NAME

runlevel − event signalling change of system runlevel

SYNOPSIS

runlevel RUNLEVEL=*RUNLEVEL* **PREVLEVEL**=*PREVLEVEL* [*ENV*]...

DESCRIPTION

This page describes the **runlevel** Upstart event, and the general implementation of runlevels in the Upstart system. For the runlevel tool, see **runlevel**(8)

The runlevel event

The **runlevel** event signals a change of system runlevel. The new system runlevel is given in the **RUNLEVEL** argument, and the previous system runlevel in the **PREVLEVEL** argument (which may be empty).

Additional environment may follow these depending on the runlevel, and the tool that emitted the event. The **shutdown**(8) tool will supply an **INIT_HALT** variable set to either *HALT* or *POWEROFF* when called with **-H** or **-P** respectively.

Runlevels

Runlevels are a concept from UNIX® System V used by the **init**(8) daemon or other system initialisation system to define modes of system operation.

Eight runlevels are permitted, the first seven are numbered **0**-**6** and the eighth is named **S** or **s** (both are permitted).

Services and other system components are said to exist in one or more runlevels. When switching from one runlevel to another, the services that should not exist in the new runlevel are stopped and the services that only exist in the new runlevel are started.

This is performed by the **/etc/init.d/rc** script executed on a change of runlevel (by jobs run on the **runlevel** event in the Upstart system). This script examines symlinks in the */etc/rc?.d* directories, symlinks beginning **K** are services to be stopped and symlinks beginning **S** are services to be started.

The authorative documentation for this process can be found in the *System run levels and init.d scripts* section of the *Debian Policy Manual. This may be currently found at* <**http://www.debian.org/doc/debian-policy/ch-opersys.html#s-sysvinit**>

Runlevels **0**, **1** and **6** are reserved. Runlevel **0** is used to halt the system and **6** to reboot the system. Runlevel **1** is used to bring the system back down into single-user mode, after which the runlevel will be **S**.

System V initialisation in Upstart

The compatible implementation of runlevels permits Upstart jobs to be run on the **runlevel** event that perform the same functionality as the original System V **init**(8) daemon.

The */etc/init/rc.conf* job is run on the **runlevel** event, thus receiving the **RUNLEVEL** and **PREVLEVEL** environment variables. Its sole job is to execute the **/etc/init.d/rc** script, passing the new runlevel as an argument.

Initial system startup is provided by the */etc/init/rc-sysinit.conf* job. This is run on the **startup**(7) event, and is primarily responsible for running the **/etc/init.d/rc** script with the special **S** argument and calling **telinit**(8) to switch into the default runlevel when done. This also handles the **-b**, **emergency**, **-s** and **single** kernel command-line options as well as specifying an alternate runlevel on the kernel command-line.

Finally the */etc/init/rcS.conf* job handles the special case of entering the single-user runlevel and providing a login shell. Once that shell terminates, this restarts the *rc-sysinit* job to re-enter the default runlevel.

Implementation of runlevels in Upstart

The Upstart **init**(8) daemon has no native concepts of runlevel, and unlike the System V daemon, makes no attempt to keep track of the current runlevel.

Instead a compatible implementation is provided by the **runlevel**(8), **telinit**(8) and **shutdown**(8) tools supplied with Upstart.

The **telinit**(8) and **shutdown**(8) tools are used by system administrators to change the runlevel, thus they both generate this **runlevel** event obtaining the value for the **PREVLEVEL** environment variable from their own environment (the **RUNLEVEL** variable) or the */var/run/utmp* file.

Additionally they update the */var/run/utmp* file with the new runlevel, and append a log entry to the */var/log/wtmp* file.

The **runlevel**(8) tool may be used by system administrators to obtain the current runlevel, this reads the **RUNLEVEL** and **PREVLEVEL** variables from its own environment or reads the current and previous runlevel from */var/run/utmp*.

The **who**(1) **-r** command may also be used to read the current runlevel from */var/run/utmp*.

This provides full compatibility with System V.

During the boot scripts, where the */var/run/utmp* file may not yet be writable, the **RUNLEVEL** and **PREVLEVEL** environment variables will be available so **telinit**(8) will still provide the correct values.

Once the boot scripts have finished, while the environment variables may no longer be available, the */var/run/utmp* file will be and the most recent **telinit**(8) invocation should have successfully written to it.

Boot time records

The **telinit**(8) tool also takes care of writing the boot time record to both */var/run/utmp* and */var/log/wtmp*.

This is written if the previous runlevel in these files does not match the previous runlevel obtained from its environment. In general this occurs when switching from runlevel **S** to the default runlevel, at which point the */var/run/utmp* and */var/log/wtmp* files are both writable, and the **telinit**(8) invocation to do the switch has **RUNLEVEL**=*S* in its environment.

EXAMPLE

A service running in the typical multi-user runlevels might use:

```
start on runlevel [2345]
stop on runlevel [!2345]
```

SEE ALSO

runlevel(8) **init**(8) **telinit**(8) **shutdown**(8) **who**(1)