Perl Programming for Systems Administrators

An introduction on how to program in Perl for Systems Administrators **(5 days)**

Synopsis

This is a specialist course in Perl programming for system administrators who want to use the language to get their jobs done more quickly and efficiently.

This Perl course includes a thorough grounding in generic Perl programming before moving on to specialised system administration tools and techniques.

Who Should Attend

Programmers who need to administer IT systems on any of today's major platforms and in multi-platform environments.

Delegates with backgrounds in C like languages will recognise elements of Perl syntax and can use much existing knowledge, but must beware the temptation to translate literally from C/C++. Perl's native way of doing things is invariably more efficient and better adapted to specialist system administration tasks.

System administrators with limited scripting (rather than pure programming) experience may need more than 4 days to benefit fully from this course.

Like all our Perl courses, this course is designed for cross-platform system administration, but Unix System Administrators will benefit particularly by recognising tools which have been incorporated into Perl from their favourite Operating System.

Course Contents

Preparing to Learn Perl

- Things you need to know and do in order to run Perl and learn Perl programming
- A module designed for complete beginners
- The perl compiler/interpreter
- Perl under Unix/Linux
- Perl under MS Windows 2000/NT/95/98/ME (perl.exe)
- ActiveState Perl
- Making programs executable
- Perl from the command line
- Specify the perl compiler/interpreter (#!)
- Using plain text
- Writing a very simple program
- Running a very simple program
- Basic syntax

Perl: The Absolute Minimum

- Enough of the language to get started
- The print command
- Variables
- Scalars — numbers and 'strings'
- Assignment
- Simple conditional tests — if "strings" "\n"
- Lists
- Arrays — for storing lists
- foreach
- Hashes
- Other loops: while, for, do, until
- Arrays — the rest
- Simple input, e.g. while(<>)
- Functions overview — recognising, writing, using
- Regular expressions — perlre
- Simple file handling — open, print
- Subroutines — parameters in and out, listification, local variables
- Help — perldoc, books, web

Perl: Beyond the Basics

- More flow control
- Statement modifiers
- Quoting mechanisms — `qq()`, etc
- Here documents
- Uppercase/lowercase conversion
- Splitting strings into lists
- Joining lists into strings
- Filtering lists with `map`
- Sorting lists
- The importance of context
- Assignment shortcuts
- Scoping rules
- Special variables

Complex Data Structures & References

- Limits of flat lists
- Nesting arrays
- Array references
- Anonymous arrays
- Named array references
- Passing multiple arrays to/from functions
- Hashes of arrays
- Hash references
- Arrays of hashes
- Hashes of hashes
- Complex nested data structures
- Code references
- Dispatch tables

Finding Out More For Yourself

- How to read Perl's documentation
- Where to find more information
- Knowing what's out there to look for
- FAQs

The Perl Debugger & Debugging Perl

- Avoiding bugs
- Perl's built-in debugger
- Invoking the debugger
- What you can do with the debugger
- Understanding the debugger's command line interface
- Knowing the debugger's basic command set
- Exploring some extended functions
- Graphical debuggers
- Alternative debugging techniques

Using Perl Modules from CPAN

- The Comprehensive Perl Archive Network (CPAN)
- Why effective Perl programmers are efficient CPAN users
- CPAN's philosophy
- Finding modules
- Installing Modules
- Using modules
- Some particularly useful modules

Command-Line Perl

- General principles
- Using Perl as a filter
- Awk-like Perl
- Many real world examples
- Many examples using regular expressions
- Command line flags

A Whistle-Stop Tour of the World of Perl

- Wheels you don't need to re-invent
- Common recipes
- Common pitfalls

Advanced File Processing with Perl

- Types of `open`
- Filehandles
- Reading line by line
- Reading paragraph by paragraph
- Reading entire files
- Special variables
- The flip-flop operator (`..`)
- File test functions
- Pipes

Perl Security Issues

- Potential security pitfalls
- Coding for security
- Taint checking
- Dangerous environment variables
- File input
- Set-user-id Perl programs
- Permissions and users

System interaction

- Connecting to other programs
- Unsafe pipes
- Using `IO::Pipe`
- Grabbing a program's output
- Other ways to run programs

Managing Users and Processes with Perl

- User identity across platforms
- Process control
- Scheduling events
- Managing disk quotas
- Querying filesystem usage
- Monitoring file operations
- Monitoring network operations
- Related perl modules

Managing Networks with Perl

- Host Files
- NIS and NIS+
- DNS
- WHOIS
- LDAP
- ADSI
- Sending and receiving email
- Related modules

Logging with Perl

- Text logs
- Binary logs
- Handling state
- Disk usage problems
- Log analysis
- Log munging
- Logging related modules

Delivery

This is a hands-on practical workshop based around the coding of real-world solutions to real-world problems.

Erion