

Systemnahe Programmierung mit C und Linux

Das umfassende Handbuch

DAS INHALTS- VERZEICHNIS

» Hier geht's
direkt
zum Buch

Auf einen Blick

1	Einführung	25
2	E/A-Funktionen	45
3	Attribute von Dateien und Verzeichnissen abfragen und ändern	139
4	Zugriff auf Systeminformationen	157
5	Devices – eine einfache Verbindung zur Hardware	183
6	System- und Benutzerdateien	227
7	Prozesse, Dämonen und Zombies	247
8	Signale	331
9	IPC – Interprozesskommunikation	355
10	Threads	443
11	Netzwerkprogrammierung	513
12	MySQL und PostgreSQL	649
13	Terminal-E/A und Benutzerschnittstellen für die Konsole	813
14	GTK+	891
15	Übersicht über weitere beliebte GUI-Bibliotheken	1043
16	Werkzeuge für Programmierer	1065

Inhalt

Materialien zum Buch	23
1 Einführung	25
1.1 Anforderungen an den Leser	25
1.2 Anforderungen an das Betriebssystem	26
1.3 UNIX, der Vorgänger von Linux	27
1.4 Die Entwicklung von Linux	29
1.5 Der Compiler GCC – eine kurze Einführung	31
1.5.1 Den GCC von der Kommandozeile aus aufrufen	31
1.5.2 Wichtige Kommandozeilen-Parameter für den GCC	32
1.5.3 Unterstützte Dateitypen beim GCC	37
1.6 POSIX, X/OPEN und ANSI C	37
1.7 Übersicht zu diesem Buch	40
1.8 Schreibkonventionen	42
1.9 Weitere Hilfen	43
2 E/A-Funktionen	45
2.1 Elementare E/A-Funktionen	45
2.2 Filedeskriptoren	47
2.2.1 Verwaltung von Filedeskriptoren	49
2.3 Funktionen, die einen Filedeskriptor verwenden	50
2.3.1 Datei öffnen – open()	50
2.3.2 Anlegen einer neuen Datei mit creat()	57
2.3.3 Eine Datei wieder schließen mit close()	57
2.3.4 Schreiben von Dateien mit write()	58
2.3.5 Lesen von Dateien mit read()	62
2.3.6 Schreib-/Lesezeiger positionieren mit lseek()	66
2.3.7 Duplizieren von Filedeskriptoren mit dup() und dup2()	69
2.3.8 Ändern oder Abfragen der Eigenschaften eines Filedeskriptors mit fcntl()	71

2.3.9	Record Locking – Sperren von Dateien einrichten	77
2.3.10	Multiplexing E/A mit select()	88
2.3.11	Unterschiedliche Operationen ausführen mit ioctl()	91
2.3.12	Lesen und Schreiben mehrerer Puffer mit writev() und readv()	93
2.3.13	Weitere Funktionen, die den Filedeskriptor verwenden	95
2.4	Standard-E/A-Funktionen	99
2.4.1	Der FILE-Zeiger	99
2.4.2	Öffnen und Schließen von Dateien	101
2.4.3	Formatierte Ausgaben erzeugen mit printf() und sprintf()	103
2.4.4	Formatierte Eingaben mit scanf()	104
2.4.5	Binäres Lesen und Schreiben	105
2.4.6	Zeichen- und zeilenweise Ein-/Ausgabe	106
2.4.7	Den Status der Ein-/Ausgabe mit feof() und ferror() überprüfen	107
2.4.8	Den Schreib-/Lesezeiger im Stream positionieren	107
2.4.9	Steuerung des Verhaltens des Schreib-/Lesepuffers	110
2.4.10	Dateien löschen und umbenennen	111
2.4.11	Temporäre Dateien erstellen	112
2.5	Die Arbeit mit Verzeichnissen	113
2.5.1	Ein neues Verzeichnis anlegen mit mkdir()	113
2.5.2	In ein anderes Verzeichnis wechseln mit chdir() und fchdir()	115
2.5.3	Ein leeres Verzeichnis löschen mit rmdir()	117
2.5.4	Format eines Dateieintrags in dem struct dirent	118
2.5.5	Einen Verzeichnisstream öffnen mit opendir()	120
2.5.6	Lesen aus dem DIR-Stream mit opendir() und Schließen des DIR-Streams mit closedir()	122
2.5.7	Positionieren des DIR-Streams	125
2.5.8	Ein komplettes Verzeichnis einlesen mit scandir()	125
2.5.9	Ganze Verzeichnisbäume durchlaufen mit ftw()	131
2.6	Fehlerbehandlung	134
2.7	Ausblick	137
3	Attribute von Dateien und Verzeichnissen abfragen und ändern	139
<hr/>		
3.1	Die Struktur stat	139
3.1.1	Dateiart und Zugriffsrechte einer Datei abfragen mit st_mode	141
3.1.2	User-ID-Bit und Group-ID-Bit ermitteln mit st_uid und st_gid	145
3.1.3	Inodes ermitteln mit st_ino	145

