** With Windows 10, version 1607, you must register UE-V templates for all inbox and custom templates. This provides flexibility for only deploying the required templates.

With Windows 10, version 1607 and later, the UE-V service is installed on user devices. Enable the service to start using UE-V. You can enable the service with the Group Policy editor or with Windows PowerShell.

### To enable the UE-V service with Group Policy

1. Open the device's **Group Policy Editor**.

2. Navigate to **Computer Configuration > Administrative Templates > Windows Components > Microsoft User Experience Virtualization**.
3. Run **Enable UEV**.
4. Restart the device.

#### To enable the UE-V service with Windows PowerShell

1. In a PowerShell window, type **Enable-UEV** and press ENTER.
2. Restart the device.
3. In a PowerShell window, type **Get-UEVStatus** and press ENTER to verify that the UE-V service was successfully enabled.

## Have a suggestion for UE-V?

Add or vote on suggestions on the [User Experience Virtualization feedback site](#).

For UE-V issues, use the [UE-V TechNet Forum](#).

## Related topics

[Prepare a UE-V deployment](#)

[Deploy UE-V for use with custom applications](#)

[Upgrade to UE-V for Windows 10](#)

# Use UE-V with custom applications

6/20/2017 • 11 min to read • [Edit Online](#)

## Applies to

- Windows 10, version 1607

User Experience Virtualization (UE-V) uses XML files called **settings location templates** to monitor and synchronize application settings and Windows settings between user devices. By default, some settings location templates are included in UE-V. However, if you want to synchronize settings for desktop applications other than those included in the default templates, you can create your own custom settings location templates with the UE-V template generator.

After you've reviewed [Prepare a UE-V Deployment](#) and decided that you want to synchronize settings for custom applications (third-party, line-of-business, e.g.), you'll need to deploy the features of UE-V described in this topic.

To start, here are the main steps required to synchronize settings for custom applications:

- [Install the UE-V template generator](#)

Use the UEV template generator to create custom XML settings location templates.

- [Configure a UE-V settings template catalog](#)

You can define this path where custom settings location templates are stored.

- [Create custom settings location templates](#)

These custom templates let users sync settings for custom applications.

- [Deploy the custom settings location templates](#)

After you test the custom template to ensure that settings are synced correctly, you can deploy these templates in one of these ways:

- With your existing electronic software distribution solution, such as Configuration Manager
- With Group Policy preferences
- With a UE-V settings template catalog

**Note** Templates that are deployed with electronic software distribution methods or Group Policy must be registered with UE-V Windows Management Instrumentation (WMI) or Windows PowerShell.

## Prepare to deploy UE-V for custom applications

Before you start deploying the UE-V features that handle custom applications, review the following important information.

### The UE-V template generator

Use the UE-V template generator to monitor, discover, and capture the locations where Win32 applications store settings. The template generator does not create settings location templates for the following types of applications:

- Virtualized applications

- Applications that are offered through Terminal Services
- Java applications
- Windows applications

**Note** UE-V settings location templates cannot be created from virtualized applications or Terminal Services applications. However, settings that are synchronized by using the templates can be applied to those applications. To create templates that support Virtual Desktop Infrastructure (VDI) and Terminal Services applications, open a version of the Windows Installer (.msi) package of the application by using the UE-V template generator. For more information about synchronizing settings for virtual applications, see [Using UE-V with virtual applications](#).

**Excluded Locations:** The discovery process excludes locations that commonly store application software files that do not synchronize well between user computers or computing environments. By default, these are excluded:

- HKEY\_CURRENT\_USER registry keys and files to which the logged-on user cannot write values
- HKEY\_CURRENT\_USER registry keys and files that are associated with the core functionality of the Windows operating system
- All registry keys that are located in the HKEY\_LOCAL\_MACHINE hive
- Files that are located in Program Files directories
- Files that are located in Users \ [User name] \ AppData \ LocalLow
- Windows operating system files that are located in %Systemroot%

If registry keys and files that are stored in excluded locations are required to synchronize application settings, you can manually add the locations to the settings location template during the template creation process.

### Replace the default Microsoft templates

A default group of settings location templates for common Microsoft applications and Windows settings is included with Windows 10, version 1607. If you customize these templates, or create settings location templates to synchronize settings for custom applications, the UE-V service can be configured to use a settings template catalog to store the templates. In this case, you will need to include the default templates with the custom templates in the settings template catalog.

**Important** After you enable the UE-V service, you'll need to register the settings location templates using the `Register-UevTemplate` cmdlet in Windows PowerShell.

When you use Group Policy to configure the settings template catalog path, you can choose to replace the default Microsoft templates. If you configure the policy settings to replace the default Microsoft templates, all of the default Microsoft templates that are installed with Windows 10, version 1607 are deleted and only the templates that are located in the settings template catalog are used.

**Note** If there are customized templates in the settings template catalog that use the same ID as the default Microsoft templates, the Microsoft templates are ignored.

You can replace the default templates by using the UE-V Windows PowerShell features. To replace the default Microsoft template with Windows PowerShell, unregister all of the default Microsoft templates, and then register the customized templates.

Old settings packages remain in the settings storage location even if you deploy new settings location templates for an application. These packages are not read by the UE-V service, but neither are they automatically deleted.

## Install the UEV template generator

Use the UE-V template generator to create custom settings location templates that you can then distribute to user devices. You can also use the template generator to edit an existing template or validate a template that was created with another XML editor.

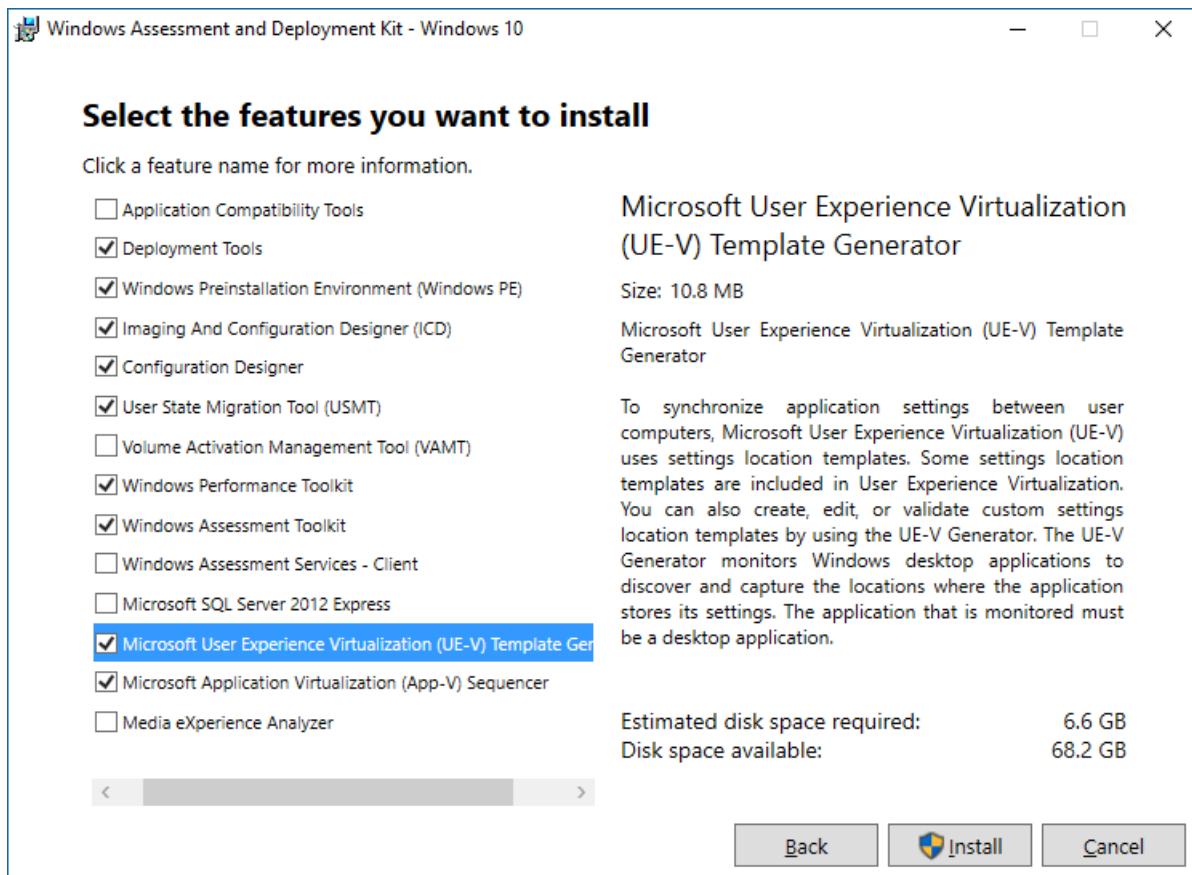
The UE-V template generator is included in the Windows Assessment and Deployment Kit (ADK) for Windows 10.

Install the UE-V template generator on a computer that you can use to create a custom settings location template. This computer should have the applications installed for which custom settings location templates need to be generated.

**Important** UE-V for Windows 10, version 1607 includes a new template generator. If you are upgrading from an existing UE-V installation, you'll need to use the new generator to create settings location templates. Templates created with previous versions of the UE-V template generator will continue to work.

### To install the UE-V template generator

1. Go to [Download the Windows ADK](#) to access the ADK.
2. Select the **Get Windows ADK for Windows 10** button on this page to start the ADK installer. On the window pictured below, select **Microsoft User Experience Virtualization (UE-V) Template Generator** and then select Install.



1. To open the generator, select **Microsoft Application Virtualization Generator** from the **Start** menu.
2. See [Working with Custom UE-V Templates and the UE-V Template Generator](#) for information about how to use the template generator.

### Deploy a settings template catalog

The UE-V settings template catalog is a folder path on UE-V computers or a Server Message Block (SMB) network share that stores all the custom settings location templates. The UE-V service checks this location one

time each day and updates its synchronization behavior, based on the templates in this folder.

The UE-V service checks this folder for templates that were added, updated, or removed. It registers new and changed templates and unregisters removed templates. By default, templates are registered and unregistered one time per day at 3:30 A.M. local time by the Task Scheduler and at system startup. To customize the frequency of this scheduled task, see [Changing the frequency of UE-V scheduled tasks](#).

You can configure the settings template catalog path with command-line options, Group Policy, WMI, or Windows PowerShell. Templates stored at the settings template catalog path are automatically registered and unregistered by a scheduled task.

### To configure the settings template catalog for UE-V

1. Create a new folder on the computer that stores the UE-V settings template catalog.
2. Set the following share-level (SMB) permissions for the settings template catalog folder.

USER ACCOUNT	RECOMMENDED PERMISSIONS
Everyone	No Permissions
Domain Computers	Read Permission Levels
Administrators	Read/Write Permission Levels

3. Set the following NTFS file system permissions for the settings template catalog folder.

USER ACCOUNT	RECOMMENDED PERMISSIONS	APPLY TO
Creator/Owner	Full Control	This Folder, Subfolders and Files
Domain Computers	List Folder Contents and Read	This Folder, Subfolders and Files
Everyone	No Permissions	No Permissions
Administrators	Full Control	This Folder, Subfolders and Files

4. Click **OK** to close the dialog boxes.

At a minimum, the network share must grant permissions for the Domain Computers group. In addition, grant access permissions for the network share folder to administrators who are to manage the stored templates.

### Create custom settings location templates

Use the UE-V template generator to create settings location templates for line-of-business applications or other custom applications. After you create the template for an application, deploy it to computers to synchronize settings for that application.

### To create a UE-V settings location template with the UE-V template generator

1. Click **Start > All Programs > Microsoft User Experience Virtualization > Microsoft User Experience Virtualization template generator**.
2. Click **Create a settings location template**.
3. Specify the application. Browse to the file path of the application (.exe) or the application shortcut (.lnk) for which you want to create a settings location template. Specify the command-line arguments, if any, and working directory, if any.

4. Click **Next** to continue.

**Note** Before the application is started, the system displays a prompt for **User Account Control**. Permission is required to monitor the registry and file locations that the application uses to store settings.

5. After the application starts, close the application. The UE-V template generator records the locations where the application stores its settings.
6. After the process is completed, click **Next** to continue.
7. Review and select the appropriate registry settings locations and settings file locations to synchronize for this application. The list includes the following two categories for settings locations:
  - **Standard:** Application settings that are stored in the registry under the HKEY\_CURRENT\_USER keys or in the file folders under \ **Users** \ [User name] \ **AppData** \ **Roaming**. The UE-V template generator includes these settings by default.
  - **Nonstandard:** Application settings that are stored outside the locations are specified in the best practices for settings data storage (optional). These include files and folders under **Users** \ [User name] \ **AppData** \ **Local**. Review these locations to determine whether to include them in the settings location template. Select the locations check boxes to include them.
8. Click **Next** to continue.
9. Review and edit any **Properties**, **Registry** locations, and **Files** locations for the settings location template.
  - Edit the following properties on the **Properties** tab:
    - **Application Name:** The application name that is written in the description of the program files properties.
    - **Program name:** The name of the program that is taken from the program file properties. This name usually has the .exe file name extension.
    - **Product version:** The product version number of the .exe file of the application. This property, in conjunction with the **File version**, helps determine which applications are targeted by the settings location template. This property accepts a major version number. If this property is empty, the settings location template applies to all versions of the product.
    - **File version:** The file version number of the .exe file of the application. This property, in conjunction with the **Product version**, helps determine which applications are targeted by the settings location template. This property accepts a major version number. If this property is empty, the settings location template applies to all versions of the program.
    - **template author name** (optional): The name of the settings location template author.
    - **template author email** (optional): The email address of the settings location template author.
  - The **Registry** tab lists the **Key** and **Scope** of the registry locations that are included in the settings location template. Edit the registry locations by using the **Tasks** drop-down menu. Tasks enable you to add new keys, edit the name or scope of existing keys, delete keys, and browse the registry where the keys are located. Use the **All Settings** scope to include all the registry settings under the specified key. Use the **All Settings and Subkeys** to include all the registry settings under the specified key, subkeys, and subkey settings.
  - The **Files** tab lists the file path and file mask of the file locations that are included in the settings location template. Edit the file locations by use of the **Tasks** drop-down menu. Tasks for file

locations enable you to add new files or folder locations, edit the scope of existing files or folders, delete files or folders, and open the selected location in Windows Explorer. Leave the file mask empty to include all files in the specified folder.

10. Click **Create**, and then click **Save** to save the settings location template on the computer.
11. Click **Close** to close the settings template wizard. Exit the UE-V template generator application.
12. After you have created the settings location template for an application, test the template. Deploy the template in a lab environment before you put it into production in the enterprise.

See [Application template schema reference for UE-V](#) for details about the XML structure of the UE-V settings location template and for guidance about editing these files.

### Deploy the Custom Settings Location templates

After you create a settings location template with the UE-V template generator, you should test it to ensure that the application settings are synchronized correctly. You can then safely deploy the settings location template to user devices in the enterprise.

You can deploy settings location templates using of these methods:

- An electronic software distribution (ESD) system such as System Center Configuration Manager
- Group Policy preferences
- A UE-V settings template catalog

Templates that are deployed by using an ESD system or Group Policy objects must be registered using UE-V Windows Management Instrumentation (WMI) or Windows PowerShell. Templates that are stored in the settings template catalog location are automatically registered by the UE-V service.

### To deploy UE-V settings location templates with a settings template catalog path

1. Browse to the network share folder that you defined as the settings template catalog.
2. Add, remove, or update settings location templates in the settings template catalog to reflect the UE-V service template configuration that you want for UE-V computers.

**Note** Templates on computers are updated daily. The update is based on changes to the settings template catalog.

3. To manually update templates on a computer that runs the UE-V service, open an elevated command prompt, and browse to **Program Files\Microsoft User Experience Virtualization \ Agent \ <x86 or x64 >**, and then run **ApplySettingstemplateCatalog.exe**.

**Note** This program runs automatically during computer startup and daily at 3:30 A. M. to gather any new templates that were recently added to the catalog.

## Have a suggestion for UE-V?

Add or vote on suggestions on the [User Experience Virtualization feedback site](#).

For UE-V issues, use the [UE-V TechNet Forum](#).

## Related topics

- [Prepare a UE-V Deployment](#)
- [Deploy Required UE-V Features](#)

# Administering UE-V

6/20/2017 • 1 min to read • [Edit Online](#)

## Applies to

- Windows 10, version 1607

After you finish deploying User Experience Virtualization (UE-V), you'll perform ongoing administrative tasks, such as managing the configuration of the UE-V service and recovering lost settings. These tasks are explained in the following sections.

## Managing UE-V configurations

In the course of the UE-V lifecycle, you'll manage the configuration of the UE-V service and also manage storage locations for resources such as settings package files.

[Manage Configurations for UE-V](#)

## Working with custom UE-V templates and the UE-V template generator

This topic explains how to use the UE-V template generator and manage custom settings location templates.

[Working with Custom UE-V Templates and the UE-V Template Generator](#)

## Back up and restore application and Windows settings that are synchronized with UE-V

Windows Management Instrumentation (WMI) and Windows PowerShell features of UE-V allow you to restore settings packages. By using WMI and Windows PowerShell commands, you can restore application and Windows settings to their original state and restore additional settings when a user adopts a new device.

[Manage Administrative Backup and Restore in UE-V](#)

## Changing the frequency of UE-V scheduled tasks

You can configure the scheduled tasks that manage when UE-V checks for new or updated settings or for updated custom settings location templates in the settings template catalog.

[Changing the Frequency of UE-V Scheduled Tasks](#)

## Migrating UE-V settings packages

You can relocate the user settings packages either when they migrate to a new server or for backup purposes.

[Migrating UE-V Settings Packages](#)

## Using UE-V with Application Virtualization applications

You can use UE-V with Microsoft Application Virtualization (App-V) to share settings between virtual applications and installed applications across multiple computers.

## Other resources for this feature

- [User Experience Virtualization for Windows overview](#)
- [Get Started with UE-V](#)
- [Prepare a UE-V Deployment](#)
- [Troubleshooting UE-V](#)
- [Technical Reference for UE-V](#)

## Have a suggestion for UE-V?

Add or vote on suggestions on the [User Experience Virtualization feedback site](#).

