Red Hat Enterprise Linux Release 9.2 Manual Pages on 'environ.7' command

$ man environ.7

ENVIRON(7)   Linux Programmer's Manual   ENVIRON(7)

NAME

    environ - user environment

SYNOPSIS

    extern char **environ;

DESCRIPTION

    The variable environ points to an array of pointers to strings called

    the "environment". The last pointer in this array has the value NULL.

    (This variable must be declared in the user program, but is declared in

    the header file <unistd.h> if the _GNU_SOURCE feature test macro is de?

    fined.)   This array of strings is made available to the process by the

    exec(3) call that started the process. When a child process is created

    via fork(2), it inherits a copy of its parent's environment.

    By convention the strings in environ have the form "name=value". Com?

    mon examples are:

    USER   The name of the logged-in user (used by some BSD-derived pro?

        grams).

    LOGNAME

        The name of the logged-in user (used by some System-V derived

        programs).

    HOME   A user's login directory, set by login(1) from the password file

        passwd(5).

    LANG   The name of a locale to use for locale categories when not over?

ridden by LC_ALL or more specific environment variables such as
LC_COLLATE, LC_CTYPE, LC_MESSAGES, LC_MONETARY, LC_NUMERIC, and
LC_TIME (see locale(7) for further details of the LC_* environ?
ment variables).

PATH   The sequence of directory prefixes that sh(1) and many other
programs apply in searching for a file known by an incomplete
pathname.   The prefixes are separated by ':'. (Similarly one
has CDPATH used by some shells to find the target of a change
directory command, MANPATH used by man(1) to find manual pages,
and so on)

PWD   The current working directory. Set by some shells.

SHELL The pathname of the user's login shell.

TERM   The terminal type for which output is to be prepared.

PAGER The user's preferred utility to display text files.

EDITOR/VISUAL

The user's preferred utility to edit text files.

Names may be placed in the shell's environment by the export command in
sh(1), or by the setenv command if you use csh(1).

The initial environment of the shell is populated in various ways, such
as definitions from /etc/environment that are processed by pam_env(8)
for all users at login time (on systems that employ pam(8)). In addi?
tion, various shell initialization scripts, such as the system-wide
/etc/profile script and per-user initializations script may include
commands that add variables to the shell's environment; see the manual
page of your preferred shell for details.

Bourne-style shells support the syntax

NAME=value command

to create an environment variable definition only in the scope of the
process that executes command. Multiple variable definitions, sepa?
rated by white space, may precede command.

Arguments may also be placed in the environment at the point of an
exec(3). A C program can manipulate its environment using the func?
tions getenv(3), putenv(3), setenv(3), and unsetenv(3).

Note that the behavior of many programs and library routines is influ‐
enced by the presence or value of certain environment variables. Exam‐
ples include the following:

* The variables LANG, LANGUAGE, NLSPATH, LOCPATH, LC_ALL, LC_MESSAGES,
  and so on influence locale handling; see catopen(3), gettext(3), and
  locale(7).

* TMPDIR influences the path prefix of names created by tempnam(3) and
  other routines, and the temporary directory used by sort(1) and
  other programs.

* LD_LIBRARY_PATH, LD_PRELOAD, and other LD_* variables influence the
  behavior of the dynamic loader/linker.

* POSIXLY_CORRECT makes certain programs and library routines follow
  the prescriptions of POSIX.

* The behavior of malloc(3) is influenced by MALLOC_* variables.

* The variable HOSTALIASES gives the name of a file containing aliases
  to be used with gethostbyname(3).

* TZ and TZDIR give timezone information used by tzset(3) and through
  that by functions like ctime(3), localtime(3), mktime(3), strf‐
  time(3). See also tzselect(8).

* TERMCAP gives information on how to address a given terminal (or
  gives the name of a file containing such information).

* COLUMNS and LINES tell applications about the window size, possibly
  overriding the actual size.

* PRINTER or LPDEST may specify the desired printer to use. See
  lpr(1).

NOTES
    The prctl(2) PR_SET_MM_ENV_START and PR_SET_MM_ENV_END operations can
    be used to control the location of the process's environment.

BUGS
    Clearly there is a security risk here. Many a system command has been
    tricked into mischief by a user who specified unusual values for IFS or
    LD_LIBRARY_PATH.

    There is also the risk of name space pollution. Programs like make and

autoconf allow overriding of default utility names from the environment with similarly named variables in all caps. Thus one uses CC to select the desired C compiler (and similarly MAKE, AR, AS, FC, LD, LEX, RM, YACC, etc.). However, in some traditional uses such an environment variable gives options for the program instead of a pathname. Thus, one has MORE, LESS, and GZIP. Such usage is considered mistaken, and to be avoided in new programs. The authors of gzip should consider re? naming their option to GZIP_OPT.

SEE ALSO

bash(1), csh(1), env(1), login(1), printenv(1), sh(1), tcsh(1), ex? ecve(2), clearenv(3), exec(3), getenv(3), putenv(3), setenv(3), un? setenv(3), locale(7), ld.so(8), pam_env(8)

COLOPHON

This page is part of release 5.10 of the Linux man-pages project. A description of the project, information about reporting bugs, and the latest version of this page, can be found at https://www.kernel.org/doc/man-pages/.

Linux   2020-08-13   ENVIRON(7)