

```
new/usr/src/cmd/man/Makefile  
*****  
993 Tue Jul 15 19:41:37 2014  
new/usr/src/cmd/man/Makefile  
Add catman, makewhatis functionality. Print an error if the whatis database  
is missing.  
*****  
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 #  
16 PROG= man  
17 LINKS= apropos whatis catman  
18 LIBLINKS = makewhatis  
19 LINKS= apropos whatis  
19 OBJS= makewhatis.o man.o stringlist.o  
20 SRCS= $(OBJS:%.o=%.c)  
22 include $(SRC)/cmd/Makefile.cmd  
24 CFLAGS += $(CCVERBOSE)  
26 ROOTLINKS= $(LINKS:=%$(ROOTBIN)/%) $(LIBLINKS:=%$(ROOTLIB)/%)  
25 ROOTLINKS= $(LINKS:=%$(ROOTBIN)/%)  
28 .KEEP_STATE :  
30 all: $(PROG)  
32 clean:  
33   $(RM) $(OBJS)  
35 install: all $(ROOTPROG) $(ROOTLINKS)  
37 lint: lint_SRCS  
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
```

```
*****
33672 Tue Jul 15 19:41:38 2014
new/usr/src/cmd/man/man.c
Add catman, makewhatis functionality. Print an error if the whatis database
is missing.
*****
unchanged_portion_omitted_
152 static int      all = 0;
153 static int      apropos = 0;
154 static int      debug = 0;
155 static int      found = 0;
156 static int      list = 0;
157 static int      makewhatis = 0;
158 static int      printmp = 0;
159 static int      sargs = 0;
160 static int      psoutput = 0;
161 static int      whatis = 0;
162 static int      makewhatishere = 0;
164 static char     *mansec;
165 static char     *pager;
167 static char     *addlocale(char *);
168 static struct man_node *build_manpath(char **, int);
169 static void      do_makewhatis(struct man_node *);
170 static char     *check_config(char *);
171 static int      cmp(const void *, const void *);
172 static int      dupcheck(struct man_node *, struct dupnode **);
173 static int      format(char *, char *, char *, char *);
174 static void      free_dupnode(struct dupnode *);
175 static void      free_manp(struct man_node *manp);
176 static void      freev(char **);
177 static void      fullpaths(struct man_node **);
178 static void      get_all_sect(struct man_node *);
179 static int      getdirs(char *, char ***, int);
180 static void      getpath(struct man_node *, char **);
181 static void      getsect(struct man_node *, char **);
182 static void      init_bintoman(void);
183 static void      lower(char *);
184 static void      mandir(char **, char *, char *, int);
185 static int      manual(struct man_node *, char *);
186 static char     *map_section(char *, char *);
187 static char     *path_to_manpath(char *);
188 static void      print_manpath(struct man_node *);
189 static void      search_whatis(char *, char *);
190 static int      searchdir(char *, char *, char *);
191 static void      sortdir(DIR *, char ***);
192 static char     **split(char *, char);
193 static void      usage_man(void);
194 static void      usage_whatapro(void);
195 static void      usage_catman(void);
196 static void      usage_makewhatis(void);
197 static void      whatapro(struct man_node *, char *);
199 static char     language[MAXPATHLEN]; /* LC_MESSAGES */
200 static char     localedir[MAXPATHLEN]; /* locale specific path component */
202 static char     *newsection = NULL;
204 static int      manwidth = 0;
206 extern const char *__progname;
208 int             main(int argc, char **argv)
```

```
210 {
211     int          c, i;
212     char         **pathv;
213     char         *manpath = NULL;
214     static struct man_node *mandirs = NULL;
215     int          bmp_flags = 0;
216     int          ret = 0;
217     char         *opts;
218     char         *mwstr;
219     int          catman = 0;
221     (void) setlocale(LC_ALL, "");
222     (void) strcpy(language, setlocale(LC_MESSAGES, (char *)NULL));
223     if (strcmp("C", language) != 0)
224         (void) strlcpy(localedir, language, MAXPATHLEN);
226 #if !defined(TEXT_DOMAIN)
227 #define TEXT_DOMAIN "SYS_TEST"
228#endif
229     (void) textdomain(TEXT_DOMAIN);
231     if (strcmp(__progname, "apropos") == 0) {
232         apropos++;
233         opts = "M:ds:";
234     } else if (strcmp(__progname, "whatis") == 0) {
235         apropos++;
236         whatis++;
237         opts = "M:ds:";
238     } else if (strcmp(__progname, "catman") == 0) {
239         catman++;
240         makewhatis++;
241         opts = "M:w";
242     } else if (strcmp(__progname, "makewhatis") == 0) {
243         makewhatis++;
244         makewhatishere++;
245         manpath = ".";
246         opts = "";
247     } else {
248         opts = "M:adfkpls:tw";
249     }
251     opterr = 0;
252     while ((c = getopt(argc, argv, opts)) != -1) {
253         switch (c) {
254             case 'M': /* Respecify path for man pages */
255                 manpath = optarg;
256                 break;
257             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;
299
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 }
311 init_bintoman();
312 if (manpath == NULL && (manpath = getenv("MANPATH")) == NULL) {
313     if ((manpath = getenv("PATH")) != NULL)
314         bmp_flags = BMP_ISPATH | BMP_APPEND_DEFMANDIR;
315     else
316         manpath = DEFMANDIR;
317 }
318 pathv = split(manpath, ':');
319 mandirs = build_manpath(pathv, bmp_flags);
320 free(pathv);
321 fullpaths(&mandirs);
322
323 if (makewhatis) {
324     do_makewhatis(mandirs);
325     exit(0);
326 }
327
328 if (printmp) {
329     print_manpath(mandirs);
330     exit(0);
331 }
332
333 /* Collect environment information */
334 if (isatty(STDOUT_FILENO) && (mwstr = getenv("MANWIDTH")) != NULL &&
335     *mwstr != '\0') {
336     if (strcasecmp(mwstr, "tty") == 0) {
337         struct winsize ws;
338
339         if (ioctl(0, TIOCGWINSZ, &ws) != 0)
340             warn("TIOCGWINSZ");
341

