```
**********************************************************
   12863 Fri May  9 12:12:12 2014
new/usr/src/cmd/printf/printf.c
4854 printf(1) doesn’t support %b and \c properly
4818 printf(1) should support n$ width and precision specifiers
**********************************************************
/*
 * Copyright 2014 Garrett D’Amore <garrett@damore.org>
 * Copyright 2010 Nexenta Systems, Inc.  All rights reserved.
 * Copyright (c) 1989, 1993
 *      The Regents of the University of California.  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. Neither the name of the University nor the names of its contributors
 *    may be used to endorse or promote products derived from this software
 *    without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ‘‘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 REGENTS OR CONTRIBUTORS 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.
 */

#include <sys/types.h>

#include <err.h>
#include <errno.h>
#include <inttypes.h>
#include <limits.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <alloca.h>
#include <ctype.h>
#include <locale.h>
#include <note.h>

#define warnx1(a, b, c)         warnx(a)
#define warnx2(a, b, c)         warnx(a, b)
#define warnx3(a, b, c)         warnx(a, b, c)

#define PTRDIFF(x, y)   ((uintptr_t)(x) - (uintptr_t)(y))

#define _(x)    gettext(x)

#define PF(f, func) do {                                                \
        char *b = NULL;                                                 \
        int dollar = 0;                                                 \
        if (*f == ’$’)  {                                               \
                dollar++;                                               \
                *f = ’%’;                                               \
```

```
        }                                                               \
        if (havewidth)                                                  \
                if (haveprec)                                           \
                        (void) asprintf(&b, f, fieldwidth, precision, func); \
                else                                                    \
                        (void) asprintf(&b, f, fieldwidth, func);       \
        else if (haveprec)                                              \
                (void) asprintf(&b, f, precision, func);                \
        else                                                            \
                (void) asprintf(&b, f, func);                           \
        if (b) {                                                        \
                (void) fputs(b, stdout);                                \
                free(b);                                                \
        }                                                               \
        if (dollar)                                                     \
                *f = ’$’;                                               \
_NOTE(CONSTCOND) } while (0)

static int       asciicode(void);
static char     *doformat(char *, int *);
static int       escape(char *, int, size_t *);
static int       getchr(void);
static int       getfloating(long double *, int);
static int       getint(int *);
static int       getnum(intmax_t *, uintmax_t *, int);
static const char
                *getstr(void);
static char     *mknum(char *, char);
static void      usage(void);

static const char digits[] = "0123456789";

static int  myargc;
static char **myargv;
static char **gargv;
static char **maxargv;

int
main(int argc, char *argv[])
{
        size_t len;
        int end, rval;
        int chopped, end, rval;
        char *format, *fmt, *start;

        (void) setlocale(LC_ALL, "");

        argv++;
        argc--;

        /*
         * POSIX says: Standard utilities that do not accept options,
         * but that do accept operands, shall recognize "--" as a
         * first argument to be discarded.
         */
        if (argc && strcmp(argv[0], "--") == 0) {
                argc--;
                argv++;
        }

        if (argc < 1) {
                usage();
                return (1);
        }

        /*
```

```
         * Basic algorithm is to scan the format string for conversion
         * specifications -- once one is found, find out if the field
         * width or precision is a ’*’; if it is, gather up value.  Note,
         * format strings are reused as necessary to use up the provided
         * arguments, arguments of zero/null string are provided to use
         * up the format string.