3.1.4	Den Linkzähler ermitteln mit <code>st_nlink</code>	146
3.1.5	Die Größe einer Datei ermitteln mit <code>st_size</code>	151
3.1.6	Zeitdaten mit <code>st_atime</code> , <code>st_mtime</code> und <code>st_ctime</code> abfragen	153
4	Zugriff auf Systeminformationen	157
4.1	Informationen aus dem <code>/proc</code>-Verzeichnis herausziehen	157
4.2	Hardware-/Systeminformationen ermitteln	160
4.2.1	CPU-Informationen – <code>/proc/cpuinfo</code>	160
4.2.2	Geräteinformationen in <code>/proc/devices</code>	161
4.2.3	Die Speicherauslastung mit <code>/proc/meminfo</code> ermitteln	161
4.2.4	Weitere Hardwareinformationen in zusammengefasster Form	162
4.3	Prozessinformationen auslesen	166
4.3.1	Das Verzeichnis <code>/proc/\$pid/cmdline</code>	167
4.3.2	Das Verzeichnis <code>/proc/\$pid/enviro</code> n	168
4.3.3	Das Verzeichnis <code>/proc/self</code>	169
4.3.4	Das Verzeichnis <code>/proc/\$pid/fd/</code>	170
4.3.5	Das Verzeichnis <code>/proc/\$pid/statm</code>	170
4.4	Kernel-Informationen in <code>/proc</code>	171
4.4.1	Das Verzeichnis <code>/proc/locks</code>	178
4.4.2	Das Verzeichnis <code>/proc/modules</code>	179
4.5	Verschiedene Dateisysteme unter Linux verwalten	180
4.5.1	Das Verzeichnis <code>/proc/mounts</code>	180
4.6	Weiterführendes	181
5	Devices – eine einfache Verbindung zur Hardware	183
5.1	Die Gerätedateitypen	183
5.2	Die Gerätedateinummern	185
5.3	Zugriff auf die Gerätedateien	186
5.4	Gerätenamen	188
5.5	Spezielle Gerätedateien	190
5.6	Gerätedateien in der Praxis einsetzen	191
5.6.1	CD auswerfen und Laufwerk wieder schließen	193
5.6.2	CD-ROM-Fähigkeiten	194

5.6.3	Audio-CD abspielen – komplett und einzelne Tracks – Pause, Fortfahren und Stopp	195
5.6.4	Aktuellen Status der Audio-CD ermitteln	200
5.6.5	Das komplette Listing	202
5.6.6	Der Grafik-Framebuffer	208
5.6.7	Zugriff auf Maus und Joystick	211
5.6.8	Zugriff auf die Audiohardware mit ALSA	220
5.6.9	Zugriff auf den Drucker über /dev/lp0	222
6	System- und Benutzerdateien	227
6.1	Die Datei /etc/passwd	227
6.1.1	Die Datei /etc/passwd auswerten	229
6.1.2	getpwuid und getpwnam – einzelne Abfrage von /etc/passwd	230
6.1.3	getpwent, setpwent und endpwent – komplette Abfrage von /etc/passwd	232
6.2	Die Datei /etc/shadow	234
6.2.1	Die Datei /etc/shadow auswerten	235
6.2.2	getspent, setspent und endspent – komplette Abfrage von /etc/shadow	236
6.3	Die Datei /etc/group	239
6.3.1	Die Datei /etc/group auswerten	241
6.3.2	getgrnam und getgrgid – einzelne Einträge aus /etc/group abfragen	241
6.3.3	getgrent, setgrent und endgrent – alle Einträge in /etc/group abfragen	243
6.4	uname – Informationen zum lokalen System erfragen	243
6.5	Das Verzeichnis /etc/skel und Network Information Service (NIS)	245
6.6	Dateien für Netzwerkinformationen	246
7	Prozesse, Dämonen und Zombies	247
7.1	Was ist ein Prozess?	247
7.2	Prozesskomponente	248
7.2.1	Die Prozessnummer (PID)	249
7.2.2	Prozessnummer des Vaterprozesses (PPID)	249
7.2.3	Benutzer- und Gruppennummer eines Prozesses (UID, EUID, GID, EGID)	249

