Power Shell 101

Module 3 Hands-on Activity – PowerShell in VSCode

4/23/2019 Developed by Jin Chang, Sion Yoon
4/27/2019 Tested by Sion Yoon, Jin Chang
Center for Information Assurance (CIAE) at City University of Seattle

Learning Outcomes

- Learn about using Get-Command and Get-Service cmdlets, the help system, and pipeline.

Resources

- Cmdlets and the help system
- https://docs.microsoft.com/en-us/powershell/scripting/overview?view=powershell-6
- Pipeline
  https://www.youtube.com/watch?v=WpFc2m_uLfM&list=PL6D474E721138865A&index=15

Activities

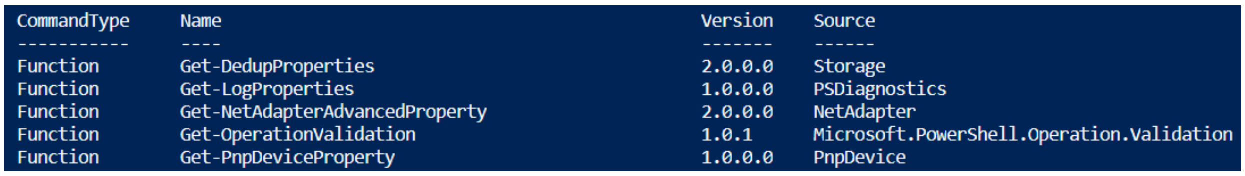
- Get-Command & Get-Service
- Help-System
- Pipeline
  - More
  - Exporting to CSV and XML
  - Code Example
- Q&A

Get-Command & Get-Service

1) Open PowerShell in VSCode

2) Before diving into the help system and pipeline, let's quickly go over the "get-command" (alias: "gcm") and "get-service" cmdlets, which are commands that you are going to use very often. The "get-command" cmdlet gets all commands that are installed on the computer and you can use this cmdlet to search for the command that you are looking for. Type:

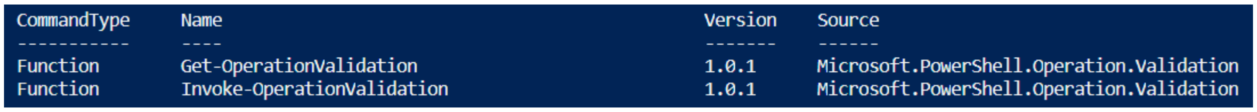
```
%get-command *oper*
```



```
CommandType     Name                                               Version    Source
-----------     ----                                               -------    ------
Function        Get-DedupProperties                                2.0.0.0    Storage
Function        Get-LogProperties                                  1.0.0.0    PSDiagnostics
Function        Get-NetAdapterAdvancedProperty                     2.0.0.0    NetAdapter
Function        Get-OperationValidation                            1.0.1      Microsoft.PowerShell.Operation.Validation
Function        Get-PnpDeviceProperty                              1.0.0.0    PnpDevice
```

You will see a list of all commands that contain "oper". If you are looking for a command that relates to operation, you can trim the search even more by typing:

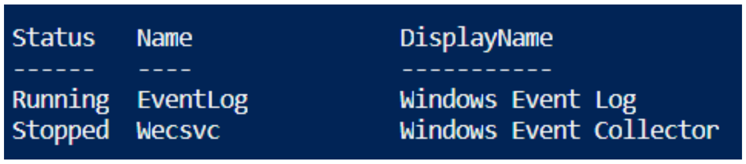
```
%get-command *opera*
```



```
CommandType     Name                                               Version    Source
-----------     ----                                               -------    ------
Function        Get-OperationValidation                            1.0.1      Microsoft.PowerShell.Operation.Validation
Function        Invoke-OperationValidation                         1.0.1      Microsoft.PowerShell.Operation.Validation
```

3) The get-service cmdlet (alias: "gsv") lists services that are running or currently stopped. To retrieve a list of services that has a display name of windows event, type,

```
%get-service -displayname "windows *event*"
```



```
Status   Name               DisplayName
------   ----               -----------
Running  EventLog           Windows Event Log
Stopped  Wecsvc             Windows Event Collector
```

<u>Help System</u>

4) The PowerShell help system is very useful in using to use PowerShell effectively. Type the following 3 help commands, see if you tell the difference and similarities. Use the 'spacebar' to move through the contents, 'enter' to proceed line-by-line, or 'q' to exit the page.

```
%get-help
%help
%man
```

