NAME

exec, eval, source – shell built-in functions to execute other commands

SYNOPSIS

sh

exec [*argument*...]
eval [*argument*...]

csh

exec *command*
eval *argument* ...
source [–**h**] *name*

ksh

† **exec** [*arg* ...]
† **eval** [*arg* ...]

DESCRIPTION

sh

The **exec** command specified by the arguments is executed in place of this shell without creating a new process. Input/output arguments may appear and, if no other arguments are given, cause the shell input/output to be modified.

The *argument*s to the **eval** built-in are read as input to the shell and the resulting command(s) executed.

csh

exec executes *command* in place of the current shell, which terminates.

eval reads its *argument*s as input to the shell and executes the resulting command(s). This is usually used to execute commands generated as the result of command or variable substitution.

source reads commands from *name*. **source** commands may be nested, but if they are nested too deeply the shell may run out of file descriptors. An error in a sourced file at any level terminates all nested **source** commands.

–**h** Place commands from the file *name* on the history list without executing them.

ksh

With the **exec** built-in, if *arg* is given, the command specified by the arguments is executed in place of this shell without creating a new process. Input/output arguments may appear and affect the current process. If no arguments are given the effect of this command is to modify file descriptors as prescribed by the input/output redirection list. In this case, any file descriptor numbers greater than 2 that are opened with this mechanism are closed when invoking another program.

The arguments to **eval** are read as input to the shell and the resulting command(s) executed.

On this man page, **ksh**(1) commands that are preceded by one or two † (daggers) are treated specially in the following ways:

1. Variable assignment lists preceding the command remain in effect when the command completes.
2. I/O redirections are processed after variable assignments.
3. Errors cause a script that contains them to abort.

4. Words, following a command preceded by ++ that are in the format of a variable assignment, are expanded with the same rules as a variable assignment. This means that tilde substitution is performed after the = sign and word splitting and file name generation are not performed.

SEE ALSO **csh**(1), **ksh**(1), **sh**(1)