Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

**Module**

**8**

**Windows 10 Application Support**

* Chapter Introduction
* **8-1**Application Environments
	+ **8-1a**Windows API
	+ **8-1b**.NET Framework
	+ **8-1c**Universal Windows Platform
	+ **8-1d**Legacy Applications
* **8-2**The Registry
	+ **8-2a**Registry Structure
	+ **8-2b**Registry Editing Tools
* **8-3**Installing Apps
	+ **8-3a**Automating MSI Installation
	+ **8-3b**UWP Apps
	+ **8-3c**MSIX Deployment
	+ **8-3d**Windows 10 in S Mode
	+ **8-3e**Microsoft 365 Apps Click-to-Run Deployment
* **8-4**App Compatibility
	+ **8-4a**Compatibility Settings for Executables
	+ **8-4b**Windows ADK Tools
	+ **8-4c**Desktop Analytics
	+ **8-4d**Client Hyper-V
	+ **8-4e**Virtual Desktop Infrastructure and RemoteApp
	+ **8-4f**App-V
* **8-5**Remote Desktop Services
	+ **8-5a**RemoteApp
	+ **8-5b**Accessing Virtual Desktops and RemoteApp
	+ **8-5c**Windows Virtual Desktop
* **8-6**Chapter Review
	+ **8-6a**Summary
	+ **8-6b**Key Terms
	+ **8-6c**Review Questions
	+ **8-6d**Case Projects

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

Chapter Introduction

After reading this module and completing the exercises, you will be able to:

* **1**Describe the application environments supported by Windows 10
* **2**Modify and back up the registry
* **3**Install various types of apps
* **4**Describe and implement options for app compatibility
* **5**Configure clients for virtual desktops and RemoteApp

As a technology support worker, it is easy to lose track of the real purpose of computers. The purpose of computers is to run applications that allow workers to be more productive. Installing and troubleshooting those applications is core to the role of desktop support.

In this module, you review the different application environments available in Windows 10, which in turn identify the types of apps supported by Windows 10. You also learn about the structure of the registry and how to edit it. Then, you learn about different installation processes for apps, including automated deployment for larger organizations. You also identify the different options for mitigating app compatibility problems when migrating to Windows 10. Finally, you learn about how to access virtual desktops and RemoteApp programs when Remote Desktop Services has been configured.

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

**8-1**Application Environments

Apps are written by developers to interact with a specific **application environment subsystem**. The application environment subsystem provides access to operating system services through an **application programming interface (API)**. The application environment subsystem is then responsible for communicating with the operating system to accomplish tasks, such as communicating on the network or saving files to disk.

To install and use an app, the application environment it is written for must be present. If the application environment is not present, the APIs necessary for the app to request operating system functions are not available. Some application environments have multiple versions, and you need to ensure that the correct version is present for apps. You might need to install Windows Updates to ensure you have the most recent version of the desired application environment.

All the application environments in Windows 10 restrict apps to running in user mode where they are isolated from the core functionality of the operating system. Running apps in user mode ensures that a poorly written app does not affect system stability. This compares with hardware drivers that operate in kernel mode where a poorly written driver can impact system stability.

**Note 1**

For an overview of applications environments oriented at programmers, see Choose your app platform at [https://docs.microsoft.com/en-us/windows/apps/desktop/choose-your-platform](https://docs.microsoft.com/en-us/windows/apps/desktop/choose-your-platform%22%20%5Ct%20%22_blank).

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

## 8-1aWindows API

The **Windows API** provides a way for apps to request services and functionality from Windows 10. Apps written in various programming languages can use this API. The modern variation of the Windows API is Win64. Win64 is implemented on 64-bit versions of Windows 10. In 32-bit versions of Windows, the Win32 version of the Windows API is implemented.

**Tip**

Sometimes the Windows API is generically referred to as Win32 even on 64-bit versions of Windows.

Most organizations use the 64-bit version of Windows 10 rather than the 32-bit version of Windows 10. Both versions support Win32, but the 64-bit version of Windows 10 supports 32-bit applications by using a **Windows on Windows 64 (WOW64)** virtualized environment to host Win32 apps. The calls to Win32 APIs are translated to an equivalent Win64 API call in the 64-bit version of Windows 10 to provide compatibility.

It is common for organizations to have a mix of 32-bit and 64-bit applications. On a 64-bit version of Windows 10, the apps installed to C:\Program Files (x86)\ are 32-bit applications.

**Activity 8-1**

### Identifying Win32 and Win64 Apps

**Time Required:**10 minutes

**Objective:**Run Win32 and Win64 apps and review how they appear in Task Manager

**Description:**The core apps included in the 64-bit version of Windows 10 are all 64-bit applications; however, 32-bit versions of some system apps are included for backward compatibility for some programs. In this activity, you run a Win32 app and a Win64 app to confirm how they are individually presented on the Processes tab in Task Manager.

1. 1

If necessary, start your computer and sign in.

1. 2

Click the **Start** button, type **notepad**, and then click **Notepad**.

1. 3

On the taskbar, click **File Explorer**.

1. 4

In the File Explorer window, browse to **C:\Windows\SysWOW64** and then double-click **notepad**.

1. 5

Compare the two Notepad windows. Notice that no visual difference is apparent between the two Notepad windows.

1. 6

Right-click the taskbar and then click **Task Manager**.

1. 7

If Task Manager is in the simplified view, click **More details**.

1. 8

On the Processes tab, if necessary, click the Name column to sort by name, and notice that two Notepad apps are listed. One of the Notepad apps has (32 bit) appended to the name. The text (32 bit) identifies Win32 processes.

1. 9

Right-click **Notepad (32 bit)** and then click **Properties**.

1. 10

In the notepad Properties window, notice that the location for this app is C:\Windows\SysWOW64 and then click **Cancel**.

1. 11

Close all open windows.

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

## 8-1b.NET Framework

The **.NET Framework** is a commonly used application environment for apps on Windows 10. You can identify whether the .NET Framework is required for an app from the documentation for the app. Most applications that require the .NET Framework verify that the correct version of .NET Framework is available during installation. Some apps even distribute the .NET Framework during installation if required.

Multiple versions of the .NET Framework exist. At the time of this writing, the current version is .NET Framework 4.8. In theory, the latest version of .NET Framework is compatible with previous versions, but the reality is that some apps require specific versions. In general, .NET Framework 3.5 supports apps written for .NET versions 2.0, 3.0, and 3.5. Apps requiring .NET 4.0 or later are generally compatible with the latest version of the .NET Framework. Windows 10 provides .NET Framework 3.5 and .NET Framework 4.8 as features.

Updates for the .NET Framework are delivered through Windows Update. If your organization has .NET applications, you should test any new .NET Framework updates before deploying them to user computers. Updates for the .NET Framework are issued regularly.

**Tip**

For instructions on how to identify which version of the .NET Framework is installed, see How to: Determine which .NET Framework versions are installed at [https://docs.microsoft.com/en-us/dotnet/framework/migration-guide/how-to-determine-which-versions-are-installed](https://docs.microsoft.com/en-us/dotnet/framework/migration-guide/how-to-determine-which-versions-are-installed%22%20%5Ct%20%22_blank).

**Activity 8-2**

### Installing .NET Framework 3.5

**Time Required:**10 minutes

**Objective:**Install .NET Framework 3.5 on Windows 10

**Description:**Some applications require an older version of .NET Framework to be installed. In Windows 10, .NET Framework 3.5 is available for backward compatibility. In this activity, you install the .NET Framework 3.5 feature on Windows 10.

