Managing Current Location

Getting Your Current Location (Get-Location)

To determine the path of your current directory location, enter the `Get-Location` command:

PowerShellCopy

```
Get-Location
```

OutputCopy

```
Path
----
C:\Documents and Settings\PowerUser
```

Note

The `Get-Location` cmdlet is similar to the **pwd** command in the BASH shell. The `Set-Location` cmdlet is similar to the **cd** command in Cmd.exe.

Setting Your Current Location (Set-Location)

The `Get-Location` command is used with the `Set-Location` command. The `Set-Location` command allows you to specify your current directory location.

PowerShellCopy

```
Set-Location -Path C:\Windows
```

After you enter the command, you will notice that you do not receive any direct feedback about the effect of the command. Most Windows PowerShell commands that perform an action produce little or no output because the output is not always useful. To verify that a successful directory change has occurred when you enter the `Set-Location` command, include the **-PassThru** parameter when you enter the `Set-Location` command:

PowerShellCopy

```
Set-Location -Path C:\Windows -PassThru
```

OutputCopy

```
Path
----
C:\WINDOWS
```

The **PassThru** parameter can be used with many Set commands in Windows PowerShell to return information about the result in cases in which there is no default output.

You can specify paths relative to your current location in the same way as you would in most UNIX and Windows command shells. In standard notation for relative paths, a period (`.`) represents your current folder, and a doubled period (`..`) represents the parent directory of your current location.

For example, if you are in the `C:\Windows` folder, a period (`.`) represents `C:\Windows` and double periods (`..`) represent `C:`. You can change from your current location to the root of the C: drive by typing:

PowerShellCopy

```
Set-Location -Path .. -PassThru
```

OutputCopy

```
Path
----
C:\
```

The same technique works on Windows PowerShell drives that are not file system drives, such as **HKLM:**. You can set your location to the `HKLM\\Software` key in the registry by typing:

PowerShellCopy

```
Set-Location -Path HKLM:\SOFTWARE -PassThru
```

OutputCopy

```
Path
----
HKLM:\SOFTWARE
```

You can then change the directory location to the parent directory, which is the root of the Windows PowerShell HKLM: drive, by using a relative path:

PowerShellCopy

```
Set-Location -Path .. -PassThru
```

OutputCopy

```
Path
----
HKLM:\
```

You can type `Set-Location` or use any of the built-in Windows PowerShell aliases for `Set-Location` (cd, chdir, sl). For example:

