

Course Description:

This intensive course rapidly trains programmers to develop applications and programs on Red Hat Enterprise Linux. Over the span of five days, you'll get hands-on training, concepts, and demonstrations with emphasis on realistic labs and programming exercises. Learn concepts and skills essential to programming and software development for Linux-based applications and products.

Who Should Attend:

The course is for experienced C programmers who want to learn key skills for creating applications and programs on Red Hat Enterprise Linux. This course is also useful for Windows and UNIX programmers migrating their programs to Linux.

Prerequisites:

Students should have experience in C programming, RH133 or equivalent UNIX or Linux workstation user skills for developers, shell scripting in a UNIX or Linux environment, and experience with editors such as vi, emacs.

Benefits of Attendance:

Upon completion of this course, students will be able to:

- Develop applications and programs on Red Hat Enterprise Linux.

Course Outline:**GCC - GNU Compiler Collection**

GNU Compiler Collection
History of GCC
Four Stages of GCC
Interrupting the Compiler
Compiling a C Program
Preprocessor Features
Predefined Preprocessor Symbols
Warnings and Extensions
Optimization
Linking

Building Software with Make

Introducing make(1)
How make Works
Makefile Rule Syntax
Example: Makefile First Steps
Makefile Improved
Implicit Rules
Example: Simpler Is Better Makefile Variables
Defining Variables
Example: Makefile with Variables
Automatic Variables
Special Targets
Defining Useful Phony Targets

The GNU C Library and System Calls

Library Goals
Library Standards
GNU C Library - glibc
Library Functions vs. System Calls
Using System Calls
Handling Errors with errno
Making Sense of errno
Using strace

Program Arguments and Environment

Program Startup
Using argc/argv
Handling Options with getopt()
Handling Options with getopt_long()
Environment
Manipulating the Environment
Program Exit
Registering Exit Handlers

Building Libraries

Why Use Libraries?
Static Versus Shared
Static Library Benefits
Shared Library Benefits

Creating a Static Library
Using Static Libraries
Creating a Shared Library
Using Shared Libraries
Shared Library Management
Library Locations
ldconfig

Time Functions

When Does Time Begin?
Time Data Types
Determining Real Time
Converting time_t
Converting tm Structure
Process Time
Time arithmetic
Second Resolution Timers
Fine-Grained Timers
< Real Time Clock (RTC)

Process Management

What a Process Is
Process Relationships
Create a Child Process
Doing Something Else
Related execve() Functions
Wait For a Child
More Precise Waiting
Changing Priority/Nice
Real Time Priority

Memory Operations

Allocating/Freeing Memory
Memory Alignment
Locked Memory
Memory Copy/Initialization
Memory Comparison/Search

Debugging

What Is My Program Doing?
Source Level Debugging
Invoking gdb
Getting Started with gdb
Examining and Changing Memory
Debuginfo Libraries
Using gdb with a Running Process
Using gdb to Autopsy a Crash
Debugging Libraries - ElectricFence
Debugging with valgrind
Profiling for Performance

Basic File Operations

Stream vs. System Calls
Opening/Closing Streams
Stream Input/Output Functions

Stream Status/Errors
Stream File Positioning
Stream Buffering
Temporary/Scratch Files
Opening/Closing File Descriptors
File Descriptor I/O
Repositioning File Descriptors
Stream/File Descriptor Conversions
cat using ANSI I/O
cat using POSIX I/O

Communicating with Pipes

Introduction to Pipes
Standard I/O: popen()/pclose()
Using popen()/pclose()
System Call: pipe()
Using pipe()
Named Pipes
Using Named Pipes
For Further Reading

Managing Signals

What Signals Are
Blocking/Checking Signals
Working with Signal Sets
Example of Blocking Signals
Handling Signals with sigaction()
sigaction() Example
Handling Signals with signal()
Sending Signals
Real-Time Signals

Programming with Threads

Introducing Threaded Programming
Applications Suited to Threads
Building Threaded Programs
Creating Threads
Thread Identity
Synchronizing by Joining
Detaching Threads
Stopping Threads
Synchronizing with Mutexes
Using Mutexes
Read/Write Locks
Conditional Variables
Using Conditional Variables
A Conditional Variable Gotcha
For Further Reading

Advanced File Operations

Directory Operations
File System Operations
Multiplexed I/O with select()
Miscellaneous I/O Functions
Memory Mapped I/O

Using Memory Mapped I/O
File Locking

Interprocess Communication (IPC)

Interprocess Communication (IPC)
POSIX IPC Overview
POSIX Shared Memory
POSIX Semaphores
POSIX Message Queues
System V IPC Overview
System V IPC Shared Memory
System V IPC Semaphore Arrays
System V IPC Message Queues

Basic Network Programming

Linux Networking Overview
Getting Started with socket()
Client Functions
Specifying IPv4 Addresses
Host Versus Network Byte Order
Example TCP/IP Client
Address Conversion Functions
Using getaddrinfo()
Server Functions
Example TCP/IP Server
Datagram Communication with UDP

Working with the Linux Community

Getting in Touch with the Community
General Considerations
Building a Community
Licenses
GPL
LGPL
BSD
Creative Commons