1. 1

If necessary, start your computer and sign in.

1. 2

Click the **Start** button, type **control**, and then click **Control Panel**.

1. 3

In the Control Panel window, click **Programs** and then click **Turn Windows features on or off**.

1. 4

In the Windows Features dialog box, notice that .NET Framework 4.8 Advanced Services is already installed by default.

1. 5

Select the **.NET Framework 3.5 (includes .NET 2.0 and 3.0)** check box and then click **OK**.

1. 6

In the Windows Features window, click **Let Windows Update download the files for you**. The installation files are not included with Windows 10. In most cases, the file download takes only a few minutes.

1. 7

When Windows has completed the requested changes, click **Close** and then close the Control Panel window.

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

## 8-1cUniversal Windows Platform

To make application development easier across multiple types of devices, Microsoft created **Universal Windows Platform (UWP)**. UWP apps are designed with a more modern interface that is flexible enough to accommodate multiple form factors, like a desktop computer and tablets. These apps have been known as Metro apps, Modern apps, Windows Store apps, Microsoft Store apps, and UWP apps.

Many of the apps included in Windows 10 are UWP apps. For example, the Calculator app is a UWP app. A defining characteristic of UWP apps is that they are distributed and updated from the Microsoft Store. This enhances app security because Microsoft verifies that the app does not contain malware and meets development standards.

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

## 8-1dLegacy Applications

If you are running a 64-bit version of Windows 10, you cannot run DOS or Win16 applications. DOS was the command-line precursor to Windows, and Win16 was used for apps in Windows 3.1. Because both these platforms were popular in the 1980s and early 1990s, very few apps in production were designed for DOS or Win16.

If you do have a legacy DOS or Win16 application that you need to run, you can use a 32-bit version of Windows 10. To support Win16 applications, a 32-bit version of Windows 10 translates Win16 API calls to Win32 API calls in the same way that 64-bit versions of Windows 10 support 32-bit applications. DOS applications are supported by a virtual instance of DOS (Ntvdm.exe).

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

**8-2**The Registry

Windows 10 has the **registry** as a central store for application and operating system configuration information. Applications and Windows 10 read their configuration information from the registry during startup and when running. When you make configuration changes to applications or Windows 10, those changes are stored in the registry so that they can be retrieved later.

In most cases, you don’t need to manually edit or view the registry. Applications and Windows 10 properly understand the structure of the registry and how their own information is stored in it. Sometimes, however, you will find support documents that identify registry keys that need to be verified or modified. In those cases, you need to understand the structure of the registry and how to modify it.

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

## 8-2aRegistry Structure

The registry is divided into sections and levels of data. Multiple sections exist to organize data by purpose. The individual sections are called hives. Each **hive** has a specific role to play and is stored in memory while it is in use. When the computer is shut down, the memory versions of the hives are written to files and folders typically found in the C:\Windows\System32\config folder. Each hive is composed of one or more files.

Within a single hive, the data is stored in keys and values identified by their name and position relative to one another. Figure 8-1 shows an example of the registry structure when viewed with a registry editing tool.

**Figure 8-1View of the Registry Structure**





The left navigation pane displays a hierarchical folder structure. Each hive appears as a top-level folder in the left pane. In the hives, each folder in the left pane is referred to as a **registry key**. Each registry key is identified by the hive it belongs to, its position relative to other keys in the hive, and its name.

The right pane shows the data values that are stored at a specific level in the registry hierarchy (i.e., within a registry key). Each registry key can store multiple data values. The data values are defined by a name that is case sensitive, a type indicating how the data is formatted (e.g., binary, string, word), and the actual data stored by the value.

**Tip**

In documentation, it is common for the registry hive names to be abbreviated. For example, HKEY\_LOCAL\_MACHINE is often referred to as HKLM. Also, HKEY\_CURRENT\_USER is often referred to as HKCU.

To aid in troubleshooting and understanding why changes might be requested by a support document, it is useful to understand the contents of each registry hive. Table 8-1 describes the contents of each registry hive.

**Table 8-1**

### Registry Hives

| **Hive Name** | **Description** |
| --- | --- |
| HKEY\_CLASSES\_ROOT | This hive defines file types (classes) and properties associated with those types. For example, file type associations are stored here. This hive is a combination of HKCU\Software\Classes and HKLM\Software\Classes. If settings for a file type are defined in both locations, the user-specific settings take precedence. |
| HKEY\_CURRENT\_USER | This hive contains the user-specific registry information from ntuser.dat stored in the user profile. Any application or operating system settings that are user specific are stored in this hive. If you are attempting to fix a user-specific application issue, the support document will have you review and modify keys here. |
| HKEY\_LOCAL\_MACHINE | This hive contains global settings for the entire computer and the applications installed on it. If you are attempting to fix any operating system or application issue that is not user specific, the support document will have you review and modify keys here. More details about HKLM are provided later in this module. |
| HKEY\_USERS | This hive contains the user-specific settings for the current user and several system services. The name of the registry key for the currently signed-in user is the security identifier (SID) of the user. The content of the key for the currently signed-in user is also available in HKCU. |
| HKEY\_CURRENT\_CONFIG | This hive contains details about the current hardware profile in use. The details report the differences between the standard configuration defined in HKLM\System and HKLM\Software and those in the active hardware profile. This hive is also a mapped view to information stored in HKLM\System\CurrentControlSet\Hardware Profiles\Current. |

### HKEY\_USERS

The HKEY\_USERS hive contains several registry keys, as shown in Figure 8-2. The .DEFAULT key contains the default registry settings that are copied for new user profiles. The shorter registry keys, such as S-1-5-18, are for system services. The registry keys with a full SID are for the currently signed-in user.

**Figure 8-2HKEY\_USERS Hive**



### HKEY\_LOCAL\_MACHINE

This hive contains important settings for Windows 10 and applications, as well as all of the general settings. Some of the important keys in this hive are as follows:

* BCD00000000—This key contains information from the boot configuration database that defined the Windows 10 boot process. Instead of editing these keys, you should use the bcdedit.exe utility.
* HARDWARE—This key contains hardware information that is detected at startup. You should not edit this information.
* SOFTWARE—This key contains information for applications. Application data is typically located in a key with the following naming structure: HKLM\SOFTWARE\vendor\application\.
* SYSTEM—This key contains information about Windows 10 and hardware drivers. Device driver and service information is in HKLM\SYSTEM\CurrentControlSet. A backup copy of this key named ControlSet001 is updated each time a user signs in. This backup can be used by Windows 10 during a recovery.

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

## 8-2bRegistry Editing Tools

The preferred method for configuring applications and Windows 10 is to use the interfaces provided for that purpose. Applications typically have settings or options that you can use to configure the application. Windows 10 has Settings, Control Panel, and various Microsoft Management Console (MMC) snap-ins.

If you do need to view and modify the registry entries, take the following precautions:

* Back up the portion of the registry you will be changing before you make any changes.
* Restrict the number of changes made at one time to limit the impact, and identify which change actually fixed the problem.
* When possible, use a test system rather than a user’s computer to verify that changes resolve the issue.
* If you are adjusting registry entries for services or drivers, ensure that the computer can boot properly after the changes have been made.

### Registry Editor

The most commonly used tool for viewing and modifying the registry is the graphical **Registry Editor** (regedit). In addition to basic editing functionality, you can use this tool to search the registry and modify permissions. You can also export and import sections of the registry. The Registry Editor is shown in Figure 8-1.