PowerShellCopy

```
cd -Path C:\Windows
```

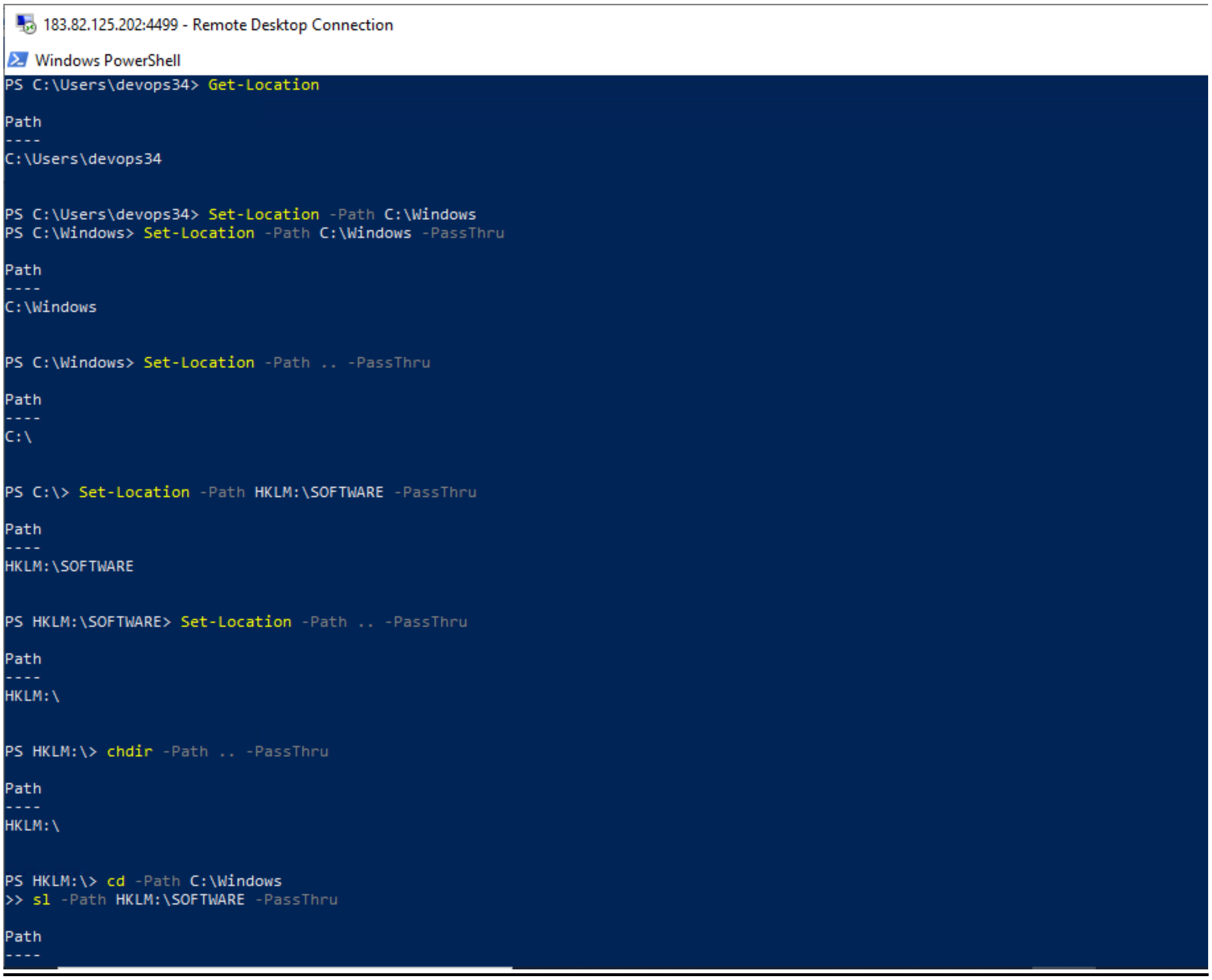
PowerShellCopy

```
chdir -Path .. -PassThru
```

PowerShellCopy

```
sl -Path HKLM:\SOFTWARE -PassThru
```

Output Screenshot:



```
PS C:\Users\devops34> Get-Location

Path
----
C:\Users\devops34


PS C:\Users\devops34> Set-Location -Path C:\Windows
PS C:\Windows> Set-Location -Path C:\Windows -PassThru

Path
----
C:\Windows


PS C:\Windows> Set-Location -Path .. -PassThru

Path
----
C:\


PS C:\> Set-Location -Path HKLM:\SOFTWARE -PassThru

Path
----
HKLM:\SOFTWARE


PS HKLM:\SOFTWARE> Set-Location -Path .. -PassThru

Path
----
HKLM:\


PS HKLM:\> chdir -Path .. -PassThru

Path
----
HKLM:\


PS HKLM:\> cd -Path C:\Windows
>> sl -Path HKLM:\SOFTWARE -PassThru

Path
----
```

Saving and Recalling Recent Locations (Push-Location and Pop-Location)

When changing locations, it is helpful to keep track of where you have been and to be able to return to your previous location. The `Push-Location` cmdlet in Windows PowerShell creates a ordered history (a "stack") of directory paths where you have been, and you can step back through the history of directory paths by using the complementary `Pop-Location` cmdlet.

For example, Windows PowerShell typically starts in the user's home directory.