```

```

342             manwidth = ws.ws_col;
343         } else {
344             manwidth = (int)strtol(mwstr, (char **)NULL, 10);
345             if (manwidth < 0)
346                 manwidth = 0;
347         }
348     }
349 }
350 if (manwidth != 0) {
351     DPRINTF(" -- Using non-standard page width: %d\n", manwidth);
352 }
353
354 if ((pager = getenv("PAGER")) == NULL || *pager == '\0')
355     pager = PAGER;
356 DPRINTF(" -- Using pager: %s\n", pager);
357
358 for (i = 0; i < argc; i++) {
359     char *cmd;
360     static struct man_node *mp;
361     char *pv[2];
362
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     }
378
379     if (apropos)
380         whatapro(mp, argv[i]);
381     else
382         ret += manual(mp, argv[i]);
383
384     if (mp != NULL && mp != mandirs) {
385         free(pv[0]);
386         free_manp(mp);
387     }
388 }
389
390 return (ret == 0 ? 0 : 1);
391 }
392
393 unchanged portion omitted
394
395 static void
396 search_whatis(char *whatpath, char *word)
397 {
398     FILE *fp;
399     char *line = NULL;
400     size_t linecap = 0;
401     char *pkwd;
402     regex_t preg;
403     char **ss = NULL;
404     char s[MAXNAMELEN];
405     int i;
406
407     if ((fp = fopen(whatpath, "r")) == NULL)
408         perror(whatpath);
409
410     while (fgetline(&line, &linecap, fp) != -1) {
411         if (pregexec(preg, line, &i) == REG_NOMATCH)
412             continue;
413
414         if (ss != NULL)
415             ss[ss - 1] = '\0';
416         ss = realloc(ss, (ss - 1) * sizeof(*ss));
417         if (ss == NULL)
418             break;
419         ss[ss - 1] = '\0';
420         ss[ss] = line;
421         ss[ss + 1] = '\0';
422
423         if (ss[0] != '\0') {
424             if (ss[0] == ' ')
425                 continue;
426
427             if (ss[0] == '#')
428                 continue;
429
430             if (ss[0] == '!' || ss[0] == '>')
431                 continue;
432
433             if (ss[0] == '!' || ss[0] == '>')
434                 continue;
435
436             if (ss[0] == '!' || ss[0] == '>')
437                 continue;
438
439             if (ss[0] == '!' || ss[0] == '>')
440                 continue;
441
442             if (ss[0] == '!' || ss[0] == '>')
443                 continue;
444
445             if (ss[0] == '!' || ss[0] == '>')
446                 continue;
447
448             if (ss[0] == '!' || ss[0] == '>')
449                 continue;
450
451             if (ss[0] == '!' || ss[0] == '>')
452                 continue;
453
454             if (ss[0] == '!' || ss[0] == '>')
455                 continue;
456
457             if (ss[0] == '!' || ss[0] == '>')
458                 continue;
459
460             if (ss[0] == '!' || ss[0] == '>')
461                 continue;
462
463             if (ss[0] == '!' || ss[0] == '>')
464                 continue;
465
466             if (ss[0] == '!' || ss[0] == '>')
467                 continue;
468
469             if (ss[0] == '!' || ss[0] == '>')
470                 continue;
471
472             if (ss[0] == '!' || ss[0] == '>')
473                 continue;
474
475             if (ss[0] == '!' || ss[0] == '>')
476                 continue;
477
478             if (ss[0] == '!' || ss[0] == '>')
479                 continue;
480
481             if (ss[0] == '!' || ss[0] == '>')
482                 continue;
483
484             if (ss[0] == '!' || ss[0] == '>')
485                 continue;
486
487             if (ss[0] == '!' || ss[0] == '>')
488                 continue;
489
490             if (ss[0] == '!' || ss[0] == '>')
491                 continue;
492
493             if (ss[0] == '!' || ss[0] == '>')
494                 continue;
495
496             if (ss[0] == '!' || ss[0] == '>')
497                 continue;
498
499             if (ss[0] == '!' || ss[0] == '>')
500                 continue;
501
502             if (ss[0] == '!' || ss[0] == '>')
503                 continue;
504
505             if (ss[0] == '!' || ss[0] == '>')
506                 continue;
507
508             if (ss[0] == '!' || ss[0] == '>')
509                 continue;
510
511             if (ss[0] == '!' || ss[0] == '>')
512                 continue;
513
514             if (ss[0] == '!' || ss[0] == '>')
515                 continue;
516
517             if (ss[0] == '!' || ss[0] == '>')
518                 continue;
519
520             if (ss[0] == '!' || ss[0] == '>')
521                 continue;
522
523             if (ss[0] == '!' || ss[0] == '>')
524                 continue;
525
526             if (ss[0] == '!' || ss[0] == '>')
527                 continue;
528
529             if (ss[0] == '!' || ss[0] == '>')
530                 continue;
531
532             if (ss[0] == '!' || ss[0] == '>')
533                 continue;
534
535             if (ss[0] == '!' || ss[0] == '>')
536                 continue;
537
538             if (ss[0] == '!' || ss[0] == '>')
539                 continue;
540
541             if (ss[0] == '!' || ss[0] == '>')
542                 continue;
543
544             if (ss[0] == '!' || ss[0] == '>')
545                 continue;
546
547             if (ss[0] == '!' || ss[0] == '>')
548                 continue;
549
550             if (ss[0] == '!' || ss[0] == '>')
551                 continue;
552
553             if (ss[0] == '!' || ss[0] == '>')
554                 continue;
555
556             if (ss[0] == '!' || ss[0] == '>')
557                 continue;
558
559             if (ss[0] == '!' || ss[0] == '>')
560                 continue;
561
562             if (ss[0] == '!' || ss[0] == '>')
563                 continue;
564
565             if (ss[0] == '!' || ss[0] == '>')
566                 continue;
567
568             if (ss[0] == '!' || ss[0] == '>')
569                 continue;
570
571             if (ss[0] == '!' || ss[0] == '>')
572                 continue;
573
574             if (ss[0] == '!' || ss[0] == '>')
575                 continue;
576
577             if (ss[0] == '!' || ss[0] == '>')
578                 continue;
579
580             if (ss[0] == '!' || ss[0] == '>')
581                 continue;
582
583             if (ss[0] == '!' || ss[0] == '>')
584                 continue;
585
586             if (ss[0] == '!' || ss[0] == '>')
587                 continue;
588
589             if (ss[0] == '!' || ss[0] == '>')
590                 continue;
591
592             if (ss[0] == '!' || ss[0] == '>')
593                 continue;
594
595             if (ss[0] == '!' || ss[0] == '>')
596                 continue;
597