For UE-V issues, use the [UE-V TechNet Forum](#).

# Manage Configurations for UE-V

6/20/2017 • 1 min to read • [Edit Online](#)

## Applies to

- Windows 10, version 1607

In the course of the User Experience Virtualization (UE-V) lifecycle, you have to manage the configuration of the UE-V service and also manage storage locations for resources such as settings package files. The following topics provide guidance for managing these UE-V resources.

## Configuring UE-V by using Group Policy Objects

You can use Group Policy Objects to modify the settings that define how UE-V synchronizes settings on computers.

### [Configuring UE-V with Group Policy Objects](#)

## Configuring UE-V with System Center Configuration Manager

You can use System Center Configuration Manager to manage the UE-V service by using the UE-V Configuration Pack.

### [Configuring UE-V with System Center Configuration Manager](#)

## Administering UE-V with PowerShell and WMI

UE-V provides Windows PowerShell cmdlets, which can help administrators perform various UE-V tasks.

### [Administering UE-V with Windows PowerShell and WMI](#)

## Examples of configuration settings for UE-V

Here are some examples of UE-V configuration settings:

- **Settings Storage Path:** Specifies the location of the file share that stores the UE-V settings.
- **Settings Template Catalog Path:** Specifies the Universal Naming Convention (UNC) path that defines the location that was checked for new settings location templates.
- **Register Microsoft Templates:** Specifies whether the default Microsoft templates should be registered during installation.
- **Synchronization Method:** Specifies whether UE-V uses the sync provider or "none". The "SyncProvider" supports computers that are disconnected from the network. "None" applies when the computer is always connected to the network. For more information about the Sync Method, see [Sync Methods for UE-V](#).
- **Synchronization Timeout:** Specifies the number of milliseconds that the computer waits before time-out when it retrieves the user settings from the settings storage location.
- **Synchronization Enable:** Specifies whether the UE-V settings synchronization is enabled or disabled.
- **Maximum Package Size:** Specifies a settings package file threshold size in bytes at which the UE-V service reports a warning.
- **Don't Sync Windows App Settings:** Specifies that UE-V should not synchronize Windows apps.

- **Enable/Disable First Use Notification:** Specifies whether UE-V displays a dialog box the first time that the UE-V service runs on a user's computer.

## Have a suggestion for UE-V?

Add or vote on suggestions on the [User Experience Virtualization feedback site](#).

For UE-V issues, use the [UE-V TechNet Forum](#).

## Related topics

[Administering UE-V](#)

[Deploy Required UE-V Features](#)

[Use UE-V with custom applications](#)

# Configuring UE-V with Group Policy Objects

6/20/2017 • 5 min to read • [Edit Online](#)

## Applies to

- Windows 10, version 1607

Some User Experience Virtualization (UE-V) Group Policy settings can be defined for computers, and other Group Policy settings can be defined for users. The Group Policy administrative templates for these settings are included in Windows 10, version 1607.

The following policy settings can be configured for UE-V.

## Group Policy settings

GROUP POLICY SETTING NAME	TARGET	GROUP POLICY SETTING DESCRIPTION	CONFIGURATION OPTIONS
Do not use the sync provider	Computers and Users	By using this Group Policy setting, you can configure whether UE-V uses the sync provider feature. This policy setting also lets you enable notification to appear when the import of user settings is delayed.	Enable this setting to configure the UE-V service not to use the sync provider.
First Use Notification	Computers Only	This Group Policy setting enables a notification in the notification area that appears when the UE-V service runs for the first time.	The default is enabled.
Roam Windows settings	Computers and Users	This Group Policy setting configures the synchronization of Windows settings.	Select which Windows settings synchronize between computers. By default, Windows themes, desktop settings, and Ease of Access settings synchronize settings between computers of the same operating system version.

GROUP POLICY SETTING NAME	TARGET	GROUP POLICY SETTING DESCRIPTION	CONFIGURATION OPTIONS
Settings package size warning threshold	Computers and Users	This Group Policy setting lets you configure the UE-V service to report when a settings package file size reaches a defined threshold.	Specify the preferred threshold for settings package sizes in kilobytes (KB). By default, the UE-V service does not have a package file size threshold.
Settings storage path	Computers and Users	This Group Policy setting configures where the user settings are to be stored.	Enter a Universal Naming Convention (UNC) path and variables such as \\Server\\SettingsShare%username%.
Settings template catalog path	Computers Only	This Group Policy setting configures where custom settings location templates are stored. This policy setting also configures whether the catalog is to be used to replace the default Microsoft templates that are installed with the UE-V service.	Enter a Universal Naming Convention (UNC) path such as \\Server\\TemplateShare or a folder location on the computer. Select the check box to replace the default Microsoft templates.
Sync settings over metered connections	Computers and Users	This Group Policy setting defines whether UE-V synchronizes settings over metered connections.	By default, the UE-V service does not synchronize settings over a metered connection.
Sync settings over metered connections even when roaming	Computers and Users	This Group Policy setting defines whether UE-V synchronizes settings over metered connections outside of the home provider network, for example, when the data connection is in roaming mode.	By default, UE-V does not synchronize settings over a metered connection when it is in roaming mode.

GROUP POLICY SETTING NAME	TARGET	GROUP POLICY SETTING DESCRIPTION	CONFIGURATION OPTIONS
Synchronization timeout	Computers and Users	This Group Policy setting configures the number of milliseconds that the computer waits before a time-out when it retrieves user settings from the remote settings location. If the remote storage location is unavailable, and the user does not use the sync provider, the application start is delayed by this many milliseconds.	Specify the preferred synchronization time-out in milliseconds. The default value is 2000 milliseconds.
Tray Icon	Computers Only	This Group Policy setting enables the User Experience Virtualization (UE-V) tray icon.	This setting only has an effect for UE-V 2.x and earlier. It has no effect for UE-V in Windows 10, version 1607.
Use User Experience Virtualization (UE-V)	Computers and Users	This Group Policy setting lets you enable or disable User Experience Virtualization (UE-V).	This setting only has an effect for UE-V 2.x and earlier. For UE-V in Windows 10, version 1607, use the <b>Enable UE-V</b> setting.
Enable UE-V	Computers and Users	This policy setting allows you to enable or disable User Experience Virtualization (UE-V) feature. Reboot is needed for enable to take effect.	This setting only has an effect for UE-V in Windows 10, version 1607. For UE-V 2.x and earlier, choose the <b>Use User Experience Virtualization (UE-V)</b> setting.

### Note

In addition, Group Policy settings are available for many desktop applications and Windows apps. You can use these settings to enable or disable settings synchronization for specific applications.

### Windows App Group Policy settings

GROUP POLICY SETTING NAME	TARGET	GROUP POLICY SETTING DESCRIPTION	CONFIGURATION OPTIONS
Do not synchronize Windows Apps	Computers and Users	This Group Policy setting defines whether the UE-V service synchronizes settings for Windows apps.	The default is to synchronize Windows apps.

GROUP POLICY SETTING NAME	TARGET	GROUP POLICY SETTING DESCRIPTION	CONFIGURATION OPTIONS
Windows App List	Computer and User	This setting lists the family package names of the Windows apps and states expressly whether UE-V synchronizes that app's settings.	You can use this setting to specify that settings of an app are never synchronized by UE-V, even if the settings of all other Windows apps are synchronized.
Sync Unlisted Windows Apps	Computer and User	This Group Policy setting defines the default settings sync behavior of the UE-V service for Windows apps that are not explicitly listed in the Windows app list.	By default, the UE-V service only synchronizes settings of those Windows apps that are included in the Windows app list.

For more information about synchronizing Windows apps, see [Windows App List](#).

### To configure computer-targeted Group Policy settings

1. Use the Group Policy Management Console (GPMC) or the Advanced Group Policy Management (AGPM) on the computer that acts as a domain controller to manage Group Policy settings for UE-V computers. Navigate to **Computer configuration**, select **Policies**, select **Administrative Templates**, click **Windows Components**, and then select **Microsoft User Experience Virtualization**.
2. Select the Group Policy setting to be edited.

### To configure user-targeted Group Policy settings

1. Use the Group Policy Management Console (GPMC) or the Advanced Group Policy Management (AGPM) tool in Microsoft Desktop Optimization Pack (MDOP) on the domain controller computer to manage Group Policy settings for UE-V. Navigate to **User configuration**, select **Policies**, select **Administrative Templates**, click **Windows Components**, and then select **Microsoft User Experience Virtualization**.
2. Select the edited Group Policy setting.

The UE-V service uses the following order of precedence to determine synchronization.

### Order of precedence for UE-V settings

1. User-targeted settings that are managed by Group Policy settings - These configuration settings are stored in the registry key by Group Policy under  
`HKEY_CURRENT_USER\Software\Policies\Microsoft\Uev\Agent\Configuration`.
2. Computer-targeted settings that are managed by Group Policy settings - These configuration settings are stored in the registry key by Group Policy under  
`HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Uev\Agent\Configuration`.
3. Configuration settings that are defined by the current user by using Windows PowerShell or Windows management Instrumentation (WMI) - These configuration settings are stored by the UE-V service under this registry location: `HKEY_CURRENT_USER\Software\Microsoft\Uev\Agent\Configuration`.
4. Configuration settings that are defined for the computer by using Windows PowerShell or WMI. These configuration settings are stored by the UE-V service under this registry location:

`HKEY_LOCAL_MACHINE\Software\Microsoft\Uev\Agent\Configuration`.

## Have a suggestion for UE-V?

Add or vote on suggestions on the [User Experience Virtualization feedback site](#).

For UE-V issues, use the [UE-V TechNet Forum](#).

## Related topics

[Administering UE-V](#)

[Manage Configurations for UE-V](#)

# Configuring UE-V with System Center Configuration Manager

6/20/2017 • 6 min to read • [Edit Online](#)

## Applies to

- Windows 10, version 1607

After you deploy User Experience Virtualization (UE-V) and its required features, you can start to configure it to meet your organization's need. The UE-V Configuration Pack provides a way for administrators to use the Compliance Settings feature of System Center Configuration Manager (2012 SP1 or later) to apply consistent configurations across sites where UE-V and Configuration Manager are installed.

## UE-V Configuration Pack supported features

The UE-V Configuration Pack includes tools to:

- Create or update UE-V settings location template distribution baselines
  - Define UE-V templates to be registered or unregistered
  - Update UE-V template configuration items and baselines as templates are added or updated
  - Distribute and register UE-V templates using standard Configuration Item remediation
- Create or update a UE-V Agent policy configuration item to set or clear these settings

Max package size	Enable/disable Windows app sync	Wait for sync on application start
Setting import delay	Sync unlisted Windows apps	Wait for sync on logon
Settings import notification	IT contact URL	Wait for sync timeout
Settings storage path	IT contact descriptive text	Settings template catalog path
Sync enablement	Tray icon enabled	Start/Stop UE-V agent service
Sync method	First use notification	Define which Windows apps will roam settings
Sync timeout		

- Verify compliance by confirming that UE-V is running.

## Generate a UE-V service policy configuration item

All UE-V service policy and configuration is distributed through a single configuration item that is generated using the UevAgentPolicyGenerator.exe tool. This tool reads the desired configuration from an XML configuration file and creates a CI containing the discovery and remediation settings needed to bring the machine into compliance.

The UE-V service policy configuration item CAB file is created using the UevTemplateBaselineGenerator.exe command line tool, which has these parameters:

- Site <site code>
- PolicyName <name> Optional: Defaults to "UE-V Agent Policy" if not present
- PolicyDescription <description> Optional: A description is provided if not present
- CabFilePath <full path to configuration item .CAB file>
- ConfigurationFile <full path to agent configuration XML file>

#### Note

It might be necessary to change the PowerShell execution policy to allow these scripts to run in your environment. Perform these steps in the Configuration Manager console:

1. Select **Administration > Client Settings > Properties**
2. In the **User Agent** tab, set the **PowerShell Execution Policy** to **Bypass**

#### Create the first UE-V policy configuration item

1. Copy the default settings configuration file from the UE-V Config Pack installation directory to a location visible to your ConfigMgr Admin Console:

```
C:\Program Files (x86)\Windows Kits\10\Microsoft User Experience
Virtualization\Management\AgentConfiguration.xml
```

The default configuration file contains five sections:

#### Computer Policy

All UE-V machine level settings. The DesiredState attribute can be

- **Set** to have the value assigned in the registry
- **Clear** to remove the setting
- **Unmanaged** to have the configuration item left at its current state

Do not remove lines from this section. Instead, set the DesiredState to 'Unmanaged' if you do not want Configuration Manager to alter current or default values.

#### CurrentComputerUserPolicy

All UE-V user level settings. These entries override the machine settings for a user. The DesiredState attribute can be

- **Set** to have the value assigned in the registry
- **Clear** to remove the setting
- **Unmanaged** to have the configuration item left at its current state

Do not remove lines from this section. Instead, set the DesiredState to 'Unmanaged' if you do not want Configuration Manager to alter current or default values.

#### Services

Entries in this section control service operation. The default configuration file contains a single entry for the UevAgentService. The DesiredState attribute can be set to **Running** or **Stopped**.

### **Windows8AppsComputerPolicy**

All machine level Windows app synchronization settings. Each PackageFamilyName listed in this section can be assigned a DesiredState of

- **Enabled** to have settings roam
- **Disabled** to prevent settings from roaming
- **Cleared** to have the entry removed from UE-V control

Additional lines can be added to this section based on the list of installed Windows apps that can be viewed using the PowerShell cmdlet GetAppxPackage.

### **Windows8AppsCurrentComputerUserPolicy**

Identical to the Windows8AppsComputerPolicy with settings that override machine settings for an individual user.

2. Edit the configuration file by changing the desired state and value fields.
3. Run this command on a machine running the ConfigMgr Admin Console:

```
C:\Program Files (x86)\Microsoft User Experience Virtualization\ConfigPack\UevAgentPolicyGenerator.exe
-Site ABC -CabFilePath "C:\MyCabFiles\UevPolicyItem.cab" -ConfigurationFile "c:\AgentConfiguration.xml"
```

4. Import the CAB file using ConfigMgr console or PowerShell Import-CMConfigurationItem

### **Update a UE-V Policy Configuration Item**

1. Edit the configuration file by changing the desired state and value fields.
2. Run the command from Step 3 in [Create the First UE-V Policy Configuration Item](#). If you changed the name with the PolicyName parameter, make sure you enter the same name.
3. Reimport the CAB file. The version in ConfigMgr will be updated.

## **Generate a UE-V Template Baseline**

UE-V templates are distributed using a baseline containing multiple configuration items. Each configuration item contains the discovery and remediation scripts needed to install one UE-V template. The actual UE-V template is embedded within the remediation script for distribution using standard Configuration Item functionality.

The UE-V template baseline is created using the UevTemplateBaselineGenerator.exe command line tool, which has these parameters:

- Site <site code>
- BaselineName <name> (Optional: defaults to "UE-V Template Distribution Baseline" if not present)
- BaselineDescription <description> (Optional: a description is provided if not present)
- TemplateFolder <UE-V template folder>
- Register <comma separated template file list>
- Unregister <comma separated template list>
- CabFilePath <Full path to baseline CAB file to generate>

The result is a baseline CAB file that is ready for import into Configuration Manager. If at a future date, you update or add a template, you can rerun the command using the same baseline name. Importing the CAB results in CI version updates on the changed templates.

### Create the First UE-V Template Baseline

1. Create a "master" set of UE-V templates in a stable folder location visible to the machine running your ConfigMgr Admin Console. As templates are added or updated, this folder is where they are pulled for distribution. The initial list of templates can be copied from a machine with UE-V installed. The default template location is C:\Program Files\Microsoft User Experience Virtualization\Templates.
2. Create a text.bat file where you can add the template generator command. This is optional, but will make regeneration simpler if you save the command parameters.
3. Add the command and parameters to the .bat file that will generate the baseline. The following example creates a baseline that distributes Notepad and Calculator:

```
C:\Program Files (x86)\Microsoft User Experience
Virtualization\ConfigPack\UevTemplateBaselineGenerator.exe -Site "ABC" -TemplateFolder
"C:\ProductionUevTemplates" -Register "MicrosoftNotepad.xml, MicrosoftCalculator.xml" -CabFilePath
"C:\MyCabFiles\UevTemplateBaseline.cab"
```

4. Run the .bat file to create UevTemplateBaseline.cab ready for import into Configuration Manager.

### Update a UE-V Template Baseline

The template generator uses the template version to determine if a template should be updated. If you make a template change and update the version, the baseline generator compares the template in your master folder with the template contained in the CI on the ConfigMgr server. If a difference is found, the generated baseline and modified CI versions are updated.

To distribute a new Notepad template, you would perform these steps:

1. Update the template and template version located in the <Version> element of the template.
2. Copy the template to your master template directory.
3. Run the command in the .bat file that you created in Step 3 in [Create the First UE-V Template Baseline](#).
4. Import the generated CAB file into ConfigMgr using the console or PowerShell Import-CMBaseline.

## Get the UE-V Configuration Pack

You can download the [System Center 2012 Configuration Pack for Microsoft User Experience Virtualization 2.0](#) from the Microsoft Download Center.

## Have a suggestion for UE-V?

Add or vote on suggestions on the [User Experience Virtualization feedback site](#).

For UE-V issues, use the [UE-V TechNet Forum](#).

## Related topics

[Manage Configurations for UE-V](#)

# Administering UE-V with Windows PowerShell and WMI

6/20/2017 • 1 min to read • [Edit Online](#)

## Applies to

- Windows 10, version 1607

User Experience Virtualization (UE-V) provides Windows PowerShell cmdlets to help administrators perform various UE-V tasks. The following sections provide more information about using Windows PowerShell in UE-V.

**Note** Administering UE-V with Windows PowerShell requires PowerShell 3.0 or higher. For a complete list of UE-V cmdlets, see [User Experience Virtualization in Windows PowerShell](#).

## Managing the UE-V service and packages by using Windows PowerShell and WMI

You can use Windows PowerShell and Windows Management Instrumentation (WMI) to manage UE-V service configuration and synchronization behavior. The following topic describes how to manage configuration and synchronization.

