

The GNU C Library Reference Manual Volume Two

GNU C 库技术手册 卷2

(英文版)



Sandra Loosemore
with
Richard M. Stallman
Roland McGrath
Andrew Oram
Ulrich Drepper

著

GNU技术文档精粹

GNU C库技术手册

卷2

(英文版)

The GNU C Library Reference Manual

Volume Two

Sandra Loosemore

with

(美) Richard M. Stallman

Roland McGrath

Andrew Oram

Ulrich Drepper

Free Software Foundation

江苏工业学院图书馆

藏书章

Sandra Loosemore with Richard. M. Stallman, Roland McGrath, Andrew Oram, and Ulrich Drepper: The GNU C Library Reference Manual, Volume Two.

Copyright © 1988, 1989, 1990, 1991, 1992, 1993, 1994, 1995, 1996, 1997, 1998, 2000 Free Software Foundation, Inc.

Permission is granted to make and distribute verbatim copies of this manual provided the copyright notice and this permission notice are preserved on all copies.

Permission is granted to copy and distribute modified versions of this manual under the conditions for verbatim copying, provided also that the entire resulting derived work is distributed under the terms of a permission notice identical to this one.

Permission is granted to copy and distribute translations of this manual into another language, under the above conditions for modified versions.

图书在版编目(CIP)数据

GNU C库技术手册 卷2: 英文版/ (美) 鲁斯摩尔 (Loosemore, S.) 等著, - 影印版.
- 北京: 机械工业出版社, 2000.8

(GNU技术文档精粹)

书名原文: The GNU C Library Reference Manual, Volume Two

ISBN 7-111-08176-5

I. G… II. 鲁… III. 操作系统, 英文 IV. TP316.81

中国版本图书馆CIP数据核字(2000)第37636号

机械工业出版社(北京市西城区百万庄大街22号 邮政编码 100037)

北京昌平第二印刷厂印刷·新华书店北京发行所发行

2000年8月第1版第1次印刷

787mm × 1092mm 1/16 · 34.75印张

印数: 0 001-5 000册

定价: 120.00元 (1、2卷)

凡购本书, 如有倒页、脱页、缺页, 由本社发行部调换

Table of Contents

1	Introduction	1
1.1	Getting Started	1
1.2	Standards and Portability	1
1.2.1	ISO C	2
1.2.2	POSIX (The Portable Operating System Interface)	2
1.2.3	Berkeley Unix	3
1.2.4	SVID (The System V Interface Description)	3
1.2.5	XPG (The X/Open Portability Guide)	4
1.3	Using the Library	4
1.3.1	Header Files	4
1.3.2	Macro Definitions of Functions	5
1.3.3	Reserved Names	6
1.3.4	Feature Test Macros	8
1.4	Roadmap to the Manual	12
2	Error Reporting	17
2.1	Checking for Errors	17
2.2	Error Codes	18
2.3	Error Messages	30
3	Memory Allocation	33
3.1	Dynamic Memory Allocation Concepts	33
3.2	Dynamic Allocation and C	33
3.3	Unconstrained Allocation	34
3.3.1	Basic Storage Allocation	34
3.3.2	Examples of <code>malloc</code>	35
3.3.3	Freeing Memory Allocated with <code>malloc</code>	36
3.3.4	Changing the Size of a Block	37
3.3.5	Allocating Cleared Space	38
3.3.6	Efficiency Considerations for <code>malloc</code>	39
3.3.7	Allocating Aligned Memory Blocks	39
3.3.8	Malloc Tunable Parameters	39
3.3.9	Heap Consistency Checking	40
3.3.10	Storage Allocation Hooks	42
3.3.11	Statistics for Storage Allocation with <code>malloc</code>	45
3.3.12	Summary of <code>malloc</code> -Related Functions	46
3.4	Allocation Debugging	47
3.4.1	How to install the tracing functionality	47
3.4.2	Example program excerpts	48
3.4.3	Some more or less clever ideas	48

3.4.4	Interpreting the traces	49
3.5	Obstacks	51
3.5.1	Creating Obstacks	51
3.5.2	Preparing for Using Obstacks	52
3.5.3	Allocation in an Obstack	53
3.5.4	Freeing Objects in an Obstack	54
3.5.5	Obstack Functions and Macros	55
3.5.6	Growing Objects	56
3.5.7	Extra Fast Growing Objects	57
3.5.8	Status of an Obstack	59
3.5.9	Alignment of Data in Obstacks	60
3.5.10	Obstack Chunks	60
3.5.11	Summary of Obstack Functions	61
3.6	Automatic Storage with Variable Size	63
3.6.1	<code>alloca</code> Example	63
3.6.2	Advantages of <code>alloca</code>	64
3.6.3	Disadvantages of <code>alloca</code>	65
3.6.4	GNU C Variable-Size Arrays	65

