

## BROUILLON

# Lancement automatique de programmes au démarrage

Ce tutoriel décrit les logiciels à installer et la démarche à suivre pour faire ...

## Première méthode : systemd

Sous Debian 8, le mécanisme init.d est remplacé par le mécanisme **systemd** que nous présentons ici.



Nous prenons l'exemple de vncserver sur une `<abbr>Raspberry pi(RPI)</abbr>`.

### Première étape

Créez avec les droits d'administration le fichier `/etc/systemd/system/monprogramme.service` pour y écrire ceci :

[/etc/systemd/system/monprogramme.service](#)

```
[Unit]
Description=Programme monprogramme

[Service]
ExecStart=/bin/monprogramme

[Install]
WantedBy=multi-user.target
```

#### **Section [Unit]**



? Description=Programme monprogramme :: Commentaire descriptif !! **Section [Service]**

? ExecStart=/bin/monprogramme :: commande qui démarre le service. !! **Section [Install]**

? WantedBy=multi-user.target :: s'exécutera au redémarrage

si on l'active par une commande enable !!

Dans certains cas, le programme doit être lancé sous un USER particulier, comme pour vncserver sous l'utilisateur **pi** :

```
[Unit]
Description=Programme
monprogramme

[Service]
RemainAfterExit=yes
Environment=VNCUSER='pi'
ExecStart=/bin/su $VNCUSER
-c 'vncserver :1'
ExecStop=/bin/su $VNCUSER
-c 'vncserver -kill :1'

[Install]
WantedBy=multi-user.target
```



**Section [Unit]**

? Description=Serveur vncserver ::  
Commentaire descriptif !! **Section [Service]**

? RemainAfterExit=yes :: Nécessaire pour pouvoir démarrer ou arrêter le service. !!

? ExecStart=/bin/su - pi -c '/usr/bin/vncserver :1' :: commande qui démarre le service. (exécutée sous l'utilisateur pi) !!

? ExecSop=/bin/su - pi -c '/usr/bin/vncserver -kill :1' :: commande qui démarre le service. (exécutée sous l'utilisateur pi) !!

**Section [Install]**

? WantedBy=multi-user.target :: s'exécutera au redémarrage si une commande enable est lancée. !!  
Pour le format du fichier **.service**, voir les pages de man :

- **systemd.service**  
<http://www.freedesktop.org/software/systemd/man/systemd>.

[service.html](#)



- sections [unit] et [install] :  
**systemd.unit**  
<http://www.freedesktop.org/software/systemd/man/systemd.unit.html>

ou la page [Systemd](#)

## Etape suivante : activation



L'activer en lançant :

- ```
systemctl daemon-reload
systemctl enable vncserver
systemctl start vncserver
```

## Deuxième méthode : avec un fichier `vncserver.desktop`

Créez le fichier

**`/home/pi/.config/monprogramme.desktop`** pour y écrire ceci :

[/home/pi/.config/vncserver.desktop](#)

```
[Desktop Entry]
Type=Application
Name=monprogramme
Exec=monprogramme <args...>
StartupNotify=false
```

Au prochain redémarrage, monprogramme sera démarré.

Pour l'arrêter, il faut faire :

- `killall monprogramme`

et pour le démarrer :

- `monprogramme <args...>`

## Troisième méthode : avec `init.d`

`<term monprogramme>nom du service à créer</term>`

Ce tutoriel décrit la démarche à suivre pour qu'un programme démarre en tant que service.

Il faut pour cela créer un lanceur dans `/etc/init.d`.

Pour un démarrage automatique, il faut l'ajouter comme service au démarrage de Linux.



### Première étape : créer le lanceur

Pour créer un lanceur dans `/etc/init.d`, le plus simple est de partir du modèle fourni `/etc/init.d/skeleton`.

Copiez le fichier `/etc/init.d/skeleton` fourni comme modèle :

- `sudo cp /etc/init.d/skeleton /etc/init.d/monprogramme`

Voici le contenu du fichier `skeleton` :