         */
        fmt = format = *argv;
        (void) escape(fmt, 1, &len);    /* backslash interpretation */
        chopped = escape(fmt, 1, &len);         /* backslash interpretation */
        rval = end = 0;
        gargv = ++argv;

        for (;;) {
                maxargv = gargv;
                char **maxargv = gargv;

                myargv = gargv;
                for (myargc = 0; gargv[myargc]; myargc++)
                        /* nop */;
                start = fmt;
                while (fmt < format + len) {
                        if (fmt[0] == ’%’) {
                                (void) fwrite(start, 1, PTRDIFF(fmt, start),
                                    stdout);
                                if (fmt[1] == ’%’) {
                                        /* %% prints a % */
                                        (void) putchar(’%’);
                                        fmt += 2;
                                } else {
                                        fmt = doformat(fmt, &rval);
                                        if (fmt == NULL)
                                                return (1);
                                        end = 0;
                                }
                                start = fmt;
                        } else
                                fmt++;
                        if (gargv > maxargv)
                                maxargv = gargv;
                }
                gargv = maxargv;

                if (end == 1) {
                        warnx1(_("missing format character"), NULL, NULL);
                        return (1);
                }
                (void) fwrite(start, 1, PTRDIFF(fmt, start), stdout);
                if (!*gargv)
                if (chopped || !*gargv)
                        return (rval);
                /* Restart at the beginning of the format string. */
                fmt = format;
                end = 1;
        }
        /* NOTREACHED */
}


static char *
doformat(char *fmt, int *rval)
doformat(char *start, int *rval)
{
        static const char skip1[] = "#’-+ 0";
        static const char skip2[] = "0123456789";
        char *fmt;
```

```
        int fieldwidth, haveprec, havewidth, mod_ldbl, precision;
        char convch, nextch;
        char *start;
        char **fargv;
        char *dptr;
        int l;

        start = alloca(strlen(fmt) + 1);
        fmt = start + 1;

        dptr = start;
        *dptr++ = ’%’;
        *dptr = 0;

        fmt++;

        /* look for "n$" field index specifier */
        l = strspn(fmt, digits);
        if ((l > 0) && (fmt[l] == ’$’)) {
                int idx = atoi(fmt);
        fmt += strspn(fmt, skip2);
        if ((*fmt == ’$’) && (fmt != (start + 1))) {
                int idx = atoi(start + 1);
                if (idx <= myargc) {
                        gargv = &myargv[idx - 1];
                } else {
                        gargv = &myargv[myargc];
                }
                if (gargv > maxargv) {
                        maxargv = gargv;
                }
                fmt += l + 1;

                /* save format argument */
                fargv = gargv;
                start = fmt;
                fmt++;
        } else {
                fargv = NULL;
                fmt = start + 1;
        }

        /* skip to field width */
        while (strchr(skip1, *fmt) != NULL) {
                *dptr++ = *fmt++;
                *dptr = 0;
        }


        fmt += strspn(fmt, skip1);
        if (*fmt == ’*’) {

                fmt++;
                l = strspn(fmt, digits);
                if ((l > 0) && (fmt[l] == ’$’)) {
                        int idx = atoi(fmt);
                        if (idx <= myargc) {
                                gargv = &myargv[idx - 1];
                        } else {
                                gargv = &myargv[myargc];
                        }
                        fmt += l + 1;
                }

                if (getint(&fieldwidth))
                        return (NULL);
                if (gargv > maxargv) {
                        maxargv = gargv;
                }
                havewidth = 1;

                *dptr++ = ’*’;
                *dptr = 0;
                ++fmt;
        } else {
                havewidth = 0;

                /* skip to possible ’.’, get following precision */
                while (isdigit(*fmt)) {
                        *dptr++ = *fmt++;
                        *dptr = 0;
                fmt += strspn(fmt, skip2);
                }
        }

        if (*fmt == ’.’) {
                /* precision present? */
                fmt++;
                *dptr++ = ’.’;

                ++fmt;
                if (*fmt == ’*’) {

                        fmt++;
                        l = strspn(fmt, digits);
                        if ((l > 0) && (fmt[l] == ’$’)) {
                                int idx = atoi(fmt);
                                if (idx <= myargc) {
                                        gargv = &myargv[idx - 1];
                                } else {
                                        gargv = &myargv[myargc];
                                }
                                fmt += l + 1;
                        }

                        if (getint(&precision))
                                return (NULL);
                        if (gargv > maxargv) {
                                maxargv = gargv;
                        }
                        haveprec = 1;
                        *dptr++ = ’*’;
                        *dptr = 0;
                        ++fmt;
                } else {
                        haveprec = 0;

                        /* skip to conversion char */
                        while (isdigit(*fmt)) {
                                *dptr++ = *fmt++;
                                *dptr = 0;
                        fmt += strspn(fmt, skip2);
                        }
                }
        } else
                haveprec = 0;
        if (!*fmt) {
                warnx1(_("missing format character"), NULL, NULL);
                return (NULL);
        }
        *dptr++ = *fmt;
        *dptr = 0;
```

```
        /*
         * Look for a length modifier.  POSIX doesn’t have these, so