4 Character Handling 67

4.1	Classification of Characters	67
4.2	Case Conversion	69
4.3	Character class determination for wide characters	70
4.4	Notes on using the wide character classes	74
4.5	Mapping of wide characters	75

5 String and Array Utilities 77

5.1	Representation of Strings	77
5.2	String and Array Conventions	78
5.3	String Length	78
5.4	Copying and Concatenation	79
5.5	String/Array Comparison	88
5.6	Collation Functions	91
5.7	Search Functions	94
5.8	Finding Tokens in a String	97
5.9	Encode Binary Data	100
5.10	Argz and Envz Vectors	102
5.10.1	Argz Functions	102
5.10.2	Envz Functions	104

6	Character Set Handling	107
6.1	Introduction to Extended Characters	107
6.2	Overview about Character Handling Functions	111
6.3	Restartable Multibyte Conversion Functions.....	111
6.3.1	Selecting the conversion and its properties.....	112
6.3.2	Representing the state of the conversion	113
6.3.3	Converting Single Characters.....	114
6.3.4	Converting Multibyte and Wide Character Strings	121
6.3.5	A Complete Multibyte Conversion Example	125
6.4	Non-reentrant Conversion Function	127
6.4.1	Non-reentrant Conversion of Single Characters..	127
6.4.2	Non-reentrant Conversion of Strings.....	129
6.4.3	States in Non-reentrant Functions.....	130
6.5	Generic Charset Conversion.....	131
6.5.1	Generic Character Set Conversion Interface.....	132
6.5.2	A complete <code>iconv</code> example.....	136
6.5.3	Some Details about other <code>iconv</code> Implementations	138
6.5.4	The <code>iconv</code> Implementation in the GNU C library	140
6.5.4.1	Format of ' <code>gconv-modules</code> ' files.....	141
6.5.4.2	Finding the conversion path in <code>iconv</code> ..	143
6.5.4.3	<code>iconv</code> module data structures.....	143
6.5.4.4	<code>iconv</code> module interfaces	147
7	Locales and Internationalization	157
7.1	What Effects a Locale Has	157
7.2	Choosing a Locale.....	158
7.3	Categories of Activities that Locales Affect.....	158
7.4	How Programs Set the Locale	159
7.5	Standard Locales.....	161
7.6	Accessing Locale Information	162
7.6.1	<code>localeconv</code> : It is portable but	162
7.6.1.1	Generic Numeric Formatting Parameters	163
7.6.1.2	Printing the Currency Symbol	164
7.6.1.3	Printing the Sign of a Monetary Amount	166
7.6.2	Pinpoint Access to Locale Data.....	167
7.7	A dedicated function to format numbers	172

8	Message Translation	177
8.1	X/Open Message Catalog Handling	177
8.1.1	The catgets function family	178
8.1.2	Format of the message catalog files	181
8.1.3	Generate Message Catalogs files	183
8.1.4	How to use the catgets interface	185
8.1.4.1	Not using symbolic names	185
8.1.4.2	Using symbolic names	185
8.1.4.3	How does to this allow to develop	186
8.2	The Uniform approach to Message Translation	188
8.2.1	The gettext family of functions	188
8.2.1.1	What has to be done to translate a message?	189
8.2.1.2	How to determine which catalog to be used	191
8.2.1.3	User influence on gettext	193
8.2.2	Programs to handle message catalogs for gettext	196
9	Searching and Sorting	199
9.1	Defining the Comparison Function	199
9.2	Array Search Function	199
9.3	Array Sort Function	200
9.4	Searching and Sorting Example	201
9.5	The hsearch function	204
9.6	The tsearch function	207
10	Pattern Matching	211
10.1	Wildcard Matching	211
10.2	Globbing	212
10.2.1	Calling glob	212
10.2.2	Flags for Globbing	214
10.2.3	More Flags for Globbing	216
10.3	Regular Expression Matching	218
10.3.1	POSIX Regular Expression Compilation	218
10.3.2	Flags for POSIX Regular Expressions	220
10.3.3	Matching a Compiled POSIX Regular Expression	221
10.3.4	Match Results with Subexpressions	222
10.3.5	Complications in Subexpression Matching	223
10.3.6	POSIX Regexp Matching Cleanup	223
10.4	Shell-Style Word Expansion	224
10.4.1	The Stages of Word Expansion	225
10.4.2	Calling wordexp	225
10.4.3	Flags for Word Expansion	227

