NAME

start−stop−daemon – start and stop system daemon programs

SYNOPSIS

start-stop-daemon [*option*...] *command*

DESCRIPTION

start-stop-daemon is used to control the creation and termination of system-level processes. Using one of the matching options, **start-stop-daemon** can be configured to find existing instances of a running process.

Note: Unless −−**pid** or −−**pidfile** are specified, **start-stop-daemon** behaves similar to **killall**(1). **start-stop-daemon** will scan the process table looking for any processes which match the process name, parent pid, uid, and/or gid (if specified). Any matching process will prevent −−**start** from starting the daemon. All matching processes will be sent the TERM signal (or the one specified via −−**signal** or −−**retry**) if −−**stop** is specified. For daemons which have long-lived children which need to live through a −−**stop**, you must specify a pidfile.

COMMANDS

−S, −−**start** [−−] *arguments*

　　Check for the existence of a specified process. If such a process exists, **start-stop-daemon** does nothing, and exits with error status 1 (0 if −−**oknodo** is specified). If such a process does not exist, it starts an instance, using either the executable specified by −−**exec** or, if specified, by −−**startas**. Any arguments given after −− on the command line are passed unmodified to the program being started.

−K, −−**stop**

　　Checks for the existence of a specified process. If such a process exists, **start-stop-daemon** sends it the signal specified by −−**signal**, and exits with error status 0. If such a process does not exist, **start-stop-daemon** exits with error status 1 (0 if −−**oknodo** is specified). If −−**retry** is specified, then **start-stop-daemon** will check that the process(es) have terminated.

−T, −−**status**

　　Check for the existence of a specified process, and returns an exit status code, according to the LSB Init Script Actions (since version 1.16.1).

−H, −−**help**

　　Show usage information and exit.

−V, −−**version**

　　Show the program version and exit.

OPTIONS

Matching options

−−**pid** *pid*

　　Check for a process with the specified *pid* (since version 1.17.6). The *pid* must be a number greater than 0.

−−**ppid** *ppid*

　　Check for a process with the specified parent pid *ppid* (since version 1.17.7). The *ppid* must be a number greater than 0.

−p, −−**pidfile** *pidfile*

　　Check whether a process has created the file *pidfile*.

　　Note: Using this matching option alone might cause unintended processes to be acted on, if the old process terminated without being able to remove the *pidfile*.

　　Warning: Using this match option with a world-writable pidfile or using it alone with a daemon that writes the pidfile as an unprivileged (non-root) user will be refused with an error (since version 1.19.3) as this is a security risk, because either any user can write to it, or if the daemon gets compromised, the contents of the pidfile cannot be trusted, and then a privileged runner (such as an init script executed as root) would end up acting on any system process. Using */dev/null* is exempt from these checks.

−x, **−−exec** *executable*

Check for processes that are instances of this *executable*. The *executable* argument should be an absolute pathname.

Note: This might not work as intended with interpreted scripts, as the executable will point to the interpreter. Take into account processes running from inside a chroot will also be matched, so other match restrictions might be needed.

−n, **−−name** *process-name*

Check for processes with the name *process-name*. The *process-name* is usually the process filename, but it could have been changed by the process itself.

Note: On most systems this information is retrieved from the process comm name from the kernel, which tends to have a relatively short length limit (assuming more than 15 characters is non-portable).

−u, **−−user** *username*|*uid*

Check for processes owned by the user specified by *username* or *uid*.

Note: Using this matching option alone will cause all processes matching the user to be acted on.

Generic options

−g, **−−group** *group*|*gid*

Change to *group* or *gid* when starting the process.

−s, **−−signal** *signal*

With **−−stop**, specifies the signal to send to processes being stopped (default TERM).

−R, **−−retry** *timeout*|*schedule*

With **−−stop**, specifies that **start-stop-daemon** is to check whether the process(es) do finish. It will check repeatedly whether any matching processes are running, until none are. If the processes do not exit it will then take further action as determined by the schedule.

If *timeout* is specified instead of *schedule*, then the schedule *signal*/*timeout*/**KILL**/*timeout* is used, where *signal* is the signal specified with **−−signal**.

schedule is a list of at least two items separated by slashes (*/*); each item may be −*signal-number* or [−]*signal-name*, which means to send that signal, or *timeout*, which means to wait that many seconds for processes to exit, or **forever**, which means to repeat the rest of the schedule forever if necessary.

If the end of the schedule is reached and **forever** is not specified, then **start-stop-daemon** exits with error status 2. If a schedule is specified, then any signal specified with **−−signal** is ignored.

−a, **−−startas** *pathname*

With **−−start**, start the process specified by *pathname*. If not specified, defaults to the argument given to **−−exec**.

−t, **−−test**

Print actions that would be taken and set appropriate return value, but take no action.

−o, **−−oknodo**

Return exit status 0 instead of 1 if no actions are (would be) taken.

−q, **−−quiet**

Do not print informational messages; only display error messages.