When you export registry keys for a backup, it is done to a .reg file. If you want to restore the contents of the .reg file, you can import it by using the Registry Editor; however, you can also double-click the file in File Explorer to restore the registry keys.

**Activity 8-3**

### Using Regedit to Back Up and Modify the Registry

**Time Required:**10 minutes

**Objective:**Use regedit to view and change registry information

**Description:**Startup information for services is stored in the registry. Although you should normally edit service startup information by using the Services administrative tool, you can also edit directly in the registry. In this activity, you modify the startup setting for a server by using regedit.

1. 1

If necessary, start your computer and sign in.

1. 2

Click the **Start** button, type **services**, and then click **Services**.

1. 3

Scroll down and then double-click the **Print Spooler** service.

1. 4

In the Print Spooler Properties (Local Computer) dialog box, read the information for the service and then click **Cancel**. Notice that the Startup type is Automatic.

1. 5

Click the **Start** button, type **regedit**, and then click **Registry Editor**.

1. 6

In the User Account Control dialog box, click **Yes**.

1. 7

In the Registry Editor, navigate to HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Services\Spooler.

1. 8

Right-click **Spooler** and then click **Export**.

1. 9

In the Export Registry File window, click **This PC** and then double-click **Local Disk (C:)**.

1. 10

In the File name box, type **SpoolerBak** and then click **Save**.

1. 11

In the Registry Editor, double-click **Start**.

1. 12

In the Edit DWORD (32-bit) Value dialog box, in the Value data box, type **4** and then click **OK**.

1. 13

In the Services window, press F5 to refresh the view, verify that the value in the Startup Type column for the Print Spooler service is now Disabled, and then close the Services window.

1. 14

In the Registry Editor, collapse all the hives so that no keys are visible and then close the Registry Editor window.

**Activity 8-4**

### Restoring a Registry Backup

**Time Required:**10 minutes

**Objective:**Restore registry settings from a .reg file

**Description:**Before modifying the registry, you should export any keys that you will be changing to a .reg file. If the registry change does not go as planned, you can import the .reg file to restore the previous level of functionality. In this activity, you restore a .reg file and verify that the contents imported properly.

1. 1

If necessary, start your computer and sign in.

1. 2

On the taskbar, click **File Explorer**, navigate to **C:\**, and then double-click **SpoolerBak**.

1. 3

In the User Account Control dialog box, click **Yes**. Notice that the Registry Editor is being started.

1. 4

In the Registry Editor window, click **Yes** to continue.

1. 5

Click **OK** to acknowledge the successful import and then close the File Explorer window.

1. 6

Click the **Start** button, type **regedit**, and then click **Registry Editor**.

1. 7

In the User Account Control dialog box, click **Yes**.

1. 8

In the Registry Editor, expand **HKEY\_LOCAL\_MACHINE** and then click **SYSTEM**.

1. 9

Click **Edit** on the menu bar and then click **Find**.

1. 10

In the Find dialog box, in the Find what box, type **spooler**.

1. 11

In the Look at area, deselect all check boxes except **Keys** and then click **Find Next**.

1. 12

Keep pressing **F3** to Find Next until the Print Spooler service identified in Activity 8-3 is located. This will take about 5–10 presses, but review the information that is found along the way.

1. 13

Read the Start value and verify that it has been set back to 2.

1. 14

Click the **Start** button, type **services**, and then click **Services**.

1. 15

Scroll down to the Print Spooler service and verify that the Startup Type is Automatic.

1. 16

Close all open windows.

### Reg.exe

If you need to read or write registry entries from batch files, you can use reg.exe, as shown in Figure 8-3. Reg.exe is a command-line utility with similar functionality to regedit. You may see some support documents that advise you to use reg.exe when modifying the registry. This is because it is easier to type out a single command in support documentation than to provide multiple steps to perform the task using a graphical interface. Regardless of the registry editing tool used, the end result is the same.

**Figure 8-3Reg.exe Command-Line Utility**





**Activity 8-5**

### Using Reg.exe

**Time Required:**10 minutes

**Objective:**Use reg.exe to view and modify registry entries

**Description:**If you need to query or modify the registry from a command-line or batch file, you can use reg.exe. In this activity, you view and modify registry entries by using reg.exe.

1. 1

If necessary, start your computer and sign in.

1. 2

Right-click the **Start** button, click **Windows PowerShell (Admin)**, and then click **Yes** in the User Account Control dialog box.

1. 3

At the Windows PowerShell prompt, type **reg /?** and then press **Enter** to view the general help information.

1. 4

Type **reg query /?** and then press **Enter** to view the query help information. Scroll up and down to read all of it.

1. 5

Type **reg query HKLM\SYSTEM\CurrentControlSet\Services\Spooler** and then press **Enter**. Verify that the Start value is 0×2.

1. 6

Type **reg add /?** and then press **Enter** to view the add help information. Scroll up and down to read all of it.

1. 7

Type **reg add HKLM\SYSTEM\CurrentControlSet\Services\Spooler /v Start /d 0×4** and then press **Enter**. This sets the Start value to 0×4.

1. 8

Press **Y** and then press **Enter** to confirm overwriting the value.

1. 9

Type **reg query HKLM\SYSTEM\CurrentControlSet\Services\Spooler** and then press **Enter**. Verify that the Start value is 0×4.

1. 10

Type **reg import /?** and then press **Enter** to view the import help information.

1. 11

Type **reg import C:\SpoolerBak.reg** and then press **Enter**.

1. 12

Type **reg query HKLM\SYSTEM\CurrentControlSet\Services\Spooler** and then press **Enter**. Verify that the Start value is 0x2 as it was restored from the backup.

1. 13

Close the Windows PowerShell prompt window.

### Windows PowerShell

In Windows 10, you can also use Windows PowerShell to view and modify registry values. This is important, as administrators are more likely to create new scripts in Windows PowerShell than to create batch files and use reg.exe.

In Windows PowerShell, the registry is accessible in a similar fashion to the file system. Both HKCU and HKLM are configured as drives. You can navigate through these drives by using the cd and dir commands. These, however, are actually aliases to the PowerShell cmdlets Set-Location and Get-ChildItem.

**Caution**

Windows PowerShell doesn’t include any built-in functionality to export or import a section of the registry to a .reg file. It can be done by using a script, but it’s much easier to use reg.exe or the Registry Editor.

**Activity 8-6**

### Viewing the Registry by Using Windows PowerShell

**Time Required:**10 minutes

**Objective:**Use Windows PowerShell to view and modify registry entries

**Description:**If you need to query or modify the registry from a command-line or batch file, you can use Windows PowerShell. In this activity, you view and modify registry entries by using Windows PowerShell.

1. 1

If necessary, start your computer and sign in.

1. 2

Right-click the **Start** button and then click **Windows PowerShell (Admin)**.

1. 3

In the User Account Control dialog box, click **Yes**.

1. 4

At the Windows PowerShell prompt, type **Get-PSDrive** and then press **Enter**. You can see that HKCU and HKLM are available.

1. 5

Type **Set-Location HKLM:** and then press **Enter**. Notice that the prompt has changed to indicate you are in HKLM.

1. 6

Type **Get-ChildItem** and then press **Enter**. You can see the same registry keys as in regedit.

1. 7

Type **Set-Location SYSTEM\CurrentControlSet\Services\Spooler** and then press **Enter**.

1. 8

Type **Get-ItemProperty.** and then press **Enter**. The period in this command represents the current folder of Spooler. Notice that Start has a value of 2.