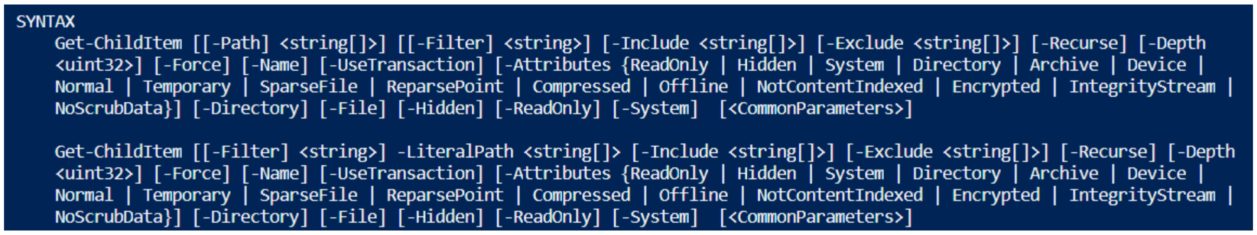
Did you notice that the screen is scrolled all the way to the bottom for "`get-help`" while you can move through the contents for "`help`" and "`man`"?

5) There are different ways in which to get help by using different parameters in the help system. A few of many parameters are:
   - **Detailed** - displays parameter descriptions and examples.
   - **Full** - displays parameter descriptions, examples, input and output object types, and additional notes.
   - **Examples** - displays only the name, synopsis, and all examples.
   - **Online** - displays the online version of the help topic in your default web browser.
   - **ShowWindow** - displays help in a pop-up window.

Test these help parameters using get-childitem.

```
%help get-childitem
%help get-childitem -full
%help get-childitem -detailed
%help get-childitem -example
%help get-childitem -online
%help get-childitem -showwindow
```

6) Within the help system, you will see that there is information about syntax



```
SYNTAX
    Get-ChildItem [[-Path] <string[]>] [[-Filter] <string>] [-Include <string[]>] [-Exclude <string[]>] [-Recurse] [-Depth
    <uint32>] [-Force] [-Name] [-UseTransaction] [-Attributes {ReadOnly | Hidden | System | Directory | Archive | Device |
    Normal | Temporary | SparseFile | ReparsePoint | Compressed | Offline | NotContentIndexed | Encrypted | IntegrityStream |
    NoScrubData}] [-Directory] [-File] [-Hidden] [-ReadOnly] [-System]  [<CommonParameters>]

    Get-ChildItem [[-Filter] <string>] -LiteralPath <string[]> [-Include <string[]>] [-Exclude <string[]>] [-Recurse] [-Depth
    <uint32>] [-Force] [-Name] [-UseTransaction] [-Attributes {ReadOnly | Hidden | System | Directory | Archive | Device |
    Normal | Temporary | SparseFile | ReparsePoint | Compressed | Offline | NotContentIndexed | Encrypted | IntegrityStream |
    NoScrubData}] [-Directory] [-File] [-Hidden] [-ReadOnly] [-System]  [<CommonParameters>]
```

7) Let's look at the first parameter between the square brackets "[]" in the first parameter set. Anything between the square brackets are optional, so you can run the command "get-childitem" without typing the parameters and it defaults to the current path. But what if you want to type a path and get directory (childitem) from a specific path. You can type:

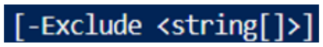
```
% get-childitem –path c:\
```

or

```
% get-childitem c:\
```

You can use either way since –path parameter name is also in the square brackets and also optional. So if you just provide a string in our example case at the first position, it will take that string and assume that it meant it the –path parameter.

8) On the other hand, how about the –exclude parameter, which is not in square brackets.



```
[-Exclude <string[]>]
```

The parameter itself is optional, but it means that you would need to type the parameter name "-exclude" before providing the value. You do not actually have to type all of the "-exclude" but just enough so that PowerShell can tell the difference between the parameter you are referring to and other existing parameter options. For example, you can just type "-e" since no other parameter starts with "–e". Type:

```
% get-childitem –e *.txt
```

Note that the square brackets that are against one another mean that you can provide more than one value. Moreover, you can use "help get-childitem –full" that we went through above to learn about each parameter in the syntax.