PowerShellCopy

```
Get-Location

Path
----
C:\Documents and Settings\PowerUser
```

Note

The word **stack** has a special meaning in many programming settings, including .NET Framework. Like a physical stack of items, the last item you put onto the stack is the first item that you can pull off the stack. Adding an item to a stack is colloquially known as "pushing" the item onto the stack. Pulling an item off the stack is colloquially known as "popping" the item off the stack.

To push the current location onto the stack, and then move to the Local Settings folder, type:

PowerShellCopy

```
Push-Location -Path "Local Settings"
```

You can then push the Local Settings location onto the stack and move to the Temp folder by typing:

PowerShellCopy

```
Push-Location -Path Temp
```

You can verify that you changed directories by entering the `Get-Location` command:

PowerShellCopy

```
Get-Location
```

OutputCopy

```
Path
----
C:\Documents and Settings\PowerUser\Local Settings\Temp
```

You can then pop back into the most recently visited directory by entering the `Pop-Location` command, and verify the change by entering the `Get-Location` command:

PowerShellCopy

```
Pop-Location
Get-Location
```

OutputCopy

```
Path
----
C:\Documents and Settings\me\Local Settings
```

Just as with the `Set-Location` cmdlet, you can include the **PassThru** parameter when you enter the `Pop-Location` cmdlet to display the directory that you entered:

PowerShellCopy

```
Pop-Location -PassThru
```

OutputCopy

```
Path
----
C:\Documents and Settings\PowerUser
```

You can also use the Location cmdlets with network paths. If you have a server named FS01 with an share named Public, you can change your location by typing

