

Some simple shell

Author: Daniele Pizzolli
Contact: ors@tovel.it
Copyright: 2010 Some Rights Reserved <http://creativecommons.org/licenses/by-sa/3.0/>
Location: Villa Sant'Ignazio — Trento
Version: Draft
Note: Please send your feedback.

Let's find some short shell script to analyze

```
find /bin/ /sbin/ /usr/bin/ /usr/sbin/ /etc/ 2> /dev/null | \  
xargs file | grep 'shell script' | cut -d : -f 1 | \  
xargs -n1 wc -l | sort -nr
```

It's better to restrict ourselves to /etc

```
find /etc/ 2> /dev/null | \  
xargs file | grep 'shell script' | cut -d : -f 1 | \  
xargs -n1 wc -l | sort -nr
```

It's better to restrict ourselves to /etc

```
for i in `find /etc/ 2> /dev/null | xargs file | grep 'shell script' | \  
cut -d : -f 1 | xargs -n1 wc -l | sort -n | cut -d ' ' -f 2-`; \  
do; \  
    echo "### START $i"; \  
    cat "$i"; \  
    echo "### STOP $i"; \  
    echo \  
    echo \  
done \  
> all.sh # NOTE: output redirect
```

Interesting script in your system (1)

Look at the usage of `||` and the source `.` in `/etc/acpi/voldownbtn.sh`

```
#!/bin/sh  
  
test -f /usr/share/acpi-support/key-constants || exit 0  
  
./usr/share/acpi-support/key-constants  
acpi_fakekey $KEY_VOLUMEDOWN
```

Interesting script in your system (2)

Look in `/etc/init.d/single` for simple parameter processing:

```

#!/bin/sh
### BEGIN INIT INFO
# Provides:          single
# Required-Start:    $local_fs $all killprocs
# Required-Stop:
# Default-Start:     1
# Default-Stop:
# Short-Description: executed by init(8) upon entering runlevel 1 (single).
### END INIT INFO

PATH=/sbin:/bin

. /lib/lsb/init-functions

do_start () {
    log_action_msg "Will now switch to single-user mode"
    exec init -t1 S
}

case "$1" in
    start)
        do_start
        ;;
    restart|reload|force-reload)
        echo "Error: argument '$1' not supported" >&2
        exit 3
        ;;
    stop)
        # No-op
        ;;
    *)
        echo "Usage: $0 start|stop" >&2
        exit 3
        ;;
esac

```

Interesting script in your system (3)

Look in `/etc/acpi/thinkpad-stretchortouchpad.sh` for simple choiches:

```

#!/bin/sh

test -f /usr/share/acpi-support/key-constants || exit 0

# Lenovo rock. They have changed the function of the Fn-F8
# combination on the LenovoPads from stretching the display (in
# hardware/BIOS) to toggling the touchpad on and off.
#
# Unfortunately they didn't bother to change the DMI strings
# consistently... so some of the new machines say 'LENOVO' and some
# still say 'IBM'. Yay for consistency(!).

# So:
# IBM && !Series60 => nothing
# IBM && Series60  => Touchpad toggle

```

```

# LENOVO && ThinkPad => Touchpad toggle

toggle_touchpad=0

system_manufacturer=`dmidecode -s system-manufacturer`
case "$system_manufacturer" in
    IBM*)
        system_version=`dmidecode -s system-version`
        case "$system_version" in
            ThinkPad\ [TXZ]60*)
                toggle_touchpad=1
            ;;
            LENOVO*)
                toggle_touchpad=1
            ;;
        esac
    ;;
esac

if [ "$toggle_touchpad" -ne 0 -a -x /etc/acpi/asus-touchpad.sh ] ; then
    /etc/acpi/asus-touchpad.sh
fi

```

Interesting script in your system (4)

Look in `/etc/gdm/failsafeDexconf` for heredoc syntax, `getopt` that is different from the `getopts` builtin.

```

#!/bin/sh

# dexconf: Debian X server configuration file writer for failsafe mode
#
# This tool is a backend which uses debconf database values. It writes an
# XFree86 X server configuration file based on the information in the database.
#
# This script is derived from the dexconf program, written by
# Branden Robinson

# Copyright 2007--2008 Canonical, Ltd.
# Copyright 2000--2004 Progeny Linux Systems, Inc.
#
# This is free software; you may redistribute it and/or modify
# it under the terms of the GNU General Public License as
# published by the Free Software Foundation; either version 2,
# or (at your option) any later version.
#
# This is distributed in the hope that it will be useful, but
# WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License with
# the Debian operating system, in /usr/share/common-licenses/GPL; if
# not, write to the Free Software Foundation, Inc., 59 Temple Place,
# Suite 330, Boston, MA 02111-1307 USA