1. 9

Type **Set-ItemProperty. -Name Start -Value 4** and then press **Enter**.

1. 10

Type **Get-ItemProperty.** and then press **Enter**. Verify that Start has a value of 4.

1. 11

Type **reg import C:\SpoolerBak.reg** and then press **Enter**.

1. 12

Close the Windows PowerShell prompt window.

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

**8-3**Installing Apps

Apps can be packaged for deployment in multiple ways. The most common way traditional Win32 or .NET Framework apps are packaged is in an **MSI file**. This type of file is read by the Windows Installer service to perform the installation. The MSI file has all the files for the app and instructions for file locations and registry keys.

For app developers, the Windows Installer service takes care of the details of app installation, repair, and removal. All the developers need to do is package the application properly.

**Tip**

Even though many apps have a setup.exe file, typically that setup.exe just starts the installation by using an MSI file.

Some app developers do not use MSI files for their apps. Typically, you see this for small developers that want to execute very simple deployment of their apps. These apps often lack repair functionality once they are installed. The installation and uninstallation are handled completely by the setup.exe file.

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

## 8-3aAutomating MSI Installation

In a small organization, you can go to each computer and install apps manually. This process is typically faster than figuring out how to automate the process. In a larger organization, however, it is worth the time to identify how apps can be automatically deployed to computers. Even if it takes only 5 minutes to install an app, 5 minutes for each of 1000 computers is a long time.

The simplest automated deployment method is to create a batch file that does a silent install of the application. Many applications have a /q or /quiet option for the setup.exe file. You can configure the command to run setup.exe /quiet in a sign-in script. This is not very sophisticated, but it can work for some simple scenarios.

The msiexec.exe command-line utility, shown in Figure 8-4, also can be used with MSI files in a script. When you run msiexec.exe, you can specify a /quiet option to suppress prompting users for input. If the app requires some input during installation, you can provide a transform file (.mst) that provides the additional information necessary for the app to install.

**Figure 8-4Msiexec.exe Command-Line Utility**





A more manageable way to deploy MSI-based applications in smaller environments is by using Group Policy. In Group Policy, you can deploy MSI-based apps to users or computers. If you deploy apps to users, the apps get installed on each computer that they use. If you deploy apps to computers, they are available to all users on that computer. Figure 8-5 shows the Group Policy settings for application deployment to a computer.

**Figure 8-5Group Policy Settings for Software Deployment**





Apps installed by Group Policy do not prompt the user for input. So, if the app typically requires user input, you need to create a transform file to provide the necessary information or repackage the app with the information already provided.

Apps can install even if the signed-in user does not have administrative permissions. This is because the Windows Installer service does the work, and it has Local System privileges.

When you deploy software by using Group Policy, you can update it afterwards by applying an MSP file to the application. Or, you can create a new version of the app and replace the existing version for upgrades.

In large environments, a better option for software deployment is Microsoft Endpoint Configuration Manager. Microsoft Endpoint Configuration Manager has many other functions, but it also performs software deployment. It can deploy both MSI-based apps and newer UWP and MSIX apps. Like Group Policy, it can include transform files and perform updates based on version; however, Microsoft Endpoint Configuration Manager is much more flexible in selecting the users and computers for which the apps are deployed. It also has centralized reporting about app deployment and can even monitor licensing. You can also use Microsoft Intune to manage Azure AD-joined devices.

**Activity 8-7**

### Installing and Removing an MSI-Based App

**Time Required:**10 minutes

**Objective:**Install and remove an MSI-based app by using msiexec.exe

**Description:**You can use msiexec.exe to silently install MSI-based applications. In this activity, you download an MSI-based app, install the app, and also remove it.

1. 1

If necessary, start your computer and sign in.

1. 2

On the taskbar, click **Microsoft Edge**.

1. 3

In Microsoft Edge, in the address bar, type [https://www.7-zip.org/download.html](https://www.7-zip.org/download.html%22%20%5Ct%20%22_blank) and then press **Enter**.

1. 4

On the Download page, in the first table, in the row with Type of .msi and Windows 64-bit x64, click **Download** and then click **Save**.

1. 5

When the download is complete, click **Open folder**.

1. 6

In the File Explorer window, double-click the file you downloaded. This starts an installation with the standard user interface.

1. 7

Click **Cancel** to stop the installation, click **Yes** to confirm, and then click **Finish**.

1. 8

Close the File Explorer window.

1. 9

Click the **Start** button, type **cmd**, and then click **Run as administrator** below Command Prompt. It is preferable to use a command prompt for this task because the syntax is not interpreted properly by a Windows PowerShell prompt.

1. 10

In the User Account Control dialog box, click **Yes**.

1. 11

At the command prompt, to change to the downloads folder, type **cd \users\user1\downloads** and then press Enter. If necessary, substitute user1 with the name of your user account.

1. 12

To view the MSI files in the current directory, type **dir \*.msi** and then press **Enter**. Verify that the file you just downloaded is listed.

1. 13

To view the options for misexec.exe, type **msiexec.exe /?** and then press **Enter**.

1. 14

In the Windows Installer dialog box, scroll down, read the available options, and then click **OK**.

1. 15

At the command prompt, to install 7-Zip silently, type **msiexec.exe /i 7z1900-x64.msi INSTALLDIR="C:\Program Files\7-Zip" /quiet** and then press **Enter**. The INSTALLDIR variable could also be set by using a transform file. If necessary, substitute 7z1900-x64.msi with the name of the file you downloaded.

1. 16

Click the **Start** button and then click **Settings**.

1. 17

In the Settings window, click **Apps** and then verify that 7-Zip is in the list of installed apps.

1. 18

Close the Settings window.

1. 19

At the command prompt, to remove 7-zip silently, type **msiexec.exe /x 7z1900-x64.msi /quiet** and then press **Enter.** If necessary, substitute 7z1900-x64.msi with the name of the file you downloaded.

1. 20

Click the **Start** button and then click **Settings**.

1. 21

In the Settings window, click **Apps** and then verify that 7-Zip is not in the list of installed apps.

1. 22

Close all open windows.

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

## 8-3bUWP Apps

UWP apps are designed for distribution through the Microsoft Store. These apps are not designed to be installed manually in the same way traditional apps are. No option to run setup.exe or distribute a UWP app by using Group Policy exists.

If the app you want is in the Microsoft Store, users can install it from the Microsoft Store. It is not very manageable, however, to instruct users to obtain apps directly from the Microsoft Store. If you have obtained the **APPX file** for a UWP app, you can sideload the app; that is, you can install an app from a source other than the Windows Store, such as your workplace.

You can manually sideload Windows Store apps much more easily in Windows 10 than you could in Windows 8.1. You can allow sideloading of apps on the Settings > Update & Security > For developers screen, as shown in Figure 8-6.

**Figure 8-6Setting to Enable Sideloading**





After sideloading is enabled, the certificate used to sign the UWP app must be trusted by the Windows 10 computer. If the app was signed by using a certificate from a public certification authority, no configuration is required. If the app was signed by using a self-signed certificate, that certificate needs to be imported on the computer running Windows 10 as a trusted root certification authority. Finally, you can install the app by using the Add-AppPackage cmdlet. To automate the deployment of UWP apps to computers running Windows, you can use Microsoft Endpoint Manager or Microsoft Intune.

**Tip**

Older documentation for installing APPX apps refers to using the Add-AppxPackage cmdlet. This is equivalent to the Add-AppPackage cmdlet.

