```
#!/bin/sh
#
# rc        This file is responsible for starting/stopping
#       services when the runlevel changes.
#
#       Optimization feature:
#       A startup script is _not_ run when the service was
#       running in the previous runlevel and it wasn't stopped
#       in the runlevel transition (most Debian services don't
#       have K?? links in rc{1,2,3,4,5} )
#
# Author:   Miquel van Smoorenburg <miquels@cistron.nl>
#       Bruce Perens <Bruce@Pixar.com>
#
# Version:  @(#)rc  2.78  07-Nov-1999  miquels@cistron.nl
#

```

更新logo进度条

```
. /etc/default/rcS
export VERBOSE

startup_progress() {
    step=$(($step + $step_change))
    if [ "$num_steps" != "0" ]; then
        progress=$((($step * $progress_size / $num_steps) + $first_step))
    else
        progress=$progress_size
    fi
    #echo "PROGRESS is $progress $runlevel $first_step + ($step of $num_steps) $step_change $progress_size"
    if type psplash-write >/dev/null 2>&1; then
        TMPDIR=/mnt/.psplash psplash-write "PROGRESS $progress" || true
    fi
    #if [ -e /mnt/.psplash/psplash_fifo ]; then
    #    echo "PROGRESS $progress" > /mnt/.psplash/psplash_fifo
    #fi
}


#
# Start script or program.
#
startup() {
  # Handle verbosity
  [ "$VERBOSE" = very ] && echo "INIT: Running $@..."

```

忽略这是三个信号

```
  case "$1" in
    *.sh)
        # Source shell script for speed.
        (
            trap - INT QUIT TSTP
            scriptname=$1
            shift
            . $scriptname
        )
        ;;
    *)
        "$@"
        ;;
  esac
  startup_progress
}

  # Ignore CTRL-C only in this shell, so we can interrupt subprocesses.
  trap ":" INT QUIT TSTP

  # Set onlcr to avoid staircase effect.
```

```
stty onlcr 0>&1

# Limit stack size for startup scripts
[ "$STACK_SIZE" == "" ] || ulimit -S -s $STACK_SIZE

# Now find out what the current and what the previous runlevel are.

runlevel=$RUNLEVEL
# Get first argument. Set new runlevel to this argument.
[ "$1" != "" ] && runlevel=$1
if [ "$runlevel" = "" ]
then
  echo "Usage: $0 <runlevel>" >&2
  exit 1
fi
previous=$PREVLEVEL
[ "$previous" = "" ] && previous=N

export runlevel previous

# Is there an rc directory for this new runlevel?
if [ -d /etc/rc$runlevel.d ]
then
  # Find out where in the progress bar the initramfs got to.
  PROGRESS_STATE=0
  #if [ -f /dev/.initramfs/progress_state ]; then
  #    . /dev/.initramfs/progress_state
  #fi

  # Split the remaining portion of the progress bar into thirds
  progress_size=$(((100 - $PROGRESS_STATE) / 3))

  case "$runlevel" in
      0|6)
          # Count down from -100 to 0 and use the entire bar
          first_step=-100
          progress_size=100
          step_change=1
          ;;
          S)
          # Begin where the initramfs left off and use 2/3
          # of the remaining space
          first_step=$PROGRESS_STATE
          progress_size=$(($progress_size * 2))
          step_change=1
          ;;
      *)
          # Begin where rcS left off and use the final 1/3 of
          # the space (by leaving progress_size unchanged)
          first_step=$(($progress_size * 2 + $PROGRESS_STATE))
          step_change=1
          ;;
  esac

  num_steps=0
  for s in /etc/rc$runlevel.d/[SK]*; do
          case "${s##/etc/rc$runlevel.d/S??}" in
              gdm|xdm|kdm|reboot|halt)
                  break
                  ;;
          esac
          num_steps=$(($num_steps + 1))
      done
      step=0

  # First, run the KILL scripts.
```

```
      if [ $previous != N ]
      then
          for i in /etc/rc$runlevel.d/K[0-9][0-9]*
          do
              # Check if the script is there.
              [ ! -f $i ] && continue

              # Stop the service.
              startup $i stop
          done
      fi

      # Now run the START scripts for this runlevel.
      for i in /etc/rc$runlevel.d/S*
      do
          [ ! -f $i ] && continue

          if [ $previous != N ] && [ $previous != S ]
          then
              #
              # Find start script in previous runlevel and
              # stop script in this runlevel.
              #
              suffix=${i#/etc/rc$runlevel.d/S[0-9][0-9]}
              stop=/etc/rc$runlevel.d/K[0-9][0-9]$suffix
              previous_start=/etc/rc$previous.d/S[0-9][0-9]$suffix
              #
              # If there is a start script in the previous level
              # and _no_ stop script in this level, we don't
              # have to re-start the service.
              #
              [ -f $previous_start ] && [ ! -f $stop ] && continue
          fi
          case "$runlevel" in
              0|6)
                  startup $i stop
                  ;;
              *)
                  startup $i start
                  ;;
          esac
      done
    fi

#Uncomment to cause psplash to exit manually, otherwise it exits when it sees a VC switch
if [ "x$runlevel" != "xS" ] && [ ! -x /etc/rc${runlevel}.d/S??xserver-nodm ]; then
    if type psplash-write >/dev/null 2>&1; then
        TMPDIR=/mnt/.psplash psplash-write "QUIT" || true
        umount -l /mnt/.psplash
    fi
fi
```