[Managing the UE-V Service and Packages with Windows PowerShell and WMI](#)

## Managing UE-V settings location templates by using Windows PowerShell and WMI

After you create and deploy UE-V settings location templates, you can manage those templates by using Windows PowerShell or WMI. The following topic describes how to manage the settings location templates by using Windows PowerShell and WMI.

[Managing UE-V Settings Location Templates Using Windows PowerShell and WMI](#)

## Have a suggestion for UE-V?

Add or vote on suggestions on the [User Experience Virtualization feedback site](#).

For UE-V issues, use the [UE-V TechNet Forum](#).

## Related topics

- [Administering UE-V](#)
- [User Experience Virtualization in Windows PowerShell](#)

# Managing the UE-V service and packages with Windows PowerShell and WMI

6/20/2017 • 7 min to read • [Edit Online](#)

## Applies to

- Windows 10, version 1607

You can use Windows Management Instrumentation (WMI) and Windows PowerShell to manage User Experience Virtualization (UE-V) service configuration and synchronization behavior.

**Note** For a complete list of UE-V cmdlets, see [User Experience Virtualization in Windows PowerShell](#).

## To configure the UE-V service with Windows PowerShell

- Open a Windows PowerShell window. To manage computer settings that affect all users of the computer by using the *Computer* parameter, open the window with an account that has administrator rights.
- Use the following Windows PowerShell commands to configure the service.

WINDOWS POWERSHELL COMMAND	DESCRIPTION
<code>Enable-UEV</code>	Turns on the UE-V service. Requires reboot.
<code>Disable-UEV</code>	Turns off the UE-V service. Requires reboot.
<code>Get-UevStatus</code>	Displays whether UE-V service is enabled or disabled, using a Boolean value.
<code>Get-UevConfiguration</code>	Gets the effective UE-V service settings. User-specific settings have precedence over the computer settings.
<code>Get-UevConfiguration -CurrentComputerUser</code>	Gets the UE-V service settings values for the current user only.
<code>Get-UevConfiguration -Computer</code>	Gets the UE-V service configuration settings values for all users on the computer.
<code>Get-UevConfiguration -Details</code>	Gets the details for each configuration setting. Displays where the setting is configured or if it uses the default value. Is displayed if the current setting is valid.
<code>Set-UevConfiguration -Computer -EnableDontSyncWindows8AppSettings</code>	Configures the UE-V service to not synchronize any Windows apps for all users on the computer.

WINDOWS POWERSHELL COMMAND	DESCRIPTION
<code>Set-UevConfiguration -CurrentComputerUser -EnableDontSyncWindows8AppSettings</code>	Configures the UE-V service to not synchronize any Windows apps for the current computer user.
<code>Set-UevConfiguration -Computer -EnableFirstUseNotification</code>	Configures the UE-V service to display notification the first time the service runs for all users on the computer.
<code>Set-UevConfiguration -Computer -DisableFirstUseNotification</code>	Configures the UE-V service to not display notification the first time that the service runs for all users on the computer.
<code>Set-UevConfiguration -Computer -EnableSettingsImportNotify</code>	<p>Configures the UE-V service to notify all users on the computer when settings synchronization is delayed.</p> <p>Use the <i>DisableSettingsImportNotify</i> parameter to disable notification.</p>
<code>Set-UevConfiguration -CurrentComputerUser -EnableSettingsImportNotify</code>	<p>Configures the UE-V service to notify the current user when settings synchronization is delayed.</p> <p>Use the <i>DisableSettingsImportNotify</i> parameter to disable notification.</p>
<code>Set-UevConfiguration -Computer -EnableSyncUnlistedWindows8Apps</code>	<p>Configures the UE-V service to synchronize all Windows apps that are not explicitly disabled by the Windows app list for all users of the computer. For more information, see "Get-UevAppxPackage" in <a href="#">Managing UE-V Settings Location Templates Using Windows PowerShell and WMI</a>.</p> <p>Use the <i>DisableSyncUnlistedWindows8Apps</i> parameter to configure the UE-V service to synchronize only Windows apps that are explicitly enabled by the Windows App List.</p>
<code>Set-UevConfiguration -CurrentComputerUser -EnableSyncUnlistedWindows8Apps</code>	<p>Configures the UE-V service to synchronize all Windows apps that are not explicitly disabled by the Windows app list for the current user on the computer. For more information, see "Get-UevAppxPackage" in <a href="#">Managing UE-V Settings Location Templates Using Windows PowerShell and WMI</a>.</p> <p>Use the <i>DisableSyncUnlistedWindows8Apps</i> parameter to configure the UE-V service to synchronize only Windows apps that are explicitly enabled by the Windows App List.</p>
<code>Set-UevConfiguration -Computer -DisableSync</code>	<p>Disables UE-V for all the users on the computer.</p> <p>Use the <i>EnableSync</i> parameter to enable or re-enable.</p>

WINDOWS POWERSHELL COMMAND	DESCRIPTION
<pre data-bbox="203 188 695 233">Set-UevConfiguration -CurrentComputerUser - DisableSync</pre>	<p>Disables UE-V for the current user on the computer.</p> <p>Use the <i>EnableSync</i> parameter to enable or re-enable.</p>
<pre data-bbox="203 345 726 368">Set-UevConfiguration -Computer -EnableTrayIcon</pre>	<p>Enables the UE-V icon in the notification area for all users of the computer.</p> <p>Use the <i>DisableTrayIcon</i> parameter to disable the icon.</p>
<pre data-bbox="203 570 631 615">Set-UevConfiguration -Computer - MaxPackageSizeInBytes &lt;size in bytes&gt;</pre>	<p>Configures the UE-V service to report when a settings package file size reaches the defined threshold for all users on the computer. Sets the threshold package size in bytes.</p>
<pre data-bbox="203 772 695 795">Set-UevConfiguration -CurrentComputerUser - MaxPackageSizeInBytes &lt;size in bytes&gt;</pre>	<p>Configures the UE-V service to report when a settings package file size reaches the defined threshold. Sets the package size warning threshold for the current user.</p>
<pre data-bbox="203 974 588 1019">Set-UevConfiguration -Computer - SettingsImportNotifyDelayInSeconds</pre>	<p>Specifies the time in seconds before the user is notified for all users of the computer</p>
<pre data-bbox="203 1109 695 1131">Set-UevConfiguration -CurrentComputerUser - SettingsImportNotifyDelayInSeconds</pre>	<p>Specifies the time in seconds before notification for the current user is sent.</p>
<pre data-bbox="203 1244 568 1289">Set-UevConfiguration -Computer - SettingsStoragePath &lt;path to _settings_storage_location&gt;</pre>	<p>Defines a per-computer settings storage location for all users of the computer.</p>
<pre data-bbox="203 1379 695 1423">Set-UevConfiguration -CurrentComputerUser - SettingsStoragePath &lt;path to _settings_storage_location&gt;</pre>	<p>Defines a per-user settings storage location.</p>
<pre data-bbox="203 1513 726 1558">Set-UevConfiguration -Computer - SettingsTemplateCatalogPath &lt;path to catalog&gt;</pre>	<p>Sets the settings template catalog path for all users of the computer.</p>
<pre data-bbox="203 1648 695 1693">Set-UevConfiguration -Computer -SyncMethod &lt;sync method&gt;</pre>	<p>Sets the synchronization method for all users of the computer: SyncProvider or None.</p>
<pre data-bbox="203 1783 695 1828">Set-UevConfiguration -CurrentComputerUser - SyncMethod &lt;sync method&gt;</pre>	<p>Sets the synchronization method for the current user: SyncProvider or None.</p>
<pre data-bbox="203 1918 631 1985">Set-UevConfiguration -Computer - SyncTimeoutInMilliseconds &lt;timeout in milliseconds&gt;</pre>	<p>Sets the synchronization time-out in milliseconds for all users of the computer</p>

WINDOWS POWERSHELL COMMAND	DESCRIPTION
<code>Set-UevConfiguration -CurrentComputerUser -SyncTimeoutInMilliseconds &lt;timeout in milliseconds&gt;</code>	Set the synchronization time-out for the current user.
<code>Clear-UevConfiguration -Computer -&lt;setting name&gt;</code>	Clears the specified setting for all users on the computer.
<code>Clear-UevConfiguration -CurrentComputerUser -&lt;setting name&gt;</code>	Clears the specified setting for the current user only.
<code>Export-UevConfiguration &lt;settings migration file&gt;</code>	<p>Exports the UE-V computer configuration to a settings migration file. The file name extension must be .uev.</p> <p>The <code>Export</code> cmdlet exports all UE-V service settings that are configurable with the <i>Computer</i> parameter.</p>
<code>Import-UevConfiguration &lt;settings migration file&gt;</code>	Imports the UE-V computer configuration from a settings migration file. The file name extension must be .uev.

## To export UE-V package settings and repair UE-V templates with Windows PowerShell

1. Open a Windows PowerShell window as an administrator.
2. Use the following Windows PowerShell commands to configure the service.

Windows PowerShell command	Description
<code>Export-UevPackage MicrosoftNotepad.pkgx</code>	Extracts the settings from a Microsoft Notepad package file and converts them into a human-readable format in XML.
<code>Repair-UevTemplateIndex</code>	Repairs the index of the UE-V settings location templates.

## To configure the UE-V service with WMI

1. User Experience Virtualization provides the following set of WMI commands. Administrators can use this interface to configure the UE-V service at the command line and automate typical configuration tasks.

Use an account with administrator rights to open a Windows PowerShell window.

2. Use the following WMI commands to configure the service.

WINDOWS POWERSHELL COMMAND	DESCRIPTION
Get-WmiObject -Namespace root\Microsoft\UEV Configuration	Displays the active UE-V service settings. User-specific settings have precedence over the computer settings.
Get-WmiObject -Namespace root\Microsoft\UEV UserConfiguration	Displays the UE-V service configuration that is defined for a user.
Get-WmiObject -Namespace root\Microsoft\UEV ComputerConfiguration	Displays the UE-V service configuration that is defined for a computer.
Get-WmiObject -Namespace root\Microsoft\Uev ConfigurationItem	Displays the details for each configuration item.
<pre>\$config = Get-WmiObject -Namespace root\Microsoft\UEV ComputerConfiguration</pre> <pre>\$config.SettingsStoragePath = &lt;path_to_settings_storage_location&gt;</pre> <pre>\$config.Put()</pre>	Defines a per-computer settings storage location.
<pre>\$config = Get-WmiObject -Namespace root\Microsoft\UEV UserConfiguration</pre> <pre>\$config.SettingsStoragePath = &lt;path_to_settings_storage_location&gt;</pre> <pre>\$config.Put()</pre>	Defines a per-user settings storage location.
<pre>\$config = Get-WmiObject -Namespace root\Microsoft\UEV ComputerConfiguration</pre> <pre>\$config.SyncTimeoutInMilliseconds = &lt;timeout_in_milliseconds&gt;</pre> <pre>\$config.Put()</pre>	Sets the synchronization time-out in milliseconds for all users of the computer.
<pre>\$config = Get-WmiObject -Namespace root\Microsoft\UEV ComputerConfiguration</pre> <pre>\$config.MaxPackageSizeInBytes = &lt;size_in_bytes&gt;</pre> <pre>\$config.Put()</pre>	Configures the UE-V service to report when a settings package file size reaches a defined threshold. Set the threshold package file size in bytes for all users of the computer.
<pre>\$config = Get-WmiObject -Namespace root\Microsoft\UEV ComputerConfiguration</pre> <pre>\$config.SyncMethod = &lt;sync_method&gt;</pre> <pre>\$config.Put()</pre>	Sets the synchronization method for all users of the computer: SyncProvider or None.

WINDOWS POWERSHELL COMMAND	DESCRIPTION
<pre>\$config = Get-WmiObject -Namespace root\Microsoft\UEV ComputerConfiguration \$config.&lt;setting name&gt; = \$true \$config.Put()</pre>	To enable a specific per-computer setting, clear the setting, and use <code>\$null</code> as the setting value. Use UserConfiguration for per-user settings.
<pre>\$config = Get-WmiObject -Namespace root\Microsoft\UEV ComputerConfiguration \$config.&lt;setting name&gt; = \$false \$config.Put()</pre>	To disable a specific per-computer setting, clear the setting, and use <code>\$null</code> as the setting value. Use User Configuration for per-user settings.
<pre>\$config = Get-WmiObject -Namespace root\Microsoft\UEV ComputerConfiguration \$config.&lt;setting name&gt; = &lt;setting value&gt; \$config.Put()</pre>	Updates a specific per-computer setting. To clear the setting, use <code>\$null</code> as the setting value.
<pre>\$config = Get-WmiObject -Namespace root\Microsoft\UEV ComputerConfiguration \$config.&lt;setting name&gt; = &lt;setting value&gt; \$config.Put()</pre>	Updates a specific per-user setting for all users of the computer. To clear the setting, use <code>\$null</code> as the setting value.

When you are finished configuring the UE-V service with WMI and Windows PowerShell, the defined configuration is stored in the registry in the following locations.

```
\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\UEV\Agent\Configuration
\HKEY_CURRENT_USER\SOFTWARE\Microsoft\UEV\Agent\Configuration
```

## To export UE-V package settings and repair UE-V templates by using WMI

1. UE-V provides the following set of WMI commands. Administrators can use this interface to export a package or repair UE-V templates.
2. Use the following WMI commands.

WMI COMMAND	DESCRIPTION
<pre>Invoke-WmiMethod -Namespace root\Microsoft\UEV -Class UserSettings -Name ExportPackage -ArgumentList &lt;package name&gt;</pre>	Extracts the settings from a package file and converts them into a human-readable format in XML.
<pre>Invoke-WmiMethod -Namespace root\Microsoft\UEV -Class SettingsLocationTemplate -Name RebuildIndex</pre>	Repairs the index of the UE-V settings location templates. Must be run as administrator.

Have a suggestion for UE-V?

Add or vote on suggestions on the [User Experience Virtualization feedback site](#).

For UE-V issues, use the [UE-V TechNet Forum](#).

## Related topics

[Administering UE-V with Windows PowerShell and WMI](#)

[Administering UE-V](#)

[User Experience Virtualization in Windows PowerShell](#)

# Managing UE-V Settings Location Templates Using Windows PowerShell and WMI

6/20/2017 • 9 min to read • [Edit Online](#)

## Applies to

- Windows 10, version 1607

User Experience Virtualization (UE-V) uses XML settings location templates to define the settings that User Experience Virtualization captures and applies. UE-V includes a set of standard settings location templates. It also includes the UE-V template generator tool that enables you to create custom settings location templates. After you create and deploy settings location templates, you can manage those templates by using Windows PowerShell and the Windows Management Instrumentation (WMI).

**Note** For a complete list of UE-V cmdlets, see [User Experience Virtualization in Windows PowerShell](#).

## Manage UE-V settings location templates by using Windows PowerShell

The WMI and Windows PowerShell features of UE-V include the ability to enable, disable, register, update, and unregister settings location templates. By using these features, you can automate the process of registering, updating, or unregistering templates with the UE-V service. You can also manually register templates by using WMI and Windows PowerShell commands. By using these features in conjunction with an electronic software distribution solution, Group Policy, or another automated deployment method such as a script, you can further automate that process.

You must have administrator permissions to update, register, or unregister a settings location template. Administrator permissions are not required to enable, disable, or list templates.

### \*To manage settings location templates by using Windows PowerShell\*

- Use an account with administrator rights to open a Windows PowerShell command prompt.
- Use the following Windows PowerShell cmdlets to register and manage the UE-V settings location templates.

WINDOWS POWERSHELL COMMAND	DESCRIPTION
<code>Get-UevTemplate</code>	Lists all the settings location templates that are registered on the computer.
<code>Get-UevTemplate -Application &lt;string&gt;</code>	Lists all the settings location templates that are registered on the computer where the application name or template name contains <string>.
<code>Get-UevTemplate -TemplateID &lt;string&gt;</code>	Lists all the settings location templates that are registered on the computer where the template ID contains <string>.

WINDOWS POWERSHELL COMMAND	DESCRIPTION
<code>Get-UevTemplate [-ApplicationOrTemplateID] &lt;string&gt;</code>	Lists all the settings location templates that are registered on the computer where the application or template name, or template ID contains <string>.
<code>Get-UevTemplateProgram [-ID] &lt;template ID&gt;</code>	Gets the name of the program and version information, which depend on the template ID.
<code>Get-UevAppXPackage</code>	Gets the effective list of Windows apps.
<code>Get-UevAppXPackage -Computer</code>	Gets the list of Windows apps that are configured for the computer.
<code>Get-UevAppXPackage -CurrentComputerUser</code>	Gets the list of Windows apps that are configured for the current user.
<code>Register-UevTemplate [-Path] &lt;template file path&gt;[,&lt;template file path&gt;]</code>	Registers one or more settings location template with UE-V by using relative paths and/or wildcard characters in file paths. After a template is registered, UE-V synchronizes the settings that are defined in the template between computers that have the template registered.
<code>Register-UevTemplate -LiteralPath &lt;template file path&gt;[,&lt;template file path&gt;]</code>	Registers one or more settings location template with UE-V by using literal paths, where no characters can be interpreted as wildcard characters. After a template is registered, UE-V synchronizes the settings that are defined in the template between computers that have the template registered.
<code>Unregister-UevTemplate [-ID] &lt;template ID&gt;</code>	Unregisters a settings location template with UE-V. When a template is unregistered, UE-V no longer synchronizes the settings that are defined in the template between computers.
<code>Unregister-UevTemplate -All</code>	Unregisters all settings location templates with UE-V. When a template is unregistered, UE-V no longer synchronizes the settings that are defined in the template between computers.
<code>Update-UevTemplate [-Path] &lt;template file path&gt;[,&lt;template file path&gt;]</code>	Updates one or more settings location templates with a more recent version of the template. Use relative paths and/or wildcard characters in the file paths. The new template should be a newer version than the existing template.