7.2.4	Der Prozessstatus	250
7.2.5	Prozessprioritäten	251
7.2.6	Timesharing-Prozesse	252
7.2.7	Prozessauslagerung auf die Festplatte durch Swapping	257
7.2.8	Steuerterminal	258
7.3	Prozesse überwachen mit ps, top und kpm	258
7.4	Der Lebenszyklus eines Prozesses	261
7.5	Umgebungsvariablen eines Prozesses	263
7.5.1	Einzelne Umgebungsvariablen abfragen	265
7.5.2	Umgebungsvariablen verändern oder hinzufügen mit putenv() und setenv()	266
7.5.3	Löschen von Umgebungsvariablen mit unsetenv() und clearenv()	270
7.6	Ressourcenlimits eines Prozesses	271
7.6.1	Mehr Sicherheit mit Ressourcenlimits	275
7.7	Prozesserkennung	275
7.8	Erzeugung von neuen Prozessen mit fork()	277
7.8.1	Pufferung	280
7.8.2	Was wird bei fork() vererbt und was nicht?	286
7.8.3	Einen Prozess mit veränderter Priorität erzeugen	286
7.9	Warten auf einen anderen Prozess	288
7.10	Die exec-Familie	296
7.10.1	execl()	297
7.10.2	execve()	298
7.10.3	execv()	298
7.10.4	execle()	298
7.10.5	execlp()	299
7.10.6	execvp()	299
7.10.7	Kindprozesse mit exec-Aufruf überlagern	300
7.11	Kommandoaufrufe aus dem Programm – system()	302
7.12	Dämonprozesse	304
7.12.1	Wie ein Prozess zum Dämon gemacht wird	304
7.12.2	Dämon, sprich mit uns	305
7.12.3	Protokollieren von Dämonen mit syslog()	305
7.12.4	syslog() in der Praxis	308
7.12.5	Dämonprozesse mit C erzeugen	311
7.13	Mehr über die Ausführung von Prozessen	315
7.13.1	Einen Dämon beim Booten mit einem init-Skript starten	315

7.13.2	Hintergrundprozesse und Jobkontrolle	323
7.13.3	Prozesse zeitgesteuert ausführen (Cron-Jobs)	327
7.14	Zusammenfassung und Ausblick	330
8	Signale	331
8.1	Grundlagen zu den Signalen	331
8.1.1	Signalmasken	334
8.1.2	Signale bei fork()	334
8.1.3	Signale und exec	335
8.1.4	Übersicht zu den Signalen	335
8.2	Das neue Signalkonzept ab Kernel 2.6.18	338
8.2.1	Wozu ein neues Signalkonzept?	338
8.3	Die Signalmenge initialisieren	339
8.4	Elemente zu der Signalmenge hinzufügen oder aus ihr entfernen	339
8.5	Signale einrichten und abfragen	340
8.5.1	Einen Signalhandler einrichten, der zurückkehrt	344
8.6	Signale an andere Prozesse senden mit kill()	347
8.7	Eine Zeitschaltuhr einrichten mit alarm()	348
8.8	Prozesse stoppen, bis ein Signal eintritt, mit pause()	349
8.9	Prozesse für eine bestimmte Zeit stoppen mit sleep() und usleep()	349
8.10	Die Signalmaske erfragen oder ändern mit sigprocmask()	350
8.11	Einen Prozess während einer Änderung der Signalmaske stoppen mit sigsuspend()	351
8.12	Prozesse synchronisieren	351
9	IPC – Interprozesskommunikation	355
9.1	Unterschiedliche Interprozesskommunikations-Techniken im Überblick	356
9.1.1	Pipes	356
9.1.2	Benannte Pipes (FIFOs)	357
9.1.3	Nachrichtenspeicher (sogenannte Message Queues)	359
9.1.4	Semaphore	360
9.1.5	Shared Memory (gemeinsamer Speicher)	360

