**Powershell for Linux Systems - Tutorial by Ryan Tennant**

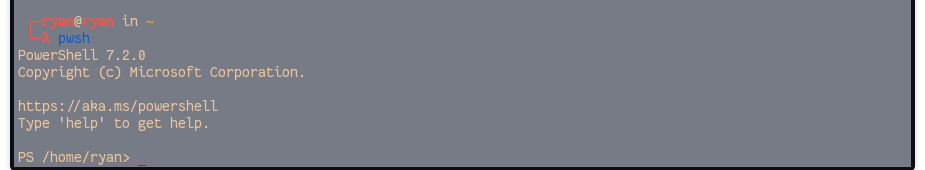
This tutorial is going to require you to have access to PowerShell on a Linux operating system, this can be in a virtual machine or a live machine. This tutorial is written for Linux PowerShell Core 7.2.0, but should also work for Mac OS.

Powershell is a great cross-platform scripting tool, but there are a few differences on other operating systems besides Windows. In this tutorial you will go over the significant adjustments that should be done when writing a script for cross-platform support.

The first step is going to be to install and launch powershell core, that’ll be a different process depending on the exact distro that you installed, but everything else will be the same for the rest of the tutorial.

Launch powershell by typing pwsh

Your terminal should display something like this.



Text

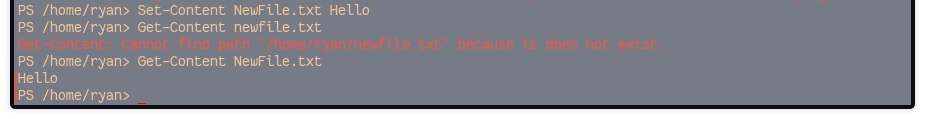
Description automatically generated

Let’s start with a simple change between Linux PowerShell and Windows PowerShell Core. Let’s take a look at some file handling.

Try running the command Set-Content NewFile.txt Hello

Make sure it saved by running the command Get-Content newfile.txt

Did it work? If it did, then good job running the correct command, Get-Content NewFile.txt. The difference is that on windows powershell doesn’t care about capitalization for filenames, but on other systems it does. This is what your output should look like.



Graphical user interface, text, application

Description automatically generated

Let’s do another experiment then. First, type $variable into the terminal to ensure that you don’t have a predetermined value there already. Next, assign $variable any value you want, I will use the string ‘Linux is cool!’ I did this by typing $variable = ‘Linux is cool!’ Then type $variable again to ensure that the value is saved. Then try $VARIABLE. Did you see the same value you originally typed? If so, what does that imply about how powershell treats variables compared to file structures?

Graphical user interface, text

Description automatically generated with medium confidence

Let’s try something else. Type ls into the command prompt. Do you notice anything different? Powershell uses the native commands for ls, cp, mv, rm, cat, man, mount, and ps. Try a couple of them now, for example cat NewFile.txt, cp NewFile.txt NewFile2.txt, rm NewFile\*

A picture containing text

Description automatically generated

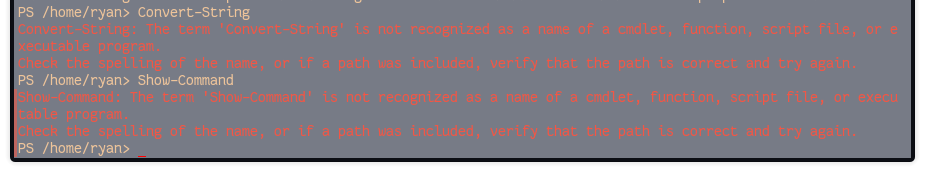
The following packages are only available on windows

* Microsoft.PowerShell.Archive
* Microsoft.PowerShell.Core
* Microsoft.PowerShell.Host
* Microsoft.PowerShell.Management
* Microsoft.PowerShell.Security
* Microsoft.PowerShell.Utility
* PackageManagement
* PowerShellGet
* PSDesiredStateConfiguration
* PSReadLine
* ThreadJob

Let’s test it out. Try any one of these commands from Microsoft.PowerShell.Utility

* Convert-String
* ConvertFrom-String
* Out-GridView
* Out-Printer
* Show-Command

You should get an error that looks like this



Text

Description automatically generated

You can check out the specifics of what Show-Command does here if you want to learn more and see that it is a real powershell command, just windows only. <https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell.utility/show-command?view=powershell-7.2>

Something to note, on Mac OS Powershell uses apple’s unified logging system, on linux it uses syslog.

Another important distinction is that powershell will not run any commands as sudo on linux, you can still run powershell as sudo to achieve the same effects.

Type exit to quit powershell.

Text

Description automatically generated

Check out this resource for a more detailed explanation of what’s different. <https://docs.microsoft.com/en-us/powershell/scripting/whats-new/unix-support?view=powershell-7.2>