Table des matières

Liens

http://www.megastep.org/makeself/
http://dawal.chrysalice.org/article.php3?id_article=36
http://wiki.mandriva.com/en/Development/Howto/XDGMenuSystem
Notion de bibliothèque : http://fr.wikipedia.org/wiki/Biblioth%C3%A8que_logicielle

Objectif

Dans le cadre du déploiement d'une application, réaliser un script d'installation de celle-ci sous Mandriva Linux.

Remarque : les commandes à exécuter sont indiquées en `gras`.

Exemple : bonjour à tous !

Mise en situation

L'exemple est le classique " Hello world ". Cette application a été réalisée avec l'environnement de développement QDevelop (voir Annexe 1 sur Qt).

L'application est disponible dans l'archive : **`qhelloworld-1.0.0-makeself.tar.gz`**.

Une fois l'application réalisée, pour créer cette archive, on a réalisé les étapes suivantes :

- créer le fichier **`Makefile`** en tapant : **`qmake`**
- fabriquer l'exécutable en tapant : **`make`**
- pour finir, on crée l'archive en tapant : **`tar -cf qhelloworld-1.0.0-makeself.tar ./tp1 && gzip -9f qhelloworld-1.0.0-makeself.tar`**

Déploiement

Pour réaliser une procédure d'installation d'une application, il faudra décomposer celle-ci en trois parties :

- fabriquer une version finale de l'application (make) et un script d'installation (setup.sh)
- fabriquer un installateur auto-extractible pour l'application (makeself)
- déployer cette application sur une machine à partir d'un script auto-extractible (setup)

Ces trois parties sont parfois dépendantes de la plateforme et des outils utilisés.

Fabriquer une version finale de l'application (release)

De manière générale, il faut résoudre un certain nombre de problèmes et répondre à quelques choix.

Droits ?
Les droits et les fichiers à déployer (consulter le contrat de licence de Qt) et les droits et les fichiers à appliquer à son application (indispensable de lire http://www.gnu.org/licenses/license-list.fr.html).

Statique ou dynamique ?
Il faudra choisir entre fabriquer un exécutable indépendant (statique) ou non (dynamique). Sous Linux, les dépendances sont nombreuses : architecture multi-platerforme, librairies, etc … (sous Windows, la dépendance est le plus souvent assurée par des DLLs).

Librairies dynamiques ?
Aide : http://fr.wikipedia.org/wiki/Biblioth%C3%A8que_logicielle

Les librairies dynamiques contiennent du code exécutable (des fonctions formant une API) qui sera susceptible d'être utilisé par un (ou plusieurs) programmes au moment de leur exécution. En cas de besoin, la librairie dynamique sera chargée en mémoire et son code sera alors utilisable par le programme demandeur. Il en résulte les avantages suivants : la taille du programme est réduite (puisque le code dont il a besoin se trouve dans la librairie), la possibilité de faire évoluer la librairie sans avoir à recompiler (si le prototype des fonctions définies dans la librairie reste inchangé). L'inconvénient majeur reste l'obligation de la présence de la librairie sur le système cible pour que le programme puisse s'exécuter.

L'extension usuelle d'une librairie dynamique sous Linux est .so (pour Windows, ce sera .dll)

Dépendances ?
Il y a plusieurs techniques pour connaître les dépendances externes d'une application :

- utiliser un utilitaire qui recherche ces dépendances (la commande ldd sous Linux, la commande rpmbuild établit aussi une liste de dépendances de paquetages nécessaires, Dependency Walker sous Windows, etc.)

- appliquer une démarche (rudimentaire !) qui teste l'application sur une machine vierge et qui résout les dépendances les unes après les autres.

Les fichiers de l'application à déployer ?
Il faut évidemment recenser les fichiers (ainsi que l'arborescence) qui composent l'application et qui devront être déployer avec celle-ci. De manière générale, on trouve :

- l'exécutable (.exe sous Windows)
- l'icône (.ico)
- des fichiers de configuration (comme les .ini)
- des fichier multimédia (comme des images)
- des fichiers d'aide (comme les .chm ou les pages man, des fichiers .html, ...)
- un fichier licence, un fichier readme, ...
- etc ...