10.4.4	wordexp Example	228
10.4.5	Details of Tilde Expansion	229
10.4.6	Details of Variable Substitution	229
11	Input/Output Overview	233
11.1	Input/Output Concepts	233
11.1.1	Streams and File Descriptors	233
11.1.2	File Position	234
11.2	File Names	235
11.2.1	Directories	235
11.2.2	File Name Resolution	236
11.2.3	File Name Errors	237
11.2.4	Portability of File Names	238
12	Input/Output on Streams	239
12.1	Streams	239
12.2	Standard Streams	239
12.3	Opening Streams	240
12.4	Closing Streams	243
12.5	Simple Output by Characters or Lines	244
12.6	Character Input	245
12.7	Line-Oriented Input	246
12.8	Unreading	248
12.8.1	What Unreading Means	248
12.8.2	Using ungetc To Do Unreading	249
12.9	Block Input/Output	250
12.10	Formatted Output	251
12.10.1	Formatted Output Basics	251
12.10.2	Output Conversion Syntax	252
12.10.3	Table of Output Conversions	254
12.10.4	Integer Conversions	255
12.10.5	Floating-Point Conversions	257
12.10.6	Other Output Conversions	259
12.10.7	Formatted Output Functions	261
12.10.8	Dynamically Allocating Formatted Output ...	262
12.10.9	Variable Arguments Output Functions	263
12.10.10	Parsing a Template String	266
12.10.11	Example of Parsing a Template String	267
12.11	Customizing printf	269
12.11.1	Registering New Conversions	269
12.11.2	Conversion Specifier Options	270
12.11.3	Defining the Output Handler	272
12.11.4	printf Extension Example	273
12.11.5	Predefined printf Handlers	274
12.12	Formatted Input	275

12.12.1	Formatted Input Basics	276
12.12.2	Input Conversion Syntax	277
12.12.3	Table of Input Conversions	278
12.12.4	Numeric Input Conversions	279
12.12.5	String Input Conversions	281
12.12.6	Dynamically Allocating String Conversions ...	282
12.12.7	Other Input Conversions	283
12.12.8	Formatted Input Functions	283
12.12.9	Variable Arguments Input Functions	284
12.13	End-Of-File and Errors	285
12.14	Text and Binary Streams	285
12.15	File Positioning	286
12.16	Portable File-Position Functions	289
12.17	Stream Buffering	291
12.17.1	Buffering Concepts	292
12.17.2	Flushing Buffers	292
12.17.3	Controlling Which Kind of Buffering	293
12.18	Other Kinds of Streams	295
12.18.1	String Streams	295
12.18.2	Obstack Streams	297
12.18.3	Programming Your Own Custom Streams	298
	12.18.3.1 Custom Streams and Cookies	298
	12.18.3.2 Custom Stream Hook Functions	300
12.19	Formatted Messages	301
12.19.1	Printing Formatted Messages	301
12.19.2	Adding Severity Classes	304
12.19.3	How to use <code>fmtmsg</code> and <code>addseverity</code>	304

13 Low-Level Input/Output 307

13.1	Opening and Closing Files	307
13.2	Input and Output Primitives	310
13.3	Setting the File Position of a Descriptor	315
13.4	Descriptors and Streams	318
13.5	Dangers of Mixing Streams and Descriptors	319
13.5.1	Linked Channels	320
13.5.2	Independent Channels	320
13.5.3	Cleaning Streams	321
13.6	Fast Scatter-Gather I/O	321
13.7	Memory-mapped I/O	323
13.8	Waiting for Input or Output	326
13.9	Synchronizing I/O operations	330
13.10	Perform I/O Operations in Parallel	331
13.10.1	Asynchronous Read and Write Operations	334
13.10.2	Getting the Status of AIO Operations	338
13.10.3	Getting into a Consistent State	340
13.10.4	Cancellation of AIO Operations	342