Pipeline: More

9) Piping is a powerful feature of PowerShell. This will greatly reduce the amount of scripting effort by simply relaying the output of the earlier command into the later command as an input. Type the following command,

```
% get-alias
```

10) This displays a very long list of aliases where you would need to scroll up and down. Wouldn't it be nice if we can display one page at a time? It will be possible by simply piping it to "more" command. In other words, you are getting the output of get-alias and piping it to the input of more. Type,

```
 % get-alias | more
 % help help | more
```

Pipeline: Exporting to CSV and XML

11) You can also use the pipeline to pipe, for example, services into a CSV file. Type:

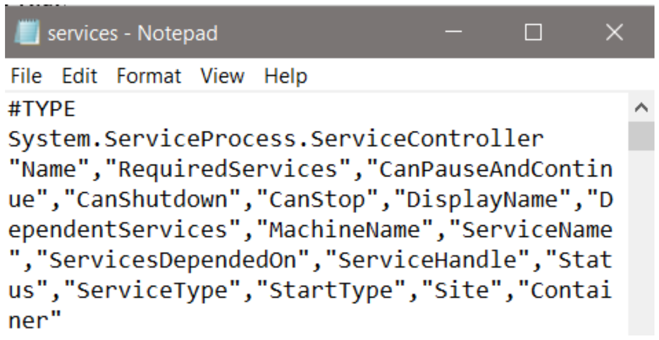
```
 % get-service | export-csv c:\services.csv
```

If you get an error message on the access to the path, redirect the path to the directory that you are on. For example:

```
% get-service | export-csv c:\Users\sion\services.csv
```

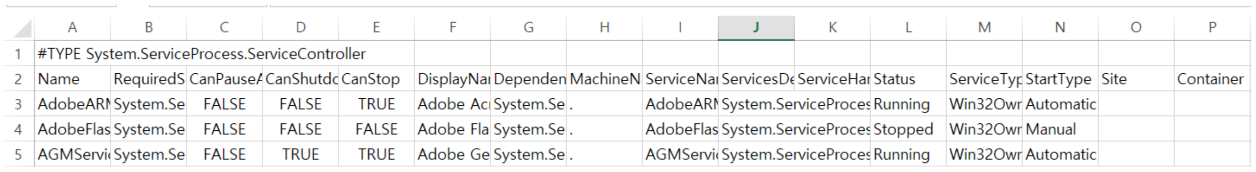
You have exported services into a CSV file name "services". To open the CSV file using notepad, type:

```
% notepad .\services.csv
```



```
#TYPE
System.ServiceProcess.ServiceController
"Name","RequiredServices","CanPauseAndContin
ue","CanShutdown","CanStop","DisplayName","D
ependentServices","MachineName","ServiceName
","ServicesDependedOn","ServiceHandle","Stat
us","ServiceType","StartType","Site","Contai
ner"
```

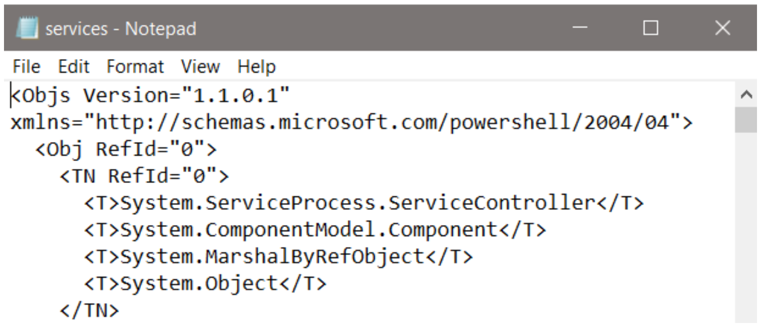
Also try opening the CSV file with Excel in your Windows file explorer.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	#TYPE System.ServiceProcess.ServiceController															
2	Name	RequiredS	CanPauseA	CanShutdo	CanStop	DisplayNa	Dependen	MachineN	ServiceNa	ServicesDe	ServiceHa	Status	ServiceTyp	StartType	Site	Container
3	AdobeARI	System.Se	FALSE	FALSE	TRUE	Adobe Ac	System.Se	.	AdobeARI	System.ServiceProces		Running	Win32Owr	Automatic		
4	AdobeFlas	System.Se	FALSE	FALSE	FALSE	Adobe Fla	System.Se	.	AdobeFlas	System.ServiceProces		Stopped	Win32Owr	Manual		
5	AGMServi	System.Se	FALSE	TRUE	TRUE	Adobe Ge	System.Se	.	AGMServi	System.ServiceProces		Running	Win32Owr	Automatic		