         * we only support them for floating-point conversions, which
         * are extensions.  This is useful because the L modifier can
         * be used to gain extra range and precision, while omitting
         * it is more likely to produce consistent results on different
         * architectures.  This is not so important for integers
         * because overflow is the only bad thing that can happen to
         * them, but consider the command  printf %a 1.1
         */
        if (*fmt == ’L’) {
                mod_ldbl = 1;
                fmt++;
                if (!strchr("aAeEfFgG", *fmt)) {
                        warnx2(_("bad modifier L for %%%c"), *fmt, NULL);
                        return (NULL);
                }
        } else {
                mod_ldbl = 0;
        }

        /* save the current arg offset, and set to the format arg */
        if (fargv != NULL) {
                gargv = fargv;
        }

        convch = *fmt;
        nextch = *++fmt;

        *fmt = ’\0’;
        switch (convch) {
        case ’b’: {
                size_t len;
                char *p;
                int getout;

                p = strdup(getstr());
                if (p == NULL) {
                        warnx2("%s", strerror(ENOMEM), NULL);
                        return (NULL);
                }
                getout = escape(p, 0, &len);
                (void) fputs(p, stdout);
                *(fmt - 1) = ’s’;
                PF(start, p);
                *(fmt - 1) = ’b’;
                free(p);

                if (getout)
                        exit(*rval);
                        return (fmt);
                break;
        }
        case ’c’: {
                char p;

                p = getchr();
                PF(start, p);
                break;
        }
        case ’s’: {
                const char *p;

                p = getstr();
                PF(start, p);
                break;
        }
        case ’d’: case ’i’: case ’o’: case ’u’: case ’x’: case ’X’: {
                char *f;
                intmax_t val;
                uintmax_t uval;
                int signedconv;

                signedconv = (convch == ’d’ || convch == ’i’);
                if ((f = mknum(start, convch)) == NULL)
                        return (NULL);
                if (getnum(&val, &uval, signedconv))
                        *rval = 1;
                if (signedconv)
                        PF(f, val);
                else
                        PF(f, uval);
                break;
        }
        case ’e’: case ’E’:
        case ’f’: case ’F’:
        case ’g’: case ’G’:
        case ’a’: case ’A’: {
                long double p;

                if (getfloating(&p, mod_ldbl))
                        *rval = 1;
                if (mod_ldbl)
                        PF(start, p);
                else
                        PF(start, (double)p);
                break;
        }
        default:
                warnx2(_("illegal format character %c"), convch, NULL);
                return (NULL);
        }
        *fmt = nextch;

        /* return the gargv to the next element */
        return (fmt);
}
```

______*unchanged_portion_omitted_*

```
static int
escape(char *fmt, int percent, size_t *len)
{
        char *save, *store, c;
        int value;

        for (save = store = fmt; ((c = *fmt) != 0); ++fmt, ++store) {
                if (c != ’\\’) {
                        *store = c;
                        continue;
                }
                switch (*++fmt) {
                case ’\0’:              /* EOS, user error */
                        *store = ’\\’;
                        *++store = ’\0’;
                        *len = PTRDIFF(store, save);
                        return (0);
                case ’\\’:              /* backslash */
                case ’\’’:              /* single quote */
                        *store = *fmt;
                        break;
                case ’a’:               /* bell/alert */
```

```
                        *store = ’\a’;
                        break;
                case ’b’:               /* backspace */
                        *store = ’\b’;
                        break;
                case ’c’:
                        if (!percent) {
                                *store = ’\0’;
                                *len = PTRDIFF(store, save);
                                return (1);
                        }
                        *store = ’c’;
                        break;
                case ’f’:               /* form-feed */
                        *store = ’\f’;
                        break;
                case ’n’:               /* newline */
                        *store = ’\n’;
                        break;
                case ’r’:               /* carriage-return */
                        *store = ’\r’;
                        break;
                case ’t’:               /* horizontal tab */
                        *store = ’\t’;
                        break;
                case ’v’:               /* vertical tab */
                        *store = ’\v’;
                        break;
                                        /* octal constant */
                case ’0’: case ’1’: case ’2’: case ’3’:
                case ’4’: case ’5’: case ’6’: case ’7’:
                        c = (!percent && *fmt == ’0’) ? 4 : 3;
                        for (value = 0;
                            c-- && *fmt >= ’0’ && *fmt <= ’7’; ++fmt) {
                                value <<= 3;
                                value += *fmt - ’0’;
                        }
                        --fmt;
                        if (percent && value == ’%’) {
                                *store++ = ’%’;
                                *store = ’%’;
                        } else
                                *store = (char)value;
                        break;
                default:
                        *store = *fmt;
                        break;
                }
        }
        *store = ’\0’;
        *len = PTRDIFF(store, save);
        return (0);
}
```

______*unchanged_portion_omitted_*

****
    1236 Fri May 9 12:12:12 2014
new/usr/src/pkg/manifests/system-test-utiltest.mf
4818 printf(1) should support n$ width and precision specifiers
****