13.10.5	How to optimize the AIO implementation	343
13.11	Control Operations on Files	344
13.12	Duplicating Descriptors	345
13.13	File Descriptor Flags	347
13.14	File Status Flags	349
13.14.1	File Access Modes	349
13.14.2	Open-time Flags	350
13.14.3	I/O Operating Modes	352
13.14.4	Getting and Setting File Status Flags	353
13.15	File Locks	354
13.16	Interrupt-Driven Input	358
13.17	Generic I/O Control operations	359
14	File System Interface	361
14.1	Working Directory	361
14.2	Accessing Directories	363
14.2.1	Format of a Directory Entry	363
14.2.2	Opening a Directory Stream	364
14.2.3	Reading and Closing a Directory Stream	365
14.2.4	Simple Program to List a Directory	366
14.2.5	Random Access in a Directory Stream	367
14.2.6	Scanning the Content of a Directory	367
14.2.7	Simple Program to List a Directory, Mark II	369
14.3	Working with Directory Trees	370
14.4	Hard Links	374
14.5	Symbolic Links	375
14.6	Deleting Files	377
14.7	Renaming Files	378
14.8	Creating Directories	379
14.9	File Attributes	380
14.9.1	The meaning of the File Attributes	380
14.9.2	Reading the Attributes of a File	385
14.9.3	Testing the Type of a File	386
14.9.4	File Owner	388
14.9.5	The Mode Bits for Access Permission	389
14.9.6	How Your Access to a File is Decided	391
14.9.7	Assigning File Permissions	392
14.9.8	Testing Permission to Access a File	394
14.9.9	File Times	395
14.9.10	File Size	397
14.10	Making Special Files	400
14.11	Temporary Files	401

15	Pipes and FIFOs	405
15.1	Creating a Pipe	405
15.2	Pipe to a Subprocess	407
15.3	FIFO Special Files	409
15.4	Atomicity of Pipe I/O	410
16	Sockets	411
16.1	Socket Concepts	411
16.2	Communication Styles	412
16.3	Socket Addresses	413
16.3.1	Address Formats	414
16.3.2	Setting the Address of a Socket	415
16.3.3	Reading the Address of a Socket	416
16.4	Interface Naming	417
16.5	The Local Namespace	418
16.5.1	Local Namespace Concepts	418
16.5.2	Details of Local Namespace	418
16.5.3	Example of Local-Namespace Sockets	419
16.6	The Internet Namespace	420
16.6.1	Internet Socket Address Formats	421
16.6.2	Host Addresses	422
16.6.2.1	Internet Host Addresses	422
16.6.2.2	Host Address Data Type	424
16.6.2.3	Host Address Functions	425
16.6.2.4	Host Names	427
16.6.3	Internet Ports	431
16.6.4	The Services Database	432
16.6.5	Byte Order Conversion	433
16.6.6	Protocols Database	434
16.6.7	Internet Socket Example	436
16.7	Other Namespaces	437
16.8	Opening and Closing Sockets	437
16.8.1	Creating a Socket	437
16.8.2	Closing a Socket	438
16.8.3	Socket Pairs	439
16.9	Using Sockets with Connections	440
16.9.1	Making a Connection	440
16.9.2	Listening for Connections	441
16.9.3	Accepting Connections	442
16.9.4	Who is Connected to Me?	443
16.9.5	Transferring Data	444
16.9.5.1	Sending Data	444
16.9.5.2	Receiving Data	445
16.9.5.3	Socket Data Options	446
16.9.6	Byte Stream Socket Example	446

16.9.7	Byte Stream Connection Server Example	448
16.9.8	Out-of-Band Data	450
16.10	Datagram Socket Operations	454
16.10.1	Sending Datagrams	454
16.10.2	Receiving Datagrams	455
16.10.3	Datagram Socket Example	455
16.10.4	Example of Reading Datagrams	457
16.11	The <code>inetd</code> Daemon	458
16.11.1	<code>inetd</code> Servers	459
16.11.2	Configuring <code>inetd</code>	459
16.12	Socket Options	460
16.12.1	Socket Option Functions	460
16.12.2	Socket-Level Options	461
16.13	Networks Database	462

17 Low-Level Terminal Interface **465**

17.1	Identifying Terminals	465
17.2	I/O Queues	466
17.3	Two Styles of Input: Canonical or Not	466
17.4	Terminal Modes	467
17.4.1	Terminal Mode Data Types	467
17.4.2	Terminal Mode Functions	468
17.4.3	Setting Terminal Modes Properly	470
17.4.4	Input Modes	471
17.4.5	Output Modes	473
17.4.6	Control Modes	474
17.4.7	Local Modes	476
17.4.8	Line Speed	479
17.4.9	Special Characters	480
17.4.9.1	Characters for Input Editing	481
17.4.9.2	Characters that Cause Signals	483
17.4.9.3	Special Characters for Flow Control ..	484
17.4.9.4	Other Special Characters	484
17.4.10	Noncanonical Input	485
17.5	Line Control Functions	487
17.6	Noncanonical Mode Example	489
17.7	Pseudo-Terminals	491
17.7.1	Allocating Pseudo-Terminals	491
17.7.2	Opening a Pseudo-Terminal Pair	493