9.1.6	STREAMs	361
9.1.7	Sockets	362
9.1.8	Lock Files (Sperrdateien)	362
9.1.9	Dateisperren (Record Locking)	363
9.2	Gründe für IPC	364
9.3	Pipes	365
9.3.1	Eigenschaften von Pipes	365
9.3.2	Pipes einrichten mit pipe()	366
9.3.3	Eigenschaften von elementaren E/A-Funktionen bei Pipes	370
9.3.4	Standard-E/A-Funktionen mit pipe() ausführen	371
9.3.5	Pipes in einen anderen Prozess umleiten	373
9.3.6	Ein Filterprogramm erstellen mithilfe einer Pipe	376
9.3.7	Einrichten einer Pipe zu einem anderen Prozess mit popen()	379
9.3.8	Mail versenden mit Pipes und sendmail	381
9.3.9	Drucken über eine Pipe mit lpr	385
9.3.10	Benannte Pipes – FIFOs	390
9.4	System-V-Interprozesskommunikation	408
9.4.1	Gemeinsamkeiten der SysV-Mechanismen	408
9.4.2	Ein Objekt einrichten, eine Verbindung herstellen und das Objekt wieder löschen	409
9.4.3	Datenaustausch zwischen nicht verwandten Prozessen	410
9.5	Semaphore	410
9.5.1	Lebenszyklus eines Semaphors	411
9.5.2	Ein Semaphor öffnen oder erstellen mit semget()	414
9.5.3	Abfragen, Ändern oder Löschen der Semaphormenge mit semctl()	415
9.5.4	Operationen auf Semaphormengen ausführen mit semop()	417
9.5.5	Semaphore im Vergleich mit Sperren	419
9.6	Message Queues	419
9.6.1	Eine Message Queue öffnen oder erzeugen mit msgget()	420
9.6.2	Nachrichten versenden mit msgsnd()	420
9.6.3	Eine Nachricht empfangen – msgrcv()	421
9.6.4	Abfragen, Ändern oder Löschen einer Message Queue mit msgctl()	422
9.7	Shared Memory	432
9.7.1	Ein Shared-Memory-Segment erstellen oder öffnen mit shmget()	432
9.7.2	Ein Shared-Memory-Segment abfragen, ändern oder löschen mit shmctl() ...	433
9.7.3	Ein Shared-Memory-Segment anbinden (attach) mit shmat()	434
9.7.4	Ein Shared-Memory-Segment loslösen mit shmdt()	434
9.7.5	Client-Server-Beispiel mit Shared-Memory-Segment	435

10	Threads	443
<hr/>		
10.1	Unterschiede zwischen Threads und Prozessen	443
10.2	Thread-Bibliotheken	444
10.3	Kernel- und User-Threads	445
10.4	Scheduling und Zustände von Threads	445
10.5	Die grundlegenden Funktionen der Thread-Programmierung	447
10.5.1	Mit <code>pthread_create</code> einen neuen Thread erzeugen	447
10.5.2	Mit <code>pthread_exit</code> einen Thread beenden	448
10.5.3	Mit <code>pthread_join</code> auf das Ende eines Threads warten	449
10.5.4	Mit <code>pthread_self</code> die ID von Threads ermitteln	449
10.5.5	Mit <code>pthread_equal</code> die ID von zwei Threads vergleichen	456
10.5.6	Mit <code>pthread_detach</code> einen Thread unabhängig vom Hauptprogramm machen	458
10.6	Die Attribute von Threads und das Scheduling	459
10.7	Threads synchronisieren	465
10.7.1	Mutexe	468
10.7.2	Condition-Variablen (Bedingungsvariablen)	477
10.7.3	Threads und Semaphore	489
10.7.4	Weitere Synchronisationstechniken im Überblick	492
10.8	Threads abbrechen (canceln)	493
10.9	Erzeugen von threadspezifischen Daten (TSD)	498
10.10	Mit <code>pthread_once()</code> einen Codeabschnitt auf einmal ausführen	501
10.11	Thread-safe-Funktionen (thread-sichere Funktionen)	504
10.12	Threads und Signale	505
10.13	Zusammenfassung und Ausblick	510
11	Netzwerkprogrammierung	513
<hr/>		
11.1	Einführung	513
11.2	Aufbau von Netzwerken	514
11.2.1	ISO/OSI-Referenzmodell	514
11.2.2	Das World Wide Web (Internet)	518