Microsoft has also created a **Microsoft Store for Business**. Each business has its own private portal with approved apps. This makes it much easier for users to find the correct apps than if users had to find the same apps in the Microsoft Store. The other benefit is that custom apps can be placed in the Microsoft Store for Business and installed without sideloading. The custom apps you place in the Microsoft Store for Business do not have to go through an approval process the way that apps for the Windows Store do.

**Activity 8-8**

### Installing and Removing a Microsoft Store App

**Time Required:**10 minutes

**Objective:**Install and remove an app from the Microsoft Store

**Description:**Many of the apps you obtain from the Microsoft Store are APPX-based apps. In this activity, you install an app from the Microsoft Store and then remove it by using Windows PowerShell.

1. 1

If necessary, start your computer and sign in.

1. 2

On the taskbar, click **Microsoft Store**.

1. 3

In the Microsoft Store window, in the Search box, type **Whiteboard** and then press **Enter**.

1. 4

In the list of search results, click **Microsoft Whiteboard**.

1. 5

Read the information about Microsoft Whiteboard and then click **Get**.

1. 6

In the Use across your devices dialog box, click **No, thanks**.

1. 7

Wait while Microsoft Whiteboard downloads and installs.

1. 8

Close the Microsoft Store window.

1. 9

Right-click the **Start** button and then click **Windows PowerShell (Admin)**.

1. 10

In the User Account Control dialog box, click **Yes**.

1. 11

At the Windows PowerShell prompt, to view a list of installed packages, type **Get-AppPackage** and then press **Enter**. Microsoft Whiteboard appears at the end of the list.

1. 12

To display a simplified list, type **Get-AppPackage | Format-List Name** and then press **Enter**.

1. 13

To view only information about Microsoft Whiteboard, type **Get-AppPackage -Name Microsoft.Whiteboard** and then press **Enter**.

1. 14

To remove Microsoft Whiteboard, type **Get-AppPackage -Name Microsoft.Whiteboard | Remove-AppPackage** and then press **Enter**.

1. 15

Close the Windows PowerShell window.

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

## 8-3cMSIX Deployment

**MSIX**, a new app packaging option created by Microsoft to improve application deployment, supports Win32 apps, .NET Framework apps, and UWP apps. It is intended to be a universal packaging option that is less restrictive than APPX but still provides some of the APPX advantages.

MSIX apps are containerized. This means that the apps run in an isolated environment that redirects file system and registry access. This ensures that files and registry keys used by MSIX apps do not conflict with one another or with different apps.

App developers can distribute MSIX apps for consumers and businesses through the Microsoft Store. The apps then can be updated automatically through the Microsoft Store.

Larger organizations might have apps that are not distributed from the Microsoft Store or Microsoft Store for Business. To install these apps, you need to enable sideloading, just like what is required for APPX apps. The same setting applies for both types of apps. In addition, you can install and manage MSIX apps by using the same \*-AppPackage cmdlets. You can automate distribution MSIX apps by using Microsoft Endpoint Configuration Manager or Microsoft Intune.

**Tip**

You can also double-click APPX and MSIX apps to install them.

MSIX can also be distributed from a web server by using App Installer. This method for delivery can be useful for internally developed apps. As part of the MSIX package, you can specify a location to check for updates to keep the app automatically updated.

**Note 2**

For detailed information about MSIX app packaging, see What is MSIX at [https://docs.microsoft.com/en-us/windows/msix/overview](https://docs.microsoft.com/en-us/windows/msix/overview%22%20%5Ct%20%22_blank).

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

## 8-3dWindows 10 in S Mode

**Windows 10 in S mode** is a configuration of Windows 10 that allows only apps from the Microsoft Store to be installed and run. This version of Windows 10 is designed to make a computer more like using a tablet or phone where all apps are installed from a trusted source. Existing MSI apps can be repackaged as MSIX apps for use with Windows 10 in S mode.

The choice to deploy Windows 10 in S mode must be made at initial deployment. In most cases, computers are received from the manufacturer using Windows 10 in S mode. As an administrator, however, you can configure a starting system image that deploys Windows 10 in S mode. Windows 10 in S mode is available for all editions of Windows 10. A computer with Windows 10 in S mode can be converted to the full operating system that can run all apps, but it cannot be switched to Windows 10 in S mode. To switch out of S mode, go to the Microsoft Store and install the Switch out of S mode app.

In addition to limiting which apps can run, Windows 10 in S mode has the following limitations:

* Can’t be domain joined—A computer running Windows 10 in S mode can’t be joined to a domain-based network for management or authentication. It can be joined to Azure AD. You can use a local user account, a Microsoft account, or an Azure AD account to sign in.
* Limited support for peripherals—Due to the limitations on software, much of the advanced peripheral management software does not work with Windows 10 in S mode. Basic functionality provided by the drivers in Windows 10 is supported for most peripherals.

**Note 3**

For detailed information about deploying Windows 10 in S mode, see Windows 10 in S mode manufacturing overview at [https://docs.microsoft.com/en-us/windows-hardware/manufacture/desktop/windows-10-s-overview](https://docs.microsoft.com/en-us/windows-hardware/manufacture/desktop/windows-10-s-overview%22%20%5Ct%20%22_blank).

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

## 8-3eMicrosoft 365 Apps Click-to-Run Deployment

For some Microsoft 365 and Office 365 licensing plans, Microsoft 365 Apps is included for users. At the time of this writing, Microsoft 365 Apps has equivalent functionality to Microsoft Office Professional Plus 2019. This is the full Microsoft Office suite that includes Word, PowerPoint, Excel, Outlook, Publisher, and Access. Both versions of the Microsoft Office Suite are distributed by using Click-to-Run.

**Click-to-Run** streams the installation of Microsoft 365 Apps. This means that it delivers the critical components first and allows the users to begin using apps before installation is complete. This is important because Microsoft 365 Apps is designed to be deployed by users from the Microsoft 365 or Office 365 website.

The Click-to-Run deployment method is also responsible for deploying updates. When updates are available, those updates are streamed to clients when they start the apps. Updates for these apps are not installed by Windows Update. By default, the apps are updated via the Internet from Microsoft servers. If you want to control network utilization or clients don’t have direct access to the Internet, however, you can configure clients to obtain updates from a shared folder.

**Caution**

Applying Microsoft 365 Apps updates from a shared folder requires significantly more administrative effort than allowing updates to be downloaded from Microsoft.

When users install Microsoft 365 Apps, no customization is possible. All the components are installed by default. If you need to limit components that are installed, you can use the Office Deployment Tool for Click-to-Run. This tool creates an XML file that can be used to limit the components that are installed. You can modify an existing installation of Microsoft 365 Apps or download a local installation source so that users can access the initial installation files locally.

Licensing for Microsoft 365 Apps is on a per-user basis. Users have an account in Office 365 or Microsoft 365, and they need to authenticate using their account every 30 days to keep their installation of Microsoft 365 Apps active. This is an alternative to the traditional license key and activation process used by volume-licensed copies of Microsoft Office. At the time of writing, one user can install Microsoft 365 Apps on up to five devices. This allows users to install Microsoft 365 Apps on their work desktop computer, a mobile computer, and their home computer with a single license.

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

**8-4**App Compatibility

One of the primary concerns when planning a deployment of Windows 10 is application compatibility. The good news is that the vast majority of apps designed for Windows 7 and newer run properly on Windows 10. Most compatibility issues are for very old legacy apps created for Windows XP.

