Training: The Linux Foundation
LFD401 Developing Applications For Linux

TRAINING GOALS:

This four days course is designed to help experienced developers get up to speed quickly on how to develop applications for a Linux environment. In this course, you'll get hands-on experience with the necessary tools and methods for Linux application development and learn about the features and techniques that are unique to Linux.

During this course you will learn:

- The tools and methods for developing C programs and doing systems programming under Linux.
- Debugging techniques and process management.
- Linux specific paid and system calls.
- And more.

The information in this course will work with any major Linux distribution.

CONSPECT:

- Introduction
  - Objectives
  - Who You Are
  - The Linux Foundation
  - Linux Foundation Training
  - Linux Distributions
  - Platforms
  - Preparing Your System
  - Using and Downloading a Virtual Machine
  - Things change in Linux
  - Course Registration
- Preliminaries
  - Procedures
  - Standards and the LSB
- How to Work in OSS Projects **

Compendium Education Center Ltd.
ul. Tatarska 5, 30-103 Kraków, tel.: (12) 298 47 77
ul. Postępu 18B, 02-676 Warszawa, tel.: (22) 417 41 70

BNP Paribas Bank Polska S.A.
EUR: PL 79 1600 1462 1853 6451 3000 0006
USD: PL 52 1600 1462 1853 6451 3000 0007

    - Overview on How to Contribute Properly
    - Stay Close to Mainline for Security and Quality
    - Study and Understand the Project DNA
    - Figure Out What Itch You Want to Scratch
    - Identify Maintainers and Their Work Flows and Methods
    - Get Early Input and Work in the Open
    - Contribute Incremental Bits, Not Large Code Dumps
    - Leave Your Ego at the Door: Don’t Be Thin-Skinned
    - Be Patient, Develop Long Term Relationships, Be Helpful
- Compilers
    - GCC
    - Other Compilers
    - Major gcc Options
    - Preprocessor
    - Integrated Development Environments (IDE)
    - Labs
- Libraries
    - Static Libraries
    - Shared Libraries
    - Linking To Libraries
    - Dynamic Linking Loader
    - Labs
- Make
    - Using make and Makefiles
    - Building large projects
    - More complicated rules
    - Built-in rules
    - Labs
- Source Control
    - Source Control
    - RCS and CVS
    - Subversion
    - git
    - Labs
- Debugging and Core Dumps
    - gdb

Compendium Education Center Ltd.
ul. Tatarska 5, 30-103 Kraków, tel.: (12) 298 47 77
ul. Postępu 18B, 02-676 Warszawa, tel.: (22) 417 41 70

    - What are Core Dump Files?
    - Producing Core Dumps
    - Examining Core Dumps
    - Labs
- Debugging Tools
    - Electric Fence
    - Getting the Time
    - Profiling and Performance
    - valgrind
    - Labs
- System Calls
    - System Calls vs. Library Functions
    - How System Calls are Made
    - Return Values and Error Numbers
    - Labs
- Memory Management and Allocation
    - Memory Management
    - Dynamical Allocation
    - Tuning malloc()
    - Locking Pages
    - Labs
- Files and Filesystems in Linux **
    - Files, Directories and Devices
    - The Virtual File System
    - The ext2/ext3 Filesystem
    - Journaling Filesystems
    - The ext4/ Filesystem
    - Labs
- File I/O
    - UNIX File I/O
    - Opening and Closing
    - Reading, Writing and Seeking
    - Positional and Vector I/O
    - Standard I/O Library
    - Large File Support (LFS)
    - Labs

Compendium Education Center Ltd.
ul. Tatarska 5, 30-103 Kraków, tel.: (12) 298 47 77
ul. Postępu 18B, 02-676 Warszawa, tel.: (22) 417 41 70

- Advanced File Operations
  - Stat Functions
  - Directory Functions
  - inotify
  - Memory Mapping
  - flock() and fcntl()
  - Making Temporary Files
  - Other System Calls
  - Labs
- Processes – I
  - What is a Process?
  - Process Limits
  - Process Groups
  - The proc Filesystem
  - Inter-Process Communication Methods
  - Labs
- Processes – II
  - Using system() to Create a Process
  - Using fork() to Create a Process
  - Using exec() to Create a Process
  - Using clone()
  - Exiting
  - Constructors and Destructors
  - Waiting
  - Daemon Processes
  - Labs
- Pipes and Fifos
  - Pipes and Inter-Process Communication
  - popen() and pclose()
  - pipe()
  - Named Pipes (FIFOs)
  - splice(), vmsplice() and tee()
  - Labs
- Asynchronous I/O**
  - What is Asynchronous I/O?
  - The POSIX Asynchronous I/O API