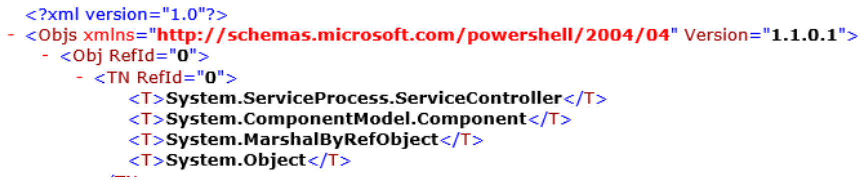
12) We can do the same for XML files. Type (change the path to as you have done in #11):

```
% get-service | export-clixml c:\services.xml
% notepad .\services.xml
```



```
<Objs Version="1.1.0.1"
xmlns="http://schemas.microsoft.com/powershell/2004/04">
  <Obj RefId="0">
    <TN RefId="0">
      <T>System.ServiceProcess.ServiceController</T>
      <T>System.ComponentModel.Component</T>
      <T>System.MarshalByRefObject</T>
      <T>System.Object</T>
    </TN>
```

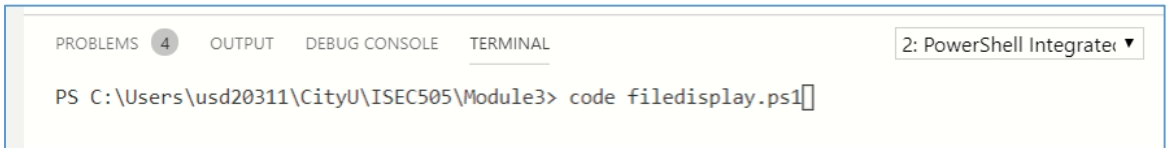
Double click the XML file on your Windows file explorer and you will see it displayed on an internet browser.



```
  <?xml version="1.0"?>
- <Objs xmlns="http://schemas.microsoft.com/powershell/2004/04" Version="1.1.0.1">
  - <Obj RefId="0">
    - <TN RefId="0">
        <T>System.ServiceProcess.ServiceController</T>
        <T>System.ComponentModel.Component</T>
        <T>System.MarshalByRefObject</T>
        <T>System.Object</T>
```

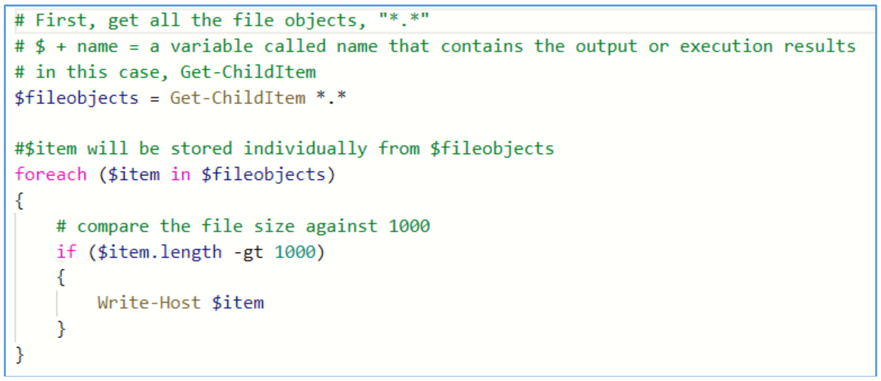
Pipeline: Code Example

13) Now, let's look little deeper in the pipe feature that can actually save your time as a system admin. Open a text file in Visual Studio Code, and name it filedisplay.ps1 like below.



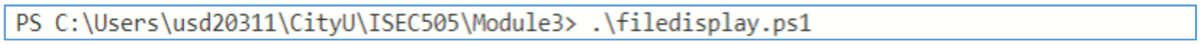
```
PS C:\Users\usd20311\CityU\ISEC505\Module3> code filedisplay.ps1
```

Go ahead and type the script that loops through a set of files and displays the files greater than 1000KB.