WINDOWS POWERSHELL COMMAND	DESCRIPTION
<pre>Update-UevTemplate -LiteralPath &lt;template file path&gt;[,&lt;template file path&gt;]</pre>	<p>Updates one or more settings location templates with a more recent version of the template. Use full paths to template files, where no characters can be interpreted as wildcard characters. The new template should be a newer version than the existing template.</p>
<pre>Clear-UevAppXPackage -Computer [-PackageFamilyName] &lt;package family name&gt;[,&lt;package family name&gt;]</pre>	<p>Removes one or more Windows apps from the computer Windows app list.</p>
<pre>Clear-UevAppXPackage -CurrentComputerUser</pre>	<p>Removes Windows app from the current user Windows app list.</p>
<pre>Clear-UevAppXPackage -Computer -All</pre>	<p>Removes all Windows apps from the computer Windows app list.</p>
<pre>Clear-UevAppXPackage [-CurrentComputerUser] [-PackageFamilyName] &lt;package family name&gt;[,&lt;package family name&gt;]</pre>	<p>Removes one or more Windows apps from the current user Windows app list.</p>
<pre>Clear-UevAppXPackage [-CurrentComputerUser] -All</pre>	<p>Removes all Windows apps from the current user Windows app list.</p>
<pre>Disable-UevTemplate [-ID] &lt;template ID&gt;</pre>	<p>Disables a settings location template for the current user of the computer.</p>
<pre>Disable-UevAppXPackage -Computer [-PackageFamilyName] &lt;package family name&gt;[,&lt;package family name&gt;]</pre>	<p>Disables one or more Windows apps in the computer Windows app list.</p>
<pre>Disable-UevAppXPackage [-CurrentComputerUser] [-PackageFamilyName] &lt;package family name&gt;[,&lt;package family name&gt;]</pre>	<p>Disables one or more Windows apps in the current user Windows app list.</p>
<pre>Enable-UevTemplate [-ID] &lt;template ID&gt;</pre>	<p>Enables a settings location template for the current user of the computer.</p>
<pre>Enable-UevAppXPackage -Computer [-PackageFamilyName] &lt;package family name&gt;[,&lt;package family name&gt;]</pre>	<p>Enables one or more Windows apps in the computer Windows app list.</p>
<pre>Enable-UevAppXPackage [-CurrentComputerUser] [-PackageFamilyName] &lt;package family name&gt;[,&lt;package family name&gt;]</pre>	<p>Enables one or more Windows apps in the current user Windows app list.</p>
<pre>Test-UevTemplate [-Path] &lt;template file path&gt;[,&lt;template file path&gt;]</pre>	<p>Determines whether one or more settings location templates comply with its XML schema. Can use relative paths and wildcard characters.</p>

WINDOWS POWERSHELL COMMAND	DESCRIPTION
<pre>Test-UevTemplate -LiteralPath &lt;template file path&gt;[,&lt;template file path&gt;]</pre>	Determines whether one or more settings location templates comply with its XML schema. The path must be a full path to the template file, but does not include wildcard characters.

The UE-V Windows PowerShell features enable you to manage a group of settings templates that are deployed in your enterprise. Use the following procedure to manage a group of templates by using Windows PowerShell.

### To manage a group of settings location templates by using Windows PowerShell

1. Modify or update the desired settings location templates.
2. If you want to modify or update the settings location templates, deploy those settings location templates to a folder that is accessible to the local computer.
3. On the local computer, open a Windows PowerShell window with administrator rights.
4. Unregister all the previously registered versions of the templates by typing the following command.

```
Unregister-UevTemplate -All
```

This command unregisters all active templates on the computer.

5. Register the updated templates by typing the following command.

```
Register-UevTemplate <path to template folder>*.xml
```

This command registers all of the settings location templates that are located in the specified template folder.

### Windows app list

By listing a Windows app in the Windows app list, you specify whether that app is enabled or disabled for settings synchronization. Apps are identified in the list by their Package Family name and whether settings synchronization should be enabled or disabled for that app. When you use these settings along with the Unlisted Default Sync Behavior setting, you can control whether Windows apps are synchronized.

To display the Package Family Name of installed Windows apps, at a Windows PowerShell command prompt, enter:

```
Get-AppxPackage | Sort-Object PackageFamilyName | Format-Table PackageFamilyName
```

To display a list of Windows apps that can synchronize settings on a computer with their package family name, enabled status, and enabled source, at a Windows PowerShell command prompt, enter: `Get-UevAppxPackage`

### Definitions of Get-UevAppxPackage properties

#### PackageFamilyName

The name of the package that is installed for the current user.

#### Enabled

Defines whether the settings for the app are configured to synchronize.

#### EnabledSource

The location where the configuration that enables or disables the app is set. Possible values are: *NotSet*, *LocalMachine*, *LocalUser*, *PolicyMachine*, and *PolicyUser*.

#### **NotSet**

The policy is not configured to synchronize this app.

#### **LocalMachine**

The enabled state is set in the local computer section of the registry.

#### **LocalUser**

The enabled state is set in the current user section of the registry.

#### **PolicyMachine**

The enabled state is set in the policy section of the local computer section of the registry.

To get the user-configured list of Windows apps, at the Windows PowerShell command prompt, enter:

```
Get-UevAppxPackage -CurrentComputerUser
```

To get the computer-configured list of Windows apps, at the Windows PowerShell command prompt, enter:

```
Get-UevAppxPackage -Computer
```

For either parameter, CurrentComputerUser or Computer, the cmdlet returns a list of the Windows apps that are configured at the user or at the computer level.

#### **Definitions of properties**

##### **PackageFamilyName**

The name of the package that is installed for the current user.

##### **Enabled**

Defines whether the settings for the app are configured to synchronize for the specified switch, that is, **user** or **computer**.

##### **Installed**

True if the app, that is, the PackageFamilyName is installed for the current user.

#### **Manage UE-V settings location templates by using WMI**

User Experience Virtualization provides the following set of WMI commands. Administrators can use these interfaces to manage settings location templates from Windows PowerShell and automate template administrative tasks.

#### **To manage settings location templates by using WMI**

1. Use an account with administrator rights to open a Windows PowerShell window.
2. Use the following WMI commands to register and manage the UE-V settings location templates.

WINDOWS POWERSHELL COMMAND	DESCRIPTION
<pre>Get-WmiObject -Namespace root\Microsoft\UEV SettingsLocationTemplate   Select-Object TemplateId,TemplateName, TemplateVersion,Enabled   Format-Table -Autosize</pre>	Lists all the settings location templates that are registered for the computer.
<pre>Invoke-WmiMethod -Namespace root\Microsoft\UEV -Class SettingsLocationTemplate -Name GetProcessInfoByTemplateId &lt;template Id&gt;</pre>	Gets the name of the program and version information, which depends on the template name.

WINDOWS POWERSHELL COMMAND	DESCRIPTION
Get-WmiObject -Namespace root\Microsoft\UEV EffectiveWindows8App	Gets the effective list of Windows apps.
Get-WmiObject -Namespace root\Microsoft\UEV MachineConfiguredWindows8App	Gets the list of Windows apps that are configured for the computer.
Get-WmiObject -Namespace root\Microsoft\UEV UserConfiguredWindows8App	Gets the list of Windows apps that are configured for the current user.
Invoke-WmiMethod -Namespace root\Microsoft\UEV -Class SettingsLocationTemplate -Name Register -ArgumentList <template path >	Registers a settings location template with UE-V.
Invoke-WmiMethod -Namespace root\Microsoft\UEV -Class SettingsLocationTemplate -Name UnregisterByTemplateId -ArgumentList <template ID>	Unregisters a settings location template with UE-V. As soon as a template is unregistered, UE-V no longer synchronizes the settings that are defined in the template between computers.
Invoke-WmiMethod -Namespace root\Microsoft\UEV -Class SettingsLocationTemplate -Name Update -ArgumentList <template path>	Updates a settings location template with UE-V. The new template should be a newer version than the existing one.
Invoke-WmiMethod -Namespace root\Microsoft\UEV -Class MachineConfiguredWindows8App -Name RemoveApp -ArgumentList <package family name   package family name>	Removes one or more Windows apps from the computer Windows app list.
Invoke-WmiMethod -Namespace root\Microsoft\UEV -Class UserConfiguredWindows8App -Name RemoveApp -ArgumentList <package family name   package family name>	Removes one or more Windows apps from the current user Windows app list.
Invoke-WmiMethod -Namespace root\Microsoft\UEV -Class SettingsLocationTemplate -Name DisableByTemplateId -ArgumentList <template ID>	Disables one or more settings location templates with UE-V.
Invoke-WmiMethod -Namespace root\Microsoft\UEV -Class MachineConfiguredWindows8App -Name DisableApp -ArgumentList <package family name   package family name>	Disables one or more Windows apps in the computer Windows app list.
Invoke-WmiMethod -Namespace root\Microsoft\UEV -Class UserConfiguredWindows8App -Name DisableApp -ArgumentList <package family name   package family name>	Disables one or more Windows apps in the current user Windows app list.

WINDOWS POWERSHELL COMMAND	DESCRIPTION
<pre>Invoke-WmiMethod -Namespace root\Microsoft\UEV -Class SettingsLocationTemplate -Name EnableByTemplateId -ArgumentList &lt;template ID&gt;</pre>	Enables a settings location template with UE-V.
<pre>Invoke-WmiMethod -Namespace root\Microsoft\UEV -Class MachineConfiguredWindows8App -Name EnableApp -ArgumentList &lt;package family name   package family name&gt;</pre>	Enables Windows apps in the computer Windows app list.
<pre>Invoke-WmiMethod -Namespace root\Microsoft\UEV -Class UserConfiguredWindows8App -Name EnableApp - ArgumentList &lt;package family name   package family name&gt;</pre>	Enables Windows apps in the current user Windows app list.
<pre>Invoke-WmiMethod -Namespace root\Microsoft\UEV -Class SettingsLocationTemplate -Name Validate - ArgumentList &lt;template path&gt;</pre>	Determines whether a given settings location template complies with its XML schema.

### Note

Where a list of Package Family Names is called by the WMI command, the list must be in quotes and separated by a pipe symbol, for example, `"<package family name | package family name>"`.

## Have a suggestion for UE-V?

Add or vote on suggestions on the [User Experience Virtualization feedback site](#).

For UE-V issues, use the [UE-V TechNet Forum](#).

## Related topics

[Administering UE-V with Windows PowerShell and WMI](#)

[Administering UE-V](#)

[User Experience Virtualization in Windows PowerShell](#)

# Working with custom UE-V templates and the UE-V template generator

6/20/2017 • 8 min to read • [Edit Online](#)

## Applies to

- Windows 10, version 1607

User Experience Virtualization (UE-V) uses XML files called **settings location templates** to monitor and synchronize application settings and Windows settings between user devices. By default, some settings location templates are included in UE-V. However, if you want to synchronize settings for desktop applications other than those included in the default templates, you can create your own custom settings location templates with the UE-V template generator. You can also edit or validate custom settings location templates with the UE-V template generator.

Use the UE-V template generator to monitor, discover, and capture the locations where Win32 applications store settings. The template generator does not create settings location templates for the following types of applications:

- Virtualized applications
- Applications that are offered through Terminal Services
- Java applications
- Windows applications

## Standard and non-standard settings locations

The UE-V template generator helps you identify where applications search for settings files and registry settings that applications use to store settings information. The generator discovers settings only in locations that are accessible to a standard user. Settings that are stored in other locations are excluded.

Discovered settings are grouped into two categories: **Standard** and **Non-standard**. Standard settings are recommended for synchronization, and UE-V can readily capture and apply them. Non-standard settings can potentially synchronize settings but, because of the rules that UE-V uses, these settings might not consistently or dependably synchronize settings. These settings might depend on temporary files, result in unreliable synchronization, or might not be useful. These settings locations are presented in the UE-V template generator. You can choose to include or exclude them on a case-by-case basis.

The UE-V template generator opens the application as part of the discovery process. The generator can capture settings in the following locations:

- **Registry Settings** – Registry locations under **HKEY\_CURRENT\_USER**
- **Application Settings Files** – Files that are stored under \b>Users\b>[User name]\AppData\Roaming

The UE-V template generator excludes locations, which commonly store application software files, but do not synchronize well between user computers or environments. The UE-V template generator excludes these locations. Excluded locations are as follows:

- HKEY\_CURRENT\_USER registry keys and files to which the logged-on user cannot write values
- HKEY\_CURRENT\_USER registry keys and files that are associated with the core functionality of the Windows operating system
- All registry keys that are located in the HKEY\_LOCAL\_MACHINE hive, which requires administrator rights

and might require to set a User Account Control (UAC) agreement

- Files that are located in Program Files directories, which requires administrator rights and might require to set a UAC agreement
- Files that are located under Users \ [User name] \ AppData \ LocalLow
- Windows operating system files that are located in %Systemroot%, which requires administrator rights and might require to set a UAC agreement

If registry keys and files that are stored in these locations are required to synchronize application settings, you can manually add the excluded locations to the settings location template during the template creation process.

## Edit settings location templates with the UE-V template generator

Use the UE-V template generator to edit settings location templates. When the revised settings are added to the templates with the UE-V template generator, the version information within the template is automatically updated to ensure that any existing templates that are deployed in the enterprise are updated correctly.

### To edit a UE-V settings location template with the UE-V template generator

1. Open the **Start** menu and navigate to **Windows Kits > Microsoft User Experience Virtualization (UE-V) Template Generator** to open the template generator.
2. Click **Edit a settings location template**.
3. In the list of recently used templates, select the template to be edited. Alternatively, click **Browse** to search for the settings template file. Click **Next** to continue.
4. Review the **Properties**, **Registry** locations, and **Files** locations for the settings template. Edit as required.
  - On the **Properties** tab, you can view and edit the following properties:
    - **Application name** The application name that is written in the description of the program file properties.
    - **Program name** The name of the program that is taken from the program file properties. This name usually has the .exe file name extension.
    - **Product version** The product version number of the .exe file of the application. This property, together with the **File version**, helps determine which applications are targeted by the settings location template. This property accepts a major version number. If this property is empty, then the settings location template applies to all versions of the product.
    - **File version** The file version number of the .exe file of the application. This property, along with the **Product version**, helps determine which applications are targeted by the settings location template. This property accepts a major version number. If this property is empty, the settings location template applies to all versions of the program.
    - **Template author name** (optional) The name of the settings template author.
    - **Template author email** (optional) The email address of the settings location template author.
  - The **Registry** tab lists the **Key** and **Scope** of the registry locations that are included in the settings location template. You can edit the registry locations by using the **Tasks** drop-down menu. In the Tasks menu, you can add new keys, edit the name or scope of existing keys, delete keys, and browse the registry in which the keys are located. When you define the scope for the registry, you can use the **All Settings** scope to include all the registry settings under the specified key. Use **All Settings** and **Subkeys** to include all the registry settings under the specified key, subkeys, and subkey settings.

- The **Files** tab lists the file path and file mask of the file locations that are included in the settings location template. You can edit the file locations by using the **Tasks** drop-down menu. In the **Tasks** menu for file locations, you can add new files or folder locations, edit the scope of existing files or folders, delete files or folders, and open the selected location in Windows Explorer. To include all files in the specified folder, leave the file mask empty.

- Click **Save** to save the changes to the settings location template.
- Click **Close** to close the Settings Template Wizard. Exit the UE-V template generator application.

After you edit the settings location template for an application, you should test the template. Deploy the revised settings location template in a lab environment before you put it into production in the enterprise.

### How to manually edit a settings location template

- Create a local copy of the settings location template .xml file. UE-V settings location templates are .xml files that identify the locations where application store settings values.

#### Note

A settings location template is unique because of the template **ID**. If you copy the template and rename the .xml file, template registration fails because UE-V reads the template **ID** tag in the .xml file to determine the name, not the file name of the .xml file. UE-V also reads the **Version** number to know if anything has changed. If the version number is higher, UE-V updates the template.

- Open the settings location template file with an XML editor.
- Edit the settings location template file. All changes must conform to the UE-V schema file that is defined in [SettingsLocationTemplate.xsd](#). By default, a copy of the .xsd file is located in `\ProgramData\Microsoft\UEV\Templates`.
- Increment the **Version** number for the settings location template.
- Save the settings location template file, and then close the XML editor.
- Validate the modified settings location template file by using the UE-V template generator.
- You must register the edited UE-V settings location template before it can synchronize settings between client computers. To register a template, open Windows PowerShell, and then run the following cmdlet: `update-uevtemplate [templatefilename]`. You can then copy the file to the settings storage catalog. The UE-V Agent on users' computers should then update as scheduled in the scheduled task.

## Validate settings location templates with the UE-V template generator

It is possible to create or edit settings location templates in an XML editor without using the UE-V template generator. If you do, you can use the UE-V template generator to validate that the new or revised XML matches the schema that has been defined for the template.

### To validate a UE-V settings location template with the UE-V template generator

- Open the **Start** menu and navigate to **Windows Kits > Microsoft User Experience Virtualization (UE-V) Template Generator** to open the template generator.
- Click **Validate a settings location template**.
- In the list of recently used templates, select the template to be edited. Alternatively, you can **Browse** to the settings template file. Click **Next** to continue.
- Click **Validate** to continue.

5. Click **Close** to close the Settings Template Wizard. Exit the UE-V template generator application.

After you validate the settings location template for an application, you should test the template. Deploy the template in a lab environment before you put it into a production environment in enterprise.

## Share settings location templates with the Template Gallery

The [User Experience Virtualization Template Gallery](#) enables administrators to share their UE-V settings location templates. Upload your settings location templates to the gallery for other users to use, and download templates that other users have created.

Before you share a settings location template on the UE-V template gallery, ensure it does not contain any personal or company information. You can use any XML viewer to open and view the contents of a settings location template file. The following template values should be reviewed before you share a template with anyone outside your company.

- Template Author Name – Specify a general, non-identifying name for the template author name or exclude this data from the template.
- Template Author Email – Specify a general, non-identifying template author email or exclude this data from the template.

Before you deploy any settings location template that you have downloaded from the UE-V gallery, you should first test the template to ensure that the application settings synchronize settings correctly in a test environment.

## Have a suggestion for UE-V?

Add or vote on suggestions on the [User Experience Virtualization feedback site](#).

For UE-V issues, use the [UE-V TechNet Forum](#).

## Related topics

[Administering UE-V](#)

[Use UE-V with custom applications](#)

# Manage Administrative Backup and Restore in UE-V

6/20/2017 • 4 min to read • [Edit Online](#)

## Applies to

- Windows 10, version 1607

As an administrator of User Experience Virtualization (UE-V), you can restore application and Windows settings to their original state. You can also restore additional settings when a user adopts a new device.

