Shell Scripting with Bash

HyperionDev

Welcome

Your Lecturer for this session

Liano Naidoo

Lecture – Housekeeping

- The use of disrespectful language is prohibited in the questions, this is a supportive, learning environment - please engage accordingly.
- No question is daft or silly - **ask them!**
- There are Q/A sessions midway and at the end of the session, should you wish to ask any follow-up questions.
- You can also submit questions here: hyperiondev.com/sbc4-cs-questions
- For all non-academic questions, please submit a query: www.hyperiondev.com/support
- Report a safeguarding incident: hyperiondev.com/safeguardreporting
- We would love your feedback on lectures: https://hyperionde.wufoo.com/forms/zsgv4m40ui4i0g/

Previously:

- Set up a virtual machine in the cloud.
- SSH into remote host using putty

Objectives

- Understanding the basics of Bash scripting, including variables, loops, and functions.

SHELL Script

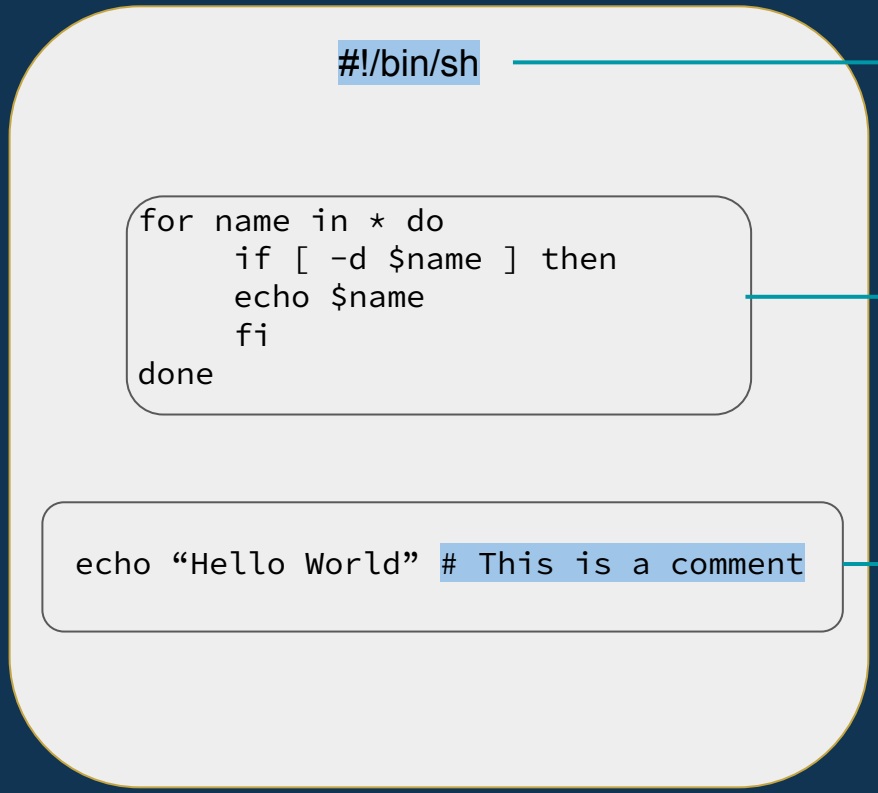
- Shell script is basically a computer program which divided into two part SHELL & SCRIPT.

- Shell is an Interpreter which converts low level & high level language in to machine language for Unix based Operating systems.

- Script stands for Set of Commands.

- Shell scripts can be used to automate repetitive or complicated tasks by running a series of commands.

Anatomy of a Shell Script



shebang!

UNIX! Commands, Syntax (looping, selecting/referring to variables)

Comments

Bash

- Bash is a popular Unix shell and command language in Linux and macOS. It includes scripting, file redirection, and piping, among other features.

- Shell scripting is a critical skill for cyber security professionals because it allows them to automate routine tasks, perform system monitoring, and detect and respond to security threats quickly.