598             if (ss[0] == '!' || ss[0] == '>')
599                 continue;
599
600             if (ss[0] == '!' || ss[0] == '>')
601                 continue;
602
603             if (ss[0] == '!' || ss[0] == '>')
604                 continue;
605
606             if (ss[0] == '!' || ss[0] == '>')
607                 continue;
608
609             if (ss[0] == '!' || ss[0] == '>')
610                 continue;
611
612             if (ss[0] == '!' || ss[0] == '>')
613                 continue;
614
615             if (ss[0] == '!' || ss[0] == '>')
616                 continue;
617
618             if (ss[0] == '!' || ss[0] == '>')
619                 continue;
620
621             if (ss[0] == '!' || ss[0] == '>')
622                 continue;
623
624             if (ss[0] == '!' || ss[0] == '>')
625                 continue;
626
627             if (ss[0] == '!' || ss[0] == '>')
628                 continue;
629
630             if (ss[0] == '!' || ss[0] == '>')
631                 continue;
632
633             if (ss[0] == '!' || ss[0] == '>')
634                 continue;
635
636             if (ss[0] == '!' || ss[0] == '>')
637                 continue;
638
639             if (ss[0] == '!' || ss[0] == '>')
640                 continue;
641
642             if (ss[0] == '!' || ss[0] == '>')
643                 continue;
644
645             if (ss[0] == '!' || ss[0] == '>')
646                 continue;
647
648             if (ss[0] == '!' || ss[0] == '>')
649                 continue;
650
651             if (ss[0] == '!' || ss[0] == '>')
652                 continue;
653
654             if (ss[0] == '!' || ss[0] == '>')
655                 continue;
656
657             if (ss[0] == '!' || ss[0] == '>')
658                 continue;
659
660             if (ss[0] == '!' || ss[0] == '>')
661                 continue;
662
663             if (ss[0] == '!' || ss[0] == '>')
664                 continue;
665
666             if (ss[0] == '!' || ss[0] == '>')
667                 continue;
668
669             if (ss[0] == '!' || ss[0] == '>')
670                 continue;
671
672             if (ss[0] == '!' || ss[0] == '>')
673                 continue;
674
675             if (ss[0] == '!' || ss[0] == '>')
676                 continue;
677
678             if (ss[0] == '!' || ss[0] == '>')
679                 continue;
680
681             if (ss[0] == '!' || ss[0] == '>')
682                 continue;
683
684             if (ss[0] == '!' || ss[0] == '>')
685                 continue;
686
687             if (ss[0] == '!' || ss[0] == '>')
688                 continue;
689
690             if (ss[0] == '!' || ss[0] == '>')
691                 continue;
692
693             if (ss[0] == '!' || ss[0] == '>')
694                 continue;
695
696             if (ss[0] == '!' || ss[0] == '>')
697                 continue;
698
699             if (ss[0] == '!' || ss[0] == '>')
700                 continue;
701
702             if (ss[0] == '!' || ss[0] == '>')
703                 continue;
704
705             if (ss[0] == '!' || ss[0] == '>')
706                 continue;
707
708             if (ss[0] == '!' || ss[0] == '>')
709                 continue;
710
711             if (fp = fopen(whatpath, "r")) == NULL)
712                 perror(whatpath);
713
714             if (fgets(s, MAXNAMELEN, fp) == NULL)
715                 break;
716
717             if (ss[0] == '!' || ss[0] == '>')
718                 continue;
719
720             if (ss[0] == '!' || ss[0] == '>')
721                 continue;
722
723             if (ss[0] == '!' || ss[0] == '>')
724                 continue;
725
726             if (ss[0] == '!' || ss[0] == '>')
727                 continue;
728
729             if (ss[0] == '!' || ss[0] == '>')
730                 continue;
731
732             if (ss[0] == '!' || ss[0] == '>')
733                 continue;
734
735             if (ss[0] == '!' || ss[0] == '>')
736                 continue;
737
738             if (ss[0] == '!' || ss[0] == '>')
739                 continue;
740
741             if (ss[0] == '!' || ss[0] == '>')
742                 continue;
743
744             if (ss[0] == '!' || ss[0] == '>')
745                 continue;
746
747             if (ss[0] == '!' || ss[0] == '>')
748                 continue;
749
750             if (ss[0] == '!' || ss[0] == '>')
751                 continue;
752
753             if (ss[0] == '!' || ss[0] == '>')
754                 continue;
755
756             if (ss[0] == '!' || ss[0] == '>')
757                 continue;
758
759             if (ss[0] == '!' || ss[0] == '>')
760                 continue;
761
762             if (ss[0] == '!' || ss[0] == '>')
763                 continue;
764
765             if (ss[0] == '!' || ss[0] == '>')
766                 continue;
767
768             if (ss[0] == '!' || ss[0] == '>')
769                 continue;
770
771             if (ss[0] == '!' || ss[0] == '>')
772                 continue;
773
774             if (ss[0] == '!' || ss[0] == '>')
775                 continue;
776
777             if (ss[0] == '!' || ss[0] == '>')
778                 continue;
779
780             if (ss[0] == '!' || ss[0] == '>')
781                 continue;
782
783             if (ss[0] == '!' || ss[0] == '>')
784                 continue;
785
786             if (ss[0] == '!' || ss[0] == '>')
787                 continue;
788
789             if (ss[0] == '!' || ss[0] == '>')
790                 continue;
791
792             if (ss[0] == '!' || ss[0] == '>')
793                 continue;
794
795             if (ss[0] == '!' || ss[0] == '>')
796                 continue;
797
798             if (ss[0] == '!' || ss[0] == '>')
799                 continue;
800
801             if (ss[0] == '!' || ss[0] == '>')
802                 continue;
803
804             if (ss[0] == '!' || ss[0] == '>')
805                 continue;
806
807             if (ss[0] == '!' || ss[0] == '>')
808                 continue;
809
810             if (ss[0] == '!' || ss[0] == '>')
811                 continue;
812
813             if (ss[0] == '!' || ss[0] == '>')
814                 continue;
815
816             if (ss[0] == '!' || ss[0] == '>')
817                 continue;
818
819             if (ss[0] == '!' || ss[0] == '>')
820                 continue;
821
822             if (ss[0] == '!' || ss[0] == '>')
823                 continue;
824
825             if (ss[0] == '!' || ss[0] == '>')
826                 continue;
827
828             if (ss[0] == '!' || ss[0] == '>')
829                 continue;
830
831             if (ss[0] == '!' || ss[0] == '>')
