Korn Shell and Bash Shell Programming

Duration: 3 days *(Face-to-Face & Remote-Live)*, or 21 Hours *(On-Demand)*

Price: $1695 *(Face-to-Face & Remote-Live)*, or $1495 *(On-Demand)*

Discounts: We offer multiple discount options. Click here for more information.

Delivery Options: Attend face-to-face in the classroom, remote-live or on-demand training.

Students Will Learn

- How to write and run shell scripts
- Using conditional constructs to control script execution
- Manipulating strings
- Command-line processing
- Using regular expressions
- String processing utilities: `sed`,`grep` and `awk`
- Counting words, lines and characters
- Working with compression utilities
- Writing functions
- Managing processes
- Using the `ksh` and `bash` commands
- Working with UNIX I/O streams

Course Description

This hands on Korn and Bash Shell scripting course provides a comprehensive introduction to writing Korn and Bash shell scripts. Besides covering fundamental syntax for program flow control, variable assignment and substitution, I/O control, and mathematical expressions, it emphasizes the powerful features of these shells, including built-in string operators, variable typesetting/conversion, functions, and coprocess communication and control. The creative use of standard UNIX and Linux utilities within scripts to solve problems is stressed throughout. The course is designed for the administrators and programmers who are developing, testing, or integrating software on UNIX or Linux, as well as for advanced UNIX or Linux users. Both the commonalities and differences between the Korn and Bash shells are examined, and students will have the opportunity to learn from examples coded in both shells. Comprehensive hands on exercises are integrated throughout to reinforce learning and develop real competency.

Course Prerequisites

Familiarity with UNIX file system and commands. Students who are not familiar with UNIX file system and commands should register for the course UNIX/Linux Fundamentals.

Course Overview

Introduction

- Comparing Popular Shells
- Korn Shell Compatibility for Bash
- Korn and Bash Shell Advantages
- Korn and Bash Shell Disadvantages
- Versions of `ksh` and `bash`
- Shell Command Syntax
- Comments
- Continuing a Command on Multiple Lines
- Special Characters
- Pattern Matching
- Complex Pattern Combinations
- Tilde Expansion

Process Management

- Process Priority
- Background Jobs
- Signals
- Traps
- DEBUG and EXIT
- `ksh` Co-Processing
- `bash` Co-Processing
- Pausing
- Time and Date
- Scheduling Execution
- Run Commands Later
- Aliases
- How the Shell Finds Commands
- Command History and Editing

Writing Scripts

- Shell Programming Language
- Reserved Words
- How to Write and Run Scripts
- Debugging
- Data Types
- Using Variables
- The `typeset` Command
- Constants
- Special Predefined Variables
- Printing to the Screen `(echo, print)`
- The `printf` Command
- Accepting Input with `ksh`
- Accepting Input with `bash`
- Arithmetic Operators
- Floating-Point Numbers
- Bitwise Operators
- Grouping Operations
- Binary, Octal, and Hexadecimal
- Indexed Arrays
- Quoting
- Command Substitution

Advanced Customization of the Shell Environment

- Command Line Options
- The `set` command
- Parents and Children
- Inheritance
- Exporting
- Dot Scripts
- Startup Scripts
- Environment Variables
- Setting Prompts in `ksh` and `bash`

Programming Logic

- Conditional Expressions
- Testing Strings
- Testing Options
- Numeric Conditions
- Testing Completion Status
- Truth in Korn and Bash Shells
- The case Statement
- Loops: `while` and `until`
- The `for` Loop
- Menus and the `select` Statement

Advanced I/O with Streams

- Redirection Review
- Opening Additional Streams
- Redirection Operators
- Inheriting and Duplicating Streams
- Caution with `exec`
- `here` Documents

- Redirection and Loops
- When to Use Different Constructs

String Manipulation

- String Comparison
- String Relations
- Concatenation
- Substring Manipulations
- Wildcards and Pattern Matching

Security

- Process Ownership
- `suid` and `sgid`
- Restricted Shells
- Other Security Features
- The `newgrp` Command
- Statement `blocks`
- `case` Statement

Command Line Processing

- Getting Data Into Scripts
- Manipulating Positional Parameters `(set, sort, shift)`
- Analyzing Switches with `getopts`
- Options
- Reserved Variables `(OPTARG, OPTIND, IFS)`
- Setting Default Values

Performance and Porting Issues

- Improving the Performance of Scripts
- Timing Commands and Scripts
- System Resources
- Setting Limits
- Portability Issues

Overview of File Manipulation Utilities

- Editing a File from a Script
- Scripting with `ed` or `sed`
- UNIX and Linux Utilities to Manipulate Files
- Regular Expressions
- `grep` and `egrep`
- The Stream Editor `sed`
- Sorting in Scripts
- Generating Reports with `awk`
- Splitting Large Files
- Counting Words, Lines, and Characters
- Transforming File Contents
- Extracting Text Strings

Additional File Processing Commands

- Examining and Comparing Files
- Reporting Differences Between Files
- Comparing Files of Any Format
- Displaying Data in Octal and Hex
- Compressing Data
- Converting File Formats

Functions

- Writing Functions
- Returning Value from Functions
- Returning String Output
- Local and Global Variables
- Defining Functions
- `ksh` Autoload Functions

Compound Commands

- Pipelines
- Command Lists
- And and Or Lists
- Background Jobs
- Command Grouping
- I/O Redirection