```
**********************************************************
   10887 Wed Jul 16 14:05:06 2014
new/usr/src/cmd/Makefile
mandoc import
**********************************************************
   1 #
   2 # CDDL HEADER START
   3 #
   4 # The contents of this file are subject to the terms of the
   5 # Common Development and Distribution License (the "License").
   6 # You may not use this file except in compliance with the License.
   7 #
   8 # You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
   9 # or http://www.opensolaris.org/os/licensing.
  10 # See the License for the specific language governing permissions
  11 # and limitations under the License.
  12 #
  13 # When distributing Covered Code, include this CDDL HEADER in each
  14 # file and include the License file at usr/src/OPENSOLARIS.LICENSE.
  15 # If applicable, add the following below this CDDL HEADER, with the
  16 # fields enclosed by brackets "[]" replaced with your own identifying
  17 # information: Portions Copyright [yyyy] [name of copyright owner]
  18 #
  19 # CDDL HEADER END
  20 #

  22 # Copyright (c) 1989, 2010, Oracle and/or its affiliates. All rights reserved.
  23 # Copyright 2010 Nexenta Systems, Inc.  All rights reserved.
  24 # Copyright (c) 2012 Joyent, Inc.  All rights reserved.
  25 # Copyright (c) 2012 by Delphix. All rights reserved.
  26 # Copyright (c) 2013 DEY Storage Systems, Inc. All rights reserved.
  27 # Copyright 2014 Garrett D'Amore <garrett@damore.org>

  29 include ../Makefile.master

  31 #
  32 #       Note that the commands 'lp', and 'perl' are first in
  31 #       Note that the commands 'agents', 'lp', 'perl', and 'man' are first in
  33 #       the list, violating alphabetical order.  This is because they are very
  34 #       long-running and should be given the most wall-clock time for a
  35 #       parallel build.
  36 #
  37 #       Commands in the FIRST_SUBDIRS list are built before starting the build
  38 #       of other commands.  Currently this includes only 'isaexec' and
  39 #       'platexec'.  This is necessary because $(ROOT)/usr/lib/isaexec or
  40 #       $(ROOT)/usr/lib/platexec must exist when some other commands are built
  41 #       because their 'make install' creates a hard link to one of them.
  42 #
  43 #       Commands are listed one per line so that TeamWare can auto-merge most
  44 #       changes.
  45 #

  47 FIRST_SUBDIRS=          \
  48         isaexec         \
  49         platexec

  51 COMMON_SUBDIRS=         \
  52         allocate        \
  53         availdevs       \
  54         lp              \
  55         perl            \
  55         man             \
  56         Adm             \
  57         abi             \
  58         adbgen          \
  59         acct            \
```



```
  60         acctadm         \
  61         arch            \
  62         asa             \
  63         ast             \
  64         audio           \
  65         auths           \
  66         autopush        \
  67         avs             \
  68         awk             \
  69         awk_xpg4        \
  70         backup          \
  71         banner          \
  72         bart            \
  73         basename        \
  74         bc              \
  75         bdiff           \
  76         beadm           \
  77         bfs             \
  78         bnu             \
  79         boot            \
  80         busstat         \
  81         cal             \
  82         calendar        \
  83         captoinfo       \
  84         cat             \
  85         cdrw            \
  86         cfgadm          \
  87         checkeq         \
  88         checknr         \
  89         chgrp           \
  90         chmod           \
  91         chown           \
  92         chroot          \
  93         clear           \
  94         clinfo          \
  95         cmd-crypto      \
  96         cmd-inet        \
  97         col             \
  98         compress        \
  99         consadm         \
 100         coreadm         \
 101         cpio            \
 102         cpc             \
 103         cron            \
 104         crypt           \
 105         csh             \
 106         csplit          \
 107         ctrun           \
 108         ctstat          \
 109         ctwatch         \
 110         datadm          \
 111         date            \
 112         dc              \
 113         dd              \
 114         deroff          \
 115         devfsadm        \
 116         syseventd       \
 117         devctl          \
 118         devinfo         \
 119         devmgmt         \
 120         devprop         \
 121         dfs.cmds        \
 122         diff            \
 123         diff3           \
 124         diffmk          \
 125         dircmp          \
```

```
126         dirname         \
127         dis             \
128         diskmgtd        \
129         dispadmin       \
130         dladm           \
131         dlstat          \
132         dmesg           \
133         dodatadm        \
134         dtrace          \
135         du              \
136         dumpadm         \
137         dumpcs          \
138         echo            \
139         ed              \
140         eeprom          \
141         egrep           \
142         eject           \
143         emul64ioctl     \
144         enhance         \
145         env             \
146         eqn             \
147         expand          \
148         expr            \
149         exstr           \
150         factor          \
151         false           \
152         fcinfo          \
153         fcoesvc         \
154         fdetach         \
155         fdformat        \
156         fdisk           \
157         filesync        \
158         fgrep           \
159         file            \
160         find            \
161         flowadm         \
162         flowstat        \
163         fm              \
164         fmt             \
165         fmthard         \
166         fmtmsg          \
167         fold            \
168         format          \
169         fs.d            \
170         fstyp           \
171         fuser           \
172         fwflash         \
173         gcore           \
174         gencat          \
175         geniconvtbl     \
176         genmsg          \
177         getconf         \
178         getdevpolicy    \
179         getent          \
180         getfacl         \
181         getmajor        \
182         getopt          \
183         gettext         \
184         gettxt          \
185         grep            \
186         grep_xpg4       \
187         groups          \
188         grpck           \
189         gss             \
190         hal             \
191         halt            \
```



```
192         head            \
193         hostid          \
194         hostname        \
195         hotplug         \
196         hotplugd        \
197         hwdata          \
198         ibd_upgrade     \
199         id              \
200         idmap           \
201         infocmp         \
202         init            \
203         initpkg         \
204         install.d       \
205         intrd           \
206         intrstat        \
207         ipcrm           \
208         ipcs            \
209         ipdadm          \
210         ipf             \
211         isainfo         \
212         isalist         \
213         itutools        \
214         iscsiadm        \
215         iscsid          \
216         iscsitsvc       \
217         isns            \
218         itadm           \
219         java            \
220         kbd             \
221         keyserv         \
222         killall         \
223         krb5            \
224         ksh             \
225         kvmstat         \
226         last            \
227         lastcomm        \
228         latencytop      \
229         ldap            \
230         ldapcachemgr    \
231         lgrpinfo        \
232         line            \
233         link            \
234         dlmgmtd         \
235         listen          \
236         loadkeys        \
237         locale          \
238         localedef       \
239         lockstat        \
240         locator         \
241         lofiadm         \
242         logadm          \
243         logger          \
244         login           \
245         logins          \
246         look            \
247         ls              \
248         luxadm          \
249         lvm             \
250         mach            \
251         machid          \
252         mail            \
253         mailx           \
254         makekey         \
255         man             \
256         mandoc          \
257         mdb             \
```

```
258         mesg            \
259         mkdir           \
260         mkfifo          \
261         mkfile          \
262         mkmsgs          \
263         mknod           \
264         mkpwdict        \
265         mktemp          \
266         modload         \
267         more            \
268         mpathadm        \
269         msgfmt          \
270         msgid           \
271         mt              \
272         mv              \
273         mvdir           \
274         ndmpadm         \
275         ndmpd           \
276         ndmpstat        \
277         netadm          \
278         netfiles        \
279         newform         \
280         newgrp          \
281         news            \
282         newtask         \
283         nice            \
284         nl              \
285         nlsadmin        \
286         nohup           \
287         nsadmin         \
288         nscd            \
289         oamuser         \
290         oawk            \
291         od              \
292         pack            \
293         pagesize        \
294         passmgmt        \
295         passwd          \
296         pathchk         \
297         pbind           \
298         pcidr           \
299         pcitool         \
300         pfexec          \
301         pfexecd         \
302         pginfo          \
303         pgstat          \
304         pgrep           \
305         picl            \
306         plimit          \
307         policykit       \
308         pools           \
309         power           \
310         powertop        \
311         ppgsz           \
312         pg              \
313         plockstat       \
314         pr              \
315         prctl           \
316         print           \
317         printf          \
318         priocntl        \
319         profiles        \
320         projadd         \
321         projects        \
322         prstat          \
323         prtconf         \
```



```
324         prtdiag         \
325         prtvtoc         \
326         ps              \
327         psradm          \
328         psrinfo         \
329         psrset          \
330         ptools          \
331         pwck            \
332         pwconv          \
333         pwd             \
334         pyzfs           \
335         raidctl         \
336         ramdiskadm      \
337         rcap            \
338         rcm_daemon      \
339         rctladm         \
340         refer           \
341         regcmp          \
342         renice          \
343         rexd            \
344         rm              \
345         rmdir           \
346         rmformat        \
347         rmmount         \
348         rmt             \
349         rmvolmgr        \
350         roles           \
351         rpcbind         \
352         rpcgen          \
353         rpcinfo         \
354         rpcsvc          \
355         runat           \
356         sa              \
357         saf             \
358         sasinfo         \
359         savecore        \
360         sbdadm          \
361         script          \
362         scsi            \
363         sdiff           \
364         sdpadm          \
365         sed             \
366         sendmail        \
367         setfacl         \
368         setmnt          \
369         setpgrp         \
370         setuname        \
371         sgs             \
372         sh              \
373         shcomp          \
374         smbios          \
375         smbsrv          \
376         smserverd       \
377         soelim          \
378         sort            \
379         spell           \
380         split           \
381         sqlite          \
382         srchtxt         \
383         srptadm         \
384         srptsvc         \
385         ssh             \
386         stat            \
387         stmfadm         \
388         stmfproxy       \
389         stmfsvc         \
```

```
390         stmsboot        \
391         streams         \
392         strings         \
393         su              \
394         sulogin         \
395         sunpc           \
396         svc             \
397         svr4pkg         \
398         swap            \
399         sync            \
400         sysdef          \
401         syseventadm     \
402         syslogd         \
403         tabs            \
404         tail            \
405         tar             \
406         tbl             \
407         tcopy           \
408         tcpd            \
409         terminfo        \
410         th_tools        \
411         tic             \
412         time            \
413         tip             \
414         tnf             \
415         touch           \
416         tput            \
417         tr              \
418         trapstat        \
419         troff           \
420         true            \
421         truss           \
422         tsol            \
423         tty             \
424         ttymon          \
425         tzreload        \
426         uadmin          \
427         ul              \
428         uname           \
429         units           \
430         unlink          \
431         unpack          \
432         userattr        \
433         users           \
434         utmp_update     \
435         utmpd           \
436         valtools        \
437         vgrind          \
438         vi              \
439         volcheck        \
440         volrmmount      \
441         vrrpadm         \
442         vscan           \
443         vt              \
444         w               \
445         wall            \
446         which           \
447         who             \
448         whodo           \
449         wracct          \
450         write           \
451         wusbadm         \
452         xargs           \
453         xstr            \
454         yes             \
455         ypcmd           \
```



```
456         yppasswd        \
457         zdb             \
458         zdump           \
459         zfs             \
460         zhack           \
461         zic             \
462         zinject         \
463         zlogin          \
464         zoneadm         \
465         zoneadmd        \
466         zonecfg         \
467         zonename        \
468         zpool           \
469         zlook           \
470         zonestat        \
471         zstreamdump     \
472         ztest

474 i386_SUBDIRS=           \
475         acpihpd         \
476         addbadsec       \
477         biosdev         \
478         diskscan        \
479         lms             \
480         ntfsprogs       \
481         parted          \
482         rtc             \
483         ucodeadm        \
484         xvm

486 sparc_SUBDIRS=          \
487         cvcd            \
488         dcs             \
489         device_remap    \
490         drd             \
491         fruadm          \
492         ldmad           \
493         oplhpd          \
494         prtdscp         \
495         prtfru          \
496         scadm           \
497         sckmd           \
498         sf880drd        \
499         virtinfo        \
500         vntsd

502 #
503 # Commands that are messaged.  Note that 'lp' comes first
504 # (see previous comment about 'lp'.)
501 # Commands that are messaged.  Note that 'lp' and 'man' come first
502 # (see previous comment about 'lp' and 'man').
505 #
506 MSGSUBDIRS=             \
507         lp              \
506         man             \
508         abi             \
509         acctadm         \
510         allocate        \
511         asa             \
512         audio           \
513         audit           \
514         auditconfig     \
515         auditd          \
516         auditrecord     \
517         auditset        \
518         auths           \
```

```
519         autopush        \
520         avs             \
521         awk             \
522         awk_xpg4        \
523         backup          \
524         banner          \
525         bart            \
526         basename        \
527         beadm           \
528         bnu             \
529         busstat         \
530         cal             \
531         cat             \
532         cdrw            \
533         cfgadm          \
534         checkeq         \
535         checknr         \
536         chgrp           \
537         chmod           \
538         chown           \
539         cmd-crypto      \
540         cmd-inet        \
541         col             \
542         compress        \
543         consadm         \
544         coreadm         \
545         cpio            \
546         cpc             \
547         cron            \
548         csh             \
549         csplit          \
550         ctrun           \
551         ctstat          \
552         ctwatch         \
553         datadm          \
554         date            \
555         dc              \
556         dcs             \
557         dd              \
558         deroff          \
559         devfsadm        \
560         dfs.cmds        \
561         diff            \
562         diffmk          \
563         dladm           \
564         dlstat          \
565         du              \
566         dumpcs          \
567         ed              \
568         eject           \
569         env             \
570         eqn             \
571         expand          \
572         expr            \
573         fcinfo          \
574         fgrep           \
575         file            \
576         filesync        \
577         find            \
578         flowadm         \
579         flowstat        \
580         fm              \
581         fold            \
582         fs.d            \
583         fwflash         \
584         geniconvtbl     \
```



```
585         genmsg          \
586         getconf         \
587         getent          \
588         gettext         \
589         gettxt          \
590         grep            \
591         grep_xpg4       \
592         grpck           \
593         gss             \
594         halt            \
595         head            \
596         hostname        \
597         hotplug         \
598         id              \
599         idmap           \
600         isaexec         \
601         iscsiadm        \
602         iscsid          \
603         isns            \
604         itadm           \
605         kbd             \
606         krb5            \
607         ksh             \
608         last            \
609         ldap            \
610         ldapcachemgr    \
611         lgrpinfo        \
612         locale          \
613         lofiadm         \
614         logadm          \
615         logger          \
616         logins          \
617         ls              \
618         luxadm          \
619         lvm             \
620         mailx           \
621         man             \
622         mesg            \
623         mkdir           \
624         mkpwdict        \
625         mktemp          \
626         more            \
627         mpathadm        \
628         msgfmt          \
629         mv              \
630         ndmpadm         \
631         ndmpstat        \
632         newgrp          \
633         newtask         \
634         nice            \
635         nohup           \
636         oawk            \
637         pack            \
638         passwd          \
639         passmgmt        \
640         pathchk         \
641         pfexec          \
642         pg              \
643         pgrep           \
644         picl            \
645         pools           \
646         power           \
647         pr              \
648         praudit         \
649         print           \
650         profiles        \
```

```
651         projadd         \
652         projects        \
653         prstat          \
654         prtdiag         \
655         ps              \
656         psrinfo         \
657         ptools          \
658         pwconv          \
659         pwd             \
660         pyzfs           \
661         raidctl         \
662         ramdiskadm      \
663         rcap            \
664         rcm_daemon      \
665         refer           \
666         regcmp          \
667         renice          \
668         roles           \
669         rm              \
670         rmdir           \
671         rmformat        \
672         rmmount         \
673         rmvolmgr        \
674         sasinfo         \
675         sbdadm          \
676         scadm           \
677         script          \
678         scsi            \
679         sdiff           \
680         sdpadm          \
681         sgs             \
682         sh              \
683         shcomp          \
684         smbsrv          \
685         sort            \
686         split           \
687         srptadm         \
688         ssh             \
689         stat            \
690         stmfadm         \
691         stmsboot        \
692         strings         \
693         su              \
694         svc             \
695         svr4pkg         \
696         swap            \
697         syseventadm     \
698         syseventd       \
699         tabs            \
700         tar             \
701         tbl             \
702         time            \
703         tnf             \
704         touch           \
705         tput            \
706         troff           \
707         tsol            \
708         tty             \
709         ttymon          \
710         tzreload        \
711         ul              \
712         uname           \
713         units           \
714         unlink          \
715         unpack          \
716         userattr        \
```



```
717         valtools        \
718         vgrind          \
719         vi              \
720         volcheck        \
721         volrmmount      \
722         vrrpadm         \
723         vscan           \
724         w               \
725         who             \
726         whodo           \
727         wracct          \
728         write           \
729         wusbadm         \
730         xargs           \
731         yppasswd        \
732         zdump           \
733         zfs             \
734         zic             \
735         zlogin          \
736         zoneadm         \
737         zoneadmd        \
738         zonecfg         \
739         zonename        \
740         zpool           \
741         zonestat

743 sparc_MSGSUBDIRS=       \
744         fruadm          \
745         prtdscp         \
746         prtfru          \
747         virtinfo        \
748         vntsd

750 i386_MSGSUBDIRS=        \
751         ucodeadm

753 #
754 # commands that use dcgettext for localized time, LC_TIME
755 #
756 DCSUBDIRS=              \
757         cal             \
758         cfgadm          \
759         diff            \
760         ls              \
761         pr              \
762         ps              \
763         tar             \
764         w               \
765         who             \
766         whodo           \
767         write

769 #
770 # commands that belong only to audit.
771 #
772 AUDITSUBDIRS=           \
773         amt             \
774         audit           \
775         audit_warn      \
776         auditconfig     \
777         auditd          \
778         auditrecord     \
779         auditreduce     \
780         auditset        \
781         auditstat       \
782         praudit
```

```
784 #
785 # commands not owned by the systems group
786 #
787 BWOSDIRS=


790 all :=          TARGET = all
791 install :=      TARGET = install
792 clean :=        TARGET = clean
793 clobber :=      TARGET = clobber
794 lint :=         TARGET = lint
795 _msg :=         TARGET = _msg
796 _dc :=          TARGET = _dc

798 .KEEP_STATE:

800 SUBDIRS = $(COMMON_SUBDIRS) $($(MACH)_SUBDIRS)

802 .PARALLEL:      $(BWOSDIRS) $(SUBDIRS) $(MSGSUBDIRS) $(AUDITSUBDIRS)

804 all install clean clobber lint: $(FIRST_SUBDIRS) .WAIT $(SUBDIRS) \
805         $(AUDITSUBDIRS)

807 #
808 # Manifests cannot be checked in parallel, because we are using
809 # the global repository that is in $(SRC)/cmd/svc/seed/global.db.
810 # For this reason, to avoid .PARALLEL and .NO_PARALLEL conflicts,
811 # we spawn off a sub-make to perform the non-parallel 'make check'
812 #
813 check:
814         $(MAKE) -f Makefile.check check

816 #
817 # The .WAIT directive works around an apparent bug in parallel make.
818 # Evidently make was getting the target _msg vs. _dc confused under
819 # some level of parallelization, causing some of the _dc objects
820 # not to be built.
821 #
822 _msg: $(MSGSUBDIRS) $($(MACH)_MSGSUBDIRS) .WAIT _dc

824 _dc: $(DCSUBDIRS)

826 #
827 # Dependencies
828 #
829 fs.d: fstyp
830 ksh:    shcomp isaexec
831 mdb:    terminfo
832 print:  lp

834 $(FIRST_SUBDIRS) $(BWOSDIRS) $(SUBDIRS) $(AUDITSUBDIRS): FRC
835         @if [ -f $@/Makefile  ]; then \
836                 cd $@; pwd; $(MAKE) $(TARGET); \
837         else \
838                 true; \
839         fi

841 FRC:
```

```
**********************************************************
     993 Wed Jul 16 14:05:06 2014
new/usr/src/cmd/man/Makefile
Add catman, makewhatis functionality.  Print an error if the whatis database
is missing.
mandoc import
**********************************************************
```

```
   1 #
   2 # This file and its contents are supplied under the terms of the
   3 # Common Development and Distribution License ("CDDL"), version 1.0.
   4 # You may only use this file in accordance with the terms of version
   5 # 1.0 of the CDDL.
   2 # CDDL HEADER START
   6 #
   7 # A full copy of the text of the CDDL should have accompanied this
   8 # source.  A copy of the CDDL is also available via the Internet at
   9 # http://www.illumos.org/license/CDDL.
   4 # The contents of this file are subject to the terms of the
   5 # Common Development and Distribution License, Version 1.0 only
   6 # (the "License").  You may not use this file except in compliance
   7 # with the License.
  10 #

   9 # You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
  10 # or http://www.opensolaris.org/os/licensing.
  11 # See the License for the specific language governing permissions
  12 # and limitations under the License.
  12 #
  13 # Copyright 2012 Nexenta Systems, Inc. All rights reserved.
  14 # When distributing Covered Code, include this CDDL HEADER in each
  15 # file and include the License file at usr/src/OPENSOLARIS.LICENSE.
  16 # If applicable, add the following below this CDDL HEADER, with the
  17 # fields enclosed by brackets "[]" replaced with your own identifying
  18 # information: Portions Copyright [yyyy] [name of copyright owner]
  14 #
  20 # CDDL HEADER END
  21 #
  22 #
  23 #ident  "%Z%%M% %I%     %E% SMI"
  24 #
  25 # Copyright (c) 1990 by Sun Microsystems, Inc.
  26 #
  27 # cmd/man/Makefile

  16 PROG=           man
  17 LINKS=          apropos whatis catman
  18 LIBLINKS =      makewhatis
  19 OBJS=           makewhatis.o man.o stringlist.o
  20 SRCS=           $(OBJS:%.o=%.c)
  29 include ../Makefile.cmd
  30 SUBDIRS = src

  22 include         $(SRCS)/cmd/Makefile.cmd
  32 all :=          TARGET= all
  33 install :=      TARGET= install
  34 clean :=        TARGET= clean
  35 clobber :=      TARGET= clobber
  36 lint :=         TARGET= lint
  37 _msg :=         TARGET= catalog

  24 CFLAGS +=       $(CCVERBOSE)
  39 #for message catalog files
  40 POFILE = man.po
  41 POFILES = src/src.po

  26 ROOTLINKS=      $(LINKS:%=$(ROOTBIN)/%) $(LIBLINKS:%=$(ROOTLIB)/%)
```



```
  43 .KEEP_STATE:

  28 .KEEP_STATE :
  45 all install clean lint: $(SUBDIRS)

  30 all:            $(PROG)
  47 clobber: $(SUBDIRS)     local_clobber

  32 clean:
  33                 $(RM) $(OBJS)
  49 local_clobber:
  50         $(RM)   $(CLOBBERFILES)

  35 install:        all $(ROOTPROG) $(ROOTLINKS)
  52 _msg:   $(SUBDIRS)
  53         $(RM)   $(POFILE)
  54         cat $(POFILES)      > $(POFILE)
  55         $(RM)  $(MSGDOMAIN)/$(POFILE)
  56         cp $(POFILE) $(MSGDOMAIN)

  37 lint:           lint_SRCS
  58 $(SUBDIRS): FRC
  59         @cd $@; pwd; $(MAKE) $(TARGET)

  39 $(PROG):        $(OBJS)
  40                 $(LINK.c) $(OBJS) -o $@ $(LDLIBS)
  41                 $(POST_PROCESS)

  43 $(ROOTLINKS):   $(ROOTPROG)
  44                 $(RM) $@; $(LN) $(ROOTPROG) $@

  46 include         $(SRC)/cmd/Makefile.targ
  61 FRC:
```

```
**********************************************************
    4680 Wed Jul 16 14:05:06 2014
new/usr/src/cmd/man/THIRDPARTYLICENSE
mandoc import
**********************************************************
```

man.c:

makewhatis.c:

stringlist.c, stringlist.h:

```
**********************************************************
   18043 Wed Jul 16 14:05:07 2014
new/usr/src/cmd/man/makewhatis.c
mandoc import
**********************************************************
```

```
/*
 * Copyright (c) 2002 John Rochester
 * All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer,
 *    in this position and unchanged.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 * 3. The name of the author may not be used to endorse or promote products
 *    derived from this software without specific prior written permission
 *
 * THIS SOFTWARE IS PROVIDED BY THE AUTHOR ''AS IS'' AND ANY EXPRESS OR
 * IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES
 * OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED.
 * IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT,
 * INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT
 * NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
 * DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
 * THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
 * (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF
 * THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 */

/*
 * Copyright 2012 Nexenta Systems, Inc. All rights reserved.
 */

#include <sys/types.h>
#include <sys/stat.h>
#include <sys/param.h>

#include <ctype.h>
#include <dirent.h>
#include <err.h>
#include <signal.h>
#include <stddef.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>

#include "man.h"
#include "stringlist.h"


/* Information collected about each man page in a section */
struct page_info {
        char    *filename;
        char    *name;
        char    *suffix;
        ino_t   inode;
};

/* An expanding string */
struct sbuf {
        char    *content;       /* the start of the buffer */
        char    *end;           /* just past the end of the content */
        char    *last;          /* the last allocated character */
};

/* Remove the last amount characters from the sbuf */
#define sbuf_retract(sbuf, amount) ((sbuf)->end -= (amount))
/* Return the length of the sbuf content */
#define sbuf_length(sbuf) ((sbuf)->end - (sbuf)->content)

typedef char *edited_copy(char *from, char *to, int length);

/*
 * While the whatis line is being formed, it is stored in whatis_proto.
 * When finished, it is reformatted into whatis_final and then appended
 * to whatis_lines.
 */
static struct sbuf      *whatis_proto;
static struct sbuf      *whatis_final;
static stringlist       *whatis_lines;  /* collected output lines */

static char tempfile[MAXPATHLEN];       /* path of temporary file, if any */

#define MDOC_COMMANDS   "ArDvErEvFlLiNmPa"


/* Free a struct page_info and its content */
static void
free_page_info(struct page_info *info)
{

        free(info->filename);
        free(info->name);
        free(info->suffix);
        free(info);
}

/*
 * Allocate and fill in a new struct page_info given the
 * name of the man section directory and the dirent of the file.
 * If the file is not a man page, return NULL.
 */
static struct page_info *
new_page_info(char *dir, struct dirent *dirent)
{
        struct page_info *info;
        int             basename_length;
        char            *suffix;
        struct stat     st;

        if ((info = malloc(sizeof (struct page_info))) == NULL)
                err(1, "malloc");
        basename_length = strlen(dirent->d_name);
        suffix = &dirent->d_name[basename_length];
        if (asprintf(&info->filename, "%s/%s", dir, dirent->d_name) == -1)
                err(1, "asprintf");
        for (;;) {
                if (--suffix == dirent->d_name || !isalnum(*suffix)) {
                        if (*suffix == '.')
                                break;
                        free(info->filename);
                        free(info);
                        return (NULL);
                }
        }
        *suffix++ = '\0';
        info->name = strdup(dirent->d_name);
```

```
128         info->suffix = strdup(suffix);
129         if (stat(info->filename, &st) < 0) {
130                 warn("%s", info->filename);
131                 free_page_info(info);
132                 return (NULL);
133         }
134         if (!S_ISREG(st.st_mode)) {
135                 free_page_info(info);
136                 return (NULL);
137         }
138         info->inode = st.st_ino;
139         return (info);
140 }

142 /*
143  * Reset sbuf length to 0.
144  */
145 static void
146 sbuf_clear(struct sbuf *sbuf)
147 {

149         sbuf->end = sbuf->content;
150 }

152 /*
153  * Allocate a new sbuf.
154  */
155 static struct sbuf *
156 new_sbuf(void)
157 {
158         struct sbuf     *sbuf;

160         if ((sbuf = malloc(sizeof (struct sbuf))) == NULL)
161                 err(1, "malloc");
162         if ((sbuf->content = (char *)malloc(LINE_ALLOC)) == NULL)
163                 err(1, "malloc");
164         sbuf->last = sbuf->content + LINE_ALLOC - 1;
165         sbuf_clear(sbuf);

167         return (sbuf);
168 }

170 /*
171  * Ensure that there is enough room in the sbuf
172  * for nchars more characters.
173  */
174 static void
175 sbuf_need(struct sbuf *sbuf, int nchars)
176 {
177         char *new_content;
178         size_t size, cntsize;

180         /* Double the size of the allocation until the buffer is big enough */
181         while (sbuf->end + nchars > sbuf->last) {
182                 size = sbuf->last + 1 - sbuf->content;
183                 size *= 2;
184                 cntsize = sbuf->end - sbuf->content;

186                 new_content = (char *)malloc(size);
187                 (void) memcpy(new_content, sbuf->content, cntsize);
188                 free(sbuf->content);
189                 sbuf->content = new_content;
190                 sbuf->end = new_content + cntsize;
191                 sbuf->last = new_content + size - 1;
192         }
193 }
```



```
195 /*
196  * Append a string of a given length to the sbuf.
197  */
198 static void
199 sbuf_append(struct sbuf *sbuf, const char *text, int length)
200 {
201         if (length > 0) {
202                 sbuf_need(sbuf, length);
203                 (void) memcpy(sbuf->end, text, length);
204                 sbuf->end += length;
205         }
206 }

208 /*
209  * Append a null-terminated string to the sbuf.
210  */
211 static void
212 sbuf_append_str(struct sbuf *sbuf, char *text)
213 {

215         sbuf_append(sbuf, text, strlen(text));
216 }

218 /*
219  * Append an edited null-terminated string to the sbuf.
220  */
221 static void
222 sbuf_append_edited(struct sbuf *sbuf, char *text, edited_copy copy)
223 {
224         int     length;

226         if ((length = strlen(text)) > 0) {
227                 sbuf_need(sbuf, length);
228                 sbuf->end = copy(text, sbuf->end, length);
229         }
230 }

232 /*
233  * Strip any of a set of chars from the end of the sbuf.
234  */
235 static void
236 sbuf_strip(struct sbuf *sbuf, const char *set)
237 {

239         while (sbuf->end > sbuf->content && strchr(set, sbuf->end[-1]) != NULL)
240                 sbuf->end--;
241 }

243 /*
244  * Return the null-terminated string built by the sbuf.
245  */
246 static char *
247 sbuf_content(struct sbuf *sbuf)
248 {

250         *sbuf->end = '\0';
251         return (sbuf->content);
252 }

254 /*
255  * Return true if no man page exists in the directory with
256  * any of the names in the stringlist.
257  */
258 static int
259 no_page_exists(char *dir, stringlist *names, char *suffix)
```

```
{
        char    path[MAXPATHLEN];
        size_t  i;

        for (i = 0; i < names->sl_cur; i++) {
                (void) snprintf(path, MAXPATHLEN, "%s/%s.%s.gz",
                    dir, names->sl_str[i], suffix);
                if (access(path, F_OK) < 0) {
                        path[strlen(path) - 3] = '\0';
                        if (access(path, F_OK) < 0)
                                continue;
                }
                return (0);
        }
        return (1);
}

/* ARGSUSED sig */
static void
trap_signal(int sig)
{

        if (tempfile[0] != '\0')
                (void) unlink(tempfile);

        exit(1);
}

/*
 * Attempt to open an output file.
 * Return NULL if unsuccessful.
 */
static FILE *
open_output(char *name)
{
        FILE    *output;

        whatis_lines = sl_init();
        (void) snprintf(tempfile, MAXPATHLEN, "%s.tmp", name);
        name = tempfile;
        if ((output = fopen(name, "w")) == NULL) {
                warn("%s", name);
                return (NULL);
        }
        return (output);
}

static int
linesort(const void *a, const void *b)
{

        return (strcmp((*(const char * const *)a), (*(const char * const *)b)));
}

/*
 * Write the unique sorted lines to the output file.
 */
static void
finish_output(FILE *output, char *name)
{
        size_t  i;
        char    *prev = NULL;

        qsort(whatis_lines->sl_str, whatis_lines->sl_cur, sizeof (char *),
            linesort);
        for (i = 0; i < whatis_lines->sl_cur; i++) {
```



```
                char *line = whatis_lines->sl_str[i];
                if (i > 0 && strcmp(line, prev) == 0)
                        continue;
                prev = line;
                (void) fputs(line, output);
                (void) putc('\n', output);
        }
        (void) fclose(output);
        sl_free(whatis_lines, 1);
        (void) rename(tempfile, name);
        (void) unlink(tempfile);
}

static FILE *
open_whatis(char *mandir)
{
        char    filename[MAXPATHLEN];

        (void) snprintf(filename, MAXPATHLEN, "%s/%s", mandir, WHATIS);
        return (open_output(filename));
}

static void
finish_whatis(FILE *output, char *mandir)
{
        char    filename[MAXPATHLEN];

        (void) snprintf(filename, MAXPATHLEN, "%s/%s", mandir, WHATIS);
        finish_output(output, filename);
}

/*
 * Remove trailing spaces from a string, returning a pointer to just
 * beyond the new last character.
 */
static char *
trim_rhs(char *str)
{
        char    *rhs;

        rhs = &str[strlen(str)];
        while (--rhs > str && isspace(*rhs))
                ;
        *++rhs = '\0';
        return (rhs);
}

/*
 * Return a pointer to the next non-space character in the string.
 */
static char *
skip_spaces(char *s)
{

        while (*s != '\0' && isspace(*s))
                s++;

        return (s);
}

/*
 * Return whether the line is of one of the forms:
 *      .Sh NAME
 *      .Sh "NAME"
 *      etc.
 * assuming that section_start is ".Sh".
```

```
 */
static int
name_section_line(char *line, const char *section_start)
{
        char            *rhs;

        if (strncmp(line, section_start, 3) != 0)
                return (0);
        line = skip_spaces(line + 3);
        rhs = trim_rhs(line);
        if (*line == '"') {
                line++;
                if (*--rhs == '"')
                        *rhs = '\0';
        }
        if (strcmp(line, "NAME") == 0)
                return (1);

        return (0);
}

/*
 * Copy characters while removing the most common nroff/troff markup:
 *      \(em, \(mi, \s[+-N], \&
 *      \fF, \f(fo, \f[font]
 *      \*s, \*(st, \*[stringvar]
 */
static char *
de_nroff_copy(char *from, char *to, int fromlen)
{
        char    *from_end = &from[fromlen];

        while (from < from_end) {
                switch (*from) {
                case '\\':
                        switch (*++from) {
                        case '(':
                                if (strncmp(&from[1], "em", 2) == 0 ||
                                    strncmp(&from[1], "mi", 2) == 0) {
                                        from += 3;
                                        continue;
                                }
                                break;
                        case 's':
                                if (*++from == '-')
                                        from++;
                                while (isdigit(*from))
                                        from++;
                                continue;
                        case 'f':
                        case '*':
                                if (*++from == '(') {
                                        from += 3;
                                } else if (*from == '[') {
                                        while (*++from != ']' &&
                                            from < from_end)
                                                ;
                                        from++;
                                } else {
                                        from++;
                                }
                                continue;
                        case '&':
                                from++;
                                continue;
                        }
```



```
                        break;
                }
                *to++ = *from++;
        }
        return (to);
}

/*
 * Append a string with the nroff formatting removed.
 */
static void
add_nroff(char *text)
{

        sbuf_append_edited(whatis_proto, text, de_nroff_copy);
}

/*
 * Appends "name(suffix), " to whatis_final
 */
static void
add_whatis_name(char *name, char *suffix)
{

        if (*name != '\0') {
                sbuf_append_str(whatis_final, name);
                sbuf_append(whatis_final, "(", 1);
                sbuf_append_str(whatis_final, suffix);
                sbuf_append(whatis_final, "), ", 3);
        }
}

/*
 * Processes an old-style man(7) line. This ignores commands with only
 * a single number argument.
 */
static void
process_man_line(char *line)
{
        char    *p;

        if (*line == '.') {
                while (isalpha(*++line))
                        ;
                p = line = skip_spaces(line);
                while (*p != '\0') {
                        if (!isdigit(*p))
                                break;
                        p++;
                }
                if (*p == '\0')
                        return;
        } else
                line = skip_spaces(line);
        if (*line != '\0') {
                add_nroff(line);
                sbuf_append(whatis_proto, " ", 1);
        }
}

/*
 * Processes a new-style mdoc(7) line.
 */
static void
process_mdoc_line(char *line)
{
```

```
524         int     xref;
525         int     arg = 0;
526         char    *line_end = &line[strlen(line)];
527         int     orig_length = sbuf_length(whatis_proto);
528         char    *next;

530         if (*line == '\0')
531                 return;
532         if (line[0] != '.' || !isupper(line[1]) || !islower(line[2])) {
533                 add_nroff(skip_spaces(line));
534                 sbuf_append(whatis_proto, " ", 1);
535                 return;
536         }
537         xref = strncmp(line, ".Xr", 3) == 0;
538         line += 3;
539         while ((line = skip_spaces(line)) < line_end) {
540                 if (*line == '"') {
541                         next = ++line;
542                         for (;;) {
543                                 next = strchr(next, '"');
544                                 if (next == NULL)
545                                         break;
546                                 (void) memmove(next, next + 1, strlen(next));
547                                 line_end--;
548                                 if (*next != '"')
549                                         break;
550                                 next++;
551                         }
552                 } else {
553                         next = strpbrk(line, " \t");
554                 }
555                 if (next != NULL)
556                         *next++ = '\0';
557                 else
558                         next = line_end;
559                 if (isupper(*line) && islower(line[1]) && line[2] == '\0') {
560                         if (strcmp(line, "Ns") == 0) {
561                                 arg = 0;
562                                 line = next;
563                                 continue;
564                         }
565                         if (strstr(line, MDOC_COMMANDS) != NULL) {
566                                 line = next;
567                                 continue;
568                         }
569                 }
570                 if (arg > 0 && strchr(",.:;?!)]", *line) == 0) {
571                         if (xref) {
572                                 sbuf_append(whatis_proto, "(", 1);
573                                 add_nroff(line);
574                                 sbuf_append(whatis_proto, ")", 1);
575                                 xref = 0;
576                         } else {
577                                 sbuf_append(whatis_proto, " ", 1);
578                         }
579                 }
580                 add_nroff(line);
581                 arg++;
582                 line = next;
583         }
584         if (sbuf_length(whatis_proto) > orig_length)
585                 sbuf_append(whatis_proto, " ", 1);
586 }

588 /*
589  * Collect a list of comma-separated names from the text.
```



```
590  */
591 static void
592 collect_names(stringlist *names, char *text)
593 {
594         char    *arg;

596         for (;;) {
597                 arg = text;
598                 text = strchr(text, ',');
599                 if (text != NULL)
600                         *text++ = '\0';
601                 (void) sl_add(names, arg);
602                 if (text == NULL)
603                         return;
604                 if (*text == ' ')
605                         text++;
606         }
607 }

609 enum { STATE_UNKNOWN, STATE_MANSTYLE, STATE_MDOCNAME, STATE_MDOCDESC };

611 /*
612  * Process a man page source into a single whatis line and add it
613  * to whatis_lines.
614  */
615 static void
616 process_page(struct page_info *page, char *section_dir)
617 {
618         FILE            *fp;
619         stringlist      *names;
620         char            *descr;
621         int             state = STATE_UNKNOWN;
622         size_t          i;
623         char            *line = NULL;
624         size_t          linecap = 0;

626         sbuf_clear(whatis_proto);
627         if ((fp = fopen(page->filename, "r")) == NULL) {
628                 warn("%s", page->filename);
629                 return;
630         }
631         while (getline(&line, &linecap, fp) > 0) {
632                 /* Skip comments */
633                 if (strncmp(line, ".\\\"", 3) == 0)
634                         continue;
635                 switch (state) {
636                 /* Haven't reached the NAME section yet */
637                 case STATE_UNKNOWN:
638                         if (name_section_line(line, ".SH"))
639                                 state = STATE_MANSTYLE;
640                         else if (name_section_line(line, ".Sh"))
641                                 state = STATE_MDOCNAME;
642                         continue;
643                 /* Inside an old-style .SH NAME section */
644                 case STATE_MANSTYLE:
645                         if (strncmp(line, ".SH", 3) == 0 ||
646                             strncmp(line, ".SS", 3) == 0)
647                                 break;
648                         (void) trim_rhs(line);
649                         if (strcmp(line, ".") == 0)
650                                 continue;
651                         if (strncmp(line, ".IX", 3) == 0) {
652                                 line += 3;
653                                 line = skip_spaces(line);
654                         }
655                         process_man_line(line);
```

```
656                     continue;
657             /* Inside a new-style .Sh NAME section (the .Nm part) */
658             case STATE_MDOCNAME:
659                     (void) trim_rhs(line);
660                     if (strncmp(line, ".Nm", 3) == 0) {
661                             process_mdoc_line(line);
662                             continue;
663                     } else {
664                             if (strcmp(line, ".") == 0)
665                                     continue;
666                             sbuf_append(whatis_proto, "- ", 2);
667                             state = STATE_MDOCDESC;
668                     }
669                     /* FALLTHROUGH */
670             /* Inside a new-style .Sh NAME section (after the .Nm-s) */
671             case STATE_MDOCDESC:
672                     if (strncmp(line, ".Sh", 3) == 0)
673                             break;
674                     (void) trim_rhs(line);
675                     if (strcmp(line, ".") == 0)
676                             continue;
677                     process_mdoc_line(line);
678                     continue;
679             }
680             break;
681     }
682     (void) fclose(fp);
683     sbuf_strip(whatis_proto, " \t.-");
684     line = sbuf_content(whatis_proto);
685     /*
686      * Line now contains the appropriate data, but without the
687      * proper indentation or the section appended to each name.
688      */
689     descr = strstr(line, " - ");
690     if (descr == NULL) {
691             descr = strchr(line, ' ');
692             if (descr == NULL)
693                     return;
694             *descr++ = '\0';
695     } else {
696             *descr = '\0';
697             descr += 3;
698     }
699     names = sl_init();
700     collect_names(names, line);
701     sbuf_clear(whatis_final);
702     if (!sl_find(names, page->name) &&
703         no_page_exists(section_dir, names, page->suffix)) {
704             /*
705              * Add the page name since that's the only
706              * thing that man(1) will find.
707              */
708             add_whatis_name(page->name, page->suffix);
709     }
710     for (i = 0; i < names->sl_cur; i++)
711             add_whatis_name(names->sl_str[i], page->suffix);
712     sl_free(names, 0);
713     /* Remove last ", " */
714     sbuf_retract(whatis_final, 2);
715     while (sbuf_length(whatis_final) < INDENT)
716             sbuf_append(whatis_final, " ", 1);
717     sbuf_append(whatis_final, " - ", 3);
718     sbuf_append_str(whatis_final, skip_spaces(descr));
719     (void) sl_add(whatis_lines, strdup(sbuf_content(whatis_final)));
720 }
```



```
722 /*
723  * Sort pages first by inode number, then by name.
724  */
725 static int
726 pagesort(const void *a, const void *b)
727 {
728         const struct page_info *p1 = *(struct page_info * const *) a;
729         const struct page_info *p2 = *(struct page_info * const *) b;
730
731         if (p1->inode == p2->inode)
732                 return (strcmp(p1->name, p2->name));
733
734         return (p1->inode - p2->inode);
735 }
736
737 /*
738  * Process a single man section.
739  */
740 static void
741 process_section(char *section_dir)
742 {
743         struct dirent   **entries;
744         int             nentries;
745         struct page_info **pages;
746         int             npages = 0;
747         int             i;
748         ino_t           prev_inode = 0;
749
750         /* Scan the man section directory for pages */
751         nentries = scandir(section_dir, &entries, NULL, alphasort);
752
753         /* Collect information about man pages */
754         pages = (struct page_info **)calloc(nentries,
755             sizeof (struct page_info *));
756         for (i = 0; i < nentries; i++) {
757                 struct page_info *info = new_page_info(section_dir, entries[i]);
758                 if (info != NULL)
759                         pages[npages++] = info;
760                 free(entries[i]);
761         }
762         free(entries);
763         qsort(pages, npages, sizeof (struct page_info *), pagesort);
764
765         /* Process each unique page */
766         for (i = 0; i < npages; i++) {
767                 struct page_info *page = pages[i];
768                 if (page->inode != prev_inode) {
769                         prev_inode = page->inode;
770                         process_page(page, section_dir);
771                 }
772                 free_page_info(page);
773         }
774         free(pages);
775 }
776
777 /*
778  * Return whether the directory entry is a man page section.
779  */
780 static int
781 select_sections(const struct dirent *entry)
782 {
783         const char      *p = &entry->d_name[3];
784
785         if (strncmp(entry->d_name, "man", 3) != 0)
786                 return (0);
787         while (*p != '\0') {
```

```
788                 if (!isalnum(*p++))
789                         return (0);
790         }
791         return (1);
792 }

794 /*
795  * Process a single top-level man directory by finding all the
796  * sub-directories named man* and processing each one in turn.
797  */
798 void
799 mwpath(char *path)
800 {
801         FILE            *fp = NULL;
802         struct dirent   **entries;
803         int             nsections;
804         int             i;

806         (void) signal(SIGINT, trap_signal);
807         (void) signal(SIGHUP, trap_signal);
808         (void) signal(SIGQUIT, trap_signal);
809         (void) signal(SIGTERM, trap_signal);

811         whatis_proto = new_sbuf();
812         whatis_final = new_sbuf();

814         nsections = scandir(path, &entries, select_sections, alphasort);
815         if ((fp = open_whatis(path)) == NULL)
816                 return;
817         for (i = 0; i < nsections; i++) {
818                 char    section_dir[MAXPATHLEN];

820                 (void) snprintf(section_dir, MAXPATHLEN, "%s/%s",
821                     path, entries[i]->d_name);
822                 process_section(section_dir);
823                 free(entries[i]);
824         }
825         free(entries);
826         finish_whatis(fp, path);
827 }
```

```
**********************************************************
   33672 Wed Jul 16 14:05:07 2014
new/usr/src/cmd/man/man.c
Add catman, makewhatis functionality.  Print an error if the whatis database
is missing.
mandoc import
**********************************************************
```

```
/*
 * CDDL HEADER START
 *
 * The contents of this file are subject to the terms of the
 * Common Development and Distribution License (the "License").
 * You may not use this file except in compliance with the License.
 *
 * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
 * or http://www.opensolaris.org/os/licensing.
 * See the License for the specific language governing permissions
 * and limitations under the License.
 *
 * When distributing Covered Code, include this CDDL HEADER in each
 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
 * If applicable, add the following below this CDDL HEADER, with the
 * fields enclosed by brackets "[]" replaced with your own identifying
 * information: Portions Copyright [yyyy] [name of copyright owner]
 *
 * CDDL HEADER END
 */

/*
 * Copyright (c) 1990, 2010, Oracle and/or its affiliates. All rights reserved.
 * Copyright 2012, Josef 'Jeff' Sipek <jeffpc@31bits.net>. All rights reserved.
 * Copyright 2013 Nexenta Systems, Inc. All rights reserved.
 */

/*      Copyright (c) 1983, 1984, 1985, 1986, 1987, 1988, 1989  AT&T.   */
/*              All rights reserved.                                    */

/*
 * University Copyright- Copyright (c) 1982, 1986, 1988
 * The Regents of the University of California
 * All Rights Reserved
 *
 * University Acknowledgment- Portions of this document are derived from
 * software developed by the University of California, Berkeley, and its
 * contributors.
 */

/*
 * Find and display reference manual pages. This version includes makewhatis
 * functionality as well.
 */

#include <sys/param.h>
#include <sys/stat.h>
#include <sys/termios.h>
#include <sys/types.h>

#include <ctype.h>
#include <dirent.h>
#include <err.h>
#include <errno.h>
#include <fcntl.h>
#include <fnmatch.h>
#include <limits.h>
#include <locale.h>
#include <malloc.h>
```



```
#include <memory.h>
#include <regex.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>

#include "man.h"


/* Mapping of old directories to new directories */
static const struct map_entry {
        char    *old_name;
        char    *new_name;
} map[] = {
        { "3b",         "3ucb"          },
        { "3e",         "3elf"          },
        { "3g",         "3gen"          },
        { "3k",         "3kstat"        },
        { "3n",         "3socket"       },
        { "3r",         "3rt"           },
        { "3s",         "3c"            },
        { "3t",         "3thr"          },
        { "3x",         "3curses"       },
        { "3xc",        "3xcurses"      },
        { "3xn",        "3xnet"         }
};

struct suffix {
        char *ds;
        char *fs;
};

/*
 * Flags that control behavior of build_manpath()
 *
 *   BMP_ISPATH                 pathv is a vector constructed from PATH.
 *                              Perform appropriate path translations for
 *                              manpath.
 *   BMP_APPEND_DEFMANDIR       Add DEFMANDIR to the end if it hasn't
 *                              already appeared earlier.
 *   BMP_FALLBACK_DEFMANDIR     Append /usr/share/man only if no other
 *                              manpath (including derived from PATH)
 *                              elements are valid.
 */
#define BMP_ISPATH              1
#define BMP_APPEND_DEFMANDIR    2
#define BMP_FALLBACK_DEFMANDIR  4

/*
 * When doing equality comparisons of directories, device and inode
 * comparisons are done.  The secnode and dupnode structures are used
 * to form a list of lists for this processing.
 */
struct secnode {
        char            *secp;
        struct secnode  *next;
};
struct dupnode {
        dev_t           dev;    /* from struct stat st_dev */
        ino_t           ino;    /* from struct stat st_ino */
        struct secnode  *secl;  /* sections already considered */
        struct dupnode  *next;
};

/*
```

```
 * Map directories that may appear in PATH to the corresponding
 * man directory.
 */
static struct pathmap {
        char    *bindir;
        char    *mandir;
        dev_t   dev;
        ino_t   ino;
} bintoman[] = {
        { "/sbin",              "/usr/share/man,1m",            0, 0 },
        { "/usr/sbin",          "/usr/share/man,1m",            0, 0 },
        { "/usr/ucb",           "/usr/share/man,1b",            0, 0 },
        { "/usr/bin",           "/usr/share/man,1,1m,1s,1t,1c", 0, 0 },
        { "/usr/xpg4/bin",      "/usr/share/man,1",             0, 0 },
        { "/usr/xpg6/bin",      "/usr/share/man,1",             0, 0 },
        { NULL,                 NULL,                           0, 0 }
};

struct man_node {
        char            *path;          /* mandir path */
        char            **secv;         /* submandir suffices */
        int             defsrch;        /* hint for man -p */
        int             frompath;       /* hint for man -d */
        struct man_node *next;
};

static int      all = 0;
static int      apropos = 0;
static int      debug = 0;
static int      found = 0;
static int      list = 0;
static int      makewhatis = 0;
static int      printmp = 0;
static int      sargs = 0;
static int      psoutput = 0;
static int      whatis = 0;
static int      makewhatishere = 0;

static char     *mansec;
static char     *pager;

static char     *addlocale(char *);
static struct man_node *build_manpath(char **, int);
static void     do_makewhatis(struct man_node *);
static char     *check_config(char *);
static int      cmp(const void *, const void *);
static int      dupcheck(struct man_node *, struct dupnode **);
static int      format(char *, char *, char *, char *);
static void     free_dupnode(struct dupnode *);
static void     free_manp(struct man_node *manp);
static void     freev(char **);
static void     fullpaths(struct man_node **);
static void     get_all_sect(struct man_node *);
static int      getdirs(char *, char ***, int);
static void     getpath(struct man_node *, char **);
static void     getsect(struct man_node *, char **);
static void     init_bintoman(void);
static void     lower(char *);
static void     mandir(char **, char *, char *, int);
static int      manual(struct man_node *, char *);
static char     *map_section(char *, char *);
static char     *path_to_manpath(char *);
static void     print_manpath(struct man_node *);
static void     search_whatis(char *, char *);
static int      searchdir(char *, char *, char *);
static void     sortdir(DIR *, char ***);
```



```
static char     **split(char *, char);
static void     usage_man(void);
static void     usage_whatapro(void);
static void     usage_catman(void);
static void     usage_makewhatis(void);
static void     whatapro(struct man_node *, char *);

static char     language[MAXPATHLEN];   /* LC_MESSAGES */
static char     localedir[MAXPATHLEN];  /* locale specific path component */

static char     *newsection = NULL;

static int      manwidth = 0;

extern const char       *__progname;

int
main(int argc, char **argv)
{
        int             c, i;
        char            **pathv;
        char            *manpath = NULL;
        static struct man_node *mandirs = NULL;
        int             bmp_flags = 0;
        int             ret = 0;
        char            *opts;
        char            *mwstr;
        int             catman = 0;

        (void) setlocale(LC_ALL, "");
        (void) strcpy(language, setlocale(LC_MESSAGES, (char *)NULL));
        if (strcmp("C", language) != 0)
                (void) strlcpy(localedir, language, MAXPATHLEN);

#if !defined(TEXT_DOMAIN)
#define TEXT_DOMAIN "SYS_TEST"
#endif
        (void) textdomain(TEXT_DOMAIN);

        if (strcmp(__progname, "apropos") == 0) {
                apropos++;
                opts = "M:ds:";
        } else if (strcmp(__progname, "whatis") == 0) {
                apropos++;
                whatis++;
                opts = "M:ds:";
        } else if (strcmp(__progname, "catman") == 0) {
                catman++;
                makewhatis++;
                opts = "M:w";
        } else if (strcmp(__progname, "makewhatis") == 0) {
                makewhatis++;
                makewhatishere++;
                manpath = ".";
                opts = "";
        } else {
                opts = "M:adfklps:tw";
        }

        opterr = 0;
        while ((c = getopt(argc, argv, opts)) != -1) {
                switch (c) {
                case 'M':       /* Respecify path for man pages */
                        manpath = optarg;
                        break;
                case 'a':
```

```
258                         all++;
259                         break;
260                 case 'd':
261                         debug++;
262                         break;
263                 case 'f':
264                         whatis++;
265                         /*FALLTHROUGH*/
266                 case 'k':
267                         apropos++;
268                         break;
269                 case 'l':
270                         list++;
271                         /*FALLTHROUGH*/
272                 case 'p':
273                         printmp++;
274                         break;
275                 case 's':
276                         mansec = optarg;
277                         sargs++;
278                         break;
279                 case 't':
280                         psoutput++;
281                         break;
282                 case 'w':
283                         makewhatis++;
284                         break;
285                 case '?':
286                 default:
287                         if (apropos)
288                                 usage_whatapro();
289                         else if (catman)
290                                 usage_catman();
291                         else if (makewhatishere)
292                                 usage_makewhatis();
293                         else
294                                 usage_man();
295                 }
296         }
297         argc -= optind;
298         argv += optind;

300         if (argc == 0) {
301                 if (apropos) {
302                         (void) fprintf(stderr, gettext("%s what?\n"),
303                             __progname);
304                         exit(1);
305                 } else if (!printmp && !makewhatis) {
306                         (void) fprintf(stderr,
307                             gettext("What manual page do you want?\n"));
308                         exit(1);
309                 }
310         }

312         init_bintoman();
313         if (manpath == NULL && (manpath = getenv("MANPATH")) == NULL) {
314                 if ((manpath = getenv("PATH")) != NULL)
315                         bmp_flags = BMP_ISPATH | BMP_APPEND_DEFMANDIR;
316                 else
317                         manpath = DEFMANDIR;
318         }
319         pathv = split(manpath, ':');
320         mandirs = build_manpath(pathv, bmp_flags);
321         freev(pathv);
322         fullpaths(&mandirs);
```



```
324         if (makewhatis) {
325                 do_makewhatis(mandirs);
326                 exit(0);
327         }

329         if (printmp) {
330                 print_manpath(mandirs);
331                 exit(0);
332         }

334         /* Collect environment information */
335         if (isatty(STDOUT_FILENO) && (mwstr = getenv("MANWIDTH")) != NULL &&
336             *mwstr != '\0') {
337                 if (strcasecmp(mwstr, "tty") == 0) {
338                         struct winsize  ws;

340                         if (ioctl(0, TIOCGWINSZ, &ws) != 0)
341                                 warn("TIOCGWINSZ");
342                         else
343                                 manwidth = ws.ws_col;
344                 } else {
345                         manwidth = (int)strtol(mwstr, (char **)NULL, 10);
346                         if (manwidth < 0)
347                                 manwidth = 0;
348                 }
349         }
350         if (manwidth != 0) {
351                 DPRINTF("-- Using non-standard page width: %d\n", manwidth);
352         }

354         if ((pager = getenv("PAGER")) == NULL || *pager == '\0')
355                 pager = PAGER;
356         DPRINTF("-- Using pager: %s\n", pager);

358         for (i = 0; i < argc; i++) {
359                 char            *cmd;
360                 static struct man_node *mp;
361                 char            *pv[2];

363                 /*
364                  * If full path to command specified, customize
365                  * the manpath accordingly.