## Restore Settings in UE-V when a User Adopts a New Device

To restore settings when a user adopts a new device, you can put a settings location template in **backup** or **roam (default)** profile using the Set-UevTemplateProfile PowerShell cmdlet. This lets computer settings sync to the new computer, in addition to user settings. Templates assigned to the backup profile are backed up for that device and configured on a per-device basis. To backup settings for a template, use the following cmdlet in Windows PowerShell:

```
Set-UevTemplateProfile -ID <TemplateID> -Profile <backup>
```

- <TemplateID> is the UE-V Template ID
- <backup> can either be Backup or Roaming

When replacing a user's device, UE-V automatically restores settings if the user's domain, username, and device name all match. All synchronized and any backup data is restored on the device automatically.

You can also use the Windows PowerShell cmdlet, Restore-UevBackup, to restore settings from a different device. To clone the settings packages for the new device, use the following cmdlet in Windows PowerShell:

```
Restore-UevBackup -Machine <MachineName>
```

where <MachineName> is the computer name of the device.

Templates such as the Office 2013 template that include many applications can either all be included in the roamed (default) or backed up profile. Individual apps in a template suite follow the group. Office 2013 in-box templates include both roaming and backup-only settings. Backup-only settings cannot be included in a roaming profile.

As part of the Backup/Restore feature, UE-V added **last known good (LKG)** to the options for rolling back to settings. In this release, you can roll back to either the original settings or LKG settings. The LKG settings let users roll back to an intermediate and stable point ahead of the pre-UE-V state of the settings.

## How to Backup/Restore Templates with UE-V

These are the key backup and restore components of UE-V:

- Template profiles
- Settings packages location within the Settings Storage Location template
- Backup trigger
- How settings are restored

## **Template Profiles**

A UE-V template profile is defined when the template is registered on the device or post registration through the PowerShell/WMI configuration utility. The profile types include:

- Roaming (default)
- Backup
- BackupOnly

All templates are included in the roaming profile when registered unless otherwise specified. These templates synchronize settings to all UE-V enabled devices with the corresponding template enabled.

Templates can be added to the Backup Profile with PowerShell or WMI using the Set-UevTemplateProfile cmdlet. Templates in the Backup Profile back up these settings to the Settings Storage Location in a special Device name directory. Specified settings are backed up to this location.

Templates designated BackupOnly include settings specific to that device that should not be synchronized unless explicitly restored. These settings are stored in the same device-specific settings package location on the settings storage location as the Backedup Settings. These templates have a special identifier embedded in the template that specifies they should be part of this profile.

### **Settings packages location within the Settings Storage Location template**

Roaming Profile settings are stored on the settings storage location. Templates assigned to the Backup or the BackupOnly profile store their settings to the Settings Storage Location in a special Device name directory. Each device with templates in these profiles has its own device name. UE-V does not clean up these directories.

### **Backup trigger**

Backup is triggered by the same events that trigger a UE-V synchronization.

### **How settings are restored**

Restoring a user's device restores the currently registered Template's settings from another device's backup folder and all synchronized settings to the current machine. Settings are restored in these two ways:

- **Automatic restore**

If the user's UE-V settings storage path, domain, and Computer name match the current user then all of the settings for that user are synchronized, with only the latest settings applied. If a user logs on to a new device for the first time and these criteria are met, the settings data is applied to that device.

#### **Note**

Accessibility and Windows Desktop settings require the user to re-logon to Windows to be applied.

- **Manual Restore**

If you want to assist users by restoring a device during a refresh, you can choose to use the Restore-UevBackup cmdlet. This command ensures that the user's current settings become the current state on the Settings Storage Location.

## **Restore Application and Windows Settings to Original State**

WMI and Windows PowerShell commands let you restore application and Windows settings to the settings values that were on the computer the first time that the application started after the UE-V service was enabled. This restoring action is performed on a per-application or Windows settings basis. The settings are restored the next time that the application runs, or the settings are restored when the user logs on to the operating system.

## To restore application settings and Windows settings with Windows PowerShell for UE-V

1. Open the Windows PowerShell window.
2. Enter the following Windows PowerShell cmdlet to restore the application settings and Windows settings.

WINDOWS POWERSHELL CMDLET	DESCRIPTION
<pre>Restore-UevUserSetting -&lt;TemplateID&gt;</pre>	Restores the user settings for an application or restores a group of Windows settings.

## To restore application settings and Windows settings with WMI

1. Open a Windows PowerShell window.
2. Enter the following WMI command to restore application settings and Windows settings.

WMI COMMAND	DESCRIPTION
<pre>Invoke-WmiMethod -Namespace root\Microsoft\UEV -Class UserSettings -Name RestoreByTemplateId -ArgumentList &lt;template_ID&gt;</pre>	Restores the user settings for an application or restores a group of Windows settings.

**\*\*Note\*\***  
UE-V does not provide a settings rollback for Windows apps.

## Have a suggestion for UE-V?

Add or vote on suggestions on the [User Experience Virtualization feedback site](#).

For UE-V issues, use the [UE-V TechNet Forum](#).

## Related topics

[Administering UE-V with Windows PowerShell and WMI](#)

[Administering UE-V](#)

# Changing the Frequency of UE-V Scheduled Tasks

6/20/2017 • 4 min to read • [Edit Online](#)

## Applies to

- Windows 10, version 1607

When the User Experience Virtualization (UE-V) service is enabled, it creates the following scheduled tasks:

- [Monitor Application Settings](#)
- [Sync Controller Application](#)
- [Synchronize Settings at Logoff](#)
- [Template Auto Update](#)

## Note

These tasks must remain enabled, because UE-V cannot function without them.

These scheduled tasks are not configurable with the UE-V tools. Administrators who want to change the scheduled task for these items can create a script that uses the Schtasks.exe command-line options.

For more information about Schtasks.exe, see [Schtasks](#).

## UE-V Scheduled Tasks

The following scheduled tasks are included in UE-V with sample scheduled task configuration commands.

### Monitor Application Settings

The **Monitor Application Settings** task is used to synchronize settings for Windows apps. It runs at logon but is delayed by 30 seconds to not affect the logon detrimentally. The Monitor Application Status task runs the UevAppMonitor.exe file, which is located in the UE-V Agent installation directory.

Task Name	Default Event
\Microsoft\UE-V\Monitor Application Status	Logon

### Sync Controller Application

The **Sync Controller Application** task is used to start the Sync Controller to synchronize settings from the computer to the settings storage location. By default, the task runs every 30 minutes. At that time, local settings are synchronized to the settings storage location, and updated settings on the settings storage location are synchronized to the computer. The Sync Controller application runs the Microsoft.Uev.SyncController.exe, which is located in the UE-V Agent installation directory.

Task Name	Default Event
\Microsoft\UE-V\Sync Controller Application	Logon, and every 30 minutes thereafter

For example, the following command configures the agent to synchronize settings every 15 minutes instead of the default 30 minutes.

```
Schtasks /change /tn "Microsoft\UE-V\Sync Controller Application" /ri 15
```

### Synchronize Settings at Logoff

The **Synchronize Settings at Logoff** task is used to start an application at logon that controls the synchronization of applications at logoff for UE-V. The Synchronize Settings at Logoff task runs the Microsoft.Uev.SyncController.exe file, which is located in the UE-V Agent installation directory.

TASK NAME	DEFAULT EVENT
\Microsoft\UE-V\Synchronize Settings at Logoff	Logon

### Template Auto Update

The **Template Auto Update** task checks the settings template catalog for new, updated, or removed templates. This task only runs if the SettingsTemplateCatalog is configured. The **Template Auto Update** task runs the ApplySettingsCatalog.exe file, which is located in the UE-V Agent installation directory.

TASK NAME	DEFAULT EVENT
\Microsoft\UE-V\Template Auto Update	System startup and at 3:30 AM every day, at a random time within a 1-hour window

**Example:** The following command configures the UE-V service to check the settings template catalog store every hour.

```
schtasks /change /tn "Microsoft\UE-V\Template Auto Update" /ri 60
```

## UE-V Scheduled Task Details

The following chart provides additional information about scheduled tasks for UE-V 2:

Task Name (file name)	Default Frequency	Power Toggle	Idle Only	Network Connection	Description
<b>Monitor Application Settings</b> (UevAppMon itor.exe)	Starts 30 seconds after logon and continues until logoff.	No	Yes	N/A	Synchronizes settings for Windows (AppX) apps.
<b>Sync Controller Application</b> (Microsoft.Ue v.SyncControl ler.exe)	At logon and every 30 min thereafter.	Yes	Yes	Only if Network is connected	Starts the Sync Controller which synchronizes local settings with the settings storage location.

<b>Synchronize Settings at Logoff</b> (Microsoft.UevSyncController.exe)	Runs at logon and then waits for Logoff to Synchronize settings.	No	Yes	N/A	Start an application at logon that controls the synchronization of applications at logoff.
<b>Template Auto Update</b> (ApplySettingsCatalog.exe)	Runs at initial logon and at 3:30 AM every day thereafter.	Yes	No	N/A	Checks the settings template catalog for new, updated, or removed templates. This task only runs if SettingsTemplateCatalog is configured.

### Legend

- **Power Toggle** – Task Scheduler will optimize power consumption when not connected to AC power. The task might stop running if the computer switches to battery power.
- **Idle Only** – The task will stop running if the computer ceases to be idle. By default the task will not restart when the computer is idle again. Instead the task will begin again on the next task trigger.
- **Network Connection** – Tasks marked “Yes” only run if the computer has a network connection available. Tasks marked “N/A” run regardless of network connectivity.

### How to Manage Scheduled Tasks

To find Scheduled Tasks, perform the following:

1. Open “Schedule Tasks” on the user computer.
2. Navigate to: Task Scheduler -> Task Scheduler Library -> Microsoft -> UE-V
3. Select the scheduled task you wish to manage and configure in the details pane.

### Additional information

The following additional information applies to UE-V scheduled tasks:

- All task sequence programs are located in the UE-V Agent installation folder, `%programFiles%\Microsoft User Experience Virtualization\Agent\[architecture]\`, by default.
- The Sync Controller Application Scheduled task is the crucial component when the UE-V SyncMethod is set to “SyncProvider” (UE-V default configuration). This scheduled task keeps the SettingsStoragePath synchronized with the locally cached versions of the settings package files. If users complain that settings do not synchronize often enough, then you can reduce the scheduled task setting to as little as 1 minute. You can also increase the 30 min default to a higher amount if necessary.
- You do not need to disable the Template Auto Update scheduled task if you use another method to keep the clients’ templates in sync (i.e. Group Policy or Configuration Manager Baselines). Leaving the SettingsTemplateCatalog property value blank prevents UE-V from checking the settings catalog for custom

templates. This scheduled task runs `ApplySettingsCatalog.exe` and will essentially return immediately.

- The Monitor Application Settings scheduled task will update Windows app (AppX) settings in real time, based on Windows app program setting triggers built into each app.

## Have a suggestion for UE-V?

Add or vote on suggestions on the [User Experience Virtualization feedback site](#).

For UE-V issues, use the [UE-V TechNet Forum](#).

## Related topics

[Administering UE-V](#)

[Deploy UE-V for Custom Applications](#)

# Migrating UE-V settings packages

6/20/2017 • 1 min to read • [Edit Online](#)

## Applies to

- Windows 10, version 1607

In the lifecycle of a User Experience Virtualization (UE-V) deployment, you might have to relocate the user settings packages either when you migrate to a new server or when you perform backups. Settings packages might have to be migrated in the following scenarios:

- Upgrade of existing server hardware to a more modern server
- Migration of a settings storage location share from a test server to a production server

Simply copying the files and folders does not preserve the security settings and permissions. The following steps describe how to correctly copy the settings package along with their NTFS file system permissions to a new share.

## To preserve UE-V settings packages when you migrate to a new server

1. In a new location on a different server, create a new folder, for example, MySettings.
2. Disable sharing for the old folder share on the old server.
3. To copy the existing settings packages to the new server with Robocopy

```
C:\start robocopy "\\servername\E$\MySettings" "\\servername\E$\MySettings" /b /sec /secrev /e
/LOG:D:\Robocopylogs\MySettings.txt
```

## Note

To monitor the copy progress, open MySettings.txt with a log viewer such as Trace32.

4. Grant share-level permissions to the new share. Leave the NTFS file system permissions as they were set by Robocopy.

On computers on which the UE-V service is enabled, update the **SettingsStoragePath** configuration setting to the Universal Naming Convention (UNC) path of the new share.

## Have a suggestion for UE-V?

Add or vote on suggestions on the [User Experience Virtualization feedback site](#).

For UE-V issues, use the [UE-V TechNet Forum](#).

## Related topics

[Administering UE-V](#)

# Using UE-V with Application Virtualization applications

6/20/2017 • 1 min to read • [Edit Online](#)

## Applies to

- Windows 10, version 1607

User Experience Virtualization (UE-V) supports Microsoft Application Virtualization (App-V) applications without any required modifications to either the App-V package or the UE-V template. However, an additional step is required because you cannot run the UE-V template generator directly on a virtualized App-V application. Instead, you must install the application locally, generate the template, and then apply the template to the virtualized application. UE-V supports App-V for Windows 10 packages and App-V 5.0 packages.

## UE-V settings synchronization for App-V applications

UE-V monitors when an application opens by the program name and, optionally, by file version numbers and product version numbers, whether the application is installed locally or virtually by using App-V. When the application starts, UE-V monitors the App-V process, applies any settings that are stored in the user's settings storage path, and then enables the application to start normally. UE-V monitors App-V applications and automatically translates the relevant file and registry paths to the virtualized location as opposed to the physical location outside the App-V computing environment.

### To implement settings synchronization for a virtualized application

- Run the UE-V template generator to collect the settings of the locally installed application whose settings you want to synchronize between computers. This process creates a settings location template. If you use a built-in template such as a Microsoft Office template, skip this step. For more information about using the UE-V template generator, see [Deploy UE-V for custom applications](#).
- Install the App-V application package if you have not already done so.
- Publish the template to the location of your settings template catalog or manually install the template by using the `Register-UEVTemplate` Windows PowerShell cmdlet.

#### Note

If you publish the newly created template to the settings template catalog, the client does not receive the template until the sync provider updates the settings. To manually start this process, open **Task Scheduler**, expand **Task Scheduler Library**, expand **Microsoft**, and expand **UE-V**. In the results pane, right-click **Template Auto Update**, and then click **Run**.

- Start the App-V package.

## Have a suggestion for UE-V?

Add or vote on suggestions on the [User Experience Virtualization feedback site](#).

For UE-V issues, use the [UE-V TechNet Forum](#).

## Related topics

[Administering UE-V](#)

# Troubleshooting UE-V

6/20/2017 • 1 min to read • [Edit Online](#)

## Applies to

- Windows 10, version 1607

For information that can help with troubleshooting UE-V for Windows 10, see:

- [UE-V FAQ Wiki](#)
- [UE-V: List of Microsoft Support Knowledge Base Articles](#)
- [User Experience Virtualization Release Notes](#)
- [Technical Reference for UE-V](#)
- [UE-V TechNet Forum](#)

## Other resources

- [User Experience Virtualization overview](#)
- [Get Started with UE-V](#)
- [Prepare a UE-V deployment](#)
- [Administering UE-V](#)

## Have a suggestion for UE-V?

Add or vote on suggestions on the [User Experience Virtualization feedback site](#).

For UE-V issues, use the [UE-V TechNet Forum](#).

# Technical Reference for UE-V

6/20/2017 • 1 min to read • [Edit Online](#)

## Applies to

- Windows 10, version 1607

This technical reference section includes additional technical documentation about the various features of User Experience Virtualization (UE-V). This information is provided to help the administrator better understand UE-V.

## Technical reference topics for UE-V

- [Sync Methods for UE-V](#)

Defines how UE-V synchronizes settings between computers and the settings storage location. Sync Provider is the default sync method for UE-V. This topic includes technical reference information for sync methods, including the Sync Provider.

- [Sync Trigger Events for UE-V](#)

Defines when the UE-V service synchronizes those settings with the settings storage location. This topic provides technical reference information about when synchronization takes place based upon the sync method deployed.

- [Synchronizing Microsoft Office with UE-V](#)

Provides guidance for downloading and enabling the Microsoft-authored UE-V settings location templates that support Microsoft Office settings synchronization.

- [Application Template Schema Reference for UE-V](#)

Details the XML structure of UE-V settings location templates and provides guidance for editing these files.

- [Security Considerations for UE-V](#)

Provides a brief overview of accounts, groups, and other security-related considerations for UE-V.

## Other resources for this feature

- [User Experience Virtualization overview](#)
- [Get Started with UE-V](#)
- [Prepare a UE-V Deployment](#)
- [Administering UE-V](#)
- [Troubleshooting UE-V](#)

## Have a suggestion for UE-V?

Add or vote on suggestions on the [User Experience Virtualization feedback site](#).

For UE-V issues, use the [UE-V TechNet Forum](#).

# Sync Methods for UE-V

6/20/2017 • 2 min to read • [Edit Online](#)

## Applies to

- Windows 10, version 1607

The User Experience Virtualization (UE-V) service lets you synchronize users' application and Windows settings with the settings storage location. The *Sync Method* configuration defines how the UE-V service uploads and downloads those settings to the settings storage location. UE-V includes a SyncMethod called the *SyncProvider*. For more information about trigger events that start the synchronization of application and Windows settings, see [Sync Trigger Events for UE-V](#).