```

```

set -e

# source debconf library
. /usr/share/debconf/confmodule

# display a usage message
usage () {
    cat <<EOF
Usage: $PROGNAME [OPTION ...]
    write an Xorg X server configuration file based on debconf database values
Options:
    -h, --help                display this usage message and exit
    -o FILE, --output=FILE    write configuration file to FILE
This help message is intended only as a quick reference.  For a description of
the usage of $PROGNAME, see the $PROGNAME(1) manual page.
EOF
}

# the error-out function
bomb () {
    echo "$PROGNAME: error: $" | fold -s -w "${COLUMNS:-80}" >&2
    exit 1
}

# wrapper around db_get to ensure that the info we try to retrieve exists; it
# is (almost) always a fatal error for the values to be null
fetch () {
    db_get "$1" || true
    if [ -z "$RET" ]; then
        ERRMSG="cannot generate configuration file; $1 not set. Aborting."
        ERRMSG="$ERRMSG Reconfigure the X server with \"dpkg-reconfigure\"
        ERRMSG="$ERRMSG xserver-xorg\" to correct this problem."
        bomb "$ERRMSG"
    fi
}

# convert a debconf comma-delimited list to a shell whitespace-delimited list
list_convert () {
    echo $(IFS=", "; set -- $RET; while [ $# -gt 0 ]; do echo \"$1\"; shift; done)
}

SERVER="xorg"
XF86CONFIG=/etc/X11/xorg.conf.failsafe
PROGNAME=${0##*/}
SHOWHELP=
EARLYEXIT=

GETOPT_OUTPUT=$(getopt --options ho: \
    --longoptions help,output: \
    -n "$PROGNAME" -- "$@")

if [ $? -ne 0 ]; then
    bomb "error while getting options; use \"$PROGNAME --help\" for help"
fi

```

```

eval set -- "$GETOPT_OUTPUT"

while ;; do
  case "$1" in
    -f|--format)
      bomb "This option, and XFree86 3.x output, are no longer supported."
      ;;
    -h|--help) SHOWHELP=yes EARLYEXIT=yes ;;
    -o|--output) XF86CONFIG="$2"; shift ;;
    --) shift; break ;;
    *)
      bomb "unrecognized option \"$1\"; use \"$PROGRAMME --help\" for help"
      ;;
  esac
  shift
done

if [ -n "$SHOWHELP" ]; then
  usage
fi

if [ -n "$EARLYEXIT" ]; then
  exit 0
fi

DEXCONFTMPDIR=

trap 'if [ -e "$DEXCONFTMPDIR/backup" ] && [ -n "$XF86CONFIG" ]; then \
  cat "$DEXCONFTMPDIR/backup" >"$XF86CONFIG"; \
fi; \
exec 4<&-; \
rm -rf "$DEXCONFTMPDIR"; \
bomb "received signal; aborting"' HUP INT QUIT TERM

# Set up a temporary directory for the files we'll be writing.
TDIR_PARENT="{TMPDIR:-/tmp}"
TDIR="{TMPDIR:-/tmp}/dexconf-tmp-$$"

if [ ! -d "$TDIR_PARENT" ]; then
  bomb "cannot create temporary work directory; \"$TDIR_PARENT\" does not \"
  \"exist or is not a directory"
fi

if [ ! -w "$TDIR_PARENT" ]; then
  bomb "cannot create temporary work directory in \"$TDIR_PARENT\"; directory \"
  \"not writable"
fi

rm -rf "$TDIR"

if mkdir -m 0700 "$TDIR"; then
  DEXCONFTMPDIR="$TDIR"
else
  bomb "creation of temporary work directory \"$TDIR\" failed"
fi

```

```

# xorg.conf sections:
# Files File pathnames
# ServerFlags Server flags NOT USED BY DEXCONF
# Module Dynamic module loading NOT USED BY DEXCONF
# InputDevice Input device description
# Device Graphics device description
# VideoAdaptor Xv video adaptor description NOT USED BY DEXCONF
# Monitor Monitor description
# Modes Video modes descriptions NOT USED BY DEXCONF
# Screen Screen configuration
# ServerLayout Overall layout NOT USED BY DEXCONF
# DRI DRI-specific configuration NOT USED BY DEXCONF
# Vendor Vendor-specific configuration NOT USED BY DEXCONF

### HEADER

# Because debconf hijacks standard output and its confmodule uses file
# descriptor 3 for its own purposes, we will write our output to file descriptor
# 4 instead of standard output.

exec 4>"$DEXCONFTMPDIR/Header"
cat >&4 <<SECTION
# xorg.conf.failsafe (X.Org X Window System server configuration file)
#
# This file was generated by dexconf, the Debian X Configuration tool, using
# values from the debconf database.
#
# Edit this file with caution, and see the xorg.conf manual page.
# (Type "man xorg.conf" at the shell prompt.)
#
# This file is automatically updated on xserver-xorg package upgrades *only*
# if it has not been modified since the last upgrade of the xserver-xorg
# package.
#
# If you have edited this file but would like it to be automatically updated
# again, run the following command:
# sudo dpkg-reconfigure -phigh xserver-xorg
SECTION

### DEVICE

db_get xserver-$SERVER/config/device/driver
DEVICE_DRIVER="$RET"
db_get xserver-$SERVER/config/device/bus_id
DEVICE_BUSID="$RET"
db_get xserver-$SERVER/config/device/use_fbdev
DEVICE_USE_FBDEV="$RET"

# Override device driver
DEVICE_DRIVER=${1:-"vesa"}

QEMU_KVM=$(grep "QEMU Virtual CPU" /proc/cpuinfo || true)
if [ -n "$QEMU_KVM" ]; then
    DEVICE_DRIVER="cirrus"
fi