366                  */
367                 if ((cmd = strrchr(argv[i], '/')) != NULL) {
368                         *cmd = '\0';
369                         if ((pv[0] = strdup(argv[i])) == NULL)
370                                 err(1, "strdup");
371                         pv[1] = NULL;
372                         *cmd = '/';
373                         mp = build_manpath(pv,
374                             BMP_ISPATH | BMP_FALLBACK_DEFMANDIR);
375                 } else {
376                         mp = mandirs;
377                 }

379                 if (apropos)
380                         whatapro(mp, argv[i]);
381                 else
382                         ret += manual(mp, argv[i]);

384                 if (mp != NULL && mp != mandirs) {
385                         free(pv[0]);
386                         free_manp(mp);
387                 }
388         }
```

```
390         return (ret == 0 ? 0 : 1);
391 }

393 /*
394  * This routine builds the manpage structure from MANPATH or PATH,
395  * depending on flags.  See BMP_* definitions above for valid
396  * flags.
397  */
398 static struct man_node *
399 build_manpath(char **pathv, int flags)
400 {
401         struct man_node *manpage = NULL;
402         struct man_node *currp = NULL;
403         struct man_node *lastp = NULL;
404         char            **p;
405         char            **q;
406         char            *mand = NULL;
407         char            *mandir = DEFMANDIR;
408         int             s;
409         struct dupnode  *didup = NULL;
410         struct stat     sb;

412         s = sizeof (struct man_node);
413         for (p = pathv; *p != NULL; ) {
414                 if (flags & BMP_ISPATH) {
415                         if ((mand = path_to_manpath(*p)) == NULL)
416                                 goto next;
417                         free(*p);
418                         *p = mand;
419                 }
420                 q = split(*p, ',');
421                 if (stat(q[0], &sb) != 0 || (sb.st_mode & S_IFDIR) == 0) {
422                         freev(q);
423                         goto next;
424                 }

426                 if (access(q[0], R_OK | X_OK) == 0) {
427                         /*
428                          * Some element exists.  Do not append DEFMANDIR as a
429                          * fallback.
430                          */
431                         flags &= ~BMP_FALLBACK_DEFMANDIR;

433                         if ((currp = (struct man_node *)calloc(1, s)) == NULL)
434                                 err(1, "calloc");

436                         currp->frompath = (flags & BMP_ISPATH);

438                         if (manpage == NULL)
439                                 lastp = manpage = currp;

441                         getpath(currp, p);
442                         getsect(currp, p);

444                         /*
445                          * If there are no new elements in this path,
446                          * do not add it to the manpage list.
447                          */
448                         if (dupcheck(currp, &didup) != 0) {
449                                 freev(currp->secv);
450                                 free(currp);
451                         } else {
452                                 currp->next = NULL;
453                                 if (currp != manpage)
454                                         lastp->next = currp;
455                                 lastp = currp;
```



```
456                         }
457                 }
458                 freev(q);
459 next:
460                 /*
461                  * Special handling of appending DEFMANDIR. After all pathv
462                  * elements have been processed, append DEFMANDIR if needed.
463                  */
464                 if (p == &mandir)
465                         break;
466                 p++;
467                 if (*p != NULL)
468                         continue;
469                 if (flags & (BMP_APPEND_DEFMANDIR | BMP_FALLBACK_DEFMANDIR)) {
470                         p = &mandir;
471                         flags &= ~BMP_ISPATH;
472                 }
473         }

475         free_dupnode(didup);

477         return (manpage);
478 }

480 /*
481  * Store the mandir path into the manp structure.
482  */
483 static void
484 getpath(struct man_node *manp, char **pv)
485 {
486         char    *s = *pv;
487         int     i = 0;

489         while (*s != '\0' && *s != ',')
490                 i++, s++;

492         if ((manp->path = (char *)malloc(i + 1)) == NULL)
493                 err(1, "malloc");
494         (void) strlcpy(manp->path, *pv, i + 1);
495 }

497 /*
498  * Store the mandir's corresponding sections (submandir
499  * directories) into the manp structure.
500  */
501 static void
502 getsect(struct man_node *manp, char **pv)
503 {
504         char    *sections;
505         char    **sectp;

507         /* Just store all sections when doing makewhatis or apropos/whatis */
508         if (makewhatis || apropos) {
509                 manp->defsrch = 1;
510                 DPRINTF("-- Adding %s\n", manp->path);
511                 manp->secv = NULL;
512                 get_all_sect(manp);
513         } else if (sargs) {
514                 manp->secv = split(mansec, ',');
515                 for (sectp = manp->secv; *sectp; sectp++)
516                         lower(*sectp);
517         } else if ((sections = strchr(*pv, ',')) != NULL) {
518                 DPRINTF("-- Adding %s: MANSECTS=%s\n", manp->path, sections);
519                 manp->secv = split(++sections, ',');
520                 for (sectp = manp->secv; *sectp; sectp++)
521                         lower(*sectp);
```

```
522                 if (*manp->secv == NULL)
523                         get_all_sect(manp);
524         } else if ((sections = check_config(*pv)) != NULL) {
525                 manp->defsrch = 1;
526                 DPRINTF("-- Adding %s: from %s, MANSECTS=%s\n", manp->path,
527                     CONFIG, sections);
528                 manp->secv = split(sections, ',');
529                 for (sectp = manp->secv; *sectp; sectp++)
530                         lower(*sectp);
531                 if (*manp->secv == NULL)
532                         get_all_sect(manp);
533         } else {
534                 manp->defsrch = 1;
535                 DPRINTF("-- Adding %s: default sort order\n", manp->path);
536                 manp->secv = NULL;
537                 get_all_sect(manp);
538         }
539 }

541 /*
542  * Get suffices of all sub-mandir directories in a mandir.
543  */
544 static void
545 get_all_sect(struct man_node *manp)
546 {
547         DIR     *dp;
548         char    **dirv;
549         char    **dv;
550         char    **p;
551         char    *prev = NULL;
552         char    *tmp = NULL;
553         int     maxentries = MAXTOKENS;
554         int     entries = 0;

556         if ((dp = opendir(manp->path)) == 0)
557                 return;

559         sortdir(dp, &dirv);

561         (void) closedir(dp);

563         if (manp->secv == NULL) {
564                 if ((manp->secv = malloc(maxentries * sizeof (char *))) == NULL)
565                         err(1, "malloc");
566         }

568         for (dv = dirv, p = manp->secv; *dv; dv++) {
569                 if (strcmp(*dv, CONFIG) == 0) {
570                         free(*dv);
571                         continue;
572                 }

574                 free(tmp);
575                 if ((tmp = strdup(*dv + 3)) == NULL)
576                         err(1, "strdup");

578                 if (prev != NULL && strcmp(prev, tmp) == 0) {
579                         free(*dv);
580                         continue;
581                 }

583                 free(prev);
584                 if ((prev = strdup(*dv + 3)) == NULL)
585                         err(1, "strdup");

587                 if ((*p = strdup(*dv + 3)) == NULL)
```



```
588                         err(1, "strdup");

590                 p++; entries++;

592                 if (entries == maxentries) {
593                         maxentries += MAXTOKENS;
594                         if ((manp->secv = realloc(manp->secv,
595                             sizeof (char *) * maxentries)) == NULL)
596                                 err(1, "realloc");
597                         p = manp->secv + entries;
598                 }
599                 free(*dv);
600         }
601         free(tmp);
602         free(prev);
603         *p = NULL;
604         free(dirv);
605 }

607 /*
608  * Build whatis databases.
609  */
610 static void
611 do_makewhatis(struct man_node *manp)
612 {
613         struct man_node *p;
614         char            *ldir;

616         for (p = manp; p != NULL; p = p->next) {
617                 ldir = addlocale(p->path);
618                 if (*localedir != '\0' && getdirs(ldir, NULL, 0) > 0)
619                         mwpath(ldir);
620                 free(ldir);
621                 mwpath(p->path);
622         }
623 }

625 /*
626  * Count mandirs under the given manpath
627  */
628 static int
629 getdirs(char *path, char ***dirv, int flag)
630 {
631         DIR             *dp;
632         struct dirent   *d;
633         int             n = 0;
634         int             maxentries = MAXDIRS;
635         char            **dv = NULL;

637         if ((dp = opendir(path)) == NULL)
638                 return (0);

640         if (flag) {
641                 if ((*dirv = malloc(sizeof (char *) *
642                     maxentries)) == NULL)
643                         err(1, "malloc");
644                 dv = *dirv;
645         }
646         while ((d = readdir(dp))) {
647                 if (strncmp(d->d_name, "man", 3) != 0)
648                         continue;
649                 n++;

651                 if (flag) {
652                         if ((*dv = strdup(d->d_name + 3)) == NULL)
653                                 err(1, "strdup");
```

```
654                         dv++;
655                         if ((dv - *dirv) == maxentries) {
656                                 int     entries = maxentries;

658                                 maxentries += MAXTOKENS;
659                                 if ((*dirv = realloc(*dirv,
660                                     sizeof (char *) * maxentries)) == NULL)
661                                         err(1, "realloc");
662                                 dv = *dirv + entries;
663                         }
664                 }
665         }

667         (void) closedir(dp);
668         return (n);
669 }


672 /*
673  * Find matching whatis or apropos entries.
674  */
675 static void
676 whatapro(struct man_node *manp, char *word)
677 {
678         char            whatpath[MAXPATHLEN];
679         struct man_node *b;
680         char            *ldir;

682         for (b = manp; b != NULL; b = b->next) {
683                 if (*localedir != '\0') {
684                         ldir = addlocale(b->path);
685                         if (getdirs(ldir, NULL, 0) != 0) {
686                                 (void) snprintf(whatpath, sizeof (whatpath),
687                                     "%s/%s", ldir, WHATIS);
688                                 search_whatis(whatpath, word);
689                         }
690                         free(ldir);
691                 }
692                 (void) snprintf(whatpath, sizeof (whatpath), "%s/%s", b->path,
693                     WHATIS);
694                 search_whatis(whatpath, word);
695         }
696 }

698 static void
699 search_whatis(char *whatpath, char *word)
700 {
701         FILE            *fp;
702         char            *line = NULL;
703         size_t          linecap = 0;
704         char            *pkwd;
705         regex_t         preg;
706         char            **ss = NULL;
707         char            s[MAXNAMELEN];
708         int             i;

710         if ((fp = fopen(whatpath, "r")) == NULL) {
711                 perror(whatpath);
712                 return;
713         }

715         DPRINTF("-- Found %s: %s\n", WHATIS, whatpath);

717         /* Build keyword regex */
718         if (asprintf(&pkwd, "%s%s%s", (whatis) ? "\\<" : "",
719             word, (whatis) ? "\\>" : "") == -1)
```



```
720                 err(1, "asprintf");

722         if (regcomp(&preg, pkwd, REG_BASIC | REG_ICASE | REG_NOSUB) != 0)
723                 err(1, "regcomp");

725         if (sargs)
726                 ss = split(mansec, ',');

728         while (getline(&line, &linecap, fp) > 0) {
729                 if (regexec(&preg, line, 0, NULL, 0) == 0) {
730                         if (sargs) {
731                                 /* Section-restricted search */
732                                 for (i = 0; ss[i] != NULL; i++) {
733                                         (void) snprintf(s, sizeof (s), "(%s)",
734                                             ss[i]);
735                                         if (strstr(line, s) != NULL) {
736                                                 (void) printf("%s", line);
737                                                 break;
738                                         }
739                                 }
740                         } else {
741                                 (void) printf("%s", line);
742                         }
743                 }
744         }

746         if (ss != NULL)
747                 freev(ss);
748         free(pkwd);
749         (void) fclose(fp);
750 }


753 /*
754  * Split a string by specified separator.
755  */
756 static char **
757 split(char *s1, char sep)
758 {
759         char    **tokv, **vp;
760         char    *mp = s1, *tp;
761         int     maxentries = MAXTOKENS;
762         int     entries = 0;

764         if ((tokv = vp = malloc(maxentries * sizeof (char *))) == NULL)
765                 err(1, "malloc");

767         for (; mp && *mp; mp = tp) {
768                 tp = strchr(mp, sep);
769                 if (mp == tp) {
770                         tp++;
771                         continue;
772                 }
773                 if (tp) {
774                         size_t  len;

776                         len = tp - mp;
777                         if ((*vp = (char *)malloc(sizeof (char) *
778                             len + 1)) == NULL)
779                                 err(1, "malloc");
780                         (void) strncpy(*vp, mp, len);
781                         *(*vp + len) = '\0';
782                         tp++;
783                         vp++;
784                 } else {
785                         if ((*vp = strdup(mp)) == NULL)
```

```
786                                 err(1, "strdup");
787                         vp++;
788                 }
789                 entries++;
790                 if (entries == maxentries) {
791                         maxentries += MAXTOKENS;
792                         if ((tokv = realloc(tokv,
793                             maxentries * sizeof (char *))) == NULL)
794                                 err(1, "realloc");
795                         vp = tokv + entries;
796                 }
797         }
798         *vp = 0;

800         return (tokv);
801 }

803 /*
804  * Free a vector allocated by split()
805  */
806 static void
807 freev(char **v)
808 {
809         int i;
810         if (v != NULL) {
811                 for (i = 0; v[i] != NULL; i++) {
812                         free(v[i]);
813                 }
814                 free(v);
815         }
816 }

818 /*
819  * Convert paths to full paths if necessary
820  */
821 static void
822 fullpaths(struct man_node **manp_head)
823 {
824         char            *cwd = NULL;
825         char            *p;
826         int             cwd_gotten = 0;
827         struct man_node *manp = *manp_head;
828         struct man_node *b;
829         struct man_node *prev = NULL;

831         for (b = manp; b != NULL; b = b->next) {
832                 if (*(b->path) == '/') {
833                         prev = b;
834                         continue;
835                 }

837                 if (!cwd_gotten) {
838                         cwd = getcwd(NULL, MAXPATHLEN);
839                         cwd_gotten = 1;
840                 }

842                 if (cwd) {
843                         /* Relative manpath with cwd: make absolute */
844                         if (asprintf(&p, "%s/%s", cwd, b->path) == -1)
845                                 err(1, "asprintf");
846                         free(b->path);
847                         b->path = p;
848                 } else {
849                         /* Relative manpath but no cwd: omit path entry */
850                         if (prev)
851                                 prev->next = b->next;
```



```
852                         else
853                                 *manp_head = b->next;

855                         free_manp(b);
856                 }
857         }
858         free(cwd);
859 }

861 /*
862  * Free a man_node structure and its contents
863  */
864 static void
865 free_manp(struct man_node *manp)
866 {
867         char    **p;

869         free(manp->path);
870         p = manp->secv;
871         while ((p != NULL) && (*p != NULL)) {
872                 free(*p);
873                 p++;
874         }
875         free(manp->secv);
876         free(manp);
877 }


880 /*
881  * Map (in place) to lower case.
882  */
883 static void
884 lower(char *s)
885 {

887         if (s == 0)
888                 return;
889         while (*s) {
890                 if (isupper(*s))
891                         *s = tolower(*s);
892                 s++;
893         }
894 }


897 /*
898  * Compare function for qsort().
899  * Sort first by section, then by prefix.
900  */
901 static int
902 cmp(const void *arg1, const void *arg2)
903 {
904         int     n;
905         char    **p1 = (char **)arg1;
906         char    **p2 = (char **)arg2;

908         /* By section */
909         if ((n = strcmp(*p1 + 3, *p2 + 3)) != 0)
910                 return (n);

912         /* By prefix reversed */
913         return (strncmp(*p2, *p1, 3));
914 }


917 /*
```

```
 * Find a manpage.
 */
static int
manual(struct man_node *manp, char *name)
{
        struct man_node *p;
        struct man_node *local;
        int             ndirs = 0;
        char            *ldir;
        char            *ldirs[2];
        char            *fullname = name;
        char            *slash;

        if ((slash = strrchr(name, '/')) != NULL)
                name = slash + 1;

        /* For each path in MANPATH */
        found = 0;

        for (p = manp; p != NULL; p = p->next) {
                DPRINTF("-- Searching mandir: %s\n", p->path);

                if (*localedir != '\0') {
                        ldir = addlocale(p->path);
                        ndirs = getdirs(ldir, NULL, 0);
                        if (ndirs != 0) {
                                ldirs[0] = ldir;
                                ldirs[1] = NULL;
                                local = build_manpath(ldirs, 0);
                                DPRINTF("-- Locale specific subdir: %s\n",
                                    ldir);
                                mandir(local->secv, ldir, name, 1);
                                free_manp(local);
                        }
                        free(ldir);
                }

                /*
                 * Locale mandir not valid, man page in locale
                 * mandir not found, or -a option present
                 */
                if (ndirs == 0 || !found || all)
                        mandir(p->secv, p->path, name, 0);

                if (found && !all)
                        break;
        }

        if (!found) {
                if (sargs) {
                        (void) fprintf(stderr, gettext(
                            "No manual entry for %s in section(s) %s\n"),
                            fullname, mansec);
                } else {
                        (void) fprintf(stderr,
                            gettext("No manual entry for %s\n"), fullname);
                }

        }

        return (!found);
}


/*
 * For a specified manual directory, read, store and sort section subdirs.
```



```
 * For each section specified, find and search matching subdirs.
 */
static void
mandir(char **secv, char *path, char *name, int lspec)
{
        DIR     *dp;
        char    **dirv;
        char    **dv, **pdv;
        int     len, dslen;

        if ((dp = opendir(path)) == NULL)
                return;

        if (lspec)
                DPRINTF("-- Searching mandir: %s\n", path);

        sortdir(dp, &dirv);

        /* Search in the order specified by MANSECTS */
        for (; *secv; secv++) {
                len = strlen(*secv);
                for (dv = dirv; *dv; dv++) {
                        dslen = strlen(*dv + 3);
                        if (dslen > len)
                                len = dslen;
                        if (**secv == '\\') {
                                if (strcmp(*secv + 1, *dv + 3) != 0)
                                        continue;
                        } else if (strncasecmp(*secv, *dv + 3, len) != 0) {
                                if (!all &&
                                    (newsection = map_section(*secv, path))
                                    == NULL) {
                                        continue;
                                }
                                if (newsection == NULL)
                                        newsection = "";
                                if (strncmp(newsection, *dv + 3, len) != 0) {
                                        continue;
                                }
                        }

                        if (searchdir(path, *dv, name) == 0)
                                continue;

                        if (!all) {
                                pdv = dirv;
                                while (*pdv) {
                                        free(*pdv);
                                        pdv++;
                                }
                                (void) closedir(dp);
                                free(dirv);
                                return;
                        }

                        if (all && **dv == 'm' && *(dv + 1) &&
                            strcmp(*(dv + 1) + 3, *dv + 3) == 0)
                                        dv++;
                }
        }
        pdv = dirv;
        while (*pdv != NULL) {
                free(*pdv);
                pdv++;
        }
        free(dirv);
```

```
1050         (void) closedir(dp);
1051 }

1053 /*
1054  * Sort directories.
1055  */
1056 static void
1057 sortdir(DIR *dp, char ***dirv)
1058 {
1059         struct dirent   *d;
1060         char            **dv;
1061         int             maxentries = MAXDIRS;
1062         int             entries = 0;

1064         if ((dv = *dirv = malloc(sizeof (char *) *
1065             maxentries)) == NULL)
1066                 err(1, "malloc");
1067         dv = *dirv;

1069         while ((d = readdir(dp))) {
1070                 if (strcmp(d->d_name, ".") == 0 ||
1071                     strcmp(d->d_name, "..") == 0)
1072                         continue;

1074                 if (strncmp(d->d_name, "man", 3) == 0 ||
1075                     strncmp(d->d_name, "cat", 3) == 0) {
1076                         if ((*dv = strdup(d->d_name)) == NULL)
1077                                 err(1, "strdup");
1078                         dv++;
1079                         entries++;
1080                         if (entries == maxentries) {
1081                                 maxentries += MAXDIRS;
1082                                 if ((*dirv = realloc(*dirv,
1083                                     sizeof (char *) * maxentries)) == NULL)
1084                                         err(1, "realloc");
1085                                 dv = *dirv + entries;
1086                         }
1087                 }
1088         }
1089         *dv = 0;

1091         qsort((void *)*dirv, dv - *dirv, sizeof (char *), cmp);

1093 }


1096 /*
1097  * Search a section subdir for a given manpage.
1098  */
1099 static int
1100 searchdir(char *path, char *dir, char *name)
1101 {
1102         DIR             *sdp;
1103         struct dirent   *sd;
1104         char            sectpath[MAXPATHLEN];
1105         char            file[MAXNAMLEN];
1106         char            dname[MAXPATHLEN];
1107         char            *last;
1108         int             nlen;

1110         (void) snprintf(sectpath, sizeof (sectpath), "%s/%s", path, dir);
1111         (void) snprintf(file, sizeof (file), "%s.", name);

1113         if ((sdp = opendir(sectpath)) == NULL)
1114                 return (0);
```



```
1116         while ((sd = readdir(sdp))) {
1117                 char    *pname;

1119                 if ((pname = strdup(sd->d_name)) == NULL)
1120                         err(1, "strdup");
1121                 if ((last = strrchr(pname, '.')) != NULL &&
1122                     (strcmp(last, ".gz") == 0 || strcmp(last, ".bz2") == 0))
1123                         *last = '\0';
1124                 last = strrchr(pname, '.');
1125                 nlen = last - pname;
1126                 (void) snprintf(dname, sizeof (dname), "%.*s.", nlen, pname);
1127                 if (strcmp(dname, file) == 0 ||
1128                     strcmp(pname, name) == 0) {
1129                         (void) format(path, dir, name, sd->d_name);
1130                         (void) closedir(sdp);
1131                         free(pname);
1132                         return (1);
1133                 }
1134                 free(pname);
1135         }
1136         (void) closedir(sdp);

1138         return (0);
1139 }

1141 /*
1142  * Check the hash table of old directory names to see if there is a
1143  * new directory name.
1144  */
1145 static char *
1146 map_section(char *section, char *path)
1147 {
1148         int     i;
1149         int     len;
1150         char    fullpath[MAXPATHLEN];

1152         if (list)  /* -l option fall through */
1153                 return (NULL);

1155         for (i = 0; i <= ((sizeof (map)/sizeof (map[0]) - 1)); i++) {
1156                 if (strlen(section) > strlen(map[i].new_name)) {
1157                         len = strlen(section);
1158                 } else {
1159                         len = strlen(map[i].new_name);
1160                 }
1161                 if (strncmp(section, map[i].old_name, len) == 0) {
1162                         (void) snprintf(fullpath, sizeof (fullpath),
1163                             "%s/sman%s", path, map[i].new_name);
1164                         if (!access(fullpath, R_OK | X_OK)) {
1165                                 return (map[i].new_name);
1166                         } else {
1167                                 return (NULL);
1168                         }
1169                 }
1170         }

1172         return (NULL);
1173 }

1175 /*
1176  * Format the manpage.
1177  */
1178 static int
1179 format(char *path, char *dir, char *name, char *pg)
1180 {
1181         char            manpname[MAXPATHLEN], catpname[MAXPATHLEN];
```

```
1182         char            cmdbuf[BUFSIZ], tmpbuf[BUFSIZ];
1183         char            *cattool;
1184         int             utf8 = 0;
1185         struct stat     sbman, sbcat;

1187         found++;

1189         if (list) {
1190                 (void) printf(gettext("%s(%s)\t-M %s\n"), name, dir + 3, path);
1191                 return (-1);
1192         }

1194         (void) snprintf(manpname, sizeof (manpname), "%s/man%s/%s", path,
1195             dir + 3, pg);
1196         (void) snprintf(catpname, sizeof (catpname), "%s/cat%s/%s", path,
1197             dir + 3, pg);

1199         /* Can't do PS output if manpage doesn't exist */
1200         if (stat(manpname, &sbman) != 0 && psoutput)
1201                 return (-1);

1203         /*
1204          * If both manpage and catpage do not exist, manpname is
1205          * broken symlink, most likely.
1206          */
1207         if (stat(catpname, &sbcat) != 0 && stat(manpname, &sbman) != 0)
1208                 err(1, "%s", manpname);

1210         /* Setup cattool */
1211         if (fnmatch("*.gz", manpname, 0) == 0)
1212                 cattool = "gzcat";
1213         else if (fnmatch("*.bz2", manpname, 0) == 0)
1214                 cattool = "bzcat";
1215         else
1216                 cattool = "gzcat -f";

1218         /* Preprocess UTF-8 input with preconv (could be smarter) */
1219         if (strstr(path, "UTF-8") != NULL)
1220                 utf8 = 1;

1222         if (psoutput) {
1223                 (void) snprintf(cmdbuf, BUFSIZ,
1224                     "cd %s; %s %s%s | mandoc -Tps | lp -Tpostscript",
1225                     path, cattool, manpname,
1226                     utf8 ? " | " PRECONV " -e UTF-8" : "");
1227                 DPRINTF("-- Using manpage: %s\n", manpname);
1228                 goto cmd;
1229         }

1231         /*
1232          * Output catpage if:
1233          * - manpage doesn't exist
1234          * - output width is standard and catpage is recent enough
1235          */
1236         if (stat(manpname, &sbman) != 0 || (manwidth == 0 &&
1237             stat(catpname, &sbcat) == 0 && sbcat.st_mtime >= sbman.st_mtime)) {
1238                 DPRINTF("-- Using catpage: %s\n", catpname);
1239                 (void) snprintf(cmdbuf, BUFSIZ, "%s %s", pager, catpname);
1240                 goto cmd;
1241         }

1243         DPRINTF("-- Using manpage: %s\n", manpname);
1244         if (manwidth > 0)
1245                 (void) snprintf(tmpbuf, BUFSIZ, "-Owidth=%d ", manwidth);
1246         (void) snprintf(cmdbuf, BUFSIZ, "cd %s; %s %s%s | mandoc -T%s %s| %s",
1247             path, cattool, manpname,
```



```
1248                 utf8 ? " | " PRECONV " -e UTF-8 " : "",
1249                 utf8 ? "utf8" : "ascii", (manwidth > 0) ? tmpbuf : "", pager);

1251 cmd:
1252         DPRINTF("-- Command: %s\n", cmdbuf);

1254         if (!debug)
1255                 return (system(cmdbuf) == 0);
1256         else
1257                 return (0);
1258 }

1260 /*
1261  * Add <localedir> to the path.
1262  */
1263 static char *
1264 addlocale(char *path)
1265 {
1266         char    *tmp;

1268         if (asprintf(&tmp, "%s/%s", path, localedir) == -1)
1269                 err(1, "asprintf");

1271         return (tmp);
1272 }

1274 /*
1275  * Get the order of sections from man.cf.
1276  */
1277 static char *
1278 check_config(char *path)
1279 {
1280         FILE            *fp;
1281         char            *rc = NULL;
1282         char            *sect;
1283         char            fname[MAXPATHLEN];
1284         char            *line = NULL;
1285         size_t          linecap = 0;

1287         (void) snprintf(fname, MAXPATHLEN, "%s/%s", path, CONFIG);

1289         if ((fp = fopen(fname, "r")) == NULL)
1290                 return (NULL);

1292         while (getline(&line, &linecap, fp) > 0) {
1293                 if ((rc = strstr(line, "MANSECTS")) != NULL)
1294                         break;
1295         }

1297         (void) fclose(fp);

1299         if (rc == NULL || (sect = strchr(line, '=')) == NULL)
1300                 return (NULL);
1301         else
1302                 return (++sect);
1303 }


1306 /*
1307  * Initialize the bintoman array with appropriate device and inode info.
1308  */
1309 static void
1310 init_bintoman(void)
1311 {
1312         int i;
1313         struct stat sb;
```

```
1315         for (i = 0; bintoman[i].bindir != NULL; i++) {
1316                 if (stat(bintoman[i].bindir, &sb) == 0) {
1317                         bintoman[i].dev = sb.st_dev;
1318                         bintoman[i].ino = sb.st_ino;
1319                 } else {
1320                         bintoman[i].dev = NODEV;
1321                 }
1322         }
1323 }

1325 /*
1326  * If a duplicate is found, return 1.
1327  * If a duplicate is not found, add it to the dupnode list and return 0.
1328  */
1329 static int
1330 dupcheck(struct man_node *mnp, struct dupnode **dnp)
1331 {
1332         struct dupnode  *curdnp;
1333         struct secnode  *cursnp;
1334         struct stat     sb;
1335         int             i;
1336         int             rv = 1;
1337         int             dupfound;

1339         /* If the path doesn't exist, treat it as a duplicate */
1340         if (stat(mnp->path, &sb) != 0)
1341                 return (1);

1343         /* If no sections were found in the man dir, treat it as duplicate */
1344         if (mnp->secv == NULL)
1345                 return (1);

1347         /*
1348          * Find the dupnode structure for the previous time this directory
1349          * was looked at.  Device and inode numbers are compared so that
1350          * directories that are reached via different paths (e.g. /usr/man and
1351          * /usr/share/man) are treated as equivalent.
1352          */
1353         for (curdnp = *dnp; curdnp != NULL; curdnp = curdnp->next) {
1354                 if (curdnp->dev == sb.st_dev && curdnp->ino == sb.st_ino)
1355                         break;
1356         }

1358         /*
1359          * First time this directory has been seen. Add a new node to the
1360          * head of the list. Since all entries are guaranteed to be unique
1361          * copy all sections to new node.
1362          */
1363         if (curdnp == NULL) {
1364                 if ((curdnp = calloc(1, sizeof (struct dupnode))) == NULL)
1365                         err(1, "calloc");
1366                 for (i = 0; mnp->secv[i] != NULL; i++) {
1367                         if ((cursnp = calloc(1, sizeof (struct secnode)))
1368                             == NULL)
1369                                 err(1, "calloc");
1370                         cursnp->next = curdnp->secl;
1371                         curdnp->secl = cursnp;
1372                         if ((cursnp->secp = strdup(mnp->secv[i])) == NULL)
1373                                 err(1, "strdup");
1374                 }
1375                 curdnp->dev = sb.st_dev;
1376                 curdnp->ino = sb.st_ino;
1377                 curdnp->next = *dnp;
1378                 *dnp = curdnp;
1379                 return (0);
```



```
1380         }

1382         /*
1383          * Traverse the section vector in the man_node and the section list
1384          * in dupnode cache to eliminate all duplicates from man_node.
1385          */
1386         for (i = 0; mnp->secv[i] != NULL; i++) {
1387                 dupfound = 0;
1388                 for (cursnp = curdnp->secl; cursnp != NULL;
1389                     cursnp = cursnp->next) {
1390                         if (strcmp(mnp->secv[i], cursnp->secp) == 0) {
1391                                 dupfound = 1;
1392                                 break;
1393                         }
1394                 }
1395                 if (dupfound) {
1396                         mnp->secv[i][0] = '\0';
1397                         continue;
1398                 }


1401                 /*
1402                  * Update curdnp and set return value to indicate that this
1403                  * was not all duplicates.
1404                  */
1405                 if ((cursnp = calloc(1, sizeof (struct secnode))) == NULL)
1406                         err(1, "calloc");
1407                 cursnp->next = curdnp->secl;
1408                 curdnp->secl = cursnp;
1409                 if ((cursnp->secp = strdup(mnp->secv[i])) == NULL)
1410                         err(1, "strdup");
1411                 rv = 0;
1412         }

1414         return (rv);
1415 }

1417 /*
1418  * Given a bindir, return corresponding mandir.
1419  */
1420 static char *
1421 path_to_manpath(char *bindir)
1422 {
1423         char            *mand, *p;
1424         int             i;
1425         struct stat     sb;

1427         /* First look for known translations for specific bin paths */
1428         if (stat(bindir, &sb) != 0) {
1429                 return (NULL);
1430         }
1431         for (i = 0; bintoman[i].bindir != NULL; i++) {
1432                 if (sb.st_dev == bintoman[i].dev &&
1433                     sb.st_ino == bintoman[i].ino) {
1434                         if ((mand = strdup(bintoman[i].mandir)) == NULL)
1435                                 err(1, "strdup");
1436                         if ((p = strchr(mand, ',')) != NULL)
1437                                 *p = '\0';
1438                         if (stat(mand, &sb) != 0) {
1439                                 free(mand);
1440                                 return (NULL);
1441                         }
1442                         if (p != NULL)
1443                                 *p = ',';
1444                         return (mand);
1445                 }
```

```
1446         }

1448         /*
1449          * No specific translation found.  Try ‘dirname $bindir‘/share/man
1450          * and ‘dirname $bindir‘/man
1451          */
1452         if ((mand = malloc(MAXPATHLEN)) == NULL)
1453                 err(1, "malloc");
1454         if (strlcpy(mand, bindir, MAXPATHLEN) >= MAXPATHLEN) {
1455                 free(mand);
1456                 return (NULL);
1457         }

1459         /*
1460          * Advance to end of buffer, strip trailing /’s then remove last
1461          * directory component.
1462          */
1463         for (p = mand; *p != ’\0’; p++)
1464                 ;
1465         for (; p > mand && *p == ’/’; p--)
1466                 ;
1467         for (; p > mand && *p != ’/’; p--)
1468                 ;
1469         if (p == mand && *p == ’.’) {
1470                 if (realpath("..", mand) == NULL) {
1471                         free(mand);
1472                         return (NULL);
1473                 }
1474                 for (; *p != ’\0’; p++)
1475                         ;
1476         } else {
1477                 *p = ’\0’;
1478         }

1480         if (strlcat(mand, "/share/man", MAXPATHLEN) >= MAXPATHLEN) {
1481                 free(mand);
1482                 return (NULL);
1483         }

1485         if ((stat(mand, &sb) == 0) && S_ISDIR(sb.st_mode)) {
1486                 return (mand);
1487         }

1489         /*
1490          * Strip the /share/man off and try /man
1491          */
1492         *p = ’\0’;
1493         if (strlcat(mand, "/man", MAXPATHLEN) >= MAXPATHLEN) {
1494                 free(mand);
1495                 return (NULL);
1496         }
1497         if ((stat(mand, &sb) == 0) && S_ISDIR(sb.st_mode)) {
1498                 return (mand);
1499         }

1501         /*
1502          * No man or share/man directory found
1503          */
1504         free(mand);
1505         return (NULL);
1506 }

1508 /*
1509  * Free a linked list of dupnode structs.
1510  */
1511 void
```



```
1512 free_dupnode(struct dupnode *dnp) {
1513         struct dupnode *dnp2;
1514         struct secnode *snp;

1516         while (dnp != NULL) {
1517                 dnp2 = dnp;
1518                 dnp = dnp->next;
1519                 while (dnp2->secl != NULL) {
1520                         snp = dnp2->secl;
1521                         dnp2->secl = dnp2->secl->next;
1522                         free(snp->secp);
1523                         free(snp);
1524                 }
1525                 free(dnp2);
1526         }
1527 }

1529 /*
1530  * Print manp linked list to stdout.
1531  */
1532 void
1533 print_manpath(struct man_node *manp)
1534 {
1535         char    colon[2] = "\0\0";
1536         char    **secp;

1538         for (; manp != NULL; manp = manp->next) {
1539                 (void) printf("%s%s", colon, manp->path);
1540                 colon[0] = ’:’;

1542                 /*
1543                  * If man.cf or a directory scan was used to create section
1544                  * list, do not print section list again.  If the output of
1545                  * man -p is used to set MANPATH, subsequent runs of man
1546                  * will re-read man.cf and/or scan man directories as
1547                  * required.
1548                  */
1549                 if (manp->defsrch != 0)
1550                         continue;

1552                 for (secp = manp->secv; *secp != NULL; secp++) {
1553                         /*
1554                          * Section deduplication may have eliminated some
1555                          * sections from the vector. Avoid displaying this
1556                          * detail which would appear as ",," in output
1557                          */
1558                         if ((*secp)[0] != ’\0’)
1559                                 (void) printf(",%s", *secp);
1560                 }
1561         }
1562         (void) printf("\n");
1563 }

1565 static void
1566 usage_man(void)
1567 {

1569         (void) fprintf(stderr, gettext(
1570 "usage: man [-alptw] [-M path] [-s section] name ...\n"
1571 "       man [-M path] [-s section] -k keyword -- emulate apropos\n"
1572 "       man [-M path] [-s section] -f keyword -- emulate whatis\n"));

1574         exit(1);
1575 }

1577 static void
```

```
1578 usage_whatapro(void)
1579 {

1581         (void) fprintf(stderr, gettext(
1582 "usage: %s [-M path] [-s section] keyword ...\n"),
1583             whatis ? "whatis" : "apropos");

1585         exit(1);
1586 }

1588 static void
1589 usage_catman(void)
1590 {
1591         (void) fprintf(stderr, gettext(
1592 "usage: catman [-M path] [-w]\n"));

1594         exit(1);
1595 }

1597 static void
1598 usage_makewhatis(void)
1599 {
1600         (void) fprintf(stderr, gettext("usage: makewhatis\n"));

1602         exit(1);
1603 }
```

```
**********************************************************
     939 Wed Jul 16 14:05:07 2014
new/usr/src/cmd/man/man.h
mandoc import
**********************************************************
```

```
   1 /*
   2  * This file and its contents are supplied under the terms of the
   3  * Common Development and Distribution License ("CDDL"), version 1.0.
   4  * You may only use this file in accordance with the terms of version
   5  * 1.0 of the CDDL.
   6  *
   7  * A full copy of the text of the CDDL should have accompanied this
   8  * source.  A copy of the CDDL is also available via the Internet at
   9  * http://www.illumos.org/license/CDDL.
  10  */

  12 /*
  13  * Copyright 2012 Nexenta Systems, Inc. All rights reserved.
  14  * Copyright 2014 Garrett D'Amore <garrett@damore.org>
  15  */

  17 /*
  18  * Common definitions
  19  */

  21 #ifndef _MAN_H_
  22 #define _MAN_H_

  24 #define CONFIG          "man.cf"
  25 #define DEFMANDIR       "/usr/share/man"
  26 #define INDENT          24
  27 #define PAGER           "less -ins"
  28 #define WHATIS          "whatis"
  29 #define PRECONV         "/usr/lib/mandoc_preconv"

  31 #define LINE_ALLOC      4096
  32 #define MAXDIRS         128
  33 #define MAXTOKENS       64

  35 #define DPRINTF         if (debug) \
  36                                 (void) printf

  38 void    mwpath(char *path);

  40 #endif  /* _MAN_H_ */
```

```
**********************************************************
    2609 Wed Jul 16 14:05:07 2014
new/usr/src/cmd/man/stringlist.c
mandoc import
**********************************************************
```

```
/*
 * Copyright (c) 1994 Christos Zoulas
 * All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 * 4. The name of the author may not be used to endorse or promote products
 *    derived from this software without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE AUTHOR ''AS IS'' AND ANY EXPRESS
 * OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
 * WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY
 * DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
 * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
 * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
 * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
 * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
 * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 */

/*
 * Copyright 2012 Nexenta Systems, Inc. All rights reserved.
 */

#include <err.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#include "stringlist.h"

#define _SL_CHUNKSIZE   20

stringlist *
sl_init(void)
{
        stringlist      *sl;

        if ((sl = malloc(sizeof (stringlist))) == NULL)
                err(1, "malloc");

        sl->sl_cur = 0;
        sl->sl_max = _SL_CHUNKSIZE;
        sl->sl_str = malloc(sl->sl_max * sizeof (char *));
        if (sl->sl_str == NULL)
                err(1, "malloc");

        return (sl);
}

int
sl_add(stringlist *sl, char *name)
{
```



```
        if (sl->sl_cur == sl->sl_max - 1) {
                sl->sl_max += _SL_CHUNKSIZE;
                sl->sl_str = realloc(sl->sl_str, sl->sl_max * sizeof (char *));
                if (sl->sl_str == NULL)
                        return (-1);
        }
        sl->sl_str[sl->sl_cur++] = name;

        return (0);
}


void
sl_free(stringlist *sl, int all)
{
        size_t i;

        if (sl == NULL)
                return;
        if (sl->sl_str) {
                if (all)
                        for (i = 0; i < sl->sl_cur; i++)
                                free(sl->sl_str[i]);
                free(sl->sl_str);
        }
        free(sl);
}


char *
sl_find(stringlist *sl, char *name)
{
        size_t  i;

        for (i = 0; i < sl->sl_cur; i++)
                if (strcmp(sl->sl_str[i], name) == 0)
                        return (sl->sl_str[i]);

        return (NULL);
}
```

```
**********************************************************
    2061 Wed Jul 16 14:05:07 2014
new/usr/src/cmd/man/stringlist.h
mandoc import
**********************************************************
```

```
/*
 * Copyright (c) 1994 Christos Zoulas
 * All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 * 3. All advertising materials mentioning features or use of this software
 *    must display the following acknowledgement:
 *      This product includes software developed by Christos Zoulas.
 * 4. The name of the author may not be used to endorse or promote products
 *    derived from this software without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE AUTHOR ''AS IS'' AND ANY EXPRESS
 * OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
 * WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY
 * DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
 * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
 * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
 * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
 * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
 * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 */

/*
 * Copyright 2012 Nexenta Systems, Inc. All rights reserved.
 */

#ifndef _STRINGLIST_H_
#define _STRINGLIST_H_

#include <sys/types.h>

typedef struct _stringlist {
        char    **sl_str;
        size_t  sl_max;
        size_t  sl_cur;
} stringlist;

stringlist      *sl_init(void);
int             sl_add(stringlist *, char *);
void            sl_free(stringlist *, int);
char            *sl_find(stringlist *, char *);

#endif  /* _STRINGLIST_H_ */
```

```
**********************************************************
    1770 Wed Jul 16 14:05:07 2014
new/usr/src/cmd/mandoc/Makefile
mandoc import
**********************************************************
```

```
#
# This file and its contents are supplied under the terms of the
# Common Development and Distribution License ("CDDL"), version 1.0.
# You may only use this file in accordance with the terms of version
# 1.0 of the CDDL.
#
# A full copy of the text of the CDDL should have accompanied this
# source.  A copy of the CDDL is also available via the Internet at
# http://www.illumos.org/license/CDDL.
#

#
# Copyright 2012 Nexenta Systems, Inc.  All rights reserved.
# Copyright 2014 Garrett D'Amore <garrett@damore.org>
#

PROGS=          mandoc mandoc_preconv
mandoc_OBJS =   arch.o att.o chars.o eqn.o eqn_html.o eqn_term.o        \
                html.o lib.o main.o man.o man_hash.o man_html.o         \
                man_macro.o man_term.o man_validate.o mandoc.o mdoc.o   \
                mdoc_argv.o mdoc_hash.o mdoc_html.o mdoc_macro.o        \
                mdoc_man.o mdoc_term.o mdoc_validate.o msec.o out.o     \
                read.o roff.o st.o tbl.o tbl_data.o tbl_html.o          \
                tbl_layout.o tbl_opts.o tbl_term.o term.o term_ascii.o  \
                term_ps.o tree.o vol.o

preconv_OBJS =  preconv.o

# We place preconv in /usr/lib.  This is done to avoid conflicting with
# GNU groff, which puts it into /usr/bin.  We also rename it so that it
# will only be seen by mandoc -- it isn't intended for general end-user use.

ROOTPROGS =     $(ROOTBIN)/mandoc $(ROOTLIB)/mandoc_preconv

OBJS=           $(preconv_OBJS) $(mandoc_OBJS)

include         $(SRC)/cmd/Makefile.cmd

CFLAGS +=       $(CC_VERBOSE)

CPPFLAGS +=     -DHAVE_CONFIG_H -DUSE_WCHAR     \
                -DOSNAME="\"illumos\""  \
                -DVERSION="\"1.12.1\""

.KEEP_STATE:

all:            $(PROGS)

mandoc_preconv: $(preconv_OBJS)
                $(LINK.c) $(preconv_OBJS) -o $@ $(LDLIBS)
                $(POST_PROCESS)

mandoc:         $(mandoc_OBJS)
                $(LINK.c) $(mandoc_OBJS) -o $@ $(LDLIBS)
                $(POST_PROCESS)

clean:
                $(RM) $(OBJS)

install:        all $(ROOTPROGS)
```



```
include         $(SRC)/cmd/Makefile.targ
```

****
825 Wed Jul 16 14:05:07 2014
new/usr/src/cmd/mandoc/THIRDPARTYLICENSE
mandoc import
****

```
**********************************************************
      41 Wed Jul 16 14:05:07 2014
new/usr/src/cmd/mandoc/THIRDPARTYLICENSE.descrip
mandoc import
**********************************************************
```

```
   1 MANDOC - FORMAT AND DISPLAY UNIX MANUALS
```

```
**********************************************************
    1208 Wed Jul 16 14:05:07 2014
new/usr/src/cmd/mandoc/config.h
mandoc import
**********************************************************
```

```
#ifndef MANDOC_CONFIG_H
#define MANDOC_CONFIG_H

#if defined(__linux__) || defined(__MINT__)
# define _GNU_SOURCE /* strptime(), getsubopt() */
#endif

#include <stdio.h>

#define HAVE_STRPTIME
#define HAVE_GETSUBOPT
#define HAVE_STRLCAT
#define HAVE_STRLCPY

#include <sys/types.h>

#if !defined(__BEGIN_DECLS)
#  ifdef __cplusplus
#  define       __BEGIN_DECLS           extern "C" {
#  else
#  define       __BEGIN_DECLS
#  endif
#endif
#if !defined(__END_DECLS)
#  ifdef __cplusplus
#  define       __END_DECLS             }
#  else
#  define       __END_DECLS
#  endif
#endif

#if defined(__APPLE__)
# define htobe32(x) OSSwapHostToBigInt32(x)
# define betoh32(x) OSSwapBigToHostInt32(x)
# define htobe64(x) OSSwapHostToBigInt64(x)
# define betoh64(x) OSSwapBigToHostInt64(x)
#elif defined(__linux__)
# define betoh32(x) be32toh(x)
# define betoh64(x) be64toh(x)
#endif

#ifndef HAVE_STRLCAT
extern  size_t    strlcat(char *, const char *, size_t);
#endif
#ifndef HAVE_STRLCPY
extern  size_t    strlcpy(char *, const char *, size_t);
#endif
#ifndef HAVE_GETSUBOPT
extern  int       getsubopt(char **, char * const *, char **);
extern  char     *suboptarg;
#endif
#ifndef HAVE_FGETLN
extern  char     *fgetln(FILE *, size_t *);
#endif

#endif /* MANDOC_CONFIG_H */
```

****
497 Wed Jul 16 14:05:07 2014
new/usr/src/cmd/mandoc/lib.in
mandoc import
****

```
/*
 * This file and its contents are supplied under the terms of the
 * Common Development and Distribution License ("CDDL"), version 1.0.
 * You may only use this file in accordance with the terms of version
 * 1.0 of the CDDL.
 *
 * A full copy of the text of the CDDL should have accompanied this
 * source.  A copy of the CDDL is also available via the Internet at
 * http://www.illumos.org/license/CDDL.
 */

/*
 * Copyright 2012 Nexenta Systems, Inc. All rights reserved.
 */

/*
 * TBD
 */
```

```
**********************************************************
   11663 Wed Jul 16 14:05:07 2014
new/usr/src/cmd/mandoc/msec.in
mandoc import
**********************************************************
/*
 * This file and its contents are supplied under the terms of the
 * Common Development and Distribution License ("CDDL"), version 1.0.
 * You may only use this file in accordance with the terms of version
 * 1.0 of the CDDL.
 *
 * A full copy of the text of the CDDL should have accompanied this
 * source.  A copy of the CDDL is also available via the Internet at
 * http://www.illumos.org/license/CDDL.
 */

/*
 * Copyright 2012 Nexenta Systems, Inc. All rights reserved.