832                 continue;
833
834             if (ss[0] == '!' || ss[0] == '>')
835                 continue;
836
837             if (ss[0] == '!' || ss[0] == '>')
838                 continue;
839
840             if (ss[0] == '!' || ss[0] == '>')
841                 continue;
842
843             if (ss[0] == '!' || ss[0] == '>')
844                 continue;
845
846             if (ss[0] == '!' || ss[0] == '>')
847                 continue;
848
849             if (ss[0] == '!' || ss[0] == '>')
850                 continue;
851
852             if (ss[0] == '!' || ss[0] == '>')
853                 continue;
854
855             if (ss[0] == '!' || ss[0] == '>')
856                 continue;
857
858             if (ss[0] == '!' || ss[0] == '>')
859                 continue;
860
861             if (ss[0] == '!' || ss[0] == '>')
862                 continue;
863
864             if (ss[0] == '!' || ss[0] == '>')
865                 continue;
866
867             if (ss[0] == '!' || ss[0] == '>')
868                 continue;
869
870             if (ss[0] == '!' || ss[0] == '>')
871                 continue;
872
873             if (ss[0] == '!' || ss[0] == '>')
874                 continue;
875
876             if (ss[0] == '!' || ss[0] == '>')
877                 continue;
878
879             if (ss[0] == '!' || ss[0] == '>')
880                 continue;
881
882             if (ss[0] == '!' || ss[0] == '>')
883                 continue;
884
885             if (ss[0] == '!' || ss[0] == '>')
886                 continue;
887
888             if (ss[0] == '!' || ss[0] == '>')
889                 continue;
890
891             if (ss[0] == '!' || ss[0] == '>')
892                 continue;
893
894             if (ss[0] == '!' || ss[0] == '>')
895                 continue;
896
897             if (ss[0] == '!' || ss[0] == '>')
898                 continue;
899
900             if (ss[0] == '!' || ss[0] == '>')
901                 continue;
902
903             if (ss[0] == '!' || ss[0] == '>')
904                 continue;
905
906             if (ss[0] == '!' || ss[0] == '>')
907                 continue;
908
909             if (ss[0] == '!' || ss[0] == '>')
910                 continue;
911
912             if (ss[0] == '!' || ss[0] == '>')
913                 continue;
914
915             if (ss[0] == '!' || ss[0] == '>')
916                 continue;
917
918             if (ss[0] == '!' || ss[0] == '>')
919                 continue;
920
921             if (ss[0] == '!' || ss[0] == '>')
922                 continue;
923
924             if (ss[0] == '!' || ss[0] == '>')
925                 continue;
926
927             if (ss[0] == '!' || ss[0] == '>')
928                 continue;
929
930             if (ss[0] == '!' || ss[0] == '>')
931                 continue;
932
933             if (ss[0] == '!' || ss[0] == '>')
934                 continue;
935
936             if (ss[0] == '!' || ss[0] == '>')
937                 continue;
938
939             if (ss[0] == '!' || ss[0] == '>')
940                 continue;
941
942             if (ss[0] == '!' || ss[0] == '>')
943                 continue;
944
945             if (ss[0] == '!' || ss[0] == '>')
946                 continue;
947
948             if (ss[0] == '!' || ss[0] == '>')
949                 continue;
950
951             if (ss[0] == '!' || ss[0] == '>')
952                 continue;
953
954             if (ss[0] == '!' || ss[0] == '>')
955                 continue;
956
957             if (ss[0] == '!' || ss[0] == '>')
958                 continue;
959
960             if (ss[0] == '!' || ss[0] == '>')
961                 continue;
962
963             if (ss[0] == '!' || ss[0] == '>')
964                 continue;
965
966             if (ss[0] == '!' || ss[0] == '>')
967                 continue;
968
969             if (ss[0] == '!' || ss[0] == '>')
970                 continue;
971
972             if (ss[0] == '!' || ss[0] == '>')
973                 continue;
974
975             if (ss[0] == '!' || ss[0] == '>')
976                 continue;
977
978             if (ss[0] == '!' || ss[0] == '>')
979                 continue;
980
981             if (ss[0] == '!' || ss[0] == '>')
982                 continue;
983
984             if (ss[0] == '!' || ss[0] == '>')
985                 continue;
986
987             if (ss[0] == '!' || ss[0] == '>')
988                 continue;
989
990             if (ss[0] == '!' || ss[0] == '>')
991                 continue;
992
993             if (ss[0] == '!' || ss[0] == '>')
994                 continue;
995
996             if (ss[0] == '!' || ss[0] == '>')
997                 continue;
998
999             if (ss[0] == '!' || ss[0] == '>')
1000                continue;
1001
1002             if (ss[0] == '!' || ss[0] == '>')
1003                continue;
1004
1005             if (ss[0] == '!' || ss[0] == '>')
1006                continue;
1007
1008             if (ss[0] == '!' || ss[0] == '>')
1009                continue;
1010
1011             if (ss[0] == '!' || ss[0] == '>')
1012                continue;
1013
1014             if (ss[0] == '!' || ss[0] == '>')
1015                continue;
1016
1017             if (ss[0] == '!' || ss[0] == '>')
1018                continue;
1019
1020             if (ss[0] == '!' || ss[0] == '>')
1021                continue;
1022
1023             if (ss[0] == '!' || ss[0] == '>')
1024                continue;
1025
1026             if (ss[0] == '!' || ss[0] == '>')
1027                continue;
1028
1029             if (ss[0] == '!' || ss[0] == '>')
1030                continue;
1031
1032             if (ss[0] == '!' || ss[0] == '>')
1033                continue;
1034
1035             if (ss[0] == '!' || ss[0] == '>')
1036                continue;
1037
1038             if (ss[0] == '!' || ss[0] == '>')
1039                continue;
1040
1041             if (ss[0] == '!' || ss[0] == '>')
1042                continue;
1043
1044             if (ss[0] == '!' || ss[0] == '>')
1045                continue;
1046
1047             if (ss[0] == '!' || ss[0] == '>')
1048                continue;
1049
1050             if (ss[0] == '!' || ss[0] == '>')
1051                continue;
1052
1053             if (ss[0] == '!' || ss[0] == '>')
1054                continue;
1055
1056             if (ss[0] == '!' || ss[0] == '>')
1057                continue;
1058
1059             if (ss[0] == '!' || ss[0] == '>')
1060                continue;
1061
1062             if (ss[0] == '!' || ss[0] == '>')
1063                continue;
1064
