IEEE P1003.2a Draft 8 – December 1991

not supported); see their descriptions.

5.1 `alias` — Define or display aliases

5.1.1 Synopsis

`alias` [*alias-name* [=*string*] . . .]

5.1.2 Description

The `alias` utility shall create or redefine alias definitions or write the values of existing alias definitions to standard output. An alias definition provides a string value that shall replace a command name when it is encountered. See 3.3.1.

An alias definition shall affect the current shell execution environment and the execution environments of the subshells of the current shell. When used as specified by this standard, the alias definition shall not affect the parent process of the current shell nor any utility environment invoked by the shell. See 3.12.

5.1.3 Options

None.

5.1.4 Operands

The following operands shall be supported by the implementation:

alias-name Write the alias definition to standard output.

alias-name =*string*
Assign the value of *string* to the alias *alias-name* .

If no operands are given, all alias definitions shall be written to standard output.

5.1.5 External Influences

5.1.5.1 Standard Input

None.

5.1.5.2 Input Files

None.

5.1.5.3 Environment Variables

The following environment variables shall affect the execution of `alias`:

LANG This variable shall determine the locale to use for the locale categories when both **LC_ALL** and the corresponding environment variable (beginning with **LC_**) do not specify a locale. See 2.6.

LC_ALL This variable shall determine the locale to be used to override any values for locale categories specified by the settings of **LANG** or any environment variables beginning with **LC_**.

LC_CTYPE This variable shall determine the locale for the interpretation of sequences of bytes of text data as characters (e.g., single- versus multibyte characters in arguments).

LC_MESSAGES This variable shall determine the language in which messages should be written.

5.1.5.4 Asynchronous Events

Default.

5.1.6 External Effects

5.1.6.1 Standard Output

The format for displaying aliases (when no operands or only *name* operands are specified) shall be:

`"%s=%s\n"`, *name* , *value*

The *value* string shall be written with appropriate quoting so that it is suitable 7
for re-input to the shell. See the description of shell quoting in 3.2. 7

5.1.6.2 Standard Error

Used only for diagnostic messages.

5.1.6.3 Output Files

None.

5.1.7 Extended Description

None.

5.1.8 Exit Status

The `alias` utility shall exit with one of the following values:

- 0 Successful completion.
- >0 One of the *name* operands specified did not have an alias definition, or an error occurred.

5.1.9 Consequences of Errors

Default.

5.1.10 Rationale. *(This subclause is not a part of P1003.2a)*

Usage, Examples

None.

History of Decisions Made

The `alias` description is based on that from historical KornShell implementations. Known differences exist between that and the C-Shell. The KornShell version was adopted to be consistent with all the other KornShell features in POSIX. 2, such as command-line editing.

Since `alias` affects the current shell execution environment, it is generally provided as a shell regular built-in.

Historical versions of the KornShell have allowed aliases to be exported to scripts that are invoked by the same shell. This is triggered by the `alias −x` flag; it is allowed by POSIX. 2 only when an explicit extension such as `−x` is used. The working group felt that aliases were of use primarily to interactive users and that they should normally not affect shell scripts called by those users; functions are available to such scripts.

Historical versions of the KornShell have not written aliases in a quoted manner
suitable for reentry to the shell, but the base POSIX. 2 standard has made this a 7
requirement for all similar output. Therefore, consistency with the base standard 7
was chosen over this minor detail of historical practice. 7

5.2 `at` — Execute commands at a later time

5.2.1 Synopsis

`at` [`−m`][`−f` *file*][`−q` *queuename*] `−t` *time*

`at` [`−m`][`−f` *file*][`−q` *queuename*] *timespec* . . .

`at` `−r` *at_job_id* . . .

`at` `−l` `−q` *queuename*

`at` `−l` [*at_job_id* . . .]

5.2.2 Description

The `at` utility shall read commands from standard input and group them together as an *at-job* , to be executed at a later time.

The at-job shall be executed in a separate invocation of the shell, running in a separate process group with no controlling terminal, except that the environment variables, current working directory, file creation mask, and other implementation-defined execution-time attributes in effect when the `at` utility is executed shall be retained and used when the at-job is executed.

When the at-job is submitted, the *at_job_id* and scheduled time shall be written to standard error. The *at_job_id* is an identifier that shall be a string consisting solely of alphanumeric characters and the period character. The *at_job_id* shall be assigned by the system when the job is scheduled such that it uniquely identifies a particular job.

User notification and the processing of the job's standard output and standard error are described under the `−m` option.

5.2.3 Options

The `at` utility shall conform to the utility argument syntax guidelines described in 2.10.2.

The following options shall be supported by the implementation:

`−f` *file* Specify the pathname of a file to be used as the source of the at-job, instead of standard input.

`−l` (The letter ell.) Report all jobs scheduled for the invoking user if no *at_job_id* operands are specified. If *at_job_id* s are specified, report only information for these jobs. The output shall be written to standard output.