 */

LINE("1",               "User Commands")
LINE("1B",              "illumos/BSD Compatibility Package Commands")
LINE("1b",              "illumos/BSD Compatibility Package Commands")
LINE("1C",              "Communication Commands")
LINE("1c",              "Communication Commands")
LINE("1F",              "FMLI Commands")
LINE("1f",              "FMLI Commands")
LINE("1G",              "Graphics and CAD Commands")
LINE("1g",              "Graphics and CAD Commands")
LINE("1HAS",            "User Commands")
LINE("1has",            "User Commands")
LINE("1M",              "Maintenance Commands")
LINE("1m",              "Maintenance Commands")
LINE("1S",              "illumos Specific Commands")
LINE("1s",              "illumos Specific Commands")
LINE("2",               "System Calls")
LINE("3",               "Introduction to Library Functions")
LINE("3AIO",            "Asynchronous I/O Library Functions")
LINE("3aio",            "Asynchronous I/O Library Functions")
LINE("3BSM",            "Security and Auditing Library Functions")
LINE("3bsm",            "Security and Auditing Library Functions")
LINE("3C",              "Standard C Library Functions")
LINE("3c",              "Standard C Library Functions")
LINE("3C_DB",           "Threads Debugging Library Functions")
LINE("3C_db",           "Threads Debugging Library Functions")
LINE("3CFGADM",         "Configuration Administration Library Functions")
LINE("3cfgadm",         "Configuration Administration Library Functions")
LINE("3COMPPUTIL",      "Communication Protocol Parser Utilities Library Functio
LINE("3compputil",      "Communication Protocol Parser Utilities Library Functio
LINE("3CONTRACT",       "Contract Management Library Functions")
LINE("3contract",       "Contract Management Library Functions")
LINE("3CPC",            "CPU Performance Counters Library Functions")
LINE("3cpc",            "CPU Performance Counters Library Functions")
LINE("3CURSES",         "Curses Library Functions")
LINE("3curses",         "Curses Library Functions")
LINE("3DAT",            "Direct Access Transport Library Functions")
LINE("3dat",            "Direct Access Transport Library Functions")
LINE("3DEVID",          "Device ID Library Functions")
LINE("3devid",          "Device ID Library Functions")
LINE("3DEVINFO",        "Device Information Library Functions")
LINE("3devinfo",        "Device Information Library Functions")
LINE("3DL",             "Dynamic Linking Library Functions")
LINE("3dl",             "Dynamic Linking Library Functions")
LINE("3DLPI",           "Data Link Provider Interface Library Functions")
LINE("3dlpi",           "Data Link Provider Interface Library Functions")
LINE("3DMI",            "DMI Library Functions")
```



```
LINE("3dmi",            "DMI Library Functions")
LINE("3DNS_SD",         "DNS Service Discovery Library Functions")
LINE("3dns_sd",         "DNS Service Discovery Library Functions")
LINE("3DOOR",           "Door Library Functions")
LINE("3door",           "Door Library Functions")
LINE("3ELF",            "ELF Library Functions")
LINE("3elf",            "ELF Library Functions")
LINE("3EXACCT",         "Extended Accounting File Access Library Functions")
LINE("3exacct",         "Extended Accounting File Access Library Functions")
LINE("3EXT",            "Extended Library Functions")
LINE("3ext",            "Extended Library Functions")
LINE("3FCOE",           "FCoE Port Management Library Functions")
LINE("3fcoe",           "FCoE Port Management Library Functions")
LINE("3FSTYP",          "File System Type Identification Library Functions")
LINE("3fstyp",          "File System Type Identification Library Functions")
LINE("3GEN",            "String Pattern-Matching Library Functions")
LINE("3gen",            "String Pattern-Matching Library Functions")
LINE("3GSS",            "Generic Security Services API Library Functions")
LINE("3gss",            "Generic Security Services API Library Functions")
LINE("3HEAD",           "Headers")
LINE("3head",           "Headers")
LINE("3ISCSIT",         "iSCSI Management Library Functions")
LINE("3iscsit",         "iSCSI Management Library Functions")
LINE("3KRB",            "Kerberos Library Functions")
LINE("3krb",            "Kerberos Library Functions")
LINE("3KSTAT",          "Kernel Statistics Library Functions")
LINE("3kstat",          "Kernel Statistics Library Functions")
LINE("3KVM",            "Kernel VM Library Functions")
LINE("3kvm",            "Kernel VM Library Functions")
LINE("3LDAP",           "LDAP Library Functions")
LINE("3ldap",           "LDAP Library Functions")
LINE("3LGRP",           "Locality Group Library Functions")
LINE("3lgrp",           "Locality Group Library Functions")
LINE("3LIB",            "Interface Libraries")
LINE("3lib",            "Interface Libraries")
LINE("3LIBUCB",         "illumos/BSD Compatibility Interface Libraries")
LINE("3libucb",         "illumos/BSD Compatibility Interface Libraries")
LINE("3M",              "Mathematical Library Functions")
LINE("3m",              "Mathematical Library Functions")
LINE("3MAIL",           "User Mailbox Library Functions")
LINE("3mail",           "User Mailbox Library Functions")
LINE("3MALLOC",         "Memory Allocation Library Functions")
LINE("3malloc",         "Memory Allocation Library Functions")
LINE("3MP",             "Multiple Precision Library Functions")
LINE("3mp",             "Multiple Precision Library Functions")
LINE("3MPAPI",          "Common Multipath Management Library Functions")
LINE("3mpapi",          "Common Multipath Management Library Functions")
LINE("3NSL",            "Networking Services Library Functions")
LINE("3nsl",            "Networking Services Library Functions")
LINE("3NVPAIR",         "Name-value Pair Library Functions")
LINE("3nvpair",         "Name-value Pair Library Functions")
LINE("3PAM",            "PAM Library Functions")
LINE("3pam",            "PAM Library Functions")
LINE("3PAPI",           "PAPI Library Functions")
LINE("3papi",           "PAPI Library Functions")
LINE("3PERL",           "Perl Library Functions")
LINE("3perl",           "Perl Library Functions")
LINE("3PICL",           "PICL Library Functions")
LINE("3picl",           "PICL Library Functions")
LINE("3PICLTREE",       "PICL Plug-In Library Functions")
LINE("3picltree",       "PICL Plug-In Library Functions")
LINE("3PLOT",           "Graphics Interface Library Functions")
LINE("3plot",           "Graphics Interface Library Functions")
LINE("3POOL",           "Pool Configuration Manipulation Library Functions")
LINE("3pool",           "Pool Configuration Manipulation Library Functions")
LINE("3PROC",           "Process Control Library Functions")
```

```
128 LINE("3proc",           "Process Control Library Functions")
129 LINE("3PROJECT",        "Project Database Access Library Functions")
130 LINE("3project",        "Project Database Access Library Functions")
131 LINE("3RAC",           "Remote Asynchronous Calls Library Functions")
132 LINE("3rac",           "Remote Asynchronous Calls Library Functions")
133 LINE("3RESOLV",        "Resolver Library Functions")
134 LINE("3resolv",        "Resolver Library Functions")
135 LINE("3RPC",           "RPC Library Functions")
136 LINE("3rpc",           "RPC Library Functions")
137 LINE("3RSM",           "Remote Shared Memory Library Functions")
138 LINE("3rsm",           "Remote Shared Memory Library Functions")
139 LINE("3RT",            "Realtime Library Functions")
140 LINE("3rt",            "Realtime Library Functions")
141 LINE("3SASL",          "Simple Authentication Security Layer Library Functions"
142 LINE("3sasl",          "Simple Authentication Security Layer Library Functions"
143 LINE("3SCF",           "Service Configuration Facility Library Functions")
144 LINE("3scf",           "Service Configuration Facility Library Functions")
145 LINE("3SCHED",         "LWP Scheduling Library Functions")
146 LINE("3sched",         "LWP Scheduling Library Functions")
147 LINE("3SEC",           "File Access Control Library Functions")
148 LINE("3sec",           "File Access Control Library Functions")
149 LINE("3SECDB",         "Security Attributes Database Library Functions")
150 LINE("3secdb",         "Security Attributes Database Library Functions")
151 LINE("3SIP",           "Session Initiation Protocol Library Functions")
152 LINE("3sip",           "Session Initiation Protocol Library Functions")
153 LINE("3SLP",           "Service Location Protocol Library Functions")
154 LINE("3slp",           "Service Location Protocol Library Functions")
155 LINE("3SNMP",          "SNMP Library Functions")
156 LINE("3snmp",          "SNMP Library Functions")
157 LINE("3SOCKET",        "Sockets Library Functions")
158 LINE("3socket",        "Sockets Library Functions")
159 LINE("3STMF",          "SCSI Target Mode Framework Library Functions")
160 LINE("3stmf",          "SCSI Target Mode Framework Library Functions")
161 LINE("3SYSEVENT",      "System Event Library Functions")
162 LINE("3sysevent",      "System Event Library Functions")
163 LINE("3TECLA",         "Interactive Command-line Input Library Functions")
164 LINE("3tecla",         "Interactive Command-line Input Library Functions")
165 LINE("3THR",           "Threads Library Functions")
166 LINE("3thr",           "Threads Library Functions")
167 LINE("3TNF",           "TNF Library Functions")
168 LINE("3tnf",           "TNF Library Functions")
169 LINE("3TSOL",          "Trusted Extensions Library Functions")
170 LINE("3tsol",          "Trusted Extensions Library Functions")
171 LINE("3UCB",           "illumos/BSD Compatibility Library Functions")
172 LINE("3ucb",           "illumos/BSD Compatibility Library Functions")
173 LINE("3UUID",          "Universally Unique Identifier Library Functions")
174 LINE("3uuid",          "Universally Unique Identifier Library Functions")
175 LINE("3VOLMGT",        "Volume Management Library Functions")
176 LINE("3volmgt",        "Volume Management Library Functions")
177 LINE("3XCURSES",       "X/Open Curses Library Functions")
178 LINE("3xcurses",       "X/Open Curses Library Functions")
179 LINE("3XFN",           "XFN Interface Library Functions")
180 LINE("3xfn",           "XFN Interface Library Functions")
181 LINE("3XNET",          "X/Open Networking Services Library Functions")
182 LINE("3xnet",          "X/Open Networking Services Library Functions")
183 LINE("3B",             "illumos/BSD Compatibility Library Functions")
184 LINE("3b",             "illumos/BSD Compatibility Library Functions")
185 LINE("3E",             "C Library Functions")
186 LINE("3e",             "C Library Functions")
187 LINE("3F",             "Fortran Library Routines")
188 LINE("3f",             "Fortran Library Routines")
189 LINE("3G",             "C Library Functions")
190 LINE("3g",             "C Library Functions")
191 LINE("3K",             "Kernel VM Library Functions")
192 LINE("3k",             "Kernel VM Library Functions")
193 LINE("3L",             "Lightweight Processes Library")
```



```
194 LINE("3l",             "Lightweight Processes Library")
195 LINE("3N",             "Network Functions")
196 LINE("3n",             "Network Functions")
197 LINE("3R",             "Realtime Library")
198 LINE("3r",             "Realtime Library")
199 LINE("3S",             "Standard I/O Functions")
200 LINE("3s",             "Standard I/O Functions")
201 LINE("3T",             "Threads Library")
202 LINE("3t",             "Threads Library")
203 LINE("3W",             "C Library Functions")
204 LINE("3w",             "C Library Functions")
205 LINE("3X",             "Miscellaneous Library Functions")
206 LINE("3x",             "Miscellaneous Library Functions")
207 LINE("3XC",            "X/Open Curses Library Functions")
208 LINE("3xc",            "X/Open Curses Library Functions")
209 LINE("3XN",            "X/Open Networking Services Library Functions")
210 LINE("3xn",            "X/Open Networking Services Library Functions")
211 LINE("4",              "File Formats")
212 LINE("4B",             "illumos/BSD Compatibility Package File Formats")
213 LINE("4b",             "illumos/BSD Compatibility Package File Formats")
214 LINE("5",              "Standards, Environments, and Macros")
215 LINE("6",              "Games and Demos")
216 LINE("7",              "Device and Network Interfaces")
217 LINE("7B",             "illumos/BSD Compatibility Special Files")
218 LINE("7b",             "illumos/BSD Compatibility Special Files")
219 LINE("7D",             "Devices")
220 LINE("7d",             "Devices")
221 LINE("7FS",            "File Systems")
222 LINE("7fs",            "File Systems")
223 LINE("7I",             "Ioctl Requests")
224 LINE("7i",             "Ioctl Requests")
225 LINE("7IPP",           "IP Quality of Service Modules")
226 LINE("7ipp",           "IP Quality of Service Modules")
227 LINE("7M",             "STREAMS Modules")
228 LINE("7m",             "STREAMS Modules")
229 LINE("7P",             "Protocols")
230 LINE("7p",             "Protocols")
231 LINE("8",              "Maintenance Procedures")
232 LINE("8C",             "Maintenance Procedures")
233 LINE("8c",             "Maintenance Procedures")
234 LINE("8S",             "Maintenance Procedures")
235 LINE("8s",             "Maintenance Procedures")
236 LINE("9",              "Device Driver Interfaces")
237 LINE("9E",             "Driver Entry Points")
238 LINE("9e",             "Driver Entry Points")
239 LINE("9F",             "Kernel Functions for Drivers")
240 LINE("9f",             "Kernel Functions for Drivers")
241 LINE("9P",             "Kernel Properties for Drivers")
242 LINE("9p",             "Kernel Properties for Drivers")
243 LINE("9S",             "Data Structures for Drivers")
244 LINE("9s",             "Data Structures for Drivers")
```

```
**********************************************************
   13186 Wed Jul 16 14:05:08 2014
new/usr/src/man/man1/Makefile
feedback from Hans
Add catman, makewhatis functionality.  Print an error if the whatis database
is missing.
mandoc import
**********************************************************
```

```
#
# This file and its contents are supplied under the terms of the
# Common Development and Distribution License ("CDDL"), version 1.0.
# You may only use this file in accordance with the terms of version
# 1.0 of the CDDL.
#
# A full copy of the text of the CDDL should have accompanied this
# source.  A copy of the CDDL is also available via the Internet
# at http://www.illumos.org/license/CDDL.
#

#
# Copyright 2011, Richard Lowe
# Copyright 2013 Nexenta Systems, Inc.  All rights reserved.
# Copyright 2014 Garrett D'Amore <garrett@damore.org>
#

include         $(SRC)/Makefile.master

MANSECT=        1

MANFILES=       acctcom.1                       \
                adb.1                           \
                addbib.1                        \
                alias.1                         \
                allocate.1                      \
                amt.1                           \
                appcert.1                       \
                apptrace.1                      \
                apropos.1                       \
                ar.1                            \
                arch.1                          \
                asa.1                           \
                at.1                            \
                atq.1                           \
                atrm.1                          \
                audioconvert.1                  \
                audioctl.1                      \
                audioplay.1                     \
                audiorecord.1                   \
                audiotest.1                     \
                auths.1                         \
                awk.1                           \
                banner.1                        \
                basename.1                      \
                bc.1                            \
                bdiff.1                         \
                bfs.1                           \
                break.1                         \
                builtin.1                       \
                cal.1                           \
                calendar.1                      \
                cancel.1                        \
                cat.1                           \
                cd.1                            \
                cdrw.1                          \
                checknr.1                       \
                chgrp.1                         \
```



```
                chkey.1                         \
                chmod.1                         \
                chown.1                         \
                ckdate.1                        \
                ckgid.1                         \
                ckint.1                         \
                ckitem.1                        \
                ckkeywd.1                       \
                ckpath.1                        \
                ckrange.1                       \
                ckstr.1                         \
                cksum.1                         \
                cktime.1                        \
                ckuid.1                         \
                ckyorn.1                        \
                clear.1                         \
                cmp.1                           \
                col.1                           \
                comm.1                          \
                command.1                       \
                compress.1                      \
                cp.1                            \
                cpio.1                          \
                cputrack.1                      \
                crle.1                          \
                crontab.1                       \
                crypt.1                         \
                csh.1                           \
                csplit.1                        \
                ctags.1                         \
                ctrun.1                         \
                ctstat.1                        \
                ctwatch.1                       \
                cut.1                           \
                date.1                          \
                dc.1                            \
                deallocate.1                    \
                deroff.1                        \
                dhcpinfo.1                      \
                diff.1                          \
                diff3.1                         \
                diffmk.1                        \
                digest.1                        \
                dircmp.1                        \
                dis.1                           \
                disown.1                        \
                dispgid.1                       \
                dispuid.1                       \
                dos2unix.1                      \
                download.1                      \
                dpost.1                         \
                du.1                            \
                dump.1                          \
                dumpcs.1                        \
                echo.1                          \
                ed.1                            \
                egrep.1                         \
                eject.1                         \
                elfdump.1                       \
                elfedit.1                       \
                elfsign.1                       \
                elfwrap.1                       \
                enable.1                        \
                encrypt.1                       \
                enhance.1                       \
                env.1                           \
```

```
125                eqn.1                        \
126                exec.1                       \
127                exit.1                       \
128                expand.1                     \
129                expr.1                       \
130                exstr.1                      \
131                factor.1                     \
132                fdformat.1                   \
133                fgrep.1                      \
134                file.1                       \
135                filesync.1                   \
136                find.1                       \
137                finger.1                     \
138                fmt.1                        \
139                fmtmsg.1                     \
140                fold.1                       \
141                ftp.1                        \
142                ftpcount.1                   \
143                ftpwho.1                     \
144                gcore.1                      \
145                gencat.1                     \
146                genmsg.1                     \
147                getconf.1                    \
148                getfacl.1                    \
149                getlabel.1                   \
150                getopt.1                     \
151                getoptcvt.1                  \
152                getopts.1                    \
153                gettext.1                    \
154                gettxt.1                     \
155                getzonepath.1                \
156                glob.1                       \
157                gprof.1                      \
158                grep.1                       \
159                groups.1                     \
160                hash.1                       \
161                head.1                       \
162                history.1                    \
163                hostid.1                     \
164                hostname.1                   \
165                iconv.1                      \
166                indxbib.1                    \
167                Intro.1                      \
168                ipcrm.1                      \
169                ipcs.1                       \
170                isainfo.1                    \
171                isalist.1                    \
172                jobs.1                       \
173                join.1                       \
174                kbd.1                        \
175                kdestroy.1                   \
176                keylogin.1                   \
177                keylogout.1                  \
178                kill.1                       \
179                kinit.1                      \
180                klist.1                      \
181                kmdb.1                       \
182                kmfcfg.1                     \
183                kpasswd.1                    \
184                krb5-config.1                \
185                ksh93.1                      \
186                ktutil.1                     \
187                lari.1                       \
188                last.1                       \
189                lastcomm.1                   \
190                ld.1                         \
```



```
191                ldap.1                       \
192                ldapdelete.1                 \
193                ldaplist.1                   \
194                ldapmodify.1                 \
195                ldapmodrdn.1                 \
196                ldapsearch.1                 \
197                ldd.1                        \
198                ld.so.1.1                    \
199                let.1                        \
200                lex.1                        \
201                lgrpinfo.1                   \
202                limit.1                      \
203                line.1                       \
204                list_devices.1               \
205                listusers.1                  \
206                ln.1                         \
207                loadkeys.1                   \
208                locale.1                     \
209                localedef.1                  \
210                logger.1                     \
211                login.1                      \
212                logname.1                    \
213                logout.1                     \
214                look.1                       \
215                lookbib.1                    \
216                lorder.1                     \
217                lp.1                         \
218                lpstat.1                     \
219                ls.1                         \
220                m4.1                         \
221                mac.1                        \
222                mach.1                       \
223                machid.1                     \
224                madv.so.1.1                  \
225                mail.1                       \
226                mailcompat.1                 \
227                mailq.1                      \
228                mailstats.1                  \
229                mailx.1                      \
230                makekey.1                    \
231                man.1                        \
232                mandoc.1                     \
233                mconnect.1                   \
234                mcs.1                        \
235                mdb.1                        \
236                mesg.1                       \
237                mkdir.1                      \
238                mkmsgs.1                     \
239                mktemp.1                     \
240                moe.1                        \
241                more.1                       \
242                mpss.so.1.1                  \
243                msgcc.1                      \
244                msgcpp.1                     \
245                msgcvt.1                     \
246                msgfmt.1                     \
247                msggen.1                     \
248                msgget.1                     \
249                mt.1                         \
250                mv.1                         \
251                nawk.1                       \
252                nc.1                         \
253                nca.1                        \
254                ncab2clf.1                   \
255                ncakmod.1                    \
256                newform.1                    \
```

```
257                 newgrp.1                        \
258                 news.1                          \
259                 newtask.1                       \
260                 nice.1                          \
261                 nl.1                            \
262                 nm.1                            \
263                 nohup.1                         \
264                 nroff.1                         \
265                 od.1                            \
266                 on.1                            \
267                 optisa.1                        \
268                 pack.1                          \
269                 pagesize.1                      \
270                 pargs.1                         \
271                 passwd.1                        \
272                 paste.1                         \
273                 pathchk.1                       \
274                 pax.1                           \
275                 pfexec.1                        \
276                 pg.1                            \
277                 pgrep.1                         \
278                 pkginfo.1                       \
279                 pkgmk.1                         \
280                 pkgparam.1                      \
281                 pkgproto.1                      \
282                 pkgtrans.1                      \
283                 pktool.1                        \
284                 plabel.1                        \
285                 plgrp.1                         \
286                 plimit.1                        \
287                 pmadvise.1                      \
288                 pmap.1                          \
289                 postio.1                        \
290                 postprint.1                     \
291                 postreverse.1                   \
292                 ppgsz.1                         \
293                 ppriv.1                         \
294                 pr.1                            \
295                 praliases.1                     \
296                 prctl.1                         \
297                 preap.1                         \
298                 prex.1                          \
299                 print.1                         \
300                 printf.1                        \
301                 priocntl.1                      \
302                 proc.1                          \
303                 prof.1                          \
304                 profiles.1                      \
305                 projects.1                      \
306                 ps.1                            \
307                 ptree.1                         \
308                 pvs.1                           \
309                 pwd.1                           \
310                 ranlib.1                        \
311                 rcapstat.1                      \
312                 rcp.1                           \
313                 rdist.1                         \
314                 read.1                          \
315                 readonly.1                      \
316                 refer.1                         \
317                 regcmp.1                        \
318                 renice.1                        \
319                 rev.1                           \
320                 rlogin.1                        \
321                 rm.1                            \
322                 rmformat.1                      \
```



```
323                 rmmount.1                       \
324                 roffbib.1                       \
325                 roles.1                         \
326                 rpcgen.1                        \
327                 rsh.1                           \
328                 runat.1                         \
329                 rup.1                           \
330                 ruptime.1                       \
331                 rusers.1                        \
332                 rwho.1                          \
333                 sar.1                           \
334                 scp.1                           \
335                 script.1                        \
336                 sdiff.1                         \
337                 sed.1                           \
338                 set.1                           \
339                 setfacl.1                       \
340                 setlabel.1                      \
341                 setpgrp.1                       \
342                 sftp.1                          \
343                 shcomp.1                        \
344                 shell_builtins.1                \
345                 shift.1                         \
346                 size.1                          \
347                 sleep.1                         \
348                 smbutil.1                       \
349                 soelim.1                        \
350                 sort.1                          \
351                 sortbib.1                       \
352                 sotruss.1                       \
353                 spell.1                         \
354                 split.1                         \
355                 srchtxt.1                       \
356                 ssh.1                           \
357                 ssh-add.1                       \
358                 ssh-agent.1                     \
359                 ssh-http-proxy-connect.1        \
360                 ssh-keygen.1                    \
361                 ssh-keyscan.1                   \
362                 ssh-socks5-proxy-connect.1      \
363                 strchg.1                        \
364                 strings.1                       \
365                 strip.1                         \
366                 stty.1                          \
367                 sum.1                           \
368                 suspend.1                       \
369                 svcprop.1                       \
370                 svcs.1                          \
371                 symorder.1                      \
372                 sys-suspend.1                   \
373                 tabs.1                          \
374                 tail.1                          \
375                 talk.1                          \
376                 tar.1                           \
377                 tbl.1                           \
378                 tcopy.1                         \
379                 tee.1                           \
380                 telnet.1                        \
381                 test.1                          \
382                 tftp.1                          \
383                 time.1                          \
384                 times.1                         \
385                 timex.1                         \
386                 tip.1                           \
387                 tnfdump.1                       \
388                 tnfxtract.1                     \
```

```
389                touch.1                         \
390                tput.1                          \
391                tr.1                            \
392                trap.1                          \
393                troff.1                         \
394                true.1                          \
395                truss.1                         \
396                tsort.1                         \
397                tty.1                           \
398                type.1                          \
399                typeset.1                       \
400                ul.1                            \
401                umask.1                         \
402                uname.1                         \
403                unifdef.1                       \
404                uniq.1                          \
405                units.1                         \
406                unix2dos.1                      \
407                uptime.1                        \
408                vacation.1                      \
409                vgrind.1                        \
410                volcheck.1                      \
411                volrmmount.1                    \
412                w.1                             \
413                wait.1                          \
414                wc.1                            \
413                whatis.1                        \
415                which.1                         \
416                who.1                           \
417                whocalls.1                      \
418                whois.1                         \
419                write.1                         \
420                xargs.1                         \
421                xgettext.1                      \
422                xstr.1                          \
423                yacc.1                          \
424                yes.1                           \
425                ypcat.1                         \
426                ypmatch.1                       \
427                yppasswd.1                      \
428                ypwhich.1                       \
429                zlogin.1                        \
430                zonename.1

432 MANLINKS=      batch.1         \
433                bg.1            \
434                case.1          \
435                chdir.1         \
436                checkeq.1       \
437                continue.1      \
438                decrypt.1       \
439                dirname.1       \
440                dirs.1          \
441                disable.1       \
442                dumpkeys.1      \
443                edit.1          \
444                errange.1       \
445                errdate.1       \
446                errgid.1        \
447                errint.1        \
448                erritem.1       \
449                errpath.1       \
450                errstr.1        \
451                errtime.1       \
452                erruid.1        \
453                erryorn.1       \
```



```
454                eval.1          \
455                export.1        \
456                false.1         \
457                fc.1            \
458                fg.1            \
459                for.1           \
460                foreach.1       \
461                function.1      \
462                goto.1          \
463                hashcheck.1     \
464                hashmake.1      \
465                hashstat.1      \
466                helpdate.1      \
467                helpgid.1       \
468                helpint.1       \
469                helpitem.1      \
470                helppath.1      \
471                helprange.1     \
472                helpstr.1       \
473                helptime.1      \
474                helpuid.1       \
475                helpyorn.1      \
476                hist.1          \
477                i286.1          \
478                i386.1          \
479                i486.1          \
480                i860.1          \
481                iAPX286.1       \
482                if.1            \
483                intro.1         \
484                jsh.1           \
485                ksh.1           \
486                ldapadd.1       \
487                neqn.1          \
488                notify.1        \
489                onintr.1        \
490                page.1          \
491                pcat.1          \
492                pcred.1         \
493                pdp11.1         \
494                pfcsh.1         \
495                pfiles.1        \
496                pfksh.1         \
497                pflags.1        \
498                pfsh.1          \
499                pkill.1         \
500                pldd.1          \
501                popd.1          \
502                prun.1          \
503                psig.1          \
504                pstack.1        \
505                pstop.1         \
506                ptime.1         \
507                pushd.1         \
508                pwait.1         \
509                pwdx.1          \
510                red.1           \
511                rehash.1        \
512                remote_shell.1  \
513                remsh.1         \
514                repeat.1        \
515                return.1        \
516                rksh.1          \
517                rksh93.1        \
518                rmail.1         \
519                rmdir.1         \
```

```
520                 rmumount.1      \
521                 select.1        \
522                 setenv.1        \
523                 settime.1       \
524                 sh.1            \
525                 snca.1          \
526                 source.1        \
527                 sparc.1         \
528                 spellin.1       \
529                 stop.1          \
530                 strconf.1       \
531                 sun.1           \
532                 switch.1        \
533                 u370.1          \
534                 u3b.1           \
535                 u3b15.1         \
536                 u3b2.1          \
537                 u3b5.1          \
538                 ulimit.1        \
539                 unalias.1       \
540                 uncompress.1    \
541                 unexpand.1      \
542                 unhash.1        \
543                 unlimit.1       \
544                 unpack.1        \
545                 unset.1         \
546                 unsetenv.1      \
547                 until.1         \
548                 valdate.1       \
549                 valgid.1        \
550                 valint.1        \
551                 valpath.1       \
552                 valrange.1      \
553                 valstr.1        \
554                 valtime.1       \
555                 valuid.1        \
556                 valyorn.1       \
557                 vax.1           \
558                 vedit.1         \
559                 whatis.1        \
560                 whence.1        \
561                 while.1         \
562                 zcat.1

564 intro.1         := LINKSRC = Intro.1

566 whatis.1        := LINKSRC = apropos.1

568 unalias.1       := LINKSRC = alias.1

570 batch.1         := LINKSRC = at.1

572 dirname.1       := LINKSRC = basename.1

574 continue.1      := LINKSRC = break.1

576 chdir.1         := LINKSRC = cd.1
577 dirs.1          := LINKSRC = cd.1
578 popd.1          := LINKSRC = cd.1
579 pushd.1         := LINKSRC = cd.1

581 errdate.1       := LINKSRC = ckdate.1
582 helpdate.1     := LINKSRC = ckdate.1
583 valdate.1      := LINKSRC = ckdate.1

585 errgid.1        := LINKSRC = ckgid.1
```



```
586 helpgid.1       := LINKSRC = ckgid.1
587 valgid.1        := LINKSRC = ckgid.1

589 errint.1        := LINKSRC = ckint.1
590 helpint.1       := LINKSRC = ckint.1
591 valint.1        := LINKSRC = ckint.1

593 erritem.1       := LINKSRC = ckitem.1
594 helpitem.1      := LINKSRC = ckitem.1

596 errpath.1       := LINKSRC = ckpath.1
597 helppath.1      := LINKSRC = ckpath.1
598 valpath.1       := LINKSRC = ckpath.1

600 errange.1       := LINKSRC = ckrange.1
601 helprange.1     := LINKSRC = ckrange.1
602 valrange.1      := LINKSRC = ckrange.1

604 errstr.1        := LINKSRC = ckstr.1
605 helpstr.1       := LINKSRC = ckstr.1
606 valstr.1        := LINKSRC = ckstr.1

608 errtime.1       := LINKSRC = cktime.1
609 helptime.1      := LINKSRC = cktime.1
610 valtime.1       := LINKSRC = cktime.1

612 erruid.1        := LINKSRC = ckuid.1
613 helpuid.1       := LINKSRC = ckuid.1
614 valuid.1        := LINKSRC = ckuid.1

616 erryorn.1       := LINKSRC = ckyorn.1
617 helpyorn.1      := LINKSRC = ckyorn.1
618 valyorn.1       := LINKSRC = ckyorn.1

620 uncompress.1    := LINKSRC = compress.1
621 zcat.1          := LINKSRC = compress.1

623 red.1           := LINKSRC = ed.1

625 disable.1       := LINKSRC = enable.1

627 decrypt.1       := LINKSRC = encrypt.1

629 checkeq.1       := LINKSRC = eqn.1
630 neqn.1          := LINKSRC = eqn.1

632 eval.1          := LINKSRC = exec.1
633 source.1        := LINKSRC = exec.1

635 goto.1          := LINKSRC = exit.1
636 return.1        := LINKSRC = exit.1

638 unexpand.1      := LINKSRC = expand.1

640 hashstat.1      := LINKSRC = hash.1
641 rehash.1        := LINKSRC = hash.1
642 unhash.1        := LINKSRC = hash.1

644 fc.1            := LINKSRC = history.1
645 hist.1          := LINKSRC = history.1

647 bg.1            := LINKSRC = jobs.1
648 fg.1            := LINKSRC = jobs.1
649 notify.1        := LINKSRC = jobs.1
650 stop.1          := LINKSRC = jobs.1
```

```
652 jsh.1           := LINKSRC = ksh93.1
653 ksh.1           := LINKSRC = ksh93.1
654 rksh.1          := LINKSRC = ksh93.1
655 rksh93.1        := LINKSRC = ksh93.1
656 sh.1            := LINKSRC = ksh93.1

658 ldapadd.1       := LINKSRC = ldapmodify.1

660 ulimit.1        := LINKSRC = limit.1
661 unlimit.1       := LINKSRC = limit.1

663 dumpkeys.1      := LINKSRC = loadkeys.1

665 i286.1          := LINKSRC = machid.1
666 i386.1          := LINKSRC = machid.1
667 i486.1          := LINKSRC = machid.1
668 i860.1          := LINKSRC = machid.1
669 iAPX286.1       := LINKSRC = machid.1
670 pdp11.1         := LINKSRC = machid.1
671 sparc.1         := LINKSRC = machid.1
672 sun.1           := LINKSRC = machid.1
673 u370.1          := LINKSRC = machid.1
674 u3b.1           := LINKSRC = machid.1
675 u3b15.1         := LINKSRC = machid.1
676 u3b2.1          := LINKSRC = machid.1
677 u3b5.1          := LINKSRC = machid.1
678 vax.1           := LINKSRC = machid.1

680 rmail.1         := LINKSRC = mail.1

682 page.1          := LINKSRC = more.1

684 snca.1          := LINKSRC = nca.1

686 pcat.1          := LINKSRC = pack.1
687 unpack.1        := LINKSRC = pack.1

689 pfcsh.1         := LINKSRC = pfexec.1
690 pfksh.1         := LINKSRC = pfexec.1
691 pfsh.1          := LINKSRC = pfexec.1

693 pkill.1         := LINKSRC = pgrep.1

695 pcred.1         := LINKSRC = proc.1
696 pfiles.1        := LINKSRC = proc.1
697 pflags.1        := LINKSRC = proc.1
698 pldd.1          := LINKSRC = proc.1
699 prun.1          := LINKSRC = proc.1
700 psig.1          := LINKSRC = proc.1
701 pstack.1        := LINKSRC = proc.1
702 pstop.1         := LINKSRC = proc.1
703 ptime.1         := LINKSRC = proc.1
704 pwait.1         := LINKSRC = proc.1
705 pwdx.1          := LINKSRC = proc.1

707 rmdir.1         := LINKSRC = rm.1

709 rmumount.1      := LINKSRC = rmmount.1

711 remote_shell.1  := LINKSRC = rsh.1
712 remsh.1         := LINKSRC = rsh.1

714 export.1        := LINKSRC = set.1
715 setenv.1        := LINKSRC = set.1
716 unset.1         := LINKSRC = set.1
717 unsetenv.1      := LINKSRC = set.1
```



```
719 case.1          := LINKSRC = shell_builtins.1
720 for.1           := LINKSRC = shell_builtins.1
721 foreach.1       := LINKSRC = shell_builtins.1
722 function.1      := LINKSRC = shell_builtins.1
723 if.1            := LINKSRC = shell_builtins.1
724 repeat.1        := LINKSRC = shell_builtins.1
725 select.1        := LINKSRC = shell_builtins.1
726 switch.1        := LINKSRC = shell_builtins.1
727 until.1         := LINKSRC = shell_builtins.1
728 while.1         := LINKSRC = shell_builtins.1

730 hashcheck.1     := LINKSRC = spell.1
731 hashmake.1      := LINKSRC = spell.1
732 spellin.1       := LINKSRC = spell.1

734 strconf.1       := LINKSRC = strchg.1

736 settime.1       := LINKSRC = touch.1

738 onintr.1        := LINKSRC = trap.1

740 false.1         := LINKSRC = true.1

742 whence.1        := LINKSRC = typeset.1

744 # Links to usr/has/man

746 edit.1          := LINKSRC = ../../../has/man/man1has/edit.1has

748 vedit.1         := LINKSRC = ../../../has/man/man1has/vi.1has

750 .KEEP_STATE:

752 include         $(SRC)/man/Makefile.man

754 install:        $(ROOTMANFILES) $(ROOTMANLINKS)
```

```
**********************************************************
    1729 Wed Jul 16 14:05:08 2014
new/usr/src/man/man1/apropos.1
feedback from Hans
mandoc import
**********************************************************
```

```
.\"
.\" This file and its contents are supplied under the terms of the
.\" Common Development and Distribution License ("CDDL"), version 1.0.
.\" You may only use this file in accordance with the terms of version
.\" 1.0 of the CDDL.
.\"
.\" A full copy of the text of the CDDL should have accompanied this
.\" source.  A copy of the CDDL is also available via the Internet at
.\" http://www.illumos.org/license/CDDL.
.\"
.\"
.\" Copyright 2012 Nexenta Systems, Inc. All rights reserved.
.\" Copyright 2014 Garrett D'Amore <garrett@damore.org>
.\"
.Dd Jul 13, 2014
.Dt APROPOS 1
.Os
.Sh NAME
.Nm apropos ,
.Nm whatis
.Nd keyword search in
.Nm whatis
database files
.Sh SYNOPSIS
.Nm
.Op Fl M Ar path
.Op Fl s Ar section
.Ar keyword ...
.Nm whatis
.Op Fl M Ar path
.Op Fl s Ar section
.Ar keyword ...
.Sh DESCRIPTION
The
.Nm
utility searches a set of
.Nm whatis
database files matching each
.Ar keyword .
The
.Nm whatis
utility does the same search but only on complete words. The
.Nm whatis
database files are created using
.Xr man 1
command.
.Bl -tag -width ".Fl d"
.It Fl M Ar path
Force a specific colon separated manual path instead of the default search path.
Overrides the
.Ev MANPATH
environment variable.
.It Fl s Ar section
Restrict search to specified
.Ar section .
.El
.Sh ENVIRONMENT
The following environment variables affect the execution of
.Nm :
.Bl -tag -width ".Ev MANPATH , PATH"
```



```
.It Ev MANPATH , PATH
Used to find the location of the
.Nm whatis
database files.
.El
.Sh DIAGNOSTICS
The
.Nm
utility exits 0 if a keyword matched and 1 if no keywords are matched or no
.Nm whatis
databases are found.
.Sh INTERFACE STABILITY
.Nm Committed .
.Sh CODE SET INDEPENDENCE
Enabled.
.Sh SEE ALSO
.Xr man 1 ,
.Xr mandoc 1
'\" te
.\"  Copyright (c) 1996, Sun Microsystems, Inc.  All Rights Reserved
.\" The contents of this file are subject to the terms of the Common Development
.\" You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE or http:
.\" When distributing Covered Code, include this CDDL HEADER in each file and in
.TH APROPOS 1 "Dec 20, 1996"
.SH NAME
apropos \- locate commands by keyword lookup
.SH SYNOPSIS
.LP
.nf
\fBapropos\fR \fIkeyword\fR...
.fi

.SH DESCRIPTION
.sp
.LP
The \fBapropos\fR utility displays the man page name, section number, and a
short description for each man page whose \fBNAME\fR line contains
\fIkeyword\fR. This information is contained in the \fB/usr/share/man/windex\fR
database created by \fBcatman\fR(1M). If \fBcatman\fR(1M) was not run, or was
run with the \fB-n\fR option, \fBapropos\fR fails. Each word is considered
separately and the case of letters is ignored. Words which are part of other
words are considered; for example, when looking for 'compile', \fBapropos\fR
finds all instances of 'compiler' also.
.sp
.LP
\fBapropos\fR is actually just the \fB-k\fR option to the \fBman\fR(1) command.
.SH EXAMPLES
.LP
\fBExample 1 \fRTo find a man page whose NAME line contains a keyword
.sp
.LP
Try

.sp
.in +2
.nf
example% \fBapropos password\fR
.fi
.in -2
.sp

.sp
.LP
and

.sp
```

```
.in +2
.nf
example% \fBapropos editor\fR
.fi
.in -2
.sp

.sp
.LP
If the line starts ‘\fIfilename\fR(\fIsection\fR) .\|.\|.’ you can run

.sp
.in +2
.nf
man -s \fIsection filename\fR
.fi
.in -2
.sp

.sp
.LP
to display the man page for \fIfilename\fR.

.LP
\fBExample 2 \fRTo find the man page for the subroutine \fBprintf()\fR
.sp
.LP
Try

.sp
.in +2
.nf
example% \fBapropos format\fR
.fi
.in -2
.sp

.sp
.LP
and then

.sp
.in +2
.nf
example% \fBman -s 3s printf\fR
.fi
.in -2
.sp

.sp
.LP
to get the manual page on the subroutine \fBprintf()\fR.

.SH FILES
.sp
.ne 2
.na
\fB\fB/usr/share/man/windex\fR \fR
.ad
.RS 26n
table of contents and keyword database
.RE

.SH ATTRIBUTES
.sp
.LP
```



```
See \fBattributes\fR(5) for descriptions of the following attributes:
.sp

.sp
.TS
box;
c | c
l | l .
ATTRIBUTE TYPE  ATTRIBUTE VALUE
_
CSI     Enabled
.TE

.SH SEE ALSO
.sp
.LP
\fBman\fR(1), \fBwhatis\fR(1), \fBcatman\fR(1M), \fBattributes\fR(5)
.SH DIAGNOSTICS
.sp
.ne 2
.na
\fB\f(CW/usr/share/man/windex: No such file or directory\fR\fR
.ad
.sp .6
.RS 4n
This database does not exist. \fBcatman\fR(1M) must be run to create it.
.RE
```

```
**********************************************************
    3099 Wed Jul 16 14:05:08 2014
new/usr/src/man/man1/man.1
feedback from Hans
mandoc import
**********************************************************
```

```
.\"
.\" This file and its contents are supplied under the terms of the
.\" Common Development and Distribution License ("CDDL"), version 1.0.
.\" You may only use this file in accordance with the terms of version
.\" 1.0 of the CDDL.
.\"
.\" A full copy of the text of the CDDL should have accompanied this
.\" source.  A copy of the CDDL is also available via the Internet at
.\" http://www.illumos.org/license/CDDL.
.\"
.\"
.\" Copyright 2012 Nexenta Systems, Inc. All rights reserved.
.\" Copyright 2014 Garrett D'Amore <garrett@damore.org>
.\"
.Dd October 18, 2012
.Dt MAN 1
.Os
.Sh NAME
.Nm man
.Nd find and display reference manual pages
.Sh SYNOPSIS
.Nm
.Op Fl alptw
.Op Fl M Ar manpath
.Op Fl s Ar mansect
.Ar page ...
.Nm
.Op Fl s Ar mansect
.Fl f
.Ar keyword ...
.Nm
.Op Fl s Ar mansect
.Fl k
.Ar keyword ...
.Sh DESCRIPTION
The
.Nm
utility finds and displays reference manual pages.
.Pp
Options that
.Nm
understands:
.Bl -tag -width indent
.It Fl M Ar manpath
Forces a specific colon separated manual path instead of the default
search path.
Overrides the
.Ev MANPATH
environment variable.
.It Fl a
Display all manual pages instead of just the first found for each
.Ar page
argument.
.It Fl f
Emulate
.Xr whatis 1 .
.It Fl k
Emulate
.Xr apropos 1 .
.It Fl l
```



```
Display the location of the manual page instead of the contents of
the manual page.
.It Fl p
Output current path used for searching.
.It Fl s Ar mansect
Restrict manual sections searched to the specified colon delimited list.
.It Fl t
Send the content formatted as PostScript to the default printer.
.It Fl w
Create
.Nm whatis
databases used by
.Xr apropos 1
and
.Xr whatis 1 .
.El
.Sh ENVIRONMENT
The following environment variables affect the execution of
.Nm :
.Bl -tag -width ".Ev MANPATH"
.It Ev LC_ALL , LC_CTYPE , LANG
Used to find locale-specific manual pages.
.It Ev MANPATH
Used to find the location of the manual files.
Corresponds to the
.Fl M
option.
.It Ev MANWIDTH
Defines the width of output. If set to
.Dq Li tty ,
and output is to a terminal, full width of terminal is used.
.It Ev PAGER
Program used to display files. If unset,
.Dq Li "less -ins"
is used.
.It Ev PATH
Used to find location of manual files if
.Ev MANPATH
and
.Fl M
are not specified.
.El
.Sh FILES
.Bl -tag -width indent -compact
.It Pa man.cf
Per-manpath configuration settings.  The file is formatted as follows:
.Bd -literal -offset indent
MANSECT=\fIsection\fR[,\fIsection\fR]...
.Ed
.Pp
Each section consists of a section in the reference manual.  The file
may also contain comment blank lines or lines consisting of comments, where
the first character in the line is '#'.  Both blank lines and comment lines are
ignored.
.El
.Sh CODE SET INDEPENDENCE
Enabled.
.Sh INTERFACE STABILITY
The
.Nm
utility is
.Nm Standard ,
as is the
.Fl k
option. The other options are
.Nm Committed .
```

```
.Sh SEE ALSO
.Xr apropos 1 ,
.Xr intro 1 ,
.Xr mandoc 1 ,
.Xr whatis 1 ,
.Xr man 5 ,
.Xr mdoc 5 ,
.Xr standards 5
.Sh NOTES
Some pages may contain information which cannot be properly displayed on
all terminals.  In such cases, some information may be lost.
'\" te
.\" Copyright (c) 2008, Sun Microsystems, Inc. All Rights Reserved.
.\" Copyright (c) 1980 Regents of the University of California. The Berkeley sof
.TH MAN 1 "May 8, 2008"
.SH NAME
man \- find and display reference manual pages
.SH SYNOPSIS
.LP
.nf
\fBman\fR [\fB-\fR] [\fB-adFlrt\fR] [\fB-M\fR \fIpath\fR] [\fB-T\fR \fImacro-pac
.fi

.LP
.nf
\fBman\fR [\fB-M\fR \fIpath\fR] \fB-k\fR \fIkeyword\fR...
.fi

.LP
.nf
\fBman\fR [\fB-M\fR \fIpath\fR] \fB-f\fR \fIfile\fR...
.fi

.SH DESCRIPTION
.sp
.LP
The \fBman\fR command displays information from the reference manuals. It
displays complete manual pages that you select by \fIname\fR, or one-line
summaries selected either by \fIkeyword\fR (\fB-k\fR), or by the name of an
associated file (\fB-f\fR). If no manual page is located, \fBman\fR prints an
error message.
.SS "Source Format"
.sp
.LP
Reference Manual pages are marked up with either \fBnroff\fR (see
\fBnroff\fR(1)) or \fBSGML\fR (Standard Generalized Markup Language) tags (see
\fBsgml\fR(5)). The \fBman\fR command recognizes the type of markup and
processes the file accordingly. The various source files are kept in separate
directories depending on the type of markup.
.SS "Location of Manual Pages"
.sp
.LP
The online Reference Manual page directories are conventionally located in
\fB/usr/share/man\fR. The nroff sources are located in the
\fB/usr/share/man/man\fR* directories. The \fBSGML\fR sources are located in
the \fB/usr/share/man/sman\fR* directories. Each directory corresponds to a
section of the manual. Since these directories are optionally installed, they
might not reside on your host. You might have to mount \fB/usr/share/man\fR
from a host on which they do reside.
.sp
.LP
If there are preformatted, up-to-date versions in the corresponding \fBcat\fR*
or \fBfmt\fR* directories, \fBman\fR simply displays or prints those versions.
If the preformatted version of interest is out of date or missing, \fBman\fR
reformats it prior to display and stores the preformatted version if \fBcat\fR*
or \fBfmt\fR* is writable. The \fBwindex\fR database is not updated. See
```



```
\fBcatman\fR(1M). If directories for the preformatted versions are not
provided, \fBman\fR reformats a page whenever it is requested. \fBman\fR uses a
temporary file to store the formatted text during display.
.sp
.LP
If the standard output is not a terminal, or if the '\fB-\fR' flag is given,
\fBman\fR pipes its output through \fBcat\fR(1). Otherwise, \fBman\fR pipes its
output through \fBmore\fR(1) to handle paging and underlining on the screen.
.SH OPTIONS
.sp
.LP
The following options are supported:
.sp
.ne 2
.na
\fB\fB-a\fR\fR
.ad
.RS 20n
Shows all manual pages matching \fIname\fR within the \fBMANPATH\fR search
path. Manual pages are displayed in the order found.
.RE

.sp
.ne 2
.na
\fB\fB-d\fR\fR
.ad
.RS 20n
Debugs. Displays what a section-specifier evaluates to, method used for
searching, and paths searched by \fBman\fR.
.RE

.sp
.ne 2
.na
\fB\fB-f\fR \fIfile ...\fR\fR
.ad
.RS 20n
\fBman\fR attempts to locate manual pages related to any of the given
\fIfile\fRs. It strips the leading path name components from each \fIfile\fR,
and then prints one-line summaries containing the resulting basename or names.
This option also uses the \fBwindex\fR database.
.RE

.sp
.ne 2
.na
\fB\fB-F\fR\fR
.ad
.RS 20n
Forces \fBman\fR to search all directories specified by \fBMANPATH\fR or the
\fBman.cf\fR file, rather than using the \fBwindex\fR lookup database. This
option is useful if the database is not up to date and it has been made the
default behavior of the \fBman\fR command. The option therefore does not have
to be invoked and is documented here for reference only.
.RE

.sp
.ne 2
.na
\fB\fB-k\fR \fIkeyword ...\fR\fR
.ad
.RS 20n
Prints out one-line summaries from the \fBwindex\fR database (table of
contents) that contain any of the given \fIkeyword\fRs. The \fBwindex\fR
database is created using \fBcatman\fR(1M).
```

```
.RE

.sp
.ne 2
.na
\fB\fB-l\fR\fR
.ad
.RS 20n
Lists all manual pages found matching \fIname\fR within the search path.
.RE

.sp
.ne 2
.na
\fB\fB-M\fR \fIpath\fR\fR
.ad
.RS 20n
Specifies an alternate search path for manual pages. \fIpath\fR is a
colon-separated list of directories that contain manual page directory
subtrees. For example, if \fIpath\fR is \fB/usr/share/man:/usr/local/man\fR,
\fBman\fR searches for \fIname\fR in the standard location, and then
\fB/usr/local/man\fR. When used with the \fB-k\fR or \fB-f\fR options, the
\fB-M\fR option must appear first. Each directory in the \fIpath\fR is assumed
to contain subdirectories of the form \fBman\fR* or \fBsman\fR* , one for each
section. This option overrides the \fBMANPATH\fR environment variable.
.RE

.sp
.ne 2
.na
\fB\fB-r\fR\fR
.ad
.RS 20n
Reformats the manual page, but does not display it. This replaces the \fBman\fR
\fB-\fR \fB-t\fR \fIname\fR combination.
.RE

.sp
.ne 2
.na
\fB\fB-s\fR \fIsection ...\fR\fR
.ad
.RS 20n
Specifies sections of the manual for \fBman\fR to search. The directories
searched for \fIname\fR are limited to those specified by \fIsection\fR.
\fIsection\fR can be a numerical digit, perhaps followed by one or more letters
to match the desired section of the manual, for example, "\fB3libucb\fR". Also,
\fIsection\fR can be a word, for example, \fBlocal\fR, \fBnew\fR, \fBold\fR,
\fBpublic\fR. \fIsection\fR can also be a letter. To specify multiple sections,
separate each section with a comma. This option overrides the \fBMANPATH\fR
environment variable and the \fBman.cf\fR file. See \fBSearch\fR \fBPath\fR
below for an explanation of how \fBman\fR conducts its search.
.RE

.sp
.ne 2
.na
\fB\fB-t\fR\fR
.ad
.RS 20n
\fBman\fR arranges for the specified manual pages to be \fBtroff\fRed to a
suitable raster output device (see \fBtroff\fR(1)). If both the \fB-\fR and
\fB-t\fR flags are given, \fBman\fR updates the \fBtroff\fRed versions of each
named \fIname\fR (if necessary), but does not display them.
.RE
```



```
.sp
.ne 2
.na
\fB\fB-T\fR \fImacro-package\fR\fR
.ad
.RS 20n
Formats manual pages using \fImacro-package\fR rather than the standard
\fB-man\fR macros defined in \fB/usr/share/lib/tmac/an\fR. See \fBSearch
Path\fR under USAGE for a complete explanation of the default search path
order.
.RE

.SH OPERANDS
.sp
.LP
The following operand is supported:
.sp
.ne 2
.na
\fB\fIname\fR\fR
.ad
.RS 8n
The name of a standard utility or a keyword.
.RE

.SH USAGE
.sp
.LP
The usage of \fBman\fR is described below:
.SS "Manual Page Sections"
.sp
.LP
Entries in the reference manuals are organized into \fIsection\fRs. A section
name consists of a major section name, typically a single digit, optionally
followed by a subsection name, typically one or more letters. An unadorned
major section name, for example, "\fB9\fR", does not act as an abbreviation for
the subsections of that name, such as "\fB9e\fR", "\fB9f\fR", or "\fB9s\fR".
That is, each subsection must be searched separately by \fBman\fR \fB-s\fR.
Each section contains descriptions apropos to a particular reference category,
with subsections refining these distinctions. See the \fBintro\fR manual pages
for an explanation of the classification used in this release.
.SS "Search Path"
.sp
.LP
Before searching for a given \fIname\fR, \fBman\fR constructs a list of
candidate directories and sections. \fBman\fR searches for \fIname\fR in the
directories specified by the \fBMANPATH\fR environment variable.
.sp
.LP
In the absence of \fBMANPATH\fR, \fBman\fR constructs its search path based
upon the \fBPATH\fR environment variable, primarily by substituting \fBman\fR
for the last component of the \fBPATH\fR element. Special provisions are added
to account for unique characteristics of directories such as \fB/sbin\fR,
\fB/usr/ucb\fR, \fB/usr/xpg4/bin\fR, and others. If the file argument contains
a \fB/\fR character, the \fIdirname\fR portion of the argument is used in place
of \fBPATH\fR elements to construct the search path.
.sp
.LP
Within the manual page directories, \fBman\fR confines its search to the
sections specified in the following order:
.RS +4
.TP
.ie t \(bu
.el o
\fIsection\fRs specified on the command line with the \fB-s\fR option
.RE
```

```
.RS +4
.TP
.ie t \(bu
.el o
\fIsection\fRs embedded in the \fBMANPATH\fR environment variable
.RE
.RS +4
.TP
.ie t \(bu
.el o
\fIsection\fRs specified in the \fBman.cf\fR file for each directory specified
in the \fBMANPATH\fR environment variable
.RE
.sp
.LP
If none of the above exist, \fBman\fR searches each directory in the manual
page path, and displays the first matching manual page found.
.sp
.LP
The \fBman.cf\fR file has the following format:
.sp
.in +2
.nf
MANSECTS=\fIsection\fR[,\fIsection\fR]...
.fi
.in -2
.sp

.sp
.LP
Lines beginning with '\fB#\fR' and blank lines are considered comments, and are
ignored. Each directory specified in \fBMANPATH\fR can contain a manual page
configuration file, specifying the default search order for that directory.
.SH FORMATTING MANUAL PAGES
.sp
.LP
Manual pages are marked up in \fBnroff\fR(1) or \fBsgml\fR(5). Nroff manual
pages are processed by \fBnroff\fR(1) or \fBtroff\fR(1) with the \fB-man\fR
macro package. Please refer to \fBman\fR(5) for information on macro usage.
\fBSGML\fR\(emtagged manual pages are processed by an \fBSGML\fR parser and
passed to the formatter.
.SS "Preprocessing Nroff Manual Pages"
.sp
.LP
When formatting an nroff manual page, \fBman\fR examines the first line to
determine whether it requires special processing. If the first line is a string
of the form:
.sp
.in +2
.nf
\&'\e" \fIX\fR
.fi
.in -2
.sp

.sp
.LP
where \fIX\fR is separated from the '\fB"\fR' by a single SPACE and consists of
any combination of characters in the following list, \fBman\fR pipes its input
to \fBtroff\fR(1) or \fBnroff\fR(1) through the corresponding preprocessors.
.sp
.ne 2
.na
\fB\fBe\fR\fR
.ad
.RS 5n
```



```
\fBeqn\fR(1), or \fBneqn\fR for \fBnroff\fR
.RE

.sp
.ne 2
.na
\fB\fBr\fR\fR
.ad
.RS 5n
\fBrefer\fR(1)
.RE

.sp
.ne 2
.na
\fB\fBt\fR\fR
.ad
.RS 5n
\fBtbl\fR(1)
.RE

.sp
.ne 2
.na
\fB\fBv\fR\fR
.ad
.RS 5n
\fBvgrind\fR(1)
.RE

.sp
.LP
If \fBeqn\fR or \fBneqn\fR is invoked, it automatically reads the file
\fB/usr/pub/eqnchar\fR (see \fBeqnchar\fR(5)). If \fBnroff\fR(1) is invoked,
\fBcol\fR(1) is automatically used.
.SS "Referring to Other nroff Manual Pages"
.sp
.LP
If the first line of the nroff manual page is a reference to another manual
page entry fitting the pattern:
.sp
.in +2
.nf
\&.so man*/\fIsourcefile\fR
.fi
.in -2
.sp

.sp
.LP
\fBman\fR processes the indicated file in place of the current one. The
reference must be expressed as a path name relative to the root of the manual
page directory subtree.
.sp
.LP
When the second or any subsequent line starts with \fB\&.so\fR, \fBman\fR
ignores it; \fBtroff\fR(1) or \fBnroff\fR(1) processes the request in the usual
manner.
.SS "Processing SGML Manual Pages"
.sp
.LP
Manual pages are identified as being marked up in SGML by the presence of the
string \fB<!DOCTYPE\fR\&. If the file also contains the string
\fBSHADOW_PAGE\fR, the file refers to another manual page for the content. The
reference is made with a file entity reference to the manual page that contains
the text. This is similar to the \fB\&.so\fR mechanism used in the nroff
```

```
386 formatted man pages.
387 .SH ENVIRONMENT VARIABLES
388 .sp
389 .LP
390 See \fBenviron\fR(5) for descriptions of the following environment variables
391 that affect the execution of \fBman\fR: \fBLANG\fR, \fBLC_ALL\fR,
392 \fBLC_CTYPE\fR, \fBLC_MESSAGES\fR, and \fBNLSPATH\fR.
393 .sp
394 .ne 2
395 .na
396 \fB\fBMANPATH\fR\fR
397 .ad
398 .RS 11n
399 A colon-separated list of directories; each directory can be followed by a
400 comma-separated list of sections. If set, its value overrides
401 \fB/usr/share/man\fR as the default directory search path, and the \fBman.cf\fR
402 file as the default section search path. The \fB-M\fR and \fB-s\fR flags, in
403 turn, override these values.)
404 .RE

406 .sp
407 .ne 2
408 .na
409 \fB\fBPAGER\fR\fR
410 .ad
411 .RS 11n
412 A program to use for interactively delivering \fBman\fR's output to the screen.
413 If not set, '\fBmore\fR \fB-s\fR' is used. See \fBmore\fR(1).
414 .RE

416 .sp
417 .ne 2
418 .na
419 \fB\fBTCAT\fR\fR
420 .ad
421 .RS 11n
422 The name of the program to use to display \fBtroff\fRed manual pages.
423 .RE

425 .sp
426 .ne 2
427 .na
428 \fB\fBTROFF\fR\fR
429 .ad
430 .RS 11n
431 The name of the formatter to use when the \fB-t\fR flag is given. If not set,
432 \fBtroff\fR(1) is used.
