Writing UNIX Device Drivers

This is the definitive reference written by UNIX System Laboratories. The book covers the file format for important system files, such as password, hosts, system initializations and special (device) files. Covers UNIX System V Release 4 and later, including BSD and ZENIX variants unified under this release.

Device Driver Interface/driver-kernel Interface Reference Manual

Pajari provides application programmers with definitive information on writing device drivers for the UNIX operating system. The comprehensive coverage includes the four major categories of UNIX device drivers: character, block, terminal, and stream drivers. (Operating Systems)

Advanced System Administration

Newly updated to include new calls and techniques introduced in Versions 2.2 and 2.4 of the Linux kernel, a definitive resource for those who want to support computer peripherals under the Linux operating system explains how to write a driver for a broad spectrum of devices, including character devices, network interfaces, and block devices. Original. (Intermediate)

UNIX System V/386, Release 4

Java Message Service

Offers practical, hands-on guidance in developing your own device drives. Clearly demonstrates how to write device drivers for adding disk drives, printers, magnetic tapes and other peripherals to your Unix system. Presents procedures for developing and testing new device drivers including how to select a convenient working directory; use make-files; preserve and boot alternative kernal versions; debug driver code and much more. Packed with examples which illustrate each operation in practice.

Exim

This is written for system administrators who may not have the time to learn about Slash by reading the source code. It collects all the current Slash knowledge from the code, Website and mailing lists and organizes it into a coherent package.

UNIX Internals

This book concentrates on how to use the system services provided by the UNIX operating system kernel. It is designed to provide information about application programming in a UNIX system environment and supplements texts on programming by concentrating on the other elements that are part of getting application programs into operation.

Writing a UNIX? Device Driver

Designed to help readers perform advanced UNIX system administration functions, this book offers procedures for installing and maintaining various services to serve the needs of users. Coverage includes: managing ports; managing storage devices; tunable parameters; process scheduling; customizing the sysadm interface; and more.
Running Weblogs with Slash

Describes the C compilation system for UNIX System V Release 4.0 or later. It also provides pointers on transitioning to ANSI/ISO C. The guide covers how to comply with standard C, debugging from the CLI, multiprocess debugging, using debug with GUIs, C profiling utilities and using the cscape Browser.

Programming with UNIX System Calls

Written for first-time system administrators and end-users, this practical guide goes step-by-step through the CUI menus for configuring, tailoring, and maintaining your Interactive UNIX Operating System V/386 Release 3.2, Version 3.0 through Version 4.1. It is also a great reference for any SVR 3.2 UNIX system. The Interactive UNIX Operating System: A Guide for System Administrators begins with a discussion of hardware configuration requirements. It provides detailed examples of the most common UNIX issues facing system administrators today and is based upon actual customer questions and issues handled by SunSoft's service support lines over the years. Loaded with instructions on adding new devices, explanations of UNIX commands, and troubleshooting examples, this book will facilitate day-to-day system administrative tasks. It is also a great book for DOS users who want to learn the UNIX system.

Java Studio by Example

Programming in Standard C

This book is a thorough introduction to Java Message Service (JMS), the standard Java application program interface (API) from Sun Microsystems that supports the formal communication known as "messaging" between computers in a network. JMS provides a common interface to standard messaging protocols and to special messaging services in support of Java programs. The messages exchange crucial data between computers, rather than between users-information such as event notification and service requests. Messaging is often used to coordinate programs in dissimilar systems or written in different programming languages. Using the JMS interface, a programmer can invoke the messaging services of IBM's MQSeries, Progress Software's SonicMQ, and other popular messaging product vendors. In addition, JMS supports messages that contain serialized Java objects and messages that contain Extensible Markup Language (XML) pages. Messaging is a powerful new paradigm that makes it easier to uncouple different parts of an enterprise application. Messaging clients work by sending messages to a message server, which is responsible for delivering the messages to their destination. Message delivery is asynchronous, meaning that the client can continue working without waiting for the message to be delivered. The contents of the message can be anything from a simple text string to a serialized Java object or an XML document. Java Message Service shows how to build applications using the point-to-point and publish-and-subscribe models; how to use features like transactions and durable subscriptions to make an application reliable; and how to use messaging within Enterprise JavaBeans. It also introduces a new EJB type, the MessageDrivenBean, that is part of EJB 2.0, and discusses integration of messaging into J2EE.

ADO ActiveX Data Objects

Operating System API Reference

The UNIX desktop is a user-friendly front end to the UNIX operating system and is available on many platforms. This book provides guidance for programmers who want to take advantage of the programming facilities available in UNIX desktop. The book covers the file class database and the Windowing Korn SHEll.