PowerShellCopy

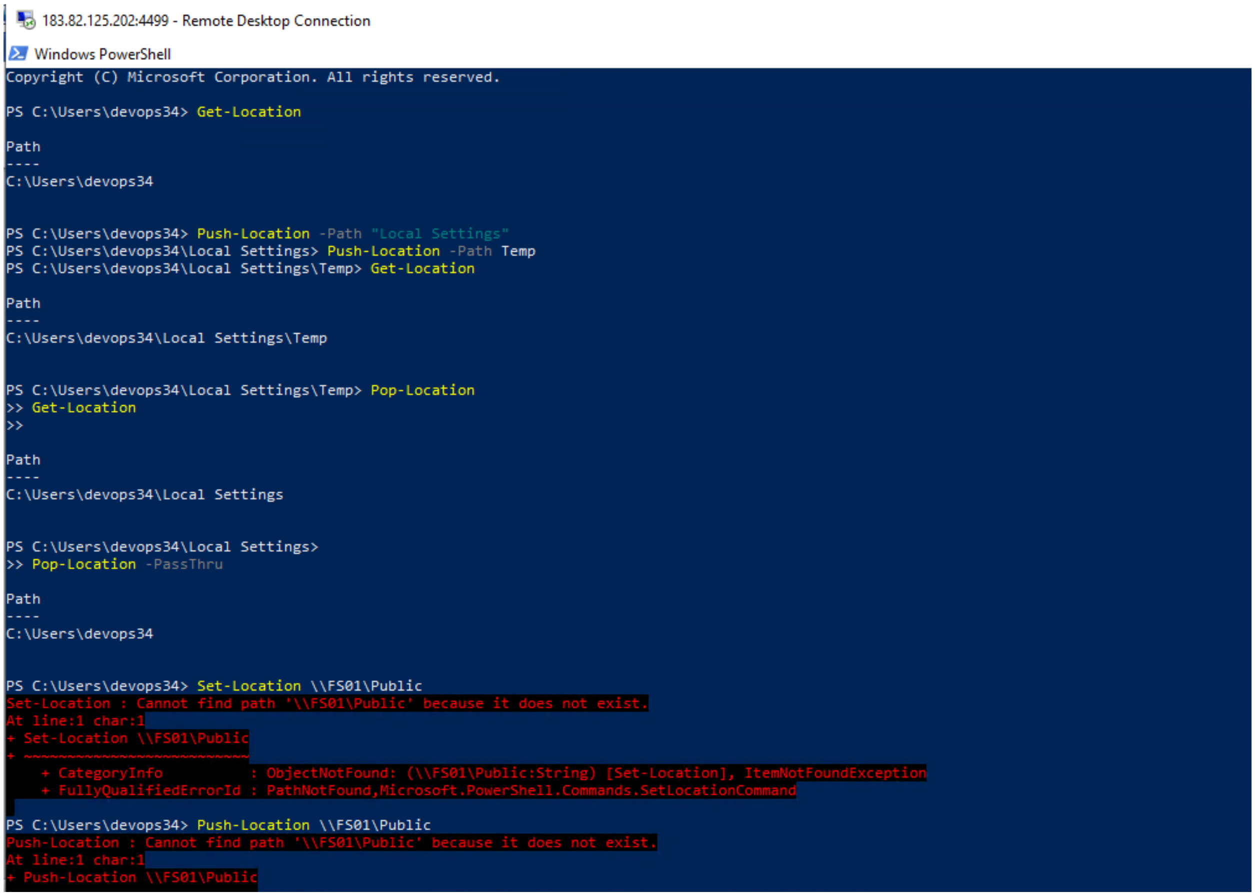
```
Set-Location \\FS01\Public
```

or

PowerShellCopy

```
Push-Location \\FS01\Public
```

<u>Output copy:</u>



183.82.125.202:4499 - Remote Desktop Connection

Windows PowerShell

```
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\devops34> Get-Location

Path
----
C:\Users\devops34


PS C:\Users\devops34> Push-Location -Path "Local Settings"
PS C:\Users\devops34\Local Settings> Push-Location -Path Temp
PS C:\Users\devops34\Local Settings\Temp> Get-Location

Path
----
C:\Users\devops34\Local Settings\Temp


PS C:\Users\devops34\Local Settings\Temp> Pop-Location
>> Get-Location
>>

Path
----
C:\Users\devops34\Local Settings


PS C:\Users\devops34\Local Settings>
>> Pop-Location -PassThru

Path
----
C:\Users\devops34


PS C:\Users\devops34> Set-Location \\FS01\Public
Set-Location : Cannot find path '\\FS01\Public' because it does not exist.
At line:1 char:1
+ Set-Location \\FS01\Public
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : ObjectNotFound: (\\FS01\Public:String) [Set-Location], ItemNotFoundException
    + FullyQualifiedErrorId : PathNotFound,Microsoft.PowerShell.Commands.SetLocationCommand

PS C:\Users\devops34> Push-Location \\FS01\Public
Push-Location : Cannot find path '\\FS01\Public' because it does not exist.
At line:1 char:1
+ Push-Location \\FS01\Public
```

You can use the `Push-Location` and `Set-Location` commands to change the location to any available drive. For example, if you have a local CD-ROM drive with drive letter D that contains a data CD, you can change the location to the CD drive by entering the `Set-Location D:` command.

If the drive is empty, you will get the following error message:

