

Exploring Expect

2 days
70% Lecture/30% Lab
Intermediate Level

Overview

Expect's unparalleled support for interacting with command-line and network applications have made it the industry standard for automated test applications. But its features also make it an excellent tool for managing interprocess communication and extending legacy applications. As one developer said, "Expect can make easy all sorts of tasks that are prohibitively difficult with anything else."

Benefits

In this course you will learn

- Core Expect commands
- How Expect interacts with the operating system
- Automated applications to avoid common pitfalls
- Logging and debugging utilities for solving hard-to-find interaction problems
- Power of regular expressions for pattern matching and data extraction
- Advanced features for automating multiple applications simultaneously as well as handling user interaction

Intended Audience

This course is recommended for people with a basic Tcl programming background who need to create Expect programs, particularly automated test, command-line automation, or network automation applications.

Prerequisites

Students should have taken the *Introduction to Tcl/Tk* course, or have equivalent Tcl knowledge. This course is normally taught in a UNIX/Linux environment, so students should also be comfortable with basic UNIX/Linux operation, including file editing, basic shell operations (cd, ls, etc.), and basic file operations (moving, deleting, file permissions, etc.).

Training Approach

This is an intensive, interactive course, which is approximately 70% lecture and 30% lab. Questions are highly encouraged. On the final day, students are given access to a zipped file containing all of the solutions to the labs and the examples used throughout the notebook.

Course Outline

Day 1

Module 1 — Overview of Expect

- Introduction to Expect
- Why Do We Need Expect?
- Who Uses Expect?
- Expect Architecture
- Invoking and Using Expect

Module 2 — Core Expect Commands

- Core Expect Commands
- Sending Strings (send)
- Waiting for Input (expect)
- Timeouts and the expect Command
- Expect Commands and Variable Scoping
- Starting a Process (spawn)
- Sending to a Process
- Different End-of-Line Characters in Different Situations
- How Expect Interacts with a Process and a User
- Reading from a Process
- Partial Automation with interact
Lab 1

Module 3 — Pattern Matching

- Expecting Multiple Patterns
- Pattern Order is Important!
- Detecting if expect Timeouts
- Using timeout to Handle Error Cases
- Detecting Process Termination
- Glob Patterns in Expect
- What Happens When expect Matches?
- Character Buffering in Expect
Lab 2

Module 4 — Logging and Debugging

- Debugging Patterns

- Enabling Diagnostic Output with exp_internal
Exercise 1
- Strategies for Using exp_internal
- Logging Process Output to a File
- (Intentionally) Slowing Down Interaction

Module 5 — Regular Expressions in Tcl

- Basic vs. Advanced Regular Expressions
- Basic Regular Expression Syntax Summary
- The Tcl regexp Command
- Using Anchors to Match Patterns at Beginning or End of Strings
- Character Ranges
- Exclusive Character Ranges
- Escaping Special Characters
- Subpatterns
- Building Complex Regular Expressions
- Capturing Subpatterns
- The Tcl regsub Command
Exercise 2
- Advanced Regular Expressions in Tcl 8.1 and Later
- Potential Areas for Incompatibility
- Backslash Escapes in Tcl
- Noncapturing Subpatterns
- Character Classes in Regular Expressions
- Character Class Shorthand Escapes
- Greedy vs. Non-Greedy Quantifiers
- Using Bounds to Match a Specific Quantity
- Useful Patterns

Day 2

Module 6 — Regular Expressions in Expect

- Which Regular Expressions Can You Use?
- Using Regular Expressions with expect
- Prompt Matching
- Regular Expression Subpatterns and expect_out
- The Most Useful Regular Expression for Expect!
Lab 3

Module 7 — Additional Expect Techniques

- Case-Insensitive Matching
- Creating Persistent Expect Patterns
- Closing Connections (close)
- Waiting for a Process to Finish (wait)
- Looping Using exp_continue

Module 8 — Handling a Process and a User

- Managing a User and a Program
- Sending Strings to the User
- Filtering Process Output
- Writing Error Messages

- Reading from a User
- Accessing Terminal After Redirection
- Changing Terminal Modes
- Cooked vs. Raw Mode
- Prompting for Passwords
- Patterns in interact
- Patterns as “Macros”
- Returning Control to the Script
Lab 4

Module 9 — Handling Multiple Processes

- Spawning Multiple Processes
- Interaction through Implicit Spawn IDs
- “Multicasting” to Several Processes
- Interaction through Explicit Spawn IDs
- “Listening” to Multiple Processes Simultaneously
- Expecting the Same Pattern from Multiple Processes
- Using Spawn Lists
- Dynamically Maintaining a Spawn List
- Using Indirect Spawn IDs
- Treating the User as Another Process
Lab 5

For more information, contact

Tom Wille
tw@tm-associates.com
503-656-4457