Stand 14.01.2026

Embedded-Linux für Test und Support - Live-Online-Training

**Ziele -
Ihr Nutzen**

Das Training vermittelt Ihnen ein Verständnis der Abläufe im Linux-System, vom Booten bis hin zum laufenden System.

Es beleuchtet das Diagnostizieren von Problemen im Testfeld und im Support.

Sie lernen, wie Sie Probleme mit den Werkzeugen des Betriebssystems sowie gängigen Open-Source-Tools eingrenzen und lösen können.

Erforderliche Kenntnisse im Umgang mit der Linux-Shell werden besprochen.

Besonderer Wert wird darauf gelegt, dass das erlernte Know-How universell sowohl auf Embedded- als auch auf Standard-Linux-Systemen einsetzbar ist.

Teilnehmer

Das (Embedded-)Linux-Training richtet sich an Ingenieure in der Qualitätssicherung, an Test-Ingenieure und Test-Manager.

Voraussetzungen

Motivation, das Betriebssystem Linux kennenlernen zu wollen

Live Online Training

* Preis je Teilnehmer, in Euro zzgl. USt.

Anmeldecode: L-LIN-T

Präsenz-Training - Deutsch

Dauer
4 Tage

Live-Online - Englisch

Dauer
4 Tage

Präsenz-Training - Englisch

Dauer
4 Tage

Embedded-Linux für Test und Support - Live-Online-Training

Inhalt

Aufbau des Linux-Systems

- Umgang mit der Shell, wichtige Kommandos
- History, wichtige Tastenkürzel, man und info
- Consolenvideos aufzeichnen und widergeben
- Bootvorgang
- Bootloader grub, u-boot und barebox
- Linux-Kernel Bootprozess, Device-Tree
- Aufgaben des Init-Dämons (System-V- und busybox-Init, systemd)
- systemd: Units, Einbindung eigener Dämonen und Programme, Mounts, Netzwerk-Einstellungen
- systemd-Tools: systemctl, journalctl, timedatectl
- udev-Dämon, udev-Rules
- TCP-, UDP-, UNIX- und Netlink-Sockets
- Verwendung von Device-Nodes, Character- und Block-Devices
- Memory-Mapping, blockierende Operationen

Diagnose-Werkzeuge

- C-Libraries (glibc, uClibc und musl)
- netcat, netstat
- tcpdump, capture-File, wireshark, iftop
- BPF einsetzen
- Tracen mit strace und ltrace
- procfs, sysfs und debugfs als Diagnosedateisysteme
- GNU-Debugger gdb, gdbserver
- Core-Dumps generieren und auswerten
- Logging von unerwarteten Signalen mit backtrace
- dmesg, Kernel-Oops auswerten, Kernel-Crashes
- Mit addr2line und objdump vom Fehler zum Sourcecode
- ftrace-Framework
- trace-cmd, kernelshark und perf
- Tracing-Events verwenden und erstellen