1065             if (ss[0] == '!' || ss[0] == '>')
1066                continue;
1067
1068             if (ss[0] == '!' || ss[0] == '>')
1069                continue;
1070
1071             if (ss[0] == '!' || ss[0] == '>')
1072                continue;
1073
1074             if (ss[0] == '!' || ss[0] == '>')
1075                continue;
1076
1077             if (ss[0] == '!' || ss[0] == '>')
1078                continue;
1079
1080             if (ss[0] == '!' || ss[0] == '>')
1081                continue;
1082
1083             if (ss[0] == '!' || ss[0] == '>')
1084                continue;
1085
1086             if (ss[0] == '!' || ss[0] == '>')
1087                continue;
1088
1089             if (ss[0] == '!' || ss[0] == '>')
1090                continue;
1091
1092             if (ss[0] == '!' || ss[0] == '>')
1093                continue;
1094
1095             if (ss[0] == '!' || ss[0] == '>')
1096                continue;
1097
1098             if (ss[0] == '!' || ss[0] == '>')
1099                continue;
1100
1101             if (ss[0] == '!' || ss[0] == '>')
1102                continue;
1103
1104             if (ss[0] == '!' || ss[0] == '>')
1105                continue;
1106
1107             if (ss[0] == '!' || ss[0] == '>')
1108                continue;
1109
1110             if (ss[0] == '!' || ss[0] == '>')
1111                continue;
1112
1113             if (ss[0] == '!' || ss[0] == '>')
1114                continue;
1115
1116             if (ss[0] == '!' || ss[0] == '>')
1117                continue;
1118
1119             if (ss[0] == '!' || ss[0] == '>')
1120                continue;
1121
1122             if (ss[0] == '!' || ss[0] == '>')
1123                continue;
1124
1125             if (ss[0] == '!' || ss[0] == '>')
1126                continue;
1127
1128             if (ss[0] == '!' || ss[0] == '>')
1129                continue;
1130
1131             if (ss[0] == '!' || ss[0] == '>')
1132                continue;
1133
1134             if (ss[0] == '!' || ss[0] == '>')
1135                continue;
1136
1137             if (ss[0] == '!' || ss[0] == '>')
1138                continue;
1139
1140             if (ss[0] == '!' || ss[0] == '>')
1141                continue;
1142
1143             if (ss[0] == '!' || ss[0] == '>')
1144                continue;
1145
1146             if (ss[0] == '!' || ss[0] == '>')
1147                continue;
1148
1149             if (ss[0] == '!' || ss[0] == '>')
1150                continue;
1151
1152             if (ss[0] == '!' || ss[0] == '>')
1153                continue;
1154
1155             if (ss[0] == '!' || ss[0] == '>')
1156                continue;
1157
1158             if (ss[0] == '!' || ss[0] == '>')
1159                continue;
1160
1161             if (ss[0] == '!' || ss[0] == '>')
1162                continue;
1163
1164             if (ss[0] == '!' || ss[0] == '>')
1165                continue;
1166
1167             if (ss[0] == '!' || ss[0] == '>')
1168                continue;
1169
1170             if (ss[0] == '!' || ss[0] == '>')
1171                continue;
1172
1173             if (ss[0] == '!' || ss[0] == '>')
1174                continue;
1175
1176             if (ss[0] == '!' || ss[0] == '>')
1177                continue;
1178
1179             if (ss[0] == '!' || ss[0] == '>')
1180                continue;
1181
1182             if (ss[0] == '!' || ss[0] == '>')
1183                continue;
1184
1185             if (ss[0] == '!' || ss[0] == '>')
1186                continue;
1187
1188             if (ss[0] == '!' || ss[0] == '>')
1189                continue;
1190
1191             if (ss[0] == '!' || ss[0] == '>')
1192                continue;
1193
1194             if (ss[0] == '!' || ss[0] == '>')
1195                continue;
1196
1197             if (ss[0] == '!' || ss[0] == '>')
1198                continue;
1199
1200             if (ss[0] == '!' || ss[0] == '>')
1201                continue;
1202
1203             if (ss[0] == '!' || ss[0] == '>')
1204                continue;
1205
1206             if (ss[0] == '!' || ss[0] == '>')
1207                continue;
1208
1209             if (ss[0] == '!' || ss[0] == '>')
1210                continue;
1211
1212             if (ss[0] == '!' || ss[0] == '>')
1213                continue;
1214
1215             if (ss[0] == '!' || ss[0] == '>')
1216                continue;
1217
1218             if (ss[0] == '!' || ss[0] == '>')
1219                continue;
1220
1221             if (ss[0] == '!' || ss[0] == '>')
1222                continue;
1223
1224             if (ss[0] == '!' || ss[0] == '>')
1225                continue;
1226
1227             if (ss[0] == '!' || ss[0] == '>')
1228                continue;
1229
1230             if (ss[0] == '!' || ss[0] == '>')
1231                continue;
1232
1233             if (ss[0] == '!' || ss[0] == '>')
1234                continue;
1235
1236             if (ss[0] == '!' || ss[0] == '>')
1237                continue;
1238
1239             if (ss[0] == '!' || ss[0] == '>')
1240                continue;
1241
1242             if (ss[0] == '!' || ss[0] == '>')
1243                continue;
1244
1245             if (ss[0] == '!' || ss[0] == '>')
1246                continue;
1247
1248             if (ss[0] == '!' || ss[0] == '>')
1249                continue;
1250
1251             if (ss[0] == '!' || ss[0] == '>')
1252                continue;
1253
1254             if (ss[0] == '!' || ss[0] == '>')
1255                continue;
1256
1257             if (ss[0] == '!' || ss[0] == '>')
1258                continue;
1259
1260             if (ss[0] == '!' || ss[0] == '>')
1261                continue;
1262
1263             if (ss[0] == '!' || ss[0] == '>')
1264                continue;
1265
1266             if (ss[0] == '!' || ss[0] == '>')
1267                continue;
1268
1269             if (ss[0] == '!' || ss[0] == '>')
1270                continue;
1271
1272             if (ss[0] == '!' || ss[0] == '>')
1273                continue;
1274
1275             if (ss[0] == '!' || ss[0] == '>')
1276                continue;
1277
1278             if (ss[0] == '!' || ss[0] == '>')
1279                continue;
1280
1281             if (ss[0] == '!' || ss[0
```

```

693     if ((fp = fopen(whatpath, "r")) == NULL)
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) sprintf(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 }
_____unchanged_portion_omitted_____
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_makewhat(is(void)
1599 {
1600     (void) fprintf(stderr, gettext("usage: makewhat(is\n"));