BASH

THE BOURNE-AGAIN SHELL

Basic Bash Commands

- **ls** – Displays information about files in the current directory.
- **pwd** – Displays the current working directory.
- **mkdir** – Creates a directory.
- **cd** – To navigate between different folders.
- **rmdir** – Removes empty directories from the directory lists.
- **cp** – Moves files from one directory to another.
- **mv** – Rename and Replace the files
- **rm** – Delete files
- **uname** – Command to get basic information about the OS
- **locate**– Find a file in the database.
- **df** – Check the details of the file system
- **wc** – Check the lines, word count, and characters in a file using different options

- **locate**– Find a file in the database.
- **touch** – Create empty files
- **ln** – Create shortcuts to other files
- **cat** – Display file contents on terminal
- **clear** – Clear terminal
- **ps-** Display the processes in terminal
- **man** – Access manual for all Linux commands
- **grep-** Search for a specific string in an output
- **echo-** Display active processes on the terminal
- **wget** – download files from the internet
- **whoami-** Create or update passwords for existing users
- **sort-** sort the file content
- **cal-** View Calendar in terminal
- **whereis** – View the exact location of any command types after this command

Variables and Data Types

Variable declaration: Variables are declared in Bash using the syntax varname=value. x=10, for example, creates a variable named x with the value 10.

Assigning values to variables: The = operator can be used to assign values to variables. For example, x=20 assigns the variable x the value 20.

Data types (strings, integers, arrays): Bash supports several data types, including strings, integers, and arrays. Strings are sequences of characters enclosed in quotes, integers are numeric values, and arrays are collections of values.

If Statements

Conditional logic is carried out using IF statements. The syntax used to write them is if [condition]; then commands; fi.

Fi determines whether the value of x is greater than 5, for instance, if [$x -gt 5]; then echo "x is greater than 5". (==,!=,, >) Comparison operators Value comparison is done using comparison operators.

For instance, the operators -eq, -ne, -lt, and -gt determine whether two values are equal or not, respectively, as well as whether the first value is greater than the second.

Logical operators are used to combine multiple conditions. Examples include && and ||. The logical operators for AND and OR are && and ||, respectively.

Question:

What is the difference between logical and comparison operators?

Loops

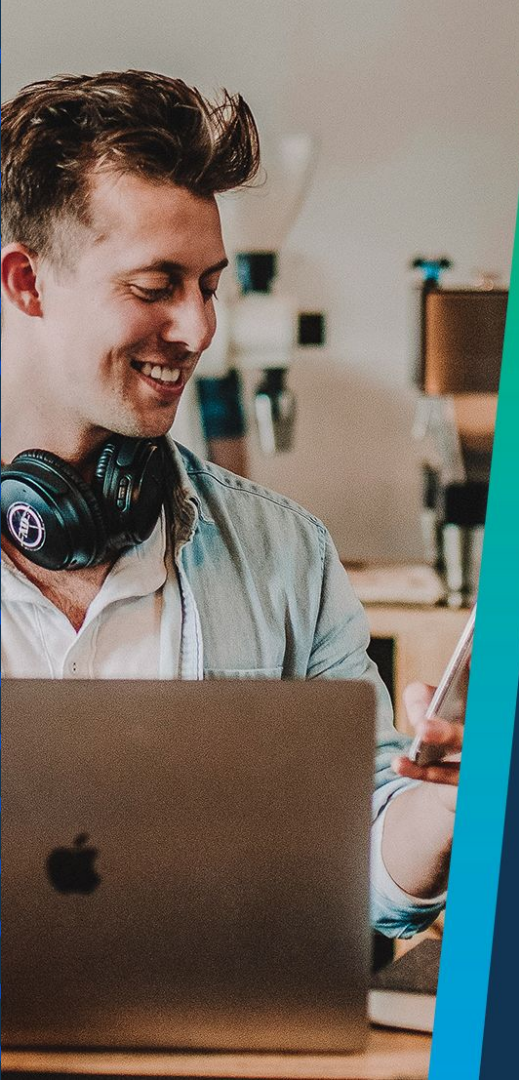
FOR loops are used to repeatedly iterate through a list of values. The syntax used for them is var in list; do commands; done. For instance, the statements for x in 1 2 3; do echo $x; done print out the numbers 1, 2, and 3.

While a condition is true, a series of commands are carried out by WHILE loops. The syntax used to write them is while [condition]; do commands; done. For instance, the output of the statements while [$x -lt 5]; do echo $x; x=$((x+1)); done outputs the numbers 0, 1, 2, 3, and 4.

Loops are useful for automating repetitive processes like processing lots of data or running a series of commands on many systems.

Questions and Answers

Questions around Client-Server Architecture and Protocols



Hyperiondev

Thank you for joining us