## SyncMethod Configuration

This table provides a description of each SyncMethod configuration:

SYNCMETHOD CONFIGURATION	DESCRIPTION
SyncProvider (Default)	<p>Settings changes for a specific application or for global Windows desktop settings are saved locally to a cache folder. These changes are then synchronized with the settings storage location when a synchronization trigger event takes place. Pushing out changes will save the local changes to the settings storage path.</p> <p>This default setting is the gold standard for computers. This option attempts to synchronize the setting and times out after a short delay to ensure that the application or operating system startup isn't delayed for a long period of time. This functionality is also tied to the Scheduled task – Sync Controller Application. The administrator controls the frequency of the Scheduled task. By default, computers synchronize their settings every 30 min after logging on.</p>
External	<p>This configuration method specifies that if UE-V settings are written to a local folder on the user computer, then any external sync engine (such as OneDrive for Business, Work Folders, Sharepoint, or Dropbox) can be used to apply these settings to the different computers that users access.</p>

SYNCMETHOD CONFIGURATION	DESCRIPTION
None	<p>This configuration setting is designed for the Virtual Desktop Infrastructure (VDI) and Streamed Application experience primarily. This setting should be used on computers running the Windows Server operating system in a datacenter, where the connection will always be available.</p> <p>Any settings changes are saved directly to the server. If the network connection to the settings storage path is not available, then the settings changes are cached on the device and are synchronized the next time that the Sync Provider runs. If the settings storage path is not found and the user profile is removed from a pooled VDI environment on logoff, then these settings changes are lost, and the user must reapply the change when the computer can again reach the settings storage path.</p> <p>Apps and OS will wait indefinitely for the location to be present. This could cause App load or OS logon time to dramatically increase if the location is not found.</p>

You can configure the sync method in these ways:

- Through [Group Policy](#) settings
- With the [System Center Configuration Pack](#) for UE-V
- With [Windows PowerShell](#) or [Windows Management Instrumentation \(WMI\)](#)

## Have a suggestion for UE-V?

Add or vote on suggestions on the [User Experience Virtualization feedback site](#).

For UE-V issues, use the [UE-V TechNet Forum](#).

## Related topics

[Deploy Required UE-V Features](#)

[Technical Reference for UE-V](#)

# Sync Trigger Events for UE-V

6/20/2017 • 2 min to read • [Edit Online](#)

## Applies to

- Windows 10, version 1607

User Experience Virtualization (UE-V) lets you synchronize your application and Windows settings across all your domain-joined devices. *Sync trigger events* define when the UE-V service synchronizes those settings with the settings storage location. For more information about Sync Method configuration, see [Sync Methods for UE-V](#).

## UE-V Sync Trigger Events

The following table explains the trigger events for classic applications and Windows settings.

UE-V Trigger Event	SyncMethod=SyncProvider	SyncMethod=None
<b>Windows Logon</b>	<ul style="list-style-type: none"><li>Application and Windows settings are imported to the local cache from the settings storage location.</li><li><a href="#">Asynchronous Windows settings</a> are applied.</li><li>Synchronous Windows settings will be applied during the next Windows logon.</li><li>Application settings will be applied when the application starts.</li></ul>	<ul style="list-style-type: none"><li>Application and Windows settings are read directly from the settings storage location.</li><li>Asynchronous and synchronous Windows settings are applied.</li><li>Application settings will be applied when the application starts.</li></ul>
<b>Windows Logoff</b>	Store changes locally and cache and copy asynchronous and synchronous Windows settings to the settings storage location server, if available	Store changes to asynchronous and synchronous Windows settings storage location
<b>Windows Connect (RDP) / Unlock</b>	Synchronize any asynchronous Windows settings from settings storage location to local cache, if available. Apply cached Windows settings	Download and apply asynchronous windows settings from settings storage location

<b>Windows Disconnect (RDP) / Lock</b>	<p>Store asynchronous Windows settings changes to the local cache.</p> <p>Synchronize any asynchronous Windows settings from the local cache to settings storage location, if available</p>	Store asynchronous Windows settings changes to the settings storage location
<b>Application start</b>	Apply application settings from local cache as the application starts	Apply application settings from settings storage location as the application starts
<b>Application closes</b>	Store any application settings changes to the local cache and copy settings to settings storage location, if available	Store any application settings changes to settings storage location
<b>Sync Controller Scheduled Task</b>	<p>Application and Windows settings are synchronized between the settings storage location and the local cache.</p> <div style="border: 1px solid black; padding: 10px;"> <p><b>Note</b></p> <p>Settings changes are not cached locally until an application closes. This trigger will not export changes made to a currently running application.</p> <p>For Windows settings, this means that any changes will not be cached locally and exported until the next Lock (Asynchronous) or Logoff (Asynchronous and Synchronous).</p> </div>	NA
<b>Asynchronous Settings updated on remote store*</b>	<p>Settings are applied in these cases:</p> <ul style="list-style-type: none"> <li>• Asynchronous Windows settings are applied directly.</li> <li>• Application settings are applied when the application starts.</li> <li>• Both asynchronous and synchronous Windows settings are applied during the next Windows logon.</li> <li>• Windows app (AppX) settings are applied during the next refresh. See <a href="#">Monitor Application Settings</a> for more information.</li> </ul>	Load and apply settings from central server

## Have a suggestion for UE-V?

Add or vote on suggestions on the [User Experience Virtualization feedback site](#).

For UE-V issues, use the [UE-V TechNet Forum](#).

## Related topics

[Technical Reference for UE-V](#)

[Changing the Frequency of UE-V Scheduled Tasks](#)

[Choose the Configuration Method for UE-V](#)

# Synchronizing Office with UE-V

6/20/2017 • 2 min to read • [Edit Online](#)

## Applies to

- Windows 10, version 1607

Microsoft User Experience Virtualization (UE-V) supports the synchronization of Microsoft Office application settings. The combination of UE-V and App-V support for Office enables the same experience on virtualized instances of Office from any UE-V-enabled device or virtualized desktop.

To synchronize Office applications settings, you can download Office templates from the [User Experience Virtualization \(UE-V\) Template Gallery](#). This resource provides Microsoft-authored UE-V settings location templates as well as community-developed settings location templates.

## Microsoft Office support in UE-V

UE-V includes settings location templates for Microsoft Office 2016, 2013, and 2010. In previous versions of UE-V, settings location templates for Office 2013 and Office 2010 were distributed and registered when you installed the UE-V agent. Now that UE-V is a feature in Windows 10, version 1607, settings location templates are installed when you install or upgrade to the new operating system.

These templates help synchronize users' Office experience between devices. Microsoft Office 2016 settings roamed by Office 365 experience are not included in these settings. For a list of Office 365-specific settings, see [Overview of user and roaming settings for Office](#).

## Synchronized Office Settings

Review the following tables for details about Office support in UE-V:

### Supported UE-V templates for Microsoft Office

OFFICE 2016 TEMPLATES (UE-V FOR WINDOWS 10 AND WINDOWS 10, VERSION 1607, AVAILABLE IN UE-V GALLERY)	OFFICE 2013 TEMPLATES (UE-V FOR WINDOWS 10 AND UE-V 2.X, AVAILABLE ON UE-V GALLERY)	OFFICE 2010 TEMPLATES (UE-V 1.0 AND 1.0 SP1)
MicrosoftOffice2016Win32.xml	MicrosoftOffice2013Win32.xml	MicrosoftOffice2010Win32.xml
MicrosoftOffice2016Win64.xml	MicrosoftOffice2013Win64.xml	MicrosoftOffice2010Win64.xml
MicrosoftSkypeForBusiness2016Win32.xml	MicrosoftLync2013Win32.xml	MicrosoftLync2010.xml
MicrosoftSkypeForBusiness2016Win64.xml	MicrosoftLync2013Win64.xml	

### Microsoft Office Applications supported by the UE-V templates

Microsoft Access 2016	Microsoft Access 2013	Microsoft Access 2010
Microsoft Lync 2016	Microsoft Lync 2013	Microsoft Lync 2010
Microsoft Excel 2016	Microsoft Excel 2013	Microsoft Excel 2010
Microsoft OneNote 2016	Microsoft InfoPath 2013	Microsoft InfoPath 2010
Microsoft Outlook 2016	Microsoft OneNote 2013	Microsoft OneNote 2010
Microsoft PowerPoint 2016	Microsoft Outlook 2013	Microsoft Outlook 2010
Microsoft Project 2016	Microsoft PowerPoint 2013	Microsoft PowerPoint 2010
Microsoft Publisher 2016	Microsoft Project 2013	Microsoft Project 2010
Microsoft SharePoint Designer 2013 (not updated for 2016)	Microsoft Publisher 2013	Microsoft Publisher 2010
Microsoft Visio 2016	Microsoft SharePoint Designer 2013	Microsoft SharePoint Designer 2010
Microsoft Word 2016	Microsoft Visio 2013	Microsoft Visio 2010
Microsoft Office Upload Manager	Microsoft Word 2013	Microsoft Word 2010
	Microsoft Office Upload Manager	

## Deploying Office templates

You can deploy UE-V settings location template with the following methods:

- **Registering template with PowerShell.** If you use Windows PowerShell to manage computers, run the following Windows PowerShell command as Administrator to register this settings location template:

```
Register-UevTemplate -Path <Path_to_Template>
```

For more information about using UE-V and Windows PowerShell, see [Managing UE-V settings location templates using Windows PowerShell and WMI](#).

- **Registering template with Template Catalog Path.** If you use the Settings Template Catalog Path to manage templates on users' computers, copy the Office template into the folder defined in the UE-V service. The next time the Template Auto Update (ApplySettingsCatalog.exe) scheduled task runs, the settings location template will be registered on the device. For more information, see [Deploy a settings template catalog](#).
- **Registering template with Configuration Manager.** If you use Configuration Manager to manage your UE-V settings storage templates, recreate the Template Baseline CAB, import it into Configuration Manager, and then deploy the baseline to user devices. For more information, see the guidance provided in the documentation for the [System Center 2012 Configuration Pack for Microsoft User Experience Virtualization 2.0](#).

## Have a suggestion for UE-V?

Add or vote on suggestions on the [User Experience Virtualization feedback site](#).

For UE-V issues, use the [UE-V TechNet Forum](#).

# Application Template Schema Reference for UE-V

6/20/2017 • 20 min to read • [Edit Online](#)

## Applies to

- Windows 10, version 1607

User Experience Virtualization (UE-V) uses XML settings location templates to define the desktop application settings and Windows settings that are captured and applied by UE-V. UE-V includes a set of default settings location templates. You can also create custom settings location templates with the UE-V template generator.

An advanced user can customize the XML file for a settings location template. This topic details the XML structure of the UE-V settings location templates and provides guidance for editing these files.

## UE-V Application Template Schema Reference

This section details the XML structure of the UE-V settings location template and provides guidance for editing this file.

### In This Section

- [XML Declaration and Encoding Attribute](#)
- [Namespace and Root Element](#)
- [Data types](#)
- [Name Element](#)
- [ID Element](#)
- [Version Element](#)
- [Author Element](#)
- [Processes and Process Element](#)
- [Application Element](#)
- [Common Element](#)
- [SettingsLocationTemplate Element](#)
- [Appendix: SettingsLocationTemplate.xsd](#)

### XML Declaration and Encoding Attribute

**Mandatory:** True

**Type:** String

The XML declaration must specify the XML version 1.0 attribute (<?xml version="1.0"?>). Settings location templates created by the UE-V template generator are saved in UTF-8 encoding, although the encoding is not explicitly specified. We recommend that you include the encoding="UTF-8" attribute in this element as a best practice. All templates included with the product specify this tag as well (see the documents in %ProgramFiles%\Microsoft User Experience Virtualization\Templates for reference). For example:

```
<?xml version="1.0" encoding="UTF-8"?>
```

## Namespace and Root Element

**Mandatory:** True

**Type:** String

UE-V uses the <http://schemas.microsoft.com/UserExperienceVirtualization/2012/SettingsLocationTemplate> namespace for all applications. SettingsLocationTemplate is the root element and contains all other elements. Reference SettingsLocationTemplate in all templates using this tag:

```
<SettingsLocationTemplate
xmlns='http://schemas.microsoft.com/UserExperienceVirtualization/2012/SettingsLocationTemplate'>
```

## Data types

These are the data types for the UE-V application template schema.

### GUID

GUID describes a standard globally unique identifier regular expression in the form "\{[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{12}\}". This is used in the Filesetting\Root\KnownFolder element to verify the formatting of well-known folders.

### FilenameString

FilenameString refers to the file name of a process to be monitored. Its values are restricted by the regex `[^\\?\\*\\<\\>/:\\]+`, (that is, they may not contain backslash characters, asterisk or question mark wild-card characters, the pipe character, the greater than or less than sign, forward slash, or colon characters).

### IDString

IDString refers to the ID value of Application elements, SettingsLocationTemplate, and Common elements (used to describe application suites that share common settings). It is restricted by the same regex as FilenameString (`[^\\?\\*\\<\\>/:\\]+`).

### TemplateVersion

TemplateVersion is an integer value used to describe the revision of the settings location template. Its value may range from 0 to 2147483647.

### Empty

Empty refers to a null value. This is used in Process\ShellProcess to indicate that there is no process to monitor. This value should not be used in any application templates.

### Author

The Author data type is a complex type that identifies the author of a template. It contains two child elements:

**Name** and **Email**. Within the Author data type, the Name element is mandatory while the Email element is optional. This type is described in more detail under the SettingsLocationTemplate element.

### Range

Range defines an integer class consisting of two child elements: **Minimum** and **Maximum**. This data type is implemented in the ProcessVersion data type. If specified, both Minimum and Maximum values must be included.

### ProcessVersion

ProcessVersion defines a type with four child elements: **Major**, **Minor**, **Build**, and **Patch**. This data type is used by the Process element to populate its ProductVersion andFileVersion values. The data for this type is a Range value. The Major child element is mandatory and the others are optional.

### Architecture

Architecture enumerates two possible values: **Win32** and **Win64**. These values are used to specify process architecture.

### Process

The Process data type is a container used to describe processes to be monitored by UE-V. It contains six child

elements: **Filename**, **Architecture**, **ProductName**, **FileDescription**, **ProductVersion**, and **FileVersion**. This table details each element's respective data type:

Element	Data Type	Mandatory
Filename	FilenameString	True
Architecture	Architecture	False
ProductName	String	False
FileDescription	String	False
ProductVersion	ProcessVersion	False
FileVersion	ProcessVersion	False

### Processes

The Processes data type represents a container for a collection of one or more Process elements. Two child elements are supported in the Processes sequence type: **Process** and **ShellProcess**. Process is an element of type Process and ShellProcess is of data type Empty. At least one item must be identified in the sequence.

### Path

Path is consumed by RegistrySetting and FileSetting to refer to registry and file paths. This element supports two optional attributes: **Recursive** and **DeleteIfNotFound**. Both values are set to default="False".

Recursive indicates that the path and all subfolders are included for file settings or that all child registry keys are included for registry settings. In both cases, all items at the current level are included in the data captured. For a FileSettings object, all files within the specified folder are included in the data captured by UE-V but folders are not included. For registry paths, all values in the current path are captured but child registry keys are not captured. In both cases, care should be taken to avoid capturing large data sets or large numbers of items.

The DeleteIfNotFound attribute removes the setting from the user's settings storage path data. This may be desirable in cases where removing these settings from the package will save a large amount of disk space on the settings storage path file server.

### FileMask

FileMask specifies only certain file types for the folder that is defined by Path. For example, Path might be `C:\users\username\files` and FileMask could be `*.txt` to include only text files.

### RegistrySetting

RegistrySetting represents a container for registry keys and values and the associated desired behavior on the part of the UE-V service. Four child elements are defined within this type: **Path**, **Name**, **Exclude**, and a sequence of the values **Path** and **Name**.

### FileSetting

FileSetting contains parameters associated with files and files paths. Four child elements are defined: **Root**, **Path**, **FileMask**, and **Exclude**. Root is mandatory and the others are optional.

### Settings

Settings is a container for all the settings that apply to a particular template. It contains instances of the Registry,

File, SystemParameter, and CustomAction settings described earlier. In addition, it can also contain the following child elements with behaviors described:

Element	Description
Asynchronous	Asynchronous settings packages are applied without blocking the application startup so that the application start proceeds while the settings are still being applied. This is useful for settings that can be applied asynchronously, such as those <code>get/set</code> through an API, like SystemParameterSetting.
PreventOverlappingSynchronization	By default, UE-V only saves settings for an application when the last instance of an application using the template is closed. When this element is set to 'false', UE-V exports the settings even if other instances of an application are running. Suited templates – those that include a Common element section – that are shipped with UE-V use this flag to enable shared settings to always export on application close, while preventing application-specific settings from exporting until the last instance is closed.
AlwaysApplySettings	This parameter forces an imported settings package to be applied even if there are no differences between the package and the current state of the application. This parameter should be used only in special cases since it can slow down settings import.

## Name Element

**Mandatory: True**

**Type: String**

Name specifies a unique name for the settings location template. This is used for display purposes when referencing the template in WMI, PowerShell, Event Viewer and debug logs. In general, avoid referencing version information, as this can be objected from the ProductVersion element. For example, specify

`<Name>My Application</Name>` rather than `<Name>My Application 1.1</Name>`.

## Note

UE-V does not reference external DTDs, so it is not possible to use named entities in a settings location template. For example, do not use ® to refer to the registered trade mark sign ®. Instead, use canonical numbered references to include these types of special characters, for example, &#174 for the ® character. This rule applies to all string values in this document.

See <http://www.w3.org/TR/xhtml1/dtds.html> for a complete list of character entities. UTF-8-encoded documents may include the Unicode characters directly. Saving templates through the UE-V template generator converts character entities to their Unicode representations automatically.

## ID Element

**Mandatory: True**

**Type: String**

ID populates a unique identifier for a particular template. This tag becomes the primary identifier that the UE-V service uses to reference the template at runtime (for example, see the output of the Get-UevTemplate and Get-

UevTemplateProgram PowerShell cmdlets). By convention, this tag should not contain any spaces, which simplifies scripting. Version numbers of applications should be specified in this element to allow for easy identification of the template, such as `<ID>MicrosoftOffice2016Win64</ID>`.

### Version Element

**Mandatory:** True

**Type:** Integer

**Minimum Value:** 0

**Maximum Value:** 2147483647

Version identifies the version of the settings location template for administrative tracking of changes. The UE-V template generator automatically increments this number by one each time the template is saved. Notice that this field must be a whole number integer; fractional values, such as `<version>2.5</Version>` are not allowed.

