NAME

= (equals) − shell assignment command

SYNOPSIS

= letter [arg1 [arg2]]

DESCRIPTION

The ‘‘=’’ command provides shell string variables. The 26*letter* v ariables (‘a’−‘z’) are referenced in later commands in the manner of shell arguments, i.e.: *$a, ..., $z*. If no arguments are given, the standard input is read to newline or EOT for the value. The exit code is set to 0 if a newline is found in the input; it is set to 1 otherwise, thus providing and end-of-file indicator. If*ar g1* is the only argument, or if two non-null arguments are given, the variable is set to *arg1*, and the exit code set to 0. If two arguments are given, and if *arg1* is a null string, the value of *arg2* is assigned to the variable, and the exit code is set to 1, permitting a convenient default mechanism:

```
= a "$1" "default value" && shift
```

The ‘‘=’’ command works either at the terminal, or in shell command files. The variables can be assigned repeatedly. Storage is assigned as needed, but there is no recovery.

Some *letter* variables have predefined meanings and are initialized once at the time the Shell begins execution:

$n The argument count. ‘‘sh file arg1 arg2 arg3’’ has the value 3. The shift command does not change the value of $n.

$p This variable holds the shell search sequence of pathname prefixes for command execution. Alternatives are separated by ‘‘:’’. The default initial value is:

```
= p ":/bin:/usr/bin"
```

which prepends successively
the null pathname (execute from current dir.),
/bin,
/usr/bin.

Using the same type of specification, users may choose their own sequence by storing it in a file named ‘‘.path’’ in their login directory. The ‘‘.path’’ information passes to successive shells (and other commands like *time*(I) or *nohup*(I)); the $p value does not. In any case, no prepending occurs when a command name contains a ‘/’.

$r exit(status) of the most recent command executed by the Shell. The value is ASCII numeric, and is initially ‘0’. At end-of-file the shell exits with the value of *$r.*

$s Name of login (starting) directory.

$t Terminal identification letter or number: /dev/tty$t is a file name for the terminal.

$w First component in $s pathname, i.e., file system name (such as /usr).

$z Is the name of the Shell. Its default value is ‘/bin/sh’, but this can be overridden by supplying a name as the second line of the ‘.path’ file.

Note that variables (‘a’ − ‘m’) are guaranteed to be initialized to null strings and usable in any way desired. Variables (‘n’ − ‘z’) may acquire special uses in the future. The values of $n, $s, $t, and $w may reasonably be modified; it is catastrophic to change $p; it is possible, but useless to modify $r.

The **‘‘=’’** command is executed within the shell. Note that it is commonly used to read the first line of output from a pipe or a line from the terminal, for example:

```
grep −c string file | = a
```

or:

```
= a </dev/tty
```

EXIT CODES

0 − normal read, or first of two arguments is not null.
1 − end-of-file, or first of two arguments is null.

SEE ALSO

expr(I), sh(I)