```
#
# 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 (c) 2012 by Delphix. All rights reserved.
# Copyright 2014, OmniTI Computer Consulting, Inc. All rights reserved.
#

set name=pkg.fmri value=pkg:/system/test/utiltest@$(PKGVERS)
set name=pkg.description value="Miscellaneous Utility Unit Tests"
set name=pkg.summary value="Utility Unit Test Suite"
set name=info.classification \
    value=org.opensolaris.category.2008:Development/System
set name=variant.arch value=$(ARCH)
dir path=opt/util-tests
dir path=opt/util-tests/bin
dir path=opt/util-tests/runfiles
dir path=opt/util-tests/tests
file path=opt/util-tests/README mode=0444
file path=opt/util-tests/bin/utiltest mode=0555
file path=opt/util-tests/runfiles/default.run mode=0444
file path=opt/util-tests/tests/printf_test mode=0555
license lic_CDDL license=lic_CDDL
depend fmri=system/test/testrunner type=require
```

```
**********************************************************
     613 Fri May  9 12:12:12 2014
new/usr/src/test/Makefile
4854 printf(1) doesn't support %b and \c properly
**********************************************************
#
# 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 (c) 2012 by Delphix. All rights reserved.
# Copyright 2014 Garrett D'Amore <garrett@damore.org>
#

.PARALLEL: $(SUBDIRS)

SUBDIRS = os-tests test-runner util-tests zfs-tests
SUBDIRS = os-tests test-runner zfs-tests

include Makefile.com
```

```
**********************************************************
     552 Fri May  9 12:12:12 2014
new/usr/src/test/util-tests/Makefile
4818 printf(1) should support n$ width and precision specifiers
**********************************************************
#
# 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 (c) 2012 by Delphix. All rights reserved.
#

.PARALLEL: $(SUBDIRS)

SUBDIRS = cmd runfiles tests doc

include $(SRC)/test/Makefile.com
```

```
**********************************************************
     847 Fri May  9 12:12:12 2014
new/usr/src/test/util-tests/cmd/Makefile
4818 printf(1) should support n$ width and precision specifiers
**********************************************************
```

```
#
# 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>
# Copyright (c) 2012 by Delphix. All rights reserved.
#

include $(SRC)/Makefile.master
include $(SRC)/test/Makefile.com

ROOTOPTPKG = $(ROOT)/opt/util-tests
ROOTBIN = $(ROOTOPTPKG)/bin

PROGS = utiltest

CMDS = $(PROGS:%=$(ROOTBIN)/%)
$(CMDS) := FILEMODE = 0555

all lint clean clobber:

install: $(CMDS)

$(CMDS): $(ROOTBIN)

$(ROOTBIN):
	$(INS.dir)

$(ROOTBIN)/%: %.ksh
	$(INS.rename)
```

```
**********************************************************
    1449 Fri May  9 12:12:12 2014
new/usr/src/test/util-tests/cmd/utiltest.ksh
4818 printf(1) should support n$ width and precision specifiers
**********************************************************
```

```
#!/usr/bin/ksh