433 .RE

435 .SH EXAMPLES
436 .LP
437 \fBExample 1 \fRCreating a PostScript Version of a man page
438 .sp
439 .LP
440 The following example creates the \fBpipe\fR(2) man page in postscript for csh,
441 tcsh, ksh and sh users:

443 .sp
444 .in +2
445 .nf
446         % env TCAT=/usr/lib/lp/postscript/dpost man -t -s 2 pipe > pipe.ps
447 .fi
448 .in -2
449 .sp

451 .sp
```



```
452 .LP
453 This is an alternative to using \fBman\fR \fB-t\fR, which sends the man page to
454 the default printer, if the user wants a postscript file version of the man
455 page.

457 .LP
458 \fBExample 2 \fRCreating a Text Version of a man page
459 .sp
460 .LP
461 The following example creates the \fBpipe\fR(2) man page in ascii text:

463 .sp
464 .in +2
465 .nf
466 man pipe.2 | col -x -b > pipe.text
467 .fi
468 .in -2
469 .sp

471 .sp
472 .LP
473 This is an alternative to using \fBman\fR \fB-t\fR, which sends the man page to
474 the default printer, if the user wants a text file version of the man page.

476 .SH EXIT STATUS
477 .sp
478 .LP
479 The following exit values are returned:
480 .sp
481 .ne 2
482 .na
483 \fB\fB0\fR\fR
484 .ad
485 .RS 6n
486 Successful completion.
487 .RE

489 .sp
490 .ne 2
491 .na
492 \fB\fB>0\fR\fR
493 .ad
494 .RS 6n
495 An error occurred.
496 .RE

498 .SH FILES
499 .sp
500 .ne 2
501 .na
502 \fB\fB/usr/share/man\fR\fR
503 .ad
504 .sp .6
505 .RS 4n
506 Root of the standard manual page directory subtree
507 .RE

509 .sp
510 .ne 2
511 .na
512 \fB\fB/usr/share/man/man?/*\fR\fR
513 .ad
514 .sp .6
515 .RS 4n
516 Unformatted nroff manual entries
517 .RE
```

```
.sp
.ne 2
.na
\fB\fB/usr/share/man/sman?/*\fR\fR
.ad
.sp .6
.RS 4n
Unformatted \fBSGML\fR manual entries
.RE

.sp
.ne 2
.na
\fB\fB/usr/share/man/cat?/*\fR\fR
.ad
.sp .6
.RS 4n
\fBnroff\fRed manual entries
.RE

.sp
.ne 2
.na
\fB\fB/usr/share/man/fmt?/*\fR\fR
.ad
.sp .6
.RS 4n
\fBtroff\fRed manual entries
.RE

.sp
.ne 2
.na
\fB\fB/usr/share/man/windex\fR\fR
.ad
.sp .6
.RS 4n
Table of contents and keyword database
.RE

.sp
.ne 2
.na
\fB\fB/usr/share/lib/tmac/an\fR\fR
.ad
.sp .6
.RS 4n
Standard \fB-man\fR macro package
.RE

.sp
.ne 2
.na
\fB\fB/usr/share/lib/sgml/locale/C/dtd/*\fR\fR
.ad
.sp .6
.RS 4n
\fBSGML\fR document type definition files
.RE

.sp
.ne 2
.na
\fB\fB/usr/share/lib/sgml/locale/C/solbook/*\fR\fR
.ad
```



```
.sp .6
.RS 4n
\fBSGML\fR style sheet and entity definitions directories
.RE

.sp
.ne 2
.na
\fB\fB/usr/share/lib/pub/eqnchar\fR\fR
.ad
.sp .6
.RS 4n
Standard definitions for \fBeqn\fR and \fBneqn\fR
.RE

.sp
.ne 2
.na
\fB\fBman.cf\fR\fR
.ad
.sp .6
.RS 4n
Default search order by section
.RE

.SH ATTRIBUTES
.sp
.LP
See \fBattributes\fR(5) for descriptions of the following attributes:
.sp

.sp
.TS
box;
c | c
l | l .
ATTRIBUTE TYPE  ATTRIBUTE VALUE
_
CSI     Enabled, see \fBNOTES\fR.
_
Interface Stability     Committed
_
Standard        See \fBstandards\fR(5).
.TE

.SH SEE ALSO
.sp
.LP
\fBapropos\fR(1), \fBcat\fR(1), \fBcol\fR(1), \fBdpost\fR(1), \fBeqn\fR(1),
\fBmore\fR(1), \fBnroff\fR(1), \fBrefer\fR(1), \fBtbl\fR(1), \fBtroff\fR(1),
\fBvgrind\fR(1), \fBwhatis\fR(1), \fBcatman\fR(1M), \fBattributes\fR(5),
\fBenviron\fR(5), \fBeqnchar\fR(5), \fBman\fR(5), \fBsgml\fR(5),
\fBstandards\fR(5)
.SH NOTES
.sp
.LP
The \fB-f\fR and \fB-k\fR options use the \fBwindex\fR database, which is
created by \fBcatman\fR(1M).
.sp
.LP
The \fBman\fR command is CSI-capable. However, some utilities invoked by the
\fBman\fR command, namely, \fBtroff\fR, \fBeqn\fR, \fBneqn\fR, \fBrefer\fR,
\fBtbl\fR, and \fBvgrind\fR, are not verified to be CSI-capable. Because of
this, the man command with the \fB-t\fR option can not handle non-EUC data.
Also, using the \fBman\fR command to display man pages that require special
processing through \fBeqn\fR, \fBneqn\fR, \fBrefer\fR, \fBtbl\fR, or
```

```
650 \fBvgrind\fR can not be CSI-capable.
651 .SH BUGS
652 .sp
653 .LP
654 The manual is supposed to be reproducible either on a phototypesetter or on an
655 \fBASCII\fR terminal. However, on a terminal some information (indicated by
656 font changes, for instance) is lost.
657 .sp
658 .LP
659 Some dumb terminals cannot process the vertical motions produced by the \fBe\fR
660 (see \fBeqn\fR(1)) preprocessing flag. To prevent garbled output on these
661 terminals, when you use \fBe\fR, also use \fBt\fR, to invoke \fBcol\fR(1)
662 implicitly. This workaround has the disadvantage of eliminating superscripts
663 and subscripts, even on those terminals that can display them. Control-q clears
664 a terminal that gets confused by \fBeqn\fR(1) output.
```

```
**********************************************************
   14743 Wed Jul 16 14:05:08 2014
new/usr/src/man/man1/mandoc.1
feedback from Hans
mandoc import
**********************************************************
```

```
.\"
.\" Permission to use, copy, modify, and distribute this software for any
.\" purpose with or without fee is hereby granted, provided that the above
.\" copyright notice and this permission notice appear in all copies.
.\"
.\" THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES
.\" WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF
.\" MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR
.\" ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES
.\" WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN
.\" ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF
.\" OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
.\"
.\"
.\" Copyright (c) 2009, 2010, 2011 Kristaps Dzonsons <kristaps@bsd.lv>
.\" Copyright 2012 Nexenta Systems, Inc. All rights reserved.
.\" Copyright 2014 Garrett D'Amore <garrett@damore.org>
.\"
.Dd Jul 16, 2014
.Dt MANDOC 1
.Os
.Sh NAME
.Nm mandoc
.Nd format and display UNIX manuals
.Sh SYNOPSIS
.Nm mandoc
.Op Fl V
.Op Fl m Ns Ar format
.Op Fl O Ns Ar option
.Op Fl T Ns Ar output
.Op Fl W Ns Ar level
.Op Ar
.Sh DESCRIPTION
The
.Nm
utility formats
.Ux
manual pages for display.
.Pp
By default,
.Nm
reads
.Xr mdoc 5
or
.Xr man 5
text from stdin, implying
.Fl m Ns Cm andoc ,
and produces
.Fl T Ns Cm ascii
output.
.Pp
The arguments are as follows:
.Bl -tag -width Ds
.It Fl m Ns Ar format
Input format.
See
.Sx Input Formats
for available formats.
Defaults to
.Fl m Ns Cm andoc .
```



```
.It Fl O Ns Ar option
Comma-separated output options.
.It Fl T Ns Ar output
Output format.
See
.Sx Output Formats
for available formats.
Defaults to
.Fl T Ns Cm ascii .
.It Fl V
Print version and exit.
.It Fl W Ns Ar level
Specify the minimum message
.Ar level
to be reported on the standard error output and to affect the exit status.
The
.Ar level
can be
.Cm warning ,
.Cm error ,
or
.Cm fatal .
The default is
.Fl W Ns Cm fatal ;
.Fl W Ns Cm all
is an alias for
.Fl W Ns Cm warning .
See
.Sx EXIT STATUS
and
.Sx DIAGNOSTICS
for details.
.Pp
The special option
.Fl W Ns Cm stop
tells
.Nm
to exit after parsing a file that causes warnings or errors of at least
the requested level.
No formatted output will be produced from that file.
If both a
.Ar level
and
.Cm stop
are requested, they can be joined with a comma, for example
.Fl W Ns Cm error , Ns Cm stop .
.It Ar file
Read input from zero or more files.
If unspecified, reads from stdin.
If multiple files are specified,
.Nm
will halt with the first failed parse.
.El
.Ss Input Formats
The
.Nm
utility accepts
.Xr mdoc 5
and
.Xr man 5
input with
.Fl m Ns Cm doc
and
.Fl m Ns Cm an ,
respectively.
The
```

```
.Xr mdoc 5
format is
.Em strongly
recommended;
.Xr man 5
should only be used for legacy manuals.
.Pp
A third option,
.Fl m Ns Cm andoc ,
which is also the default, determines encoding on-the-fly: if the first
non-comment macro is
.Sq \&Dd
or
.Sq \&Dt ,
the
.Xr mdoc 5
parser is used; otherwise, the
.Xr man 5
parser is used.
.Pp
If multiple
files are specified with
.Fl m Ns Cm andoc ,
each has its file-type determined this way.
If multiple files are
specified and
.Fl m Ns Cm doc
or
.Fl m Ns Cm an
is specified, then this format is used exclusively.
.Ss Output Formats
The
.Nm
utility accepts the following
.Fl T
arguments, which correspond to output modes:
.Bl -tag -width "-Tlocale"
.It Fl T Ns Cm ascii
Produce 7-bit ASCII output.
This is the default.
See
.Sx ASCII Output .
.It Fl T Ns Cm html
Produce strict CSS1/HTML-4.01 output.
See
.Sx HTML Output .
.It Fl T Ns Cm lint
Parse only: produce no output.
Implies
.Fl W Ns Cm warning .
.It Fl T Ns Cm locale
Encode output using the current locale.
See
.Sx Locale Output .
.It Fl T Ns Cm man
Produce
.Xr man 5
format output.
See
.Sx Man Output .
.It Fl T Ns Cm pdf
Produce PDF output.
See
.Sx PDF Output .
.It Fl T Ns Cm ps
Produce PostScript output.
```



```
See
.Sx PostScript Output .
.It Fl T Ns Cm tree
Produce an indented parse tree.
.It Fl T Ns Cm utf8
Encode output in the UTF\-8 multi-byte format.
See
.Sx UTF\-8 Output .
.It Fl T Ns Cm xhtml
Produce strict CSS1/XHTML-1.0 output.
See
.Sx XHTML Output .
.El
.Pp
If multiple input files are specified, these will be processed by the
corresponding filter in-order.
.Ss ASCII Output
Output produced by
.Fl T Ns Cm ascii ,
which is the default, is rendered in standard 7-bit ASCII documented in
.Xr ascii 5 .
.Pp
Font styles are applied by using back-spaced encoding such that an
underlined character
.Sq c
is rendered as
.Sq _ Ns \e[bs] Ns c ,
where
.Sq \e[bs]
is the back-space character number 8.
Emboldened characters are rendered as
.Sq c Ns \e[bs] Ns c .
.Pp
The special characters documented in
.Xr mandoc_char 5
are rendered best-effort in an ASCII equivalent.
If no equivalent is found,
.Sq \&?
is used instead.
.Pp
Output width is limited to 78 visible columns unless literal input lines
exceed this limit.
.Pp
The following
.Fl O
arguments are accepted:
.Bl -tag -width Ds
.It Cm indent Ns = Ns Ar indent
The left margin for normal text is set to
.Ar indent
blank characters instead of the default of five for
.Xr mdoc 5
and seven for
.Xr man 5 .
Increasing this is not recommended; it may result in degraded formatting,
for example overfull lines or ugly line breaks.
.It Cm width Ns = Ns Ar width
The output width is set to
.Ar width ,
which will normalise to \(>=60.
.El
.Ss HTML Output
Output produced by
.Fl T Ns Cm html
conforms to HTML-4.01 strict.
.Pp
```

```
The
.Pa example.style.css
file documents style-sheet classes available for customising output.
If a style-sheet is not specified with
.Fl O Ns Ar style ,
.Fl T Ns Cm html
defaults to simple output readable in any graphical or text-based web
browser.
.Pp
Special characters are rendered in decimal-encoded UTF\-8.
.Pp
The following
.Fl O
arguments are accepted:
.Bl -tag -width Ds
.It Cm fragment
Omit the
.Aq !DOCTYPE
declaration and the
.Aq html ,
.Aq head ,
and
.Aq body
elements and only emit the subtree below the
.Aq body
element.
The
.Cm style
argument will be ignored.
This is useful when embedding manual content within existing documents.
.It Cm includes Ns = Ns Ar fmt
The string
.Ar fmt ,
for example,
.Ar ../src/%I.html ,
is used as a template for linked header files (usually via the
.Sq \&In
macro).
Instances of
.Sq \&%I
are replaced with the include filename.
The default is not to present a
hyperlink.
.It Cm man Ns = Ns Ar fmt
The string
.Ar fmt ,
for example,
.Ar ../html%S/%N.%S.html ,
is used as a template for linked manuals (usually via the
.Sq \&Xr
macro).
Instances of
.Sq \&%N
and
.Sq %S
are replaced with the linked manual’s name and section, respectively.
If no section is included, section 1 is assumed.
The default is not to
present a hyperlink.
.It Cm style Ns = Ns Ar style.css
The file
.Ar style.css
is used for an external style-sheet.
This must be a valid absolute or
relative URI.
.El
```



```
.Ss Locale Output
Locale-depending output encoding is triggered with
.Fl T Ns Cm locale .
This option is not available on all systems: systems without locale
support, or those whose internal representation is not natively UCS-4,
will fall back to
.Fl T Ns Cm ascii .
See
.Sx ASCII Output
for font style specification and available command-line arguments.
.Ss Man Output
Translate input format into
.Xr man 5
output format.
This is useful for distributing manual sources to legacy systems
lacking
.Xr mdoc 5
formatters.
.Pp
If
.Xr mdoc 5
is passed as input, it is translated into
.Xr man 5 .
If the input format is
.Xr man 5 ,
the input is copied to the output, expanding any
.Xr mandoc_roff 5
.Sq so
requests.
The parser is also run, and as usual, the
.Fl W
level controls which
.Sx DIAGNOSTICS
are displayed before copying the input to the output.
.Ss PDF Output
PDF-1.1 output may be generated by
.Fl T Ns Cm pdf .
See
.Sx PostScript Output
for
.Fl O
arguments and defaults.
.Ss PostScript Output
PostScript
.Qq Adobe-3.0
Level-2 pages may be generated by
.Fl T Ns Cm ps .
Output pages default to letter sized and are rendered in the Times font
family, 11-point.
Margins are calculated as 1/9 the page length and width.
Line-height is 1.4m.
.Pp
Special characters are rendered as in
.Sx ASCII Output .
.Pp
The following
.Fl O
arguments are accepted:
.Bl -tag -width Ds
.It Cm paper Ns = Ns Ar name
The paper size
.Ar name
may be one of
.Ar a3 ,
.Ar a4 ,
.Ar a5 ,
```

```
.Ar legal ,
or
.Ar letter .
You may also manually specify dimensions as
.Ar NNxNN ,
width by height in millimetres.
If an unknown value is encountered,
.Ar letter
is used.
.El
.Ss UTF\-8 Output
Use
.Fl T Ns Cm utf8
to force a UTF\-8 locale.
See
.Sx Locale Output
for details and options.
.Ss XHTML Output
Output produced by
.Fl T Ns Cm xhtml
conforms to XHTML-1.0 strict.
.Pp
See
.Sx HTML Output
for details; beyond generating XHTML tags instead of HTML tags, these
output modes are identical.
.Sh EXIT STATUS
The
.Nm
utility exits with one of the following values, controlled by the message
.Ar level
associated with the
.Fl W
option:
.Pp
.Bl -tag -width Ds -compact
.It 0
No warnings or errors occurred, or those that did were ignored because
they were lower than the requested
.Ar level .
.It 2
At least one warning occurred, but no error, and
.Fl W Ns Cm warning
was specified.
.It 3
At least one parsing error occurred, but no fatal error, and
.Fl W Ns Cm error
or
.Fl W Ns Cm warning
was specified.
.It 4
A fatal parsing error occurred.
.It 5
Invalid command line arguments were specified.
No input files have been read.
.It 6
An operating system error occurred, for example memory exhaustion or an
error accessing input files.
Such errors cause
.Nm
to exit at once, possibly in the middle of parsing or formatting a file.
.El
.Pp
Note that selecting
.Fl T Ns Cm lint
output mode implies
```



```
.Fl W Ns Cm warning .
.Sh EXAMPLES
To page manuals to the terminal:
.Pp
.Dl $ mandoc \-Wall,stop mandoc.1 2\*(Gt&1 | less
.Dl $ mandoc mandoc.1 mdoc.5 | less
.Pp
To produce HTML manuals with
.Ar style.css
as the style-sheet:
.Pp
.Dl $ mandoc \-Thtml -Ostyle=style.css mdoc.5 \*(Gt mdoc.5.html
.Pp
To check over a large set of manuals:
.Pp
.Dl $ mandoc \-Tlint ‘find /usr/src -name \e*\e.[1-9]‘
.Pp
To produce a series of PostScript manuals for A4 paper:
.Pp
.Dl $ mandoc \-Tps \-Opaper=a4 mdoc.5 man.5 \*(Gt manuals.ps
.Pp
Convert a modern
.Xr mdoc 5
manual to the older
.Xr man 5
format, for use on systems lacking an
.Xr mdoc 5
parser:
.Pp
.Dl $ mandoc \-Tman foo.mdoc \*(Gt foo.man
.Sh DIAGNOSTICS
Standard error messages reporting parsing errors are prefixed by
.Pp
.Sm off
.D1 Ar file : line : column : \ level :
.Sm on
.Pp
where the fields have the following meanings:
.Bl -tag -width "column"
.It Ar file
The name of the input file causing the message.
.It Ar line
The line number in that input file.
Line numbering starts at 1.
.It Ar column
The column number in that input file.
Column numbering starts at 1.
If the issue is caused by a word, the column number usually
points to the first character of the word.
.It Ar level
The message level, printed in capital letters.
.El
.Pp
Message levels have the following meanings:
.Bl -tag -width "warning"
.It Cm fatal
The parser is unable to parse a given input file at all.
No formatted output is produced from that input file.
.It Cm error
An input file contains syntax that cannot be safely interpreted,
either because it is invalid or because
.Nm
does not implement it yet.
By discarding part of the input or inserting missing tokens,
the parser is able to continue, and the error does not prevent
generation of formatted output, but typically, preparing that
```

```
output involves information loss, broken document structure
or unintended formatting.
.It Cm warning
An input file uses obsolete, discouraged or non-portable syntax.
All the same, the meaning of the input is unambiguous and a correct
rendering can be produced.
Documents causing warnings may render poorly when using other
formatting tools instead of
.Nm .
.El
.Pp
Messages of the
.Cm warning
and
.Cm error
levels are hidden unless their level, or a lower level, is requested using a
.Fl W
option or
.Fl T Ns Cm lint
output mode.
.Pp
The
.Nm
utility may also print messages related to invalid command line arguments
or operating system errors, for example when memory is exhausted or
input files cannot be read.
Such messages do not carry the prefix described above.
.Sh COMPATIBILITY
This section summarises
.Nm
compatibility with GNU troff.
Each input and output format is separately noted.
.Ss ASCII Compatibility
.Bl -bullet -compact
.It
Unrenderable unicode codepoints specified with
.Sq \e[uNNNN]
escapes are printed as
.Sq \&?
in mandoc.
In GNU troff, these raise an error.
.It
The
.Sq \&Bd \-literal
and
.Sq \&Bd \-unfilled
macros of
.Xr mdoc 5
in
.Fl T Ns Cm ascii
are synonyms, as are \-filled and \-ragged.
.It
In historic GNU troff, the
.Sq \&Pa
.Xr mdoc 5
macro does not underline when scoped under an
.Sq \&It
in the FILES section.
This behaves correctly in
.Nm .
.It
A list or display following the
.Sq \&Ss
.Xr mdoc 5
macro in
.Fl T Ns Cm ascii
```



```
does not assert a prior vertical break, just as it doesn't with
.Sq \&Sh .
.It
The
.Sq \&na
.Xr man 5
macro in
.Fl T Ns Cm ascii
has no effect.
.It
Words aren't hyphenated.
.El
.Ss HTML/XHTML Compatibility
.Bl -bullet -compact
.It
The
.Sq \efP
escape will revert the font to the previous
.Sq \ef
escape, not to the last rendered decoration, which is now dictated by
CSS instead of hard-coded.
It also will not span past the current scope,
for the same reason.
Note that in
.Sx ASCII Output
mode, this will work fine.
.It
The
.Xr mdoc 5
.Sq \&Bl \-hang
and
.Sq \&Bl \-tag
list types render similarly (no break following overreached left-hand
side) due to the expressive constraints of HTML.
.It
The
.Xr man 5
.Sq IP
and
.Sq TP
lists render similarly.
.El
.Sh INTERFACE STABILITY
The
.Nm
utility is
.Nm Committed ,
but the details of specific output formats other than ASCII are
.Nm Uncommitted .
.Sh SEE ALSO
.Xr eqn 5 ,
.Xr man 5 ,
.Xr mandoc_char 5 ,
.Xr mdoc 5 ,
.Xr mandoc_roff 5 ,
.Xr tbl 5
.Sh AUTHORS
The
.Nm
utility was written by
.An Kristaps Dzonsons ,
.Mt kristaps@bsd.lv .
.Sh CAVEATS
In
.Fl T Ns Cm html
and
```

```
.Fl T Ns Cm xhtml ,
the maximum size of an element attribute is determined by
.Dv BUFSIZ ,
which is usually 1024 bytes.
Be aware of this when setting long link
formats such as
.Fl O Ns Cm style Ns = Ns Ar really/long/link .
.Pp
Nesting elements within next-line element scopes of
.Fl m Ns Cm an ,
such as
.Sq br
within an empty
.Sq B ,
will confuse
.Fl T Ns Cm html
and
.Fl T Ns Cm xhtml
and cause them to forget the formatting of the prior next-line scope.
.Pp
The
.Sq \(aq
control character is an alias for the standard macro control character
and does not emit a line-break as stipulated in GNU troff.
```

```
**********************************************************
    1543 Wed Jul 16 14:05:08 2014
new/usr/src/man/man1m/catman.1m
feedback from Hans
mandoc import
**********************************************************
```

```
.\"
.\" This file and its contents are supplied under the terms of the
.\" Common Development and Distribution License ("CDDL"), version 1.0.
.\" You may only use this file in accordance with the terms of version
.\" 1.0 of the CDDL.
.\"
.\" A full copy of the text of the CDDL should have accompanied this
.\" source.  A copy of the CDDL is also available via the Internet at
.\" http://www.illumos.org/license/CDDL.
.\"
.\"
.\" Copyright 2014 Garrett D'Amore <garrett@damore.org>
.\"
.Dd Jul 16, 2014
.Dt CATAMN 1M
.Os
.Sh NAME
.Nm catman
.Nd generate
.Nm whatis
database files
.Sh SYNOPSIS
.Nm
.Op Fl M Ar path
.Sh DESCRIPTION
The
.Nm
utility generates a set of
.Nm whatis
database files suitable for use with
.Xr apropos 1
and
.Xr whatis 1 .
It is supplied for compatibility reasons.  The same databases can
be generated using the
.Fl w
option with
.Xr man 1 ,
and that command should be used instead.
.Bl -tag -width ".Fl d"
.It Fl M Ar path
Generate the
.Nm whatis
database files within the specified colon separated manual paths.
Overrides the
.Ev MANPATH
environment variable.
.El
.Sh ENVIRONMENT
The following environment variables affect the execution of
.Nm :
.Bl -tag -width ".Ev MANPATH"
.It Ev MANPATH
Used to specify a colon seperated list of manual paths within
which to generate
.Nm whatis
database files.
.El
.Sh DIAGNOSTICS
The
```



```
.Nm
utility exits 0 on success, and non-zero otherwise.
.Sh INTERFACE STABILITY
.Nm "Obsolete Committed" .
.Sh CODE SET INDEPENDENCE
Enabled.
.Sh SEE ALSO
.Xr apropos 1 ,
.Xr man 1 ,
.Xr whatis 1
'\" te
.\"  Copyright (c) 1998 Sun Microsystems, Inc.  All Rights Reserved.
.\" The contents of this file are subject to the terms of the Common Development
.\" You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE or http:
.\" When distributing Covered Code, include this CDDL HEADER in each file and in
.TH CATMAN 1M "Feb 27, 1998"
.SH NAME
catman \- create the formatted files for the reference manual
.SH SYNOPSIS
.LP
.nf
\fB/usr/bin/catman\fR [\fB-c\fR] [\fB-n\fR] [\fB-p\fR] [\fB-t\fR] [\fB-w\fR] [\f
     [\fB-T\fR \fImacro-package\fR] [\fIsections\fR]
.fi

.SH DESCRIPTION
.sp
.LP
The \fBcatman\fR utility creates the preformatted versions of the on-line
manual from the \fBnroff\fR(1) or \fBsgml\fR(5) input files. This feature
allows easy distribution of the preformatted manual pages among a group of
associated machines (for example, with \fBrdist\fR(1)), since it makes the
directories of preformatted manual pages self-contained and independent of the
unformatted entries.
.sp
.LP
\fBcatman\fR also creates the  \fBwindex\fR database file in the directories
specified by the  \fBMANPATH\fR or the  \fB-M\fR option. The \fBwindex\fR
database file is a three column list consisting of a keyword, the reference
page that the keyword points to, and a line of text that describes the purpose
of the utility or interface  documented on the reference page. Each keyword is
taken from the comma separated list of words on the \fBNAME\fR line before the
'\(mi' (dash). The reference page that the keyword points to is the first word
on the  \fBNAME\fR line. The text after the \(mi on the \fBNAME\fR line is the
descriptive text in the third column. The \fBNAME\fR line must be immediately
preceded by the page heading line created by the \fB\&.TH\fR macro (see
\fBNOTES\fR for required format).
.sp
.LP
Each manual page is examined and those whose preformatted versions are missing
or out of date are recreated. If any changes are made, \fBcatman\fR recreates
the \fBwindex\fR database.
.sp
.LP
If a manual page is a \fIshadow\fR page, that is, it sources another manual
page for its contents, a symbolic link is made in the \fBcat\fR\fIx\fR or
\fBfmt\fR\fIx\fR directory to the appropriate preformatted manual page.
.sp
.LP
Shadow files in an unformatted nroff source file are identified by the first
line being of the form \fB\&.so man\fR\fIx\fR\fB/yyy.\fR\fIx\fR\fB\&.\fR
.sp
.LP
Shadow files in the \fBSGML\fR sources are identified by the string
\fBSHADOW_PAGE\fR. The file entity declared in the shadow file identifies the
file to be sourced.
```

```
 57 .SH OPTIONS
 58 .sp
 59 .LP
 60 The following options are supported:
 61 .sp
 62 .ne 2
 63 .na
 64 \fB\fB-c\fR\fR
 65 .ad
 66 .RS 20n
 67 Create unformatted nroff source files in the appropriate \fBman\fR
 68 subdirectories from the  \fBSGML\fR sources. This option will overwrite any
 69 existing file in the \fBman\fR directory of the same name as the  \fBSGML\fR
 70 file.
 71 .RE

 73 .sp
 74 .ne 2
 75 .na
 76 \fB\fB-n\fR\fR
 77 .ad
 78 .RS 20n
 79 Do not create (or recreate) the \fBwindex\fR database. If the \fB-n\fR option
 80 is specified, the \fBwindex\fR database is not created and the \fBapropos\fR,
 81 \fBwhatis\fR, \fBman\fR \fB-f\fR, and \fBman\fR \fB-k\fR commands will fail.
 82 .RE

 84 .sp
 85 .ne 2
 86 .na
 87 \fB\fB-p\fR\fR
 88 .ad
 89 .RS 20n
 90 Print what would be done instead of doing it.
 91 .RE

 93 .sp
 94 .ne 2
 95 .na
 96 \fB\fB-t\fR\fR
 97 .ad
 98 .RS 20n
 99 Create \fBtroff\fRed entries in the appropriate \fBfmt\fR subdirectories
100 instead of \fBnroff\fRing into the \fBcat\fR subdirectories.
101 .RE

103 .sp
104 .ne 2
105 .na
106 \fB\fB-w\fR\fR
107 .ad
108 .RS 20n
109 Only create the \fBwindex\fR database that is used by \fBwhatis\fR(1) and the
110 \fBman\fR(1) \fB-f\fR and \fB-k\fR options.  No manual reformatting is done.
111 .RE

113 .sp
114 .ne 2
115 .na
116 \fB\fB-M\fR\fI directory\fR\fR
117 .ad
118 .RS 20n
119 Update manual pages located in the specified \fIdirectory\fR,
120 (\fB/usr/share/man\fR by default). If the  \fB-M\fR option is specified, the
121 directory argument must not contain a ',' (comma), since a comma is used to
122 delineate section numbers. See \fBman\fR(1).
```



```
123 .RE

125 .sp
126 .ne 2
127 .na
128 \fB\fB-T\fR\fI macro-package\fR\fR
129 .ad
130 .RS 20n
131 Use \fImacro-package\fR in place of the standard manual page macros, (
132 \fBman\fR(5) by default).
133 .RE

135 .SH OPERANDS
136 .sp
137 .LP
138 The following operand is supported:
139 .sp
140 .ne 2
141 .na
142 \fB\fIsections\fR\fR
143 .ad
144 .RS 12n
145 If there is one parameter not starting with a '\fB\(mi\fR\&', it is taken to be
146 a space separated list of manual sections to be processed by \fBcatman\fR. If
147 this operand is specified, only the manual sections in the list will be
148 processed.  For example,
149 .sp
150 .in +2
151 .nf
152 \fBcatman 1 2 3\fR
153 .fi
154 .in -2
155 .sp

157 only updates manual sections \fB1\fR, \fB2\fR, and \fB3\fR. If specific
158 sections are not listed, all sections in the  \fBman\fR directory specified by
159 the environment variable \fBMANPATH\fR are processed.
160 .RE

162 .SH ENVIRONMENT VARIABLES
163 .sp
164 .ne 2
165 .na
166 \fB\fBTROFF\fR\fR
167 .ad
168 .RS 11n
169 The name of the formatter to use when the \fB-t\fR flag is given. If not set,
170 \fBtroff\fR(1) is used.
171 .RE

173 .sp
174 .ne 2
175 .na
176 \fB\fBMANPATH\fR\fR
177 .ad
178 .RS 11n
179 A colon-separated list of directories that are processed by \fBcatman\fR and
180 \fBman\fR(1). Each directory can be followed by a comma-separated list of
181 sections. If set, its value overrides \fB/usr/share/man\fR as the default
182 directory search path, and the \fBman.cf\fR file as the default section search
183 path. The \fB-M\fR and \fB-s\fR flags, in turn, override these values.
184 .RE

186 .SH FILES
187 .sp
188 .ne 2
```

```
.na
\fB\fB/usr/share/man\fR\fR
.ad
.RS 28n
default manual directory location
.RE

.sp
.ne 2
.na
\fB\fB/usr/share/man/man*/*.*\fR\fR
.ad
.RS 28n
raw nroff input files
.RE

.sp
.ne 2
.na
\fB\fB/usr/share/man/sman*/*.*\fR\fR
.ad
.RS 28n
raw  \fBSGML\fR input files
.RE

.sp
.ne 2
.na
\fB\fB/usr/share/man/cat*/*.*\fR\fR
.ad
.RS 28n
preformatted \fBnroff\fRed manual pages
.RE

.sp
.ne 2
.na
\fB\fB/usr/share/man/fmt*/*.*\fR\fR
.ad
.RS 28n
preformatted \fBtroff\fRed manual pages
.RE

.sp
.ne 2
.na
\fB\fB/usr/share/man/windex\fR\fR
.ad
.RS 28n
table of contents and keyword database
.RE

.sp
.ne 2
.na
\fB\fB/usr/lib/makewhatis\fR\fR
.ad
.RS 28n
command script to make  \fBwindex\fR database
.RE

.sp
.ne 2
.na
\fB\fB/usr/share/lib/tmac/an\fR\fR
.ad
```



```
.RS 28n
default macro package
.RE

.SH ATTRIBUTES
.sp
.LP
See \fBattributes\fR(5) for descriptions of the following attributes:
.sp

.sp
.TS
box;
c | c
l | l .
ATTRIBUTE TYPE  ATTRIBUTE VALUE
_
CSI     Enabled
.TE

.SH SEE ALSO
.sp
.LP
\fBapropos\fR(1), \fBman\fR(1), \fBnroff\fR(1), \fBrdist\fR(1), \fBrm\fR(1),
\fBtroff\fR(1), \fBwhatis\fR(1), \fBattributes\fR(5), \fBman\fR(5),
\fBsgml\fR(5)
.SH DIAGNOSTICS
.sp
.ne 2
.na
\fB\fBman?/xxx.? (.so'ed from man?/yyy.?): No such file or directory\fR\fR
.ad
.sp .6
.RS 4n
The file outside the parentheses is missing, and is referred to by the file
inside them.
.RE

.sp
.ne 2
.na
\fB\fBtarget of .so in man?/xxx.? must be relative to /usr/man\fR\fR
.ad
.sp .6
.RS 4n
\fBcatman\fR only allows references to filenames that are relative to the
directory \fB/usr/man\fR.
.RE

.sp
.ne 2
.na
\fB\fBopendir:man?:\fR \fBNo\fR \fBsuch\fR \fBfile\fR \fBor\fR
\fBdirectory\fR\fR
.ad
.sp .6
.RS 4n
A harmless warning message indicating that one of the directories \fBcatman\fR
normally looks for is missing.
.RE

.sp
.ne 2
.na
\fB\fB*.*:\fR \fBNo\fR \fBsuch\fR \fBfile\fR \fBor\fR \fBdirectory\fR\fR
.ad
```

```
.sp .6
.RS 4n
A harmless warning message indicating \fBcatman\fR came across an empty
directory.
.RE

.SH WARNINGS
.sp
.LP
If a user, who has previously run \fBcatman\fR to install the \fBcat*\fR
directories, upgrades the operating system, the entire \fBcat*\fR directory
structure should be removed prior to running \fBcatman\fR. See \fBrm\fR(1).
.sp
.LP
Do not re-run \fBcatman\fR to re-build the \fBwhatis\fR database unless the
complete set of \fBman*\fR directories is present. \fBcatman\fR builds this
\fBwindex\fR file based on the  \fBman*\fR directories.
.SH NOTES
.sp
.LP
To generate a valid windex index file, \fBcatman\fR has certain requirements.
Within the individual man page file, \fBcatman\fR requires two macro lines to
have a specific format. These are the \fB\&.TH \fRpage heading line and the
\fB\&.SH NAME \fRline.
.sp
.LP
The \fB\&.TH \fRmacro requires at least the first three arguments, that is, the
filename, section number, and the date.  The \fB\&.TH \fRline starts off with
the \fB\&.TH \fRmacro, followed by a space, the man page filename, a single
space, the section number, another single space, and the date. The date should
appear in double quotes and is specified as "day month year," with the month
always abbreviated to the first three letters (Jan, Feb, Mar, and so forth).
.sp
.LP
The \fB\&.SH NAME \fRmacro, also known as the \fBNAME \fRline, must immediately
follow the \fB\&.TH \fRline, with nothing in between those lines. No font
changes are permitted in the \fBNAME \fRline. The \fBNAME \fRline is
immediately followed by a line containing the man page filename; then shadow
page names, if applicable, separated by commas; a dash; and a brief summary
statement. These elements should all be on one line; no carriage returns are
permitted.
.sp
.LP
An example of proper coding of these lines is:
.sp
.in +2
.nf
\&.TH NISMATCH 1M "Apr "10, 1998""
\&.SH NAME
nismatch, nisgrep \e- utilities for searching NIS+ tables
.fi
.in -2
```

```
**********************************************************
    4030 Wed Jul 16 14:05:08 2014
new/usr/src/man/man5/Makefile
mandoc import
**********************************************************
#
# This file and its contents are supplied under the terms of the
# Common Development and Distribution License ("CDDL"), version 1.0.
# You may only use this file in accordance with the terms of version
# 1.0 of the CDDL.
#
# A full copy of the text of the CDDL should have accompanied this
# source.  A copy of the CDDL is also available via the Internet
# at http://www.illumos.org/license/CDDL.
#

#
# Copyright 2011, Richard Lowe
# Copyright (c) 2012 by Delphix. All rights reserved.
# Copyright 2013 Nexenta Systems, Inc.  All rights reserved.
# Copyright 2014 Garrett D'Amore <garrett@damore.org>
#

include         $(SRC)/Makefile.master

MANSECT=        5

MANFILES=       Intro.5                 \
                acl.5                   \
                ad.5                    \
                ascii.5                 \
                attributes.5            \
                audit_binfile.5         \
                audit_remote.5          \
                audit_syslog.5          \
                brands.5                \
                cancellation.5          \
                charmap.5               \
                condition.5             \
                crypt_bsdbf.5           \
                crypt_bsdmd5.5          \
                crypt_sha256.5          \
                crypt_sha512.5          \
                crypt_sunmd5.5          \
                crypt_unix.5            \
                device_clean.5          \
                dhcp.5                  \
                dhcp_modules.5          \
                environ.5               \
                eqn.5                   \
                eqnchar.5               \
                extendedFILE.5          \
                filesystem.5            \
                fnmatch.5               \
                formats.5               \
                fsattr.5                \
                grub.5                  \
                gss_auth_rules.5        \
                hal.5                   \
                iconv.5                 \
                iconv_unicode.5         \
                ieee802.11.5            \
                ipfilter.5              \
                isalist.5               \
                kerberos.5              \
                krb5_auth_rules.5       \
```



```
                krb5envvar.5            \
                largefile.5             \
                lf64.5                  \
                lfcompile.5             \
                lfcompile64.5           \
                locale.5                \
                man.5                   \
                mandoc_char.5           \
                mandoc_roff.5           \
                mdoc.5                  \
                mansun.5                \
                me.5                    \
                mech_spnego.5           \
                mm.5                    \
                ms.5                    \
                mutex.5                 \
                nfssec.5                \
                pam_allow.5             \
                pam_authtok_check.5     \
                pam_authtok_get.5       \
                pam_authtok_store.5     \
                pam_deny.5              \
                pam_dhkeys.5            \
                pam_dial_auth.5         \
                pam_krb5.5              \
                pam_krb5_migrate.5      \
                pam_ldap.5              \
                pam_list.5              \
                pam_passwd_auth.5       \
                pam_rhosts_auth.5       \
                pam_roles.5             \
                pam_sample.5            \
                pam_smb_passwd.5        \
                pam_smbfs_login.5       \
                pam_tsol_account.5      \
                pam_unix_account.5      \
                pam_unix_auth.5         \
                pam_unix_cred.5         \
                pam_unix_session.5      \
                pkcs11_kernel.5         \
                pkcs11_softtoken.5      \
                pkcs11_tpm.5            \
                privileges.5            \
                prof.5                  \
                rbac.5                  \
                regex.5                 \
                regexp.5                \
                resource_controls.5     \
                smf.5                   \
                smf_bootstrap.5         \
                smf_method.5            \
                smf_restarter.5         \
                smf_security.5          \
                smf_template.5          \
                standards.5             \
                sticky.5                \
                tbl.5                   \
                tecla.5                 \
                term.5                  \
                threads.5               \
                trusted_extensions.5    \
                vgrindefs.5             \
                zones.5                 \
                zpool-features.5

MANLINKS=       ANSI.5          \
```

```
127                 C++.5           \
128                 C.5             \
129                 CSI.5           \
130                 ISO.5           \
131                 MT-Level.5      \
132                 POSIX.1.5       \
133                 POSIX.2.5       \
134                 POSIX.5         \
135                 RBAC.5          \
136                 SUS.5           \
137                 SUSv2.5         \
138                 SUSv3.5         \
139                 SVID.5          \
140                 SVID3.5         \
141                 XNS.5           \
142                 XNS4.5          \
143                 XNS5.5          \
144                 XPG.5           \
145                 XPG3.5          \
146                 XPG4.5          \
147                 XPG4v2.5        \
148                 advance.5       \
149                 architecture.5  \
150                 availability.5  \
151                 compile.5       \
152                 intro.5         \
153                 pthreads.5      \
154                 stability.5     \
155                 standard.5      \
156                 step.5          \
157                 teclarc.5

159 intro.5         := LINKSRC = Intro.5

161 CSI.5           := LINKSRC = attributes.5
162 MT-Level.5      := LINKSRC = attributes.5
163 architecture.5  := LINKSRC = attributes.5
164 availability.5  := LINKSRC = attributes.5
165 stability.5     := LINKSRC = attributes.5
166 standard.5      := LINKSRC = attributes.5

168 RBAC.5          := LINKSRC = rbac.5

170 advance.5       := LINKSRC = regexp.5
171 compile.5       := LINKSRC = regexp.5
172 step.5          := LINKSRC = regexp.5

174 ANSI.5          := LINKSRC = standards.5
175 C++.5           := LINKSRC = standards.5
176 C.5             := LINKSRC = standards.5
177 ISO.5           := LINKSRC = standards.5
178 POSIX.1.5       := LINKSRC = standards.5
179 POSIX.2.5       := LINKSRC = standards.5
180 POSIX.5         := LINKSRC = standards.5
181 SUS.5           := LINKSRC = standards.5
182 SUSv2.5         := LINKSRC = standards.5
183 SUSv3.5         := LINKSRC = standards.5
184 SVID.5          := LINKSRC = standards.5
185 SVID3.5         := LINKSRC = standards.5
186 XNS.5           := LINKSRC = standards.5
187 XNS4.5          := LINKSRC = standards.5
188 XNS5.5          := LINKSRC = standards.5
189 XPG.5           := LINKSRC = standards.5
190 XPG3.5          := LINKSRC = standards.5
191 XPG4.5          := LINKSRC = standards.5
192 XPG4v2.5        := LINKSRC = standards.5
```



```
194 teclarc.5       := LINKSRC = tecla.5

196 pthreads.5      := LINKSRC = threads.5

198 .KEEP_STATE:

200 include         $(SRC)/man/Makefile.man

202 install:        $(ROOTMANFILES) $(ROOTMANLINKS)
```

```
**********************************************************
    7157 Wed Jul 16 14:05:08 2014
new/usr/src/man/man5/eqn.5
mandoc import
**********************************************************
.\"
.\" Permission to use, copy, modify, and distribute this software for any
.\" purpose with or without fee is hereby granted, provided that the above
.\" copyright notice and this permission notice appear in all copies.
.\"
.\" THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES
.\" WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF
.\" MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR
.\" ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES
.\" WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN
.\" ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF
.\" OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
.\"
.\"
.\" Copyright (c) 2011 Kristaps Dzonsons <kristaps@bsd.lv>
.\" Copyright 2012 Nexenta Systems, Inc. All rights reserved.
.\"
.Dd Sep 25, 2011
.Dt EQN 5
.Os
.Sh NAME
.Nm eqn
.Nd eqn language reference for mandoc
.Sh DESCRIPTION
The
.Nm eqn
language is an equation-formatting language.
It is used within
.Xr mdoc 5
and
.Xr man 5
.Ux
manual pages.
It describes the
.Em structure
of an equation, not its mathematical meaning.
This manual describes the
.Nm
language accepted by the
.Xr mandoc 1
utility, which corresponds to the Second Edition eqn specification (see
.Sx SEE ALSO
for references).
.Pp
Equations within
.Xr mdoc 5
or
.Xr man 5
documents are enclosed by the standalone
.Sq \&.EQ
and
.Sq \&.EN
tags.
Equations are multi-line blocks consisting of formulas and control
statements.
.Sh EQUATION STRUCTURE
Each equation is bracketed by
.Sq \&.EQ
and
.Sq \&.EN
strings.
```



```
.Em Note :
these are not the same as
.Xr roff 5
macros, and may only be invoked as
.Sq \&.EQ .
.Pp
The equation grammar is as follows, where quoted strings are
case-sensitive literals in the input:
.Bd -literal -offset indent
eqn     : box | eqn box
box     : text
        | \*q{\*q eqn \*q}\*q
        | \*qdefine\*q text text
        | \*qndefine\*q text text
        | \*qtdefine\*q text text
        | \*qgfont\*q text
        | \*qgsize\*q text
        | \*qset\*q text text
        | \*qundef\*q text
        | box pos box
        | box mark
        | \*qmatrix\*q \*q{\*q [col \*q{\*q list \*q}\*q ]*
        | pile \*q{\*q list \*q}\*q
        | font box
        | \*qsize\*q text box
        | \*qleft\*q text eqn [\*qright\*q text]
col     : \*qlcol\*q | \*qrcol\*q | \*qccol\*q | \*qcol\*q
text    : [^space\e\*q]+ | \e\*q.*\e\*q
pile    : \*qlpile\*q | \*qcpile\*q | \*qrpile\*q | \*qpile\*q
pos     : \*qover\*q | \*qsup\*q | \*qsub\*q | \*qto\*q | \*qfrom\*q
mark    : \*qdot\*q | \*qdotdot\*q | \*qhat\*q | \*qtilde\*q | \*qvec\*q
        | \*qdyad\*q | \*qbar\*q | \*qunder\*q
font    : \*qroman\*q | \*qitalic\*q | \*qbold\*q | \*qfat\*q
list    : eqn
        | list \*qabove\*q eqn
space   : [\e^~ \et]
.Ed
.Pp
White-space consists of the space, tab, circumflex, and tilde
characters.
If within a quoted string, these space characters are retained.
Quoted strings are also not scanned for replacement definitions.
.Pp
The following text terms are translated into a rendered glyph, if
available: alpha, beta, chi, delta, epsilon, eta, gamma, iota, kappa,
lambda, mu, nu, omega, omicron, phi, pi, psi, rho, sigma, tau, theta,
upsilon, xi, zeta, DELTA, GAMMA, LAMBDA, OMEGA, PHI, PI, PSI, SIGMA,
THETA, UPSILON, XI, inter (intersection), union (union), prod (product),
int (integral), sum (summation), grad (gradient), del (vector
differential), times (multiply), cdot (centre-dot), nothing (zero-width
space), approx (approximately equals), prime (prime), half (one-half),
partial (partial differential), inf (infinity), >> (much greater), <<
(much less), \-> (left arrow), <\- (right arrow), += (plus-minus), !=
(not equal), == (equivalence), <= (less-than-equal), and >=
(more-than-equal).
.Pp
The following control statements are available:
.Bl -tag -width Ds
.It Cm define
Replace all occurrences of a key with a value.
Its syntax is as follows:
.Pp
.D1 define Ar key cvalc
.Pp
The first character of the value string,
.Ar c ,
```

```
is used as the delimiter for the value
.Ar val .
This allows for arbitrary enclosure of terms (not just quotes), such as
.Pp
.D1 define Ar foo ’bar baz’
.D1 define Ar foo cbar bazc
.Pp
It is an error to have an empty
.Ar key
or
.Ar val .
Note that a quoted
.Ar key
causes errors in some
.Nm
implementations and should not be considered portable.
It is not expanded for replacements.
Definitions may refer to other definitions; these are evaluated
recursively when text replacement occurs and not when the definition is
created.
.Pp
Definitions can create arbitrary strings, for example, the following is
a legal construction.
.Bd -literal -offset indent
define foo ’define’
foo bar ’baz’
.Ed
.Pp
Self-referencing definitions will raise an error.
The
.Cm ndefine
statement is a synonym for
.Cm define ,
while
.Cm tdefine
is discarded.
.It Cm gfont
Set the default font of subsequent output.
Its syntax is as follows:
.Pp
.D1 gfont Ar font
.Pp
In mandoc, this value is discarded.
.It Cm gsize
Set the default size of subsequent output.
Its syntax is as follows:
.Pp
.D1 gsize Ar size
.Pp
The
.Ar size
value should be an integer.
.It Cm set
Set an equation mode.
In mandoc, both arguments are thrown away.
Its syntax is as follows:
.Pp
.D1 set Ar key val
.Pp
The
.Ar key
and
.Ar val
are not expanded for replacements.
This statement is a GNU extension.
.It Cm undef
```



```
Unset a previously-defined key.
Its syntax is as follows:
.Pp
.D1 define Ar key
.Pp
Once invoked, the definition for
.Ar key
is discarded.
The
.Ar key
is not expanded for replacements.
This statement is a GNU extension.
.El
.Sh COMPATIBILITY
This section documents the compatibility of mandoc
.Nm
and the troff
.Nm
implementation (including GNU troff).
.Pp
.Bl -dash -compact
.It
The text string
.Sq \e\*q
is interpreted as a literal quote in troff.
In mandoc, this is interpreted as a comment.
.It
In troff, The circumflex and tilde white-space symbols map to
fixed-width spaces.
In mandoc, these characters are synonyms for the space character.
.It
The troff implementation of
.Nm
allows for equation alignment with the
.Cm mark
and
.Cm lineup
tokens.
mandoc discards these tokens.
The
.Cm back Ar n ,
.Cm fwd Ar n ,
.Cm up Ar n ,
and
.Cm down Ar n
commands are also ignored.
.El
.Sh SEE ALSO
.Xr mandoc 1 ,
.Xr man 7 ,
.Xr mandoc_char 5 ,
.Xr mdoc 5 ,
.Xr roff 5
.Rs
.%A Brian W. Kernighan
.%A Lorinda L. Cherry
.%T System for Typesetting Mathematics
.%J Communications of the ACM
.%V 18
.%P 151\(en157
.%D March, 1975
.Re
.Rs
.%A Brian W. Kernighan
.%A Lorinda L. Cherry
.%T Typesetting Mathematics, User’s Guide
```

```
.%D 1976
.Re
.Rs
.%A Brian W. Kernighan
.%A Lorinda L. Cherry
.%T Typesetting Mathematics, User's Guide (Second Edition)
.%D 1978
.Re
.Sh HISTORY
The eqn utility, a preprocessor for troff, was originally written by
Brian W. Kernighan and Lorinda L. Cherry in 1975.
The GNU reimplementation of eqn, part of the GNU troff package, was
released in 1989 by James Clark.
The eqn component of
.Xr mandoc 1
was added in 2011.
.Sh AUTHORS
This
.Nm
reference was written by
.An Kristaps Dzonsons ,
.Mt kristaps@bsd.lv .
```

```
**********************************************************
   23340 Wed Jul 16 14:05:08 2014
new/usr/src/man/man5/man.5
feedback from Hans
mandoc import
**********************************************************
```

```
.\"
.\" Permission to use, copy, modify, and distribute this software for any
.\" purpose with or without fee is hereby granted, provided that the above
.\" copyright notice and this permission notice appear in all copies.
.\"
.\" THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES
.\" WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF
.\" MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR
.\" ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES
.\" WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN
.\" ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF
.\" OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
.\"
.\"
.\" Copyright (c) 2009, 2010, 2011 Kristaps Dzonsons <kristaps@bsd.lv>
.\" Copyright (c) 2011 Ingo Schwarze <schwarze@openbsd.org>
.\" Copyright 2012 Nexenta Systems, Inc. All rights reserved.
.\"
.Dd Jan 3, 2012
.Dt MAN 5
.Os
.Sh NAME
.Nm man
.Nd legacy formatting language for manual pages
.Sh DESCRIPTION
Traditionally, the
.Nm man
language has been used to write
.Ux
manuals for the
.Xr man 1
utility.
It supports limited control of presentational details like fonts,
indentation and spacing.
This reference document describes the structure of manual pages
and the syntax and usage of the man language.
.Pp
.Bf -emphasis
Do not use
.Nm
to write your manuals:
.Ef
It lacks support for semantic markup.
Use the
.Xr mdoc 5
language, instead.
.Pp
In a
.Nm
document, lines beginning with the control character
.Sq \&.
are called
.Dq macro lines .
The first word is the macro name.
It usually consists of two capital letters.
For a list of available macros, see
.Sx MACRO OVERVIEW .
The words following the macro name are arguments to the macro.
.Pp
Lines not beginning with the control character are called
```



```
.Dq text lines .
They provide free-form text to be printed; the formatting of the text
depends on the respective processing context:
.Bd -literal -offset indent
\&.SH Macro lines change control state.
Text lines are interpreted within the current state.
.Ed
.Pp
Many aspects of the basic syntax of the
.Nm
language are based on the
.Xr roff 5
language; see the
.Em LANGUAGE SYNTAX
and
.Em MACRO SYNTAX
sections in the
.Xr roff 5
manual for details, in particular regarding
comments, escape sequences, whitespace, and quoting.
.Sh MANUAL STRUCTURE
Each
.Nm
document must contain the
.Sx \&TH
macro describing the document’s section and title.
It may occur anywhere in the document, although conventionally it
appears as the first macro.
.Pp
Beyond
.Sx \&TH ,
at least one macro or text line must appear in the document.
.Pp
The following is a well-formed skeleton
.Nm
file for a utility
.Qq progname :
.Bd -literal -offset indent
\&.TH PROGNAME 1 "Oct 10, 2009"
\&.SH NAME
\efBprogname\efR \e(en a description goes here
\&.\e\(dq .SH LIBRARY
\&.\e\(dq For sections 2 & 3 only.
\&.SH SYNOPSIS
\efBprogname\efR [\efB\e-options\efR] arguments...
\&.SH DESCRIPTION
The \efBfoo\efR utility processes files...
\&.\e\(dq .SH IMPLEMENTATION NOTES
\&.\e\(dq .SH RETURN VALUES
\&.\e\(dq For sections 2, 3, & 9 only.
\&.\e\(dq .SH ENVIRONMENT
\&.\e\(dq For sections 1, 1M, 5, & 6 only.
\&.\e\(dq .SH FILES
\&.\e\(dq .SH EXIT STATUS
\&.\e\(dq For sections 1, 1M, & 6 only.
\&.\e\(dq .SH EXAMPLES
\&.\e\(dq .SH DIAGNOSTICS
\&.\e\(dq For sections 1, 1M, 5, 6, & 7 only.