11.3 TCP/IP – Aufbau und Struktur	520
11.3.1 Die Netzwerkschicht (Datenübertragung)	520
11.3.2 Die Internetschicht	520
11.3.3 Die Transportschicht (TCP, UDP)	521
11.3.4 Anwendungsschicht	522
11.4 TCP-Socket	524
11.5 Das Kommunikationsmodell der Sockets	525
11.6 Grundlegende Funktionen zum Zugriff auf die Socket-Schnittstelle	525
11.6.1 Ein Socket anlegen mit socket()	526
11.6.2 Verbindungsaufbau mit connect()	528
11.6.3 Socket mit einer Adresse verknüpfen durch bind()	531
11.6.4 Auf Verbindungen warten mit listen() und accept()	531
11.6.5 Senden und Empfangen von Daten mit write() und read()	532
11.6.6 Senden und Empfangen von Daten mit send() und recv()	533
11.6.7 Eine Verbindung wieder schließen mit close()	535
11.7 Aufbau eines Clientprogramms	535
11.7.1 Zusammenfassung: Clientanwendung und Quellcode	538
11.8 Aufbau des Serverprogramms	540
11.8.1 Zusammenfassung: Serveranwendung und Quellcode	541
11.9 IP-Adressen konvertieren, manipulieren und extrahieren	545
11.9.1 inet_aton(), inet_pton() und inet_addr()	545
11.9.2 inet_ntoa() und inet_ntop()	547
11.9.3 inet_network()	547
11.9.4 inet_netof()	548
11.9.5 inet_lnaof()	548
11.9.6 inet_makeaddr()	549
11.10 Namen und IP-Adressen ineinander umwandeln	552
11.10.1 Nameserver	553
11.10.2 Informationen zum Rechner im Netz – gethostbyname und gethostbyaddr	553
11.10.3 Service-Informationen ermitteln mit getservbyname() und getservbyport()	558
11.11 Pufferung bei Netzwerk-Sockets	562
11.12 Standard-E/A-Funktionen verwenden	563
11.12.1 Pufferung von Standard-E/A-Funktionen	565
11.13 Parallele Server	565
11.14 Synchrones Multiplexing mit select()	582

11.15 POSIX-Threads und Netzwerkprogrammierung	604
11.16 Optionen für Sockets setzen und abfragen	610
11.16.1 Optionen setzen mit setsockopt()	610
11.16.2 Optionen abfragen mit getsockopt()	610
11.16.3 Verfügbare Socket-Optionen	611
11.17 Das UDP-Protokoll (User Datagram Protocol)	615
11.17.1 Die UDP-Clientanwendung	617
11.17.2 Die UDP-Serveranwendung	618
11.17.3 recvfrom() und sendto()	618
11.17.4 bind() verwenden oder weglassen – das ist hier die Frage	623
11.18 Unix Domain Sockets	624
11.18.1 Die Adressstruktur von Unix Domain Sockets	624
11.18.2 Lokale Sockets erzeugen mit socketpair()	628
11.19 Multicast-Socket	630
11.19.1 Anwendungsgebiete von Multicast-Verbindungen	638
11.20 Nicht blockierende I/O-Sockets	639
11.21 Streams, TLI, Raw Socket und XTI	642
11.21.1 Raw Sockets	642
11.21.2 TLI und XTI	643
11.21.3 RPC (Remote Procedure Call)	644
11.22 IPv4 und IPv6	644
11.22.1 Was IPv6 von IPv4 unterscheidet	645
11.23 Netzwerksoftware nach IPv6 portieren	646
11.23.1 Neue Konstanten bei IPv6	646
11.23.2 Neue Strukturen bei IPv6	647
11.23.3 Neue Funktionen für IPv6	647
11.24 Sicherheit und Verschlüsselung	647
12 MySQL und PostgreSQL	649
<hr/>	
12.1 Relationale Datenbanksysteme	649
12.2 Der relationale Datenbankserver	653
12.3 SQL-Server im Überblick	653
12.4 Die MySQL-Datenbank	654
12.4.1 Anwendungsgebiete von MySQL	655