[/etc/init.d/skeleton](#)



```
#!/bin/sh
### BEGIN INIT
INFO
# Provides:
# skeleton
# Required-Start:
# $remote_fs
```



```
# Required-Stop:
$remote_fs
# Default-Start:
2 3 4 5
# Default-Stop:
0 1 6
# Short-
Description:
Example
initscript
# Description:
This file should
be used to
construct scripts
to be
#
placed in
/etc/init.d.
### END INIT INFO

# Author: Foo Bar
<foobar@baz.org>
#
# Please remove
the "Author"
lines above and
replace them
# with your own
name if you copy
and modify this
script.

# Do NOT "set -e"

# PATH should
only include
/usr/* if it runs
after the
mountnfs.sh
script
PATH=/sbin:/usr/s
bin:/bin:/usr/bin
DESC="Description
of the service"
NAME=daemonexecut
ablename
DAEMON=/usr/sbin/
$NAME
DAEMON_ARGS="--
options args"
PIDFILE=/var/run/
```



```
$NAME.pid
SCRIPTNAME=/etc/i
nit.d/$NAME

# Exit if the
package is not
installed
[ -x "$DAEMON" ]
|| exit 0

# Read
configuration
variable file if
it is present
[ -r
/etc/default/$NAM
E ] && .
/etc/default/$NAM
E

# Load the
VERBOSE setting
and other rcS
variables
.
/lib/init/vars.sh

# Define LSB
log_* functions.
# Depend on lsb-
base (>= 3.0-6)
to ensure that
this file is
present.
. /lib/lsb/init-
functions

#
# Function that
starts the
daemon/service
#
do_start()
{
    # Return
    # 0 if
daemon has been
started
    # 1 if
daemon was
already running
```



```
# 2 if
daemon could not
be started
    start-stop-
daemon --start --
quiet --pidfile
$PIDFILE --exec
$DAEMON --test >
/dev/null \
    || return
1
    start-stop-
daemon --start --
quiet --pidfile
$PIDFILE --exec
$DAEMON -- \
$DAEMON_ARGS \
    || return
2
    # Add code
here, if
necessary, that
waits for the
process to be
ready
    # to handle
requests from
services started
subsequently
which depend
    # on this
one. As a last
resort, sleep for
some time.
}

#
# Function that
stops the
daemon/service
#
do_stop()
{
    # Return
    # 0 if
daemon has been
stopped
    # 1 if
daemon was
already stopped
    # 2 if
```



```
daemon could not
be stopped
    # other if
a failure
occurred
    start-stop-
daemon --stop --
quiet --
retry=TERM/30/KIL
L/5 --pidfile
$PIDFILE --name
$NAME
    RETVAL="$?"
    [ "$RETVAL" =
2 ] && return 2
    # Wait for
children to
finish too if
this is a daemon
that forks
    # and if the
daemon is only
ever run from
this initscript.
    # If the
above conditions
are not satisfied
then add some
other code
    # that waits
for the process
to drop all
resources that
could be
    # needed by
services started
subsequently. A
last resort is to
    # sleep for
some time.
    start-stop-
daemon --stop --
quiet --oknodo --
retry=0/30/KILL/5
--exec $DAEMON
    [ "$?" = 2 ]
&& return 2
    # Many
daemons don't
delete their
pidfiles when
```





```
they exit.
    rm -f
    $PIDFILE
    return
    "$RETVAL"
}

#
# Function that
# sends a SIGHUP to
# the
# daemon/service
#
do_reload() {
    #
    # If the
    # daemon can reload
    # its configuration
    # without
    # restarting
    # (for example,
    # when it is sent a
    # SIGHUP),
    # then
    # implement that
    # here.
    #
    start-stop-
    daemon --stop --
    signal 1 --quiet
    --pidfile
    $PIDFILE --name
    $NAME
    return 0
}

case "$1" in
    start)
        [ "$VERBOSE"
        != no ] &&
        log_daemon_msg
        "Starting $DESC"
        "$NAME"
        do_start
        case "$?" in
            0|1) [
            "$VERBOSE" != no
            ] && log_end_msg
            0 ;;
            2) [
            "$VERBOSE" != no
```



```
] && log_end_msg
1 ;;
    esac
    ;;
    stop)
    [ "$VERBOSE"
!= no ] &&
log_daemon_msg
"Stopping $DESC"
"$NAME"
    do_stop
    case "$?" in
        0|1) [
"$VERBOSE" != no
] && log_end_msg
0 ;;
        2) [
"$VERBOSE" != no
] && log_end_msg
1 ;;
    esac
    ;;
    #reload|force-
reload)
    #
    # If
do_reload() is
not implemented
then leave this
commented out
    # and leave
'force-reload' as
an alias for
'restart'.
    #
#log_daemon_msg
"Reloading $DESC"
"$NAME"
    #do_reload
    #log_end_msg
$?
    #;;
    restart|force-
reload)
    #
    # If the
"reload" option
is implemented
then remove the
    # 'force-
reload' alias
```

```
#
log_daemon_msg
"Restarting
$DESC" "$NAME"
do_stop
case "$?" in
0|1)
do_start
case "$?"
in
0)
log_end_msg 0 ;;
1)
log_end_msg 1 ;;
# Old process is
still running
*)
log_end_msg 1 ;;
# Failed to start
esac
;;
*)
#
Failed to stop
log_end_msg 1
;;
esac
;;
*)
#echo "Usage:
$SCRIPTNAME
{start|stop|resta
rt|reload|force-
reload}" >&2
echo "Usage:
$SCRIPTNAME
{start|stop|resta
rt|force-reload}"
>&2
exit 3
;;
esac
:
```