#
# 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 (c) 2012 by Delphix. All rights reserved.
# Copyright 2014, OmniTI Computer Consulting, Inc. All rights reserved.
# Copyright 2014 Garrett D’Amore <garrett@damore.org>
#

export MY_TESTS="/opt/util-tests"
runner="/opt/test-runner/bin/run"

function fail
{
        echo $1
        exit ${2:-1}
}

function find_runfile
{
        typeset distro=
        if [[ -d /opt/delphix && -h /etc/delphix/version ]]; then
                distro=delphix
        elif [[ 0 -ne $(grep -c OpenIndiana /etc/release 2>/dev/null) ]]; then
                distro=openindiana
        elif [[ 0 -ne $(grep -c OmniOS /etc/release 2>/dev/null) ]]; then
                distro=omnios
        elif [[ -f $MY_TESTS/runfiles/default.run ]]; then
                # optional catch-all
                distro=default
        fi

        [[ -n $distro ]] && echo $MY_TESTS/runfiles/$distro.run
}

while getopts c: c; do
        case $c in
        ’c’)
                runfile=$OPTARG
                [[ -f $runfile ]] || fail "Cannot read file: $runfile"
                ;;
        esac
done
shift $((OPTIND - 1))

[[ -z $runfile ]] && runfile=$(find_runfile)
[[ -z $runfile ]] && fail "Couldn’t determine distro"

$runner -c $runfile

exit $?
```

```
**********************************************************
     800 Fri May  9 12:12:12 2014
new/usr/src/test/util-tests/doc/Makefile
4818 printf(1) should support n$ width and precision specifiers
**********************************************************
```

```
#
# 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>
# Copyright (c) 2012 by Delphix. All rights reserved.
#

include $(SRC)/Makefile.master

READMES = README

ROOTOPTPKG = $(ROOT)/opt/util-tests

FILES = $(READMES:%=$(ROOTOPTPKG)/%)
$(FILES) := FILEMODE = 0444

all: $(READMES)

install: $(ROOTOPTPKG) $(FILES)

clean lint clobber:

$(ROOTOPTPKG):
        $(INS.dir)

$(ROOTOPTPKG)/%: %
        $(INS.file)
```

```
************************************************************
    2180 Fri May  9 12:12:12 2014
new/usr/src/test/util-tests/doc/README
4818 printf(1) should support n$ width and precision specifiers
************************************************************
#
# 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>
# Copyright (c) 2012 by Delphix. All rights reserved.
#

Utils Unit Test Suite README

1. What the Utils Unit Test Suite tests
2. Building and installing the Utils Unit Test Suite
3. Running the Utils Unit Test Suite
4. Test results

--------------------------------------------------------------------------------

1. What the Utils Unit Test Suite tests

The Utils unit test suite is for testing standard shell / POSIX utilities.
For example utilities such as "printf" are tested.

2. Building and installing the Utils Unit Test Suite

The Utils Unit Test Suite runs under the testrunner framework (which can be
installed as pkg:/system/test/testrunner). To build both the Utils Unit Test
Suite and the testrunner without running a full nightly:

        build_machine$ bldenv [-d] <your_env_file>
        build_machine$ cd $SRC/test
        build_machine$ dmake install
        build_machine$ cd $SRC/pkg
        build_machine$ dmake install

Then set the publisher on the test machine to point to your repository and
install the Utils Unit Test Suite.

        test_machine# pkg install pkg:/system/test/utiltest

Note, the framework will be installed automatically, as the Utils Unit Test
Suite depends on it.

3. Running the Utils Unit Test Suite

The pre-requisites for running the OS Unit Test Suite are:
        - Any user may perform these tests.

Once the pre-requisites are satisfied, simply run the ostest script:

        test_machine$ /opt/util-tests/bin/utiltest

4. Test results

```

```
While the OS Unit Test Suite is running, one informational line is printed at
the end of each test, and a results summary is printed at the end of the run.
The results summary includes the location of the complete logs, which is of the
form /var/tmp/test_results/<ISO 8601 date>.