```

```

1602     exit(1);
1603 }
_____unchanged_portion_omitted_____

```

```
*****
13202 Tue Jul 15 19:41:38 2014
new/usr/src/man/man1/Makefile
Add catman, makewhatis functionality. Print an error if the whatis database
is missing.
*****
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
9 # at http://www.illumos.org/license/CDDL.
10 #

12 #
13 # Copyright 2011, Richard Lowe
14 # Copyright 2013 Nexenta Systems, Inc. All rights reserved.
15 # Copyright 2014 Garrett D'Amore <garrett@damore.org>
16 #

18 include      $(SRC)/Makefile.master

20 MANSECT=    1

22 MANFILES=    acctcom.1          \
23                 adb.1           \
24                 addbib.1        \
25                 alias.1         \
26                 allocate.1      \
27                 amt.1          \
28                 appcert.1       \
29                 apptrace.1      \
30                 apropos.1       \
31                 ar.1            \
32                 arch.1          \
33                 asa.1            \
34                 at.1             \
35                 atq.1            \
36                 atrm.1          \
37                 audioclient.1     \
38                 audioplay.1      \
39                 audiorecord.1    \
40                 audiotest.1      \
41                 auths.1          \
42                 awk.1            \
43                 banner.1         \
44                 basename.1       \
45                 bc.1              \
46                 bdiff.1          \
47                 bfs.1            \
48                 break.1          \
49                 builtin.1        \
50                 cal.1            \
51                 calendar.1      \
52                 cancel.1         \
53                 cat.1            \
54                 catman.1          \
55                 cd.1              \
56                 cdrw.1            \
57                 checknr.1        \
58                 chgrp.1          \
59                 chkey.1          \
60
```

```
61                 chmod.1          \
62                 chown.1          \
63                 ckdate.1         \
64                 ckgid.1          \
65                 ckint.1          \
66                 ckitem.1         \
67                 ckkeywd.1        \
68                 ckpath.1         \
69                 ckrange.1        \
70                 ckstr.1          \
71                 cksum.1          \
72                 cktime.1         \
73                 ckuid.1          \
74                 ckyorn.1         \
75                 clear.1          \
76                 cmp.1            \
77                 col.1            \
78                 comm.1           \
79                 command.1       \
80                 compress.1      \
81                 cp.1              \
82                 cpio.1            \
83                 cputrack.1       \
84                 crle.1            \
85                 crontab.1         \
86                 crypt.1          \
87                 csh.1              \
88                 csplit.1          \
89                 ctags.1           \
90                 ctrun.1           \
91                 ctstat.1          \
92                 ctwatch.1         \
93                 cut.1              \
94                 date.1            \
95                 dc.1              \
96                 deallocate.1      \
97                 deroff.1          \
98                 dhcpinfo.1       \
99                 diff.1             \
100                diff3.1           \
101                diffmk.1          \
102                digest.1          \
103                dircmp.1          \
104                dis.1              \
105                disown.1          \
106                dispgid.1         \
107                dispuid.1         \
108                dos2unix.1         \
109                download.1        \
110                dpost.1           \
111                du.1              \
112                dump.1            \
113                dumpcs.1          \
114                echo.1             \
115                ed.1              \
116                egrep.1            \
117                eject.1           \
118                elfdump.1         \
119                elfedit.1         \
120                elfsign.1         \
121                elfwrap.1          \
122                enable.1           \
123                encrypt.1          \
124                enhance.1         \
125                env.1              \
126                eqn.1
```

```

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

