Training Catalogue
29/05/2020

KAÏNA-COM
TRAINING CATALOGUE

Linux Kernel and Device Drivers

This hands-on course focuses on Linux kernel internals programming including device drivers

Nos locaux
KAÏNA-COM France
LE CARRÉ HAUSSMANN II
6 Allée de la Connaissance
77 127 Lieusaint

Contact
+33(0)9 50 20 91 64

E-mail
info@kaina-com.fr

Site Internet
www.kaina-com.fr

KLI002 – Linux Kernel and Device Drivers

Reference

KLI002

Experience

- ☐ Beginner
- ☒ Intermediate
- ☐ Advanced

Duration

Training Program:

- 5 days

Training Method

- ☐ I: i-learning, individual training (web-based training)
- ☒ V: v-learning, virtual class
- ☐ C: c-learning, classroom training

KAÏNA-COM
LE CARRÉ HAUSSMANN II,
6 Allée de la Connaissance
77127 Lieusaint - France

Price

2.526,00 € HT

Prerequisite

Linux experience is a must, including user mode programming (using gcc, Make) and editing.
Programming experience in ANSI C, with the standard library, including sockets programming is also a requirement.

Audience

Software architects, designers, developers and analysts with Linux experience who need to learn and program in the kernel environment, including device drivers.

Continued on next page

KLI002 – Linux Kernel and Device Drivers, Continued

Objective

This hands-on course focuses on Linux kernel internals programming including device drivers. Participants will learn about the Linux kernel architecture, programming in the kernel environment, space considerations, network device drivers and debugging mechanisms. Following this course, participants will be able to develop Linux kernel modules and device drivers.
Examples are in C.
Course exercises include the implementation of a functional character device driver, and a skeletal network device driver, using kernel 3.10 (RHEL 7.X).

Continued on next page

KLI002 – Linux Kernel and Device Drivers, Continued

Course Contents

Course Contents :

Table 1: KLI002 - Course Contents

Chapter	Description
Introduction	• The Linux Kernel
Kernel Architecture	• Linux kernel general properties • System calls • Task Scheduler – Details and evolution • I/O Schedulers – Elevators – CFQ – No op • Kernel Preemption • Threads NPTL
The Kernel Perspective	• Files and FileSystems • Devices – SysFS • Processes • Floating Point
Module Programming (+Exercises)	• Implementing Kernel modules • Module writing guidelines • Kernel structures • Printk

Continued on next page

KLI002 – Linux Kernel and Device Drivers, Continued

Course Contents, continued

Chapter	Description
Character Device Drivers (+Exercises)	• Device numbers • Essential kernel structures – inode – file – file_operations – cdev • Registering a character device
Character Device Drivers (Continued)	• Device System Calls • open, close Working with User Space memory • Implementing read, write and ioctl • Virtual Memory Management – overview • mmap • devtmpfs • udev
Kernel Space Considerations (+Exercises)	• Timing issues and kernel timers • Synchronicity – semaphores – spinlocks – wait queues • read and write with support of both blocking and non blocking i/o • poll • Handling Interrupts • Bottom Halves • SoftIRQs, Work Queues, TaskLets and threaded irq's

Continued on next page

KLI002 – Linux Kernel and Device Drivers, Continued

Course Contents, continued

Chapter	Description
Network Device Drivers (+ Exercises)	• The Linux Protocol Stack • Packet flow – from the interface to the application and back • Socket buffer operations • PF_PACKET • Hooking with NetFilter • Overriding network system calls
Debugging mechanisms	• Kernel debugging techniques in Linux – strace – standard /proc and /sys entries • Implementing entries in /proc • Handling Oops and Panics • debugfs • KProbes • Magic SYSRQ • KDB
The End	• Summary • Q&A • Evaluation