Exemple :

```
./setup.sh
./AUTHORS
./README
./NEWS
./ChangeLog
./INSTALL
./COPYING
./TODO
./usr/share/man/man1/qhelloworld.1.lzma
./usr/bin/
./usr/bin/qhelloworld
```

On désire regrouper tous ces fichiers dans un même et seul fichier exécutable : l'installateur.

Fabriquer un installateur auto-extractible pour l'application (makeself)

Pour le déploiement de l'application, notre choix se porte sur l'utilitaire **makeself**.

Makeself est un petit script shell qui génère une archive auto-extractible (tar) d'un répertoire. Le résultat apparait comme un script shell exécutable. L'archive va ensuite s'auto-décompresser et lancer un installateur de votre choix (un autre script par exemple). Cela est assez similaire aux archive Winzip auto extractibles. Il gère aussi l'intégrité via un checksum CRC et/ou MD5 (les deux par défaut).

A ce jour, ce script est utilisé par les entreprises suivantes :
- Id Software (Editeur de jeux vidéo : Quake 3, Return To Castle Wolfenstien) .
- Loki Software pour les jeux qu?il édite.
- Les drivers pour Linux du constructeur nVidia.
- L'installateur de Google Earth pour Linux
- Le package Makeself lui-même.

Lien : http://www.megastep.org/makeself/

Un **fichier auto-extractible**, également connu sous le sigle SFX (*self-extracting archive*), est un type de fichier informatique compressé qui contient en lui-même les outils nécessaires à sa propre décompression, de sorte que contrairement aux autres fichiers compressés, il n'est pas nécessaire de disposer d'un logiciel de compression pour accéder aux données qu'il contient. Il s'agit d'un fichier exécutable qui contient la charge utile. Il suffit généralement d'en changer l'extension (*.exe sous Windows ou .bin sous Linux) et de double-cliquer dessus pour que la décompression s'opère. Naturellement il faut que ce fichier exécutable soit exécutable sous le système d'exploitation considéré.

Remarque : Sécurité
Distribuer des archives sous forme auto-extractible peut rebuter certains utilisateurs, car de tels fichiers, en tant qu'exécutables, pourraient contenir du code malicieux tel que des virus ou des chevaux de Troie, si bien que des fichiers compressés de manière habituelle, sans code exécutable, leur sont parfois préférés. Cependant, de nombreux logiciels de compression sont capables d'ouvrir des fichiers auto-extractibles comme s'il s'agissait d'archives habituelles.

La fabrication d'un script auto-extractible est décrite dans le chapitre suivant.

Déployer une application sur une machine (setup)

Pour installer l'application (sous root) :
```
# ./setup.sh
Verifying archive integrity... All good.
Uncompressing Script d'installation pour Mandriva Linux (bts iris) by tv .....
...
```

Pour désinstaller l'application (sous root) :
```
# uninstall-qhelloworld-1.0.0.sh
```

Pour obtenir des infos sur le paquetage, exécuter le script :
```
$ ./setup.sh --info
```

Fabriquer un installateur auto-extractible avec makeself

Installation requise

Vous devez avoir installé au préalable l'archive **makeself.run**.

```
# ./makeself.run
```

Vous pouvez installer, par exemple, **makeself** dans le répertoire **/usr/local/**.