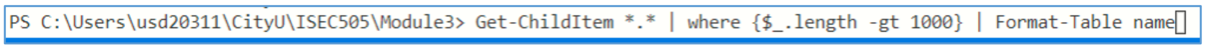
```
# First, get all the file objects, "*.*"
# $ + name = a variable called name that contains the output or execution results
# in this case, Get-ChildItem
$fileobjects = Get-ChildItem *.*

#$item will be stored individually from $fileobjects
foreach ($item in $fileobjects)
{
    # compare the file size against 1000
    if ($item.length -gt 1000)
    {
        Write-Host $item
    }
}
```

Save the file, and run it on the command line as follows;



```
PS C:\Users\usd20311\CityU\ISEC505\Module3> .\filedisplay.ps1
```

We just spent our effort to write 8 lines of code just to display the files that have 1000KB or bigger. But here is a much quicker way to this in one shot thanks to pipe!



```
PS C:\Users\usd20311\CityU\ISEC505\Module3> Get-ChildItem *.* | where {$_.length -gt 1000} | Format-Table name
```

This is only possible because of piping mechanism where " | " carries over the output of the previous command to the next command. Note "$_" is an intermediate output variable.

Q&A

1. **What is the cmdlet that stops a service?**
   Stop-Service

2. **What is the cmdlet that gets service for Windows audio service? Hint: the answer is "get-service *name*". Use the get-service –displayname parameter that we went over in #3 to find the name of the service.**

   get-service Audiosrv

3. **Using the pipeline, what is an operation that gets Windows audio service and then stop the service? This command should NOT run without the service name as it will stop all the services and you do not want to do that. Hint: answer from question #2 | answer from questions #1.**

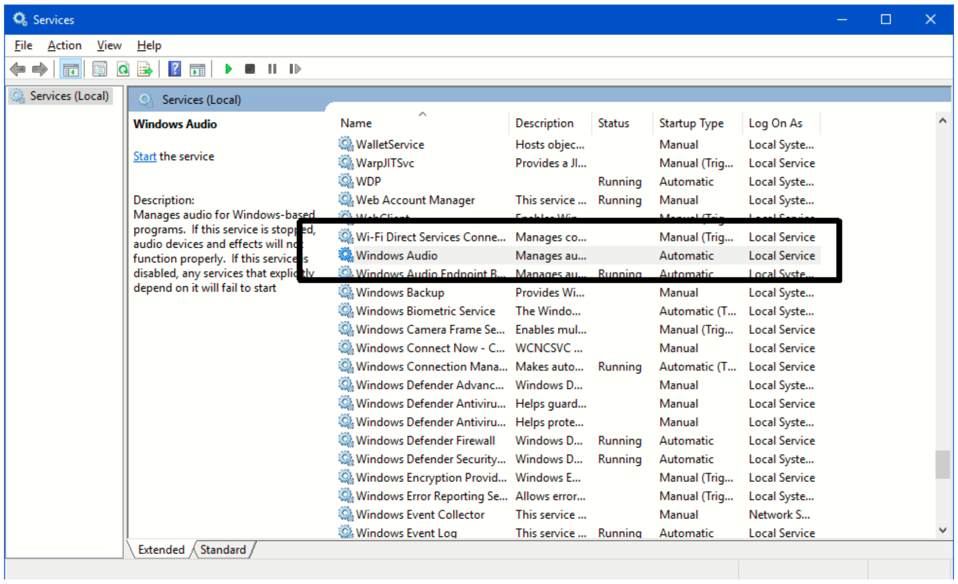
   get-service Audiosrv | stop-service

4. **Test the command line from question 3. Before you do so, you need administrator privilege to perform this operation. Run the following command prior to modifying the service status.**

```
% Start-Process PowerShell -Verb RunAs
```

If successful, you can see the speaker icon on the bottom taskbar showing

Go to the Windows Start menu and type "services" and click on the "Services" manager. Scroll down to "Windows Audio" and you will see that there is nothing indicating for its status as shown as below:



