

Introduction to Perl Programming: Hands-On - 4 Days

Course 431 Overview

You Will Learn How To

- Quickly create effective, reusable Perl scripts
- Take advantage of Perl 5's many small operators and built-in functions
- Structure code with user-generated subroutines
- Exploit the many additional object-oriented packages available on the Internet
- Build real-world scripts that can be used on UNIX, Linux and Win32 platforms
- Solve complex data manipulation tasks with ease

Course Benefits

Perl is a powerful object-oriented scripting language used extensively with UNIX, Linux, Win32 and the Internet. In this course, you learn to write reusable scripts with Perl 5. Through intensive exercises, you learn to use Perl in your operating system and application environments as well as apply built-in functions of the language and make use of external modules.

Who Should Attend

UNIX, Linux and Windows administrators, software engineers, programmers and power users. Previous experience with a procedural or object-oriented programming language is assumed. Knowledge of UNIX scripting tools and regular expressions is helpful.

Hands-On Training

During this course, you perform extensive exercises that provide in-depth, hands-on experience. Exercises include:

- Parsing and manipulating text with Perl regular expressions
- Reading and writing standard I/O streams and real-world data files
- Extracting and arranging information from multiple files
- Empowering scripts by using supplied and third-party OO modules
- Building network-capable scripts to access e-mail, database and Web servers

Introduction to Perl Programming: Hands-On - 4 Days

Course 431 Outline

Introduction to Perl

- Perl strengths: ease of programming with operators, flexibility, execution speed
- Reusable tool building for system admin, text manipulation, the Internet
- Popular application areas: networking, text filtering, Web application programming

Perl Syntax Fundamentals

Running Perl programs

- Invoking Perl scripts under UNIX/Linux with "shebang" syntax
- Starting Perl scripts under Win32 via command line associations
- Getting help with **perldoc**

Variable types and contexts

- Scalars vs. lists vs. hashes
- Strings and interpolated strings
- Making sense of special variables
- Working in the right context

Compound data structures

- Generating references to named variables
- Creating references to anonymous data
- Building multidimensional arrays
- Working with multidimensional hashes

Managing files and user input

- Handling standard I/O streams
- Defining and using file handles
- Parsing command line arguments
- Reading and writing data files

Pattern Matching and Operators

Perl regular expressions

- Extracting important text information
- Building on UNIX regular expressions
- Altering data with substitutions
- Performing global and case-insensitive matches

Perl's small operator groups

- Manipulating arithmetic expressions
- Replicating and growing strings
- Saving time with assignment operators
- Obtaining file attributes
- Making decisions with logical operators
- Establishing and using ranges

Looping, Decisions and Flow Control

Perl support for conventional flow control constructs

- Making decisions with **if/else/elsif**
- Creating loops with **do, while, until, for** and **foreach**

Perl-specific constructs

- **if** and **unless** as statement modifiers
- Altering flow with **next** and **last**
- Constructing switch statements

Subroutines

Writing subroutines

- Defining and calling a subroutine
- Passing and receiving parameters
- Returning values to the caller

Making data work for you

- Localizing variables: **my** and **local**
- Accessing global variables
- Extracting local variables with **shift**

Built-in and Add-on Functions

Common data manipulation requirements

- String functions for text manipulation
- Processing arrays with list functions
- Arranging information with **sort**
- Sorting data on multiple fields

I/O and tool building

- Manipulating file system entries
- Reading binary files
- Dissecting and creating records with **split** and **join**
- Formatting tabular output

Perl and Object Orientation

How Perl implements object orientation

- An introduction to OO in Perl
- Methods, classes and constructors
- Surveying and obtaining third-party packages from CPAN

Accessing OO packages

- How to use **use**
- Defining a schema to employ OO modules
- Calling methods with the **->** syntax
- Passing initialization parameters