12.4.2	Die Schnittstellen von MySQL	656
12.4.3	Die Installation von MySQL	656
12.4.4	Den MySQL-Server starten und stoppen	656
12.4.5	Die Konfigurationsdatei my.cnf	657
12.4.6	Kommandozeilenwerkzeuge für mysql	659
12.4.7	Grafische Clients	663
12.4.8	Kleiner MySQL-Crashkurs	664
12.4.9	SQL-Datentypen	665
12.4.10	Datenbank anlegen, verwenden und löschen	668
12.4.11	Eine neue Tabelle anlegen	670
12.4.12	Schlüsselfelder für Tabellenspalten anlegen	672
12.4.13	Indizes	672
12.4.14	Tabellentypen beim Anlegen von Tabellen	673
12.4.15	Autowerte definieren	674
12.4.16	Tabellen umbenennen und ändern	674
12.4.17	Daten einfügen, ändern und löschen	677
12.4.18	Daten importieren	680
12.4.19	Daten ausgeben	680
12.4.20	Was ist NULL? Ist NULL 0 oder undefiniert?	683
12.4.21	Unschärfe Suche	683
12.5	Die MySQL-C-API	684
12.5.1	Eine Verbindung mit dem MySQL-Server aufbauen	686
12.5.2	Aufgetretene Fehler ermitteln mit mysql_errno() und mysql_error()	689
12.5.3	Schließen der Verbindung zum Server mit mysql_close()	689
12.5.4	Ein erstes Beispiel	690
12.5.5	Verschiedene Informationen ermitteln	695
12.5.6	Datenbanken, Tabellen und Felder ausgeben mit MYSQL_RES	699
12.5.7	Ergebnismenge zeilenweise bearbeiten (MYSQL_ROW)	701
12.5.8	Ergebnismenge spaltenweise einlesen und ausgeben (MYSQL_FIELD)	703
12.5.9	Ein weiteres Beispiel	708
12.5.10	Die Ergebnismenge auswerten – weitere wichtige Funktionen	715
12.5.11	Befehle an den Server – mysql_query() und mysql_real_query()	717
12.5.12	Weitere wichtige API-Funktionen	721
12.5.13	Veraltete Funktionen, die nicht mehr benutzt werden sollten	727
12.6	Beispiel: Ein einfaches Newssystem	728
12.6.1	Die Headerdatei my CGI.h	729
12.6.2	Weitere Planung des Projektes	735
12.6.3	Die Datenbank und die Tabellen anlegen	735
12.6.4	MySQL-Clients mit GUI	759
12.6.5	Eine (nicht ganz unwichtige) Randnotiz	759
12.7	Neue SQL-Funktionen für die Shell – MySQL erweitern	759

12.8 MySQL-Funktionen mit der UDF-Schnittstelle entwerfen	760
12.8.1 UDF-Sequenzen erstellen	762
12.8.2 Die UDF_INIT-Struktur	763
12.8.3 Die UDF_ARGS-Struktur	764
12.8.4 Rückgabewerte	765
12.8.5 Benutzerdefinierte Funktionen erstellen	766
12.8.6 Benutzerdefinierte Funktion kompilieren, installieren und ausführen	768
12.9 PostgreSQL: Ein objektrelationales Datenbankverwaltungssystem	771
12.9.1 PostgreSQL im Vergleich zu MySQL	771
12.9.2 Syntax-Unterschiede zwischen MySQL und PostgreSQL	773
12.9.3 PostgreSQL installieren	774
12.9.4 Die Konfigurationsdateien bei PostgreSQL – postgresql.conf und pg_hba.conf	776
12.9.5 Kleiner Crashkurs in PostgreSQL	778
12.9.6 Die PostgreSQL-C-API libpg	786
12.9.7 Umgebungsvariablen und Passwortdatei	809
12.9.8 PostgreSQL und Threads	810
12.9.9 Ausblick	810

13 Terminal-E/A und Benutzerschnittstellen für die Konsole 813

13.1 termios	813
13.1.1 Terminalattribute bearbeiten	815
13.1.2 Flags setzen und löschen	821
13.1.3 Terminalidentifizierung über spezielle Funktionen	829
13.1.4 Die Baudrate von Terminals einstellen	831
13.2 Terminalinformationen in terminfo	837
13.2.1 terminfo verwenden	838
13.2.2 terminfo initialisieren mit setupterm()	840
13.2.3 Eigenschaften eines Terminals abfragen mit tigetflag(), tigetnum() und tigetstr()	840
13.2.4 Mit terminfo-Eigenschaften arbeiten – putp(), tputs() und tparm()	844
13.3 Halbgrafik erstellen mit ncurses	847
13.3.1 ncurses initialisieren	848
13.3.2 Tastaturmodus und Ein- und Ausgabe bei ncurses	849
13.3.3 Eigenschaft von ncurses-Fenstern ändern und abfragen	862
13.3.4 Das Scrolling ein- und ausschalten	865

