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 'confstr.3p' command

$ man confstr.3p

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

confstr ? get configurable variables

SYNOPSIS

#include <unistd.h>

size_t confstr(int name, char *buf, size_t len);

DESCRIPTION

The confstr() function shall return configuration-defined string val? ues. Its use and purpose are similar to sysconf(), but it is used where string values rather than numeric values are returned.

The name argument represents the system variable to be queried. The im? plementation shall support the following name values, defined in <unistd.h>. It may support others:

_CS_PATH

_CS_POSIX_V7_ILP32_OFF32_CFLAGS

_CS_POSIX_V7_ILP32_OFF32_LDFLAGS

_CS_POSIX_V7_ILP32_OFF32_LIBS

_CS_POSIX_V7_ILP32_OFFBIG_CFLAGS

_CS_POSIX_V7_ILP32_OFFBIG_LDFLAGS

_CS_POSIX_V7_ILP32_OFFBIG_LIBS

_CS_POSIX_V7_LP64_OFF64_CFLAGS

_CS_POSIX_V7_LP64_OFF64_LDFLAGS

_CS_POSIX_V7_LP64_OFF64_LIBS

_CS_POSIX_V7_LPBIG_OFFBIG_CFLAGS

_CS_POSIX_V7_LPBIG_OFFBIG_LDFLAGS

_CS_POSIX_V7_LPBIG_OFFBIG_LIBS

_CS_POSIX_V7_THREADS_CFLAGS

_CS_POSIX_V7_THREADS_LDFLAGS

_CS_POSIX_V7_WIDTH_RESTRICTED_ENVS

_CS_V7_ENV

_CS_POSIX_V6_ILP32_OFF32_CFLAGS

_CS_POSIX_V6_ILP32_OFF32_LDFLAGS

_CS_POSIX_V6_ILP32_OFF32_LIBS

_CS_POSIX_V6_ILP32_OFFBIG_CFLAGS

_CS_POSIX_V6_ILP32_OFFBIG_LDFLAGS

_CS_POSIX_V6_ILP32_OFFBIG_LIBS

_CS_POSIX_V6_LP64_OFF64_CFLAGS

_CS_POSIX_V6_LP64_OFF64_LDFLAGS

_CS_POSIX_V6_LP64_OFF64_LIBS

_CS_POSIX_V6_LPBIG_OFFBIG_CFLAGS

_CS_POSIX_V6_LPBIG_OFFBIG_LDFLAGS

_CS_POSIX_V6_LPBIG_OFFBIG_LIBS

_CS_POSIX_V6_WIDTH_RESTRICTED_ENVS

_CS_V6_ENV

If len is not 0, and if name has a configuration-defined value, conf? str() shall copy that value into the len-byte buffer pointed to by buf. If the string to be returned is longer than len bytes, including the terminating null, then confstr() shall truncate the string to len-1 bytes and null-terminate the result. The application can detect that the string was truncated by comparing the value returned by confstr() with len.

If len is 0 and buf is a null pointer, then confstr() shall still re?
turn the integer value as defined below, but shall not return a string.
If len is 0 but buf is not a null pointer, the result is unspecified.
After a call to:

confstr(_CS_V7_ENV, buf, sizeof(buf))

the string stored in buf shall contain a <space>-separated list of the variable=value environment variable pairs an implementation requires as part of specifying a conforming environment, as described in the imple?mentations' conformance documentation.
If the implementation supports the POSIX shell option, the string stored in buf after a call to:

confstr(_CS_PATH, buf, sizeof(buf))

can be used as a value of the PATH environment variable that accesses all of the standard utilities of POSIX.1?2008, that are provided in a manner accessible via the exec family of functions, if the return value is less than or equal to sizeof(buf).

RETURN VALUE

If name has a configuration-defined value, confstr() shall return the size of buffer that would be needed to hold the entire configuration-defined value including the terminating null. If this return value is greater than len, the string returned in buf is truncated.
If name is invalid, confstr() shall return 0 and set errno to indicate the error.
If name does not have a configuration-defined value, confstr() shall return 0 and leave errno unchanged.

ERRORS

The confstr() function shall fail if:

EINVAL The value of the name argument is invalid.

The following sections are informative.

EXAMPLES

None.

APPLICATION USAGE

An application can distinguish between an invalid name parameter value

and one that corresponds to a configurable variable that has no config?
uration-defined value by checking if errno is modified. This mirrors
the behavior of sysconf().

The original need for this function was to provide a way of finding the
configuration-defined default value for the environment variable PATH.
Since PATH can be modified by the user to include directories that
could contain utilities replacing the standard utilities in the Shell
and Utilities volume of POSIX.1?2017, applications need a way to deter?
mine the system-supplied PATH environment variable value that contains
the correct search path for the standard utilities.

An application could use:

```
confstr(name, (char *)NULL, (size_t)0)
```

to find out how big a buffer is needed for the string value; use mal?
loc() to allocate a buffer to hold the string; and call confstr() again
to get the string. Alternately, it could allocate a fixed, static buf?
fer that is big enough to hold most answers (perhaps 512 or 1024
bytes), but then use malloc() to allocate a larger buffer if it finds
that this is too small.

RATIONALE

Application developers can normally determine any configuration vari?
able by means of reading from the stream opened by a call to:

```
popen("command -p getconf variable", "r");
```

The confstr() function with a name argument of _CS_PATH returns a
string that can be used as a PATH environment variable setting that
will reference the standard shell and utilities as described in the
Shell and Utilities volume of POSIX.1?2017.

The confstr() function copies the returned string into a buffer sup?
plied by the application instead of returning a pointer to a string.
This allows a cleaner function in some implementations (such as those
with lightweight threads) and resolves questions about when the appli?
cation must copy the string returned.

FUTURE DIRECTIONS

None.

SEE ALSO

exec, fpathconf(), sysconf()

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

The Shell and Utilities volume of POSIX.1?2017, c99