If you have older apps that do not run properly on Windows 10, the preferred option is to upgrade the app to a newer version that works properly. Unfortunately, some apps might not have an upgraded version available. For example, a line-of-business app may have been custom developed many years ago and no easy alternative exists. If no path for upgrading is possible, you need to explore alternate options for compatibility.

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

## 8-4aCompatibility Settings for Executables

Windows 10 includes some basic features for application compatibility that you can set when viewing the properties of an executable file, shown in Figure 8-7. Some of the common issues that can be addressed are:

* Reliance on older Windows versions—Some Windows apps specifically verify which version of Windows they are running on and do not run if the reported version is not recognized. For example, an app might look for Windows 7 as an acceptable version and not recognize Windows 10. To resolve this issue, you can set the compatibility mode as Windows Vista (RTM, Service Pack 1, or Service Pack 2), Windows 7, or Windows 8. When you set the compatibility mode, and an app queries the operating system version, the configured compatibility level is reported back to the app.
* Display issues—Many older apps were designed for smaller and lower-resolution displays. These apps don’t display properly on newer displays with higher resolution and operating systems that expect higher color depth. Windows 10 can force the app to run in a 640 × 480 resolution window, eliminate scaling on high-resolution displays, and reduce color depth.
* Reliance on administrator permissions—Some apps require administrative permissions to run properly. This is due to poor programming practices by developers that require the user to write to privileged areas of the file system or the registry. In Windows 10, with User Access Control (UAC), even if you are an administrator of the local computer, apps are run with standard user permissions unless a manifest file for the application indicates that the app requires elevated privileges. Older apps do not have this manifest file; however, you can force apps to run with elevated privileges by selecting the Run this program as an administrator option. This avoids the need to manually select Run as administrator each time you start the app. The user still needs to be a local administrator on the computer.

**Figure 8-7Compatibility Settings for an Executable**



If you are unsure of how to resolve compatibility issues for an application, you can run the compatibility troubleshooter. This wizard attempts to automatically detect the compatibility problem. If automatic detection fails, you can utilize a wizard that asks you a series of questions to identify a likely resolution. After the wizard is complete, the applied settings are visible on the Compatibility tab and can be disabled there.

**Activity 8-9**

### Configuring App Compatibility

**Time Required:**10 minutes

**Objective:**Configure compatibility settings for an app

**Description:**Some older apps do not run properly in newer versions of Windows, including Windows 10. In this activity, you configure compatibility settings for an app.

1. 1

If necessary, start your computer and sign in.

1. 2

On the taskbar, click **File Explorer** and then navigate to **C:\Windows**.

1. 3

In the File Explorer window, right-click **write**, click **Copy**, and then exit File Explorer.

1. 4

Right-click the desktop and then click **Paste**.

1. 5

On the desktop, right-click **write** and then click **Troubleshoot compatibility**.

1. 6

In the Program Compatibility Troubleshooter, on the Select Troubleshooting option screen, click **Troubleshoot program**.

1. 7

On the What problems do you notice? screen, select the **The program requires additional permissions** check box and then click **Next**.

1. 8

On the Test compatibility settings for the program screen, click **Test the program**.

1. 9

In the User Account Control dialog box, click **Yes**. The UAC dialog box may be minimized on the taskbar. If so, click it to make it active.

1. 10

Exit WordPad.

1. 11

In the Program Compatibility Troubleshooter, click **Next**.

1. 12

On the Troubleshooting has completed screen, click **Yes, save these settings for this program**.

1. 13

On the Troubleshooting has completed screen, click **Close**.

1. 14

On the desktop, right-click **write** and then click **Properties**.

1. 15

In the write Properties window, click the **Compatibility** tab, and read the options.

1. 16

Verify that the **Run this program as an administrator** check box is selected and then click **Cancel**.

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

## 8-4bWindows ADK Tools

To help identify and resolve app compatibility issues, the Windows 10 Assessment and Deployment Kit (ADK), shown in Figure 8-8, includes Compatibility Administrator and the Standard User Analyzer Tool (SUA). If you have an older application that you can’t get to run by using the basic compatibility settings, these tools provide more detailed fixes.

**Figure 8-8Windows 10 ADK Application Compatibility Tools**





**Caution**

You might see some references to Windows 10 ADK containing the Application Compatibility Tool kit. This has been removed from newer versions of the Windows 10 ADK. Only Compatibility Administrator and SUA are still included.

**Compatibility Administrator**, shown in Figure 8-9, contains a list of older commercial applications and known fixes for them. Most of these applications were produced prior to 2010. If you don’t have the budget to replace these applications with newer versions, you can use Compatibility Administrator to create a compatibility database for the apps.

**Figure 8-9Compatibility Administrator**





The bitness of Compatibility Administrator needs to match the bitness of the application you are mitigating. For example, you need to use the 32-bit version of Compatibility Administrator to create a compatibility database for 32-bit applications.

For noncommercial apps that are not part of Compatibility Administrator, you might be able to identify what is required for compatibility by using SUA. SUA monitors application activity and identifies potential fixes. Those potential fixes are saved as a compatibility database in the same format generated by Compatibility Administrator.

In some cases, compatibility issues with custom apps can be resolved by using the **Standard User Analyzer (SUA)**. This tool, which is included with Compatibility Monitor, is used to closely monitor the activity of a single app and provide a resolution for problems if possible. SUA also generates a compatibility-fix database.

To apply a compatibility database (.sdb) to multiple computers, you can use sdbinst.exe. This tool can be scripted and run at sign-in or by a software management tool, such as Microsoft Endpoint Configuration Manager.

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

## 8-4cDesktop Analytics

**Desktop Analytics** is a cloud-based solution to identify compatibility problems with applications and suggest mitigations. This service integrates with Microsoft Endpoint Configuration Manager to leverage existing application inventory data and provide an infrastructure for deploying mitigations. Because the deployment of Windows Desktop operating systems is based on new feature updates, this solution is optimized for that purpose.

Because Desktop Analytics is cloud-based, the service has access to compatibility data gathered by many customers. When a compatibility issue is identified by other customers, you can be warned about this same issue even before you do any testing. Desktop Analytics can identify this based on the hardware and software inventory collected by Configuration Manager.

Another key benefit of Desktop Analytics is identifying a pilot group for testing. Desktop Analytics uses the hardware and software inventory to identify the smallest possible group of computers that include the largest possible range of configurations. This makes it more likely that you identify issues with software updates before you perform organization-wide deployment.

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

## 8-4dClient Hyper-V

If an app is not compatible with Windows 10 and there is no way to mitigate the issue, you need to run an older operating system to keep using that application. For example, if all users are being upgraded to a 64-bit version of Windows 10, another operating system is required to support a legacy 16-bit Windows app. Although it is possible to maintain an older physical computer for the purpose of running the older app, that older computer will be prone to failure over time.

**Client Hyper-V** is virtualization software for Windows 10. It allows you to create virtual machines with a completely independent operating system that shares the hardware on the computer running Windows 10. In the virtual machine, you can install the required operating system and application.

The potential downsides to using Client Hyper-V are as follows:

* No easy way exists to move data from the virtual machine to the host computer.
* The operating system in the virtual machine still needs to be managed with software updates and any other management considerations.
* Some users find it confusing to use a separate virtual machine for some tasks.

**Note 4**

For more information, see Introduction to Hyper-V on Windows 10 at [https://docs.microsoft.com/en-us/virtualization/hyper-v-on-windows/about/](https://docs.microsoft.com/en-us/virtualization/hyper-v-on-windows/about/%22%20%5Ct%20%22_blank).

