

Logiciel

optimizeVideo : Remplace une vidéo par sa plus petite version sans perte de qualité perceptible

[optimizeVideo-master/install-uninstall.sh](#)

```
#!/bin/bash

: '
=== HOW TO RUN ===

For running this installer, in the application "Terminal" type:
"/pathToThisFolder/install-uninstall.sh"

For a list of programs installed like this type:
ls /etc/install-uninstall

=== PURPOSE ===

This installer is mostly for trying out the software
If you end liking it, ask someone how to package it

If your system includes the app "pacman", you can get those
packages at:
https://gitlab.com/es20490446e/express-repository/-/wikis/home

=== LEGALESE ===

Installer by Alberto Salvia Novella (es20490446e.wordpress.com)
Under the latest GNU Affero License
https://gitlab.com/es20490446e/install-uninstall.sh
'

# For testing the installer:
update="true"
simulate="false"
clean="false"

# Others:
here="$(realpath "$(dirname "${0}")")"
program="$(cd ${here}; echo ${PWD##*/})"
in="${here}/root"
```

```
etc="etc/install-uninstall"
lists="${etc}/${program}"
fileList="${lists}/files"
dirList="${lists}/dirs"

mainFuntion () {
    if [ ! -d "${out}/${lists}" ]; then
        checkDependencies
        builds
        installs
    else
        uninstalls
    fi
}

builds () {
    if [ ! -d "${in}" ] && [ -f "${here}/build.sh" ]; then
        bash "${here}/build.sh"
        chown --recursive "$(logname)" "${in}"
    fi

    if [ ! -d "${in}" ] && [ -f "${here}/build.sh" ]; then
        echo "build.sh hasn't built anything on: ${in}" >&2
        exit 1
    fi
}

checkDependencies () {
    list="${here}/info/dependencies.txt"

    if [ -f "${list}" ]; then
        readarray -t lines < <(cat "${list}")
        missing=()

        for line in "${lines[@]"; do
            name=$(echo "${line}" | cut --delimiter='"' --
fields=2)
            path=$(echo "${line}" | cut --delimiter='"' --
fields=4)
            web=$(echo "${line}" | cut --delimiter='"' --fields=6)

            if [ ! -z "${web}" ]; then
                web="(${web})"
            fi

            if ! $(ls $path &> /dev/null); then
```

```
        missing+=("${name} ${web}")
    fi
done

if [ ! -z "${missing}" ]; then
    echo "Missing required software:" >&2
    echo >&2
    printf '%s\n' "${missing[@]}" >&2
    echo >&2
    echo "Get those installed first"
    echo "and run this installer again"
    exit 1
fi
fi
}

checkPermissions () {
    if [ "${simulate}" == "false" ] && [ "$(id -u)" -ne 0 ]; then
        sudo "${0}"
        exit ${?}
    fi
}

cleanUp () {
    if [ "${clean}" == "true" ]; then
        if [ ! -z "${out}" ] && [ -d "${out}" ]; then
            rm --recursive "${out}"
        fi
    elif [ "${clean}" != "false" ]; then
        invalidVariable "clean"
    fi
}

createLists () {
    if [ ! -d "${lists}" ]; then
        mkdir --parents "${out}/${lists}"
    fi

    echo "${fileList}" > "${out}/${fileList}"
    echo "${dirList}" >> "${out}/${fileList}"
    echo "${etc}" > "${out}/${dirList}"
    echo "${lists}" >> "${out}/${dirList}"
}

dirsInFolder () {
    folder="${1}"
```

```
dirs=$(
  cd "${folder}"
  find . -type d |
  cut --delimiter="/" --fields=2-
)

echo "${dirs}" | tail -n +2
}

fileMime () {
  file="${1}"

  file --brief --mime "${file}" |
  cut --delimiter=';' --fields=1
}

fileParents () {
  file="${1}"

  echo ${file} |
  rev |
  cut --delimiter="/" --fields=2- |
  rev
}

installFile () {
  file="${1}"

  if [ "$(fileMime "${in}/${file}")" == "inode/symlink" ]; then
    installSymlink "${file}"
  else
    install -D "${in}/${file}" "${out}/${file}"
  fi
}

installSymlink () {
  symlink="${1}"
  target="$(realpath "${in}/${symlink}")"

  makeParents "${symlink}"
  ln --symbolic --force "${target}" "${out}/${symlink}"
}

installs () {
```

```
readarray -t files < <(toInstall)
createLists

for file in "${files[@]"; do
    installFile "${file}"
    echo "${file}" >> "${out}/${fileList}"
done

dirsInFolder "${in}" >> "${out}/${dirList}"
echo "installed"
}

invalidVariable () {
    variable="${1}"

    echo "The variable \"${variable}\" has an invalid value" >&2
    echo "It can either be \"true\" or \"false\""
    exit 1
}

makeParents () {
    file="${1}"
    parents="$(fileParents "${file}")"
    mkdir --parents "${out}/${parents}"
}

prepareEnvironment () {
    set -e
    updateInstaller
    setOut
    checkPermissions
    trap "" INT QUIT TERM EXIT
    cleanUp
}

setOut () {
    if [ "${simulate}" == "false" ]; then
        out=""
    elif [ "${simulate}" == "true" ]; then
        out="${here}/simulated install"
    else
        invalidVariable "simulate"
    fi
}

toInstall () {
```

```
toInstall=$(
    cd "${here}/root"
    find . -not -type d |
    cut --delimiter='/' --fields=2-
)

echo "${toInstall}"
}

uninstalls () {
    readarray -t files < <(cat "${out}/${fileList}")
    readarray -t dirss < <(cat "${out}/${dirList}")

    for file in "${files[@]"; do
        rm "${out}/${file}"
    done

    for dir in "${dirss[@]"; do
        if [ -d "${out}/${dir}" ] && [ -z "$(find "${out}/${dir}"
-not -type d)" ]; then
            rm --recursive --force "${out}/${dir}"
        fi
    done

    if [ ! -z "${out}" ] && [ -z "$(find "${out}" -not -type d)"
]; then
        rm --recursive "/${out}"
    fi

    echo "uninstalled"
}

updateInstaller () {
    if [ "${update}" == "true" ]; then
        remote="$(curl --silent
"https://gitlab.com/es20490446e/install-uninstall.sh/-/raw/master/
install-uninstall.sh")"
        local="$(cat "${0}")"

        if [ -z "${remote}" ]; then
            if [ -z "$(curl --silent google.com)" ]; then
                echo "No Internet, which is required" >&2
            else
                echo "Cannot get the updated installer" >&2
                echo "Ask developers to fix this"
            fi
        fi
    fi
}
```

```
        exit 1
    fi

    if [ "${remote}" != "${local}" ]; then
        echo "${remote}" > "${0}"
        sudo "${0}"
        exit ${?}
    fi
elif [ "${update}" != "false" ]; then
    invalidVariable "update"
fi
}

prepareEnvironment "${@}"
mainFuntion
```