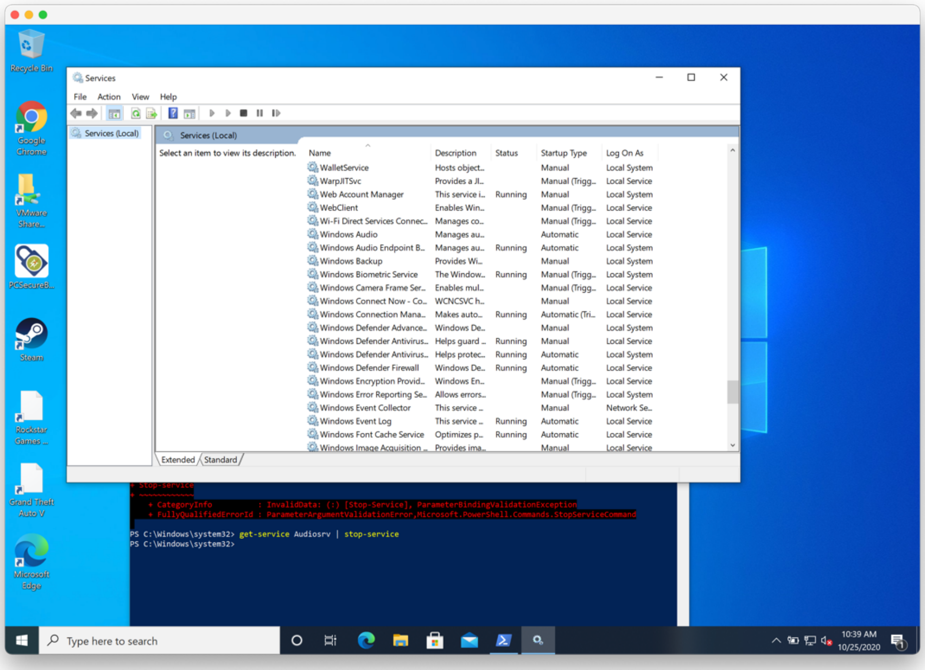
Windows Audio

Start the service

Description:
Manages audio for Windows-based programs. If this service is stopped, audio devices and effects will not function properly. If this service is disabled, any services that explicitly depend on it will fail to start

Name	Description	Status	Startup Type	Log On As
WalletService	Hosts objec...		Manual	Local Syste...
WarpJITSvc	Provides a JI...		Manual (Trig...	Local Service
WDP		Running	Automatic	Local Syste...
Web Account Manager	This service ...	Running	Manual	Local Syste...
WebClient	Enables Win...		Manual (Trig...	Local Service
Wi-Fi Direct Services Conne...	Manages co...		Manual (Trig...	Local Service
Windows Audio	Manages au...		Automatic	Local Service
Windows Audio Endpoint B...	Manages au...	Running	Automatic	Local Syste...
Windows Backup	Provides Wi...		Manual	Local Syste...
Windows Biometric Service	The Windo...		Automatic (T...	Local Syste...
Windows Camera Frame Se...	Enables mul...		Manual (Trig...	Local Service
Windows Connect Now - C...	WCNCSVC ...		Manual	Local Service
Windows Connection Mana...	Makes auto...	Running	Automatic (T...	Local Service
Windows Defender Advanc...	Windows D...		Manual	Local Syste...
Windows Defender Antiviru...	Helps guard...		Manual	Local Service
Windows Defender Antiviru...	Helps prote...		Manual	Local Syste...
Windows Defender Firewall	Windows D...	Running	Automatic	Local Service
Windows Defender Security...	Windows D...	Running	Automatic	Local Syste...
Windows Encryption Provid...	Windows E...		Manual (Trig...	Local Service
Windows Error Reporting Se...	Allows error...		Manual (Trig...	Local Syste...
Windows Event Collector	This service ...		Manual	Network S...
Windows Event Log	This service ...	Running	Automatic	Local Service

Take a screenshot of yours and paste it here:



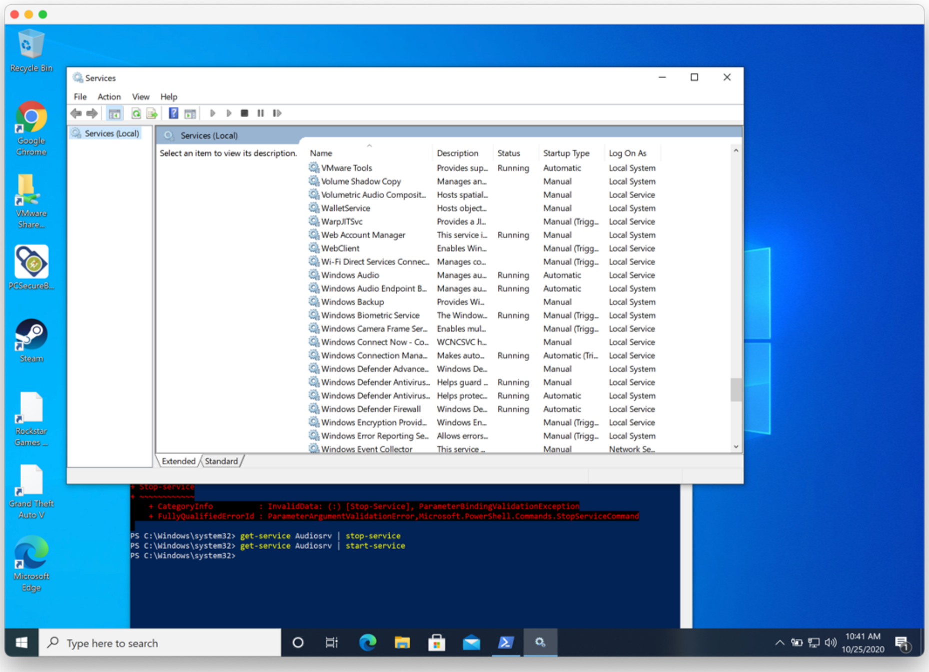
Select an item to view its description.