```

```
**********************************************************
     909 Fri May  9 12:12:12 2014
new/usr/src/test/util-tests/runfiles/Makefile
4818 printf(1) should support n$ width and precision specifiers
**********************************************************
```

```
#
# 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 (c) 2012 by Delphix. All rights reserved.
# Copyright 2014, OmniTI Computer Consulting, Inc. All rights reserved.
# Copyright 2014 Garrett D'Amore <garrett@damore.org>
#

include $(SRC)/Makefile.master

SRCS = default.run

ROOTOPTPKG = $(ROOT)/opt/util-tests
RUNFILES = $(ROOTOPTPKG)/runfiles

CMDS = $(SRCS:%=$(RUNFILES)/%)
$(CMDS) := FILEMODE = 0444

all: $(SRCS)

install: $(CMDS)

clean lint clobber:

$(CMDS): $(RUNFILES) $(SRCS)

$(RUNFILES):
        $(INS.dir)

$(RUNFILES)/%: %
        $(INS.file)
```

```
**********************************************************
     654 Fri May  9 12:12:12 2014
new/usr/src/test/util-tests/runfiles/default.run
4818 printf(1) should support n$ width and precision specifiers
**********************************************************
```

```
#
# 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 (c) 2012 by Delphix. All rights reserved.
# Copyright 2014 Garrett D'Amore <garrett@damore.org>
#

[DEFAULT]
pre =
verbose = False
quiet = False
timeout = 60
post =
outputdir = /var/tmp/test_results

[/opt/util-tests/tests/printf_test]
```

```
**********************************************************
     567 Fri May  9 12:12:12 2014
new/usr/src/test/util-tests/tests/Makefile
4818 printf(1) should support n$ width and precision specifiers
**********************************************************
```

```
#
# 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 (c) 2012 by Delphix. All rights reserved.
# Copyright 2014 Garrett D'Amore <garrett@damore.org>
#

SUBDIRS = printf

include $(SRC)/test/Makefile.com
```

```
**********************************************************
     867 Fri May  9 12:12:12 2014
new/usr/src/test/util-tests/tests/printf/Makefile
4818 printf(1) should support n$ width and precision specifiers
**********************************************************
```

```
#
# 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 (c) 2012 by Delphix. All rights reserved.
# Copyright 2014 Garrett D’Amore <garrett@damore.org>
#

include $(SRC)/cmd/Makefile.cmd
include $(SRC)/test/Makefile.com

PROG = printf_test

ROOTOPTPKG = $(ROOT)/opt/util-tests
TESTDIR = $(ROOTOPTPKG)/tests

CMDS = $(PROG:%=$(TESTDIR)/%)
$(CMDS) := FILEMODE = 0555

all lint clean clobber:

install: all $(CMDS)

$(CMDS): $(TESTDIR) $(PROG).ksh

$(TESTDIR):
        $(INS.dir)

$(TESTDIR)/%: %.ksh
        $(INS.rename)
```

```
**********************************************************
    4477 Fri May  9 12:12:12 2014
new/usr/src/test/util-tests/tests/printf/printf_test.ksh
4854 printf(1) doesn’t support %b and \c properly
4818 printf(1) should support n$ width and precision specifiers
**********************************************************
```

```
#! /usr/bin/ksh
#
#
# 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>
#

PRINTF=${PRINTF:=/usr/bin/printf}

test_start() {
        print "TEST STARTING ${1}: ${2}"
}

test_pass() {
        print "TEST PASS: ${1}"
}

test_fail() {
        print "TEST FAIL: ${1}: ${2}"
        exit -1
}

checkrv() {
        if [[ $? -ne 0 ]]; then
                test_fail $1 "exit failure"
        fi
}

compare() {
        if [[ "$2" != "$3" ]]; then
                test_fail $1 "compare mismatch, got [$2] expected [$3]"
        fi
}

typeset -A tests=()


typeset -A tests[01]=()
tests[01][desc]="hexadecimal lowercase"
