ProTech protechtraining.com *"Charting the Course ... ... to Your Success!"*

UNIX Shell Scripting

Course Summary

Description

This course introduces the UNIX operating system and enables you to write UNIX Shell Scripts. Students will incrementally build scripts of increasing complexity, and learn to perform tasks such as console I/O, text manipulation, arithmetic, process control and signal handling. The course is taught using the Korn shell, but also discusses Bash, Bourne and C shells. The course can be delivered on any version of UNIX, but typically Red Hat or Ubuntu Linux is used. Tools specific to particular flavors of UNIX (e.g. DTrace on Solaris) are not covered by default, but can usually be included if required.

Topics

- Review of Core Concepts
- Starting Shell Programming
- Regular Expressions
- Creating Useful Scripts
- Using the Sed Tool in Shell Scripts
- Using the Awk Tool in Shell Scripts
- Advanced Shell Programming
- Alternatives to Shell Scripting

Prerequisites

There are no prerequisites for this course.

Duration

Three days

Due to the nature of this material, this document refers to numerous hardware and software products by their trade names. References to other companies and their products are for informational purposes only, and all trademarks are the properties of their respective companies. It is not the intent of ProTech Professional Technical Services, Inc. to use any of these names generically

UNIX Shell Scripting

Course Outline

I. Review of Core Concepts
- A. The history and evolution of UNIX
- B. The filesystem, mounting devices and inodes
- C. Processes, threads, scheduling and signals
- D. Using the essential UNIX commands
- E. Comparing the different UNIX shells
- F. Picking the right shell for the job

II. Starting Shell Programming
- A. Why shell scripts are an essential tool
- B. Using the shebang line to pick an interpreter
- C. Declaring and working with shell variables
- D. Built in operators for testing and arithmetic
- E. Evaluating expressions using the expr command
- F. Testing variables with if and case
- G. Creating and controlling for, while and until loops
- H. Debugging your shell scripts

III. Regular Expressions
- A. How a Regular Expression Engine operates
- B. The difference between basic and extended regular expressions
- C. Options for using regular expressions in shell scripts
- D. Creating character classes and specifying multiplicities
- E. Meta-characters for specifying positions in the input
- F. Using parenthesis for grouping and sub matches
- G. The non-greedy versions of *, + and ?
- H. Using parenthesis that do not capture
- I. Applying modifiers to only part of the expression
- J. Using look-around assertions to match without capturing

IV. Creating Useful Scripts
- A. Creating formatted output with printf
- B. Working with environment variables
- C. Testing the exit status of a command
- D. Looping till a command is successful
- E. Discarding command output using /dev/null
- F. Using /dev/zero to create test files
- G. Testing the attributes of files
- H. Returning an exit status from your script
- I. Processing command line arguments
- J. Reading and parsing input from the keyboard
- K. Processing files one line at a time
- L. Creating and working with arrays

V. Using the Sed Tool in Shell Scripts
- A. Understanding how Sed transforms text input
- B. Working with commands and addresses
- C. Manipulating the contents of the pattern space
- D. Using the hold buffer to copy and paste text
- E. Sed commands for selection and iteration

VI. Using the Awk Tool in Shell Scripts
- A. Understanding how Awk processes text
- B. Breaking up input into records and fields
- C. Writing actions to processes records
- D. Using regular expressions in actions
- E. Awk commands for selection and iteration
- F. Adding functions to Awk scripts

VII. Advanced Shell Programming
- A. Trapping and handling signals
- B. Running other programs via exec
- C. Starting, monitoring and deleting processes
- D. Breaking up scripts into functions
- E. Security issues with shell scripts

VIII. Alternatives to Shell Scripting
- A. How Perl improves and extends shell scripting
- B. Object oriented scripting using Ruby and Python

Due to the nature of this material, this document refers to numerous hardware and software products by their trade names. References to other companies and their products are for informational purposes only, and all trademarks are the properties of their respective companies. It is not the intent of ProTech Professional Technical Services, Inc. to use any of these names generically