13.3.5	Schriftattribute und Farben setzen	868
13.3.6	Fensterrountinen in ncurses	874
13.3.7	Mausprogrammierung mit ncurses	881
14	GTK+	891
14.1	Was ist GTK+?	891
14.1.1	Was sind GDK und GLib?	892
14.1.2	Schnittstellen von GTK+ zu anderen Programmiersprachen	893
14.1.3	GTK+ und GNOME	893
14.1.4	GTK-Versionen	894
14.1.5	Zum Aufbau dieses Kapitels	895
14.2	GTK+-Anwendungen übersetzen	896
14.3	Eine Einführung in die GLib-Bibliothek	897
14.3.1	Datentypen in der GLib	898
14.3.2	GLib-Routinen	899
14.3.3	Assertion-Funktionen	901
14.3.4	Speicherverwaltung mit der GLib	903
14.3.5	Stringbearbeitung mit der GLib	906
14.3.6	Der selbstverwaltende Stringpuffer	911
14.3.7	GLib-Timer	914
14.3.8	Dynamische Arrays	916
14.3.9	Listen, Hashtabellen und binäre Bäume	921
14.3.10	Ausblick GLib	923
14.4	Grundlagen der GTK+-Programmierung	923
14.4.1	Die Umgebung initialisieren	924
14.4.2	Widgets erzeugen und gegebenenfalls die Attribute setzen	925
14.4.3	Eine Callback-Funktion einrichten, um Events abzufangen	927
14.4.4	Eine GTK+-Anwendung beenden	930
14.4.5	Die hierarchische Anordnung der Widgets definieren	931
14.4.6	Widgets anzeigen	932
14.4.7	Signale und Events abfangen und bearbeiten – die Event-Verarbeitungsschleife	933
14.4.8	GTK+ und Umlaute – etwas über die Zeichenkodierung unter GTK+	934
14.5	Fenster anlegen mit GtkWindow	935
14.5.1	Dialogfenster (sogenannte Dialogboxen) erstellen	939
14.5.2	GtkMessageDialog	944

14.6 Anzeigeelemente	944
14.6.1 Text anzeigen mit GtkLabel	947
14.6.2 Trennlinien erstellen mit GtkSeparator	951
14.6.3 Grafiken anzeigen mit GtkImage	951
14.6.4 Statusleisten erstellen mit GtkStatusbar	952
14.6.5 Fortschrittsbalken erstellen mit GtkProgressBar	952
14.7 Behälter für Widgets	953
14.7.1 Boxen (GtkBox)	953
14.7.2 Aufteilungen, Register und Button-Boxen	955
14.7.3 Tabellen erstellen mit GtkTable	962
14.7.4 Ausrichten von Widgets mit GtkAlignment	967
14.8 Buttons und Toggled-Buttons	967
14.8.1 Allgemeines über Buttons	974
14.8.2 Radio-Buttons erstellen mit GtkRadioButton	975
14.8.3 GtkRadioButton, GtkCheckButton und GtkToggleButton	976
14.8.4 Signale für Buttons (GtkButton)	976
14.9 Dateneingaben auswerten	977
14.9.1 Textfelder erstellen mit GtkEntry	984
14.9.2 Schieberegler erstellen mit GtkScale	986
14.9.3 Zahlenfelder erstellen mit GtkSpinButton	987
14.9.4 Einstellungen mit GtkAdjustment	988
14.9.5 GtkEditable	989
14.10 Menüs und Toolbars erstellen	989
14.10.1 Menüs auf die einfache Art erstellen mit GtkItemFactory	995
14.10.2 Toolbars erzeugen mit GtkToolbar	1001
14.10.3 Optionsmenüs erzeugen mit GtkOptionsMenu	1004
14.10.4 Combo-Boxen erstellen mit GtkCombo	1005
14.11 Mehrzeiligen Text erstellen	1009
14.11.1 Texteditoren erstellen – GtkTextView, GtkTextBuffer	1018
14.11.2 Scrollende Fenster erstellen mit GtkScrolledWindow	1022
14.12 Auswählen von Widgets (Selection)	1023
14.12.1 Eine Dateiauswahl erstellen mit GtkFileSelection	1031
14.13 Events auswerten	1033
14.14 Weitere Widget- und GTK+-Elemente im Überblick	1039
14.14.1 Mit GTK+ Bäume und Listen erstellen	1039
14.14.2 Lineale	1040
14.14.3 Zwischenablage (Clipboard)	1040
14.14.4 Drag and Drop	1040