```

```

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

```

```

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

```

```

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

```

```

391      tput.1
392      tr.1
393      trap.1
394      troff.1
395      true.1
396      truss.1
397      tsort.1
398      tty.1
399      type.1
400      typeset.1
401      ul.1
402      umask.1
403      uname.1
404      undef.1
405      uniq.1
406      units.1
407      unix2dos.1
408      uptime.1
409      vacation.1
410      vgrind.1
411      volcheck.1
412      volrmmount.1
413      w.1
414      wait.1
415      wc.1
416      which.1
417      who.1
418      whocalls.1
419      whois.1
420      write.1
421      xargs.1
422      xgettext.1
423      xstr.1
424      yacc.1
425      yes.1
426      ypcat.1
427      ypmatch.1
428      yppasswd.1
429      ypwhich.1
430      zlogin.1
431      zonename.1
433 MANLINKS=
434      batch.1
435      bg.1
436      case.1
437      chdir.1
438      checkeq.1
439      continue.1
440      decrypt.1
441      dirname.1
442      dirs.1
443      disable.1
444      dumpkeys.1
445      edit.1
446      errange.1
447      errdate.1
448      errgid.1
449      errint.1
450      erritem.1
451      errpath.1
452      errstr.1
453      errtime.1
454      erruid.1
455      erryorn.1
456      eval.1
        export.1

```

```

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

```

```

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

565 intro.1        := LINKSRC = Intro.1
567 whatis.1        := LINKSRC = apropos.1
569 unalias.1       := LINKSRC = alias.1
571 batch.1         := LINKSRC = at.1
573 dirname.1       := LINKSRC = basename.1
575 continue.1      := LINKSRC = break.1
577 chdir.1         := LINKSRC = cd.1
578 dirs.1          := LINKSRC = cd.1
579 popd.1          := LINKSRC = cd.1
580 pushd.1         := LINKSRC = cd.1
582 errdate.1       := LINKSRC = ckdate.1
583 helpdate.1      := LINKSRC = ckdate.1
584 validate.1      := LINKSRC = ckdate.1
586 errgid.1         := LINKSRC = ckgid.1
587 helpgid.1        := LINKSRC = ckgid.1
588 valgid.1         := LINKSRC = ckgid.1

```

```

590 errint.1        := LINKSRC = ckint.1
591 helpint.1        := LINKSRC = ckint.1
592 valint.1         := LINKSRC = ckint.1
594 erritem.1        := LINKSRC = ckitem.1
595 helpitem.1        := LINKSRC = ckitem.1
597 errpath.1        := LINKSRC = ckpath.1
598 helppath.1        := LINKSRC = ckpath.1
599 valpath.1         := LINKSRC = ckpath.1
601 errange.1        := LINKSRC = ckrange.1
602 helprange.1       := LINKSRC = ckrange.1
603 valrange.1        := LINKSRC = ckrange.1
605 errstr.1          := LINKSRC = ckstr.1
606 helpstr.1         := LINKSRC = ckstr.1
607 valstr.1          := LINKSRC = ckstr.1
609 errtime.1         := LINKSRC = cktime.1
610 helptime.1        := LINKSRC = cktime.1
611 valtime.1          := LINKSRC = cktime.1
613 erruid.1          := LINKSRC = ckuid.1
614 helpuid.1         := LINKSRC = ckuid.1
615 valuid.1          := LINKSRC = ckuid.1
617 erryorn.1         := LINKSRC = ckyorn.1
618 helpyorn.1        := LINKSRC = ckyorn.1
619 valyorn.1          := LINKSRC = ckyorn.1
621 uncompress.1       := LINKSRC = compress.1
622 zcat.1              := LINKSRC = compress.1
624 red.1              := LINKSRC = ed.1
626 disable.1         := LINKSRC = enable.1
628 decrypt.1         := LINKSRC = encrypt.1
630 checkeq.1         := LINKSRC = egn.1
631 neqn.1              := LINKSRC = egn.1
633 eval.1              := LINKSRC = exec.1
634 source.1            := LINKSRC = exec.1
636 goto.1              := LINKSRC = exit.1
637 return.1            := LINKSRC = exit.1
639 unexpand.1          := LINKSRC = expand.1
641 hashstat.1         := LINKSRC = hash.1
642 rehash.1             := LINKSRC = hash.1
643 unhash.1             := LINKSRC = hash.1
645 fc.1                := LINKSRC = history.1
646 hist.1              := LINKSRC = history.1
648 bg.1                := LINKSRC = jobs.1
649 fg.1                := LINKSRC = jobs.1
650 notify.1             := LINKSRC = jobs.1
651 stop.1              := LINKSRC = jobs.1
653 jsh.1                := LINKSRC = ksh93.1
654 ksh.1                := LINKSRC = ksh93.1

```

```

655 rksh.1          := LINKSRC = ksh93.1
656 rksh93.1        := LINKSRC = ksh93.1
657 sh.1            := LINKSRC = ksh93.1

659 ldapadd.1       := LINKSRC = ldapmodify.1

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

664 dumpkeys.1      := LINKSRC = loadkeys.1

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

681 rmail.1          := LINKSRC = mail.1

683 page.1           := LINKSRC = more.1

685 snca.1           := LINKSRC = nca.1

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

690 pfcs.1           := LINKSRC = pfexec.1
691 pfksh.1           := LINKSRC = pfexec.1
692 pfsh.1            := LINKSRC = pfexec.1

694 pkill.1          := LINKSRC = pgrep.1

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

708 rmdir.1           := LINKSRC = rm.1

710 rmmount.1         := LINKSRC = rmmount.1

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

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

720 case.1             := LINKSRC = shell_builtin.1