**Hint:** You can save notes about version changes using XML comment tags `<!-- -->`, for example:

```
<!--
 Version History

 Version 1 Jul 05, 2012 Initial template created by Generator - Denise@Contoso.com
 Version 2 Jul 31, 2012 Added support for app.exe v2.1.3 - Mark@Contoso.com
 Version 3 Jan 01, 2013 Added font settings support - Mark@Contoso.com
 Version 4 Jan 31, 2013 Added support for plugin settings - Tony@Contoso.com
-->
<Version>4</Version>
```

### Important

This value is queried to determine if a new version of a template should be applied to an existing template in these instances:

- When the scheduled Template Auto Update task executes
- When the Update-UevTemplate PowerShell cmdlet is executed
- When the `microsoft\uev:SettingsLocationTemplate` Update method is called through WMI

### Author Element

**Mandatory:** False

**Type:** String

Author identifies the creator of the settings location template. Two optional child elements are supported: **Name** and **Email**. Both attributes are optional, but, if the Email child element is specified, it must be accompanied by the Name element. Author refers to the full name of the contact for the settings location template, and email should refer to an email address for the author. We recommend that you include this information in templates published publicly, for example, on the [UE-V Template Gallery](#).

### Processes and Process Element

**Mandatory:** True

**Type:** Element

Processes contains at least one `<Process>` element, which in turn contains the following child elements: **Filename**, **Architecture**, **ProductName**, **FileDescription**, **ProductVersion**, and **FileVersion**. The **Filename** child element is mandatory and the others are optional. A fully populated element contains tags similar to this example:

```

<Process>
 <Filename>MyApplication.exe</Filename>
 <Architecture>Win64</Architecture>
 <ProductName> MyApplication </ProductName>
 <FileDescription>MyApplication.exe</FileDescription>
 <ProductVersion>
 <Major Minimum="2" Maximum="2" />
 <Minor Minimum="0" Maximum="0" />
 <Build Minimum="0" Maximum="0" />
 <Patch Minimum="5" Maximum="5" />
 </ProductVersion>
 <FileVersion>
 <Major Minimum="2" Maximum="2" />
 <Minor Minimum="0" Maximum="0" />
 <Build Minimum="0" Maximum="0" />
 <Patch Minimum="5" Maximum="5" />
 </FileVersion>
</Process>

```

## **Filename**

**Mandatory: True**

**Type: String**

Filename refers to the actual file name of the executable as it appears in the file system. This element specifies the primary criterion that UE-V uses to evaluate whether a template applies to a process or not. This element must be specified in the settings location template XML.

Valid filenames must not match the regular expression `[^\\\?\\*\\|<>/:]+`, that is, they may not contain backslash characters, asterisk or question mark wild-card characters, the pipe character, the greater than or less than sign, forward slash, or colon (the \ ? \* | < > / or : characters.).

**Hint:** To test a string against this regex, use a PowerShell command window and substitute your executable's name for **YourFileName**:

```
"YourFileName.exe" -match "[\\\?*\\|<>/:]+"
```

A value of **True** indicates that the string contains illegal characters. Here are some examples of illegal values:

- \\server\share\program.exe
- Program\*.exe
- Pro?ram.exe
- Program<1>.exe

## **Note**

The UE-V template generator encodes the greater than and less than characters as > and < respectively.

In rare circumstances, the FileName value will not necessarily include the .exe extension, but it should be specified as part of the value. For example, `<Filename>MyApplication.exe</Filename>` should be specified instead of `<Filename>MyApplication</Filename>`. The second example will not apply the template to the process if the actual name of the executable file is "MyApplication.exe".

## **Architecture**

**Mandatory: False**

**Type: Architecture (String)**

Architecture refers to the processor architecture for which the target executable was compiled. Valid values are

Win32 for 32-bit applications or Win64 for 64-bit applications. If present, this tag limits the applicability of the settings location template to a particular application architecture. For an example of this, compare the %ProgramFiles%\Microsoft User Experience Virtualization\templates\ MicrosoftOffice2016Win32.xml and MicrosoftOffice2016Win64.xml files included with UE-V. This is useful when relative paths change between different versions of an executable or if settings have been added or removed when moving from one processor architecture to another.

If this element is absent, the settings location template ignores the process' architecture and applies to both 32 and 64-bit processes if the file name and other attributes apply.

#### **Note**

UE-V does not support ARM processors in this version.

#### **ProductName**

**Mandatory: False**

#### **Type: String**

ProductName is an optional element used to identify a product for administrative purposes or reporting.

ProductName differs from Filename in that there are no regular expression restrictions on its value. This allows for more easily understood descriptions of a process where the executable name may not be obvious. For example:

```
<Process>
 <Filename>MyApplication.exe</Filename>
 <ProductName>My Application 6.x by Contoso.com</ProductName>
 <ProductVersion>
 <Major Minimum="6" Maximum="6" />
 </ProductVersion>
</Process>
```

#### **FileDescription**

**Mandatory: False**

#### **Type: String**

FileDescription is an optional tag that allows for an administrative description of the executable file. This is a free text field and can be useful in distinguishing multiple executables within a software package where there is a need to identify the function of the executable.

For example, in a suited application, it might be useful to provide reminders about the function of two executables (MyApplication.exe and MyApplicationHelper.exe), as shown here:

```
<Processes>

 <Process>
 <Filename>MyApplication.exe</Filename>
 <FileDescription>My Application Main Engine</ FileDescription>
 <ProductVersion>
 <Major Minimum="6" Maximum="6" />
 </ProductVersion>
 </Process>
 <Process>
 <Filename>MyApplicationHelper.exe</Filename>
 <FileDescription>My Application Background Process Executable</FileDescription>
 <ProductVersion>
 <Major Minimum="6" Maximum="6" />
 </ProductVersion>
 </Process>
</Processes>
```

## **ProductVersion**

**Mandatory:** False

**Type:** String

ProductVersion refers to the major and minor product versions of a file, as well as a build and patch level. ProductVersion is an optional element, but if specified, it must contain at least the Major child element. The value must express a range in the form Minimum="X" Maximum="Y" where X and Y are integers. The Minimum and Maximum values can be identical.

The product and file version elements may be left unspecified. Doing so makes the template "version agnostic", meaning that the template will apply to all versions of the specified executable.

### **Example 1:**

Product version: 1.0 specified in the UE-V template generator produces the following XML:

```
<ProductVersion>
 <Major Minimum="1" Maximum="1" />
 <Minor Minimum="0" Maximum="0" />
</ProductVersion>
```

### **Example 2:**

File version: 5.0.2.1000 specified in the UE-V template generator produces the following XML:

```
<FileVersion>
 <Major Minimum="5" Maximum="5" />
 <Minor Minimum="0" Maximum="0" />
 <Build Minimum="2" Maximum="2" />
 <Patch Minimum="1000" Maximum="1000" />
</FileVersion>
```

### **Incorrect Example 1 – incomplete range:**

Only the Minimum attribute is present. Maximum must be included in a range as well.

```
<ProductVersion>
 <Major Minimum="2" />
</ProductVersion>
```

### **Incorrect Example 2 – Minor specified without Major element:**

Only the Minor element is present. Major must be included as well.

```
<ProductVersion>
 <Minor Minimum="0" Maximum="0" />
</ProductVersion>
```

## **FileVersion**

**Mandatory:** False

**Type:** String

FileVersion differentiates between the release version of a published application and the internal build details of a component executable. For the majority of commercial applications, these numbers are identical. Where they vary, the product version of a file indicates a generic version identification of a file, while file version indicates a specific

build of a file (as in the case of a hotfix or update). This uniquely identifies files without breaking detection logic.

To determine the product version and file version of a particular executable, right-click on the file in Windows Explorer, select Properties, then click on the Details tab.

Including a FileVersion element for an application allows for more granular fine-tuning detection logic, but is not necessary for most applications. The ProductVersion element settings are checked first, and then FileVersion is checked. The more restrictive setting will apply.

The child elements and syntax rules for FileVersion are identical to those of ProductVersion.

```
<Process>
 <Filename>MSACCESS.EXE</Filename>
 <Architecture>Win32</Architecture>
 <ProductVersion>
 <Major Minimum="14" Maximum="14" />
 <Minor Minimum="0" Maximum="0" />
 </ProductVersion>
 <FileVersion>
 <Major Minimum="14" Maximum="14" />
 <Minor Minimum="0" Maximum="0" />
 </FileVersion>
</Process>
```

## Application Element

Application is a container for settings that apply to a particular application. It is a collection of the following fields/types.

Field/Type	Description
Name	Specifies a unique name for the settings location template. This is used for display purposes when referencing the template in WMI, PowerShell, Event Viewer and debug logs. For more information, see <a href="#">Name</a> .
ID	Populates a unique identifier for a particular template. This tag becomes the primary identifier that the UE-V service uses to reference the template at runtime. For more information, see <a href="#">ID</a> .
Description	An optional description of the template.
LocalizedNames	An optional name displayed in the UI, localized by a language locale.
LocalizedDescriptions	An optional template description localized by a language locale.
Version	Identifies the version of the settings location template for administrative tracking of changes. For more information, see <a href="#">Version</a> .

DeferToMSAccount	Controls whether this template is enabled in conjunction with a Microsoft account or not. If MSA syncing is enabled for a user on a machine, then this template will automatically be disabled.
DeferToOffice365	Similar to MSA, this controls whether this template is enabled in conjunction with Office365. If Office 365 is being used to sync settings, this template will automatically be disabled.
FixedProfile	Specifies that this template can only be associated with the profile specified within this element, and cannot be changed via WMI or PowerShell.
Processes	A container for a collection of one or more Process elements. For more information, see <a href="#">Processes</a> .
Settings	A container for all the settings that apply to a particular template. It contains instances of the Registry, File, SystemParameter, and CustomAction settings. For more information, see <a href="#">Settings</a> in <a href="#">Data types</a> .

## Common Element

Common is similar to an Application element, but it is always associated with two or more Application elements. The Common section represents the set of settings that are shared between those Application instances. It is a collection of the following fields/types.

Field/Type	Description
Name	Specifies a unique name for the settings location template. This is used for display purposes when referencing the template in WMI, PowerShell, Event Viewer and debug logs. For more information, see <a href="#">Name</a> .
ID	Populates a unique identifier for a particular template. This tag becomes the primary identifier that the UE-V service uses to reference the template at runtime. For more information, see <a href="#">ID</a> .
Description	An optional description of the template.
LocalizedNames	An optional name displayed in the UI, localized by a language locale.
LocalizedDescriptions	An optional template description localized by a language locale.

Version	Identifies the version of the settings location template for administrative tracking of changes. For more information, see <a href="#">Version</a> .
DeferToMSAccount	Controls whether this template is enabled in conjunction with a Microsoft account or not. If MSA syncing is enabled for a user on a machine, then this template will automatically be disabled.
DeferToOffice365	Similar to MSA, this controls whether this template is enabled in conjunction with Office365. If Office 365 is being used to sync settings, this template will automatically be disabled.
FixedProfile	Specifies that this template can only be associated with the profile specified within this element, and cannot be changed via WMI or PowerShell.
Settings	A container for all the settings that apply to a particular template. It contains instances of the Registry, File, SystemParameter, and CustomAction settings. For more information, see <a href="#">Settings</a> in <a href="#">Data types</a> .

### SettingsLocationTemplate Element

This element defines the settings for a single application or a suite of applications.

Field/Type	Description
Name	Specifies a unique name for the settings location template. This is used for display purposes when referencing the template in WMI, PowerShell, Event Viewer and debug logs. For more information, see <a href="#">Name</a> .
ID	Populates a unique identifier for a particular template. This tag becomes the primary identifier that the UE-V service uses to reference the template at runtime. For more information, see <a href="#">ID</a> .
Description	An optional description of the template.
LocalizedNames	An optional name displayed in the UI, localized by a language locale.
LocalizedDescriptions	An optional template description localized by a language locale.

### Appendix: SettingsLocationTemplate.xsd

Here is the SettingsLocationTemplate.xsd file showing its elements, child elements, attributes, and parameters:

```

<?xml version="1.0" encoding="utf-8"?>
<xss:schema id="UevSettingsLocationTemplate"
 targetNamespace="http://schemas.microsoft.com/UserExperienceVirtualization/2013A/SettingsLocationTemplate"
 elementFormDefault="qualified"
 xmlns="http://schemas.microsoft.com/UserExperienceVirtualization/2013A/SettingsLocationTemplate"
 xmlns:mstns="http://schemas.microsoft.com/UserExperienceVirtualization/2013A/SettingsLocationTemplate"
 xmlns:xs="http://www.w3.org/2001/XMLSchema">

 <xss:simpleType name="Guid">
 <xss:restriction base="xs:string">
 <xss:pattern value="\\{[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{12}\\}" />
 </xss:restriction>
 </xss:simpleType>

 <xss:simpleType name="FilenameString">
 <xss:restriction base="xs:string">
 <xss:pattern value="[^\\\\?*\\|<>/:.]+"/>
 </xss:restriction>
 </xss:simpleType>

 <xss:simpleType name="IDString">
 <xss:restriction base="xs:string">
 <xss:pattern value="[^\\\\?*\\|<>/:.]+"/>
 </xss:restriction>
 </xss:simpleType>

 <xss:simpleType name="CompositeIDString">
 <xss:restriction base="xs:string">
 <xss:pattern value="[^\\\\?*\\|<>/:.]+([.][^\\\\?*\\|<>/:.]+)*"/>
 </xss:restriction>
 </xss:simpleType>

 <xss:simpleType name="TemplateVersion">
 <xss:restriction base="xs:integer">
 <xss:minInclusive value="0"/>
 <xss:maxInclusive value="2147483647"/>
 </xss:restriction>
 </xss:simpleType>

 <xss:complexType name="Empty">
 <xss:sequence/>
 </xss:complexType>

 <xss:complexType name="LocalizedString">
 <xss:simpleContent>
 <xss:extension base="xs:string">
 <xss:attribute name="Locale" type="xs:string" use="required"/>
 </xss:extension>
 </xss:simpleContent>
 </xss:complexType>

 <xss:complexType name="LocalizedName">
 <xss:sequence>
 <xss:element name="Name" type="LocalizedString" minOccurs="1" maxOccurs="unbounded"/>
 </xss:sequence>
 </xss:complexType>

 <xss:complexType name="LocalizedDescription">
 <xss:sequence>
 <xss:element name="Description" type="LocalizedString" minOccurs="1" maxOccurs="unbounded"/>
 </xss:sequence>
 </xss:complexType>

 <xss:complexType name="ReplacedTemplates">
 <xss:sequence>
 <xss:element name="ID" type="CompositeIDString" minOccurs="1" maxOccurs="unbounded"/>
 </xss:sequence>
 </xss:complexType>

```

```

</xs:complexType>

<xs:complexType name="Author">
 <xs:all>
 <xs:element name="Name" type="xs:string" minOccurs="1" />
 <xs:element name="Email" type="xs:string" minOccurs="0" />
 </xs:all>
</xs:complexType>

<xs:complexType name="Range">
 <xs:attribute name="Minimum" type="xs:integer" use="required"/>
 <xs:attribute name="Maximum" type="xs:integer" use="required"/>
</xs:complexType>

<xs:complexType name="ProcessVersion">
 <xs:sequence>
 <xs:element name="Major" type="Range" minOccurs="1" />
 <xs:element name="Minor" type="Range" minOccurs="0" />
 <xs:element name="Build" type="Range" minOccurs="0" />
 <xs:element name="Patch" type="Range" minOccurs="0" />
 </xs:sequence>
</xs:complexType>

<xs:simpleType name="Architecture">
 <xs:restriction base="xs:string">
 <xs:enumeration value="Win32"/>
 <xs:enumeration value="Win64"/>
 </xs:restriction>
</xs:simpleType>

<xs:complexType name="Process">
 <xs:sequence>
 <xs:element name="Filename" type="FilenameString" minOccurs="1" />
 <xs:element name="Architecture" type="Architecture" minOccurs="0" />
 <xs:element name="ProductName" type="xs:string" minOccurs="0" />
 <xs:element name="FileDescription" type="xs:string" minOccurs="0" />
 <xs:element name="ProductVersion" type="ProcessVersion" minOccurs="0" maxOccurs="unbounded"/>
 <xs:element name="FileVersion" type="ProcessVersion" minOccurs="0" maxOccurs="unbounded"/>
 </xs:sequence>
</xs:complexType>

<xs:complexType name="Processes">
 <xs:sequence>
 <xs:choice minOccurs="1">
 <xs:element name="Process" type="Process" />
 <xs:element name="ShellProcess" type="Empty" />
 </xs:choice>
 <xs:element name="Process" type="Process" minOccurs="0" maxOccurs="unbounded" />
 </xs:sequence>
</xs:complexType>

<xs:complexType name="Path">
 <xs:simpleContent>
 <xs:extension base="xs:string">
 <xs:attribute name="Recursive" type="xs:boolean" default="false"/>
 <xs:attribute name="DeleteIfNotFound" type="xs:boolean" default="false"/>
 </xs:extension>
 </xs:simpleContent>
</xs:complexType>

<xs:complexType name="RegistrySetting">
 <xs:sequence>
 <xs:element name="Path" type="Path" />
 <xs:element name="Name" type="xs:string" minOccurs="0" maxOccurs="unbounded" />
 <xs:element name="Exclude" minOccurs="0" maxOccurs="unbounded">
 <xs:complexType>
 <xs:sequence>
 <xs:element name="Path" type="Path" minOccurs="0" />
 <xs:element name="Name" type="xs:string" minOccurs="0" maxOccurs="unbounded" />
 </xs:sequence>
 </xs:complexType>
 </xs:element>
 </xs:sequence>
</xs:complexType>