14.14.5	Stock-Items – Repertoire-Einträge	1040
14.14.6	Signale	1041
14.14.7	Ressource-Files	1041
14.14.8	Widget-Entwicklung	1041
14.14.9	GDK	1041
15	Übersicht über weitere beliebte GUI-Bibliotheken	1043
15.1	gtkmm – GTK+ für C++	1043
15.1.1	Programmbeispiele	1044
15.1.2	GUI-Designer Glade für GTK+ und gtkmm	1046
15.2	wxWidgets	1047
15.2.1	Ein einfaches Programmbeispiel für wxWidgets	1048
15.2.2	GUI-Designer wxFormBuilder	1050
15.3	FLTK	1051
15.3.1	Ein einfaches FLTK-Programmbeispiel	1051
15.3.2	GUI-Designer FLUID	1053
15.4	Qt	1053
15.4.1	Ein Qt-Programmbeispiel	1054
15.4.2	Der GUI-Designer von Qt	1055
15.5	Die niedrige Ebene: X-Window-Programmierung	1056
15.6	Multimediabibliotheken	1057
15.6.1	Die SDL-Bibliothek	1057
15.6.2	Die Allegro-Bibliothek	1057
15.6.3	OpenGL (bzw. Mesa 3D)	1059
15.6.4	Programmbeispiel für GLUT	1060
16	Werkzeuge für Programmierer	1065
16.1	Der Compiler GCC	1065
16.1.1	Standardgebrauch des GCC	1067
16.1.2	Dazulinken von Programmbibliotheken	1068
16.1.3	Dateien, die GCC kennt	1069
16.1.4	Ausgabedateien bei jedem einzelnen Schritt der Übersetzung erstellen lassen	1070
16.1.5	Noch mehr Optionen für den GCC	1071

16.1.6	Optionen für Warnmeldungen	1071
16.1.7	Präprozessor-Optionen	1072
16.1.8	Debugging und Profiling	1073
16.1.9	Optimierungsflags	1073
16.2	make	1074
16.2.1	Erzeugen eines Makefiles	1076
16.2.2	Variablen, Makros und Abkürzungen	1083
16.2.3	Implizite Regeln	1087
16.2.4	Musterregeln	1088
16.2.5	make zur Installation verwenden	1089
16.2.6	make-Optionen	1090
16.2.7	Ausblick	1090
16.3	Eigene Bibliotheken erstellen	1091
16.3.1	Statische Bibliotheken erstellen	1091
16.3.2	Dynamische Bibliotheken (Shared Libraries) erstellen	1095
16.3.3	Dynamisches Nachladen von Bibliotheken	1098
16.4	RPM-Pakete	1101
16.4.1	Einführung in das RPM-System	1102
16.4.2	Verzeichnisse, die RPM benötigt	1104
16.4.3	Ein eigenes RPM-Paket erstellen	1105
16.4.4	Sources	1105
16.4.5	Die Spec-Datei	1106
16.4.6	Das Paket erstellen	1110
16.4.7	Das Paket installieren	1112
16.5	RCS und CVS	1114
16.5.1	Software-Configuration-Management-Systeme (SCM)	1114
16.5.2	RCS	1116
16.5.3	CVS	1126
16.6	Laufzeitmessung von Programmen	1143
16.6.1	Einfache Zeitmessung mit TIME – die Laufzeit von Prozessen ermitteln	1143
16.6.2	Profiling mit GPROF – Laufzeit von Funktionen	1144
16.6.3	Analyse mit GCOV	1148
16.7	Debuggen mit GDB und DDD	1151
16.8	STRACE – Systemaufrufe verfolgen	1164
16.9	Memory Leaks und unerlaubte Speicherzugriffe	1167
16.9.1	efence	1167
16.9.2	valgrind	1171
16.10	Ausblick	1175

Anhang	1177
A Sicherheit unter Linux	1179
B Funktionsreferenz	1201
C Linux/UNIX-Kommandoreferenz	1283
Index	1365

Materialien zum Buch

Auf der Webseite zu diesem Buch stehen folgende Materialien für Sie zum Download bereit:

► **alle Beispielprogramme**

Gehen Sie auf www.rheinwerk-verlag.de/5774. Klicken Sie auf den Reiter MATERIALIEN. Sie sehen die herunterladbaren Dateien samt einer Kurzbeschreibung des Dateiinhalts. Klicken Sie auf den Button HERUNTERLADEN, um den Download zu starten. Je nach Größe der Datei (und Ihrer Internetverbindung) kann es einige Zeit dauern, bis der Download abgeschlossen ist.