Name	Description	Status	Startup Type	Log On As
WalletService	Hosts object...		Manual	Local System
WarpJITSvc	Provides a JI...		Manual (Trigg...	Local Service
Web Account Manager	This service i...	Running	Manual	Local System
WebClient	Enables Win...		Manual (Trigg...	Local Service
Wi-Fi Direct Services Connec...	Manages co...		Manual (Trigg...	Local Service
Windows Audio	Manages au...		Automatic	Local Service
Windows Audio Endpoint B...	Manages au...	Running	Automatic	Local System
Windows Backup	Provides Wi...		Manual	Local System
Windows Biometric Service	The Window...	Running	Manual (Trigg...	Local System
Windows Camera Frame Ser...	Enables mul...		Manual (Trigg...	Local Service
Windows Connect Now - Co...	WCNCSVC h...		Manual	Local Service
Windows Connection Mana...	Makes auto...	Running	Automatic (Tri...	Local Service
Windows Defender Advance...	Windows De...		Manual	Local System
Windows Defender Antivirus...	Helps guard ...	Running	Manual	Local Service
Windows Defender Antivirus...	Helps protec...	Running	Automatic	Local System
Windows Defender Firewall	Windows De...	Running	Automatic	Local Service
Windows Encryption Provid...	Windows En...		Manual (Trigg...	Local Service
Windows Error Reporting Se...	Allows errors...		Manual (Trigg...	Local System
Windows Event Collector	This service ...		Manual	Network Se...
Windows Event Log	This service ...	Running	Automatic	Local Service
Windows Font Cache Service	Optimizes p...	Running	Automatic	Local Service
Windows Image Acquisition ...	Provides ima...		Manual	Local Service

```
+ Stop-service
+ ~~~~~~~~~~~~
    + CategoryInfo          : InvalidData: (:) [Stop-Service], ParameterBindingValidationException
    + FullyQualifiedErrorId : ParameterArgumentValidationError,Microsoft.PowerShell.Commands.StopServiceCommand

PS C:\Windows\system32> get-service Audiosrv | stop-service
PS C:\Windows\system32>
```

5. **Using the pipeline, what is an operation that gets Windows audio service and then start the service? Run the command as an administrator and take a screenshot of the Windows Audio service on the Services manager with the changed status.**

   get-service Audiosrv | start-service



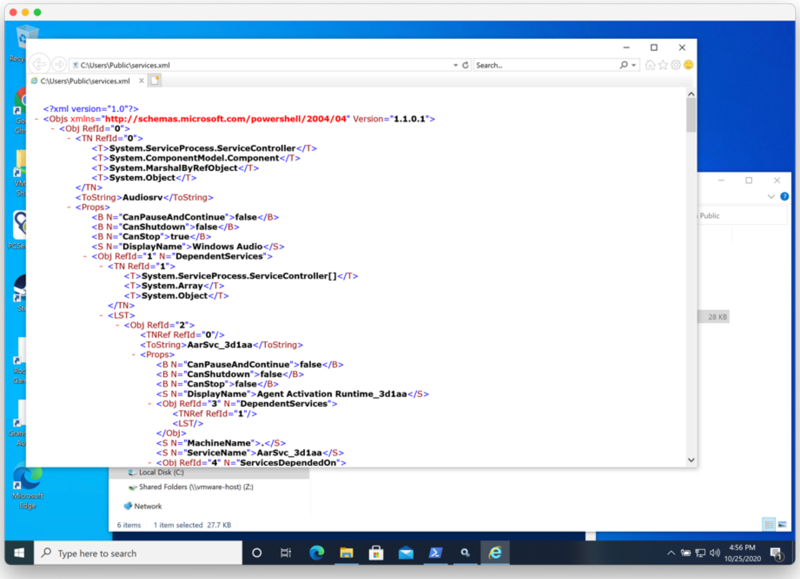
Services (Local)

Select an item to view its description.

Name	Description	Status	Startup Type	Log On As
VMware Tools	Provides sup...	Running	Automatic	Local System
Volume Shadow Copy	Manages an...		Manual	Local System
Volumetric Audio Composit...	Hosts spatial...		Manual	Local Service
WalletService	Hosts object...		Manual	Local System
WarpJITSvc	Provides a JI...		Manual (Trigg...	Local Service
Web Account Manager	This service i...	Running	Manual	Local System
WebClient	Enables Win...		Manual (Trigg...	Local Service
Wi-Fi Direct Services Connec...	Manages co...		Manual (Trigg...	Local Service
Windows Audio	Manages au...	Running	Automatic	Local Service
Windows Audio Endpoint B...	Manages au...	Running	Automatic	Local System
Windows Backup	Provides Wi...		Manual	Local System
Windows Biometric Service	The Window...	Running	Manual (Trigg...	Local System
Windows Camera Frame Ser...	Enables mul...		Manual (Trigg...	Local Service
Windows Connect Now - Co...	WCNCSVC h...		Manual	Local Service
Windows Connection Mana...	Makes auto...	Running	Automatic (Tri...	Local Service
Windows Defender Advance...	Windows De...		Manual	Local System
Windows Defender Antivirus...	Helps guard ...	Running	Manual	Local Service
Windows Defender Antivirus...	Helps protec...	Running	Automatic	Local System
Windows Defender Firewall	Windows De...	Running	Automatic	Local Service
Windows Encryption Provid...	Windows En...		Manual (Trigg...	Local Service
Windows Error Reporting Se...	Allows errors...		Manual (Trigg...	Local System
Windows Event Collector	This service ...		Manual	Network Se...