```

```

721 for.1            := LINKSRC = shell_builtin.1
722 foreach.1          := LINKSRC = shell_builtin.1
723 function.1         := LINKSRC = shell_builtin.1
724 if.1               := LINKSRC = shell_builtin.1
725 repeat.1           := LINKSRC = shell_builtin.1
726 select.1           := LINKSRC = shell_builtin.1
727 switch.1            := LINKSRC = shell_builtin.1
728 until.1             := LINKSRC = shell_builtin.1
729 while.1             := LINKSRC = shell_builtin.1

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

735 strconf.1           := LINKSRC = strchg.1
737 settime.1            := LINKSRC = touch.1
739 onintr.1             := LINKSRC = trap.1
741 false.1              := LINKSRC = true.1
743 whence.1             := LINKSRC = typeset.1
745 # Links to usr/has/man

747 edit.1              := LINKSRC = ../../../../has/man/man1has/edit.1has
749 vedit.1              := LINKSRC = ../../../../has/man/man1has/vi.1has
751 .KEEP_STATE:
753 include              $(SRC)/man/Makefile.man
755 install:              $(ROOTMANFILES) $(ROOTMANLINKS)

```

```
new/usr/src/man/man1/catman.1
```

```
*****
1525 Tue Jul 15 19:41:38 2014
new/usr/src/man/man1/catman.1
Add catman, makewhatis functionality. Print an error if the whatis database
is missing.
*****
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 .\""
11 .\""
12 .\" Copyright 2014 Garrett D'Amore <garrett@damore.org>
13 .\""
14 .Dd Jul 13, 2014
15 .Dt CATAMN 1
16 .Os
17 .Sh NAME
18 .Nm catman
19 .Nd generate
20 .Nm whatis
21 database files
22 .Sh SYNOPSIS
23 .Nm
24 .Op Fl M Ar path
25 .Sh DESCRIPTION
26 The
27 .Nm
28 utility generates a set of
29 .Nm whatis
30 database files suitable for use with
31 .Xr apropos 1
32 and
33 .Xr whatis 1 .
34 It is supplied for compatibility reasons. The same databases can
35 be generated using the
36 .Fl w
37 option with
38 .Xr man 1 ,
39 and that command should be used instead.
40 .Bl -tag -width ".Fl d"
41 .It Fl M Ar path
42 Generate the
43 .Nm whatis
44 database files within the specified colon separated manual paths.
45 Overrides the
46 .Ev MANPATH
47 environment variable.
48 .El
49 .Sh ENVIRONMENT
50 The following environment variables affect the execution of
51 .Nm :
52 .Bl -tag -width ".Ev MANPATH"
53 .It Ev MANPATH
54 Used to specify a colon seperated list of manual paths within
55 which to generate
56 .Nm whatis
57 database files.
58 .El
59 .Sh DIAGNOSTICS
60 The
```

```
1
```

```
new/usr/src/man/man1/catman.1
```

```
61 .Nm
62 utility exits 0 on success, and non-zero otherwise.
63 .Sh INTERFACE STABILITY
64 Obsolete.
65 .Sh CODE SET INDEPENDENCE
66 Enabled.
67 .Sh SEE ALSO
68 .Xr apropos 1 ,
69 .Xr man 1 ,
70 .Xr whatis 1
```

```
2
```

new/usr/src/pkg/manifests/system-man.mf

```
*****
2493 Tue Jul 15 19:41:38 2014
new/usr/src/pkg/manifests/system-man.mf
Add catman, makewhatis functionality. Print an error if the whatis database
is missing.
*****
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 set name(pkg) fmri value=pkg:/system/man@$(_PKGVERS)
18 set name(pkg).description \
19     value="utilities for display and formatting of reference manual pages"
20 set name(pkg).summary value="Reference Manual Pages Tools"
21 set name(info.classification \
22     value="org.opensolaris.category.2008:System/Text Tools"
23 set name(variant.arch value=$(ARCH)
24 dir path=usr/bin
25 dir path=usr/lib
26 dir path=usr/share
27 dir path=usr/share/man
28 dir path=usr/share/man/man1
29 dir path=usr/share/man/man5
30 file path=usr/bin/catman mode=0555
31 file path=usr/bin/man mode=0555
32 file path=usr/bin/mandoc mode=0555
33 file path=usr/lib/makewhatis mode=0555
34 file path=usr/lib/mandoc_preconv mode=0555
35 file path=usr/share/man/man1/apropos.1
36 file path=usr/share/man/man1/catman.1
37 file path=usr/share/man/man1/man.1
38 file path=usr/share/man/man1/mandoc.1
39 file path=usr/share/man/man5/egn.5
40 file path=usr/share/man/man5/man.5
41 file path=usr/share/man/man5/mandoc_char.5
42 file path=usr/share/man/man5/mandoc_roff.5
43 file path=usr/share/man/man5/mdoc.5
44 file path=usr/share/man/man5/tbl.5
45 hardlink path=usr/bin/apropos target=../../usr/bin/man
46 hardlink path=usr/bin/whatis target=../../usr/bin/man
47 license lic_CDDL license=lic_CDDL
48 license usr/src/cmd/man/THIRDPARTYLICENSE \
49     license=usr/src/cmd/man/THIRDPARTYLICENSE
50 license usr/src/cmd/mandoc/THIRDPARTYLICENSE \
51     license=usr/src/cmd/mandoc/THIRDPARTYLICENSE
52 link path=usr/man target=../share/man
53 link path=usr/share/man/man1/whatis.1 target=apropos.1
54 # arguably we also need lp, for man -t support, but really we don't
55 # want to make this mandatory, so we don't express teh dependency here,
56 # gzcat/bzcat are used for displaying compressed manpages. However,
57 # as we don't format such pages this way by default, lets leave the
58 # dependency out.
59 #depend fmri=compress/bzip2 type=require
60 #depend fmri=compress/gzip type=require
```

1

new/usr/src/pkg/manifests/system-man.mf

```
61 # less is the default (per user environment) pager. We really should just
62 # import this into illumos-gate.
63 depend fmri=text/less type=require
```

2