Compendium Education Center Ltd.
ul. Tatarska 5, 30-103 Kraków, tel.: (12) 298 47 77
ul. Postępu 18B, 02-676 Warszawa, tel.: (22) 417 41 70

- Linux Implementation
- Labs

- Signals – I
  - What are Signals?
  - Signals Available
  - Dispatching Signals
  - Alarms, Pausing and Sleeping
  - Setting up a Signal Handler
  - Signal Sets
  - sigaction()
  - Labs
- Signals – II
  - Reentrancy and Signal Handlers
  - Jumping and Non-Local Returns
  - siginfo and sigqueue()
  - Real Time Signals
  - Labs
- POSIX Threads – I
  - Multi-threading under Linux
  - Basic Program Structure
  - Creating and Destroying Threads
  - Signals and Threads
  - Forking vs. Threading
  - Labs
- POSIX Threads – II
  - Deadlocks and Race Conditions
  - Mutex Operations
  - Semaphores
  - Futexes
  - Conditional Operations
  - Labs
- Networking and Sockets
  - Networking Layers
  - What are Sockets?
  - Stream Sockets
  - Datagram Sockets

Compendium Education Center Ltd.
ul. Tatarska 5, 30-103 Kraków, tel.: (12) 298 47 77
ul. Postępu 18B, 02-676 Warszawa, tel.: (22) 417 41 70

BNP Paribas Bank Polska S.A.
EUR: PL 79 1600 1462 1853 6451 3000 0006
USD: PL 52 1600 1462 1853 6451 3000 0007

    - Raw Sockets
    - Byte Ordering
    - Labs
- Sockets – Addresses and Hosts
    - Socket Address Structures
    - Converting IP Addresses
    - Host Information
    - Labs
- Sockets – Ports and Protocols
    - Service Port Information
    - Protocol Information
    - Labs
- Sockets – Clients
    - Basic Client Sequence
    - socket()
    - connect()
    - close() and shutdown()
    - UNIX Client
    - Internet Client
    - Labs
- Sockets – Servers
    - Basic Server Sequence
    - bind()
    - listen()
    - accept()
    - UNIX Server
    - Internet Server
    - Labs
- Sockets – Input/Output Operations
    - write(), read()
    - send(), recv()
    - sendto(), recvfrom()
    - sendmsg(), recvmsg()
    - sendfile()
    - socketpair()
    - Labs

Compendium Education Center Ltd.
ul. Tatarska 5, 30-103 Kraków, tel.: (12) 298 47 77
ul. Postępu 18B, 02-676 Warszawa, tel.: (22) 417 41 70

- Sockets – Options
  - Getting and Setting Socket Options
  - fcntl()
  - ioctl()
  - getsockopt() and setsockopt()
  - Labs
- Netlink Sockets**
  - What are netlink Sockets?
  - Opening a netlink Socket
  - netlink Messages
  - Labs
- Sockets – Multiplexing and Concurrent Servers
  - Multiplexed and Asynchronous Socket I/O
  - select()
  - poll()
  - pselect() and ppoll()
  - epoll
  - Signal Driven and Asynchronous I/O
  - Concurrent Servers
  - Labs
- Inter Process Communication
  - Methods of IPC
  - POSIX IPC
  - System V IPC**
  - Labs
- Shared Memory
  - What is Shared Memory?
  - POSIX Shared Memory
  - System V Shared Memory**
  - Labs
- Semaphores
  - What is a Semaphore?
  - POSIX Semaphores
  - System V Semaphores**
  - Labs
- Message Queues

Compendium Education Center Ltd.
ul. Tatarska 5, 30-103 Kraków, tel.: (12) 298 47 77
ul. Postępu 18B, 02-676 Warszawa, tel.: (22) 417 41 70

- What are Message Queues?
- POSIX Message Queues
- System V Message Queues**
- Labs

- Closing and Evaluation Survey

** These sections may be considered in part or in whole as optional. They contain either background reference material, specialized topics, or advanced subjects. The instructor may choose to cover or not cover them depending on classroom experience and time constraints.

REQUIREMENTS:

This course is for experienced developers. Students should be proficient in C programming, and be familiar with basic Linux utilities and text editors.

Difficulty level

CERTIFICATE:

The participants will obtain certificates signed by The Linux Foundation.

TRAINER:

Certified The Linux Foundation Trainer.

Compendium Education Center Ltd.
ul. Tatarska 5, 30-103 Kraków, tel.: (12) 298 47 77
ul. Postępu 18B, 02-676 Warszawa, tel.: (22) 417 41 70