\&.\e\(dq .SH ERRORS
\&.\e\(dq For sections 2, 3, & 9 only.
\&.\e\(dq .SH SEE ALSO
\&.\e\(dq .BR foo ( 1 )
\&.\e\(dq .SH STANDARDS
\&.\e\(dq .SH HISTORY
\&.\e\(dq .SH AUTHORS
\&.\e\(dq .SH CAVEATS
```

```
\&.\e\(dq .SH BUGS
\&.\e\(dq .SH SECURITY CONSIDERATIONS
.Ed
.Pp
The sections in a
.Nm
document are conventionally ordered as they appear above.
Sections should be composed as follows:
.Bl -ohang -offset indent
.It Em NAME
The name(s) and a short description of the documented material.
The syntax for this is generally as follows:
.Pp
.D1 \efBname\efR \e(en description
.It Em LIBRARY
The name of the library containing the documented material, which is
assumed to be a function in a section 2 or 3 manual.
For functions in the C library, this may be as follows:
.Pp
.D1 Standard C Library (libc, -lc)
.It Em SYNOPSIS
Documents the utility invocation syntax, function call syntax, or device
configuration.
.Pp
For the first, utilities (sections 1, 1M, and 6), this is
generally structured as follows:
.Pp
.D1 \efBname\efR [-\efBab\efR] [-\efBc\efR\efIarg\efR] \efBpath\efR...
.Pp
For the second, function calls (sections 2, 3, 9):
.Pp
.D1 \&.B char *name(char *\efIarg\efR);
.Pp
And for the third, configurations (section 7):
.Pp
.D1 \&.B name* at cardbus ? function ?
.Pp
Manuals not in these sections generally don't need a
.Em SYNOPSIS .
.It Em DESCRIPTION
This expands upon the brief, one-line description in
.Em NAME .
It usually contains a break-down of the options (if documenting a
command).
.It Em IMPLEMENTATION NOTES
Implementation-specific notes should be kept here.
This is useful when implementing standard functions that may have side
effects or notable algorithmic implications.
.It Em RETURN VALUES
This section documents the return values of functions in sections 2, 3, and 9.
.It Em ENVIRONMENT
Documents any usages of environment variables, e.g.,
.Xr environ 5 .
.It Em FILES
Documents files used.
It's helpful to document both the file name and a short description of how
the file is used (created, modified, etc.).
.It Em EXIT STATUS
This section documents the command exit status for
section 1, 6, and 8 utilities.
Historically, this information was described in
.Em DIAGNOSTICS ,
a practise that is now discouraged.
.It Em EXAMPLES
Example usages.
This often contains snippets of well-formed,
```



```
well-tested invocations.
Make sure that examples work properly!
.It Em DIAGNOSTICS
Documents error conditions.
This is most useful in section 4 manuals.
Historically, this section was used in place of
.Em EXIT STATUS
for manuals in sections 1, 6, and 8; however, this practise is
discouraged.
.It Em ERRORS
Documents error handling in sections 2, 3, and 9.
.It Em SEE ALSO
References other manuals with related topics.
This section should exist for most manuals.
.Pp
.D1 \&.BR bar \&( 1 \&),
.Pp
Cross-references should conventionally be ordered
first by section, then alphabetically.
.It Em STANDARDS
References any standards implemented or used, such as
.Pp
.D1 IEEE Std 1003.2 (\e(lqPOSIX.2\e(rq)
.Pp
If not adhering to any standards, the
.Em HISTORY
section should be used.
.It Em HISTORY
A brief history of the subject, including where support first appeared.
.It Em AUTHORS
Credits to the person or persons who wrote the code and/or documentation.
Authors should generally be noted by both name and email address.
.It Em CAVEATS
Common misuses and misunderstandings should be explained
in this section.
.It Em BUGS
Known bugs, limitations, and work-arounds should be described
in this section.
.It Em SECURITY CONSIDERATIONS
Documents any security precautions that operators should consider.
.El
.Sh MACRO OVERVIEW
This overview is sorted such that macros of similar purpose are listed
together, to help find the best macro for any given purpose.
Deprecated macros are not included in the overview, but can be found
in the alphabetical reference below.
.Ss Page header and footer meta-data
.Bl -column "PP, LP, P" description
.It Sx TH Ta set the title: Ar title section date Op Ar source Op Ar volume
.It Sx AT Ta display AT&T UNIX version in the page footer (<= 1 argument)
.It Sx UC Ta display BSD version in the page footer (<= 1 argument)
.El
.Ss Sections and paragraphs
.Bl -column "PP, LP, P" description
.It Sx SH Ta section header (one line)
.It Sx SS Ta subsection header (one line)
.It Sx PP , LP , P Ta start an undecorated paragraph (no arguments)
.It Sx RS , RE Ta reset the left margin: Op Ar width
.It Sx IP Ta indented paragraph: Op Ar head Op Ar width
.It Sx TP Ta tagged paragraph: Op Ar width
.It Sx HP Ta hanged paragraph: Op Ar width
.It Sx \&br Ta force output line break in text mode (no arguments)
.It Sx \&sp Ta force vertical space: Op Ar height
.It Sx fi , nf Ta fill mode and no-fill mode (no arguments)
.It Sx in Ta additional indent: Op Ar width
.El
```

```
.Ss Physical markup
.Bl -column "PP, LP, P" description
.It Sx B Ta boldface font
.It Sx I Ta italic font
.It Sx R Ta roman (default) font
.It Sx SB Ta small boldface font
.It Sx SM Ta small roman font
.It Sx BI Ta alternate between boldface and italic fonts
.It Sx BR Ta alternate between boldface and roman fonts
.It Sx IB Ta alternate between italic and boldface fonts
.It Sx IR Ta alternate between italic and roman fonts
.It Sx RB Ta alternate between roman and boldface fonts
.It Sx RI Ta alternate between roman and italic fonts
.El
.Ss Semantic markup
.Bl -column "PP, LP, P" description
.It Sx OP Ta optional arguments
.El
.Sh MACRO REFERENCE
This section is a canonical reference to all macros, arranged
alphabetically.
For the scoping of individual macros, see
.Sx MACRO SYNTAX .
.Ss \&AT
Sets the volume for the footer for compatibility with man pages from
.Tn AT&T UNIX
releases.
The optional arguments specify which release it is from.
.Ss \&B
Text is rendered in bold face.
.Pp
See also
.Sx \&I
and
.Sx \&R .
.Ss \&BI
Text is rendered alternately in bold face and italic.
Thus,
.Sq .BI this word and that
causes
.Sq this
and
.Sq and
to render in bold face, while
.Sq word
and
.Sq that
render in italics.
Whitespace between arguments is omitted in output.
.Pp
Examples:
.Pp
.Dl \&.BI bold italic bold italic
.Pp
The output of this example will be emboldened
.Dq bold
and italicised
.Dq italic ,
with spaces stripped between arguments.
.Pp
See also
.Sx \&IB ,
.Sx \&BR ,
.Sx \&RB ,
.Sx \&RI ,
and
```



```
.Sx \&IR .
.Ss \&BR
Text is rendered alternately in bold face and roman (the default font).
Whitespace between arguments is omitted in output.
.Pp
See
.Sx \&BI
for an equivalent example.
.Pp
See also
.Sx \&BI ,
.Sx \&IB ,
.Sx \&RB ,
.Sx \&RI ,
and
.Sx \&IR .
.Ss \&DT
Has no effect.
Included for compatibility.
.Ss \&HP
Begin a paragraph whose initial output line is left-justified, but
subsequent output lines are indented, with the following syntax:
.Bd -filled -offset indent
.Pf \. Sx \&HP
.Op Cm width
.Ed
.Pp
The
.Cm width
argument must conform to
.Sx Scaling Widths .
If specified, it’s saved for later paragraph left-margins; if unspecified, the
saved or default width is used.
.Pp
See also
.Sx \&IP ,
.Sx \&LP ,
.Sx \&P ,
.Sx \&PP ,
and
.Sx \&TP .
.Ss \&I
Text is rendered in italics.
.Pp
See also
.Sx \&B
and
.Sx \&R .
.Ss \&IB
Text is rendered alternately in italics and bold face.
Whitespace between arguments is omitted in output.
.Pp
See
.Sx \&BI
for an equivalent example.
.Pp
See also
.Sx \&BI ,
.Sx \&BR ,
.Sx \&RB ,
.Sx \&RI ,
and
.Sx \&IR .
.Ss \&IP
Begin an indented paragraph with the following syntax:
.Bd -filled -offset indent
```

```
.Pf \. Sx \&IP
.Op Cm head Op Cm width
.Ed
.Pp
The
.Cm width
argument defines the width of the left margin and is defined by
.Sx Scaling Widths .
It’s saved for later paragraph left-margins; if unspecified, the saved or
default width is used.
.Pp
The
.Cm head
argument is used as a leading term, flushed to the left margin.
This is useful for bulleted paragraphs and so on.
.Pp
See also
.Sx \&HP ,
.Sx \&LP ,
.Sx \&P ,
.Sx \&PP ,
and
.Sx \&TP .
.Ss \&IR
Text is rendered alternately in italics and roman (the default font).
Whitespace between arguments is omitted in output.
.Pp
See
.Sx \&BI
for an equivalent example.
.Pp
See also
.Sx \&BI ,
.Sx \&IB ,
.Sx \&BR ,
.Sx \&RB ,
and
.Sx \&RI .
.Ss \&LP
Begin an undecorated paragraph.
The scope of a paragraph is closed by a subsequent paragraph,
sub-section, section, or end of file.
The saved paragraph left-margin width is reset to the default.
.Pp
See also
.Sx \&HP ,
.Sx \&IP ,
.Sx \&P ,
.Sx \&PP ,
and
.Sx \&TP .
.Ss \&OP
Optional command-line argument.
This has the following syntax:
.Bd -filled -offset indent
.Pf \. Sx \&OP
.Cm key Op Cm value
.Ed
.Pp
The
.Cm key
is usually a command-line flag and
.Cm value
its argument.
.Ss \&P
Synonym for
```



```
.Sx \&LP .
.Pp
See also
.Sx \&HP ,
.Sx \&IP ,
.Sx \&LP ,
.Sx \&PP ,
and
.Sx \&TP .
.Ss \&PP
Synonym for
.Sx \&LP .
.Pp
See also
.Sx \&HP ,
.Sx \&IP ,
.Sx \&LP ,
.Sx \&P ,
and
.Sx \&TP .
.Ss \&R
Text is rendered in roman (the default font).
.Pp
See also
.Sx \&I
and
.Sx \&B .
.Ss \&RB
Text is rendered alternately in roman (the default font) and bold face.
Whitespace between arguments is omitted in output.
.Pp
See
.Sx \&BI
for an equivalent example.
.Pp
See also
.Sx \&BI ,
.Sx \&IB ,
.Sx \&BR ,
.Sx \&RI ,
and
.Sx \&IR .
.Ss \&RE
Explicitly close out the scope of a prior
.Sx \&RS .
The default left margin is restored to the state of the original
.Sx \&RS
invocation.
.Ss \&RI
Text is rendered alternately in roman (the default font) and italics.
Whitespace between arguments is omitted in output.
.Pp
See
.Sx \&BI
for an equivalent example.
.Pp
See also
.Sx \&BI ,
.Sx \&IB ,
.Sx \&BR ,
.Sx \&RB ,
and
.Sx \&IR .
.Ss \&RS
Temporarily reset the default left margin.
This has the following syntax:
```

```
.Bd -filled -offset indent
.Pf \. Sx \&RS
.Op Cm width
.Ed
.Pp
The
.Cm width
argument must conform to
.Sx Scaling Widths .
If not specified, the saved or default width is used.
.Pp
See also
.Sx \&RE .
.Ss \&SB
Text is rendered in small size (one point smaller than the default font)
bold face.
.Ss \&SH
Begin a section.
The scope of a section is only closed by another section or the end of
file.
The paragraph left-margin width is reset to the default.
.Ss \&SM
Text is rendered in small size (one point smaller than the default
font).
.Ss \&SS
Begin a sub-section.
The scope of a sub-section is closed by a subsequent sub-section,
section, or end of file.
The paragraph left-margin width is reset to the default.
.Ss \&TH
Sets the title of the manual page with the following syntax:
.Bd -filled -offset indent
.Pf \. Sx \&TH
.Ar title section date
.Op Ar source Op Ar volume
.Ed
.Pp
Conventionally, the document
.Ar title
is given in all caps.
The recommended
.Ar date
format is
.Sy YYYY-MM-DD
as specified in the ISO-8601 standard;
if the argument does not conform, it is printed verbatim.
If the
.Ar date
is empty or not specified, the current date is used.
The optional
.Ar source
string specifies the organisation providing the utility.
The
.Ar volume
string replaces the default rendered volume, which is dictated by the
manual section.
.Pp
Examples:
.Pp
.Dl \&.TH CVS 5 "1992-02-12" GNU
.Ss \&TP
Begin a paragraph where the head, if exceeding the indentation width, is
followed by a newline; if not, the body follows on the same line after a
buffer to the indentation width.
Subsequent output lines are indented.
The syntax is as follows:
```



```
.Bd -filled -offset indent
.Pf \. Sx \&TP
.Op Cm width
.Ed
.Pp
The
.Cm width
argument must conform to
.Sx Scaling Widths .
If specified, it's saved for later paragraph left-margins; if
unspecified, the saved or default width is used.
.Pp
See also
.Sx \&HP ,
.Sx \&IP ,
.Sx \&LP ,
.Sx \&P ,
and
.Sx \&PP .
.Ss \&UC
Sets the volume for the footer for compatibility with man pages from
BSD releases.
The optional first argument specifies which release it is from.
.Ss \&br
Breaks the current line.
Consecutive invocations have no further effect.
.Pp
See also
.Sx \&sp .
.Ss \&fi
End literal mode begun by
.Sx \&nf .
.Ss \&ft
Change the current font mode.
See
.Sx Text Decoration
for a listing of available font modes.
.Ss \&in
Indent relative to the current indentation:
.Pp
.D1 Pf \. Sx \&in Op Cm width
.Pp
If
.Cm width
is signed, the new offset is relative.
Otherwise, it is absolute.
This value is reset upon the next paragraph, section, or sub-section.
.Ss \&na
Don't align to the right margin.
.Ss \&nf
Begin literal mode: all subsequent free-form lines have their end of
line boundaries preserved.
May be ended by
.Sx \&fi .
Literal mode is implicitly ended by
.Sx \&SH
or
.Sx \&SS .
.Ss \&sp
Insert vertical spaces into output with the following syntax:
.Bd -filled -offset indent
.Pf \. Sx \&sp
.Op Cm height
.Ed
.Pp
Insert
```

```
.Cm height
spaces, which must conform to
.Sx Scaling Widths .
If 0, this is equivalent to the
.Sx \&br
macro.
Defaults to 1, if unspecified.
.Pp
See also
.Sx \&br .
.Sh MACRO SYNTAX
The
.Nm
macros are classified by scope: line scope or block scope.
Line macros are only scoped to the current line (and, in some
situations, the subsequent line).
Block macros are scoped to the current line and subsequent lines until
closed by another block macro.
.Ss Line Macros
Line macros are generally scoped to the current line, with the body
consisting of zero or more arguments.
If a macro is scoped to the next line and the line arguments are empty,
the next line, which must be text, is used instead.
Thus:
.Bd -literal -offset indent
\&.I
foo
.Ed
.Pp
is equivalent to
.Sq \&.I foo .
If next-line macros are invoked consecutively, only the last is used.
If a next-line macro is followed by a non-next-line macro, an error is
raised, except for
.Sx \&br ,
.Sx \&sp ,
and
.Sx \&na .
.Pp
The syntax is as follows:
.Bd -literal -offset indent
\&.YO \(lBbody...\(rB
\(lBbody...\(rB
.Ed
.Bl -column "MacroX" "ArgumentsX" "ScopeXXXXX" "CompatX" -offset indent
.It Em Macro Ta Em Arguments Ta Em Scope     Ta Em Notes
.It Sx \&AT   Ta    <=1        Ta    current   Ta    \&
.It Sx \&B    Ta    n          Ta    next-line Ta    \&
.It Sx \&BI   Ta    n          Ta    current   Ta    \&
.It Sx \&BR   Ta    n          Ta    current   Ta    \&
.It Sx \&DT   Ta    0          Ta    current   Ta    \&
.It Sx \&I    Ta    n          Ta    next-line Ta    \&
.It Sx \&IB   Ta    n          Ta    current   Ta    \&
.It Sx \&IR   Ta    n          Ta    current   Ta    \&
.It Sx \&OP   Ta    0, 1       Ta    current   Ta    compat
.It Sx \&R    Ta    n          Ta    next-line Ta    \&
.It Sx \&RB   Ta    n          Ta    current   Ta    \&
.It Sx \&RI   Ta    n          Ta    current   Ta    \&
.It Sx \&SB   Ta    n          Ta    next-line Ta    \&
.It Sx \&SM   Ta    n          Ta    next-line Ta    \&
.It Sx \&TH   Ta    >1, <6     Ta    current   Ta    \&
.It Sx \&UC   Ta    <=1        Ta    current   Ta    \&
.It Sx \&br   Ta    0          Ta    current   Ta    compat
.It Sx \&fi   Ta    0          Ta    current   Ta    compat
.It Sx \&ft   Ta    1          Ta    current   Ta    compat
.It Sx \&in   Ta    1          Ta    current   Ta    compat
```



```
.It Sx \&na   Ta    0          Ta    current   Ta    compat
.It Sx \&nf   Ta    0          Ta    current   Ta    compat
.It Sx \&sp   Ta    1          Ta    current   Ta    compat
.El
.Pp
Macros marked as
.Qq compat
are included for compatibility with the significant corpus of existing
manuals that mix dialects of roff.
These macros should not be used for portable
.Nm
manuals.
.Ss Block Macros
Block macros comprise a head and body.
As with in-line macros, the head is scoped to the current line and, in
one circumstance, the next line (the next-line stipulations as in
.Sx Line Macros
apply here as well).
.Pp
The syntax is as follows:
.Bd -literal -offset indent
\&.YO \(lBhead...\(rB
\(lBhead...\(rB
\(lBbody...\(rB
.Ed
.Pp
The closure of body scope may be to the section, where a macro is closed
by
.Sx \&SH ;
sub-section, closed by a section or
.Sx \&SS ;
part, closed by a section, sub-section, or
.Sx \&RE ;
or paragraph, closed by a section, sub-section, part,
.Sx \&HP ,
.Sx \&IP ,
.Sx \&LP ,
.Sx \&P ,
.Sx \&PP ,
or
.Sx \&TP .
No closure refers to an explicit block closing macro.
.Pp
As a rule, block macros may not be nested; thus, calling a block macro
while another block macro scope is open, and the open scope is not
implicitly closed, is syntactically incorrect.
.Bl -column "MacroX" "ArgumentsX" "Head ScopeX" "sub-sectionX" "compatX" -offset
.It Em Macro Ta Em Arguments Ta Em Head Scope Ta Em Body Scope  Ta Em Notes
.It Sx \&HP   Ta    <2         Ta    current    Ta    paragraph   Ta    \&
.It Sx \&IP   Ta    <3         Ta    current    Ta    paragraph   Ta    \&
.It Sx \&LP   Ta    0          Ta    current    Ta    paragraph   Ta    \&
.It Sx \&P    Ta    0          Ta    current    Ta    paragraph   Ta    \&
.It Sx \&PP   Ta    0          Ta    current    Ta    paragraph   Ta    \&
.It Sx \&RE   Ta    0          Ta    current    Ta    none        Ta    compat
.It Sx \&RS   Ta    1          Ta    current    Ta    part        Ta    compat
.It Sx \&SH   Ta    >0         Ta    next-line  Ta    section     Ta    \&
.It Sx \&SS   Ta    >0         Ta    next-line  Ta    sub-section Ta    \&
.It Sx \&TP   Ta    n          Ta    next-line  Ta    paragraph   Ta    \&
.El
.Pp
Macros marked
.Qq compat
are as mentioned in
.Sx Line Macros .
.Pp
If a block macro is next-line scoped, it may only be followed by in-line
```

```
787 macros for decorating text.
788 .Ss Font handling
789 In
790 .Nm
791 documents, both
792 .Sx Physical markup
793 macros and
794 .Xr roff 5
795 .Ql \ef
796 font escape sequences can be used to choose fonts.
797 In text lines, the effect of manual font selection by escape sequences
798 only lasts until the next macro invocation; in macro lines, it only lasts
799 until the end of the macro scope.
800 Note that macros like
801 .Sx \&BR
802 open and close a font scope for each argument.
803 .Sh COMPATIBILITY
804 This section documents areas of questionable portability between
805 implementations of the
806 .Nm
807 language.
808 .Pp
809 .Bl -dash -compact
810 .It
811 Do not depend on
812 .Sx \&SH
813 or
814 .Sx \&SS
815 to close out a literal context opened with
816 .Sx \&nf .
817 This behaviour may not be portable.
818 .It
819 In quoted literals, GNU troff allowed pair-wise double-quotes to produce
820 a standalone double-quote in formatted output.
821 It is not known whether this behaviour is exhibited by other formatters.
822 .It
823 troff suppresses a newline before
824 .Sq \(aq
825 macro output; in mandoc, it is an alias for the standard
826 .Sq \&.
827 control character.
828 .It
829 The
830 .Sq \eh
831 .Pq horizontal position ,
832 .Sq \ev
833 .Pq vertical position ,
834 .Sq \em
835 .Pq text colour ,
836 .Sq \eM
837 .Pq text filling colour ,
838 .Sq \ez
839 .Pq zero-length character ,
840 .Sq \ew
841 .Pq string length ,
842 .Sq \ek
843 .Pq horizontal position marker ,
844 .Sq \eo
845 .Pq text overstrike ,
846 and
847 .Sq \es
848 .Pq text size
849 escape sequences are all discarded in mandoc.
850 .It
851 The
852 .Sq \ef
```



```
853 scaling unit is accepted by mandoc, but rendered as the default unit.
854 .It
855 The
856 .Sx \&sp
857 macro does not accept negative values in mandoc.
858 In GNU troff, this would result in strange behaviour.
859 .It
860 In page header lines, GNU troff versions up to and including 1.21
861 only print
862 .Ar volume
863 names explicitly specified in the
864 .Sx \&TH
865 macro; mandoc and newer groff print the default volume name
866 corresponding to the
867 .Ar section
868 number when no
869 .Ar volume
870 is given, like in
871 .Xr mdoc 5 .
872 .El
873 .Pp
874 The
875 .Sx OP
876 macro is part of the extended
877 .Nm
878 macro set, and may not be portable to non-GNU troff implementations.
879 .Sh INTERFACE STABILITY
880 .Nm "Obsolete Committed" .
881 .Sh SEE ALSO
882 .Xr man 1 ,
883 .Xr mandoc 1 ,
884 .Xr eqn 5 ,
885 .Xr mandoc_char 5 ,
886 .Xr mdoc 5 ,
887 .Xr roff 5 ,
888 .Xr tbl 5
889 .Sh HISTORY
890 The
891 .Nm
892 language first appeared as a macro package for the roff typesetting
893 system in
894 .At v7 .
895 It was later rewritten by James Clark as a macro package for groff.
896 Eric S. Raymond wrote the extended
897 .Nm
898 macros for groff in 2007.
899 The stand-alone implementation that is part of the
900 .Xr mandoc 1
901 utility written by Kristaps Dzonsons appeared in
902 .Ox 4.6 .
903 .Sh AUTHORS
904 This
905 .Nm
906 reference was written by
907 .An Kristaps Dzonsons ,
908 .Mt kristaps@bsd.lv .
909 .Sh CAVEATS
910 Do not use this language.
911 Use
912 .Xr mdoc 5 ,
913 instead.
  1 '\" te
  2 .\" Copyright (c) 1995, Sun Microsystems, Inc.
  3 .\" The contents of this file are subject to the terms of the Common Development
  4 .\" You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE or http:
  5 .\" When distributing Covered Code, include this CDDL HEADER in each file and in
```

```
.TH MAN 5 "Jan 30, 1995"
.SH NAME
man \- macros to format Reference Manual pages
.SH SYNOPSIS
.LP
.nf
\fBnroff\fR \fB-man\fR \fIfilename\fR...
.fi

.LP
.nf
\fBtroff\fR  \fB-man\fR \fIfilename\fR...
.fi

.SH DESCRIPTION
.sp
.LP
These macros are used to lay out the reference pages in this manual. Note: if
\fIfilename\fR contains format input for a preprocessor, the commands shown
above must be piped through the appropriate preprocessor. This is handled
automatically by the \fBman\fR(1) command. See the ‘‘Conventions’’ section.
.sp
.LP
Any text argument \fIt\fR may be zero to six words. Quotes may be used to
include SPACE characters in a "word".  If \fItext\fR is empty, the special
treatment is applied to the next input line with text to be printed. In this
way \fB\&.I\fR may be used to italicize a whole line, or \fB\&.SB\fR may be
used to make small bold letters.
.sp
.LP
A prevailing indent distance is remembered between successive indented
paragraphs, and is reset to default value upon reaching a non-indented
paragraph.  Default units for indents \fIi\fR are ens.
.sp
.LP
Type font and size are reset to default values before each paragraph, and after
processing font and size setting macros.
.sp
.LP
These strings are predefined by \fB-man\fR:
.sp
.ne 2
.na
\fB\fB\e*R\fR\fR
.ad
.RS 8n
‘\(rg’, ‘(Reg)’ in \fBnroff\fR.
.RE

.sp
.ne 2
.na
\fB\fB\e*S\fR\fR
.ad
.RS 8n
Change to default type size.
.RE

.SS "Requests"
.sp
.LP
* n.t.l. = next text line; p.i. = prevailing indent
.sp

.sp
.TS
```



```
c c c c
c c c c .
\fIRequest\fR   \fICause\fR     \fIIf no\fR     \fIExplanation\fR
        \fIBreak\fR     \fIArgument\fR
\fB\&.B \fR\fIt\fR      no      \fIt\fR=n.t.l.* Text is in bold font.
\fB\&.BI \fR\fIt\fR     no      \fIt\fR=n.t.l.  Join words, alternating bold and
\fB\&.BR \fR\fIt\fR     no      \fIt\fR=n.t.l.  Join words, alternating bold and
\fB\&.DT\fR     no      \&.5i 1i...     Restore default tabs.
\fB\&.HP \fR\fIi\fR     yes     \fIi\fR=p.i.*   T{
Begin paragraph with hanging indent. Set prevailing indent to \fIi\fR.
T}
\fB\&.I \fR\fIt\fR      no      \fIt\fR=n.t.l.  Text is italic.
\fB\&.IB \fR\fIt\fR     no      \fIt\fR=n.t.l.  Join words, alternating italic a
\fB\&.IP \fR\fIx i\fR   yes     \fIx\fR=""      Same as \fB\&.TP\fR with tag \fI
\fB\&.IR \fR\fIt\fR     no      \fIt\fR=n.t.l.  T{
Join words, alternating italic and roman.
T}
\fB\&.IX \fR\fIt\fR     no      -       Index macro, for SunSoft internal use.
\fB\&.LP\fR     yes     -       T{
Begin left-aligned paragraph. Set prevailing indent to .5i.
T}
\fB\&.P\fR      yes     -       Same as \fB\&.LP\fR.
\fB\&.PD \fR\fId\fR     no      \fId\fR=.4v     T{
Set vertical distance between paragraphs.
T}
\fB\&.PP\fR     yes     -       Same as \fB\&.LP\fR.
\fB\&.RE\fR     yes     -       T{
End of relative indent. Restores prevailing indent.
T}
\fB\&.RB \fR\fIt\fR     no      \fIt\fR=n.t.l.  Join words, alternating roman an
\fB\&.RI \fR\fIt\fR     no      \fIt\fR=n.t.l.  T{
Join words, alternating roman and italic.
T}
\fB\&.RS \fR\fIi\fR     yes     \fIi\fR=p.i.    T{
Start relative indent, increase indent by \fIi\fR. Sets prevailing indent to .5i
T}
\fB\&.SB \fR\fIt\fR     no      -       T{
Reduce size of text by 1 point, make text bold.
T}
\fB\&.SH \fR\fIt\fR     yes     -       Section Heading.
\fB\&.SM \fR\fIt\fR     no      \fIt\fR=n.t.l.  Reduce size of text by 1 point.
\fB\&.SS \fR\fIt\fR     yes     \fIt\fR=n.t.l.  Section Subheading.
\fB\&.TH \FR\FIN S "f d, m\fR"
\fB\&.TH \fR\fIn s d f m\fR     yes     -       T{
Begin reference page \fIn\fR, of of section \fIs\fR; \fId\fR is the date of the
T}
\fB\&.TP \fR\fIi\fR     yes     \fIi\fR=p.i.    T{
Begin indented paragraph, with the tag given on the next text line. Set prevaili
T}
\fB\&.TX \fR\fIt \fR\fIp\fR     no      -       T{
Resolve the title abbreviation \fIt\fR; join to punctuation mark (or text) \fIp\
T}
.TE

.SS "Conventions"
.sp
.LP
When formatting a manual page, \fBman\fR examines the first line to determine
whether it requires special processing. For example a first line consisting of:
.sp
.LP
\fB\&’\e" t\fR
.sp
.LP
indicates that the manual page must be run through the \fBtbl\fR(1)
preprocessor.
```

```
.sp
.LP
A typical manual page for a command or function is laid out as follows:
.sp
.ne 2
.na
\fB\&.TH\fI TITLE \FR[1-9]\FR " , "
.ad
.RS 23n
The name of the command or function, which serves as the title of the manual
page. This is followed by the number of the section in which it appears.
.RE

.sp
.ne 2
.na
\fB\&.SH NAME\fR
.ad
.RS 23n
The name, or list of names, by which the command is called, followed by a dash
and then a one-line summary of the action performed. All in roman font, this
section contains no \fBtroff\fR(1) commands or escapes, and no macro requests.
It is used to generate the \fBwindex\fR database, which is used by the
\fBwhatis\fR(1) command.
.RE

.sp
.ne 2
.na
\fB\&.SH SYNOPSIS\fR
.ad
.RS 23n
.sp
.ne 2
.na
\fBCommands:\fR
.ad
.RS 13n
The syntax of the command and its arguments, as typed on the command line.
When in boldface, a word must be typed exactly as printed.  When in italics, a
word can be replaced with an argument that you supply. References to bold or
italicized items are not capitalized in other sections, even when they begin a
sentence.
.sp
Syntactic symbols appear in roman face:
.sp
.ne 2
.na
\fB[ ]\fR
.ad
.RS 13n
An argument, when surrounded by brackets is optional.
.RE

.sp
.ne 2
.na
\fB|\fR
.ad
.RS 13n
Arguments separated by a vertical bar are exclusive. You can supply only one
item from such a list.
.RE

.sp
.ne 2
```



```
.na
\fB\&.\|.\|.\fR
.ad
.RS 13n
Arguments followed by an ellipsis can be repeated. When an ellipsis follows a
bracketed set, the expression within the brackets can be repeated.
.RE

.RE

.sp
.ne 2
.na
\fBFunctions:\fR
.ad
.RS 14n
If required, the data declaration, or \fB#include\fR directive, is shown first,
followed by the  function declaration. Otherwise, the function declaration is
shown.
.RE

.RE

.sp
.ne 2
.na
\fB\&.SH DESCRIPTION\fR
.ad
.RS 23n
A narrative overview of the command or function's external behavior. This
includes how it interacts with files or data, and how it handles the standard
input, standard output and standard error. Internals and implementation details
are normally omitted. This section attempts to provide a succinct overview in
answer to the question, "what does it do?"
.sp
Literal text from the synopsis appears in constant width, as do literal
filenames and references to items that appear elsewhere in the  reference
manuals. Arguments are italicized.
.sp
If a command interprets either subcommands or an input grammar, its command
interface or input grammar is normally described in a \fBUSAGE\fR section,
which follows the \fBOPTIONS\fR section.  The  \fBDESCRIPTION\fR section only
describes the behavior of the command itself, not that of subcommands.
.RE

.sp
.ne 2
.na
\fB\&.SH OPTIONS\fR
.ad
.RS 23n
The list of options along with a description of how each affects the command's
operation.
.RE

.sp
.ne 2
.na
\fB\&.SH RETURN VALUES\fR
.ad
.RS 23n
A list of the values the library routine will return to the calling  program
and the conditions that cause these values to be returned.
.RE

.sp
```

```
.ne 2
.na
\fB\&.SH EXIT STATUS\fR
.ad
.RS 23n
A list of the values the utility will return to the calling  program or shell,
and the conditions that cause these values to be  returned.
.RE

.sp
.ne 2
.na
\fB\&.SH FILES\fR
.ad
.RS 23n
A list of files associated with the command or function.
.RE

.sp
.ne 2
.na
\fB\&.SH SEE ALSO\fR
.ad
.RS 23n
A comma-separated list of related manual pages, followed by references to other
published materials.
.RE

.sp
.ne 2
.na
\fB\&.SH DIAGNOSTICS\fR
.ad
.RS 23n
A list of diagnostic messages and an explanation of each.
.RE

.sp
.ne 2
.na
\fB\&.SH BUGS\fR
.ad
.RS 23n
A description of limitations, known defects, and possible problems associated
with the command or function.
.RE

.SH FILES
.sp
.ne 2
.na
\fB\fB/usr/share/lib/tmac/an\fR \fR
.ad
.RS 27n

.RE

.sp
.ne 2
.na
\fB\fB/usr/share/man/windex\fR\fR
.ad
.RS 27n

.RE
```



```
.SH SEE ALSO
.sp
.LP
\fBman\fR(1), \fBnroff\fR(1), \fBtroff\fR(1), \fBwhatis\fR(1)
.sp
.LP
Dale Dougherty and   Tim O’Reilly, \fIUnix\fR \fIText\fR \fIProcessing\fR
```

```
**********************************************************
   26074 Wed Jul 16 14:05:09 2014
new/usr/src/man/man5/mandoc_char.5
mandoc import
**********************************************************
```

```
.\"
.\" Permission to use, copy, modify, and distribute this software for any
.\" purpose with or without fee is hereby granted, provided that the above
.\" copyright notice and this permission notice appear in all copies.
.\"
.\" THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES
.\" WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF
.\" MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR
.\" ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES
.\" WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN
.\" ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF
.\" OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
.\"
.\"
.\" Copyright (c) 2003 Jason McIntyre <jmc@openbsd.org>
.\" Copyright (c) 2009, 2010, 2011 Kristaps Dzonsons <kristaps@bsd.lv>
.\" Copyright (c) 2011 Ingo Schwarze <schwarze@openbsd.org>
.\" Copyright 2012 Nexenta Systems, Inc. All rights reserved.
.\"
.Dd Nov 23, 2011
.Dt MANDOC_CHAR 5
.Os
.Sh NAME
.Nm mandoc_char
.Nd mandoc special characters
.Sh DESCRIPTION
This page documents the
.Xr roff 5
escape sequences accepted by
.Xr mandoc 1
to represent special characters in
.Xr mdoc 5
and
.Xr man 5
documents.
.Pp
The rendering depends on the
.Xr mandoc 1
output mode; in ASCII output, most characters are completely
unintelligible.
For that reason, using any of the special characters documented here,
except those discussed in the
.Sx DESCRIPTION ,
is strongly discouraged; they are supported merely for backwards
compatibility with existing documents.
.Pp
In particular, in English manual pages, do not use special-character
escape sequences to represent national language characters in author
names; instead, provide ASCII transcriptions of the names.
.Ss Dashes and Hyphens
In typography there are different types of dashes of various width:
the hyphen (-),
the minus sign (\-),
the en-dash (\(en),
and the em-dash (\(em).
.Pp
Hyphens are used for adjectives;
to separate the two parts of a compound word;
or to separate a word across two successive lines of text.
The hyphen does not need to be escaped:
.Bd -unfilled -offset indent
```



```
blue-eyed
lorry-driver
.Ed
.Pp
The mathematical minus sign is used for negative numbers or subtraction.
It should be written as
.Sq \e- :
.Bd -unfilled -offset indent
a = 3 \e- 1;
b = \e-2;
.Ed
.Pp
The en-dash is used to separate the two elements of a range,
or can be used the same way as an em-dash.
It should be written as
.Sq \e(en :
.Bd -unfilled -offset indent
pp. 95\e(en97.
Go away \e(en or else!
.Ed
.Pp
The em-dash can be used to show an interruption
or can be used the same way as colons, semi-colons, or parentheses.
It should be written as
.Sq \e(em :
.Bd -unfilled -offset indent
Three things \e(em apples, oranges, and bananas.
This is not that \e(em rather, this is that.
.Ed
.Pp
Note:
hyphens, minus signs, and en-dashes look identical under normal ASCII output.
Other formats, such as PostScript, render them correctly,
with differing widths.
.Ss Spaces
To separate words in normal text, for indenting and alignment
in literal context, and when none of the following special cases apply,
just use the normal space character
.Pq Sq \  .
.Pp
When filling text, lines may be broken between words, i.e. at space
characters.
To prevent a line break between two particular words,
use the non-breaking space escape sequence
.Pq Sq \e~
instead of the normal space character.
For example, the input string
.Dq number\e~1
will be kept together as
.Dq number\~1
on the same output line.
.Pp
On request and macro lines, the normal space character serves as an
argument delimiter.
To include whitespace into arguments, quoting is usually the best choice.
In some cases, using either the non-breaking
.Pq Sq \e~
or the breaking
.Pq Sq \e\ \&
space escape sequence may be preferable.
To escape macro names and to protect whitespace at the end
of input lines, the zero-width space
.Pq Sq \e&
is often useful.
For example, in
.Xr mdoc 5 ,
```

```
a normal space character can be displayed in single quotes in either
of the following ways:
.Pp
.Dl .Sq \(dq \(dq
.Dl .Sq \e \e&
.Ss Quotes
On request and macro lines, the double-quote character
.Pq Sq \(dq
is handled specially to allow quoting.
One way to prevent this special handling is by using the
.Sq \e(dq
escape sequence.
.Pp
Note that on text lines, literal double-quote characters can be used
verbatim.
All other quote-like characters can be used verbatim as well,
even on request and macro lines.
.Ss Periods
The period
.Pq Sq \&.
is handled specially at the beginning of an input line,
where it introduces a
.Xr roff 5
request or a macro, and when appearing alone as a macro argument in
.Xr mdoc 5 .
In such situations, prepend a zero-width space
.Pq Sq \e&.
to make it behave like normal text.
.Pp
Do not use the
.Sq \e.
escape sequence.
It does not prevent special handling of the period.
.Ss Backslashes
To include a literal backslash
.Pq Sq \e
into the output, use the
.Pq Sq \ee
escape sequence.
.Pp
Note that doubling it
.Pq Sq \e\e
is not the right way to output a backslash.
Because
.Xr mandoc 1
does not implement full
.Xr roff 5
functionality, it may work with
.Xr mandoc 1 ,
but it may have weird effects on complete
.Xr roff 5
implementations.
.Sh SPECIAL CHARACTERS
Special characters are encoded as
.Sq \eX
.Pq for a one-character escape ,
.Sq \e(XX
.Pq two-character ,
and
.Sq \e[N]
.Pq N-character .
For details, see the
.Em Special Characters
subsection of the
.Xr roff 5
manual.
```



```
.Pp
Spacing:
.Bl -column "Input" "Description" -offset indent -compact
.It Em Input Ta Em Description
.It \e~      Ta non-breaking, non-collapsing space
.It \e       Ta breaking, non-collapsing n-width space
.It \e^      Ta zero-width space
.It \e%      Ta zero-width space
.It \e&      Ta zero-width space
.It \e|      Ta zero-width space
.It \e0      Ta breaking, non-collapsing digit-width space
.It \ec      Ta removes any trailing space (if applicable)
.El
.Pp
Lines:
.Bl -column "Input" "Rendered" "Description" -offset indent -compact
.It Em Input Ta Em Rendered Ta Em Description
.It \e(ba    Ta \(ba        Ta bar
.It \e(br    Ta \(br        Ta box rule
.It \e(ul    Ta \(ul        Ta underscore
.It \e(rl    Ta \(rl        Ta overline
.It \e(bb    Ta \(bb        Ta broken bar
.It \e(sl    Ta \(sl        Ta forward slash
.It \e(rs    Ta \(rs        Ta backward slash
.El
.Pp
Text markers:
.Bl -column "Input" "Rendered" "Description" -offset indent -compact
.It Em Input Ta Em Rendered Ta Em Description
.It \e(ci    Ta \(ci        Ta circle
.It \e(bu    Ta \(bu        Ta bullet
.It \e(dd    Ta \(dd        Ta double dagger
.It \e(dg    Ta \(dg        Ta dagger
.It \e(lz    Ta \(lz        Ta lozenge
.It \e(sq    Ta \(sq        Ta white square
.It \e(ps    Ta \(ps        Ta paragraph
.It \e(sc    Ta \(sc        Ta section
.It \e(lh    Ta \(lh        Ta left hand
.It \e(rh    Ta \(rh        Ta right hand
.It \e(at    Ta \(at        Ta at
.It \e(sh    Ta \(sh        Ta hash (pound)
.It \e(CR    Ta \(CR        Ta carriage return
.It \e(OK    Ta \(OK        Ta check mark
.El
.Pp
Legal symbols:
.Bl -column "Input" "Rendered" "Description" -offset indent -compact
.It Em Input Ta Em Rendered Ta Em Description
.It \e(co    Ta \(co        Ta copyright
.It \e(rg    Ta \(rg        Ta registered
.It \e(tm    Ta \(tm        Ta trademarked
.El
.Pp
Punctuation:
.Bl -column "Input" "Rendered" "Description" -offset indent -compact
.It Em Input Ta Em Rendered Ta Em Description
.It \e(em    Ta \(em        Ta em-dash
.It \e(en    Ta \(en        Ta en-dash
.It \e(hy    Ta \(hy        Ta hyphen
.It \ee      Ta \e          Ta back-slash
.It \e.      Ta \.          Ta period
.It \e(r!    Ta \(r!        Ta upside-down exclamation
.It \e(r?    Ta \(r?        Ta upside-down question
.El
.Pp
Quotes:
```

```
.Bl -column "Input" "Rendered" "Description" -offset indent -compact
.It Em Input Ta Em Rendered Ta Em Description
.It \e(Bq    Ta \(Bq        Ta right low double-quote
.It \e(bq    Ta \(bq        Ta right low single-quote
.It \e(lq    Ta \(lq        Ta left double-quote
.It \e(rq    Ta \(rq        Ta right double-quote
.It \e(oq    Ta \(oq        Ta left single-quote
.It \e(cq    Ta \(cq        Ta right single-quote
.It \e(aq    Ta \(aq        Ta apostrophe quote (text)
.It \e(dq    Ta \(dq        Ta double quote (text)
.It \e(Fo    Ta \(Fo        Ta left guillemet
.It \e(Fc    Ta \(Fc        Ta right guillemet
.It \e(fo    Ta \(fo        Ta left single guillemet
.It \e(fc    Ta \(fc        Ta right single guillemet
.El
.Pp
Brackets:
.Bl -column "xxbracketrightbpx" Rendered Description -offset indent -compact
.It Em Input Ta Em Rendered Ta Em Description
.It \e(lB    Ta \(lB        Ta left bracket
.It \e(rB    Ta \(rB        Ta right bracket
.It \e(lC    Ta \(lC        Ta left brace
.It \e(rC    Ta \(rC        Ta right brace
.It \e(la    Ta \(la        Ta left angle
.It \e(ra    Ta \(ra        Ta right angle
.It \e(bv    Ta \(bv        Ta brace extension
.It \e[braceex] Ta \[braceex] Ta brace extension
.It \e[bracketlefttp] Ta \[bracketlefttp] Ta top-left hooked bracket
.It \e[bracketleftbp] Ta \[bracketleftbp] Ta bottom-left hooked bracket
.It \e[bracketleftex] Ta \[bracketleftex] Ta left hooked bracket extension
.It \e[bracketrighttp] Ta \[bracketrighttp] Ta top-right hooked bracket
.It \e[bracketrightbp] Ta \[bracketrightbp] Ta bottom-right hooked bracket
.It \e[bracketrightex] Ta \[bracketrightex] Ta right hooked bracket extension
.It \e(lt    Ta \(lt        Ta top-left hooked brace
.It \e[bracelefttp] Ta \[bracelefttp] Ta top-left hooked brace
.It \e(lk    Ta \(lk        Ta mid-left hooked brace
.It \e[braceleftmid] Ta \[braceleftmid] Ta mid-left hooked brace
.It \e(lb    Ta \(lb        Ta bottom-left hooked brace
.It \e[braceleftbp] Ta \[braceleftbp] Ta bottom-left hooked brace
.It \e[braceleftex] Ta \[braceleftex] Ta left hooked brace extension
.It \e(rt    Ta \(rt        Ta top-left hooked brace
.It \e[bracerighttp] Ta \[bracerighttp] Ta top-right hooked brace
.It \e(rk    Ta \(rk        Ta mid-right hooked brace
.It \e[bracerightmid] Ta \[bracerightmid] Ta mid-right hooked brace
.It \e(rb    Ta \(rb        Ta bottom-right hooked brace
.It \e[bracerightbp] Ta \[bracerightbp] Ta bottom-right hooked brace
.It \e[bracerightex] Ta \[bracerightex] Ta right hooked brace extension
.It \e[parenlefttp] Ta \[parenlefttp] Ta top-left hooked parenthesis
.It \e[parenleftbp] Ta \[parenleftbp] Ta bottom-left hooked parenthesis
.It \e[parenleftex] Ta \[parenleftex] Ta left hooked parenthesis extension
.It \e[parenrighttp] Ta \[parenrighttp] Ta top-right hooked parenthesis
.It \e[parenrightbp] Ta \[parenrightbp] Ta bottom-right hooked parenthesis
.It \e[parenrightex] Ta \[parenrightex] Ta right hooked parenthesis extension
.El
.Pp
Arrows:
.Bl -column "Input" "Rendered" "Description" -offset indent -compact
.It Em Input Ta Em Rendered Ta Em Description
.It \e(<-    Ta \(<-        Ta left arrow
.It \e(->    Ta \(->        Ta right arrow
.It \e(<>    Ta \(<>        Ta left-right arrow
.It \e(da    Ta \(da        Ta down arrow
.It \e(ua    Ta \(ua        Ta up arrow
.It \e(va    Ta \(va        Ta up-down arrow
.It \e(lA    Ta \(lA        Ta left double-arrow
.It \e(rA    Ta \(rA        Ta right double-arrow
```



```
.It \e(hA    Ta \(hA        Ta left-right double-arrow
.It \e(uA    Ta \(uA        Ta up double-arrow
.It \e(dA    Ta \(dA        Ta down double-arrow
.It \e(vA    Ta \(vA        Ta up-down double-arrow
.El
.Pp
Logical:
.Bl -column "Input" "Rendered" "Description" -offset indent -compact
.It Em Input Ta Em Rendered Ta Em Description
.It \e(AN    Ta \(AN        Ta logical and
.It \e(OR    Ta \(OR        Ta logical or
.It \e(no    Ta \(no        Ta logical not
.It \e[tno]  Ta \[tno]      Ta logical not (text)
.It \e(te    Ta \(te        Ta existential quantifier
.It \e(fa    Ta \(fa        Ta universal quantifier
.It \e(st    Ta \(st        Ta such that
.It \e(tf    Ta \(tf        Ta therefore
.It \e(3d    Ta \(3d        Ta therefore
.It \e(or    Ta \(or        Ta bitwise or
.El
.Pp
Mathematical:
.Bl -column "xxcoproductxx" "Rendered" "Description" -offset indent -compact
.It Em Input Ta Em Rendered Ta Em Description
.It \e(pl    Ta \(pl        Ta plus
.It \e(mi    Ta \(mi        Ta minus
.It \e-      Ta \-          Ta minus (text)
.It \e(-+    Ta \(-+        Ta minus-plus
.It \e(+-    Ta \(+-        Ta plus-minus
.It \e[t+-]  Ta \[t+-]      Ta plus-minus (text)
.It \e(pc    Ta \(pc        Ta centre-dot
.It \e(mu    Ta \(mu        Ta multiply
.It \e[tmu]  Ta \[tmu]      Ta multiply (text)
.It \e(c*    Ta \(c*        Ta circle-multiply
.It \e(c+    Ta \(c+        Ta circle-plus
.It \e(di    Ta \(di        Ta divide
.It \e[tdi]  Ta \[tdi]      Ta divide (text)
.It \e(f/    Ta \(f/        Ta fraction
.It \e(**    Ta \(**        Ta asterisk
.It \e(<=    Ta \(<=        Ta less-than-equal
.It \e(>=    Ta \(>=        Ta greater-than-equal
.It \e(<<    Ta \(<<        Ta much less
.It \e(>>    Ta \(>>        Ta much greater
.It \e(eq    Ta \(eq        Ta equal
.It \e(!=    Ta \(!=        Ta not equal
.It \e(==    Ta \(==        Ta equivalent
.It \e(ne    Ta \(ne        Ta not equivalent
.It \e(=~    Ta \(=~        Ta congruent
.It \e(-~    Ta \(-~        Ta asymptotically congruent
.It \e(ap    Ta \(ap        Ta asymptotically similar
.It \e(~~    Ta \(~~        Ta approximately similar
.It \e(~=    Ta \(~=        Ta approximately equal
.It \e(pt    Ta \(pt        Ta proportionate
.It \e(es    Ta \(es        Ta empty set
.It \e(mo    Ta \(mo        Ta element
.It \e(nm    Ta \(nm        Ta not element
.It \e(sb    Ta \(sb        Ta proper subset
.It \e(nb    Ta \(nb        Ta not subset
.It \e(sp    Ta \(sp        Ta proper superset
.It \e(nc    Ta \(nc        Ta not superset
.It \e(ib    Ta \(ib        Ta reflexive subset
.It \e(ip    Ta \(ip        Ta reflexive superset
.It \e(ca    Ta \(ca        Ta intersection
.It \e(cu    Ta \(cu        Ta union
.It \e(/_    Ta \(/_        Ta angle
.It \e(pp    Ta \(pp        Ta perpendicular
```

```
.It \e(is    Ta \(is        Ta integral
.It \e[integral] Ta \[integral] Ta integral
.It \e[sum]    Ta \[sum]   Ta summation
.It \e[product] Ta \[product] Ta product
.It \e[coproduct] Ta \[coproduct] Ta coproduct
.It \e(gr    Ta \(gr        Ta gradient
.It \e(sr    Ta \(sr        Ta square root
.It \e[sqrt] Ta \[sqrt]     Ta square root
.It \e(lc    Ta \(lc        Ta left-ceiling
.It \e(rc    Ta \(rc        Ta right-ceiling
.It \e(lf    Ta \(lf        Ta left-floor
.It \e(rf    Ta \(rf        Ta right-floor
.It \e(if    Ta \(if        Ta infinity
.It \e(Ah    Ta \(Ah        Ta aleph
.It \e(Im    Ta \(Im        Ta imaginary
.It \e(Re    Ta \(Re        Ta real
.It \e(pd    Ta \(pd        Ta partial differential
.It \e(-h    Ta \(-h        Ta Planck constant over 2\(*p
.It \e[12]   Ta \[12]       Ta one-half
.It \e[14]   Ta \[14]       Ta one-fourth
.It \e[34]   Ta \[34]       Ta three-fourths
.El
.Pp
Ligatures:
.Bl -column "Input" "Rendered" "Description" -offset indent -compact
.It Em Input Ta Em Rendered Ta Em Description
.It \e(ff    Ta \(ff        Ta ff ligature
.It \e(fi    Ta \(fi        Ta fi ligature
.It \e(fl    Ta \(fl        Ta fl ligature
.It \e(Fi    Ta \(Fi        Ta ffi ligature
.It \e(Fl    Ta \(Fl        Ta ffl ligature
.It \e(AE    Ta \(AE        Ta AE
.It \e(ae    Ta \(ae        Ta ae
.It \e(OE    Ta \(OE        Ta OE
.It \e(oe    Ta \(oe        Ta oe
.It \e(ss    Ta \(ss        Ta German eszett
.It \e(IJ    Ta \(IJ        Ta IJ ligature
.It \e(ij    Ta \(ij        Ta ij ligature
.El
.Pp
Accents:
.Bl -column "Input" "Rendered" "Description" -offset indent -compact
.It Em Input Ta Em Rendered Ta Em Description
.It \e(a"    Ta \(a"        Ta Hungarian umlaut
.It \e(a-    Ta \(a-        Ta macron
.It \e(a.    Ta \(a.        Ta dotted
.It \e(a^    Ta \(a^        Ta circumflex
.It \e(aa    Ta \(aa        Ta acute
.It \e'      Ta \'          Ta acute
.It \e(ga    Ta \(ga        Ta grave
.It \e`      Ta \`          Ta grave
.It \e(ab    Ta \(ab        Ta breve
.It \e(ac    Ta \(ac        Ta cedilla
.It \e(ad    Ta \(ad        Ta dieresis
.It \e(ah    Ta \(ah        Ta caron
.It \e(ao    Ta \(ao        Ta ring
.It \e(a~    Ta \(a~        Ta tilde
.It \e(ho    Ta \(ho        Ta ogonek
.It \e(ha    Ta \(ha        Ta hat (text)
.It \e(ti    Ta \(ti        Ta tilde (text)
.El
.Pp
Accented letters:
.Bl -column "Input" "Rendered" "Description" -offset indent -compact
.It Em Input Ta Em Rendered Ta Em Description
.It \e('A    Ta \('A        Ta acute A
```



```
.It \e('E    Ta \('E        Ta acute E
.It \e('I    Ta \('I        Ta acute I
.It \e('O    Ta \('O        Ta acute O
.It \e('U    Ta \('U        Ta acute U
.It \e('a    Ta \('a        Ta acute a
.It \e('e    Ta \('e        Ta acute e
.It \e('i    Ta \('i        Ta acute i
.It \e('o    Ta \('o        Ta acute o
.It \e('u    Ta \('u        Ta acute u
.It \e(`A    Ta \(`A        Ta grave A
.It \e(`E    Ta \(`E        Ta grave E
.It \e(`I    Ta \(`I        Ta grave I
.It \e(`O    Ta \(`O        Ta grave O
.It \e(`U    Ta \(`U        Ta grave U
.It \e(`a    Ta \(`a        Ta grave a
.It \e(`e    Ta \(`e        Ta grave e
.It \e(`i    Ta \(`i        Ta grave i
.It \e(`o    Ta \(`i        Ta grave o
.It \e(`u    Ta \(`u        Ta grave u
.It \e(~A    Ta \(~A        Ta tilde A
.It \e(~N    Ta \(~N        Ta tilde N
.It \e(~O    Ta \(~O        Ta tilde O
.It \e(~a    Ta \(~a        Ta tilde a
.It \e(~n    Ta \(~n        Ta tilde n
.It \e(~o    Ta \(~o        Ta tilde o
.It \e(:A    Ta \(:A        Ta dieresis A
.It \e(:E    Ta \(:E        Ta dieresis E
.It \e(:I    Ta \(:I        Ta dieresis I
.It \e(:O    Ta \(:O        Ta dieresis O
.It \e(:U    Ta \(:U        Ta dieresis U
.It \e(:a    Ta \(:a        Ta dieresis a
.It \e(:e    Ta \(:e        Ta dieresis e
.It \e(:i    Ta \(:i        Ta dieresis i
.It \e(:o    Ta \(:o        Ta dieresis o
.It \e(:u    Ta \(:u        Ta dieresis u
.It \e(:y    Ta \(:y        Ta dieresis y
.It \e(^A    Ta \(^A        Ta circumflex A
.It \e(^E    Ta \(^E        Ta circumflex E
.It \e(^I    Ta \(^I        Ta circumflex I
.It \e(^O    Ta \(^O        Ta circumflex O
.It \e(^U    Ta \(^U        Ta circumflex U
.It \e(^a    Ta \(^a        Ta circumflex a
.It \e(^e    Ta \(^e        Ta circumflex e
.It \e(^i    Ta \(^i        Ta circumflex i
.It \e(^o    Ta \(^o        Ta circumflex o
.It \e(^u    Ta \(^u        Ta circumflex u
.It \e(,C    Ta \(,C        Ta cedilla C
.It \e(,c    Ta \(,c        Ta cedilla c
.It \e(/L    Ta \(/L        Ta stroke L
.It \e(/l    Ta \(/l        Ta stroke l
.It \e(/O    Ta \(/O        Ta stroke O
.It \e(/o    Ta \(/o        Ta stroke o
.It \e(oA    Ta \(oA        Ta ring A
.It \e(oa    Ta \(oa        Ta ring a
.El
.Pp
Special letters:
.Bl -column "Input" "Rendered" "Description" -offset indent -compact
.It Em Input Ta Em Rendered Ta Em Description
.It \e(-D    Ta \(-D        Ta Eth
.It \e(Sd    Ta \(Sd        Ta eth
.It \e(TP    Ta \(TP        Ta Thorn
.It \e(Tp    Ta \(Tp        Ta thorn
.It \e(.i    Ta \(.i        Ta dotless i
.It \e(.j    Ta \(.j        Ta dotless j
.El
```

```
.Pp
Currency:
.Bl -column "Input" "Rendered" "Description" -offset indent -compact
.It Em Input Ta Em Rendered Ta Em Description
.It \e(Do    Ta \(Do        Ta dollar
.It \e(ct    Ta \(ct        Ta cent
.It \e(Eu    Ta \(Eu        Ta Euro symbol
.It \e(eu    Ta \(eu        Ta Euro symbol
.It \e(Ye    Ta \(Ye        Ta yen
.It \e(Po    Ta \(Po        Ta pound
.It \e(Cs    Ta \(Cs        Ta Scandinavian
.It \e(Fn    Ta \(Fn        Ta florin
.El
.Pp
Units:
.Bl -column "Input" "Rendered" "Description" -offset indent -compact
.It Em Input Ta Em Rendered Ta Em Description
.It \e(de    Ta \(de        Ta degree
.It \e(%0    Ta \(%0        Ta per-thousand
.It \e(fm    Ta \(fm        Ta minute
.It \e(sd    Ta \(sd        Ta second
.It \e(mc    Ta \(mc        Ta micro
.El
.Pp
Greek letters:
.Bl -column "Input" "Rendered" "Description" -offset indent -compact
.It Em Input Ta Em Rendered Ta Em Description
.It \e(*A    Ta \(*A        Ta Alpha
.It \e(*B    Ta \(*B        Ta Beta
.It \e(*G    Ta \(*G        Ta Gamma
.It \e(*D    Ta \(*D        Ta Delta
.It \e(*E    Ta \(*E        Ta Epsilon
.It \e(*Z    Ta \(*Z        Ta Zeta
.It \e(*Y    Ta \(*Y        Ta Eta
.It \e(*H    Ta \(*H        Ta Theta
.It \e(*I    Ta \(*I        Ta Iota
.It \e(*K    Ta \(*K        Ta Kappa
.It \e(*L    Ta \(*L        Ta Lambda
.It \e(*M    Ta \(*M        Ta Mu
.It \e(*N    Ta \(*N        Ta Nu
.It \e(*C    Ta \(*C        Ta Xi
.It \e(*O    Ta \(*O        Ta Omicron
.It \e(*P    Ta \(*P        Ta Pi
.It \e(*R    Ta \(*R        Ta Rho
.It \e(*S    Ta \(*S        Ta Sigma
.It \e(*T    Ta \(*T        Ta Tau
.It \e(*U    Ta \(*U        Ta Upsilon
.It \e(*F    Ta \(*F        Ta Phi
.It \e(*X    Ta \(*X        Ta Chi
.It \e(*Q    Ta \(*Q        Ta Psi
.It \e(*W    Ta \(*W        Ta Omega
.It \e(*a    Ta \(*a        Ta alpha
.It \e(*b    Ta \(*b        Ta beta
.It \e(*g    Ta \(*g        Ta gamma
.It \e(*d    Ta \(*d        Ta delta
.It \e(*e    Ta \(*e        Ta epsilon
.It \e(*z    Ta \(*z        Ta zeta
.It \e(*y    Ta \(*y        Ta eta
.It \e(*h    Ta \(*h        Ta theta
.It \e(*i    Ta \(*i        Ta iota
.It \e(*k    Ta \(*k        Ta kappa
.It \e(*l    Ta \(*l        Ta lambda
.It \e(*m    Ta \(*m        Ta mu
.It \e(*n    Ta \(*n        Ta nu
.It \e(*c    Ta \(*c        Ta xi
.It \e(*o    Ta \(*o        Ta omicron
```



```
.It \e(*p    Ta \(*p        Ta pi
.It \e(*r    Ta \(*r        Ta rho
.It \e(*s    Ta \(*s        Ta sigma
.It \e(*t    Ta \(*t        Ta tau
.It \e(*u    Ta \(*u        Ta upsilon
.It \e(*f    Ta \(*f        Ta phi
.It \e(*x    Ta \(*x        Ta chi
.It \e(*q    Ta \(*q        Ta psi
.It \e(*w    Ta \(*w        Ta omega
.It \e(+h    Ta \(+h        Ta theta variant
.It \e(+f    Ta \(+f        Ta phi variant
.It \e(+p    Ta \(+p        Ta pi variant
.It \e(+e    Ta \(+e        Ta epsilon variant
.It \e(ts    Ta \(ts        Ta sigma terminal
.El
.Sh PREDEFINED STRINGS
Predefined strings are inherited from the macro packages of historical
troff implementations.