Contenu

On désire créer un fichier auto-extractible **setup.sh** (ou install.sh) contenant :

- les fichiers de l'application :
  ```
  ./tp1/AUTHORS
  ./tp1/README
  ./tp1/NEWS
  ./tp1/ChangeLog
  ./tp1/INSTALL
  ./tp1/COPYING
  ./tp1/TODO
  ./tp1/usr/share/man/man1/qhelloworld.1.lzma
  ./tp1/usr/bin/qhelloworld
  ```

- le script d'installation à exécuter au lancement :
  ```
  ./tp1/setup.sh
  ```

Vous pouvez récupérer ces fichiers dans l'archive et la désarchiver avec la commande :

```
$ tar zxvf qhelloworld-1.0.0-makeself.tar.gz
```

Configuration de l'installation pour Mandriva Linux

Pour pouvoir installer correctement l'application, il suffit d'éditer le script d'installation **setup.sh** (voir Annexe 3) et de paramétrer les variables à sa convenance.

```
$ vim setup.sh
### Informations sur l'installation
NAME="qhelloworld"
VERSION="1.0.0"
SUMMARY="Affiche \"Hello world !\" dans une boite de dialogue"
LICENCE="GPL"
AUTEUR="Thierry Vaira <thierry.vaira@orange.fr>"

HAVE_MENU_KDE="yes"
ICON="qdevelop"
CATEGORY="Qt;Development;IDE;"
MIMETYPE="application/x-qdevelop;"
#####################################
```

```
### liste des ressources à installer
BIN_FILES="${NAME}"
DOC_FILES="README NEWS COPYING AUTHORS TODO ChangeLog"
MAN_FILES="man1/*"
MAN_FILES="man1/qhelloworld.1.*"
EXTRA_FILES=""
####################################

### repertoires
BINDIR="usr/bin"
DATADIR="usr/share"
DOCDIR="${DATADIR}/doc/${NAME}"
MANDIR="${DATADIR}/man"
EXTRADIR="${DATADIR}/${NAME}"
TMPDIR="/tmp"
```

On peut tester manuellement avant de construire l'installateur :

```
# ./tp1/setup.sh
```

Construire l'installateur

Pour construire l'installateur, il suffit de taper une seule commande :

```
# ./makeself.sh ./tp1 setup.sh "Script d'installation pour Mandriva
Linux (bts iris) by tv" ./setup.sh
```

Les arguments passés ici à makeself sont :

- **`./tp1`** : le répertoire qui contient les fichiers à « empaqueter » (qui seront ajoutés à l'archive)
- **`setup.sh`** : le nom de l'installateur auto-extractible (le nom de l'archive)
- **`"blabla"`** : un label d'identification
- **`./setup.sh`** : le nom du script qui sera automatiquement exécuté au lancement

On peut aussi consulter l'aide en ligne de makeself :

```
Usage: ./makeself.sh [params] archive_dir file_name label [startup_script] [args]
...
```

Lire : http://www.megastep.org/makeself/

Installer

Pour installer le paquetage (sous **root**) :

```
# ./setup.sh
Verifying archive integrity... All good.
Uncompressing Script d'installation pour Mandriva Linux (bts iris) by tv ..........
```

