

Shell Programming for System Administrators

(SA-245-SHELL, 5 days)



Course Description

The Shell Programming for System Administrators course provides students with the skills to read, write, and debug UNIX shell scripts. The course begins by describing simple scripts to automate frequently executed commands and continues by describing conditional logic, user interaction, loops, menus, traps, and functions. This course is intended for system administrators who have mastered the basics of a UNIX Operating Environment (OE) such as the Solaris OE or Linux and who would like to read and understand the various boot scripts and write their own scripts to automate their day-to-day tasks. This course explores, in detail, the Bourne and Korn shell scripting languages.

Who Can Benefit

Students who can benefit from this course are system administrators, system programmers, UNIX operators, database administrators, and Web administrators.

Prerequisites

To succeed fully in this course, students should be able to:

- Use basic UNIX commands, such as rm, cp, man, more, mkdir, ps, and chmod
- Create and edit text files in vi or a text editor

Note: These skills are typically acquired through attendance in the System administration courses.

Skills Gained

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

- Write real world administration scripts.
- Use regular expressions with the grep, sed, and nawk
- Manipulate text files with grep, sed, and nawk
- Write reporting scripts
- Maintain existing scripts.
- Use flow control constructs, such as branching and looping
- Customize system-wide shell initialization files
- Develop and debug scripts
- Use local and environmental variables and shell metacharacters in scripts
- Use the exit status of a command to determine if the command succeeded or failed
- Develop interactive scripts
- Write a script that uses functions
- Write a script that uses a trap to catch a signal
- Access and process command-line arguments passed into a script
- Write sed scripts to perform noninteractive editing tasks
- Write nawk scripts to manipulate individual fields within a record
- Write nawk scripts to write reports based upon an input file
- Perform string manipulation and integer arithmetic on shell variables
- Develop a USAGE message to display when a script is invoked incorrectly

Related Courses

Before:

- Basic Unix System administration courses of any version (eg. SA-100-S10)

After:

- Intermediate System Administration for the Solaris 10 Operating System (SA-200-S10)

REGISTRATION AND INFORMATION

education@ecs.com.sg
www.ecs.com.sg/training



Shell Programming for System Administrators

(SA-245-SHELL, 5 days)

Course Content

Module 1 - UNIX Shells and Shell Scripts

- Describe the role of shells in the UNIX environment
- Describe the standard shells
- Define the components of a shell script
- Write a simple shell script

Module 2 - Writing and Debugging Scripts

- Start a script with #!
- Put comments in a script
- Change permissions on a script
- Execute a script
- Debug a script

Module 3 - The Shell Environment

- Use Bourne and Korn shell variables
- Assign values to shell variables
- Display the value of shell variables
- Make variables available to subprocesses using the export statement
- Display the value of environment variables
- Unset shell and environment variables
- Customize the user environment using the .profile file
- Perform arithmetic operations
- Create and use aliases
- Display aliases and the values assigned to them
- Define the built-in aliases
- Customize the Bourne and Korn shell environments
- Use the tilde expansion and command substitution features of the Korn shell

Module 4 - Regular Expressions and the grep Command

- Use and describe regular expressions
- Describe the grep command
- Use the grep command to find patterns in a file
- Use the regular expression characters with the grep command

Module 5 - The sed Editor

- Use the sed editor to perform noninteractive editing tasks
- Use regular expression characters with the sed command

Module 6 - The awk Programming Language

- Use awk commands from the command line
- Write simple awk programs to generate data reports from text files
- Write simple awk programs to generate numeric and text reports from text files

Module 7 - Conditionals

- Use the exit status of a command as conditional control
- Use the "if" statement to test a condition
- Pass values using command-line arguments (positional parameters) into a script
- Create USAGE messages
- Place parameters on the command line
- Use conditional if, then, elif, else, and fi constructs
- Use exit, let, and test statements ([[]], " ")
- Apply the &&, ||, and ! Boolean logic operators
- Use the case statement

Module 8 - Interactive Scripts

- Use the print and echo commands to display text
- Use the read command to interactively assign data to a shell variable
- Read user input into one or more variables, using one read statement
- Use special characters, with print and echo, to make the displayed text more user friendly
- Create a "here" document
- Use file descriptors to read/write to multiple files

Module 9 - Loops

- Write scripts that use for, while, and until loops
- Write a script using the select statement
- Describe when to use loops within a script
- Generate argument lists using command, variable, and file-name substitution

Module 10 - Advanced Variables, Parameters, and Argument Lists

- Declare strings, integers, and array variables
- Manipulate string variables
- Change the values of the positional parameters using the set statement within a script
- Use Korn shell arrays
- Set default values for parameters & Use the Korn shell built-in let, print, set, and typeset statements

Module 11 - Functions

- Create user-defined functions in a shell script
- Create, invoke, and display functions from the command line
- Pass arguments into a function
- Call functions from special (function) files that are saved in one or more function directories
- Describe where functions are available for use

Module 12 - Traps

- Describe how the trap statement works
- Include trap statements in a script
- Use the trap statement to catch signals and handle errors

REGISTRATION AND INFORMATION

education@ecs.com.sg
www.ecs.com.sg/training