PC-interface Administration

New for UNIX System V Release 4.2, this guide contains the latest information for writing, installing and testing UNIX System V device drivers. It provides an in-depth explanation of new SVR4.2 features such as dynamically loadable kernel modules, the new device driver installation tools and the new system configuration file formats.

Portable Device Interface (PDI)

The definitive source of information for kernel-level STREAMS programming--in both uniprocessor and multiprocessor UNIX System V Release 4 environments. This guide is an indispensable resource for network and systems programmers responsible for designing and writing STREAMS-based modules and device drivers.

Biblioteka

Streams Modules and Drivers
Over 30 recipes to develop custom drivers for your embedded Linux applications. Key Features

- Use Kernel facilities to develop powerful drivers
- Via a practical approach, learn core concepts of developing device drivers
- Program a custom character device to get access to kernel internals

Book Description

Linux is a unified kernel that is widely used to develop embedded systems. As Linux has turned out to be one of the most popular operating systems used, the interest in developing proprietary device drivers has also increased. Device drivers play a critical role in how the system performs and ensures that the device works in the manner intended. By offering several examples on the development of character devices and how to use other kernel internals, such as interrupts, kernel timers, and wait queue, as well as how to manage a device tree, you will be able to add proper management for custom peripherals to your embedded system. You will begin by installing the Linux kernel and then configuring it. Once you have installed the system, you will learn to use the different kernel features and the character drivers. You will also cover interrupts in-depth and how you can manage them. Later, you will get into the kernel internals required for developing applications. Next, you will implement advanced character drivers and also become an expert in writing important Linux device drivers. By the end of the book, you will be able to easily write a custom character driver and kernel code as per your requirements. What you will learn

- Become familiar with the latest kernel releases (4.19+/5.x) running on the ESPRESSObin devkit, an ARM 64-bit machine
- Download, configure, modify, and build kernel sources
- Add and remove a device driver or a module from the kernel
- Master kernel programming
- Understand how to implement character drivers to manage different kinds of computer peripherals
- Become well versed with kernel helper functions and objects that can be used to build kernel applications
- Acquire a knowledge of in-depth concepts to manage custom hardware with Linux from both the kernel and user space

Who this book is for

This book will help anyone who wants to develop their own Linux device drivers for embedded systems. Having basic hand-on with Linux operating system and embedded concepts is necessary.

Proceedings of the Winter 1993 USENIX Conference

Don't miss this guide to building networked and distributed applications for UNIX® System V. Using many helpful examples, the author provides a solid introduction to networking and UNIX programming, plus information on the programming interfaces most important to building communication software in System V, such as STREAMS, the Transport Layer Interface library, Sockets, and Remote Procedure Calls. The book also explains how to write kernel-level communication software, including STREAMS drivers, modules, and multiplexors. A final chapter on SLIP is essential reading.

Linux Device Drivers

New for UNIX System V Rel 4.2, this book explains a new programming interface for developing easily portable block-oriented device drivers. The guide presents in-depth information to organize, simplify, and standardize the way most BUS adapted, SCSI, and non-SCSI target drivers operate in UNIX System V.

UNIX System V Network Programming

This manual explains how to install and administer the LAN and RS-232 versions of the PC-Interface for DOS software on a host computer running a UNIX System V operation system with TC/IPC networking services. It covers day-to-day operation and maintenance of the PC-Interface software.

Computer Security Policies and SunScreen Firewalls

Unix System V Release 4

This book offers an up-to-date, in-depth, and broad-based exploration of the latest advances in UNIX-based operating systems. Focusing on the design and implementation of the operating system itself, this text compares and analyzes the alternatives offered by several important UNIX variants, and covers several advanced subjects, such as multi-processors and threads.

Interactive UNIX Operating System

This comprehensive reference consists of two parts. The first part describes the Device Driver Interface/Driver-Kernel Interface (DDI/DKI). The second part describes routines of the Portable Device Interface (PDI). Intended for programmers, software developers and administrators working with UNIX System V Release 4 or later.

UNIX System V Release 4

System Files and Devices Reference
UNIX Review

Device Driver Interface/driver-kernel Interface Reference Manual

Exim delivers electronic mail, both local and remote. It has all the virtues of a good postman: it's easy to talk to, reliable, efficient, and eager to accommodate even the most complex special requests. It's the default mail transport agent installed on some Linux systems, runs on many versions of Unix, and is suitable for any TCP/IP network with any combination of hosts and end-user mail software. Exim is growing in popularity because it is open source, scalable, and rich in features such as the following: Compatibility with the calling interfaces and options of Sendmail (for which Exim is usually a drop-in replacement) Lookups in LDAP servers, MySQL and PostgreSQL databases, and NIS or NIS+ services Support for many kinds of address parsing, including regular expressions that are compatible with Perl 5 Sophisticated error handling Innumerable tuning parameters for improving performance and handling enormous volumes of mail Best of all, Exim is easy to configure. You never have to deal with ruleset 3 or worry that a misplaced asterisk will cause an inadvertent mail bomb. While a basic configuration is easy to read and can be created quickly, Exim's syntax and behavior do get more subtle as you enter complicated areas like virtual hosting, filtering, and automatic replies. This book is a comprehensive survey that provides quick information for people in a hurry as well as thorough coverage of more advanced material.