```

```

VBOX_VIDEO=$(grep -e "^vbox " /proc/modules || true)
if [ -n "$VBOX_VIDEO" ]; then
    DEVICE_DRIVER="vboxvideo"
fi

exec 4>"$DEXCONFTMPDIR/Device"
cat >&4 <<SECTION
Section "Device"
■Identifier■"Configured Video Device"
SECTION
if [ -n "$DEVICE_DRIVER" ]; then
    printf "\tDriver\t\t\"$DEVICE_DRIVER\"\n" >&4
fi
PS3_FB=$(grep -i PS3 /proc/fb 2>/dev/null || true)
if [ -n "$PS3_FB" ]; then
    printf "\tOption\t\t\"ShadowFB\"\t\t\"false\"\n" >&4
fi
if [ -n "$DEVICE_BUSID" ]; then
    printf "\tBusID\t\t\"$DEVICE_BUSID\"\n" >&4
fi
if [ "$DEVICE_USE_FBDEV" = "true" ]; then
    printf "\tOption\t\t\"UseFBDev\"\t\t\"$DEVICE_USE_FBDEV\"\n" >&4
fi
printf "EndSection\n" >&4

### MONITOR

exec 4>"$DEXCONFTMPDIR/Monitor"
cat >&4 <<SECTION
Section "Monitor"
■Identifier■"Configured Monitor"
SECTION

if [ -n "$QEMU_KVM" ]; then
    printf "\tHorizSync\t30-70\n" >&4
    printf "\tVertRefresh\t50-160\n" >&4
fi
cat >&4 <<SECTION
EndSection
SECTION

### SCREEN

exec 4>"$DEXCONFTMPDIR/Screen"
cat >&4 <<SECTION
Section "Screen"
■Identifier■"Default Screen"
■Monitor■■"Configured Monitor"
■Device■■"Configured Video Device"
SECTION
if [ -n "$PS3_FB" ]; then
    printf "\tDefaultFbBpp 32\n" >&4
fi
if [ -n "$QEMU_KVM" ]; then
cat >&4 <<SUBSECTION
■DefaultDepth■24

```

```

■SubSection "Display"
■Depth■24
■Modes■"1280x800" "1152x768" "1024x768" "800x600" "640x480"
■EndSubSection
SUBSECTION
fi
printf "EndSection\n" >&4

# Close file descriptor 4 before we delete temporary files
exec 4<&-

# Tell debconf to stop listening to us.
db_stop

# Write the configuration file. Put a blank line before every section we write
# except the first.

OUTFILE="$DEXCONF_TMPDIR/dexconf-out"
umask 022
: >"$OUTFILE"

SPACER=
for SECTION in Header Files InputDeviceKeyboard InputDeviceMouse \
    Device Monitor Screen; do
    if [ -e "$DEXCONF_TMPDIR/$SECTION" ]; then
        eval $SPACER
        cat "$DEXCONF_TMPDIR/$SECTION" >>"$OUTFILE"
        SPACER='echo "" >>"$OUTFILE"'
    fi
done

# Ensure we can write to our destination if it already exists.
if [ -e "$XF86CONFIG" ]; then
    if [ ! -w "$XF86CONFIG" ]; then
        bomb "unable to write to \"$XF86CONFIG\""
    fi
fi

BACKUP=
# Create a backup of the existing configuration file if it already exists.
if [ -e "$XF86CONFIG" ]; then
    cat "$XF86CONFIG" >"$DEXCONF_TMPDIR/backup"
    BACKUP=true
fi

# Move the new file into place.
if ! cat "$OUTFILE" >"$XF86CONFIG"; then
    # Failed; try to restore the backup.
    if [ -n "$BACKUP" ]; then
        cat "$DEXCONF_TMPDIR/backup" >"$XF86CONFIG"
    fi
fi

rm -rf "$DEXCONF_TMPDIR"

exit 0

# vim:set ai et sts=2 sw=2 tw=80:

```

Interesting script in your system (5)

Look for mktemp in /etc/X11/Xsession

```
#!/bin/sh
#
# /etc/X11/Xsession
#
# global Xsession file -- used by display managers and xinit (startx)

# $Id: Xsession 967 2005-12-27 07:20:55Z dnusinow $

set -e

PROGRAMME=Xsession

message () {
    # pretty-print messages of arbitrary length; use xmessage if it
    # is available and $DISPLAY is set
    MESSAGE="$PROGRAMME: $"
    echo "$MESSAGE" | fold -s -w ${COLUMNS:-80} >&2
    if [ -n "$DISPLAY" ] && which xmessage > /dev/null 2>&1; then
        echo "$MESSAGE" | fold -s -w ${COLUMNS:-80} | xmessage -center -file -
    fi
}

message_nonl () {
    # pretty-print messages of arbitrary length (no trailing newline); use
    # xmessage if it is available and $DISPLAY is set
    MESSAGE="$PROGRAMME: $"
    echo -n "$MESSAGE" | fold -s -w ${COLUMNS:-80} >&2;
    if [ -n "$DISPLAY" ] && which xmessage > /dev/null 2>&1; then
        echo -n "$MESSAGE" | fold -s -w ${COLUMNS:-80} | xmessage -center -file -
    fi
}

errormsg () {
    # exit script with error
    message "$*"
    exit 1
}

internal_errormsg () {
    # exit script with error; essentially a "THIS SHOULD NEVER HAPPEN" message
    # One big call to message() for the sake of xmessage; if we had two then
    # the user would have dismissed the error we want reported before seeing the
    # request to report it.
    errormsg "$*" \
        "Please report the installed version of the \"x11-common\" \" \
        \"package and the complete text of this error message to\" \" \
        \"<debian-x@lists.debian.org>."
}

# initialize variables for use by all session scripts

OPTIONFILE=/etc/X11/Xsession.options
```

```

SYSRESOURCES=/etc/X11/Xresources
USRRESOURCES=$HOME/.Xresources

SYSSESSIONDIR=/etc/X11/Xsession.d
USERXSESSION=$HOME/.xsession
USERXSESSIONRC=$HOME/.xsessionrc
ALTUSERXSESSION=$HOME/.Xsession
ERRFILE=$HOME/.xsession-errors

# attempt to create an error file; abort if we cannot
if (umask 077 && touch "$ERRFILE") 2> /dev/null && [ -w "$ERRFILE" ] &&
  [ ! -L "$ERRFILE" ]; then
  chmod 600 "$ERRFILE"
elif ERRFILE=$(tempfile 2> /dev/null); then
  if ! ln -sf "$ERRFILE" "${TMPDIR:=/tmp}/xsession-$USER"; then
    message "warning: unable to symlink \"${TMPDIR}/xsession-$USER\" to" \
      "\"$ERRFILE\"; look for session log/errors in" \
      "\"${TMPDIR}/xsession-$USER\"."
  fi
else
  errormsg "unable to create X session log/error file; aborting."
fi

# truncate ERRFILE if it is too big to avoid disk usage DoS
if [ "`stat -c%s \"$ERRFILE\"`" -gt 500000 ]; then
  T=`mktemp -p "$HOME"`
  tail -c 500000 "$ERRFILE" > "$T" && mv -f "$T" "$ERRFILE" || rm -f "$T"
fi

exec >>"$ERRFILE" 2>&1

echo "$PROGNAME: X session started for $LOGNAME at $(date)"

# sanity check; is our session script directory present?
if [ ! -d "$SYSSESSIONDIR" ]; then
  errormsg "no \"$SYSSESSIONDIR\" directory found; aborting."
fi

# Attempt to create a file of non-zero length in /tmp; a full filesystem can
# cause mysterious X session failures. We do not use touch, :, or test -w
# because they won't actually create a file with contents. We also let standard
# error from tempfile and echo go to the error file to aid the user in
# determining what went wrong.
WRITE_TEST=$(tempfile)
if ! echo "*" >>"$WRITE_TEST"; then
  message "warning: unable to write to ${WRITE_TEST%/*}; X session may exit" \
    "with an error"
fi
rm -f "$WRITE_TEST"

# use run-parts to source every file in the session directory; we source
# instead of executing so that the variables and functions defined above
# are available to the scripts, and so that they can pass variables to each
# other
SESSIONFILES=$(run-parts --list $SYSSESSIONDIR)

```

```
if [ -n "$SESSIONFILES" ]; then
  set +e
  for SESSIONFILE in $SESSIONFILES; do
    . $SESSIONFILE
  done
  set -e
fi

exit 0

# vim:set ai et sts=2 sw=2 tw=80:
```