```

```

 </xs:sequence>
 </xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>

<xs:complexType name="FileSetting">
 <xs:sequence>

 <xs:element name="Root">
 <xs:complexType>
 <xs:choice>
 <xs:element name="KnownFolder" type="Guid" />
 <xs:element name="RegistryEntry" type="xs:string" />
 <xs:element name="EnvironmentVariable" type="xs:string" />
 </xs:choice>
 </xs:complexType>
 </xs:element>

 <xs:element name="Path" minOccurs="0" type="Path" />
 <xs:element name="FileMask" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>

 <xs:element name="Exclude" minOccurs="0" maxOccurs="unbounded">
 <xs:complexType>
 <xs:sequence>
 <xs:element name="Path" type="Path" minOccurs="0" />
 <xs:element name="FileMask" type="xs:string" minOccurs="0" maxOccurs="unbounded" />
 </xs:sequence>
 </xs:complexType>
 </xs:element>

 </xs:sequence>
</xs:complexType>

<xs:simpleType name="CustomActionSetting">
 <xs:restriction base="xs:anyURI"/>
</xs:simpleType>

<xs:simpleType name="SystemParameterSetting">
 <xs:restriction base="xs:string">

 <!-- Accessibility parameters -->
 <xs:enumeration value="AccessTimeout"/>
 <xs:enumeration value="AudioDescription"/>
 <xs:enumeration value="ClientAreaAnimation"/>
 <xs:enumeration value="DisableOverlappedContent"/>
 <xs:enumeration value="FilterKeys"/>
 <xs:enumeration value="FocusBorderHeight"/>
 <xs:enumeration value="FocusBorderWidth"/>
 <xs:enumeration value="HighContrast"/>
 <xs:enumeration value="MessageDuration"/>
 <xs:enumeration value="MouseClickLock"/>
 <xs:enumeration value="MouseClickLockTime"/>
 <xs:enumeration value="MouseKeys"/>
 <xs:enumeration value="MouseSonar"/>
 <xs:enumeration value="MouseVanish"/>
 <xs:enumeration value="ScreenReader"/>
 <xs:enumeration value="ShowSounds"/>
 <xs:enumeration value="SoundSentry"/>
 <xs:enumeration value="StickyKeys"/>
 <xs:enumeration value="ToggleKeys"/>

 <!-- Input parameters -->
 <xs:enumeration value="Beep"/>
 <xs:enumeration value="BlockSendInputResets"/>
 <xs:enumeration value="DefaultInputLang"/>
 <xs:enumeration value="DoubleClickTime"/>
 <xs:enumeration value="DoubleClkHeight"/>
 <xs:enumeration value="DoubleClkWidth"/>
 </xs:restriction>
</xs:simpleType>

```

```

<xs:enumeration value="KeyboardCues"/>
<xs:enumeration value="KeyboardDelay"/>
<xs:enumeration value="KeyboardPref"/>
<xs:enumeration value="KeyboardSpeed"/>
<xs:enumeration value="Mouse"/>
<xs:enumeration value="MouseButtonSwap"/>
<xs:enumeration value="MouseHoverHeight"/>
<xs:enumeration value="MouseHoverTime"/>
<xs:enumeration value="MouseHoverWidth"/>
<xs:enumeration value="MouseSpeed"/>
<xs:enumeration value="MouseTrails"/>
<xs:enumeration value="SnapToDefButton"/>
<xs:enumeration value="WheelScrollChars"/>
<xs:enumeration value="WheelScrollLines"/>

<!-- Desktop parameters (limited subset) -->
<xs:enumeration value="DeskWallpaper"/>
<xs:enumeration value="DesktopColor"/>

</xs:restriction>
</xs:simpleType>

<xs:complexType name="Settings">
<xs:sequence>
<xs:element name="Asynchronous" type="xs:boolean" minOccurs="0" />
<xs:element name="PreventOverlappingSynchronization" type="xs:boolean" minOccurs="0" />
<xs:element name="AlwaysApplySettings" type="xs:boolean" minOccurs="0" />
<xs:choice minOccurs="0" maxOccurs="unbounded">
<xs:element name="Registry" type="RegistrySetting" />
<xs:element name="File" type="FileSetting" />
<xs:element name="SystemParameter" type="SystemParameterSetting" />
<xs:element name="CustomAction" type="CustomActionSetting" />
</xs:choice>
</xs:sequence>
</xs:complexType>

<xs:complexType name="Common">
<xs:sequence>
<xs:element name="Name" type="xs:string" />
<xs:element name="ID" type="IDString" />
<xs:element name="ReplacedTemplates" type="ReplacedTemplates" minOccurs="0" />
<xs:element name="Description" type="xs:string" minOccurs="0" />
<xs:element name="LocalizedNames" type="LocalizedString" minOccurs="0" />
<xs:element name="LocalizedDescriptions" type="LocalizedDescription" minOccurs="0" />
<xs:element name="Version" type="xs:integer" />
<xs:element name="DeferToMSAccount" type="Empty" minOccurs="0" />
<xs:element name="DeferToOffice365" type="Empty" minOccurs="0" />
<xs:element name="Settings" type="Settings" />
</xs:sequence>
</xs:complexType>

<xs:complexType name="Application">
<xs:sequence>
<xs:element name="Name" type="xs:string" />
<xs:element name="ID" type="IDString" />
<xs:element name="ReplacedTemplates" type="ReplacedTemplates" minOccurs="0" />
<xs:element name="Description" type="xs:string" minOccurs="0" />
<xs:element name="LocalizedNames" type="LocalizedString" minOccurs="0" />
<xs:element name="LocalizedDescriptions" type="LocalizedDescription" minOccurs="0" />
<xs:element name="Version" type="xs:integer" />
<xs:element name="DeferToMSAccount" type="Empty" minOccurs="0" />
<xs:element name="DeferToOffice365" type="Empty" minOccurs="0" />
<xs:element name="Processes" type="Processes" />
<xs:element name="Settings" type="Settings" />
</xs:sequence>
</xs:complexType>

<xs:element name="SettingsLocationTemplate">

```

```

<xs:element name="SettingsLocationTemplate" type="xsd:string">
 <xs:complexType>
 <xs:sequence>
 <xs:element name="Name" type="xs:string" />
 <xs:element name="ID" type="IDString" />
 <xs:element name="Description" type="xs:string" minOccurs="0" />
 <xs:element name="LocalizedNames" type="LocalizedString" minOccurs="0" />
 <xs:element name="LocalizedDescriptions" type="LocalizedDescription" minOccurs="0" />

 <xs:choice>

 <!-- Single application -->
 <xs:sequence>
 <xs:element name="ReplacedTemplates" type="ReplacedTemplates" minOccurs="0" />
 <xs:element name="Version" type="TemplateVersion" />
 <xs:element name="Author" type="Author" minOccurs="0" />
 <xs:element name="FixedProfile" type="xs:string" minOccurs="0" />
 <xs:element name="DeferToMSAccount" type="Empty" minOccurs="0" />
 <xs:element name="DeferToOffice365" type="Empty" minOccurs="0" />
 <xs:element name="Processes" type="Processes" />
 <xs:element name="Settings" type="Settings" />
 </xs:sequence>

 <!-- Suite of applications -->
 <xs:sequence>
 <xs:element name="ManageSuiteOnly" type="xs:boolean" minOccurs="0" />
 <xs:element name="Author" type="Author" minOccurs="0" />
 <xs:element name="FixedProfile" type="xs:string" minOccurs="0" />
 <xs:element name="Common" type="Common" />
 <xs:element name="Application" type="Application" minOccurs="2" maxOccurs="unbounded" />
 </xs:sequence>
 </xs:choice>
 </xs:sequence>
 </xs:complexType>
</xs:element>
<!-- SettingsLocationTemplate -->
</xs:schema>

```

## Have a suggestion for UE-V?

Add or vote on suggestions on the [User Experience Virtualization feedback site](#).

For UE-V issues, use the [UE-V TechNet Forum](#).

## Related topics

[Working with Custom UE-V Templates and the UE-V Template Generator](#)

[Technical Reference for UE-V](#)

# Security Considerations for UE-V

6/20/2017 • 7 min to read • [Edit Online](#)

## Applies to

- Windows 10, version 1607

This topic contains a brief overview of accounts and groups, log files, and other security-related considerations for User Experience Virtualization (UE-V). For more information, follow the links that are provided here.

## Security considerations for UE-V configuration

**Important** When you create the settings storage share, limit the share access to users who require access.

Because settings packages might contain personal information, you should take care to protect them as well as possible. In general, do the following:

- Restrict the share to only those users who require access. Create a security group for users who have redirected folders on a particular share and limit access to only those users.
- When you create the share, hide the share by putting a \$ after the share name. This addition hides the share from casual browsers, and the share is not visible in My Network Places.
- Only give users the minimum amount of permissions that they must have. The following tables show the required permissions.

- Set the following share-level SMB permissions for the setting storage location folder.

USER ACCOUNT	RECOMMENDED PERMISSIONS
Everyone	No permissions
Security group of UE-V	Full control

- Set the following NTFS file system permissions for the settings storage location folder.

USER ACCOUNT	RECOMMENDED PERMISSIONS	FOLDER
Creator/Owner	No permissions	No permissions
Domain Admins	Full control	This folder, subfolders, and files
Security group of UE-V users	List folder/read data, create folders/append data	This folder only
Everyone	Remove all permissions	No permissions

- Set the following share-level SMB permissions for the settings template catalog folder.

USER ACCOUNT	RECOMMEND PERMISSIONS
Everyone	No permissions
Domain computers	Read permission Levels
Administrators	Read/write permission levels

4. Set the following NTFS permissions for the settings template catalog folder.

USER ACCOUNT	RECOMMENDED PERMISSIONS	APPLY TO
Creator/Owner	Full control	This folder, subfolders, and files
Domain Computers	List folder contents and Read permissions	This folder, subfolders, and files
Everyone	No permissions	No permissions
Administrators	Full Control	This folder, subfolders, and files

### Use Windows Server as of Windows Server 2003 to host redirected file shares

User settings package files contain personal information that is transferred between the client computer and the server that stores the settings packages. Because of this process, you should ensure that the data is protected while it travels over the network.

User settings data is vulnerable to these potential threats: interception of the data as it passes over the network, tampering with the data as it passes over the network, and spoofing of the server that hosts the data.

As of Windows Server 2003, several features of the Windows Server operating system can help secure user data:

- **Kerberos** - Kerberos is standard on all versions of Microsoft Windows 2000 Server and Windows Server beginning with Windows Server 2003. Kerberos ensures the highest level of security to network resources. NTLM authenticates the client only; Kerberos authenticates the server and the client. When NTLM is used, the client does not know whether the server is valid. This difference is particularly important if the client exchanges personal files with the server, as is the case with Roaming User Profiles. Kerberos provides better security than NTLM. Kerberos is not available on the Microsoft Windows NT Server 4.0 or earlier operating systems.
- **IPsec** - The IP Security Protocol (IPsec) provides network-level authentication, data integrity, and encryption. IPsec ensures the following:
  - Roamed data is safe from data modification while data is en route.
  - Roamed data is safe from interception, viewing, or copying.
  - Roamed data is safe from access by unauthenticated parties.
- **SMB Signing** - The Server Message Block (SMB) authentication protocol supports message authentication, which prevents active message and "man-in-the-middle" attacks. SMB signing provides this authentication by placing a digital signature into each SMB. The digital signature is then verified by both the client and the

server. In order to use SMB signing, you must first either enable it, or you must require it on both the SMB client and the SMB server. Note that the SMB signing imposes a performance penalty. It does not consume any more network bandwidth, but it uses more CPU cycles on the client and server side.

### **Always use the NTFS file system for volumes that hold user data**

For the most secure configuration, configure servers that host the UE-V settings files to use the NTFS file system. Unlike the FAT file system, NTFS supports Discretionary access control lists (DACLs) and system access control lists (SACLs). DACLs and SACLs control who can perform operations on a file and what events trigger the logging of actions that is performed on a file.

### **Do not rely on EFS to encrypt user files when they are transmitted over the network**

When you use the Encrypting File System (EFS) to encrypt files on a remote server, the encrypted data is not encrypted during transit over the network; it only becomes encrypted when it is stored on disk.

This encryption process does not apply when your system includes Internet Protocol security (IPsec) or Web Distributed Authoring and Versioning (WebDAV). IPsec encrypts data while it is transported over a TCP/IP network. If the file is encrypted before it is copied or moved to a WebDAV folder on a server, it remains encrypted during the transmission and while it is stored on the server.

### **Let the UE-V service create folders for each user**

To ensure that UE-V works optimally, create only the root share on the server, and let the UE-V service create the folders for each user. UE-V creates these user folders with the appropriate security.

This permission configuration enables users to create folders for settings storage. The UE-V service creates and secures a settings package folder while it runs in the context of the user. Users receive full control to their settings package folder. Other users do not inherit access to this folder. You do not have to create and secure individual user directories. The UE-V service that runs in the context of the user does it automatically.

**Note** Additional security can be configured when a Windows Server is used for the settings storage share. UE-V can be configured to verify that either the local Administrators group or the current user is the owner of the folder where settings packages are stored. To enable additional security, use the following command:

1. Add the REG\_DWORD registry key RepositoryOwnerCheckEnabled to

`HKEY_LOCAL_MACHINE\Software\Microsoft\UEV\Agent\Configuration`

2. Set the registry key value to 1.

When this configuration setting is in place, the UE-V service verifies that the local Administrators group or current user is the owner of the settings package folder. If not, then the UE-V service does not grant access to the folder.

If you must create folders for the users, ensure that you have the correct permissions set.

We strongly recommend that you do not pre-create folders. Instead, let the UE-V service create the folder for the user.

### **Ensure correct permissions to store UE-V 2 settings in a home directory or custom directory**

If you redirect UE-V settings to a user's home directory or a custom Active Directory (AD) directory, ensure that the permissions on the directory are set appropriately for your organization.

### **Review the contents of settings location templates and control access to them as needed**

When creating a settings location template, the UE-V generator uses a Lightweight Directory Access Protocol (LDAP) query to get username and email address of the current logged in user. This information is stored in the template as the template author name and template author email. (None of this information is sent to Microsoft.)

If you plan to share settings location templates with anyone outside your organization you should review all the settings locations and ensure the settings location templates do not contain any personal or company information.

You can view the contents by opening the settings location template files using any XML viewer. The following are ways you can view and remove any personal or company information from the settings location template files before sharing with anyone outside your company:

- **Template Author Name** – Specify a general, non-identifying name for the template author name or exclude this data from the template.
- **Template Author Email** – Specify a general, non-identifying template author email or exclude this data from the template.

To remove the template author name or template author email, you can use the UE-V generator application. From the generator, select **Edit a Settings Location Template**. Select the settings location template to edit from the recently used templates or Browse to the settings template file. Select **Next** to continue. On the Properties page, remove the data from the Template author name or Template author email text fields. Save the settings location template.

## Have a suggestion for UE-V?

Add or vote on suggestions on the [User Experience Virtualization feedback site](#).

For UE-V issues, use the [UE-V TechNet Forum](#).

## Related topics

[Technical Reference for UE-V](#)

# Change history for Configure Windows 10

11/8/2017 • 2 min to read • [Edit Online](#)

This topic lists new and updated topics in the [Configure Windows 10 documentation](#) for Windows 10 and Windows 10 Mobile.

## November 2017

NEW OR CHANGED TOPIC	DESCRIPTION
<a href="#">Create a provisioning package with multivariant settings</a>	Add support for desktop to <a href="#">Conditions</a> table.

## October 2017

NEW OR CHANGED TOPIC	DESCRIPTION
<a href="#">Guidelines for choosing an app for assigned access</a>	Added that Microsoft Edge is not supported for assigned access

## RELEASE: Windows 10, version 1709

The topics in this library have been updated for Windows 10, version 1709 (also known as the Fall Creators Update). The following new topics have been added:

- [Create a Windows 10 kiosk that runs multiple apps](#)
- [Multi-app kiosk XML reference](#)
- [Windows 10, version 1709 basic diagnostic events and fields](#)
- [Windows 10, version 1709 enhanced telemetry events and fields used by Windows Analytics](#)

## September 2017

NEW OR CHANGED TOPIC	DESCRIPTION
<a href="#">Beginning your General Data Protection Regulation (GDPR) journey for Windows 10</a>	New conceptual info about Windows 10 and the upcoming GDPR-compliance requirements.
<a href="#">Manage connections from Windows operating system components to Microsoft services</a>	Added that Windows Spotlight can be managed by the Experience/AllowWindowsSpotlight MDM policy.

## August 2017

NEW OR CHANGED TOPIC	DESCRIPTION
<a href="#">Windows Configuration Designer provisioning settings (reference)</a>	New section; reference content from <a href="#">Windows Provisioning settings reference</a> is being relocated here from MSDN.

## July 2017

NEW OR CHANGED TOPIC	DESCRIPTION
<a href="#">Windows 10, version 1703 Diagnostic Data</a>	Updated categories and included diagnostic data.
<a href="#">Add image for secondary tiles</a>	Added XML example for Edge secondary tiles and <b>ImportEdgeAssets</b>
<a href="#">Customize and export Start layout</a>	Added explanation for tile behavior when the app is not installed
<a href="#">Guidelines for choosing an app for assigned access</a>	Added that Microsoft Edge is not supported for assigned access
<a href="#">Windows 10, version 1703 basic level Windows diagnostic events and fields</a>	Updated several Appraiser events and added Census.Speech.
<a href="#">Manage connections from Windows operating system components to Microsoft-services</a>	Updated Date & Time and Windows spotlight sections.

## June 2017

NEW OR CHANGED TOPIC	DESCRIPTION
<a href="#">Guidelines for choosing an app for assigned access</a>	Added guidelines for using Remote Desktop app as the kiosk app and added a general guideline that apps generated using the Desktop App Converter cannot be used for kiosk apps
<a href="#">Set up a kiosk on Windows 10 Pro, Enterprise, or Education</a>	Added warning about using Shell Launcher to set a custom shell with an application that launches a different process and then exits
<a href="#">Windows Configuration Designer command-line interface (reference)</a>	Removed references to imaging

## May 2017

NEW OR CHANGED TOPIC	DESCRIPTION
<a href="#">Configure cellular settings for tablets and PCs</a>	New
<a href="#">Manage connections from Windows operating system components to Microsoft services</a>	Added MDM policies for privacy settings

## April 2017

NEW OR CHANGED TOPIC	DESCRIPTION
<a href="#">Set up a shared or guest PC with Windows 10</a>	Added instructions for using WMI bridge to configure shared PC

## RELEASE: Windows 10, version 1703

The topics in this library have been updated for Windows 10, version 1703 (also known as the Creators Update).

The following new topics have been added:

- [Use the Lockdown Designer app to create a Lockdown XML file](#)
- [Add image for secondary tiles](#)
- [Provision PCs with apps](#)
- [Windows 10, version 1703 basic level Windows diagnostic events and fields](#)
- [Windows 10, version 1703 Diagnostic Data](#)