Modofier comme ceci :

[optimizeVideo-master/info/dependencies.txt](#)

```
"ffmpeg" "/usr/bin/ffmpeg"
"libvpx-dev" "/usr/share/doc/libvpx-dev/copyright"
"libopus-dev" "/usr/share/doc/libopus-dev/copyright"
"args" "/usr/bin/args" "https://gitlab.com/es20490446e/args"
"solve" "/usr/bin/solve" "https://gitlab.com/es20490446e/solve"
```

Pré-requis

Installez  **ffmpeg**

```
...@...:~$ sudo apt install ffmpeg
```

Installez  **yasm**

```
...@...:~$ sudo apt install yasm
```

Installez  **doxygen**

```
...@...:~$ sudo apt install doxygen
```

Installez  **curl**

```
...@...:~$ sudo apt install curl
```

Installez **coreutils**

```
...@...:~$ sudo apt install coreutils
```

Installez **libvpx-dev**

```
...@...:~$ sudo apt install libvpx-dev
```

Installez **libopus-dev**

```
...@...:~$ sudo apt install libopus-dev
```

args :

```
...@...:~ $ cd ~/tmp
...@...:~/tmp$ curl -O
https://gitlab.com/es20490446e/args/-/archive/master/args-master.zip
...@...:~/tmp$ unzip args-master.zip
...
...@...:~/tmp$ cd args-master/
...@...:~/tmp/args-master$ ./install-uninstall.sh
installed
```

solve :

```
...@...:~ $ cd ~/tmp
...@...:~/tmp$ curl -O
https://gitlab.com/es20490446e/solve/-/archive/master/solve-master.zip
...@...:~/tmp$ unzip solve-master.zip
...
...@...:~/tmp$ cd solve-master/
...@...:~/tmp/solve-master$ ./install-uninstall.sh
installed
```

Installation

```
...@...:~/tmp$ cd optimizeVideo-master
...@...:~/tmp/optimizeVideo-master$ ./install-uninstall.sh
installed
```

Configuration

Utilisation

```
...@...:~ $ optimizeVideo mavideo.mp4
```

Désinstallation

```
...@...:~/tmp$ cd optimizeVideo-master  
...@...:~/tmp/optimizeVideo-master$ ./install-uninstall.sh  
uninstalled
```

Problèmes connus

Voir aussi

- (fr) <http://Article>

Basé sur « [Article](#) » par Auteur.

From:
<https://www.nfrappe.fr/doc-0/> - **Documentation du Dr Nicolas Frappé**

Permanent link:
<https://www.nfrappe.fr/doc-0/doku.php?id=logiciel:multimedia:video:optimizevideo:start>

Last update: **2022/08/13 22:14**

