Hands-On

Perl Scripting

Course Description

This hands on Perl scripting class provides a thorough introduction to the Perl programming language, teaching attendees how to develop and maintain portable scripts useful for system management and data manipulation.

Emphasis is placed on built-in subroutines that can be used to help conveniently build fast, portable and efficient scripts. Extensive hands on exercises provide practice in report creation, pattern matching, string manipulation, file I/O, command line processing, and debugging.

Students are shown how to extend Perl's basic functionality with packages and loadable modules. Attendees who also want an in-depth introduction to CGI Programming and the use of the Perl DBI module for database access should attend the five-day Perl Programming and CGI Scripting course instead.

Students Will Learn

- **Language Syntax**
- **Pattern Matching**
- **Flow Control**
- **Arrays & Hashes**
- **I/O and File Manipulation**
- **Subroutines**

Prerequisites

Prior scripting experience or knowledge of fundamental programming concepts.

Course Outline

Introduction to Perl

- Origin and Design Goals of Perl
- Overview of Perl Features
- Getting and Installing Perl

- Accessing Documentation via perldoc
- HTML-Format Reference Documentation
- Perl Strengths and Limitations

Getting Started With Perl

- Explicit Invocation of the Perl Interpreter
    - Running Perl on UNIX vs. Windows
    - Running Perl from the Command Line
    - Using Command Line Options
    - Using Debug Mode
- Implicit Invocation of the Perl Interpreter
- Running and Debugging Perl Scripts
- Simple and Compound Statements
- Fundamental Input Techniques
- Using the print Function to Generate Standard Output

Using Variables

- Scalar Variables
- Introduction to Standard Data Types
- Retrieving Standard Input Using the Default Variable $_
- Assigning Strings and Numbers to Scalar Variables
- Declaring Constants for Persistent Values
- Using strict to Declare Variables

Pattern Matching in Perl

- Regular Expressions in Perl
- Using Pattern Matching Operators
- Altering Data with Substitutions in Regular Expressions
- Using Backreferences to Capture Data from Regular Expression Matching
- Global and Case-Insensitive Matches
- Altering Data with Character Translation
- Using Variables in Patterns

Operators

- Introduction to Fundamental Operators
- Operator Precedence and Associativity
- Using the Ternary Operator ?: as a Shortcut for the if Statement
- Using and <> File I/O Operators for Standard Input/Output
- Using the Shortcut Operators +=, -=, *=, /=

String Manipulation

- String Comparison
- String Relations
- Concatenation
- Substring Manipulation
- Using chomp and chop to Eliminate EOL Characters
- Escape Characters for Formatting
- String Manipulation Functions

Flow Control: Conditional Statements and Looping

- Conditional Expressions and Logical Operators
- if/else/elsif and unless
- Constructing switch/case Equivalent Expressions
- while Loops and do Loops
- for and foreach Loops
- Labels

- Altering Program Flow with next, last, and redo
- Trapping Errors with the eval Function
- Terminating a Script with exit

Subroutines and Parameters

- Simplifying Scripts with Subroutines
- Defining and Calling a Subroutine
- Passing Arguments by Value
- Passing Arguments by Reference
- Using return to Return a Value
- Controlling Variable Scope using my and local Keywords

Arrays and Hashes

- Defining Numeric Index Arrays
- Defining Associative Arrays
- Sorting Arrays with the sort Function
- Adding and Deleting Items Using push, pop, shift, and unshift
- Using slice, splice, and reverse
- Other Array Manipulation Techniques
- Looping through an Array
- Merging Arrays
- Introduction to Hashes
- Preallocating Memory to Optimize Hash Performance

Packages and Modules

- The Power of Packages and Modules
- Introduction to Standard Modules
- Where to Find Modules on the Internet
- Installing a Module on UNIX or Windows
- Creating Packages for Portability
- Using Packages to Create Isolated Namespaces and to Separate Code

File Manipulation

- Using open and close
- Difference Between print and write
- Reading and Writing Arrays
- Directory Manipulation Using opendir, closedir, readdir, chdir, mkdir and rmdir

Input/Output Processing

- Parsing Input
- Using Standard Input, Standard Output, and Standard Error
- String and Field Processing
- Using Streams and Pipes
- Using die to Quit with an Error
- Redirecting Standard Output and Standard Error to a File
- Getting Standard Input from a File

Implementing Command Line Arguments

- Reading Command Line Arguments from @ARGV
- Manipulating Positional Parameters with push, pop, shift
- Processing Command Line Options with getopt or getopts
- Analyzing Command Line Argument Values with the Getopt::Std and Getopt::Long Modules
- Reserved Variables
- Manipulating Identifiable Options Using GetOptions

Perl Report Formatting

- Defining Report Formats
- Justifying Text (Left, Right, Center)
- Using write to Generate Reports
- Defining here Documents for Report Customization
- Creating Report Headers
- Using Built-in Variables to Control Report Appearance
- Printing Line Numbers on a Report
- Formatting Multi-Line Output
- Writing Formatted Text to a File

Debugging In Perl

- Using the Built-in Perl Debugger
- Starting the Debugger
- Debugger Command Syntax
- Checking for Script Syntax Errors
- Solving Compile-Time Errors
- Single-Stepping through a Script
- Executing to Breakpoints
- Setting Global Watches
- Printing Values of Variables
- Listing All Variables Used in the Script
- Using strict Error Checking
- Quitting the Debugger

Equipment Requirements

(This apply's to our hands-on courses only)

BTS always provides equipment to have a very successful Hands-On course. BTS also encourages all attendees to bring their own equipment to the course. This will provide attendees the opportunity to incorporate their own gear into the labs and gain valuable training using their specific equipment.

Course Length

3 Days