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

## 8-4eVirtual Desktop Infrastructure and RemoteApp

Windows Server can host and provide access to virtual desktops. This functionality is provided by **Remote Desktop Services**. Several variations exist, but for all of them, the application runs on a computer and the Remote Desktop Protocol (RDP) is used to access the visual information for the app.

By running the app on a remote computer, any application conflicts on the local computer are avoided. For example, when Microsoft 365 Apps is installed on a computer, you can’t have older MSI-based versions of apps, such as Visio, installed locally. By using Remote Desktop Services, you can remote control the application running elsewhere while it appears to run locally.

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

## 8-4fApp-V

**Application Virtualization (App-V)** is a technology that both deploys and manages apps. Instead of a standard installation, apps are streamed to computers and can begin executing before all the app files are on the destination computer. Updates are never installed directly on the client computers. Instead, the source app is updated and the updates stream to the clients. This is similar to how Microsoft 365 Apps is distributed because the Click-to-Run functionality for Microsoft 365 Apps is based on App-V.

Apps distributed by using App-V run in a virtualized environment within the operating system. From a user perspective, the app appears to be installed like a normal application, but conflicts with other applications running on the computer are avoided. You can use App-V to resolve compatibility conflicts when two apps cannot both be installed on the same computer. The apps, however, must still be compatible with the operating system.

**Caution**

App-V is no longer being developed as a technology. Microsoft hopes that the virtualization within MSIX packaging will meet this need instead.

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

**8-5**Remote Desktop Services

The Remote Desktop Services role in Windows Server provides a way to run apps on a remote server or virtual machine and have the display appear on the local computer. This way, users can run apps without ever installing them on their local computer. This system can be used when app compatibility might cause concerns, but more often, it is used to provide remote access to apps.

The remote desktop protocol (RDP) used by Remote Desktop Services is very efficient. This makes it feasible to provide access to remote desktops and apps over the Internet and even over public Wi-Fi. It also provides the flexibility to use apps from anywhere and on any device that has an RDP client. Microsoft provides Remote Desktop clients not only for Windows, but also for iOS and Android devices.

Remote Desktop Services can provide these two types of virtual desktops:

* *Session-based virtual desktops*—This type of virtual desktop is hosted on a Remote Desktop Session Host (RD Session Host). An RD Session Host is running a Windows Server operating system, such as Windows Server 2019. A single RD Session Host is shared by multiple users at one time. When the users connect, each gets their own independent desktop and runs their own applications; however, the core operating system services are shared, which allows many users to share the same server.
* *Virtual machine-based virtual desktops*—This type of virtual desktop is hosted on a Windows server running Hyper-V. Each user has an independent virtual machine with a completely independent operating system, which is typically Windows 10. The hardware of the server is shared among the virtual machines, but each virtual machine is independent. This type of virtual desktop has lower density than session-based virtual desktops. Figure 8-10 illustrates the difference between session-based virtual desktops and virtual machine-based virtual desktops.

**Figure 8-10Types of Virtual Desktops**





Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

## 8-5aRemoteApp

Providing users with full remote desktops can sometimes be confusing for the users if they don’t understand conceptually how they can have two desktops at the same time. To simplify access to apps, there is also **RemoteApp**. RemoteApp still uses either session-based virtual desktops or virtual machine-based virtual desktops to execute apps, but the user interface is different. When users start a RemoteApp running on a virtual desktop, it behaves like a regular application. A window opens only for that application. For most users, this is a simpler interface and easier to understand than a full desktop.

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

## 8-5bAccessing Virtual Desktops and RemoteApp

The most popular configuration to access virtual desktops and RemoteApp programs requires you to sign in to a website. This website is hosted on a Remote Desktop Web Access (RD Web Access) server. Unlike older virtual desktop solutions, like Terminal Services, you do not directly access an RD Session Host.

After the users authenticate to RD Web Access, they see a webpage with the RemoteApp programs and virtual desktops to which they have been given access. Users click the RemoteApp program or virtual desktop that they want to access, and Remote Desktop Connection is run to access it. Remote Desktop Connection is the RDP client included in Windows 10.

RemoteApp and Desktop Connections is an alternative to RD Web Access for accessing RemoteApp programs and virtual desktops hosted by Remote Desktop Services. This client reads the list of RemoteApp programs and virtual desktops by accessing a web feed URL on the RD Web Access server. The web feed URL is [https://<servername>/RDWeb/Feed/webfeed.aspx](https://%3Cservername%3E/RDWeb/Feed/webfeed.aspx%22%20%5Ct%20%22_blank), where <servername> is the host name of the RD Web Access server.

The three ways to configure RemoteApp and Desktop Connections with the URL for the web feed are as follows:

* Manually enter the URL—You can provide users with instructions on how to configure the URL the first time they start it. This solution is not very scalable and runs the risk of users making typing errors, but it is simple to implement.
* Use Group Policy—You can configure the URL in a Group Policy Object at \User Configuration\Policies\Administrative Templates\Windows Components\Remote Desktop Services\RemoteApp and Desktop Connections. If all your computers are domain joined, this is a fast way to configure all of them in a single step.
* Based on email address—The initial configuration for RemoteApp and Desktop Connections can use a DNS record based on an email address provided by the user. For the domain of the email address, RemoteApp and Desktop Connections looks for an \_rdac TXT record. The value of that record is the URL for the web feed. This solution requires users to initiate it, but it works for internal domain-joined computers and devices that are not domain joined.

The wizard for configuring RemoteApp and Desktop Connections prompts for either a URL or email address, as shown in Figure 8-11. Based on this information, the wizard connects with the web feed and downloads the list of RemoteApp programs and virtual desktops to which the user has access.

**Figure 8-11Configuring RemoteApp and Desktops**





**Tip**

RemoteApp and Desktop Connections refreshes data from the web feed only once per day. It can be refreshed manually if required.

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

## 8-5cWindows Virtual Desktop

For organizations that are moving to cloud-based services, Microsoft provides **Windows Virtual Desktop** in Microsoft Azure. This is a similar configuration for Remote Desktop Services, but the virtual machines that build the infrastructure are hosted in Microsoft Azure. Like Remote Desktop Services, Windows Virtual Desktop can provide access to full desktops or RemoteApp programs.

Because Windows Virtual Desktop is hosted as a cloud service, it is much faster to deploy and easier to license. You can deploy RD Session hosts based on Windows Server 2012 R2 or later. You can configure VDI by using Windows 7 with extended support until 2023 or Windows 10. A new multisession version of Windows 10 also is available as an RD Session Host. This provides some of the scalability and cost effectiveness of using Windows Server as an RD Session Host with application compatibility of using Windows 10. Windows 10 multisession is not available for use on premises.

**Tip**

To deploy Microsoft Office Suite on a server-based RD Session Host, you need to use a perpetual license version. Using Microsoft 365 Apps is supported on Windows 10 virtual desktops (including Windows 10 multisession).

