

UNIX Shell Programming

4 days

50% lecture, 50% labs

Basic Level

Overview: UNIX Shell programming is useful for automating everyday tasks by executing UNIX commands from within a script. Shell programming is especially popular with system administrators, but can be helpful to programmers, database administrators, CGI programmers, and end users as well.

Benefits:

Upon completion of this course, students will be able to:

- read, write, customize, and debug Shell scripts
- understand the Shell's role and responsibilities; e.g. handling processes, command execution, subshells, pipes, I/O redirection, wildcards, job control, etc.
- customize Shell initialization files, such as the `.cshrc`, `.profile`, `.login`, etc.
- use the power of regular expressions to extract data from files and pipes with `grep`, `sed` and `awk` (`nawk`, `gawk`)
- understand how to use the Shell interactively
- use the Shell as a programming language including constructs such as loops, decision making constructs, switch statements, functions
- test file attributes
- debug scripts

Intended Audience: *UNIX Shell Programming* is recommended for people who are familiar with basic UNIX and want to learn more sophisticated commands and how to integrate them in Shell scripts in order to automate tasks.

Prerequisites: Students need to be familiar with basic UNIX commands, such as `cd`, `ls`, `pwd`, `cp`, `mv` etc. and be able to navigate the directory tree. They should also be familiar with one of the UNIX text editors, such as `vi`, `emacs`, or `textedit`.

Suggested follow-on course: Perl Programming and Advanced Perl Programming

Training Approach: This is an intensive, interactive course, which is approximately 50% lecture and 50% lab. Questions are highly encouraged. On the final day, students are given access to a zipped file containing all of the solutions to the labs and the examples used throughout the notebook.

Course Outline

Day 1

Module 1-- Introduction to the Shell

- What is Shell?
- The Three Major Shells
- History of the Shell
- System Startup
- The Shell as a Command Interpreter
- Definition of a Shell Script

Lab Exercise 1

Module 2 -- The UNIX Tool Box

- What are Regular Expressions?
 - Regular Expression Metacharacters and how they work
 - Anchors
 - The Dot
 - Character Sets
 - Metasymbols
 - Greedy Quantifiers
 - Alternation
 - Grouping
 - Capturing
 - Repeating
- The *grep* Family (*grep*, *egrep*, *fgrep*)

Lab Exercise 2

- The *sed* Editor

Day 2

Lab Exercise 3

- The *awk* Programming Language

Module 2 --(Awk continued)

Lab Exercise 4

Module 3

- **The Interactive Shell**
 - Startup
 - The Environment
 - Command line
 - Exit status
 - Command line history
 - Aliases
 - Job Control
 - Metacharacters
 - Filename Substitution
 - Variables
 - Quoting
 - Command Substitution
 - Redirection and Pipes

Lab Exercise 5

Day 3

Module 4 -- Programming the Shell

- Construction of a Shell Script
- Execution of a Shell Script

Lab Exercise 6

- Positional Parameters
- Variables
- Reading input

Lab Exercise 7

- Conditional Constructs (if if/else)
- Arithmetic
- File testing

Day 4

Lab Exercise 8

- The Case/Switch Mechanism
- Loops

Lab Exercise 9

- Arrays
- Functions

Lab Exercise 10

- Debugging
- Handling Signals
- Review (A complete script is provided to review course content)

For more information, contact:

Tom Wille

tw@tm-associates.com

503-656-4457

TM Associates, Inc.

www.tm-associates.com