UNIX System V Release 4

This helpful reference provides a comprehensive description of UNIX system calls and library functions found in UNIX System V Release 4.2. Covers C language, math, networking, and specialized libraries.

UniForum Monthly

PLEASE PROVIDE?

Real-time Systems Education III

Covers the development tools needed to create applications based on a client/server model of computing. The book describes the programming interfaces to SVRR4.2 networking facilities such as Transport Library Interface (TLI), Sockets, Remote Procedure Call (RPC), Connection Server and REXEC.

Understanding the Linux Kernel

Building the New Enterprise

For beginning, intermediate, and advanced users, this book offers complete coverage of UNIX. Offering information on basic UNIX, programming UNIX, communications and networking, the book also discusses new, more advanced tools such as Perl and presents in-depth discussions of the Internet, Windows, Linux, the bestselling UNIX systems, and more.

The British National Bibliography

Revised to match the official updated Motorola documentation (Version 3), this manual shows how to modify and maintain drivers that run under UNIX System V Release 4 for Motorola Processors. It covers driver data definitions, driver entry point routines, kernel utility routines, kernel data structures, kernel defines.

Graphical User Interface Programming

To thoroughly understand what makes Linux tick and why it's so efficient, you need to delve deep into the heart of the operating system--into the Linux kernel itself. The kernel is Linux—in the case of the Linux operating system, it's the only bit of software to which the term "Linux" applies. The kernel handles all the requests or completed I/O operations and determines which programs will share its processing time, and in what order. Responsible for the sophisticated memory management of the whole system, the Linux kernel is the force behind the legendary Linux efficiency. The new edition of Understanding the Linux Kernel takes you on a guided tour through the most significant data structures, many algorithms, and programming tricks used in the kernel. Probing beyond the superficial features, the authors offer valuable insights to people who want to know how things really work inside their machine. Relevant segments of code are dissected and discussed line by line. The book covers more than just the functioning of the code, it explains the theoretical underpinnings for why Linux does things the way it does. The new edition of the book has been updated to cover version 2.4 of the kernel, which is quite different from version 2.2: the virtual memory system is entirely new, support for multiprocessor systems is improved, and whole new classes of hardware devices have been added. The authors explore each new feature in detail. Other topics in the book include: Memory management including file buffering, process swapping, and Direct memory Access (DMA) The Virtual Filesystem and the Second Extended
Filesystem Process creation and scheduling Signals, interrupts, and the essential interfaces to device drivers Timing Synchronization in the kernel Interprocess Communication (IPC) Program execution Understanding the Linux Kernel, Second Edition will acquaint you with all the inner workings of Linux, but is more than just an academic exercise. You’ll learn what conditions bring out Linux's best performance, and you’ll see how it meets the challenge of providing good system response during process scheduling, file access, and memory management in a wide variety of environments. If knowledge is power, then this book will help you make the most of your Linux system.

**Device Driver Reference**

**Network Programming Interface**

A reference and instructional guide to Microsoft's ActiveX Data Objects introduces the updated form of database communication to developers and Web programmers.

**Linux Device Driver Development Cookbook**

Poznan, Poland was selected as the site of the third annual workshop on this topic to coincide with the Software Engineering Education Symposium (SEES’98), and also to establish an international presence in Central Europe “although we are not an event on the scale of olympic games.” Twenty-six participants (pictured) present papers related to: real-time systems programs and curricula, teaching RT systems design and verification, components of RT labs, RT systems in control engineering education and in other disciplines. Also includes invited talks on RT devices at practical prices and the essentials of RT education. For the uninitiated, IEEE stands for the Institute of Electrical and Electronic Engineers. Author index only. Annotation copyrighted by Book News, Inc., Portland, OR.

**Device Driver Programming**

An insider’s guide to writing Java-powered Web pages with JavaStudio, this book shows how without writing a single line of code. The hands-on format can be used as both a tutorial and reference, depending on the experience level. The CD-ROM contains a full, working 30-day “try and buy” version of JavaStudio.

**Device Driver Interface/driver-kernel Interface Reference Manual**

A guide to implementing Client/Server technologies that covers the people, the processes and the technologies that are critical to making the transition.