|  |
| --- |
|  |
| Shell scripting in Linux |
| Visual basic Presentation on a IT subject |
|  |
| **Brock York** |
|  |

|  |
| --- |
|  |

Contents

[Introduction 3](#_Toc230946985)

[What is shell scripting 3](#_Toc230946986)

[What is needed to shell script? 3](#_Toc230946987)

[What platforms can I shell script on 3](#_Toc230946988)

[What does a shell script need to run 3](#_Toc230946989)

[Limitations of shell scripting 3](#_Toc230946990)

[How to write a shell script 4](#_Toc230946991)

[Advanced shell script breakdown 4](#_Toc230946992)

# Introduction

Shell scripting is writing a script for the shell of a operating system also called batch scripting for Microsoft Windows operating systems. Shell scripting is extremely popular and powerful because of it’s time saving feature, but it also suffers from some simple yet possibly deadly problems. To shell script all that is required is a shell that will support input from a file. Shell scripting is available on all Linux operating systems and Unix like operating systems. Microsoft Windows and OS/2 use batch scripting instead of shell scripting it is extremely similar to shell scripting it executes through the Windows Terminal with a different syntax and also less powerful.

# What is the shell

The shell also known as the command line is a interface used to input commands to the computer . Before Graphical user interfaces everything was done from the shell from software installation to running games. The shell is still used today especially in Unix like operating systems this is because instead of browsing through many Graphical menus or utilities a task can be completed in a few keystrokes.

# What is shell scripting

Shell scripting is a form of programming. You write commands into a file then execute it through the terminal. It has most features of a programming language i.e. for loops, if statements and so on. It can be used in many powerful ways. Although powerful shell scripts do have they’re limitations. It is interpreted meaning that it isn’t compiled into ones and zeros but interpreted on the fly line by line. A shell script is like a Windows Batch script but only much more powerful and has more features and commands available to it.

# What is needed to shell script?

To shell script you need a operating system that supports shell scripts and has a shell that supports input from a file. This is standard in most if not all Linux operating systems. That is the only software requirements knowing the syntax for the specific shell is also required.

# What platforms can I shell script on

Shell scripting is technically available on all platforms, because Microsoft windows has a similar technique called batch scripting. But really shell scripting can only be achieved on Unix derived operating systems like Linux, the BSD family and Solaris. Depending on which Operating system and Shell you are using depends on the commands and syntax available to you.

# What does a shell script need to run

A shell script runs the Operating System default shell and then parses lines of commands to the shell the shell then runs these commands in the order that it is given to it.

With the power of a programming language syntax behind it shell scripts can automate many processes. Shells scripts are commonly used on Linux to compile source code instead of typing out a thousand lines of commands into the shell to get the source code ready you just parse some simple Variables to the shell script and away it goes adjusting the source code to suit your computer.

# Somethings a shell script can be used for.

AutoMagically setting up a build environment to compile source code.

Running a system process

Starting/Stopping system processes.

File operations

# Limitations of shell scripting

With all of it’s goodness there is of course the bad side of shell scripting. Shell script can have simple syntax errors that can cause mass damage to a filesystem the Unix folklore command rm –rf \* / instead of rm –rf \*/ which will remove all files in the root of the filesystem. This can be a simple error when trying to delete a file in the current directory the shell script is in. Shell scripts can also have slow execution speeds due to nearly every command having to run a process. Each shell has it’s own syntax so writing a shell script that is compatible with multiple shells is near on impossible. Alot of the disadvantages caused by shell scripting is because of the shell scripts syntax itself.

# How to write a shell script

Writing a shell script is a very easy process

Firstly you need to create a new file in linux. The file should have the extension of .sh but doesn’t have to have it to work it’s more so that us humans can tell it is a shell script.

Now open the text file you have created. The very first line of your shell script will be the same of every shell script it defines it as a shell script. This is the firstline of every shell script:

!#/bin/bash

What this does it tells the shell to use the bash shell to run these commands. There are many shells available and the syntax for each differs.

After this line you can write the commands to be executed.

echo “Hello World”

echo “First shell script”

These two lines will print to the terminal saying what is enclosed in the quotes

# Advanced shell script breakdown

**Runs the shell**

!#/bin/bash

**Outputs text to the screen for the user to interpret**

#echo "Enter the name you want the image to be called"

**Waits for input from the keyboard and stores that input in a variable called imagename**

#read imagename

#Ask for where the image should be stored

**Outputs the text to the user**

#echo "Enter the path to where the image should be stored(Entering nothing will result in the default /media/storage/Images/PSXBackups)"

**Waits for input from the keyboard and stores that input in a variable**

#read imagepath

**#Check to see if nothing was inputed if so then set ImagePath to the default value of /media/storage/Images/PSXBackups**

#if [ -z $imagepath ];

# then

# imagepath=/media/storage/Images/PSXBackups

#fi

**#Make sure the image path is exists, If not tell the user file not found**

if [ -d $1 ]

then

cdrdao read-cd --read-raw --datafile "$1/""$2.iso" --device /dev/scd0 --driver generic-mmc-raw "$1/""$2.toc"

echo "Image Ripped successfully"

else

echo "Path to file not found"

fi