tests[01][format]=’%04x’
tests[01][args]="255"
tests[01][result]="00ff"

typeset -A tests[02]=()
tests[02][desc]="hexadecimal 32-bit"
tests[02][format]=’%08x’
tests[02][args]=’65537’
tests[02][result]=00010001

typeset -A tests[03]=()
```

```
tests[03][desc]="multiple arguments"
tests[03][format]=’%d %s ’
tests[03][args]="1 one 2 two 3 three"
tests[03][result]=’1 one 2 two 3 three ’

typeset -A tests[04]=()
tests[04][desc]="variable position parameters"
tests[04][format]=’%2$s %1$d ’
tests[04][args]="1 one 2 two 3 three"
tests[04][result]=’one 1 two 2 three 3 ’

typeset -A tests[05]=()
tests[05][desc]="width"
tests[05][format]=’%10s’
tests[05][args]="abcdef"
tests[05][result]=’    abcdef’

typeset -A tests[06]=()
tests[06][desc]="width and precision"
tests[06][format]=’%10.3s’
tests[06][args]="abcdef"
tests[06][result]=’       abc’

typeset -A tests[07]=()
tests[07][desc]="variable width and precision"
tests[07][format]=’%*.*s’
tests[07][args]="10 3 abcdef"
tests[07][result]=’       abc’

typeset -A tests[08]=()
tests[08][desc]="variable position width and precision"
tests[08][format]=’%2$*1$.*3$s’
tests[08][args]="10 abcdef 3"
tests[08][result]=’       abc’

typeset -A tests[09]=()
tests[09][desc]="multi variable position width and precision"
tests[09][format]=’%2$*1$.*3$s’
tests[09][args]="10 abcdef 3 5 xyz 1"
tests[09][result]=’       abc    x’

typeset -A tests[10]=()
tests[10][desc]="decimal from hex"
tests[10][format]=’%d ’
tests[10][args]="0x1000 0XA"
tests[10][result]=’4096 10 ’

typeset -A tests[11]=()
tests[11][desc]="negative dec (64-bit)"
tests[11][format]=’%x’
tests[11][args]="-1"
tests[11][result]=’ffffffffffffffff’

typeset -A tests[12]=()
tests[12][desc]="float (basic)"
tests[12][format]=’%f’
tests[12][args]="3.14"
tests[12][result]=’3.140000’

typeset -A tests[12]=()
tests[12][desc]="float precision"
tests[12][format]=’%.2f’
tests[12][args]="3.14159"
tests[12][result]=’3.14’

typeset -A tests[13]=()
```

```
tests[13][desc]="left justify"
tests[13][format]=’%-5d’
tests[13][args]="45"
tests[13][result]=’45   ’

typeset -A tests[14]=()
tests[14][desc]="newlines"
tests[14][format]=’%s\n%s\n%s’
tests[14][args]="one two three"
tests[14][result]=’one
two
three’

typeset -A tests[15]=()
tests[15][desc]="embedded octal escape"
tests[15][format]=’%s\41%s’
tests[15][args]="one two"
tests[15][result]=’one!two’

typeset -A tests[16]=()
tests[16][desc]="backslash string (%b)"
tests[16][format]=’%b’
tests[16][args]=’\0101\0102\0103’
tests[16][result]=’ABC’

typeset -A tests[17]=()
tests[17][desc]="backslash c in %b"
tests[17][format]=’%b%s’
tests[17][args]=’\0101\cone two’
tests[17][result]=’A’

typeset -A tests[18]=()
tests[18][desc]="backslash octal in format"
tests[18][format]=’HI\1120K\0112tabbed\11again’
tests[18][args]=
tests[18][result]=’HIJ0K        2tabbed again’

typeset -A tests[19]=()
tests[19][desc]="backslash octal in %b"
tests[19][format]="%b"
tests[19][args]=’HI\0112K\011tabbed’
tests[19][result]=’HIJK tabbed’

typeset -A tests[20]=()
tests[20][desc]="numeric %d and ASCII conversions"
tests[20][format]=’%d ’
tests[20][args]="3 +3 -3 \"3 \"+ ’-"
tests[20][result]=’3 3 -3 51 43 45 ’

#debug=yes

for i in "${!tests[@]}"; do
        t=test_$i
        desc=${tests[$i][desc]}
        format=${tests[$i][format]}
        args="${tests[$i][args]}"
        result=${tests[$i][result]}

        test_start $t "${tests[$i][desc]}"
        [[ -n "$debug" ]] && echo $PRINTF "$format" "${args[@]}"
        comp=$($PRINTF "$format" ${args[@]})
        checkrv $t
        [[ -n "$debug" ]] && echo "got [$comp]"
        good=$result
        compare $t "$comp" "$good"
        test_pass $t
```

```
done
```