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

# Chapter Review

## 8-6a**Summary**

* Windows 10 supports running apps by providing various application environments. You can run Win32, Win64, .NET Framework, and UWP apps.
* The registry is a central store of configuration information for apps and the operating system. It is composed of hives that contain registry keys and values. Two commonly accessed hives are HKCU and HKLM.
* To edit the registry, you can use the graphical Registry Editor, reg.exe, or Windows PowerShell cmdlets. Before editing the registry, you should export the keys so that you can restore them if required.
* MSI-based apps are installed by the Windows Installer service. You can automate installation of MSI-based apps by using Group Policy or Configuration Manager.
* Installation of AWP apps on desktop computers can be automated by using Microsoft Endpoint Configuration Manager. For phones, tablets, and non-domain-joined computers, you can automate deployment by using mobile device management software, such as Microsoft Intune. The Microsoft Store for Business is a customizable website that you can use to provide your users with access only to specific apps.
* MSIX is a newer app packaging option that can be used to deploy Win32 apps, .NET Framework apps, and UWP apps. This packaging format can also be distributed through the Microsoft Store.
* Microsoft 365 Apps installs by using a new technology called Click-to-Run.
* Windows 10 provides some basic functionality for resolving app compatibility issues; however, you can also use Compatibility Administrator, Client Hyper-V, Remote Desktop Services, and App-V.
* Remote Desktop Services provides access to virtual desktops and RemoteApp programs. A virtual desktop is a complete desktop with applications. RemoteApp provides access to only an app in a single window. Windows Virtual Desktop is cloud-based infrastructure to provide access to remote desktops and RemoteApp programs.

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

# Chapter Review

## 8-6b**Key Terms**

* **.NET Framework**
* **application environment subsystem**
* **application programming interface (API)**
* **Application Virtualization (App-V)**
* **APPX file**
* **Click-to-Run**
* **Client Hyper-V**
* **Compatibility Administrator**
* **Desktop Analytics**
* **hive**
* **Microsoft Store for Business**
* **MSI file**
* **MSIX**
* **registry**
* **Registry Editor**
* **registry key**
* **RemoteApp**
* **Remote Desktop Services**
* **Standard User Analyzer (SUA)**
* **Universal Windows Platform (UWP)**
* **Windows 10 in S mode**
* **Windows API**
* **Windows on Windows 64 (WOW64)**
* **Windows Virtual Desktop**

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

# Chapter Review

## 8-6c**Review Questions**

1. Which types of apps are not supported by a 64-bit edition of Windows 10? (Choose all that apply.)
	1. DOS
	2. Win16
	3. Win32
	4. Win64
	5. .NET Framework
2. Which registry hive is used to store global information about apps regardless of the user who is signed in?
	1. HKEY\_CLASSES\_ROOT
	2. HKEY\_CURRENT\_USER
	3. HKEY\_LOCAL\_MACHINE
	4. HKEY\_DYN\_DATA
	5. HKEY\_GLOBAL\_CONFIG
3. On a 64-bit version of Windows 10, where are 32-bit apps typically installed?
	1. C:\Program Files
	2. C:\Windows\SysWOW64
	3. C:\Program Files (WOW64)
	4. C:\Program Files (x86)
	5. C:\Windows\WOW64
4. Which versions of .NET Framework are included with Windows 10? (Choose all that apply.)
	1. .NET Framework 1.1
	2. .NET Framework 2.0
	3. .NET Framework 3.5
	4. .NET Framework 4.0
	5. .NET Framework 4.8
5. All Microsoft Store apps are UWP apps. True or False?
6. You can use a 32-bit version of Windows 10 to run legacy Win16 apps. True or False?
7. Which registry hive contains settings that are imported from ntuser.dat?
	1. HKEY\_CLASSES\_ROOT
	2. HKEY\_CURRENT\_USER
	3. HKEY\_LOCAL\_MACHINE
	4. HKEY\_DYN\_DATA
	5. HKEY\_GLOBAL\_CONFIG
8. Which tools or methods can you use to import a .reg file? (Choose all that apply.)
	1. Registry Editor (regedit)
	2. regimp.exe
	3. reg.exe
	4. double-click the .reg file in File Explorer
	5. Import-Registry
9. You can automate installation of Microsoft Store apps by using Group Policy. True or False?
10. Which cmdlet do you use to sideload UWP apps?
	1. Add-AppxPackage
	2. Install-WinApp
	3. Install-WinStoreApp
	4. Add-WinApp
	5. New-Sideload
11. Microsoft 365 Apps never receives updates through Windows Update. True or False?
12. Which compatibility issues can be fixed for an app by using the capabilities included in Windows 10? (Choose all that apply.)
	1. wrong version of .NET Framework
	2. a version check to verify that the app is running on Windows 7
	3. poor display quality on full screen when the screen resolution is 1920 × 1024
	4. required to select Run as administrator
	5. odd colors when the color depth is greater than 16-bit
13. Which tool is used to deploy compatibility fixes to multiple computers?
	1. Compatibility Administrator
	2. fixinst.exe
	3. Runtime-analysis package
	4. Standard User Analyzer
	5. sdbinst.exe
14. If you have a computer running a 64-bit version of Windows 10 with Client Hyper-V, it is possible to run a 16-bit Windows app in a virtual machine. True or False?
15. Which of the following are true about App-V? (Choose all that apply.)
	1. Apps are streamed for installation.
	2. RDP is used to access the app.
	3. Virtual environments prevent conflicts between apps.
	4. Virtual environments allow a 16-bit app to be run on a 64-bit operating system.
	5. Apps are updated when the source on the server is updated.
16. Microsoft Store for Business contains Azure RemoteApp programs. True or False?
17. Which three methods can you use to configure RemoteApp and Desktop Connections with the URL of the web feed? (Choose all that apply.)
	1. Manually enter the URL during configuration.
	2. Create a CNAME record in DNS that includes the URL and have users enter their email address during configuration.
	3. Configure a Group Policy Object with the correct URL.
	4. Create a TXT record in DNS that includes the URL and have users enter their email address during configuration.
	5. Package an MSI file with the correct configuration information.
18. Which PowerShell cmdlet is used to modify a registry key value?
	1. Set-ItemProperty
	2. Set-RegKeyValue
	3. Set-ChildItem
	4. Set-LocationValue
	5. Set-ChildItemValue
19. Which of the following are true about Windows 10 multi-session? (Choose all that apply.)
	1. It is available only as part of Windows Virtual Desktop.
	2. It is available for on-premises installation.
	3. It supports up to 5 simultaneous RDP sessions.
	4. It supports up to 10 simultaneous RDP sessions.
	5. It can be used to provide RemoteApp programs.
20. Developers can use MSIX packaging to distribute .NET Framework and UWP apps, but not Win32 apps. True or False?

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)

# Chapter Review

## 8-6d**Case Projects**

**Case Project 8-1**

### Application Compatibility

Gigantic Life Insurance has thousands of desktop computers running a wide variety of apps. Many desktops are still running Windows 7 due to concerns about application compatibility with Windows 10. Many legacy apps simply don’t run on Windows 10. What are your options to address this problem and allow an upgrade to Windows 10?

**Case Project 8-2**

### Remote Access to Apps

Hyperactive Media now has more than 40 salespeople on the road meeting with customers. The salespeople currently have laptops with locally installed apps; however, this means that salespeople need to come back to the office and transfer sales orders from their laptops to the central order system. Describe how salespeople could be given remote access to this system by using RemoteApp. What concerns might you have about using RemoteApp?

**Case Project 8-3**

### .NET Framework Updates

Gigantic Life Insurance has thousands of desktop computers running many .NET Framework apps. As a best practice, you realize that you need to apply security updates as quickly as possible. A colleague is rightly concerned that apps might start to fail if .NET Framework updates are applied. What can you do to ensure that .NET Framework updates can be deployed as quickly as possible?

Go to pg.

**help**

Application Opened

[Main content](https://ng.cengage.com/static/nbreader/ui/apps/nbreader/fullbook.html#header)