−c, **−−chuid** *username*|*uid*[**:***group*|*gid*]

Change to this username/uid before starting the process. You can also specify a group by appending a **:**, then the group or gid in the same way as you would for the **chown**(1) command (*user***:***group*). If a user is specified without a group, the primary GID for that user is used. When using this option you must realize that the primary and supplemental groups are set as well, even if the **−−group** option is not specified. The **−−group** option is only for groups that the user isn't normally a member of (like adding per process group membership for generic users like **nobody**).

−r, **−−chroot** *root*
> Change directory and chroot to *root* before starting the process. Please note that the pidfile is also written after the chroot.

−d, **−−chdir** *path*
> Change directory to *path* before starting the process. This is done after the chroot if the **−r|−−chroot** option is set. When not specified, **start-stop-daemon** will change directory to the root directory before starting the process.

−b, **−−background**
> Typically used with programs that don't detach on their own. This option will force **start-stop-daemon** to fork before starting the process, and force it into the background.

> **Warning: start-stop-daemon** cannot check the exit status if the process fails to execute for **any** reason. This is a last resort, and is only meant for programs that either make no sense forking on their own, or where it's not feasible to add the code for them to do this themselves.

−−notify−await
> Wait for the background process to send a readiness notification before considering the service started (since version 1.19.3). This implements parts of the systemd readiness protocol, as specified in the **sd_notify**(3) man page. The following variables are supported:

> **READY=1**
>> The program is ready to give service, so we can exit safely.

> **EXTEND_TIMEOUT_USEC=***number*
>> The program requests to extend the timeout by *number* microseconds. This will reset the current timeout to the specified value.

> **ERRNO=***number*
>> The program is exiting with an error. Do the same and print the user-friendly string for the **errno** value.

−−notify−timeout *timeout*
> Set a timeout for the **−−notify−await** option (since version 1.19.3). When the timeout is reached, **start-stop-daemon** will exit with an error code, and no readiness notification will be awaited. The default is **60** seconds.

−C, **−−no−close**
> Do not close any file descriptor when forcing the daemon into the background (since version 1.16.5). Used for debugging purposes to see the process output, or to redirect file descriptors to log the process output. Only relevant when using **−−background**.

−O, **−−output** *pathname*
> Redirect **stdout** and **stderr** to *pathname* when forcing the daemon into the background (since version 1.20.6). Only relevant when using **−−background**.

−N, **−−nicelevel** *int*
> This alters the priority of the process before starting it.

−P, **−−procsched** *policy***:***priority*
> This alters the process scheduler policy and priority of the process before starting it (since version 1.15.0). The priority can be optionally specified by appending a **:** followed by the value. The default *priority* is 0. The currently supported policy values are **other**, **fifo** and **rr**.

−I, **−−iosched** *class***:***priority*
> This alters the IO scheduler class and priority of the process before starting it (since version 1.15.0). The priority can be optionally specified by appending a **:** followed by the value. The default *priority* is 4, unless *class* is **idle**, then *priority* will always be 7. The currently supported values for *class* are **idle**, **best-effort** and **real-time**.

−k, **−−umask** *mask*
> This sets the umask of the process before starting it (since version 1.13.22).

−m, **−−make−pidfile**
> Used when starting a program that does not create its own pid file. This option will make **start-stop-daemon** create the file referenced with **−−pidfile** and place the pid into it just before executing the process. Note, the file will only be removed when stopping the program if **−−remove−pidfile** is used.
>
> **Note:** This feature may not work in all cases. Most notably when the program being executed forks from its main process. Because of this, it is usually only useful when combined with the **−−background** option.

−−remove−pidfile
> Used when stopping a program that does not remove its own pid file (since version 1.17.19). This option will make **start-stop-daemon** remove the file referenced with **−−pidfile** after terminating the process.

−v, **−−verbose**
> Print verbose informational messages.

EXIT STATUS

0   The requested action was performed. If **−−oknodo** was specified, it's also possible that nothing had to be done. This can happen when **−−start** was specified and a matching process was already running, or when **−−stop** was specified and there were no matching processes.

1   If **−−oknodo** was not specified and nothing was done.

2   If **−−stop** and **−−retry** were specified, but the end of the schedule was reached and the processes were still running.

3   Any other error.

When using the **−−status** command, the following status codes are returned:

0   Program is running.

1   Program is not running and the pid file exists.

3   Program is not running.

4   Unable to determine program status.

EXAMPLE

Start the **food** daemon, unless one is already running (a process named food, running as user food, with pid in food.pid):

```
start-stop-daemon --start --oknodo --user food --name food \
      --pidfile /run/food.pid --startas /usr/sbin/food \
      --chuid food -- --daemon
```

Send **SIGTERM** to **food** and wait up to 5 seconds for it to stop:

```
start-stop-daemon --stop --oknodo --user food --name food \
      --pidfile /run/food.pid --retry 5
```

Demonstration of a custom schedule for stopping **food**:

```
start-stop-daemon --stop --oknodo --user food --name food \
      --pidfile /run/food.pid --retry=TERM/30/KILL/5
```