## Autres étapes

### Édition du lanceur

Ouvrez avec les droits d'administration le fichier **/etc/init.d/monprogramme** et repérez les lignes suivantes:

```
PATH=/sbin:/usr/sbin:/bin:/usr/bin
DESC="Description of the service"
NAME=daemonexecutablename
DAEMON=/usr/sbin/$NAME
DAEMON_ARGS="--options args"
PIDFILE=/var/run/$NAME.pid
SCRIPTNAME=/etc/init.d/$NAME
```

Renseignez ce qui concerne **monprogramme** :

| Variable           | Valeur                 | Commentaire                                                              |
|--------------------|------------------------|--------------------------------------------------------------------------|
| <b>DESC</b>        | <b>mon programme</b>   | description                                                              |
| <b>NAME</b>        | <b>monprogramme</b>    | nom du service dans <b>/usr/bin</b>                                      |
| <b>DAEMON</b>      | <b>/usr/bin/\$NAME</b> | monprogramme se trouve dans <b>/usr/bin</b> et non dans <b>/usr/sbin</b> |
| <b>DAEMON_ARGS</b> | <b>""</b>              | arguments pour lancer monprogramme                                       |



Ce qui donne:

```
PATH=/usr/sbin:/usr/bin:/sbin:/bin
DESC="Programme de ..."
NAME=monprogramme
DAEMON=/usr/bin/$NAME
DAEMON_ARGS=""
PIDFILE=/var/run/$NAME.pid
SCRIPTNAME=/etc/init.d/$NAME
```

Editez l'en-tête :

| le code                                                                                                                                                                                                                                                                                                                               | devient                                                                                                                                                                                                                                                                                                 |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre>### BEGIN INIT INFO # Provides: # skeleton # Required-Start: # \$remote_fs # Required-Stop: # \$remote_fs # Default-Start: 2 # 3 4 5 # Default-Stop: 0 # 1 6 # Short-Description: # Example initscript # Description: # This file should be # used to construct # scripts to be # placed in /etc/init.d. ### END INIT INFO</pre> | <pre>### BEGIN INIT INFO # Provides: # monprogramme # Required-Start: # \$remote_fs # Required-Stop: # \$remote_fs # Default-Start: 2 # 3 4 5 # Default-Stop: 0 # 1 6 # Short-Description: # Programme monprogramme # Description: # Ce fichier lance le # service monprogramme ### END INIT INFO</pre> |

Pour que le programme crée un fichier **.pid** <sup>1)</sup>, ajoutez l'option **-make-pidfile** (ou **-m**) dans les deux lignes commençant par **start-stop-daemon -start...** de la procédure **do\_start()**



Ce qui donne dans **/etc/init.d/monprogramme** :

```
#
# Function that starts the daemon/service
#
do_start()
{
    start-stop-daemon --start --quiet --m --
pidfile $PIDFILE --exec $DAEMON --test >
/dev/null \
    || return 1
    start-stop-daemon --start --quiet --m --
pidfile $PIDFILE --exec $DAEMON -- \
    $DAEMON_ARGS \
    || return 2
[... ]
}
```

Pour plus de détails, voir la page [Résumé de la page de man de start-stop-daemon](#)

### Activation

Rendez le script exécutable :

- `sudo chmod +x /etc/init.d/monprogramme`

et activez-le :

- `sudo update-rc.d monprogramme defaults`

## Autres étapes

## Conclusion

## Problèmes connus

## Voir aussi



- (en) <https://learn.adafruit.com/downloads/pdf/running-programs-automatically-on-your-tiny-computer.pdf>

---


*Contributeurs principaux : [Jamaïque](#).*

*Basé sur « [Titre original de l'article](#) » par [Auteur Original].*

<sup>1)</sup>

nécessaire pour pouvoir l'arrêter

From: <http://doc.nfrappe.fr/> - **Documentation du Dr Nicolas Frappé**

Permanent link: <http://doc.nfrappe.fr/doku.php?id=tutoriel:systeme:autoboot> 

Last update: **2022/11/08 19:41**