NAME

command – execute a simple command

SYNOPSIS

command [**–p**] *command_name* [*argument*. . .]
command [**–v** | **–V**] *command_name*

AVAILABILITY

SUNWcsu

DESCRIPTION

The **command** utility causes the shell to treat the arguments as a simple command, suppressing the shell function lookup.

If the *command_name* is the same as the name of one of the special built-in utilities, the special properties will not occur. In every other respect, if *command_name* is not the name of a function, the effect of **command** will be the same as omitting **command**.

The **command** utility also provides information concerning how a command name will be interpreted by the shell; see **–v** and **–V**.

OPTIONS

The following options are supported:

–p Perform the command search using a default value for **PATH** that is guaranteed to find all of the standard utilities.

–v Write a string to standard output that indicates the path or command that will be used by the shell, in the current shell execution environment to invoke *command_name*.

- Utilities, regular built-in utilities, *command_names* including a slash character, and any implementation-provided functions that are found using the **PATH** variable will be written as absolute path names.
- Shell functions, special built-in utilities, regular built-in utilities not associated with a **PATH** search, and shell reserved words will be written as just their names.
- An alias will be written as a command line that represents its alias definition.
- Otherwise, no output will be written and the exit status will reflect that the name was not found.

–V Write a string to standard output that indicates how the name given in the *command_name* operand will be interpreted by the shell, in the current shell execution environment. Although the format of this string is unspecified, it will indicate in which of the following categories *command_name* falls and include the information stated:

- Utilities, regular built-in utilities, and any implementation-provided functions that are found using the **PATH** variable will be identified as such and include the absolute path name in the string.
- Other shell functions will be identified as functions.
- Aliases will be identified as aliases and their definitions will be included in

the string.

- Special built-in utilities will be identified as special built-in utilities.
- Regular built-in utilities not associated with a **PATH** search will be identified as regular built-in utilities.
- Shell reserved words will be identified as reserved words.

OPERANDS

The following operands are supported:

argument One of the strings treated as an argument to *command_name*.

command_name The name of a utility or a special built-in utility.

EXAMPLES

1. Make a version of **cd** that always prints out the new working directory exactly once:

```
cd() {
        command cd "$@" >/dev/null
        pwd
}
```

2. Start off a ''secure shell script'' in which the script avoids being spoofed by its parent:

```
IFS='
'
#       The preceding value should be <space><tab><newline>.
#       Set IFS to its default value.

\unalias -a
#       Unset all possible aliases.
#       Note that unalias is escaped to prevent an alias
#       being used for unalias.

unset -f command
#       Ensure command is not a user function.

PATH="$(command -p getconf _CS_PATH):$PATH"
#       Put on a reliable PATH prefix.

#       . . .
```

At this point, given correct permissions on the directories called by **PATH**, the script has the ability to ensure that any utility it calls is the intended one. It is being very cautious because it assumes that implementation extensions may be present that would allow user functions to exist when it is invoked; this capability is not specified by this document, but it is not prohibited as an extension. For example, the **ENV** variable precedes the invocation of the script with a user startup script. Such a script could define functions to spoof the application.

ENVIRONMENT

See **environ**(5) for descriptions of the following environment variables that affect the execution of **command**: **LC_CTYPE**, **LC_MESSAGES**, and **NLSPATH**.

PATH Determine the search path used during the command search, except as described under the **–p** option.

EXIT STATUS

When the −**v** or −**V** options are specified, the following exit values are returned:

0 Successful completion.

>0 The *command_name* could not be found or an error occurred.

Otherwise, the following exit values are returned:

126 The utility specified by *command_name* was found but could not be invoked.

127 An error occurred in the **command** utility or the utility specified by *command_name* could not be found.

Otherwise, the exit status of **command** will be that of the simple command specified by the arguments to **command**.

SEE ALSO

sh(1), **type**(1)