```
+ Stop-service
+ ~~~~~~~~~~~~
    + CategoryInfo          : InvalidData: (:) [Stop-Service], ParameterBindingValidationException
    + FullyQualifiedErrorId : ParameterArgumentValidationError,Microsoft.PowerShell.Commands.StopServiceCommand

PS C:\Windows\system32> get-service Audiosrv | stop-service
PS C:\Windows\system32> get-service Audiosrv | start-service
PS C:\Windows\system32>
```

6. **Export the audio service into an XML file, open it up on a web browser, and provide the screenshot of the XML.**



C:\Users\Public\services.xml

```
  <?xml version="1.0"?>
- <Objs xmlns="http://schemas.microsoft.com/powershell/2004/04" Version="1.1.0.1">
  - <Obj RefId="0">
    - <TN RefId="0">
        <T>System.ServiceProcess.ServiceController</T>
        <T>System.ComponentModel.Component</T>
        <T>System.MarshalByRefObject</T>
        <T>System.Object</T>
      </TN>
      <ToString>Audiosrv</ToString>
    - <Props>
        <B N="CanPauseAndContinue">false</B>
        <B N="CanShutdown">false</B>
        <B N="CanStop">true</B>
        <S N="DisplayName">Windows Audio</S>
      - <Obj RefId="1" N="DependentServices">
        - <TN RefId="1">
            <T>System.ServiceProcess.ServiceController[]</T>
            <T>System.Array</T>
            <T>System.Object</T>
          </TN>
        - <LST>
          - <Obj RefId="2">
              <TNRef RefId="0"/>
              <ToString>AarSvc_3d1aa</ToString>
            - <Props>
                <B N="CanPauseAndContinue">false</B>
                <B N="CanShutdown">false</B>
                <B N="CanStop">false</B>
                <S N="DisplayName">Agent Activation Runtime_3d1aa</S>
              - <Obj RefId="3" N="DependentServices">
                  <TNRef RefId="1"/>
                  <LST/>
                </Obj>
                <S N="MachineName">.</S>
                <S N="ServiceName">AarSvc_3d1aa</S>
              - <Obj RefId="4" N="ServicesDependedOn">
```

PUSH YOUR WORK TO GITHUB

Once you completed the Hands-on practice, **export this file to PDF**, then do the following to push your work to GitHub

Open the terminal from the VSCode by hitting the control + ~ key, make sure you are in the right path, for example: KimNguyen/Desktop/ISEC505/HOP03-KimNguyenMai/Module 3

Type the following command:

>>> **git add .** (to copy all changes you have made)
>>> **git commit -m "Submission for Module 3 – YOUR NAME"** (To add a message to your submission)
>>> **git push origin master** (to upload your work to Github)