Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'asctime.3p' command

$ man asctime.3p

ASCTIME(3P) POSIX Programmer's Manual ASCTIME(3P)

PROLOG

This manual page is part of the POSIX Programmer's Manual. The Linux implementation of this interface may differ (consult the corresponding Linux manual page for details of Linux behavior), or the interface may not be implemented on Linux.

NAME

asctime, asctime_r ? convert date and time to a string

SYNOPSIS

```
#include <time.h>
char *asctime(const struct tm *timeptr);
char *asctime_r(const struct tm *restrict tm, char *restrict buf);
```

DESCRIPTION

For asctime(): The functionality described on this reference page is aligned with the ISO C standard. Any conflict between the requirements described here and the ISO C standard is unintentional. This volume of POSIX.1?2017 defers to the ISO C standard.

The asctime() function shall convert the broken-down time in the struc? ture pointed to by timeptr into a string in the form:

```
Sun Sep 16 01:03:52 1973\n\0
```

using the equivalent of the following algorithm:

```
char *asctime(const struct tm *timeptr)
{
```

```
        static char wday_name[7][3] = {
            "Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat"
        };
        static char mon_name[12][3] = {
            "Jan", "Feb", "Mar", "Apr", "May", "Jun",
            "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"
        };
        static char result[26];
        sprintf(result, "%.3s %.3s%3d %.2d:%.2d:%.2d %d\n",
            wday_name[timeptr->tm_wday],
            mon_name[timeptr->tm_mon],
            timeptr->tm_mday, timeptr->tm_hour,
            timeptr->tm_min, timeptr->tm_sec,
            1900 + timeptr->tm_year);
        return result;
    }
```

However, the behavior is undefined if timeptr->tm_wday or timeptr->tm_mon are not within the normal ranges as defined in <time.h>, or if timeptr->tm_year exceeds {INT_MAX}-1990, or if the above algorithm would attempt to generate more than 26 bytes of output (including the terminating null).

The tm structure is defined in the <time.h> header.

The asctime(), ctime(), gmtime(), and localtime() functions shall re? turn values in one of two static objects: a broken-down time structure and an array of type char. Execution of any of the functions may over? write the information returned in either of these objects by any of the other functions.

The asctime() function need not be thread-safe.

The asctime_r() function shall convert the broken-down time in the structure pointed to by tm into a string (of the same form as that re? turned by asctime(), and with the same undefined behavior when input or output is out of range) that is placed in the user-supplied buffer pointed to by buf (which shall contain at least 26 bytes) and then re?

turn buf.

RETURN VALUE

Upon successful completion, asctime() shall return a pointer to the string. If the function is unsuccessful, it shall return NULL.

Upon successful completion, asctime_r() shall return a pointer to a character string containing the date and time. This string is pointed to by the argument buf. If the function is unsuccessful, it shall re? turn NULL.

ERRORS

No errors are defined.

The following sections are informative.

EXAMPLES

None.

APPLICATION USAGE

These functions are included only for compatibility with older imple? mentations. They have undefined behavior if the resulting string would be too long, so the use of these functions should be discouraged. On implementations that do not detect output string length overflow, it is possible to overflow the output buffers in such a way as to cause ap? plications to fail, or possible system security violations. Also, these functions do not support localized date and time formats. To avoid these problems, applications should use strftime() to generate strings from broken-down times.

Values for the broken-down time structure can be obtained by calling gmtime() or localtime().

The asctime_r() function is thread-safe and shall return values in a user-supplied buffer instead of possibly using a static data area that may be overwritten by each call.

RATIONALE

The standard developers decided to mark the asctime() and asctime_r() functions obsolescent even though asctime() is in the ISO C standard due to the possibility of buffer overflow. The ISO C standard also pro? vides the strftime() function which can be used to avoid these prob?

lems.

FUTURE DIRECTIONS

These functions may be removed in a future version.

SEE ALSO

clock(), ctime(), difftime(), gmtime(), localtime(), mktime(), strf?

time(), strptime(), time(), utime()

The Base Definitions volume of POSIX.1?2017, <time.h>

COPYRIGHT