They are
.Em not recommended
for use, as they differ across implementations.
Manuals using these predefined strings are almost certainly not
portable.
.Pp
Their syntax is similar to special characters, using
.Sq \e*X
.Pq for a one-character escape ,
.Sq \e*(XX
.Pq two-character ,
and
.Sq \e*[N]
.Pq N-character .
For details, see the
.Em Predefined Strings
subsection of the
.Xr roff 5
manual.
.Bl -column "Input" "Rendered" "Description" -offset indent
.It Em Input Ta Em Rendered Ta Em Description
.It \e*(Ba   Ta \*(Ba       Ta vertical bar
.It \e*(Ne   Ta \*(Ne       Ta not equal
.It \e*(Ge   Ta \*(Ge       Ta greater-than-equal
.It \e*(Le   Ta \*(Le       Ta less-than-equal
.It \e*(Gt   Ta \*(Gt       Ta greater-than
.It \e*(Lt   Ta \*(Lt       Ta less-than
.It \e*(Pm   Ta \*(Pm       Ta plus-minus
.It \e*(If   Ta \*(If       Ta infinity
.It \e*(Pi   Ta \*(Pi       Ta pi
.It \e*(Na   Ta \*(Na       Ta NaN
.It \e*(Am   Ta \*(Am       Ta ampersand
.It \e*R     Ta \*R         Ta restricted mark
.It \e*(Tm   Ta \*(Tm       Ta trade mark
.It \e*q     Ta \*q         Ta double-quote
.It \e*(Rq   Ta \*(Rq       Ta right-double-quote
.It \e*(Lq   Ta \*(Lq       Ta left-double-quote
.It \e*(lp   Ta \*(lp       Ta right-parenthesis
.It \e*(rp   Ta \*(rp       Ta left-parenthesis
.It \e*(lq   Ta \*(lq       Ta left double-quote
.It \e*(rq   Ta \*(rq       Ta right double-quote
.It \e*(ua   Ta \*(ua       Ta up arrow
.It \e*(va   Ta \*(va       Ta up-down arrow
.It \e*(<=   Ta \*(<=       Ta less-than-equal
.It \e*(>=   Ta \*(>=       Ta greater-than-equal
.It \e*(aa   Ta \*(aa       Ta acute
.It \e*(ga   Ta \*(ga       Ta grave
.It \e*(Px   Ta \*(Px       Ta POSIX standard name
```

```
.It \e*(Ai   Ta \*(Ai       Ta ANSI standard name
.El
.Sh UNICODE CHARACTERS
The escape sequence
.Pp
.Dl \e[uXXXX]
.Pp
is interpreted as a Unicode codepoint.
The codepoint must be in the range above U+0080 and less than U+10FFFF.
For compatibility, points must be zero-padded to four characters; if
greater than four characters, no zero padding is allowed.
Unicode surrogates are not allowed.
.\" .Pp
.\" Unicode glyphs attenuate to the
.\" .Sq \&?
.\" character if invalid or not rendered by current output media.
.Sh NUMBERED CHARACTERS
For backward compatibility with existing manuals,
.Xr mandoc 1
also supports the
.Pp
.Dl \eN\(aq Ns Ar number Ns \(aq
.Pp
escape sequence, inserting the character
.Ar number
from the current character set into the output.
Of course, this is inherently non-portable and is already marked
as deprecated in the Heirloom roff manual.
For example, do not use \eN’34’, use \e(dq, or even the plain
.Sq \(dq
character where possible.
.Sh COMPATIBILITY
This section documents compatibility between mandoc and other other
troff implementations, at this time limited to GNU troff
.Pq Qq groff .
.Pp
.Bl -dash -compact
.It
The \eN\(aq\(aq escape sequence is limited to printable characters; in
groff, it accepts arbitrary character numbers.
.It
In
.Fl T Ns Cm ascii ,
the
\e(ss, \e(nm, \e(nb, \e(nc, \e(ib, \e(ip, \e(pp, \e[sum], \e[product],
\e[coproduct], \e(gr, \e(\-h, and \e(a. special characters render
differently between mandoc and groff.
.It
In
.Fl T Ns Cm html
and
.Fl T Ns Cm xhtml ,
the \e(~=, \e(nb, and \e(nc special characters render differently
between mandoc and groff.
.It
The
.Fl T Ns Cm ps
and
.Fl T Ns Cm pdf
modes format like
.Fl T Ns Cm ascii
instead of rendering glyphs as in groff.
.It
The \e[radicalex], \e[sqrtex], and \e(ru special characters have been omitted
from mandoc either because they are poorly documented or they have no
known representation.
```



```
.El
.Sh SEE ALSO
.Xr mandoc 1 ,
.Xr man 5 ,
.Xr mdoc 5 ,
.Xr roff 5
.Sh AUTHORS
The
.Nm
manual page was written by
.An Kristaps Dzonsons ,
.Mt kristaps@bsd.lv .
.Sh CAVEATS
The
.Sq \e*(Ba
escape mimics the behaviour of the
.Sq \&|
character in
.Xr mdoc 5 ;
thus, if you wish to render a vertical bar with no side effects, use
the
.Sq \e(ba
escape.
```

```
**********************************************************
   23887 Wed Jul 16 14:05:09 2014
new/usr/src/man/man5/mandoc_roff.5
mandoc import
**********************************************************
```

```
.\"
.\" Permission to use, copy, modify, and distribute this software for any
.\" purpose with or without fee is hereby granted, provided that the above
.\" copyright notice and this permission notice appear in all copies.
.\"
.\" THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES
.\" WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF
.\" MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR
.\" ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES
.\" WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN
.\" ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF
.\" OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
.\"
.\"
.\" Copyright (c) 2010, 2011 Kristaps Dzonsons <kristaps@bsd.lv>
.\" Copyright (c) 2010, 2011 Ingo Schwarze <schwarze@openbsd.org>
.\" Copyright 2012 Nexenta Systems, Inc. All rights reserved.
.\" Copyright 2014 Garrett D'Amore <garrett@damore.org>
.\"
.Dd Jul 13, 2014
.Dt MANDOC_ROFF 5
.Os
.Sh NAME
.Nm mandoc_roff
.Nd roff language reference for mandoc
.Sh DESCRIPTION
The
.Nm roff
language is a general purpose text formatting language.
Since traditional implementations of the
.Xr mdoc 5
and
.Xr man 5
manual formatting languages are based on it,
many real-world manuals use small numbers of
.Nm
requests intermixed with their
.Xr mdoc 5
or
.Xr man 5
code.
To properly format such manuals, the
.Xr mandoc 1
utility supports a tiny subset of
.Nm
requests.
Only these requests supported by
.Xr mandoc 1
are documented in the present manual,
together with the basic language syntax shared by
.Nm ,
.Xr mdoc 5 ,
and
.Xr man 5 .
For complete
.Nm
manuals, consult the
.Sx SEE ALSO
section.
.Pp
Input lines beginning with the control character
```



```
.Sq \&.
are parsed for requests and macros.
Such lines are called
.Dq request lines
or
.Dq macro lines ,
respectively.
Requests change the processing state and manipulate the formatting;
some macros also define the document structure and produce formatted
output.
The single quote
.Pq Qq \(aq
is accepted as an alternative control character,
treated by
.Xr mandoc 1
just like
.Ql \&.
.Pp
Lines not beginning with control characters are called
.Dq text lines .
They provide free-form text to be printed; the formatting of the text
depends on the respective processing context.
.Sh LANGUAGE SYNTAX
.Nm
documents may contain only graphable 7-bit ASCII characters, the space
character, and, in certain circumstances, the tab character.
The back-space character
.Sq \e
indicates the start of an escape sequence for
.Sx Comments ,
.Sx Special Characters ,
.Sx Predefined Strings ,
and
user-defined strings defined using the
.Sx ds
request.
.Ss Comments
Text following an escaped double-quote
.Sq \e\(dq ,
whether in a request, macro, or text line, is ignored to the end of the line.
A request line beginning with a control character and comment escape
.Sq \&.\e\(dq
is also ignored.
Furthermore, request lines with only a control character and optional
trailing whitespace are stripped from input.
.Pp
Examples:
.Bd -literal -offset indent -compact
\&.\e\(dq This is a comment line.
\&.\e\(dq The next line is ignored:
\&.
\&.Sh EXAMPLES \e\(dq This is a comment, too.
\&example text \e\(dq And so is this.
.Ed
.Ss Special Characters
Special characters are used to encode special glyphs and are rendered
differently across output media.
They may occur in request, macro, and text lines.
Sequences begin with the escape character
.Sq \e
followed by either an open-parenthesis
.Sq \&(
for two-character sequences; an open-bracket
.Sq \&[
for n-character sequences (terminated at a close-bracket
.Sq \&] ) ;
```

```
or a single one character sequence.
.Pp
Examples:
.Bl -tag -width Ds -offset indent -compact
.It Li \e(em
Two-letter em dash escape.
.It Li \ee
One-letter backslash escape.
.El
.Pp
See
.Xr mandoc_char 5
for a complete list.
.Ss Text Decoration
Terms may be text-decorated using the
.Sq \ef
escape followed by an indicator: B (bold), I (italic), R (regular), or P
(revert to previous mode).
A numerical representation 3, 2, or 1 (bold, italic, and regular,
respectively) may be used instead.
The indicator or numerical representative may be preceded by C
(constant-width), which is ignored.
.Pp
Examples:
.Bl -tag -width Ds -offset indent -compact
.It Li \efBbold\efR
Write in bold, then switch to regular font mode.
.It Li \efIitalic\efP
Write in italic, then return to previous font mode.
.El
.Pp
Text decoration is
.Em not
recommended for
.Xr mdoc 5 ,
which encourages semantic annotation.
.Ss Predefined Strings
Predefined strings, like
.Sx Special Characters ,
mark special output glyphs.
Predefined strings are escaped with the slash-asterisk,
.Sq \e* :
single-character
.Sq \e*X ,
two-character
.Sq \e*(XX ,
and N-character
.Sq \e*[N] .
.Pp
Examples:
.Bl -tag -width Ds -offset indent -compact
.It Li \e*(Am
Two-letter ampersand predefined string.
.It Li \e*q
One-letter double-quote predefined string.
.El
.Pp
Predefined strings are not recommended for use,
as they differ across implementations.
Those supported by
.Xr mandoc 1
are listed in
.Xr mandoc_char 5 .
Manuals using these predefined strings are almost certainly not portable.
.Ss Whitespace
Whitespace consists of the space character.
```



```
In text lines, whitespace is preserved within a line.
In request and macro lines, whitespace delimits arguments and is discarded.
.Pp
Unescaped trailing spaces are stripped from text line input unless in a
literal context.
In general, trailing whitespace on any input line is discouraged for
reasons of portability.
In the rare case that a blank character is needed at the end of an
input line, it may be forced by
.Sq \e\ \e& .
.Pp
Literal space characters can be produced in the output
using escape sequences.
In macro lines, they can also be included in arguments using quotation; see
.Sx MACRO SYNTAX
for details.
.Pp
Blank text lines, which may include whitespace, are only permitted
within literal contexts.
If the first character of a text line is a space, that line is printed
with a leading newline.
.Ss Scaling Widths
Many requests and macros support scaled widths for their arguments.
The syntax for a scaled width is
.Sq Li [+-]?[0-9]*.[0-9]*[:unit:] ,
where a decimal must be preceded or followed by at least one digit.
Negative numbers, while accepted, are truncated to zero.
.Pp
The following scaling units are accepted:
.Pp
.Bl -tag -width Ds -offset indent -compact
.It c
centimetre
.It i
inch
.It P
pica (~1/6 inch)
.It p
point (~1/72 inch)
.It f
synonym for
.Sq u
.It v
default vertical span
.It m
width of rendered
.Sq m
.Pq em
character
.It n
width of rendered
.Sq n
.Pq en
character
.It u
default horizontal span
.It M
mini-em (~1/100 em)
.El
.Pp
Using anything other than
.Sq m ,
.Sq n ,
.Sq u ,
or
.Sq v
```

```
260 is necessarily non-portable across output media.
261 See
262 .Sx COMPATIBILITY .
263 .Pp
264 If a scaling unit is not provided, the numerical value is interpreted
265 under the default rules of
266 .Sq v
267 for vertical spaces and
268 .Sq u
269 for horizontal ones.
270 .Pp
271 Examples:
272 .Bl -tag -width ".Bl -tag -width 2i" -offset indent -compact
273 .It Li \&.Bl -tag -width 2i
274 two-inch tagged list indentation in
275 .Xr mdoc 5
276 .It Li \&.HP 2i
277 two-inch tagged list indentation in
278 .Xr man 5
279 .It Li \&.sp 2v
280 two vertical spaces
281 .El
282 .Ss Sentence Spacing
283 Each sentence should terminate at the end of an input line.
284 By doing this, a formatter will be able to apply the proper amount of
285 spacing after the end of sentence (unescaped) period, exclamation mark,
286 or question mark followed by zero or more non-sentence closing
287 delimiters
288 .Po
289 .Sq \&) ,
290 .Sq \&] ,
291 .Sq \&' ,
292 .Sq \&"
293 .Pc .
294 .Pp
295 The proper spacing is also intelligently preserved if a sentence ends at
296 the boundary of a macro line.
297 .Pp
298 Examples:
299 .Bd -literal -offset indent -compact
300 Do not end sentences mid-line like this.  Instead,
301 end a sentence like this.
302 A macro would end like this:
303 \&.Xr mandoc 1 \&.
304 .Ed
305 .Sh REQUEST SYNTAX
306 A request or macro line consists of:
307 .Pp
308 .Bl -enum -compact
309 .It
310 the control character
311 .Sq \&.
312 or
313 .Sq \(aq
314 at the beginning of the line,
315 .It
316 optionally an arbitrary amount of whitespace,
317 .It
318 the name of the request or the macro, which is one word of arbitrary
319 length, terminated by whitespace,
320 .It
321 and zero or more arguments delimited by whitespace.
322 .El
323 .Pp
324 Thus, the following request lines are all equivalent:
325 .Bd -literal -offset indent
```



```
326 \&.ig end
327 \&.ig    end
328 \&.   ig end
329 .Ed
330 .Sh MACRO SYNTAX
331 Macros are provided by the
332 .Xr mdoc 5
333 and
334 .Xr man 5
335 languages and can be defined by the
336 .Sx \&de
337 request.
338 When called, they follow the same syntax as requests, except that
339 macro arguments may optionally be quoted by enclosing them
340 in double quote characters
341 .Pq Sq \(dq .
342 Quoted text, even if it contains whitespace or would cause
343 a macro invocation when unquoted, is always considered literal text.
344 Inside quoted text, pairs of double quote characters
345 .Pq Sq Qq
346 resolve to single double quote characters.
347 .Pp
348 To be recognised as the beginning of a quoted argument, the opening
349 quote character must be preceded by a space character.
350 A quoted argument extends to the next double quote character that is not
351 part of a pair, or to the end of the input line, whichever comes earlier.
352 Leaving out the terminating double quote character at the end of the line
353 is discouraged.
354 For clarity, if more arguments follow on the same input line,
355 it is recommended to follow the terminating double quote character
356 by a space character; in case the next character after the terminating
357 double quote character is anything else, it is regarded as the beginning
358 of the next, unquoted argument.
359 .Pp
360 Both in quoted and unquoted arguments, pairs of backslashes
361 .Pq Sq \e\e
362 resolve to single backslashes.
363 In unquoted arguments, space characters can alternatively be included
364 by preceding them with a backslash
365 .Pq Sq \e\~ ,
366 but quoting is usually better for clarity.
367 .Pp
368 Examples:
369 .Bl -tag -width Ds -offset indent -compact
370 .It Li .Fn strlen \(dqconst char *s\(dq
371 Group arguments
372 .Qq const char *s
373 into one function argument.
374 If unspecified,
375 .Qq const ,
376 .Qq char ,
377 and
378 .Qq *s
379 would be considered separate arguments.
380 .It Li .Op \(dqFl a\(dq
381 Consider
382 .Qq \&Fl a
383 as literal text instead of a flag macro.
384 .El
385 .Sh REQUEST REFERENCE
386 The
387 .Xr mandoc 1
388 .Nm
389 parser recognises the following requests.
390 Note that the
391 .Nm
```

```
language defines many more requests not implemented in
.Xr mandoc 1 .
.Ss \&ad
Set line adjustment mode.
This line-scoped request is intended to have one argument to select
normal, left, right, or centre adjustment for subsequent text.
Currently, it is ignored including its arguments,
and the number of arguments is not checked.
.Ss \&am
Append to a macro definition.
The syntax of this request is the same as that of
.Sx \&de .
It is currently ignored by
.Xr mandoc 1 ,
as are its children.
.Ss \&ami
Append to a macro definition, specifying the macro name indirectly.
The syntax of this request is the same as that of
.Sx \&dei .
It is currently ignored by
.Xr mandoc 1 ,
as are its children.
.Ss \&am1
Append to a macro definition, switching roff compatibility mode off
during macro execution.
The syntax of this request is the same as that of
.Sx \&de1 .
It is currently ignored by
.Xr mandoc 1 ,
as are its children.
.Ss \&de
Define a
.Nm
macro.
Its syntax can be either
.Bd -literal -offset indent
.Pf . Cm \&de Ar name
.Ar macro definition
\&..
.Ed
.Pp
or
.Bd -literal -offset indent
.Pf . Cm \&de Ar name Ar end
.Ar macro definition
.Pf . Ar end
.Ed
.Pp
Both forms define or redefine the macro
.Ar name
to represent the
.Ar macro definition ,
which may consist of one or more input lines, including the newline
characters terminating each line, optionally containing calls to
.Nm
requests,
.Nm
macros or high-level macros like
.Xr man 5
or
.Xr mdoc 5
macros, whichever applies to the document in question.
.Pp
Specifying a custom
.Ar end
macro works in the same way as for
```



```
.Sx \&ig ;
namely, the call to
.Sq Pf . Ar end
first ends the
.Ar macro definition ,
and after that, it is also evaluated as a
.Nm
request or
.Nm
macro, but not as a high-level macro.
.Pp
The macro can be invoked later using the syntax
.Pp
.D1 Pf . Ar name Op Ar argument Op Ar argument ...
.Pp
Regarding argument parsing, see
.Sx MACRO SYNTAX
above.
.Pp
The line invoking the macro will be replaced
in the input stream by the
.Ar macro definition ,
replacing all occurrences of
.No \e\e$ Ns Ar N ,
where
.Ar N
is a digit, by the
.Ar N Ns th Ar argument .
For example,
.Bd -literal -offset indent
\&.de ZN
\efI\e^\e\e$1\e^\efP\e\e$2
\&..
\&.ZN XtFree .
.Ed
.Pp
produces
.Pp
.D1 \efI\e^XtFree\e^\efP.
.Pp
in the input stream, and thus in the output: \fI\^XtFree\^\fP.
.Pp
Since macros and user-defined strings share a common string table,
defining a macro
.Ar name
clobbers the user-defined string
.Ar name ,
and the
.Ar macro definition
can also be printed using the
.Sq \e*
string interpolation syntax described below
.Sx ds ,
but this is rarely useful because every macro definition contains at least
one explicit newline character.
.Pp
In order to prevent endless recursion, both groff and
.Xr mandoc 1
limit the stack depth for expanding macros and strings
to a large, but finite number.
Do not rely on the exact value of this limit.
.Ss \&dei
Define a
.Nm
macro, specifying the macro name indirectly.
The syntax of this request is the same as that of
```

```
.Sx \&de .
It is currently ignored by
.Xr mandoc 1 ,
as are its children.
.Ss \&de1
Define a
.Nm
macro that will be executed with
.Nm
compatibility mode switched off during macro execution.
This is a GNU extension not available in traditional
.Nm
implementations and not even in older versions of groff.
Since
.Xr mandoc 1
does not implement
.Nm
compatibility mode at all, it handles this request as an alias for
.Sx \&de .
.Ss \&ds
Define a user-defined string.
Its syntax is as follows:
.Pp
.D1 Pf . Cm \&ds Ar name Oo \(dq Oc Ns Ar string
.Pp
The
.Ar name
and
.Ar string
arguments are space-separated.
If the
.Ar string
begins with a double-quote character, that character will not be part
of the string.
All remaining characters on the input line form the
.Ar string ,
including whitespace and double-quote characters, even trailing ones.
.Pp
The
.Ar string
can be interpolated into subsequent text by using
.No \e* Ns Bq Ar name
for a
.Ar name
of arbitrary length, or \e*(NN or \e*N if the length of
.Ar name
is two or one characters, respectively.
Interpolation can be prevented by escaping the leading backslash;
that is, an asterisk preceded by an even number of backslashes
does not trigger string interpolation.
.Pp
Since user-defined strings and macros share a common string table,
defining a string
.Ar name
clobbers the macro
.Ar name ,
and the
.Ar name
used for defining a string can also be invoked as a macro,
in which case the following input line will be appended to the
.Ar string ,
forming a new input line passed to the
.Nm
parser.
For example,
.Bd -literal -offset indent
```



```
\&.ds badidea .S
\&.badidea
H SYNOPSIS
.Ed
.Pp
invokes the
.Cm SH
macro when used in a
.Xr man 5
document.
Such abuse is of course strongly discouraged.
.Ss \&el
The
.Qq else
half of an if/else conditional.
Pops a result off the stack of conditional evaluations pushed by
.Sx \&ie
and uses it as its conditional.
If no stack entries are present (e.g., due to no prior
.Sx \&ie
calls)
then false is assumed.
The syntax of this request is similar to
.Sx \&if
except that the conditional is missing.
.Ss \&EN
End an equation block.
See
.Sx \&EQ .
.Ss \&EQ
Begin an equation block.
See
.Xr eqn 5
for a description of the equation language.
.Ss \&hy
Set automatic hyphenation mode.
This line-scoped request is currently ignored.
.Ss \&ie
The
.Qq if
half of an if/else conditional.
The result of the conditional is pushed into a stack used by subsequent
invocations of
.Sx \&el ,
which may be separated by any intervening input (or not exist at all).
Its syntax is equivalent to
.Sx \&if .
.Ss \&if
Begins a conditional.
Right now, the conditional evaluates to true
if and only if it starts with the letter
.Sy n ,
indicating processing in nroff style as opposed to troff style.
If a conditional is false, its children are not processed, but are
syntactically interpreted to preserve the integrity of the input
document.
Thus,
.Pp
.D1 \&.if t .ig
.Pp
will discard the
.Sq \&.ig ,
which may lead to interesting results, but
.Pp
.D1 \&.if t .if t \e{\e
.Pp
```

```
will continue to syntactically interpret to the block close of the final
conditional.
Sub-conditionals, in this case, obviously inherit the truth value of
the parent.
This request has the following syntax:
.Bd -literal -offset indent
\&.if COND \e{\e
BODY...
\&.\e}
.Ed
.Bd -literal -offset indent
\&.if COND \e{ BODY
BODY... \e}
.Ed
.Bd -literal -offset indent
\&.if COND \e{ BODY
BODY...
\&.\e}
.Ed
.Bd -literal -offset indent
\&.if COND \e
BODY
.Ed
.Pp
COND is a conditional statement.
roff allows for complicated conditionals; mandoc is much simpler.
At this time, mandoc supports only
.Sq n ,
evaluating to true;
and
.Sq t ,
.Sq e ,
and
.Sq o ,
evaluating to false.
All other invocations are read up to the next end of line or space and
evaluate as false.
.Pp
If the BODY section is begun by an escaped brace
.Sq \e{ ,
scope continues until a closing-brace escape sequence
.Sq \.\e} .
If the BODY is not enclosed in braces, scope continues until
the end of the line.
If the COND is followed by a BODY on the same line, whether after a
brace or not, then requests and macros
.Em must
begin with a control character.
It is generally more intuitive, in this case, to write
.Bd -literal -offset indent
\&.if COND \e{\e
\&.foo
bar
\&.\e}
.Ed
.Pp
than having the request or macro follow as
.Pp
.D1 \&.if COND \e{ .foo
.Pp
The scope of a conditional is always parsed, but only executed if the
conditional evaluates to true.
.Pp
Note that the
.Sq \e}
is converted into a zero-width escape sequence if not passed as a
```



```
standalone macro
.Sq \&.\e} .
For example,
.Pp
.D1 \&.Fl a \e} b
.Pp
will result in
.Sq \e}
being considered an argument of the
.Sq \&Fl
macro.
.Ss \&ig
Ignore input.
Its syntax can be either
.Bd -literal -offset indent
.Pf . Cm \&ig
.Ar ignored text
\&..
.Ed
.Pp
or
.Bd -literal -offset indent
.Pf . Cm \&ig Ar end
.Ar ignored text
.Pf . Ar end
.Ed
.Pp
In the first case, input is ignored until a
.Sq \&..
request is encountered on its own line.
In the second case, input is ignored until the specified
.Sq Pf . Ar end
macro is encountered.
Do not use the escape character
.Sq \e
anywhere in the definition of
.Ar end ;
it would cause very strange behaviour.
.Pp
When the
.Ar end
macro is a roff request or a roff macro, like in
.Pp
.D1 \&.ig if
.Pp
the subsequent invocation of
.Sx \&if
will first terminate the
.Ar ignored text ,
then be invoked as usual.
Otherwise, it only terminates the
.Ar ignored text ,
and arguments following it or the
.Sq \&..
request are discarded.
.Ss \&ne
Declare the need for the specified minimum vertical space
before the next trap or the bottom of the page.
This line-scoped request is currently ignored.
.Ss \&nh
Turn off automatic hyphenation mode.
This line-scoped request is currently ignored.
.Ss \&rm
Remove a request, macro or string.
This request is intended to have one argument,
the name of the request, macro or string to be undefined.
```

```
Currently, it is ignored including its arguments,
and the number of arguments is not checked.
.Ss \&nr
Define a register.
A register is an arbitrary string value that defines some sort of state,
which influences parsing and/or formatting.
Its syntax is as follows:
.Pp
.D1 Pf \. Cm \&nr Ar name Ar value
.Pp
The
.Ar value
may, at the moment, only be an integer.
So far, only the following register
.Ar name
is recognised:
.Bl -tag -width Ds
.It Cm nS
If set to a positive integer value, certain
.Xr mdoc 5
macros will behave in the same way as in the
.Em SYNOPSIS
section.
If set to 0, these macros will behave in the same way as outside the
.Em SYNOPSIS
section, even when called within the
.Em SYNOPSIS
section itself.
Note that starting a new
.Xr mdoc 5
section with the
.Cm \&Sh
macro will reset this register.
.El
.Ss \&ns
Turn on no-space mode.
This line-scoped request is intended to take no arguments.
Currently, it is ignored including its arguments,
and the number of arguments is not checked.
.Ss \&ps
Change point size.
This line-scoped request is intended to take one numerical argument.
Currently, it is ignored including its arguments,
and the number of arguments is not checked.
.Ss \&so
Include a source file.
Its syntax is as follows:
.Pp
.D1 Pf \. Cm \&so Ar file
.Pp
The
.Ar file
will be read and its contents processed as input in place of the
.Sq \&.so
request line.
To avoid inadvertent inclusion of unrelated files,
.Xr mandoc 1
only accepts relative paths not containing the strings
.Qq ../
and
.Qq /.. .
.Pp
This request requires
.Xr man 1
to change to the right directory before calling
.Xr mandoc 1 ,
```



```
per convention to the root of the manual tree.
Typical usage looks like:
.Pp
.Dl \&.so man3/Xcursor.3
.Pp
As the whole concept is rather fragile, the use of
.Sx \&so
is discouraged.
Use
.Xr ln 1
instead.
.Ss \&ta
Set tab stops.
This line-scoped request can take an arbitrary number of arguments.
Currently, it is ignored including its arguments.
.Ss \&tr
Output character translation.
Its syntax is as follows:
.Pp
.D1 Pf \. Cm \&tr Ar [ab]+
.Pp
Pairs of
.Ar ab
characters are replaced
.Ar ( a
for
.Ar b ) .
Replacement (or origin) characters may also be character escapes; thus,
.Pp
.Dl tr \e(xx\e(yy
.Pp
replaces all invocations of \e(xx with \e(yy.
.Ss \&T&
Re-start a table layout, retaining the options of the prior table
invocation.
See
.Sx \&TS .
.Ss \&TE
End a table context.
See
.Sx \&TS .
.Ss \&TS
Begin a table, which formats input in aligned rows and columns.
See
.Xr tbl 5
for a description of the tbl language.
.Sh COMPATIBILITY
This section documents compatibility between mandoc and other other
.Nm
implementations, at this time limited to GNU troff
.Pq Qq groff .
The term
.Qq historic groff
refers to groff version 1.15.
.Pp
.Bl -dash -compact
.It
In mandoc, the
.Sx \&EQ ,
.Sx \&TE ,
.Sx \&TS ,
and
.Sx \&T& ,
macros are considered regular macros.
In all other
.Nm
```

```
implementations, these are special macros that must be specified without
spacing between the control character (which must be a period) and the
macro name.
.It
The
.Cm nS
register is only compatible with OpenBSD's groff-1.15.
.It
Historic groff did not accept white-space before a custom
.Ar end
macro for the
.Sx \&ig
request.
.It
The
.Sx \&if
and family would print funny white-spaces with historic groff when
using the next-line syntax.
.El
.Sh SEE ALSO
.Xr mandoc 1 ,
.Xr eqn 5 ,
.Xr man 5 ,
.Xr mandoc_char 5 ,
.Xr mdoc 5 ,
.Xr tbl 5
.Rs
.%A Joseph F. Ossanna
.%A Brian W. Kernighan
.%I AT&T Bell Laboratories
.%T Troff User's Manual
.%R Computing Science Technical Report
.%N 54
.%C Murray Hill, New Jersey
.%D 1976 and 1992
.%U http://www.kohala.com/start/troff/cstr54.ps
.Re
.Rs
.%A Joseph F. Ossanna
.%A Brian W. Kernighan
.%A Gunnar Ritter
.%T Heirloom Documentation Tools Nroff/Troff User's Manual
.%D September 17, 2007
.%U http://heirloom.sourceforge.net/doctools/troff.pdf
.Re
.Sh HISTORY
The RUNOFF typesetting system, whose input forms the basis for
.Nm ,
was written in MAD and FAP for the CTSS operating system by Jerome E.
Saltzer in 1964.
Doug McIlroy rewrote it in BCPL in 1969, renaming it
.Nm .
Dennis M. Ritchie rewrote McIlroy's
.Nm
in PDP-11 assembly for
.At v1 ,
Joseph F. Ossanna improved roff and renamed it nroff
for
.At v2 ,
then ported nroff to C as troff, which Brian W. Kernighan released with
.At v7 .
In 1989, James Clarke re-implemented troff in C++, naming it groff.
.Sh AUTHORS
.An -nosplit
This
.Nm
```



```
reference was written by
.An Kristaps Dzonsons ,
.Mt kristaps@bsd.lv ;
and
.An Ingo Schwarze ,
.Mt schwarze@openbsd.org .
```

```
**********************************************************
   76141 Wed Jul 16 14:05:09 2014
new/usr/src/man/man5/mdoc.5
feedback from Hans
mandoc import
**********************************************************
```

```
.\"
.\" Permission to use, copy, modify, and distribute this software for any
.\" purpose with or without fee is hereby granted, provided that the above
.\" copyright notice and this permission notice appear in all copies.
.\"
.\" THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES
.\" WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF
.\" MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR
.\" ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES
.\" WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN
.\" ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF
.\" OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
.\"
.\"
.\" Copyright (c) 2009, 2010, 2011 Kristaps Dzonsons <kristaps@bsd.lv>
.\" Copyright (c) 2010, 2011 Ingo Schwarze <schwarze@openbsd.org>
.\" Copyright 2012 Nexenta Systems, Inc. All rights reserved.
.\" Copyright 2014 Garrett D'Amore <garrett@dmaore.org>
.\"
.Dd Jul 16, 2014
.Dt MDOC 5
.Os
.Sh NAME
.Nm mdoc
.Nd semantic markup language for formatting manual pages
.Sh DESCRIPTION
The
.Nm mdoc
language supports authoring of manual pages for the
.Xr man 1
utility by allowing semantic annotations of words, phrases,
page sections and complete manual pages.
Such annotations are used by formatting tools to achieve a uniform
presentation across all manuals written in
.Nm ,
and to support hyperlinking if supported by the output medium.
.Pp
This reference document describes the structure of manual pages
and the syntax and usage of the
.Nm
language.
The reference implementation of a parsing and formatting tool is
.Xr mandoc 1 ;
the
.Sx COMPATIBILITY
section describes compatibility with other implementations.
.Pp
In an
.Nm
document, lines beginning with the control character
.Sq \&.
are called
.Dq macro lines .
The first word is the macro name.
It consists of two or three letters.
Most macro names begin with a capital letter.
For a list of available macros, see
.Sx MACRO OVERVIEW .
The words following the macro name are arguments to the macro, optionally
including the names of other, callable macros; see
```



```
.Sx MACRO SYNTAX
for details.
.Pp
Lines not beginning with the control character are called
.Dq text lines .
They provide free-form text to be printed; the formatting of the text
depends on the respective processing context:
.Bd -literal -offset indent
\&.Sh Macro lines change control state.
Text lines are interpreted within the current state.
.Ed
.Pp
Many aspects of the basic syntax of the
.Nm
language are based on the
.Xr roff 5
language; see the
.Em LANGUAGE SYNTAX
and
.Em MACRO SYNTAX
sections in the
.Xr roff 5
manual for details, in particular regarding
comments, escape sequences, whitespace, and quoting.
However, using
.Xr roff 5
requests in
.Nm
documents is discouraged;
.Xr mandoc 1
supports some of them merely for backward compatibility.
.Sh MANUAL STRUCTURE
A well-formed
.Nm
document consists of a document prologue followed by one or more
sections.
.Pp
The prologue, which consists of the
.Sx \&Dd ,
.Sx \&Dt ,
and
.Sx \&Os
macros in that order, is required for every document.
.Pp
The first section (sections are denoted by
.Sx \&Sh )
must be the NAME section, consisting of at least one
.Sx \&Nm
followed by
.Sx \&Nd .
.Pp
Following that, convention dictates specifying at least the
.Em SYNOPSIS
and
.Em DESCRIPTION
sections, although this varies between manual sections.
.Pp
The following is a well-formed skeleton
.Nm
file for a utility
.Qq progname :
.Bd -literal -offset indent
\&.Dd Jan 1, 1970
\&.Dt PROGNAME section
\&.Os
\&.Sh NAME
```

```
\&.Nm progname
\&.Nd one line description
\&.\e\(dq .Sh LIBRARY
\&.\e\(dq For sections 2, 3, & 9 only.
\&.Sh SYNOPSIS
\&.Nm progname
\&.Op Fl options
\&.Ar
\&.Sh DESCRIPTION
The
\&.Nm
utility processes files ...
\&.\e\(dq .Sh IMPLEMENTATION NOTES
\&.\e\(dq .Sh RETURN VALUES
\&.\e\(dq For sections 2, 3, & 9 only.
\&.\e\(dq .Sh ENVIRONMENT
\&.\e\(dq For sections 1, 1M, 5, & 6 only.
\&.\e\(dq .Sh FILES
\&.\e\(dq .Sh EXIT STATUS
\&.\e\(dq For sections 1, 1M, & 6 only.
\&.\e\(dq .Sh EXAMPLES
\&.\e\(dq .Sh DIAGNOSTICS
\&.\e\(dq For sections 1, 1M, 5, 6, & 7 only.
\&.\e\(dq .Sh ERRORS
\&.\e\(dq For sections 2, 3, & 9 only.
\&.\e\(dq .Sh ARCHITECTURE
\&.\e\(dq .Sh CODE SET INDEPENDENCE
\&.\e\(dq For sections 1, 1M, & 3 only.
\&.\e\(dq .Sh INTERFACE STABILITY
\&.\e\(dq .Sh MT-LEVEL
\&.\e\(dq For sections 2 & 3 only.
\&.\e\(dq .Sh SEE ALSO
\&.\e\(dq .Xr foobar 1
\&.\e\(dq .Sh STANDARDS
\&.\e\(dq .Sh HISTORY
\&.\e\(dq .Sh AUTHORS
\&.\e\(dq .Sh CAVEATS
\&.\e\(dq .Sh BUGS
\&.\e\(dq .Sh SECURITY CONSIDERATIONS
\&.\e\(dq Not used in OpenBSD.
.Ed
.Pp
The sections in an
.Nm
document are conventionally ordered as they appear above.
Sections should be composed as follows:
.Bl -ohang -offset Ds
.It Em NAME
The name(s) and a one line description of the documented material.
The syntax for this as follows:
.Bd -literal -offset indent
\&.Nm name0 ,
\&.Nm name1 ,
\&.Nm name2
\&.Nd a one line description
.Ed
.Pp
Multiple
.Sq \&Nm
names should be separated by commas.
.Pp
The
.Sx \&Nm
macro(s) must precede the
.Sx \&Nd
macro.
```



```
.Pp
See
.Sx \&Nm
and
.Sx \&Nd .
.It Em LIBRARY
The name of the library containing the documented material, which is
assumed to be a function in a section 2, 3, or 9 manual.
The syntax for this is as follows:
.Bd -literal -offset indent
\&.Lb libarm
.Ed
.Pp
See
.Sx \&Lb .
.It Em SYNOPSIS
Documents the utility invocation syntax, function call syntax, or device
configuration.
.Pp
For the first, utilities (sections 1, 1M, and 6), this is
generally structured as follows:
.Bd -literal -offset indent
\&.Nm bar
\&.Op Fl v
\&.Op Fl o Ar file
\&.Op Ar
\&.Nm foo
\&.Op Fl v
\&.Op Fl o Ar file
\&.Op Ar
.Ed
.Pp
Commands should be ordered alphabetically.
.Pp
For the second, function calls (sections 2, 3, 9):
.Bd -literal -offset indent
\&.In header.h
\&.Vt extern const char *global;
\&.Ft "char *"
\&.Fn foo "const char *src"
\&.Ft "char *"
\&.Fn bar "const char *src"
.Ed
.Pp
Ordering of
.Sx \&In ,
.Sx \&Vt ,
.Sx \&Fn ,
and
.Sx \&Fo
macros should follow C header-file conventions.
.Pp
And for the third, configurations (section 7):
.Bd -literal -offset indent
\&.Cd \(dqit* at isa? port 0x2e\(dq
\&.Cd \(dqit* at isa? port 0x4e\(dq
.Ed
.Pp
Manuals not in these sections generally don't need a
.Em SYNOPSIS .
.Pp
Some macros are displayed differently in the
.Em SYNOPSIS
section, particularly
.Sx \&Nm ,
.Sx \&Cd ,
```

```
.Sx \&Fd ,
.Sx \&Fn ,
.Sx \&Fo ,
.Sx \&In ,
.Sx \&Vt ,
and
.Sx \&Ft .
All of these macros are output on their own line.
If two such dissimilar macros are pairwise invoked (except for
.Sx \&Ft
before
.Sx \&Fo
or
.Sx \&Fn ) ,
they are separated by a vertical space, unless in the case of
.Sx \&Fo ,
.Sx \&Fn ,
and
.Sx \&Ft ,
which are always separated by vertical space.
.Pp
When text and macros following an
.Sx \&Nm
macro starting an input line span multiple output lines,
all output lines but the first will be indented to align
with the text immediately following the
.Sx \&Nm
macro, up to the next
.Sx \&Nm ,
.Sx \&Sh ,
or
.Sx \&Ss
macro or the end of an enclosing block, whichever comes first.
.It Em DESCRIPTION
This begins with an expansion of the brief, one line description in
.Em NAME :
.Bd -literal -offset indent
The
\&.Nm
utility does this, that, and the other.
.Ed
.Pp
It usually follows with a breakdown of the options (if documenting a
command), such as:
.Bd -literal -offset indent
The arguments are as follows:
\&.Bl \-tag \-width Ds
\&.It Fl v
Print verbose information.
\&.El
.Ed
.Pp
Manuals not documenting a command won't include the above fragment.
.Pp
Since the
.Em DESCRIPTION
section usually contains most of the text of a manual, longer manuals
often use the
.Sx \&Ss
macro to form subsections.
In very long manuals, the
.Em DESCRIPTION
may be split into multiple sections, each started by an
.Sx \&Sh
macro followed by a non-standard section name, and each having
several subsections, like in the present
```



```
.Nm
manual.
.It Em IMPLEMENTATION NOTES
Implementation-specific notes should be kept here.
This is useful when implementing standard functions that may have side
effects or notable algorithmic implications.
.It Em RETURN VALUES
This section documents the
return values of functions in sections 2, 3, and 9.
.Pp
See
.Sx \&Rv .
.It Em ENVIRONMENT
Lists the environment variables used by the utility,
and explains the syntax and semantics of their values.
The
.Xr environ 5
manual provides examples of typical content and formatting.
.Pp
See
.Sx \&Ev .
.It Em FILES
Documents files used.
It's helpful to document both the file name and a short description of how
the file is used (created, modified, etc.).
.Pp
See
.Sx \&Pa .
.It Em EXIT STATUS
This section documents the
command exit status for section 1, 6, and 8 utilities.
Historically, this information was described in
.Em DIAGNOSTICS ,
a practise that is now discouraged.
.Pp
See
.Sx \&Ex .
.It Em EXAMPLES
Example usages.
This often contains snippets of well-formed, well-tested invocations.
Make sure that examples work properly!
.It Em DIAGNOSTICS
Documents error conditions.
This is most useful in section 4 manuals.
Historically, this section was used in place of
.Em EXIT STATUS
for manuals in sections 1, 6, and 8; however, this practise is
discouraged.
.Pp
See
.Sx \&Bl
.Fl diag .
.It Em ERRORS
Documents error handling in sections 2, 3, and 9.
.Pp
See
.Sx \&Er .
.It Em ARCHITECTURE
This section is usually absent, but will be present when the
interface is specific to one or more architectures.
.It Em CODE SET INDEPENDENCE
Indicates whether the interface operates correctly with various different
code sets.  True independent code sets will support not only ASCII and
Extended UNIX Codesets (EUC), but also other multi-byte encodings such as
UTF-8 and GB2312.
.Pp
```

```
391 Generally there will be some limitations that are fairly standard.  See
392 .Xr standards 5 for more information about some of these.  Most interfaces
393 should support at least UTF-8 in addition to ASCII.
394 .It Em INTERFACE STABILITY
395 Indicates the level of commitment to the interface. Interfaces can be described
396 with in the following ways:
397 .Bl -tag
398 .It Nm Standard
399 Indicates that the interface is defined by one or more standards bodies.
400 Generally, changes to the interface will be carefully managed to conform
401 to the relevant standards.  These interfaces are generally the most suitable
402 for use in portable programs.
403 .It Nm Committed
404 Indicates that the interface is intended to be preserved for the long-haul, and
405 will rarely, if ever change, and never without notification (barring
406 extraordinary and extenuating circumstances). These interfaces are
407 preferred over other interfaces with the exeception of
408 .Nm Standard
409 interfaces.
410 .It Nm Uncommitted
411 Indicates that the interface may change.  Generally, changes to these interfaces
412 should be infrequent, and some effort will be made to address compatibility
413 considerations when changing or removing such interfaces.  However, there is
414 no firm commitment to the preservation of the interface.  Most often this
415 is applied to interfaces where operational experience with the interface
416 is still limited and some need to change may be anticipated.
417 .Pp
418 Consumers should expect to revalidate any
419 .Nm Uncommitted
420 interfaces when crossing release boundaries.  Products intended for
421 use on many releases or intended to support compatibility with future
422 releases should avoid these interfaces.
423 .It Nm Volatile
424 The interface can change at any time for any reason. Often this relates to
425 interfaces that are part of external software components that are still evolving
426 rapidly.  Consumers should not expect that the interface (either binary or
427 source level) will be unchanged from one release to the next.
428 .It Nm Not-an-Interface
429 Describes something that is specifically not intended for programmatic
430 consumption.  For example, specific human-readable output, or the layout
431 of graphical items on a user interface, may be described this way.  Generally
432 programmatic alternatives to these will be available, and should be used
433 when programmatic consumption is needed.
434 .It Nm Private
435 This is an internal interface.  Generally these interfaces should only be
436 used within the project, and should not be used by other programs or modules.
437 The interface can and will change without notice as the project needs, at
438 any time.
439 .Pp
440 Most often, Private interfaces will lack any documentation whatsoever, and
441 generally any undocumented interface can be assumed to be Private.
442 .It Nm Obsolete
443 The interface is not intended for use in new projects or programs, and may
444 be removed at a future date.  The
445 .Nm Obsolete
446 word is a modifier that can
447 be applied to other commitment levels. For example an
448 .Nm Obsolete Committed
449 interface is unlikely to be removed or changed, but nonetheless new use
450 is discouraged (perhaps a better newer alternative is present).
451 .El
452 .It Em MT-LEVEL
453 This section describes considerations for the interface when used within
454 programs that use multiple threads.  More discussion of these considerations
455 is made in the MT-Level section of
456 .Xr attributes 5 .
```



```
457 The interface can be described in the following ways.
458 .Bl -tag
459 .It Nm Safe
460 Indicates the interface is safe for use within multiple threads.  There
461 may be additional caveats that apply, in which case those will be
462 described.  Note that some interfaces have semantics which may affect
463 other threads, but these should be an intrinsic part of the interface
464 rather than an unexpected side effect.  For example, closing a file in
465 one thread will cause that file to be closed in all threads.
466 .It Nm Unsafe
467 Indicates the interface is unsuitable for concurrent use within multiple
468 threads.  A threaded application may still make use of the interface, but
469 will be required to provide external synchronization means to ensure that
470 only a single thread calls the interface at a time.
471 .It Nm MT-Safe
472 Indicates that the interface is not only safe for concurrent use, but is
473 designed for such use.  For example, a
474 .Nm Safe
475 interface may make use of a global lock to provide safety, but at reduced
476 internal concurrency, whereas an
477 .Nm MT-Safe
478 interface will be designed to be efficient even when used concurrently.
479 .It Nm Async-Signal-Safe
480 Indicates that the library is safe for use within a signal handler.  An
481 .Nm MT-Safe
482 interface can be made
483 .Nm Async-Signal-Safe
484 by ensuring that it blocks signals when acquiring locks.
485 .It Nm Safe with Exections
486 As for
487 .Nm Safe
488 but with specific exceptions noted.
489 .It Nm MT-Safe with Exections
490 As for
491 .Nm MT-Safe
492 but with specific exceptions noted.
493 .El
494 .It Em SEE ALSO
495 References other manuals with related topics.
496 This section should exist for most manuals.
497 Cross-references should conventionally be ordered first by section, then
498 alphabetically.
499 .Pp
500 References to other documentation concerning the topic of the manual page,
501 for example authoritative books or journal articles, may also be
502 provided in this section.
503 .Pp
504 See
505 .Sx \&Rs
506 and
507 .Sx \&Xr .
508 .It Em STANDARDS
509 References any standards implemented or used.
510 If not adhering to any standards, the
511 .Em HISTORY
512 section should be used instead.
513 .Pp
514 See
515 .Sx \&St .
516 .It Em HISTORY
517 A brief history of the subject, including where it was first implemented,
518 and when it was ported to or reimplemented for the operating system at hand.
519 .It Em AUTHORS
520 Credits to the person or persons who wrote the code and/or documentation.
521 Authors should generally be noted by both name and email address.
522 .Pp
```

```
See
.Sx \&An .
.It Em CAVEATS
Common misuses and misunderstandings should be explained
in this section.
.It Em BUGS
Known bugs, limitations, and work-arounds should be described
in this section.
.It Em SECURITY CONSIDERATIONS
Documents any security precautions that operators should consider.
.El
.Sh MACRO OVERVIEW
This overview is sorted such that macros of similar purpose are listed
together, to help find the best macro for any given purpose.
Deprecated macros are not included in the overview, but can be found below
in the alphabetical
.Sx MACRO REFERENCE .