18	Mathematics	495
18.1	Predefined Mathematical Constants	495
18.2	Trigonometric Functions	496
18.3	Inverse Trigonometric Functions	498
18.4	Exponentiation and Logarithms	499
18.5	Hyperbolic Functions	504
18.6	Special Functions	505
18.7	Pseudo-Random Numbers	508
18.7.1	ISO C Random Number Functions	508
18.7.2	BSD Random Number Functions	509
18.7.3	SVID Random Number Function	510
18.8	Is Fast Code or Small Code preferred?	515
19	Arithmetic Functions	517
19.1	Floating Point Numbers	517
19.2	Floating-Point Number Classification Functions	517
19.3	Errors in Floating-Point Calculations	519
19.3.1	FP Exceptions	519
19.3.2	Infinity and NaN	521
19.3.3	Examining the FPU status word	523
19.3.4	Error Reporting by Mathematical Functions	524
19.4	Rounding Modes	525
19.5	Floating-Point Control Functions	527
19.6	Arithmetic Functions	528
19.6.1	Absolute Value	528
19.6.2	Normalization Functions	529
19.6.3	Rounding Functions	531
19.6.4	Remainder Functions	532
19.6.5	Setting and modifying single bits of FP values	533
19.6.6	Floating-Point Comparison Functions	535
19.6.7	Miscellaneous FP arithmetic functions	536
19.7	Complex Numbers	537
19.8	Projections, Conjugates, and Decomposing of Complex Numbers	538
19.9	Integer Division	539
19.10	Parsing of Numbers	540
19.10.1	Parsing of Integers	541
19.10.2	Parsing of Floats	544
19.11	Old-fashioned System V number-to-string functions	545

20	Date and Time	549
20.1	Processor Time	549
20.1.1	Basic CPU Time Inquiry	549
20.1.2	Detailed Elapsed CPU Time Inquiry	550
20.2	Calendar Time	551
20.2.1	Simple Calendar Time	552
20.2.2	High-Resolution Calendar	552
20.2.3	Broken-down Time	555
20.2.4	Formatting Date and Time	558
20.2.5	Convert textual time and date information back	564
20.2.5.1	Interpret string according to given format	564
20.2.5.2	A user-friendlier way to parse times and dates	570
20.2.6	Specifying the Time Zone with TZ	572
20.2.7	Functions and Variables for Time Zones	575
20.2.8	Time Functions Example	576
20.3	Precision Time	576
20.4	Setting an Alarm	579
20.5	Sleeping	582
20.6	Resource Usage	583
20.7	Limiting Resource Usage	585
20.8	Process Priority	588
21	Non-Local Exits	591
21.1	Introduction to Non-Local Exits	591
21.2	Details of Non-Local Exits	593
21.3	Non-Local Exits and Signals	594
22	Signal Handling	595
22.1	Basic Concepts of Signals	595
22.1.1	Some Kinds of Signals	595
22.1.2	Concepts of Signal Generation	596
22.1.3	How Signals Are Delivered	596
22.2	Standard Signals	597
22.2.1	Program Error Signals	598
22.2.2	Termination Signals	601
22.2.3	Alarm Signals	602
22.2.4	Asynchronous I/O Signals	603
22.2.5	Job Control Signals	603
22.2.6	Operation Error Signals	605
22.2.7	Miscellaneous Signals	606
22.2.8	Signal Messages	607
22.3	Specifying Signal Actions	608

