NAME

lines – script for the init process

DESCRIPTION

The *lines* file supplies the script to *init*'s role as a general process dispatcher. The process that constitutes the majority of *init*'s process dispatching activities is the line process **/etc/getty** which initiates individual terminal lines. Other processes typically dispatched by *init* are daemons and the shell.

The lines file is composed of entries that are position dependent and have the following format:

id:rstate:action:shellcm:process

Each entry is delimited by a newline, however a backslash (\) preceding a newline indicates a continuation of the entry. Up to 512 characters per entry are permitted and comments may be inserted before or after an entry by using the C comment convention (i.e. /* comment */). (Comments are not included in the 512 character limit.) There are no limits (other than maximum entry size) imposed on the number of entries within the *lines* file. The entry fields are:

id This is one or two characters (other than *xx, !B, RL, OT, NT*) used to uniquely identify an entry. For compactness of syntax, however, when spawning terminal processes these characters must be the name of the line the terminal process is to open (e.g., *aa* causes **/dev/ln***aa* to be the line on which a terminal process is spawned). If a line monitor other than **/etc/getty** is spawned this convention should also be observed in order to produce consistent line accounting.

rstate This defines the *run state* in which this entry is to be processed. Run states effectively correspond to a configuration of processes in the system. That is, each process spawned by *init* is assigned a run state or run states in which it is allowed to exist. The *run states* are represented by a number ranging from 0 through 6. There is a *run state* 7; however, this is predefined as single-user mode and *init* does not scan the lines file in single-user mode so that a process with run state 7 in the lines file is meaningless. As an example, if the system is in *run state* 1, only those entries having a 1 in the *run state* field will be processed. When *init* is requested to change *run states,* all processes which do not have an entry in the *rstate* field for the target *run state* will be sent the warning signal and allowed a 20 second grace period before being forcibly terminated by a kill signal. The *rstate* field can define multiple *run states* for a process by selecting more than one run state in any combination from 0 through 6. The *run states* must appear as an unbroken string (i.e. '024, '3526, '0125, etc.) in the *rstate* field. If no *run state* is specified (no blanks or tabs may appear in the *rstate* field); then the process is assumed to be valid at *all run states.* There are three other values *a, b,* and *c* which can appear in the *rstate* field even though they are not true *run states.* Entries which have these characters in the *rstate* field are processed only when the *telinit* process requests them to be run (regardless of the current *run state* of the system). They differ from *run states* in that *init* can never enter *run state a, b* or *c.* Also, a request for the execution of any of these processes does not change the current *run state.*

action Key words in this field tell *init* how to treat the process specified in the *process* field. The actions recognized by *init* are as follows:

respawn If the process does not exist then start the process, do not wait for its termination (continue scanning the *lines* file), and when it dies restart the process. If the process currently exists then do nothing and continue scanning the *lines* file.

wait Upon *init's* entering the run state that matches the entry's *rstate,* start the process and wait for its termination. All subsequent reads of the *lines*