.Ss Document preamble and NAME section macros
.Bl -column "Brq, Bro, Brc" description
.It Sx \&Dd Ta document date: Ar month day , year
.It Sx \&Dt Ta document title: Ar TITLE SECTION Op Ar volume | arch
.It Sx \&Os Ta operating system version: Op Ar system Op Ar version
.It Sx \&Nm Ta document name (one argument)
.It Sx \&Nd Ta document description (one line)
.El
.Ss Sections and cross references
.Bl -column "Brq, Bro, Brc" description
.It Sx \&Sh Ta section header (one line)
.It Sx \&Ss Ta subsection header (one line)
.It Sx \&Sx Ta internal cross reference to a section or subsection
.It Sx \&Xr Ta cross reference to another manual page: Ar name section
.It Sx \&Pp , \&Lp Ta start a text paragraph (no arguments)
.El
.Ss Displays and lists
.Bl -column "Brq, Bro, Brc" description
.It Sx \&Bd , \&Ed Ta display block:
.Fl Ar type
.Op Fl offset Ar width
.Op Fl compact
.It Sx \&D1 Ta indented display (one line)
.It Sx \&Dl Ta indented literal display (one line)
.It Sx \&Bl , \&El Ta list block:
.Fl Ar type
.Op Fl width Ar val
.Op Fl offset Ar val
.Op Fl compact
.It Sx \&It Ta list item (syntax depends on Fl Ar type )
.It Sx \&Ta Ta table cell separator in Sx \&Bl Fl column No lists
.It Sx \&Rs , \&%* , \&Re Ta bibliographic block (references)
.El
.Ss Spacing control
.Bl -column "Brq, Bro, Brc" description
.It Sx \&Pf Ta prefix, no following horizontal space (one argument)
.It Sx \&Ns Ta roman font, no preceding horizontal space (no arguments)
.It Sx \&Ap Ta apostrophe without surrounding whitespace (no arguments)
.It Sx \&Sm Ta switch horizontal spacing mode: Cm on | off
.It Sx \&Bk , \&Ek Ta keep block: Fl words
.It Sx \&br Ta force output line break in text mode (no arguments)
.It Sx \&sp Ta force vertical space: Op Ar height
.El
.Ss Semantic markup for command line utilities:
.Bl -column "Brq, Bro, Brc" description
.It Sx \&Nm Ta start a SYNOPSIS block with the name of a utility
.It Sx \&Fl Ta command line options (flags) (>=0 arguments)
.It Sx \&Cm Ta command modifier (>0 arguments)
.It Sx \&Ar Ta command arguments (>=0 arguments)
```



```
.It Sx \&Op , \&Oo , \&Oc Ta optional syntax elements (enclosure)
.It Sx \&Ic Ta internal or interactive command (>0 arguments)
.It Sx \&Ev Ta environmental variable (>0 arguments)
.It Sx \&Pa Ta file system path (>=0 arguments)
.El
.Ss Semantic markup for function libraries:
.Bl -column "Brq, Bro, Brc" description
.It Sx \&Lb Ta function library (one argument)
.It Sx \&In Ta include file (one argument)
.It Sx \&Ft Ta function type (>0 arguments)
.It Sx \&Fo , \&Fc Ta function block: Ar funcname
.It Sx \&Fn Ta function name:
.Op Ar functype
.Ar funcname
.Oo
.Op Ar argtype
.Ar argname
.Oc
.It Sx \&Fa Ta function argument (>0 arguments)
.It Sx \&Vt Ta variable type (>0 arguments)
.It Sx \&Va Ta variable name (>0 arguments)
.It Sx \&Dv Ta defined variable or preprocessor constant (>0 arguments)
.It Sx \&Er Ta error constant (>0 arguments)
.It Sx \&Ev Ta environmental variable (>0 arguments)
.El
.Ss Various semantic markup:
.Bl -column "Brq, Bro, Brc" description
.It Sx \&An Ta author name (>0 arguments)
.It Sx \&Lk Ta hyperlink: Ar uri Op Ar name
.It Sx \&Mt Ta Do mailto Dc hyperlink: Ar address
.It Sx \&Cd Ta kernel configuration declaration (>0 arguments)
.It Sx \&Ad Ta memory address (>0 arguments)
.It Sx \&Ms Ta mathematical symbol (>0 arguments)
.It Sx \&Tn Ta tradename (>0 arguments)
.El
.Ss Physical markup
.Bl -column "Brq, Bro, Brc" description
.It Sx \&Em Ta italic font or underline (emphasis) (>0 arguments)
.It Sx \&Sy Ta boldface font (symbolic) (>0 arguments)
.It Sx \&Li Ta typewriter font (literal) (>0 arguments)
.It Sx \&No Ta return to roman font (normal) (no arguments)
.It Sx \&Bf , \&Ef Ta font block:
.Op Fl Ar type | Cm \&Em | \&Li | \&Sy
.El
.Ss Physical enclosures
.Bl -column "Brq, Bro, Brc" description
.It Sx \&Dq , \&Do , \&Dc Ta enclose in typographic double quotes: Dq text
.It Sx \&Qq , \&Qo , \&Qc Ta enclose in typewriter double quotes: Qq text
.It Sx \&Sq , \&So , \&Sc Ta enclose in single quotes: Sq text
.It Sx \&Ql Ta single-quoted literal text: Ql text
.It Sx \&Pq , \&Po , \&Pc Ta enclose in parentheses: Pq text
.It Sx \&Bq , \&Bo , \&Bc Ta enclose in square brackets: Bq text
.It Sx \&Brq , \&Bro , \&Brc Ta enclose in curly braces: Brq text
.It Sx \&Aq , \&Ao , \&Ac Ta enclose in angle brackets: Aq text
.It Sx \&Eo , \&Ec Ta generic enclosure
.El
.Ss Text production
.Bl -column "Brq, Bro, Brc" description
.It Sx \&Ex Fl std Ta standard command exit values: Op Ar utility ...
.It Sx \&Rv Fl std Ta standard function return values: Op Ar function ...
.It Sx \&St Ta reference to a standards document (one argument)
.It Sx \&Ux Ta Ux
.It Sx \&At Ta At
.It Sx \&Bx Ta Bx
.It Sx \&Bsx Ta Bsx
.It Sx \&Nx Ta Nx
```

```
.It Sx \&Fx Ta Fx
.It Sx \&Ox Ta Ox
.It Sx \&Dx Ta Dx
.El
.Sh MACRO REFERENCE
This section is a canonical reference of all macros, arranged
alphabetically.
For the scoping of individual macros, see
.Sx MACRO SYNTAX .
.Ss \&%A
Author name of an
.Sx \&Rs
block.
Multiple authors should each be accorded their own
.Sx \%%A
line.
Author names should be ordered with full or abbreviated forename(s)
first, then full surname.
.Ss \&%B
Book title of an
.Sx \&Rs
block.
This macro may also be used in a non-bibliographic context when
referring to book titles.
.Ss \&%C
Publication city or location of an
.Sx \&Rs
block.
.Ss \&%D
Publication date of an
.Sx \&Rs
block.
Recommended formats of arguments are
.Ar month day , year
or just
.Ar year .
.Ss \&%I
Publisher or issuer name of an
.Sx \&Rs
block.
.Ss \&%J
Journal name of an
.Sx \&Rs
block.
.Ss \&%N
Issue number (usually for journals) of an
.Sx \&Rs
block.
.Ss \&%O
Optional information of an
.Sx \&Rs
block.
.Ss \&%P
Book or journal page number of an
.Sx \&Rs
block.
.Ss \&%Q
Institutional author (school, government, etc.) of an
.Sx \&Rs
block.
Multiple institutional authors should each be accorded their own
.Sx \&%Q
line.
.Ss \&%R
Technical report name of an
.Sx \&Rs
```



```
block.
.Ss \&%T
Article title of an
.Sx \&Rs
block.
This macro may also be used in a non-bibliographical context when
referring to article titles.
.Ss \&%U
URI of reference document.
.Ss \&%V
Volume number of an
.Sx \&Rs
block.
.Ss \&Ac
Close an
.Sx \&Ao
block.
Does not have any tail arguments.
.Ss \&Ad
Memory address.
Do not use this for postal addresses.
.Pp
Examples:
.Dl \&.Ad [0,$]
.Dl \&.Ad 0x00000000
.Ss \&An
Author name.
Can be used both for the authors of the program, function, or driver
documented in the manual, or for the authors of the manual itself.
Requires either the name of an author or one of the following arguments:
.Pp
.Bl -tag -width "-nosplitX" -offset indent -compact
.It Fl split
Start a new output line before each subsequent invocation of
.Sx \&An .
.It Fl nosplit
The opposite of
.Fl split .
.El
.Pp
The default is
.Fl nosplit .
The effect of selecting either of the
.Fl split
modes ends at the beginning of the
.Em AUTHORS
section.
In the
.Em AUTHORS
section, the default is
.Fl nosplit
for the first author listing and
.Fl split
for all other author listings.
.Pp
Examples:
.Dl \&.An -nosplit
.Dl \&.An Kristaps Dzonsons \&Aq kristaps@bsd.lv
.Ss \&Ao
Begin a block enclosed by angle brackets.
Does not have any head arguments.
.Pp
Examples:
.Dl \&.Fl -key= \&Ns \&Ao \&Ar val \&Ac
.Pp
See also
```

```
.Sx \&Aq .
.Ss \&Ap
Inserts an apostrophe without any surrounding whitespace.
This is generally used as a grammatical device when referring to the verb
form of a function.
.Pp
Examples:
.Dl \&.Fn execve \&Ap d
.Ss \&Aq
Encloses its arguments in angle brackets.
.Pp
Examples:
.Dl \&.Fl -key= \&Ns \&Aq \&Ar val
.Pp
.Em Remarks :
this macro is often abused for rendering URIs, which should instead use
.Sx \&Lk
or
.Sx \&Mt ,
or to note pre-processor
.Dq Li #include
statements, which should use
.Sx \&In .
.Pp
See also
.Sx \&Ao .
.Ss \&Ar
Command arguments.
If an argument is not provided, the string
.Dq file ...\&
is used as a default.
.Pp
Examples:
.Dl ".Fl o Ar file"
.Dl ".Ar"
.Dl ".Ar arg1 , arg2 ."
.Pp
The arguments to the
.Sx \&Ar
macro are names and placeholders for command arguments;
for fixed strings to be passed verbatim as arguments, use
.Sx \&Fl
or
.Sx \&Cm .
.Ss \&At
Formats an AT&T version.
Accepts one optional argument:
.Pp
.Bl -tag -width "v[1-7] | 32vX" -offset indent -compact
.It Cm v[1-7] | 32v
A version of
.At .
.It Cm III
.At III .
.It Cm V[.[1-4]]?
A version of
.At V .
.El
.Pp
Note that these arguments do not begin with a hyphen.
.Pp
Examples:
.Dl \&.At
.Dl \&.At III
.Dl \&.At V.1
.Pp
```



```
See also
.Sx \&Bsx ,
.Sx \&Bx ,
.Sx \&Dx ,
.Sx \&Fx ,
.Sx \&Nx ,
.Sx \&Ox ,
and
.Sx \&Ux .
.Ss \&Bc
Close a
.Sx \&Bo
block.
Does not have any tail arguments.
.Ss \&Bd
Begin a display block.
Its syntax is as follows:
.Bd -ragged -offset indent
.Pf \. Sx \&Bd
.Fl Ns Ar type
.Op Fl offset Ar width
.Op Fl compact
.Ed
.Pp
Display blocks are used to select a different indentation and
justification than the one used by the surrounding text.
They may contain both macro lines and text lines.
By default, a display block is preceded by a vertical space.
.Pp
The
.Ar type
must be one of the following:
.Bl -tag -width 13n -offset indent
.It Fl centered
Produce one output line from each input line, and centre-justify each line.
Using this display type is not recommended; many
.Nm
implementations render it poorly.
.It Fl filled
Change the positions of line breaks to fill each line, and left- and
right-justify the resulting block.
.It Fl literal
Produce one output line from each input line,
and do not justify the block at all.
Preserve white space as it appears in the input.
Always use a constant-width font.
Use this for displaying source code.
.It Fl ragged
Change the positions of line breaks to fill each line, and left-justify
the resulting block.
.It Fl unfilled
The same as
.Fl literal ,
but using the same font as for normal text, which is a variable width font
if supported by the output device.
.El
.Pp
The
.Ar type
must be provided first.
Additional arguments may follow:
.Bl -tag -width 13n -offset indent
.It Fl offset Ar width
Indent the display by the
.Ar width ,
which may be one of the following:
```

```
.Bl -item
.It
One of the pre-defined strings
.Cm indent ,
the width of a standard indentation (six constant width characters);
.Cm indent-two ,
twice
.Cm indent ;
.Cm left ,
which has no effect;
.Cm right ,
which justifies to the right margin; or
.Cm center ,
which aligns around an imagined centre axis.
.It
A macro invocation, which selects a predefined width
associated with that macro.
The most popular is the imaginary macro
.Ar \&Ds ,
which resolves to
.Sy 6n .
.It
A width using the syntax described in
.Sx Scaling Widths .
.It
An arbitrary string, which indents by the length of this string.
.El
.Pp
When the argument is missing,
.Fl offset
is ignored.
.It Fl compact
Do not assert vertical space before the display.
.El
.Pp
Examples:
.Bd -literal -offset indent
\&.Bd \-literal \-offset indent \-compact
   Hello       world.
\&.Ed
.Ed
.Pp
See also
.Sx \&D1
and
.Sx \&Dl .
.Ss \&Bf
Change the font mode for a scoped block of text.
Its syntax is as follows:
.Bd -ragged -offset indent
.Pf \. Sx \&Bf
.Oo
.Fl emphasis | literal | symbolic |
.Cm \&Em | \&Li | \&Sy
.Oc
.Ed
.Pp
The
.Fl emphasis
and
.Cm \&Em
argument are equivalent, as are
.Fl symbolic
and
.Cm \&Sy ,
and
```



```
.Fl literal
and
.Cm \&Li .
Without an argument, this macro does nothing.
The font mode continues until broken by a new font mode in a nested
scope or
.Sx \&Ef
is encountered.
.Pp
See also
.Sx \&Li ,
.Sx \&Ef ,
.Sx \&Em ,
and
.Sx \&Sy .
.Ss \&Bk
For each macro, keep its output together on the same output line,
until the end of the macro or the end of the input line is reached,
whichever comes first.
Line breaks in text lines are unaffected.
The syntax is as follows:
.Pp
.D1 Pf \. Sx \&Bk Fl words
.Pp
The
.Fl words
argument is required; additional arguments are ignored.
.Pp
The following example will not break within each
.Sx \&Op
macro line:
.Bd -literal -offset indent
\&.Bk \-words
\&.Op Fl f Ar flags
\&.Op Fl o Ar output
\&.Ek
.Ed
.Pp
Be careful in using over-long lines within a keep block!
Doing so will clobber the right margin.
.Ss \&Bl
Begin a list.
Lists consist of items specified using the
.Sx \&It
macro, containing a head or a body or both.
The list syntax is as follows:
.Bd -ragged -offset indent
.Pf \. Sx \&Bl
.Fl Ns Ar type
.Op Fl width Ar val
.Op Fl offset Ar val
.Op Fl compact
.Op HEAD ...
.Ed
.Pp
The list
.Ar type
is mandatory and must be specified first.
The
.Fl width
and
.Fl offset
arguments accept
.Sx Scaling Widths
or use the length of the given string.
The
```

```
.Fl offset
is a global indentation for the whole list, affecting both item heads
and bodies.
For those list types supporting it, the
.Fl width
argument requests an additional indentation of item bodies,
to be added to the
.Fl offset .
Unless the
.Fl compact
argument is specified, list entries are separated by vertical space.
.Pp
A list must specify one of the following list types:
.Bl -tag -width 12n -offset indent
.It Fl bullet
No item heads can be specified, but a bullet will be printed at the head
of each item.
Item bodies start on the same output line as the bullet
and are indented according to the
.Fl width
argument.
.It Fl column
A columnated list.
The
.Fl width
argument has no effect; instead, each argument specifies the width
of one column, using either the
.Sx Scaling Widths
syntax or the string length of the argument.
If the first line of the body of a
.Fl column
list is not an
.Sx \&It
macro line,
.Sx \&It
contexts spanning one input line each are implied until an
.Sx \&It
macro line is encountered, at which point items start being interpreted as
described in the
.Sx \&It
documentation.
.It Fl dash
Like
.Fl bullet ,
except that dashes are used in place of bullets.
.It Fl diag
Like
.Fl inset ,
except that item heads are not parsed for macro invocations.
Most often used in the
.Em DIAGNOSTICS
section with error constants in the item heads.
.It Fl enum
A numbered list.
No item heads can be specified.
Formatted like
.Fl bullet ,
except that cardinal numbers are used in place of bullets,
starting at 1.
.It Fl hang
Like
.Fl tag ,
except that the first lines of item bodies are not indented, but follow
the item heads like in
.Fl inset
lists.
```



```
.It Fl hyphen
Synonym for
.Fl dash .
.It Fl inset
Item bodies follow items heads on the same line, using normal inter-word
spacing.
Bodies are not indented, and the
.Fl width
argument is ignored.
.It Fl item
No item heads can be specified, and none are printed.
Bodies are not indented, and the
.Fl width
argument is ignored.
.It Fl ohang
Item bodies start on the line following item heads and are not indented.
The
.Fl width
argument is ignored.
.It Fl tag
Item bodies are indented according to the
.Fl width
argument.
When an item head fits inside the indentation, the item body follows
this head on the same output line.
Otherwise, the body starts on the output line following the head.
.El
.Pp
Lists may be nested within lists and displays.
Nesting of
.Fl column
and
.Fl enum
lists may not be portable.
.Pp
See also
.Sx \&El
and
.Sx \&It .
.Ss \&Bo
Begin a block enclosed by square brackets.
Does not have any head arguments.
.Pp
Examples:
.Bd -literal -offset indent -compact
\&.Bo 1 ,
\&.Dv BUFSIZ \&Bc
.Ed
.Pp
See also
.Sx \&Bq .
.Ss \&Bq
Encloses its arguments in square brackets.
.Pp
Examples:
.Dl \&.Bq 1 , \&Dv BUFSIZ
.Pp
.Em Remarks :
this macro is sometimes abused to emulate optional arguments for
commands; the correct macros to use for this purpose are
.Sx \&Op ,
.Sx \&Oo ,
and
.Sx \&Oc .
.Pp
See also
```

```
.Sx \&Bo .
.Ss \&Brc
Close a
.Sx \&Bro
block.
Does not have any tail arguments.
.Ss \&Bro
Begin a block enclosed by curly braces.
Does not have any head arguments.
.Pp
Examples:
.Bd -literal -offset indent -compact
\&.Bro 1 , ... ,
\&.Va n \&Brc
.Ed
.Pp
See also
.Sx \&Brq .
.Ss \&Brq
Encloses its arguments in curly braces.
.Pp
Examples:
.Dl \&.Brq 1 , ... , \&Va n
.Pp
See also
.Sx \&Bro .
.Ss \&Bsx
Format the BSD/OS version provided as an argument, or a default value if
no argument is provided.
.Pp
Examples:
.Dl \&.Bsx 1.0
.Dl \&.Bsx
.Pp
See also
.Sx \&At ,
.Sx \&Bx ,
.Sx \&Dx ,
.Sx \&Fx ,
.Sx \&Nx ,
.Sx \&Ox ,
and
.Sx \&Ux .
.Ss \&Bt
Prints
.Dq is currently in beta test.
.Ss \&Bx
Format the BSD version provided as an argument, or a default value if no
argument is provided.
.Pp
Examples:
.Dl \&.Bx 4.3 Tahoe
.Dl \&.Bx 4.4
.Dl \&.Bx
.Pp
See also
.Sx \&At ,
.Sx \&Bsx ,
.Sx \&Dx ,
.Sx \&Fx ,
.Sx \&Nx ,
.Sx \&Ox ,
and
.Sx \&Ux .
.Ss \&Cd
Kernel configuration declaration.
```



```
This denotes strings accepted by
.Xr config 8 .
It is most often used in section 4 manual pages.
.Pp
Examples:
.Dl \&.Cd device le0 at scode?
.Pp
.Em Remarks :
this macro is commonly abused by using quoted literals to retain
whitespace and align consecutive
.Sx \&Cd
declarations.
This practise is discouraged.
.Ss \&Cm
Command modifiers.
Typically used for fixed strings passed as arguments, unless
.Sx \&Fl
is more appropriate.
Also useful when specifying configuration options or keys.
.Pp
Examples:
.Dl ".Nm mt Fl f Ar device Cm rewind"
.Dl ".Nm ps Fl o Cm pid , Ns Cm command"
.Dl ".Nm dd Cm if= Ns Ar file1 Cm of= Ns Ar file2"
.Dl ".Cm IdentityFile Pa ~/.ssh/id_rsa"
.Dl ".Cm LogLevel Dv DEBUG"
.Ss \&D1
One-line indented display.
This is formatted by the default rules and is useful for simple indented
statements.
It is followed by a newline.
.Pp
Examples:
.Dl \&.D1 \&Fl abcdefgh
.Pp
See also
.Sx \&Bd
and
.Sx \&Dl .
.Ss \&Db
Switch debugging mode.
Its syntax is as follows:
.Pp
.D1 Pf \. Sx \&Db Cm on | off
.Pp
This macro is ignored by
.Xr mandoc 1 .
.Ss \&Dc
Close a
.Sx \&Do
block.
Does not have any tail arguments.
.Ss \&Dd
Document date.
This is the mandatory first macro of any
.Nm
manual.
Its syntax is as follows:
.Pp
.D1 Pf \. Sx \&Dd Ar month day , year
.Pp
The
.Ar month
is the full English month name, the
.Ar day
is an optionally zero-padded numeral, and the
```

```
.Ar year
is the full four-digit year.
.Pp
Other arguments are not portable; the
.Xr mandoc 1
utility handles them as follows:
.Bl -dash -offset 3n -compact
.It
To have the date automatically filled in by the
.Ox
version of
.Xr cvs 1 ,
the special string
.Dq $\&Mdocdate$
can be given as an argument.
.It
A few alternative date formats are accepted as well
and converted to the standard form.
.It
If a date string cannot be parsed, it is used verbatim.
.It
If no date string is given, the current date is used.
.El
.Pp
Examples:
.Dl \&.Dd $\&Mdocdate$
.Dl \&.Dd $\&Mdocdate: July 21 2007$
.Dl \&.Dd July 21, 2007
.Pp
See also
.Sx \&Dt
and
.Sx \&Os .
.Ss \&Dl
One-line intended display.
This is formatted as literal text and is useful for commands and
invocations.
It is followed by a newline.
.Pp
Examples:
.Dl \&.Dl % mandoc mdoc.5 \e(ba less
.Pp
See also
.Sx \&Bd
and
.Sx \&D1 .
.Ss \&Do
Begin a block enclosed by double quotes.
Does not have any head arguments.
.Pp
Examples:
.Bd -literal -offset indent -compact
\&.Do
April is the cruellest month
\&.Dc
\e(em T.S. Eliot
.Ed
.Pp
See also
.Sx \&Dq .
.Ss \&Dq
Encloses its arguments in
.Dq typographic
double-quotes.
.Pp
Examples:
```



```
.Bd -literal -offset indent -compact
\&.Dq April is the cruellest month
\e(em T.S. Eliot
.Ed
.Pp
See also
.Sx \&Qq ,
.Sx \&Sq ,
and
.Sx \&Do .
.Ss \&Dt
Document title.
This is the mandatory second macro of any
.Nm
file.
Its syntax is as follows:
.Bd -ragged -offset indent
.Pf \. Sx \&Dt
.Oo
.Ar title
.Oo
.Ar section
.Op Ar volume
.Op Ar arch
.Oc
.Oc
.Ed
.Pp
Its arguments are as follows:
.Bl -tag -width Ds -offset Ds
.It Ar title
The document’s title (name), defaulting to
.Dq UNKNOWN
if unspecified.
It should be capitalised.
.It Ar section
The manual section. It should correspond to the manual’s filename suffix
and defaults to
.Dq 1
if unspecified.
.It Ar volume
This overrides the volume inferred from
.Ar section .
This field is optional.
.It Ar arch
This specifies the machine architecture a manual page applies to,
where relevant.
.El
.Ss \&Dv
Defined variables such as preprocessor constants, constant symbols,
enumeration values, and so on.
.Pp
Examples:
.Dl \&.Dv NULL
.Dl \&.Dv BUFSIZ
.Dl \&.Dv STDOUT_FILENO
.Pp
See also
.Sx \&Er
and
.Sx \&Ev
for special-purpose constants and
.Sx \&Va
for variable symbols.
.Ss \&Dx
Format the DragonFly BSD version provided as an argument, or a default
```

```
value if no argument is provided.
.Pp
Examples:
.Dl \&.Dx 2.4.1
.Dl \&.Dx
.Pp
See also
.Sx \&At ,
.Sx \&Bsx ,
.Sx \&Bx ,
.Sx \&Fx ,
.Sx \&Nx ,
.Sx \&Ox ,
and
.Sx \&Ux .
.Ss \&Ec
Close a scope started by
.Sx \&Eo .
Its syntax is as follows:
.Pp
.D1 Pf \. Sx \&Ec Op Ar TERM
.Pp
The
.Ar TERM
argument is used as the enclosure tail, for example, specifying \e(rq
will emulate
.Sx \&Dc .
.Ss \&Ed
End a display context started by
.Sx \&Bd .
.Ss \&Ef
End a font mode context started by
.Sx \&Bf .
.Ss \&Ek
End a keep context started by
.Sx \&Bk .
.Ss \&El
End a list context started by
.Sx \&Bl .
.Pp
See also
.Sx \&Bl
and
.Sx \&It .
.Ss \&Em
Denotes text that should be
.Em emphasised .
Note that this is a presentation term and should not be used for
stylistically decorating technical terms.
Depending on the output device, this is usually represented
using an italic font or underlined characters.
.Pp
Examples:
.Dl \&.Em Warnings!
.Dl \&.Em Remarks :
.Pp
See also
.Sx \&Bf ,
.Sx \&Li ,
.Sx \&No ,
and
.Sx \&Sy .
.Ss \&En
This macro is obsolete and not implemented in
.Xr mandoc 1 .
.Ss \&Eo
```



```
An arbitrary enclosure.
Its syntax is as follows:
.Pp
.D1 Pf \. Sx \&Eo Op Ar TERM
.Pp
The
.Ar TERM
argument is used as the enclosure head, for example, specifying \e(lq
will emulate
.Sx \&Do .
.Ss \&Er
Error constants for definitions of the
.Va errno
libc global variable.
This is most often used in section 2 and 3 manual pages.
.Pp
Examples:
.Dl \&.Er EPERM
.Dl \&.Er ENOENT
.Pp
See also
.Sx \&Dv
for general constants.
.Ss \&Es
This macro is obsolete and not implemented.
.Ss \&Ev
Environmental variables such as those specified in
.Xr environ 5 .
.Pp
Examples:
.Dl \&.Ev DISPLAY
.Dl \&.Ev PATH
.Pp
See also
.Sx \&Dv
for general constants.
.Ss \&Ex
Insert a standard sentence regarding command exit values of 0 on success
and >0 on failure.
This is most often used in section 1, 6, and 8 manual pages.
Its syntax is as follows:
.Pp
.D1 Pf \. Sx \&Ex Fl std Op Ar utility ...
.Pp
If
.Ar utility
is not specified, the document’s name set by
.Sx \&Nm
is used.
Multiple
.Ar utility
arguments are treated as separate utilities.
.Pp
See also
.Sx \&Rv .
.Ss \&Fa
Function argument.
Its syntax is as follows:
.Bd -ragged -offset indent
.Pf \. Sx \&Fa
.Op Cm argtype
.Cm argname
.Ed
.Pp
This may be invoked for names with or without the corresponding type.
It is also used to specify the field name of a structure.
```

```
Most often, the
.Sx \&Fa
macro is used in the
.Em SYNOPSIS
within
.Sx \&Fo
section when documenting multi-line function prototypes.
If invoked with multiple arguments, the arguments are separated by a
comma.
Furthermore, if the following macro is another
.Sx \&Fa ,
the last argument will also have a trailing comma.
.Pp
Examples:
.Dl \&.Fa \(dqconst char *p\(dq
.Dl \&.Fa \(dqint a\(dq \(dqint b\(dq \(dqint c\(dq
.Dl \&.Fa foo
.Pp
See also
.Sx \&Fo .
.Ss \&Fc
End a function context started by
.Sx \&Fo .
.Ss \&Fd
Historically used to document include files.
This usage has been deprecated in favour of
.Sx \&In .
Do not use this macro.
.Pp
See also
.Sx MANUAL STRUCTURE
and
.Sx \&In .
.Ss \&Fl
Command-line flag or option.
Used when listing arguments to command-line utilities.
Prints a fixed-width hyphen
.Sq \-
directly followed by each argument.
If no arguments are provided, a hyphen is printed followed by a space.
If the argument is a macro, a hyphen is prefixed to the subsequent macro
output.
.Pp
Examples:
.Dl ".Fl R Op Fl H | L | P"
.Dl ".Op Fl 1AaCcdFfgHhikLlmnopqRrSsTtux"
.Dl ".Fl type Cm d Fl name Pa CVS"
.Dl ".Fl Ar signal_number"
.Dl ".Fl o Fl"
.Pp
See also
.Sx \&Cm .
.Ss \&Fn
A function name.
Its syntax is as follows:
.Bd -ragged -offset indent
.Pf \. Ns Sx \&Fn
.Op Ar functype
.Ar funcname
.Op Oo Ar argtype Oc Ar argname
.Ed
.Pp
Function arguments are surrounded in parenthesis and
are delimited by commas.
If no arguments are specified, blank parenthesis are output.
In the
```



```
.Em SYNOPSIS
section, this macro starts a new output line,
and a blank line is automatically inserted between function definitions.
.Pp
Examples:
.Dl \&.Fn \(dqint funcname\(dq \(dqint arg0\(dq \(dqint arg1\(dq
.Dl \&.Fn funcname \(dqint arg0\(dq
.Dl \&.Fn funcname arg0
.Pp
.Bd -literal -offset indent -compact
\&.Ft functype
\&.Fn funcname
.Ed
.Pp
When referring to a function documented in another manual page, use
.Sx \&Xr
instead.
See also
.Sx MANUAL STRUCTURE ,
.Sx \&Fo ,
and
.Sx \&Ft .
.Ss \&Fo
Begin a function block.
This is a multi-line version of
.Sx \&Fn .
Its syntax is as follows:
.Pp
.D1 Pf \. Sx \&Fo Ar funcname
.Pp
Invocations usually occur in the following context:
.Bd -ragged -offset indent
.Pf \. Sx \&Ft Ar functype
.br
.Pf \. Sx \&Fo Ar funcname
.br
.Pf \. Sx \&Fa Oo Ar argtype Oc Ar argname
.br
\&.\.\.
.br
.Pf \. Sx \&Fc
.Ed
.Pp
A
.Sx \&Fo
scope is closed by
.Sx \&Fc .
.Pp
See also
.Sx MANUAL STRUCTURE ,
.Sx \&Fa ,
.Sx \&Fc ,
and
.Sx \&Ft .
.Ss \&Fr
This macro is obsolete and not implemented in
.Xr mandoc 1 .
.Pp
It was used to show function return values.
The syntax was:
.Pp
.Dl Pf . Sx \&Fr Ar value
.Ss \&Ft
A function type.
Its syntax is as follows:
.Pp
```

```
.D1 Pf \. Sx \&Ft Ar functype
.Pp
In the
.Em SYNOPSIS
section, a new output line is started after this macro.
.Pp
Examples:
.Dl \&.Ft int
.Bd -literal -offset indent -compact
\&.Ft functype
\&.Fn funcname
.Ed
.Pp
See also
.Sx MANUAL STRUCTURE ,
.Sx \&Fn ,
and
.Sx \&Fo .
.Ss \&Fx
Format the
.Fx
version provided as an argument, or a default value
if no argument is provided.
.Pp
Examples:
.Dl \&.Fx 7.1
.Dl \&.Fx
.Pp
See also
.Sx \&At ,
.Sx \&Bsx ,
.Sx \&Bx ,
.Sx \&Dx ,
.Sx \&Nx ,
.Sx \&Ox ,
and
.Sx \&Ux .
.Ss \&Hf
This macro is not implemented in
.Xr mandoc 1 .
.Pp
It was used to include the contents of a (header) file literally.
The syntax was:
.Pp
.Dl Pf . Sx \&Hf Ar filename
.Ss \&Ic
Designate an internal or interactive command.
This is similar to
.Sx \&Cm
but used for instructions rather than values.
.Pp
Examples:
.Dl \&.Ic :wq
.Dl \&.Ic hash
.Dl \&.Ic alias
.Pp
Note that using
.Sx \&Bd Fl literal
or
.Sx \&D1
is preferred for displaying code; the
.Sx \&Ic
macro is used when referring to specific instructions.
.Ss \&In
An
.Dq include
```



```
file.
When invoked as the first macro on an input line in the
.Em SYNOPSIS
section, the argument is displayed in angle brackets
and preceded by
.Dq #include ,
and a blank line is inserted in front if there is a preceding
function declaration.
This is most often used in section 2, 3, and 9 manual pages.
.Pp
Examples:
.Dl \&.In sys/types.h
.Pp
See also
.Sx MANUAL STRUCTURE .
.Ss \&It
A list item.
The syntax of this macro depends on the list type.
.Pp
Lists
of type
.Fl hang ,
.Fl ohang ,
.Fl inset ,
and
.Fl diag
have the following syntax:
.Pp
.D1 Pf \. Sx \&It Ar args
.Pp
Lists of type
.Fl bullet ,
.Fl dash ,
.Fl enum ,
.Fl hyphen
and
.Fl item
have the following syntax:
.Pp
.D1 Pf \. Sx \&It
.Pp
with subsequent lines interpreted within the scope of the
.Sx \&It
until either a closing
.Sx \&El
or another
.Sx \&It .
.Pp
The
.Fl tag
list has the following syntax:
.Pp
.D1 Pf \. Sx \&It Op Cm args
.Pp
Subsequent lines are interpreted as with
.Fl bullet
and family.
The line arguments correspond to the list’s left-hand side; body
arguments correspond to the list’s contents.
.Pp
The
.Fl column
list is the most complicated.
Its syntax is as follows:
.Pp
.D1 Pf \. Sx \&It Ar cell Op <TAB> Ar cell ...
```

```
.D1 Pf \. Sx \&It Ar cell Op Sx \&Ta Ar cell ...
.Pp
The arguments consist of one or more lines of text and macros
representing a complete table line.
Cells within the line are delimited by tabs or by the special
.Sx \&Ta
block macro.
The tab cell delimiter may only be used within the
.Sx \&It
line itself; on following lines, only the
.Sx \&Ta
macro can be used to delimit cells, and
.Sx \&Ta
is only recognised as a macro when called by other macros,
not as the first macro on a line.
.Pp
Note that quoted strings may span tab-delimited cells on an
.Sx \&It
line.
For example,
.Pp
.Dl .It \(dqcol1 ; <TAB> col2 ;\(dq \&;
.Pp
will preserve the semicolon whitespace except for the last.
.Pp
See also
.Sx \&Bl .
.Ss \&Lb
Specify a library.
The syntax is as follows:
.Pp
.D1 Pf \. Sx \&Lb Ar library
.Pp
The
.Ar library
parameter may be a system library, such as
.Cm libz
or
.Cm libpam ,
in which case a small library description is printed next to the linker
invocation; or a custom library, in which case the library name is
printed in quotes.
This is most commonly used in the
.Em SYNOPSIS
section as described in
.Sx MANUAL STRUCTURE .
.Pp
Examples:
.Dl \&.Lb libz
.Dl \&.Lb mdoc
.Ss \&Li
Denotes text that should be in a
.Li literal
font mode.
Note that this is a presentation term and should not be used for
stylistically decorating technical terms.
.Pp
On terminal output devices, this is often indistinguishable from
normal text.
.Pp
See also
.Sx \&Bf ,
.Sx \&Em ,
.Sx \&No ,
and
.Sx \&Sy .
```



```
.Ss \&Lk
Format a hyperlink.
Its syntax is as follows:
.Pp
.D1 Pf \. Sx \&Lk Ar uri Op Ar name
.Pp
Examples:
.Dl \&.Lk http://bsd.lv \(dqThe BSD.lv Project\(dq
.Dl \&.Lk http://bsd.lv
.Pp
See also
.Sx \&Mt .
.Ss \&Lp
Synonym for
.Sx \&Pp .
.Ss \&Ms
Display a mathematical symbol.
Its syntax is as follows:
.Pp
.D1 Pf \. Sx \&Ms Ar symbol
.Pp
Examples:
.Dl \&.Ms sigma
.Dl \&.Ms aleph
.Ss \&Mt
Format a
.Dq mailto:
hyperlink.
Its syntax is as follows:
.Pp
.D1 Pf \. Sx \&Mt Ar address
.Pp
Examples:
.Dl \&.Mt discuss@manpages.bsd.lv
.Ss \&Nd
A one line description of the manual’s content.
This may only be invoked in the
.Em SYNOPSIS
section subsequent the
.Sx \&Nm
macro.
.Pp
Examples:
.Dl .Dl Pf . Sx \&Nd mdoc language reference
.Dl .Dl Pf . Sx \&Nd format and display UNIX manuals
.Pp
The
.Sx \&Nd
macro technically accepts child macros and terminates with a subsequent
.Sx \&Sh
invocation.
Do not assume this behaviour: some
.Xr whatis 1
database generators are not smart enough to parse more than the line
arguments and will display macros verbatim.
.Pp
See also
.Sx \&Nm .
.Ss \&Nm
The name of the manual page, or \(em in particular in section 1, 6,
and 8 pages \(em of an additional command or feature documented in
the manual page.
When first invoked, the
.Sx \&Nm
macro expects a single argument, the name of the manual page.
Usually, the first invocation happens in the
```

```
.Em NAME
section of the page.
The specified name will be remembered and used whenever the macro is
called again without arguments later in the page.
The
.Sx \&Nm
macro uses
.Sx Block full-implicit
semantics when invoked as the first macro on an input line in the
.Em SYNOPSIS
section; otherwise, it uses ordinary
.Sx In-line
semantics.
.Pp
Examples:
.Bd -literal -offset indent
\&.Sh SYNOPSIS
\&.Nm cat
\&.Op Fl benstuv
\&.Op Ar
.Ed
.Pp
In the
.Em SYNOPSIS
of section 2, 3 and 9 manual pages, use the
.Sx \&Fn
macro rather than
.Sx \&Nm
to mark up the name of the manual page.
.Ss \&No
Normal text.
Closes the scope of any preceding in-line macro.
When used after physical formatting macros like
.Sx \&Em
or
.Sx \&Sy ,
switches back to the standard font face and weight.
Can also be used to embed plain text strings in macro lines
using semantic annotation macros.
.Pp
Examples:
.Dl ".Em italic , Sy bold , No and roman"
.Pp
.Bd -literal -offset indent -compact
\&.Sm off
\&.Cm :C No / Ar pattern No / Ar replacement No /
\&.Sm on
.Ed
.Pp
See also
.Sx \&Em ,
.Sx \&Li ,
and
.Sx \&Sy .
.Ss \&Ns
Suppress a space between the output of the preceding macro
and the following text or macro.
Following invocation, input is interpreted as normal text
just like after an
.Sx \&No
macro.
.Pp
This has no effect when invoked at the start of a macro line.
.Pp
Examples:
.Dl ".Ar name Ns = Ns Ar value"
```



```
.Dl ".Cm :M Ns Ar pattern"
.Dl ".Fl o Ns Ar output"
.Pp
See also
.Sx \&No
and
.Sx \&Sm .
.Ss \&Nx
Format the
.Nx
version provided as an argument, or a default value if
no argument is provided.
.Pp
Examples:
.Dl \&.Nx 5.01
.Dl \&.Nx
.Pp
See also
.Sx \&At ,
.Sx \&Bsx ,
.Sx \&Bx ,
.Sx \&Dx ,
.Sx \&Fx ,
.Sx \&Ox ,
and
.Sx \&Ux .
.Ss \&Oc
Close multi-line
.Sx \&Oo
context.
.Ss \&Oo
Multi-line version of
.Sx \&Op .
.Pp
Examples:
.Bd -literal -offset indent -compact
\&.Oo
\&.Op Fl flag Ns Ar value
\&.Oc
.Ed
.Ss \&Op
Optional part of a command line.
Prints the argument(s) in brackets.
This is most often used in the
.Em SYNOPSIS
section of section 1 and 8 manual pages.
.Pp
Examples:
.Dl \&.Op \&Fl a \&Ar b
.Dl \&.Op \&Ar a | b
.Pp
See also
.Sx \&Oo .
.Ss \&Os
Document operating system version.
This is the mandatory third macro of
any
.Nm
file.
Its syntax is as follows:
.Pp
.D1 Pf \. Sx \&Os Op Ar system Op Ar version
.Pp
The optional
.Ar system
parameter specifies the relevant operating system or environment.
```

```
Left unspecified, it defaults to the local operating system version.
This is the suggested form.
.Pp
Examples:
.Dl \&.Os
.Dl \&.Os KTH/CSC/TCS
.Dl \&.Os BSD 4.3
.Pp
See also
.Sx \&Dd
and
.Sx \&Dt .
.Ss \&Ot
This macro is obsolete and not implemented in
.Xr mandoc 1 .
.Pp
Historical
.Xr mdoc 5
packages described it as
.Dq "old function type (FORTRAN)" .
.Ss \&Ox
Format the
.Ox
version provided as an argument, or a default value
if no argument is provided.
.Pp
Examples:
.Dl \&.Ox 4.5
.Dl \&.Ox
.Pp
See also
.Sx \&At ,
.Sx \&Bsx ,
.Sx \&Bx ,
.Sx \&Dx ,
.Sx \&Fx ,
.Sx \&Nx ,
and
.Sx \&Ux .
.Ss \&Pa
An absolute or relative file system path, or a file or directory name.
If an argument is not provided, the character
.Sq \(ti
is used as a default.
.Pp
Examples:
.Dl \&.Pa /usr/bin/mandoc
.Dl \&.Pa /usr/share/man/man5/mdoc.5
.Pp
See also
.Sx \&Lk .
.Ss \&Pc
Close parenthesised context opened by
.Sx \&Po .
.Ss \&Pf
Removes the space between its argument
.Pq Dq prefix
and the following macro.
Its syntax is as follows:
.Pp
.D1 .Pf Ar prefix macro arguments ...
.Pp
This is equivalent to:
.Pp
.D1 .No Ar prefix No \&Ns Ar macro arguments ...
.Pp
```



```
Examples:
.Dl ".Pf $ Ar variable_name"
.Dl ".Pf 0x Ar hex_digits"
.Pp
See also
.Sx \&Ns
and
.Sx \&Sm .
.Ss \&Po
Multi-line version of
.Sx \&Pq .
.Ss \&Pp
Break a paragraph.
This will assert vertical space between prior and subsequent macros
and/or text.
.Pp
Paragraph breaks are not needed before or after
.Sx \&Sh
or
.Sx \&Ss
macros or before displays
.Pq Sx \&Bd
or lists
.Pq Sx \&Bl
unless the
.Fl compact
flag is given.
.Ss \&Pq
Parenthesised enclosure.
.Pp
See also
.Sx \&Po .
.Ss \&Qc
Close quoted context opened by
.Sx \&Qo .
.Ss \&Ql
Format a single-quoted literal.
See also
.Sx \&Qq
and
.Sx \&Sq .
.Ss \&Qo
Multi-line version of
.Sx \&Qq .
.Ss \&Qq
Encloses its arguments in
.Qq typewriter
double-quotes.
Consider using
.Sx \&Dq .
.Pp
See also
.Sx \&Dq ,
.Sx \&Sq ,
and
.Sx \&Qo .
.Ss \&Re
Close an
.Sx \&Rs
block.
Does not have any tail arguments.
.Ss \&Rs
Begin a bibliographic
.Pq Dq reference
block.
Does not have any head arguments.
```

```
The block macro may only contain
.Sx \&%A ,
.Sx \&%B ,
.Sx \&%C ,
.Sx \&%D ,
.Sx \&%I ,
.Sx \&%J ,
.Sx \&%N ,
.Sx \&%O ,
.Sx \&%P ,
.Sx \&%Q ,
.Sx \&%R ,
.Sx \&%T ,
.Sx \&%U ,
and
.Sx \&%V
child macros (at least one must be specified).
.Pp
Examples:
.Bd -literal -offset indent -compact
\&.Rs
\&.%A J. E. Hopcroft
\&.%A J. D. Ullman
\&.%B Introduction to Automata Theory, Languages, and Computation
\&.%I Addison-Wesley
\&.%C Reading, Massachusettes
\&.%D 1979
\&.Re
.Ed
.Pp
If an
.Sx \&Rs
block is used within a SEE ALSO section, a vertical space is asserted
before the rendered output, else the block continues on the current
line.
.Ss \&Rv
Insert a standard sentence regarding a function call’s return value of 0
on success and \-1 on error, with the
.Va errno
libc global variable set on error.
Its syntax is as follows:
.Pp
.D1 Pf \. Sx \&Rv Fl std Op Ar function ...
.Pp
If
.Ar function
is not specified, the document’s name set by
.Sx \&Nm
is used.
Multiple
.Ar function
arguments are treated as separate functions.
.Pp
See also
.Sx \&Ex .
.Ss \&Sc
Close single-quoted context opened by
.Sx \&So .
.Ss \&Sh
Begin a new section.
For a list of conventional manual sections, see
.Sx MANUAL STRUCTURE .
These sections should be used unless it’s absolutely necessary that
custom sections be used.
.Pp
Section names should be unique so that they may be keyed by
```



```
.Sx \&Sx .
Although this macro is parsed, it should not consist of child node or it
may not be linked with
.Sx \&Sx .
.Pp
See also
.Sx \&Pp ,
.Sx \&Ss ,
and
.Sx \&Sx .
.Ss \&Sm
Switches the spacing mode for output generated from macros.
Its syntax is as follows:
.Pp
.D1 Pf \. Sx \&Sm Cm on | off
.Pp
By default, spacing is
.Cm on .
When switched
.Cm off ,
no white space is inserted between macro arguments and between the
output generated from adjacent macros, but text lines
still get normal spacing between words and sentences.
.Ss \&So
Multi-line version of
.Sx \&Sq .
.Ss \&Sq
Encloses its arguments in
.Sq typewriter
single-quotes.
.Pp
See also
.Sx \&Dq ,
.Sx \&Qq ,
and
.Sx \&So .
.Ss \&Ss
Begin a new subsection.
Unlike with
.Sx \&Sh ,
there is no convention for the naming of subsections.
Except
.Em DESCRIPTION ,
the conventional sections described in
.Sx MANUAL STRUCTURE
rarely have subsections.
.Pp
Sub-section names should be unique so that they may be keyed by
.Sx \&Sx .
Although this macro is parsed, it should not consist of child node or it
may not be linked with
.Sx \&Sx .
.Pp
See also
.Sx \&Pp ,
.Sx \&Sh ,
and
.Sx \&Sx .
.Ss \&St
Replace an abbreviation for a standard with the full form.
The following standards are recognised:
.Pp
.Bl -tag -width "-p1003.1g-2000X" -compact
.It \-p1003.1-88
.St -p1003.1-88
.It \-p1003.1-90
```

```
.St -p1003.1-90
.It \-p1003.1-96
.St -p1003.1-96
.It \-p1003.1-2001
.St -p1003.1-2001
.It \-p1003.1-2004
.St -p1003.1-2004
.It \-p1003.1-2008
.St -p1003.1-2008
.It \-p1003.1
.St -p1003.1
.It \-p1003.1b
.St -p1003.1b
.It \-p1003.1b-93
.St -p1003.1b-93
.It \-p1003.1c-95
.St -p1003.1c-95
.It \-p1003.1g-2000
.St -p1003.1g-2000
.It \-p1003.1i-95
.St -p1003.1i-95
.It \-p1003.2-92
.St -p1003.2-92
.It \-p1003.2a-92
.St -p1003.2a-92
.It \-p1387.2-95
.St -p1387.2-95
.It \-p1003.2
.St -p1003.2
.It \-p1387.2
.St -p1387.2
.It \-isoC
.St -isoC
.It \-isoC-90
.St -isoC-90
.It \-isoC-amd1
.St -isoC-amd1
.It \-isoC-tcor1
.St -isoC-tcor1
.It \-isoC-tcor2
.St -isoC-tcor2
.It \-isoC-99
.St -isoC-99
.It \-isoC-2011
.St -isoC-2011
.It \-iso9945-1-90
.St -iso9945-1-90
.It \-iso9945-1-96
.St -iso9945-1-96
.It \-iso9945-2-93
.St -iso9945-2-93
.It \-ansiC
.St -ansiC
.It \-ansiC-89
.St -ansiC-89
.It \-ansiC-99
.St -ansiC-99
.It \-ieee754
.St -ieee754
.It \-iso8802-3
.St -iso8802-3
.It \-iso8601
.St -iso8601
.It \-ieee1275-94
.St -ieee1275-94
.It \-xpg3
```



```
.St -xpg3
.It \-xpg4
.St -xpg4
.It \-xpg4.2
.St -xpg4.2
.It \-xpg4.3
.St -xpg4.3
.It \-xbd5
.St -xbd5
.It \-xcu5
.St -xcu5
.It \-xsh5
.St -xsh5
.It \-xns5
.St -xns5
.It \-xns5.2
.St -xns5.2
.It \-xns5.2d2.0
.St -xns5.2d2.0
.It \-xcurses4.2
.St -xcurses4.2
.It \-susv2
.St -susv2
.It \-susv3
.St -susv3
.It \-svid4
.St -svid4
.El
.Ss \&Sx
Reference a section or subsection in the same manual page.
The referenced section or subsection name must be identical to the
enclosed argument, including whitespace.
.Pp
Examples:
.Dl \&.Sx MANUAL STRUCTURE
.Pp
See also
.Sx \&Sh
and
.Sx \&Ss .
.Ss \&Sy
Format enclosed arguments in symbolic
.Pq Dq boldface .
Note that this is a presentation term and should not be used for
stylistically decorating technical terms.
.Pp
See also
.Sx \&Bf ,
.Sx \&Em ,
.Sx \&Li ,
and
.Sx \&No .
.Ss \&Ta
Table cell separator in
.Sx \&Bl Fl column
lists; can only be used below
.Sx \&It .
.Ss \&Tn
Format a tradename.
.Pp
Since this macro is often implemented to use a small caps font,
it has historically been used for acronyms (like ASCII) as well.
Such usage is not recommended because it would use the same macro
sometimes for semantical annotation, sometimes for physical formatting.
.Pp
Examples:
```

```
.Dl \&.Tn IBM
.Ss \&Ud
Prints out
.Dq currently under development.
.Ss \&Ux
Format the UNIX name.
Accepts no argument.
.Pp
Examples:
.Dl \&.Ux
.Pp
See also
.Sx \&At ,
.Sx \&Bsx ,
.Sx \&Bx ,
.Sx \&Dx ,
.Sx \&Fx ,
.Sx \&Nx ,
and
.Sx \&Ox .
.Ss \&Va
A variable name.
.Pp
Examples:
.Dl \&.Va foo
.Dl \&.Va const char *bar ;
.Ss \&Vt
A variable type.
This is also used for indicating global variables in the
.Em SYNOPSIS
section, in which case a variable name is also specified.
Note that it accepts
.Sx Block partial-implicit
syntax when invoked as the first macro on an input line in the
.Em SYNOPSIS
section, else it accepts ordinary
.Sx In-line
syntax.
In the former case, this macro starts a new output line,
and a blank line is inserted in front if there is a preceding
function definition or include directive.
.Pp
Note that this should not be confused with
.Sx \&Ft ,
which is used for function return types.
.Pp
Examples:
.Dl \&.Vt unsigned char
.Dl \&.Vt extern const char * const sys_signame[] \&;
.Pp
See also
.Sx MANUAL STRUCTURE
and
.Sx \&Va .
.Ss \&Xc
Close a scope opened by
.Sx \&Xo .
.Ss \&Xo
Extend the header of an
.Sx \&It
macro or the body of a partial-implicit block macro
beyond the end of the input line.
This macro originally existed to work around the 9-argument limit
of historic
.Xr roff 5 .
.Ss \&Xr
```



```
Link to another manual
.Pq Qq cross-reference .
Its syntax is as follows:
.Pp
.D1 Pf \. Sx \&Xr Ar name section
.Pp
The
.Ar name
and
.Ar section
are the name and section of the linked manual.
If
.Ar section
is followed by non-punctuation, an
.Sx \&Ns
is inserted into the token stream.
This behaviour is for compatibility with
GNU troff.
.Pp
Examples:
.Dl \&.Xr mandoc 1
.Dl \&.Xr mandoc 1 \&;
.Dl \&.Xr mandoc 1 \&Ns s behaviour
.Ss \&br
Emits a line-break.
This macro should not be used; it is implemented for compatibility with
historical manuals.
.Pp
Consider using
.Sx \&Pp
in the event of natural paragraph breaks.
.Ss \&sp
Emits vertical space.
This macro should not be used; it is implemented for compatibility with
historical manuals.
Its syntax is as follows:
.Pp
.D1 Pf \. Sx \&sp Op Ar height
.Pp
The
.Ar height
argument must be formatted as described in
.Sx Scaling Widths .
If unspecified,
.Sx \&sp
asserts a single vertical space.
.Sh MACRO SYNTAX
The syntax of a macro depends on its classification.
In this section,
.Sq \-arg
refers to macro arguments, which may be followed by zero or more
.Sq parm
parameters;
.Sq \&Yo
opens the scope of a macro; and if specified,
.Sq \&Yc
closes it out.
.Pp
The
.Em Callable
column indicates that the macro may also be called by passing its name
as an argument to another macro.
For example,
.Sq \&.Op \&Fl O \&Ar file
produces
.Sq Op Fl O Ar file .
```

```
To prevent a macro call and render the macro name literally,
escape it by prepending a zero-width space,
.Sq \e& .
For example,
.Sq \&Op \e&Fl O
produces
.Sq Op \&Fl O .
If a macro is not callable but its name appears as an argument
to another macro, it is interpreted as opaque text.
For example,
.Sq \&.Fl \&Sh
produces
.Sq Fl \&Sh .
.Pp
The
.Em Parsed
column indicates whether the macro may call other macros by receiving
their names as arguments.
If a macro is not parsed but the name of another macro appears
as an argument, it is interpreted as opaque text.
.Pp
The
.Em Scope
column, if applicable, describes closure rules.
.Ss Block full-explicit
Multi-line scope closed by an explicit closing macro.
All macros contains bodies; only
.Sx \&Bf
and
.Pq optionally
.Sx \&Bl
contain a head.