PowerShellCopy

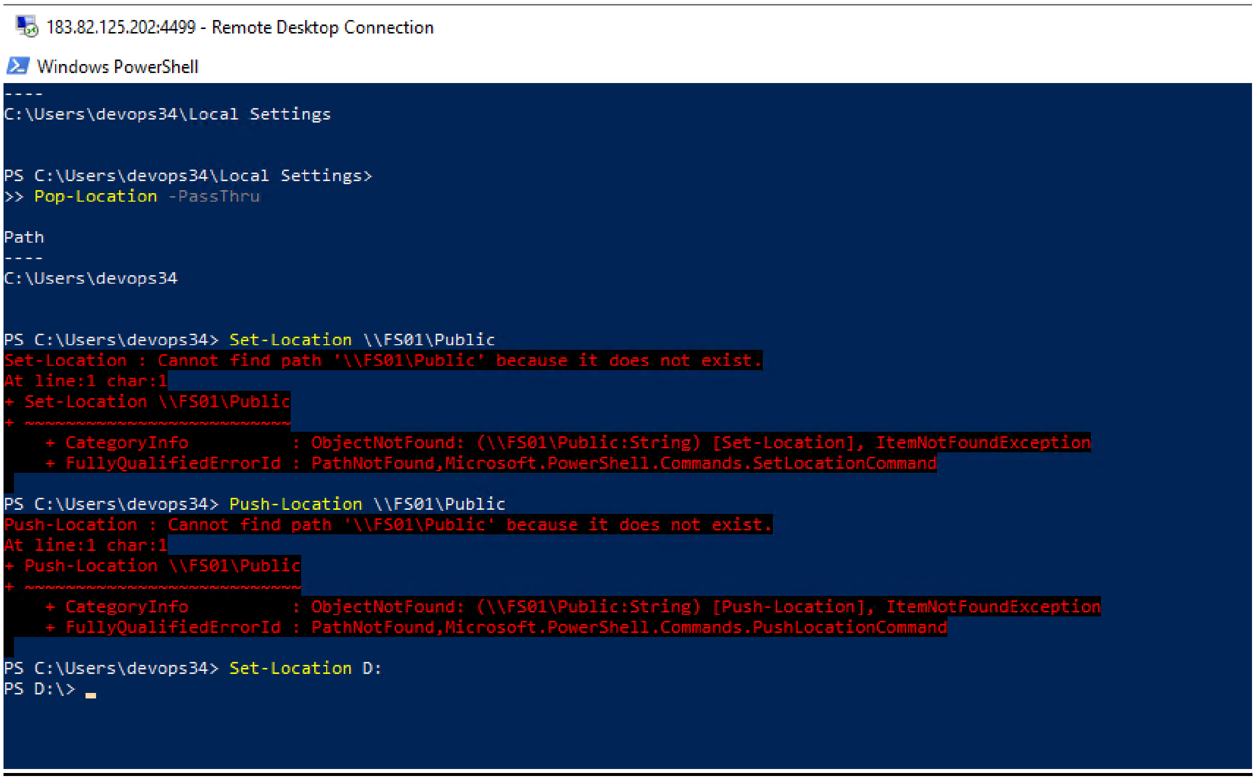
```
Set-Location D:
```

OutputCopy

```
Set-Location : Cannot find path 'D:\' because it does not exist.
```

When you are using a command-line interface, it is not convenient to use File Explorer to examine the available physical drives. Also, File Explorer would not show you the all of the Windows PowerShell drives. Windows PowerShell provides a set of commands for manipulating Windows PowerShell drives, and we will talk about these next.

Output :



```
----
C:\Users\devops34\Local Settings


PS C:\Users\devops34\Local Settings>
>> Pop-Location -PassThru

Path
----
C:\Users\devops34


PS C:\Users\devops34> Set-Location \\FS01\Public
Set-Location : Cannot find path '\\FS01\Public' because it does not exist.
At line:1 char:1
+ Set-Location \\FS01\Public
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : ObjectNotFound: (\\FS01\Public:String) [Set-Location], ItemNotFoundException
    + FullyQualifiedErrorId : PathNotFound,Microsoft.PowerShell.Commands.SetLocationCommand

PS C:\Users\devops34> Push-Location \\FS01\Public
Push-Location : Cannot find path '\\FS01\Public' because it does not exist.
At line:1 char:1
+ Push-Location \\FS01\Public
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : ObjectNotFound: (\\FS01\Public:String) [Push-Location], ItemNotFoundException
    + FullyQualifiedErrorId : PathNotFound,Microsoft.PowerShell.Commands.PushLocationCommand

PS C:\Users\devops34> Set-Location D:
PS D:\> _
```