22.3.1	Basic Signal Handling	608
22.3.2	Advanced Signal Handling	610
22.3.3	Interaction of signal and sigaction	612
22.3.4	sigaction Function Example	612
22.3.5	Flags for sigaction	614
22.3.6	Initial Signal Actions	614
22.4	Defining Signal Handlers	615
22.4.1	Signal Handlers that Return	616
22.4.2	Handlers That Terminate the Process	617
22.4.3	Nonlocal Control Transfer in Handlers	618
22.4.4	Signals Arriving While a Handler Runs	619
22.4.5	Signals Close Together Merge into One	620
22.4.6	Signal Handling and Nonreentrant Functions ..	623
22.4.7	Atomic Data Access and Signal Handling	624
	22.4.7.1 Problems with Non-Atomic Access ...	625
	22.4.7.2 Atomic Types	626
	22.4.7.3 Atomic Usage Patterns	626
22.5	Primitives Interrupted by Signals	627
22.6	Generating Signals	628
	22.6.1 Signaling Yourself	628
	22.6.2 Signaling Another Process	629
	22.6.3 Permission for using kill	630
	22.6.4 Using kill for Communication	631
22.7	Blocking Signals	633
	22.7.1 Why Blocking Signals is Useful	633
	22.7.2 Signal Sets	633
	22.7.3 Process Signal Mask	635
	22.7.4 Blocking to Test for Delivery of a Signal	636
	22.7.5 Blocking Signals for a Handler	637
	22.7.6 Checking for Pending Signals	638
	22.7.7 Remembering a Signal to Act On Later	639
22.8	Waiting for a Signal	641
	22.8.1 Using pause	641
	22.8.2 Problems with pause	641
	22.8.3 Using sigsuspend	642
22.9	Using a Separate Signal Stack	643
22.10	BSD Signal Handling	646
	22.10.1 BSD Function to Establish a Handler	646
	22.10.2 BSD Functions for Blocking Signals	647

23	Process Startup and Termination	649
23.1	Program Arguments	649
23.1.1	Program Argument Syntax Conventions	650
23.1.2	Parsing Program Arguments	651
23.2	Parsing program options using getopt	651
23.2.1	Using the getopt function	651
23.2.2	Example of Parsing Arguments with getopt	652
23.2.3	Parsing Long Options with getopt_long	654
23.2.4	Example of Parsing Long Options with getopt_long	656
23.3	Parsing Program Options with Argp	659
23.3.1	The argp_parse Function	659
23.3.2	Argp Global Variables	660
23.3.3	Specifying Argp Parsers	660
23.3.4	Specifying Options in an Argp Parser	662
23.3.4.1	Flags for Argp Options	663
23.3.5	Argp Parser Functions	664
23.3.5.1	Special Keys for Argp Parser Functions	665
23.3.5.2	Functions For Use in Argp Parsers	667
23.3.5.3	Argp Parsing State	668
23.3.6	Combining Multiple Argp Parsers	670
23.3.7	Flags for argp_parse	671
23.3.8	Customizing Argp Help Output	672
23.3.8.1	Special Keys for Argp Help Filter Functions	672
23.3.9	The argp_help Function	673
23.3.10	Flags for the argp_help Function	673
23.3.11	Argp Examples	675
23.3.11.1	A Minimal Program Using Argp	675
23.3.11.2	A Program Using Argp with Only Default Options	675
23.3.11.3	A Program Using Argp with User Options	677
23.3.11.4	A Program Using Multiple Combined Argp Parsers	681
23.3.12	Argp User Customization	685
23.3.12.5	Parsing of Suboptions	686
23.3.13	Parsing of Suboptions Example	686
23.4	Environment Variables	688
23.4.1	Environment Access	689
23.4.2	Standard Environment Variables	691
23.5	Program Termination	693
23.5.1	Normal Termination	693
23.5.2	Exit Status	694
23.5.3	Cleanups on Exit	695

23.5.4	Aborting a Program	696
23.5.5	Termination Internals	696
24	Processes	699
24.1	Running a Command	699
24.2	Process Creation Concepts	700
24.3	Process Identification	700
24.4	Creating a Process	701
24.5	Executing a File	702
24.6	Process Completion	705
24.7	Process Completion Status	708
24.8	BSD Process Wait Functions	709
24.9	Process Creation Example	710
25	Job Control	713
25.1	Concepts of Job Control	713
25.2	Job Control is Optional	714
25.3	Controlling Terminal of a Process	714
25.4	Access to the Controlling Terminal	715
25.5	Orphaned Process Groups	715
25.6	Implementing a Job Control Shell	716
25.6.1	Data Structures for the Shell	716
25.6.2	Initializing the Shell	718
25.6.3	Launching Jobs	720
25.6.4	Foreground and Background	724
25.6.5	Stopped and Terminated Jobs	725
25.6.6	Continuing Stopped Jobs	730
25.6.7	The Missing Pieces	730
25.7	Functions for Job Control	731
25.7.1	Identifying the Controlling Terminal	731
25.7.2	Process Group Functions	732
25.7.3	Functions for Controlling Terminal Access	734