.Bd -literal -offset indent
\&.Yo \(lB\-arg \(lBparm...\(rB\(rB \(lBhead...\(rB
\(lBbody...\(rB
\&.Yc
.Ed
.Bl -column "MacroX" "CallableX" "ParsedX" "closed by XXX" -offset indent
.It Em Macro Ta Em Callable Ta Em Parsed Ta Em Scope
.It Sx \&Bd  Ta    \&No     Ta    \&No     Ta    closed by Sx \&Ed
.It Sx \&Bf  Ta    \&No     Ta    \&No     Ta    closed by Sx \&Ef
.It Sx \&Bk  Ta    \&No     Ta    \&No     Ta    closed by Sx \&Ek
.It Sx \&Bl  Ta    \&No     Ta    \&No     Ta    closed by Sx \&El
.It Sx \&Ed  Ta    \&No     Ta    \&No     Ta    opened by Sx \&Bd
.It Sx \&Ef  Ta    \&No     Ta    \&No     Ta    opened by Sx \&Bf
.It Sx \&Ek  Ta    \&No     Ta    \&No     Ta    opened by Sx \&Bk
.It Sx \&El  Ta    \&No     Ta    \&No     Ta    opened by Sx \&Bl
.El
.Ss Block full-implicit
Multi-line scope closed by end-of-file or implicitly by another macro.
All macros have bodies; some
.Po
.Sx \&It Fl bullet ,
.Fl hyphen ,
.Fl dash ,
.Fl enum ,
.Fl item
.Pc
don’t have heads; only one
.Po
.Sx \&It
in
.Sx \&Bl Fl column
.Pc
has multiple heads.
.Bd -literal -offset indent
```



```
\&.Yo \(lB\-arg \(lBparm...\(rB\(rB \(lBhead... \(lBTa head...\(rB\(rB
\(lBbody...\(rB
.Ed
.Bl -column "MacroX" "CallableX" "ParsedX" "closed by XXXXXXXXXX" -offset inden
.It Em Macro Ta Em Callable Ta Em Parsed Ta Em Scope
.It Sx \&It Ta \&No Ta Yes  Ta closed by Sx \&It , Sx \&El
.It Sx \&Nd Ta \&No Ta \&No Ta closed by Sx \&Sh
.It Sx \&Nm Ta \&No Ta Yes  Ta closed by Sx \&Nm , Sx \&Sh , Sx \&Ss
.It Sx \&Sh Ta \&No Ta Yes  Ta closed by Sx \&Sh
.It Sx \&Ss Ta \&No Ta Yes  Ta closed by Sx \&Sh , Sx \&Ss
.El
.Pp
Note that the
.Sx \&Nm
macro is a
.Sx Block full-implicit
macro only when invoked as the first macro
in a
.Em SYNOPSIS
section line, else it is
.Sx In-line .
.Ss Block partial-explicit
Like block full-explicit, but also with single-line scope.
Each has at least a body and, in limited circumstances, a head
.Po
.Sx \&Fo ,
.Sx \&Eo
.Pc
and/or tail
.Pq Sx \&Ec .
.Bd -literal -offset indent
\&.Yo \(lB\-arg \(lBparm...\(rB\(rB \(lBhead...\(rB
\(lBbody...\(rB
\&.Yc \(lBtail...\(rB

\&.Yo \(lB\-arg \(lBparm...\(rB\(rB \(lBhead...\(rB \
\(lBbody...\(rB \&Yc \(lBtail...\(rB
.Ed
.Bl -column "MacroX" "CallableX" "ParsedX" "closed by XXXX" -offset indent
.It Em Macro Ta Em Callable Ta Em Parsed Ta Em Scope
.It Sx \&Ac  Ta    Yes      Ta    Yes      Ta    opened by Sx \&Ao
.It Sx \&Ao  Ta    Yes      Ta    Yes      Ta    closed by Sx \&Ac
.It Sx \&Bc  Ta    Yes      Ta    Yes      Ta    closed by Sx \&Bo
.It Sx \&Bo  Ta    Yes      Ta    Yes      Ta    opened by Sx \&Bc
.It Sx \&Brc Ta    Yes      Ta    Yes      Ta    opened by Sx \&Bro
.It Sx \&Bro Ta    Yes      Ta    Yes      Ta    closed by Sx \&Brc
.It Sx \&Dc  Ta    Yes      Ta    Yes      Ta    opened by Sx \&Do
.It Sx \&Do  Ta    Yes      Ta    Yes      Ta    closed by Sx \&Dc
.It Sx \&Ec  Ta    Yes      Ta    Yes      Ta    opened by Sx \&Eo
.It Sx \&Eo  Ta    Yes      Ta    Yes      Ta    closed by Sx \&Ec
.It Sx \&Fc  Ta    Yes      Ta    Yes      Ta    opened by Sx \&Fo
.It Sx \&Fo  Ta    \&No     Ta    \&No     Ta    closed by Sx \&Fc
.It Sx \&Oc  Ta    Yes      Ta    Yes      Ta    closed by Sx \&Oo
.It Sx \&Oo  Ta    Yes      Ta    Yes      Ta    opened by Sx \&Oc
.It Sx \&Pc  Ta    Yes      Ta    Yes      Ta    closed by Sx \&Po
.It Sx \&Po  Ta    Yes      Ta    Yes      Ta    opened by Sx \&Pc
.It Sx \&Qc  Ta    Yes      Ta    Yes      Ta    opened by Sx \&Oo
.It Sx \&Qo  Ta    Yes      Ta    Yes      Ta    closed by Sx \&Oc
.It Sx \&Re  Ta    \&No     Ta    \&No     Ta    opened by Sx \&Rs
.It Sx \&Rs  Ta    \&No     Ta    \&No     Ta    closed by Sx \&Re
.It Sx \&Sc  Ta    Yes      Ta    Yes      Ta    opened by Sx \&So
.It Sx \&So  Ta    Yes      Ta    Yes      Ta    closed by Sx \&Sc
.It Sx \&Xc  Ta    Yes      Ta    Yes      Ta    opened by Sx \&Xo
.It Sx \&Xo  Ta    Yes      Ta    Yes      Ta    closed by Sx \&Xc
.El
.Ss Block partial-implicit
```

```
Like block full-implicit, but with single-line scope closed by the
end of the line.
.Bd -literal -offset indent
\&.Yo \(lB\-arg \(lBval...\(rB\(rB \(lBbody...\(rB \(lBres...\(rB
.Ed
.Bl -column "MacroX" "CallableX" "ParsedX" -offset indent
.It Em Macro Ta Em Callable Ta Em Parsed
.It Sx \&Aq  Ta    Yes      Ta    Yes
.It Sx \&Bq  Ta    Yes      Ta    Yes
.It Sx \&Brq Ta    Yes      Ta    Yes
.It Sx \&D1  Ta    \&No     Ta    \&Yes
.It Sx \&Dl  Ta    \&No     Ta    Yes
.It Sx \&Dq  Ta    Yes      Ta    Yes
.It Sx \&Op  Ta    Yes      Ta    Yes
.It Sx \&Pq  Ta    Yes      Ta    Yes
.It Sx \&Ql  Ta    Yes      Ta    Yes
.It Sx \&Qq  Ta    Yes      Ta    Yes
.It Sx \&Sq  Ta    Yes      Ta    Yes
.It Sx \&Vt  Ta    Yes      Ta    Yes
.El
.Pp
Note that the
.Sx \&Vt
macro is a
.Sx Block partial-implicit
only when invoked as the first macro
in a
.Em SYNOPSIS
section line, else it is
.Sx In-line .
.Ss Special block macro
The
.Sx \&Ta
macro can only be used below
.Sx \&It
in
.Sx \&Bl Fl column
lists.
It delimits blocks representing table cells;
these blocks have bodies, but no heads.
.Bl -column "MacroX" "CallableX" "ParsedX" "closed by XXXX" -offset indent
.It Em Macro Ta Em Callable Ta Em Parsed Ta Em Scope
.It Sx \&Ta  Ta    Yes      Ta    Yes    Ta closed by Sx \&Ta , Sx \&It
.El
.Ss In-line
Closed by the end of the line, fixed argument lengths,
and/or subsequent macros.
In-line macros have only text children.
If a number (or inequality) of arguments is
.Pq n ,
then the macro accepts an arbitrary number of arguments.
.Bd -literal -offset indent
\&.Yo \(lB\-arg \(lBval...\(rB\(rB \(lBargs...\(rB \(lBres...\(rB

\&.Yo \(lB\-arg \(lBval...\(rB\(rB \(lBargs...\(rB Yc...

\&.Yo \(lB\-arg \(lBval...\(rB\(rB arg0 arg1 argN
.Ed
.Bl -column "MacroX" "CallableX" "ParsedX" "Arguments" -offset indent
.It Em Macro Ta Em Callable Ta Em Parsed Ta Em Arguments
.It Sx \&%A  Ta    \&No     Ta    \&No     Ta    >0
.It Sx \&%B  Ta    \&No     Ta    \&No     Ta    >0
.It Sx \&%C  Ta    \&No     Ta    \&No     Ta    >0
.It Sx \&%D  Ta    \&No     Ta    \&No     Ta    >0
.It Sx \&%I  Ta    \&No     Ta    \&No     Ta    >0
.It Sx \&%J  Ta    \&No     Ta    \&No     Ta    >0
```



```
.It Sx \&%N  Ta    \&No     Ta    \&No     Ta    >0
.It Sx \&%O  Ta    \&No     Ta    \&No     Ta    >0
.It Sx \&%P  Ta    \&No     Ta    \&No     Ta    >0
.It Sx \&%Q  Ta    \&No     Ta    \&No     Ta    >0
.It Sx \&%R  Ta    \&No     Ta    \&No     Ta    >0
.It Sx \&%T  Ta    \&No     Ta    \&No     Ta    >0
.It Sx \&%U  Ta    \&No     Ta    \&No     Ta    >0
.It Sx \&%V  Ta    \&No     Ta    \&No     Ta    >0
.It Sx \&Ad  Ta    Yes      Ta    Yes      Ta    >0
.It Sx \&An  Ta    Yes      Ta    Yes      Ta    >0
.It Sx \&Ap  Ta    Yes      Ta    Yes      Ta    0
.It Sx \&Ar  Ta    Yes      Ta    Yes      Ta    n
.It Sx \&At  Ta    Yes      Ta    Yes      Ta    1
.It Sx \&Bsx Ta    Yes      Ta    Yes      Ta    n
.It Sx \&Bt  Ta    \&No     Ta    \&No     Ta    0
.It Sx \&Bx  Ta    Yes      Ta    Yes      Ta    n
.It Sx \&Cd  Ta    Yes      Ta    Yes      Ta    >0
.It Sx \&Cm  Ta    Yes      Ta    Yes      Ta    >0
.It Sx \&Db  Ta    \&No     Ta    \&No     Ta    1
.It Sx \&Dd  Ta    \&No     Ta    \&No     Ta    n
.It Sx \&Dt  Ta    \&No     Ta    \&No     Ta    n
.It Sx \&Dv  Ta    Yes      Ta    Yes      Ta    >0
.It Sx \&Dx  Ta    Yes      Ta    Yes      Ta    n
.It Sx \&Em  Ta    Yes      Ta    Yes      Ta    >0
.It Sx \&En  Ta    \&No     Ta    \&No     Ta    0
.It Sx \&Er  Ta    Yes      Ta    Yes      Ta    >0
.It Sx \&Es  Ta    \&No     Ta    \&No     Ta    0
.It Sx \&Ev  Ta    Yes      Ta    Yes      Ta    >0
.It Sx \&Ex  Ta    \&No     Ta    \&No     Ta    n
.It Sx \&Fa  Ta    Yes      Ta    Yes      Ta    >0
.It Sx \&Fd  Ta    \&No     Ta    \&No     Ta    >0
.It Sx \&Fl  Ta    Yes      Ta    Yes      Ta    n
.It Sx \&Fn  Ta    Yes      Ta    Yes      Ta    >0
.It Sx \&Fr  Ta    \&No     Ta    \&No     Ta    n
.It Sx \&Ft  Ta    Yes      Ta    Yes      Ta    >0
.It Sx \&Fx  Ta    Yes      Ta    Yes      Ta    n
.It Sx \&Hf  Ta    \&No     Ta    \&No     Ta    n
.It Sx \&Ic  Ta    Yes      Ta    Yes      Ta    >0
.It Sx \&In  Ta    \&No     Ta    \&No     Ta    1
.It Sx \&Lb  Ta    \&No     Ta    \&No     Ta    1
.It Sx \&Li  Ta    Yes      Ta    Yes      Ta    >0
.It Sx \&Lk  Ta    Yes      Ta    Yes      Ta    >0
.It Sx \&Lp  Ta    \&No     Ta    \&No     Ta    0
.It Sx \&Ms  Ta    Yes      Ta    Yes      Ta    >0
.It Sx \&Mt  Ta    Yes      Ta    Yes      Ta    >0
.It Sx \&Nm  Ta    Yes      Ta    Yes      Ta    n
.It Sx \&No  Ta    Yes      Ta    Yes      Ta    0
.It Sx \&Ns  Ta    Yes      Ta    Yes      Ta    0
.It Sx \&Nx  Ta    Yes      Ta    Yes      Ta    n
.It Sx \&Os  Ta    \&No     Ta    \&No     Ta    n
.It Sx \&Ot  Ta    \&No     Ta    \&No     Ta    n
.It Sx \&Ox  Ta    Yes      Ta    Yes      Ta    n
.It Sx \&Pa  Ta    Yes      Ta    Yes      Ta    n
.It Sx \&Pf  Ta    Yes      Ta    Yes      Ta    1
.It Sx \&Pp  Ta    \&No     Ta    \&No     Ta    0
.It Sx \&Rv  Ta    \&No     Ta    \&No     Ta    n
.It Sx \&Sm  Ta    \&No     Ta    \&No     Ta    1
.It Sx \&St  Ta    \&No     Ta    Yes      Ta    1
.It Sx \&Sx  Ta    Yes      Ta    Yes      Ta    >0
.It Sx \&Sy  Ta    Yes      Ta    Yes      Ta    >0
.It Sx \&Tn  Ta    Yes      Ta    Yes      Ta    >0
.It Sx \&Ud  Ta    \&No     Ta    \&No     Ta    0
.It Sx \&Ux  Ta    Yes      Ta    Yes      Ta    n
.It Sx \&Va  Ta    Yes      Ta    Yes      Ta    n
.It Sx \&Vt  Ta    Yes      Ta    Yes      Ta    >0
.It Sx \&Xr  Ta    Yes      Ta    Yes      Ta    >0
```

```
.It Sx \&br  Ta    \&No     Ta    \&No     Ta    0
.It Sx \&sp  Ta    \&No     Ta    \&No     Ta    1
.El
.Ss Delimiters
When a macro argument consists of one single input character
considered as a delimiter, the argument gets special handling.
This does not apply when delimiters appear in arguments containing
more than one character.
Consequently, to prevent special handling and just handle it
like any other argument, a delimiter can be escaped by prepending
a zero-width space
.Pq Sq \e& .
In text lines, delimiters never need escaping, but may be used
as normal punctuation.
.Pp
For many macros, when the leading arguments are opening delimiters,
these delimiters are put before the macro scope,
and when the trailing arguments are closing delimiters,
these delimiters are put after the macro scope.
For example,
.Pp
.D1 Pf \. \&Aq "( [ word ] ) ."
.Pp
renders as:
.Pp
.D1 Aq ( [ word ] ) .
.Pp
Opening delimiters are:
.Pp
.Bl -tag -width Ds -offset indent -compact
.It \&(
left parenthesis
.It \&[
left bracket
.El
.Pp
Closing delimiters are:
.Pp
.Bl -tag -width Ds -offset indent -compact
.It \&.
period
.It \&,
comma
.It \&:
colon
.It \&;
semicolon
.It \&)
right parenthesis
.It \&]
right bracket
.It \&?
question mark
.It \&!
exclamation mark
.El
.Pp
Note that even a period preceded by a backslash
.Pq Sq \e.\&
gets this special handling; use
.Sq \e&.
to prevent that.
.Pp
Many in-line macros interrupt their scope when they encounter
delimiters, and resume their scope when more arguments follow that
are not delimiters.
```



```
For example,
.Pp
.D1 Pf \. \&Fl "a ( b | c \e*(Ba d ) e"
.Pp
renders as:
.Pp
.D1 Fl a ( b | c \*(Ba d ) e
.Pp
This applies to both opening and closing delimiters,
and also to the middle delimiter:
.Pp
.Bl -tag -width Ds -offset indent -compact
.It \&|
vertical bar
.El
.Pp
As a special case, the predefined string \e*(Ba is handled and rendered
in the same way as a plain
.Sq \&|
character.
Using this predefined string is not recommended in new manuals.
.Ss Font handling
In
.Nm
documents, usage of semantic markup is recommended in order to have
proper fonts automatically selected; only when no fitting semantic markup
is available, consider falling back to
.Sx Physical markup
macros.
Whenever any
.Nm
macro switches the
.Xr roff 5
font mode, it will automatically restore the previous font when exiting
its scope.
Manually switching the font using the
.Xr roff 5
.Ql \ef
font escape sequences is never required.
.Sh COMPATIBILITY
This section documents compatibility between mandoc and other other
troff implementations, at this time limited to GNU troff
.Pq Qq groff .
The term
.Qq historic groff
refers to groff versions before 1.17,
which featured a significant update of the
.Pa doc.tmac
file.
.Pp
Heirloom troff, the other significant troff implementation accepting
\-mdoc, is similar to historic groff.
.Pp
The following problematic behaviour is found in groff:
.ds hist (Historic groff only.)
.Pp
.Bl -dash -compact
.It
Display macros
.Po
.Sx \&Bd ,
.Sx \&Dl ,
and
.Sx \&D1
.Pc
may not be nested.
```

```
\*[hist]
.It
.Sx \&At
with unknown arguments produces no output at all.
\*[hist]
Newer groff and mandoc print
.Qq AT&T UNIX
and the arguments.
.It
.Sx \&Bl Fl column
does not recognise trailing punctuation characters when they immediately
precede tabulator characters, but treats them as normal text and
outputs a space before them.
.It
.Sx \&Bd Fl ragged compact
does not start a new line.
\*[hist]
.It
.Sx \&Dd
with non-standard arguments behaves very strangely.
When there are three arguments, they are printed verbatim.
Any other number of arguments is replaced by the current date,
but without any arguments the string
.Dq Epoch
is printed.
.It
.Sx \&Fl
does not print a dash for an empty argument.
\*[hist]
.It
.Sx \&Fn
does not start a new line unless invoked as the line macro in the
.Em SYNOPSIS
section.
\*[hist]
.It
.Sx \&Fo
with
.Pf non- Sx \&Fa
children causes inconsistent spacing between arguments.
In mandoc, a single space is always inserted between arguments.
.It
.Sx \&Ft
in the
.Em SYNOPSIS
causes inconsistent vertical spacing, depending on whether a prior
.Sx \&Fn
has been invoked.
See
.Sx \&Ft
and
.Sx \&Fn
for the normalised behaviour in mandoc.
.It
.Sx \&In
ignores additional arguments and is not treated specially in the
.Em SYNOPSIS .
\*[hist]
.It
.Sx \&It
sometimes requires a
.Fl nested
flag.
\*[hist]
In new groff and mandoc, any list may be nested by default and
.Fl enum
```



```
lists will restart the sequence only for the sub-list.
.It
.Sx \&Li
followed by a delimiter is incorrectly used in some manuals
instead of properly quoting that character, which sometimes works with
historic groff.
.It
.Sx \&Lk
only accepts a single link-name argument; the remainder is misformatted.
.It
.Sx \&Pa
does not format its arguments when used in the FILES section under
certain list types.
.It
.Sx \&Ta
can only be called by other macros, but not at the beginning of a line.
.It
.Sx \&%C
is not implemented.
.It
Historic groff only allows up to eight or nine arguments per macro input
line, depending on the exact situation.
Providing more arguments causes garbled output.
The number of arguments on one input line is not limited with mandoc.
.It
Historic groff has many un-callable macros.
Most of these (excluding some block-level macros) are callable
in new groff and mandoc.
.It
.Sq \(ba
(vertical bar) is not fully supported as a delimiter.
\*[hist]
.It
.Sq \ef
.Pq font face
and
.Sq \ef
.Pq font family face
.Sx Text Decoration
escapes behave irregularly when specified within line-macro scopes.
.It
Negative scaling units return to prior lines.
Instead, mandoc truncates them to zero.
.El
.Pp
The following features are unimplemented in mandoc:
.Pp
.Bl -dash -compact
.It
.Sx \&Bd
.Fl file Ar file .
.It
.Sx \&Bd
.Fl offset Ar center
and
.Fl offset Ar right .
Groff does not implement centred and flush-right rendering either,
but produces large indentations.
.It
The
.Sq \eh
.Pq horizontal position ,
.Sq \ev
.Pq vertical position ,
.Sq \em
.Pq text colour ,
```

```
.Sq \eM
.Pq text filling colour ,
.Sq \ez
.Pq zero-length character ,
.Sq \ew
.Pq string length ,
.Sq \ek
.Pq horizontal position marker ,
.Sq \eo
.Pq text overstrike ,
and
.Sq \es
.Pq text size
escape sequences are all discarded in mandoc.
.It
The
.Sq \ef
scaling unit is accepted by mandoc, but rendered as the default unit.
.It
In quoted literals, groff allows pairwise double-quotes to produce a
standalone double-quote in formatted output.
This is not supported by mandoc.
.El
.Sh SEE ALSO
.Xr man 1 ,
.Xr mandoc 1 ,
.Xr eqn 5 ,
.Xr man 5 ,
.Xr mandoc_char 5 ,
.Xr roff 5 ,
.Xr tbl 5
.Sh HISTORY
The
.Nm
language first appeared as a troff macro package in
.Bx 4.4 .
It was later significantly updated by Werner Lemberg and Ruslan Ermilov
in groff-1.17.
The standalone implementation that is part of the
.Xr mandoc 1
utility written by Kristaps Dzonsons appeared in
.Ox 4.6 .
in July, 2014.
.Sh AUTHORS
The
.Nm
reference was written by
.An Kristaps Dzonsons ,
.Mt kristaps@bsd.lv .
```

```
**********************************************************
    7891 Wed Jul 16 14:05:09 2014
new/usr/src/man/man5/tbl.5
mandoc import
**********************************************************
```

```
.\"
.\" Permission to use, copy, modify, and distribute this software for any
.\" purpose with or without fee is hereby granted, provided that the above
.\" copyright notice and this permission notice appear in all copies.
.\"
.\" THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES
.\" WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF
.\" MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR
.\" ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES
.\" WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN
.\" ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF
.\" OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
.\"
.\"
.\" Copyright (c) 2010, 2011 Kristaps Dzonsons <kristaps@bsd.lv>
.\" Copyright 2012 Nexenta Systems, Inc. All rights reserved.
.\"
.Dd Sep 3, 2011
.Dt TBL 5
.Os
.Sh NAME
.Nm tbl
.Nd tbl language reference for mandoc
.Sh DESCRIPTION
The
.Nm tbl
language is a table-formatting language.
It is used within
.Xr mdoc 5
and
.Xr man 5
.Ux
manual pages.
This manual describes the subset of the
.Nm
language accepted by the
.Xr mandoc 1
utility.
.Pp
Tables within
.Xr mdoc 5
or
.Xr man 5
are enclosed by the
.Sq TS
and
.Sq TE
macro tags, whose precise syntax is documented in
.Xr roff 5 .
Tables consist of a series of options on a single line, followed by the
table layout, followed by data.
.Pp
For example, the following creates a boxed table with digits centred in
the cells.
.Bd -literal -offset indent
\&.TS
tab(:) box;
c5 c5 c5.
1:2:3
4:5:6
\&.TE
```



```
.Ed
.Pp
When formatted, the following output is produced:
.Bd -filled -offset indent -compact
.TS
tab(:) box;
c5 c5 c5.
1:2:3
4:5:6
.TE
.Ed
.Pp
The
.Nm
implementation in
.Xr mandoc 1
is
.Ud
.Sh TABLE STRUCTURE
Tables are enclosed by the
.Sq TS
and
.Sq TE
.Xr roff 5
macros.
A table consists of an optional single line of table
.Sx Options
terminated by a semicolon, followed by one or more lines of
.Sx Layout
specifications terminated by a period, then
.Sx Data .
All input must be 7-bit ASCII.
Example:
.Bd -literal -offset indent
\&.TS
box tab(:);
c | c
| c | c.
1:2
3:4
\&.TE
.Ed
.Pp
Table data is
.Em pre-processed ,
that is, data rows are parsed then inserted into the underlying stream
of input data.
This allows data rows to be interspersed by arbitrary
.Xr roff 5 ,
.Xr mdoc 5 ,
and
.Xr man 5
macros such as
.Bd -literal -offset indent
\&.TS
tab(:);
c c c.
1:2:3
\&.Ao
3:2:1
\&.Ac
\&.TE
.Ed
.Pp
in the case of
.Xr mdoc 5
```

```
or
.Bd -literal -offset indent
\&.TS
tab(:);
c c c.
\&.ds ab 2
1:\e*(ab:3
\&.I
3:2:1
\&.TE
.Ed
.Pp
in the case of
.Xr man 5 .
.Ss Options
The first line of a table consists of space-separated option keys and
modifiers terminated by a semicolon.
If the first line does not have a terminating semicolon, it is assumed
that no options are specified and instead a
.Sx Layout
is processed.
Some options accept arguments enclosed by parenthesis.
The following case-insensitive options are available:
.Bl -tag -width Ds
.It Cm center
This option is not supported by
.Xr mandoc 1 .
This may also be invoked with
.Cm centre .
.It Cm delim
Accepts a two-character argument.
This option is not supported by
.Xr mandoc 1 .
.It Cm expand
This option is not supported by
.Xr mandoc 1 .
.It Cm box
Draw a single-line box around the table.
This may also be invoked with
.Cm frame .
.It Cm doublebox
Draw a double-line box around the table.
This may also be invoked with
.Cm doubleframe .
.It Cm allbox
This option is not supported by
.Xr mandoc 1 .
.It Cm tab
Accepts a single-character argument.
This character is used as a delimiter between data cells, which otherwise
defaults to the tab character.
.It Cm linesize
Accepts a natural number (all digits).
This option is not supported by
.Xr mandoc 1 .
.It Cm nokeep
This option is not supported by
.Xr mandoc 1 .
.It Cm decimalpoint
Accepts a single-character argument.
This character will be used as the decimal point with the
.Cm n
layout key.
.It Cm nospaces
This option is not supported by
.Xr mandoc 1 .
```



```
.El
.Ss Layout
The table layout follows
.Sx Options
or a
.Sq \&T&
macro invocation.
Layout specifies how data rows are displayed on output.
Each layout line corresponds to a line of data; the last layout line
applies to all remaining data lines.
Layout lines may also be separated by a comma.
Each layout cell consists of one of the following case-insensitive keys:
.Bl -tag -width Ds
.It Cm c
Centre a literal string within its column.
.It Cm r
Right-justify a literal string within its column.
.It Cm l
Left-justify a literal string within its column.
.It Cm n
Justify a number around its last decimal point.
If the decimal point is not found on the number, it’s assumed to trail
the number.
.It Cm s
Horizontally span columns from the last
.No non- Ns Cm s
data cell.
It is an error if spanning columns follow a
.Cm \-
or
.Cm \(ba
cell, or come first.
This option is not supported by
.Xr mandoc 1 .
.It Cm a
Left-justify a literal string and pad with one space.
.It Cm ^
Vertically span rows from the last
.No non- Ns Cm ^
data cell.
It is an error to invoke a vertical span on the first layout row.
Unlike a horizontal spanner, you must specify an empty cell (if it not
empty, the data is discarded) in the corresponding data cell.
.It Cm \-
Replace the data cell (its contents will be lost) with a single
horizontal line.
This may also be invoked with
.Cm _ .
.It Cm =
Replace the data cell (its contents will be lost) with a double
horizontal line.
.It Cm \(ba
Emit a vertical bar instead of data.
.It Cm \(ba\(ba
Emit a double-vertical bar instead of data.
.El
.Pp
Keys may be followed by a set of modifiers.
A modifier is either a modifier key or a natural number for specifying
the minimum width of a column.
The following case-insensitive modifier keys are available:
.Cm z ,
.Cm u ,
.Cm e ,
.Cm t ,
.Cm d ,
```

```
.Cm b ,
.Cm i ,
.Cm r ,
and
.Cm f
.Po
followed by
.Cm b ,
.Cm i ,
.Cm r ,
.Cm 3 ,
.Cm 2 ,
or
.Cm 1
.Pc .
All of these are ignored by
.Xr mandoc 1 .
.Pp
For example, the following layout specifies a centre-justified column of
minimum width 10, followed by vertical bar, followed by a left-justified
column of minimum width 10, another vertical bar, then a column
justified about the decimal point in numbers:
.Pp
.Dl c10 | l10 | n
.Ss Data
The data section follows the last layout row.
By default, cells in a data section are delimited by a tab.
This behaviour may be changed with the
.Cm tab
option.
If
.Cm _
or
.Cm =
is specified, a single or double line, respectively, is drawn across the
data field.
If
.Cm \e-
or
.Cm \e=
is specified, a line is drawn within the data field (i.e. terminating
within the cell and not draw to the border).
If the last cell of a line is
.Cm T{ ,
all subsequent lines are included as part of the cell until
.Cm T}
is specified as its own data cell.
It may then be followed by a tab
.Pq or as designated by Cm tab
or an end-of-line to terminate the row.
.Sh COMPATIBILITY
This section documents compatibility between mandoc and other
.Nm
implementations, at this time limited to GNU tbl.
.Pp
.Bl -dash -compact
.It
In GNU tbl, comments and macros are disallowed prior to the data block
of a table.
The
.Xr mandoc 1
implementation allows them.
.El
.Sh SEE ALSO
.Xr mandoc 1 ,
.Xr man 5 ,
```



```
.Xr mandoc_char 5 ,
.Xr mdoc 5 ,
.Xr roff 5
.Rs
.%A M. E. Lesk
.%T Tbl\(emA Program to Format Tables
.%D June 11, 1976
.Re
.Sh HISTORY
The tbl utility, a preprocessor for troff, was originally written by M.
E. Lesk at Bell Labs in 1975.
The GNU reimplementation of tbl, part of the groff package, was released
in 1990 by James Clark.
A standalone tbl implementation was written by Kristaps Dzonsons in
2010.
This formed the basis of the implementation that is part of the
.Xr mandoc 1
utility.
.Sh AUTHORS
This
.Nm
reference was written by
.An Kristaps Dzonsons ,
.Mt kristaps@bsd.lv .
```

```
**********************************************************
    2562 Wed Jul 16 14:05:09 2014
new/usr/src/pkg/manifests/system-man.mf
fix link for catman.
feedback from Hans
Add catman, makewhatis functionality.  Print an error if the whatis database
is missing.
mandoc import
**********************************************************
```

```
#
# This file and its contents are supplied under the terms of the
# Common Development and Distribution License ("CDDL"), version 1.0.
# You may only use this file in accordance with the terms of version
# 1.0 of the CDDL.
#
# A full copy of the text of the CDDL should have accompanied this
# source.  A copy of the CDDL is also available via the Internet at
# http://www.illumos.org/license/CDDL.
#

#
# Copyright 2012 Nexenta Systems, Inc. All rights reserved.
# Copyright 2014 Garrett D'Amore <garrett@damore.org>
#

set name=pkg.fmri value=pkg:/system/man@$(PKGVERS)
set name=pkg.description \
    value="utilities for display and formatting of reference manual pages"
set name=pkg.summary value="Reference Manual Pages Tools"
set name=info.classification \
    value="org.opensolaris.category.2008:System/Text Tools"
set name=variant.arch value=$(ARCH)
dir path=usr/bin
dir path=usr/lib
dir path=usr/share
dir path=usr/share/man
dir path=usr/share/man/man1
dir path=usr/share/man/man1m
dir path=usr/share/man/man5
file path=usr/bin/man mode=0555
file path=usr/bin/mandoc mode=0555
file path=usr/lib/mandoc_preconv mode=0555
file path=usr/share/man/man1/apropos.1
file path=usr/share/man/man1/man.1
file path=usr/share/man/man1/mandoc.1
file path=usr/share/man/man1m/catman.1m
file path=usr/share/man/man5/eqn.5
file path=usr/share/man/man5/man.5
file path=usr/share/man/man5/mandoc_char.5
file path=usr/share/man/man5/mandoc_roff.5
file path=usr/share/man/man5/mdoc.5
file path=usr/share/man/man5/tbl.5
hardlink path=usr/bin/apropos target=../../usr/bin/man
hardlink path=usr/bin/catman target=../../usr/bin/man
hardlink path=usr/bin/whatis target=../../usr/bin/man
hardlink path=usr/lib/makewhatis target=../../usr/bin/man
license lic_CDDL license=lic_CDDL
license usr/src/cmd/man/THIRDPARTYLICENSE \
    license=usr/src/cmd/man/THIRDPARTYLICENSE
license usr/src/cmd/mandoc/THIRDPARTYLICENSE \
    license=usr/src/cmd/mandoc/THIRDPARTYLICENSE
link path=usr/man target=./share/man
link path=usr/share/man/man1/whatis.1 target=apropos.1
# arguably we also need lp, for man -t support, but really we don't
# want to make this mandatory, so we don't express teh dependency here.
# gzcat/bzcat are used for displaying compressed manpages.  However,
```



```
# as we don't format such pages this way by default, lets leave the
# dependency out.
#depend fmri=compress/bzip2 type=require
#depend fmri=compress/gzip type=require
# less is the default (per user environment) pager.  We really should just
# import this into illumos-gate.
depend fmri=text/less type=require
```

```
**********************************************************
   16215 Wed Jul 16 14:05:09 2014
new/usr/src/pkg/manifests/text-doctools.mf
mandoc import
**********************************************************
```

```
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#

#
# Copyright (c) 2010, Oracle and/or its affiliates. All rights reserved.
# Copyright 2012 Nexenta Systems, Inc. All rights reserved.
# Copyright 2014 Garrett D'Amore <garrett@damore.org>
#

# man moved to system/man
set name=pkg.fmri value=pkg:/text/doctools@$(PKGVERS)
set name=pkg.description \
    value="utilities and fonts for development, display, and production of docum
set name=pkg.summary value="Documentation Tools"
set name=info.classification \
    value="org.opensolaris.category.2008:System/Text Tools"
set name=variant.arch value=$(ARCH)
dir path=usr group=sys
dir path=usr/bin
dir path=usr/lib
dir path=usr/lib/font
dir path=usr/lib/font/devpost group=lp
dir path=usr/lib/font/devpost/charlib group=lp
dir path=usr/lib/font/devpost/fontmaps group=lp
dir path=usr/lib/refer
dir path=usr/lib/refer/papers
dir path=usr/share
dir path=usr/share/lib
dir path=usr/share/lib/nterm
dir path=usr/share/lib/pub
dir path=usr/share/lib/tmac
dir path=usr/share/man
dir path=usr/share/man/man1
dir path=usr/share/man/man1m
dir path=usr/share/man/man5
file path=usr/bin/addbib mode=0555
file path=usr/bin/apropos mode=0555
file path=usr/bin/checkeq mode=0555
file path=usr/bin/checknr mode=0555
file path=usr/bin/deroff mode=0555
file path=usr/bin/diffmk mode=0555
file path=usr/bin/eqn mode=0555
file path=usr/bin/indxbib mode=0555
```



```
file path=usr/bin/lookbib mode=0555
file path=usr/bin/neqn mode=0555
file path=usr/bin/nroff mode=0555
file path=usr/bin/refer mode=0555
file path=usr/bin/roffbib mode=0555
file path=usr/bin/soelim mode=0555
file path=usr/bin/sortbib mode=0555
file path=usr/bin/ta mode=0555
file path=usr/bin/tbl mode=0555
file path=usr/bin/troff mode=0555
file path=usr/bin/ul mode=0555
file path=usr/bin/vgrind mode=0555
file path=usr/lib/font/devpost/AB group=lp mode=0444
file path=usr/lib/font/devpost/AB.name group=lp mode=0444
file path=usr/lib/font/devpost/AB.out group=lp mode=0444
file path=usr/lib/font/devpost/AI group=lp mode=0444
file path=usr/lib/font/devpost/AI.name group=lp mode=0444
file path=usr/lib/font/devpost/AI.out group=lp mode=0444
file path=usr/lib/font/devpost/AR group=lp mode=0444
file path=usr/lib/font/devpost/AR.name group=lp mode=0444
file path=usr/lib/font/devpost/AR.out group=lp mode=0444
file path=usr/lib/font/devpost/AX group=lp mode=0444
file path=usr/lib/font/devpost/AX.name group=lp mode=0444
file path=usr/lib/font/devpost/AX.out group=lp mode=0444
file path=usr/lib/font/devpost/B group=lp mode=0444
file path=usr/lib/font/devpost/B.name group=lp mode=0444
file path=usr/lib/font/devpost/B.out group=lp mode=0444
file path=usr/lib/font/devpost/BI group=lp mode=0444
file path=usr/lib/font/devpost/BI.name group=lp mode=0444
file path=usr/lib/font/devpost/BI.out group=lp mode=0444
file path=usr/lib/font/devpost/CB group=lp mode=0444
file path=usr/lib/font/devpost/CB.name group=lp mode=0444
file path=usr/lib/font/devpost/CB.out group=lp mode=0444
file path=usr/lib/font/devpost/CI group=lp mode=0444
file path=usr/lib/font/devpost/CI.name group=lp mode=0444
file path=usr/lib/font/devpost/CI.out group=lp mode=0444
file path=usr/lib/font/devpost/CO group=lp mode=0444
file path=usr/lib/font/devpost/CO.name group=lp mode=0444
file path=usr/lib/font/devpost/CO.out group=lp mode=0444
file path=usr/lib/font/devpost/CW group=lp mode=0444
file path=usr/lib/font/devpost/CW.name group=lp mode=0444
file path=usr/lib/font/devpost/CW.out group=lp mode=0444
file path=usr/lib/font/devpost/CX group=lp mode=0444
file path=usr/lib/font/devpost/CX.name group=lp mode=0444
file path=usr/lib/font/devpost/CX.out group=lp mode=0444
file path=usr/lib/font/devpost/DESC group=lp mode=0444
file path=usr/lib/font/devpost/DESC.out group=lp mode=0444
file path=usr/lib/font/devpost/G.out group=lp mode=0444
file path=usr/lib/font/devpost/GI.out group=lp mode=0444
file path=usr/lib/font/devpost/GR group=lp mode=0444
file path=usr/lib/font/devpost/GR.name group=lp mode=0444
file path=usr/lib/font/devpost/GR.out group=lp mode=0444
file path=usr/lib/font/devpost/H group=lp mode=0444
file path=usr/lib/font/devpost/H.name group=lp mode=0444
file path=usr/lib/font/devpost/H.out group=lp mode=0444
file path=usr/lib/font/devpost/HB group=lp mode=0444
file path=usr/lib/font/devpost/HB.name group=lp mode=0444
file path=usr/lib/font/devpost/HB.out group=lp mode=0444
file path=usr/lib/font/devpost/HI group=lp mode=0444
file path=usr/lib/font/devpost/HI.name group=lp mode=0444
file path=usr/lib/font/devpost/HI.out group=lp mode=0444
file path=usr/lib/font/devpost/HK.out group=lp mode=0444
file path=usr/lib/font/devpost/HL.out group=lp mode=0444
file path=usr/lib/font/devpost/HM.out group=lp mode=0444
file path=usr/lib/font/devpost/HX group=lp mode=0444
file path=usr/lib/font/devpost/HX.name group=lp mode=0444
```

```
127 file path=usr/lib/font/devpost/HX.out group=lp mode=0444
128 file path=usr/lib/font/devpost/I group=lp mode=0444
129 file path=usr/lib/font/devpost/I.name group=lp mode=0444
130 file path=usr/lib/font/devpost/I.out group=lp mode=0444
131 file path=usr/lib/font/devpost/JB group=lp mode=0444
132 file path=usr/lib/font/devpost/JB.name group=lp mode=0444
133 file path=usr/lib/font/devpost/JB.out group=lp mode=0444
134 file path=usr/lib/font/devpost/JI group=lp mode=0444
135 file path=usr/lib/font/devpost/JI.name group=lp mode=0444
136 file path=usr/lib/font/devpost/JI.out group=lp mode=0444
137 file path=usr/lib/font/devpost/JR group=lp mode=0444
138 file path=usr/lib/font/devpost/JR.name group=lp mode=0444
139 file path=usr/lib/font/devpost/JR.out group=lp mode=0444
140 file path=usr/lib/font/devpost/JX group=lp mode=0444
141 file path=usr/lib/font/devpost/JX.name group=lp mode=0444
142 file path=usr/lib/font/devpost/JX.out group=lp mode=0444
143 file path=usr/lib/font/devpost/KB group=lp mode=0444
144 file path=usr/lib/font/devpost/KB.name group=lp mode=0444
145 file path=usr/lib/font/devpost/KB.out group=lp mode=0444
146 file path=usr/lib/font/devpost/KI group=lp mode=0444
147 file path=usr/lib/font/devpost/KI.name group=lp mode=0444
148 file path=usr/lib/font/devpost/KI.out group=lp mode=0444
149 file path=usr/lib/font/devpost/KR group=lp mode=0444
150 file path=usr/lib/font/devpost/KR.name group=lp mode=0444
151 file path=usr/lib/font/devpost/KR.out group=lp mode=0444
152 file path=usr/lib/font/devpost/KX group=lp mode=0444
153 file path=usr/lib/font/devpost/KX.name group=lp mode=0444
154 file path=usr/lib/font/devpost/KX.out group=lp mode=0444
155 file path=usr/lib/font/devpost/NB group=lp mode=0444
156 file path=usr/lib/font/devpost/NB.name group=lp mode=0444
157 file path=usr/lib/font/devpost/NB.out group=lp mode=0444
158 file path=usr/lib/font/devpost/NI group=lp mode=0444
159 file path=usr/lib/font/devpost/NI.name group=lp mode=0444
160 file path=usr/lib/font/devpost/NI.out group=lp mode=0444
161 file path=usr/lib/font/devpost/NR group=lp mode=0444
162 file path=usr/lib/font/devpost/NR.name group=lp mode=0444
163 file path=usr/lib/font/devpost/NR.out group=lp mode=0444
164 file path=usr/lib/font/devpost/NX group=lp mode=0444
165 file path=usr/lib/font/devpost/NX.name group=lp mode=0444
166 file path=usr/lib/font/devpost/NX.out group=lp mode=0444
167 file path=usr/lib/font/devpost/PA group=lp mode=0444
168 file path=usr/lib/font/devpost/PA.name group=lp mode=0444
169 file path=usr/lib/font/devpost/PA.out group=lp mode=0444
170 file path=usr/lib/font/devpost/PB group=lp mode=0444
171 file path=usr/lib/font/devpost/PB.name group=lp mode=0444
172 file path=usr/lib/font/devpost/PB.out group=lp mode=0444
173 file path=usr/lib/font/devpost/PI group=lp mode=0444
174 file path=usr/lib/font/devpost/PI.name group=lp mode=0444
175 file path=usr/lib/font/devpost/PI.out group=lp mode=0444
176 file path=usr/lib/font/devpost/PX group=lp mode=0444
177 file path=usr/lib/font/devpost/PX.name group=lp mode=0444
178 file path=usr/lib/font/devpost/PX.out group=lp mode=0444
179 file path=usr/lib/font/devpost/R group=lp mode=0444
180 file path=usr/lib/font/devpost/R.name group=lp mode=0444
181 file path=usr/lib/font/devpost/R.out group=lp mode=0444
182 file path=usr/lib/font/devpost/S group=lp mode=0444
183 file path=usr/lib/font/devpost/S.name group=lp mode=0444
184 file path=usr/lib/font/devpost/S.out group=lp mode=0444
185 file path=usr/lib/font/devpost/S1 group=lp mode=0444
186 file path=usr/lib/font/devpost/S1.name group=lp mode=0444
187 file path=usr/lib/font/devpost/S1.out group=lp mode=0444
188 file path=usr/lib/font/devpost/VB group=lp mode=0444
189 file path=usr/lib/font/devpost/VB.name group=lp mode=0444
190 file path=usr/lib/font/devpost/VB.out group=lp mode=0444
191 file path=usr/lib/font/devpost/VI group=lp mode=0444
192 file path=usr/lib/font/devpost/VI.name group=lp mode=0444
```



```
193 file path=usr/lib/font/devpost/VI.out group=lp mode=0444
194 file path=usr/lib/font/devpost/VR group=lp mode=0444
195 file path=usr/lib/font/devpost/VR.name group=lp mode=0444
196 file path=usr/lib/font/devpost/VR.out group=lp mode=0444
197 file path=usr/lib/font/devpost/VX group=lp mode=0444
198 file path=usr/lib/font/devpost/VX.name group=lp mode=0444
199 file path=usr/lib/font/devpost/VX.out group=lp mode=0444
200 file path=usr/lib/font/devpost/ZD group=lp mode=0444
201 file path=usr/lib/font/devpost/ZD.name group=lp mode=0444
202 file path=usr/lib/font/devpost/ZD.out group=lp mode=0444
203 file path=usr/lib/font/devpost/ZI group=lp mode=0444
204 file path=usr/lib/font/devpost/ZI.name group=lp mode=0444
205 file path=usr/lib/font/devpost/ZI.out group=lp mode=0444
206 file path=usr/lib/font/devpost/charlib/12 group=lp mode=0444
207 file path=usr/lib/font/devpost/charlib/14 group=lp mode=0444
208 file path=usr/lib/font/devpost/charlib/34 group=lp mode=0444
209 file path=usr/lib/font/devpost/charlib/BRACKETS_NOTE group=lp mode=0444
210 file path=usr/lib/font/devpost/charlib/Fi group=lp mode=0444
211 file path=usr/lib/font/devpost/charlib/Fl group=lp mode=0444
212 file path=usr/lib/font/devpost/charlib/L1 group=lp mode=0444
213 file path=usr/lib/font/devpost/charlib/L1.map group=lp mode=0444
214 file path=usr/lib/font/devpost/charlib/Lb group=lp mode=0444
215 file path=usr/lib/font/devpost/charlib/Lb.map group=lp mode=0444
216 file path=usr/lib/font/devpost/charlib/README group=lp mode=0444
217 file path=usr/lib/font/devpost/charlib/Sl group=lp mode=0444
218 file path=usr/lib/font/devpost/charlib/bx group=lp mode=0444
219 file path=usr/lib/font/devpost/charlib/ci group=lp mode=0444
220 file path=usr/lib/font/devpost/charlib/ff group=lp mode=0444
221 file path=usr/lib/font/devpost/charlib/lc group=lp mode=0444
222 file path=usr/lib/font/devpost/charlib/lf group=lp mode=0444
223 file path=usr/lib/font/devpost/charlib/lh group=lp mode=0444
224 file path=usr/lib/font/devpost/charlib/ob group=lp mode=0444
225 file path=usr/lib/font/devpost/charlib/rc group=lp mode=0444
226 file path=usr/lib/font/devpost/charlib/rf group=lp mode=0444
227 file path=usr/lib/font/devpost/charlib/rh group=lp mode=0444
228 file path=usr/lib/font/devpost/charlib/sq group=lp mode=0444
229 file path=usr/lib/font/devpost/charlib/~ group=lp mode=0444
230 file path=usr/lib/font/devpost/fontmaps/post group=lp mode=0444
231 file path=usr/lib/font/makedev group=lp mode=0555
231 file path=usr/lib/getNAME mode=0555
232 file path=usr/lib/makewhatis mode=0555
232 file path=usr/lib/refer/hunt mode=0555
233 file path=usr/lib/refer/inv mode=0555
234 file path=usr/lib/refer/mkey mode=0555
235 file path=usr/lib/refer/papers/Rbstjissue
236 file path=usr/lib/refer/papers/Rv7man
237 file path=usr/lib/refer/papers/runinv mode=0755
238 file path=usr/lib/vfontedpr mode=0555
239 file path=usr/lib/vgrindefs mode=0444
240 file path=usr/share/lib/nterm/tab.2631
241 file path=usr/share/lib/nterm/tab.2631-c
242 file path=usr/share/lib/nterm/tab.2631-e
243 file path=usr/share/lib/nterm/tab.300
244 file path=usr/share/lib/nterm/tab.300-12
245 file path=usr/share/lib/nterm/tab.300S
246 file path=usr/share/lib/nterm/tab.300S-12
247 file path=usr/share/lib/nterm/tab.37
248 file path=usr/share/lib/nterm/tab.382
249 file path=usr/share/lib/nterm/tab.4000A
250 file path=usr/share/lib/nterm/tab.450
251 file path=usr/share/lib/nterm/tab.450-12
252 file path=usr/share/lib/nterm/tab.832
253 file path=usr/share/lib/nterm/tab.8510
254 file path=usr/share/lib/nterm/tab.X
255 file path=usr/share/lib/nterm/tab.lp
256 file path=usr/share/lib/nterm/tab.tn300
```

```
257 file path=usr/share/lib/pub/ascii
258 file path=usr/share/lib/pub/eqnchar
259 file path=usr/share/lib/pub/greek
260 file path=usr/share/lib/pub/iso
261 file path=usr/share/lib/tmac/acm.me
262 file path=usr/share/lib/tmac/an
263 file path=usr/share/lib/tmac/ansun
264 file path=usr/share/lib/tmac/ansun.tbl
265 file path=usr/share/lib/tmac/bib
266 file path=usr/share/lib/tmac/chars.me
267 file path=usr/share/lib/tmac/deltext.me
268 file path=usr/share/lib/tmac/e
269 file path=usr/share/lib/tmac/eqn.me
270 file path=usr/share/lib/tmac/float.me
271 file path=usr/share/lib/tmac/footnote.me
272 file path=usr/share/lib/tmac/index.me
273 file path=usr/share/lib/tmac/local.me
274 file path=usr/share/lib/tmac/m
275 file path=usr/share/lib/tmac/mmn
276 file path=usr/share/lib/tmac/mmt
277 file path=usr/share/lib/tmac/ms.acc
278 file path=usr/share/lib/tmac/ms.cov
279 file path=usr/share/lib/tmac/ms.eqn
280 file path=usr/share/lib/tmac/ms.ref
281 file path=usr/share/lib/tmac/ms.tbl
282 file path=usr/share/lib/tmac/ms.ths
283 file path=usr/share/lib/tmac/ms.toc
284 file path=usr/share/lib/tmac/null.me
285 file path=usr/share/lib/tmac/refer.me
286 file path=usr/share/lib/tmac/s
287 file path=usr/share/lib/tmac/sh.me
288 file path=usr/share/lib/tmac/tbl.me
289 file path=usr/share/lib/tmac/thesis.me
290 file path=usr/share/lib/tmac/tmac.bib
291 file path=usr/share/lib/tmac/tmac.vgrind
292 file path=usr/share/lib/tmac/tz.map
293 file path=usr/share/lib/tmac/v
294 file path=usr/share/lib/tmac/vgrind
295 file path=usr/share/man/man1/addbib.1
297 file path=usr/share/man/man1/apropos.1
296 file path=usr/share/man/man1/checknr.1
297 file path=usr/share/man/man1/deroff.1
298 file path=usr/share/man/man1/diffmk.1
299 file path=usr/share/man/man1/eqn.1
300 file path=usr/share/man/man1/indxbib.1
301 file path=usr/share/man/man1/lookbib.1
304 file path=usr/share/man/man1/man.1
302 file path=usr/share/man/man1/nroff.1
303 file path=usr/share/man/man1/refer.1
304 file path=usr/share/man/man1/roffbib.1
305 file path=usr/share/man/man1/soelim.1
306 file path=usr/share/man/man1/sortbib.1
307 file path=usr/share/man/man1/tbl.1
308 file path=usr/share/man/man1/troff.1
309 file path=usr/share/man/man1/ul.1
310 file path=usr/share/man/man1/vgrind.1
314 file path=usr/share/man/man1/whatis.1
315 file path=usr/share/man/man1m/catman.1m
311 file path=usr/share/man/man5/eqnchar.5
317 file path=usr/share/man/man5/man.5
318 file path=usr/share/man/man5/mansun.5
312 file path=usr/share/man/man5/me.5
313 file path=usr/share/man/man5/mm.5
314 file path=usr/share/man/man5/ms.5
315 file path=usr/share/man/man5/vgrindefs.5
323 hardlink path=usr/bin/catman target=../../usr/bin/apropos
```



```
324 hardlink path=usr/bin/man target=../../usr/bin/apropos
325 hardlink path=usr/bin/whatis target=../../usr/bin/apropos
316 hardlink path=usr/lib/font/devpost/charlib/~= target=./~
317 hardlink path=usr/share/lib/nterm/tab.300s \
318     target=../../../../usr/share/lib/nterm/tab.300S
319 hardlink path=usr/share/lib/nterm/tab.300s-12 \
320     target=../../../../usr/share/lib/nterm/tab.300S-12
321 hardlink path=usr/share/lib/nterm/tab.4000a \
322     target=../../../../usr/share/lib/nterm/tab.4000A
323 legacy pkg=SUNWdoc \
324     desc="utilities and fonts for development, display, and production of docume
325     name="Documentation Tools"
326 license cr_Sun license=cr_Sun
327 license usr/src/cmd/checkeq/THIRDPARTYLICENSE \
328     license=usr/src/cmd/checkeq/THIRDPARTYLICENSE
329 license usr/src/cmd/checknr/THIRDPARTYLICENSE \
330     license=usr/src/cmd/checknr/THIRDPARTYLICENSE
331 license usr/src/cmd/eqn/THIRDPARTYLICENSE \
332     license=usr/src/cmd/eqn/THIRDPARTYLICENSE
343 license usr/src/cmd/man/src/THIRDPARTYLICENSE \
344     license=usr/src/cmd/man/src/THIRDPARTYLICENSE
333 license usr/src/cmd/refer/THIRDPARTYLICENSE \
334     license=usr/src/cmd/refer/THIRDPARTYLICENSE
335 license usr/src/cmd/soelim/THIRDPARTYLICENSE \
336     license=usr/src/cmd/soelim/THIRDPARTYLICENSE
337 license usr/src/cmd/tbl/THIRDPARTYLICENSE \
338     license=usr/src/cmd/tbl/THIRDPARTYLICENSE
339 license usr/src/cmd/ul/THIRDPARTYLICENSE \
340     license=usr/src/cmd/ul/THIRDPARTYLICENSE
341 license usr/src/cmd/vgrind/THIRDPARTYLICENSE \
342     license=usr/src/cmd/vgrind/THIRDPARTYLICENSE
343 link path=usr/lib/tmac target=../share/lib/tmac
356 link path=usr/man target=./share/man
344 link path=usr/share/man/man1/checkeq.1 target=eqn.1
345 link path=usr/share/man/man1/neqn.1 target=eqn.1
346 depend fmri=system/man type=require
359 # lp is used by man -t and -r
360 depend fmri=print/lp/print-client-commands type=require
361 # awk is used by catman (via makewhatis)
362 depend fmri=system/extended-system-utilities type=require
363 # gpic is used as a preprocessor by the apropos command
364 depend fmri=text/groff type=require
365 # less is the default (per user environment) pager
366 depend fmri=text/less type=require
```