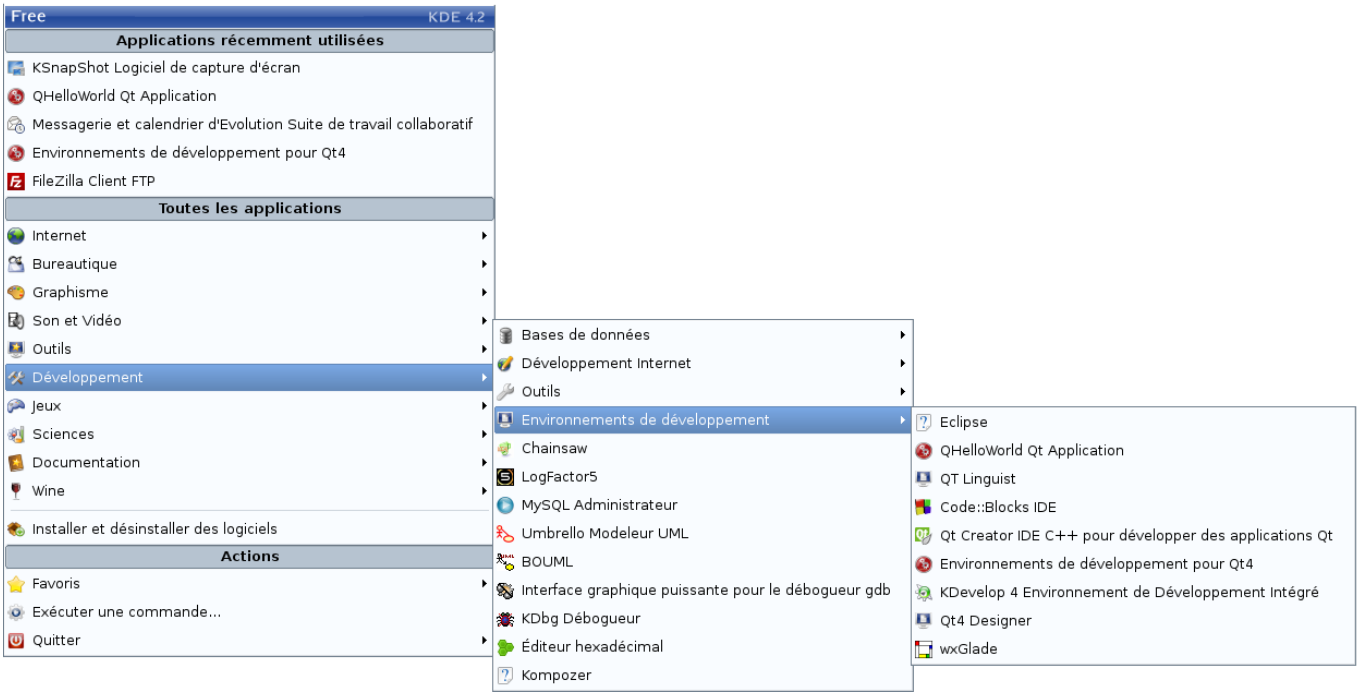
Le script d'installation prend ensuite le relais pour assurer l'installation …

On peut tester :

```
$ qhelloworld

$ man qhelloworld
```

On a même une entrée dans les menus KDE :



Puis, pour désinstaller (sous **root**) :

```
# uninstall-qhelloworld-1.0.0.sh
```

Et bien évidemment, cela ne marche plus :

```
$ qhelloworld
bash: /usr/bin/qhelloworld: Aucun fichier ou dossier de ce type

$ man qhelloworld
Il n'y a pas de page de manuel pour qhelloworld.
```

Tests

Avant d'installer l'application, on va récupérer des informations sur l'installateur.

On exécute le script avec les options suivantes :

```
# ./setup.sh --info
Identification: Script d'installation pour Mandriva Linux (bts iris) by tv
Target directory: tp1
Uncompressed size: 484 KB
Compression: gzip
Date of packaging: Sat Apr 17 14:27:04 CEST 2010
Built with Makeself version 2.1.5 on linux-gnu
Build command was: ./makeself.sh \
    "./tp1" \
    "setup.sh" \
    "Script d'installation pour Mandriva Linux (bts iris) by tv" \
    "./setup.sh"
Script run after extraction:
     ./setup.sh
tp1 will be removed after extraction

# ./setup.sh --list
Target directory: tp1
drwxr-xr-x root/root         0 2010-04-17 14:26 ./
-rwxr-xr-x root/root     14849 2010-04-17 14:24 ./setup.sh
-rw-r--r-- root/root        54 2010-04-17 14:24 ./AUTHORS
-rw-r--r-- root/root        20 2010-04-17 14:24 ./README
-rw-r--r-- root/root         0 2010-04-17 14:24 ./NEWS
-rw-r--r-- root/root       266 2010-04-17 14:24 ./ChangeLog
-rw-r--r-- root/root      9240 2010-04-17 14:24 ./INSTALL
-rw-r--r-- root/root     17992 2010-04-17 14:24 ./COPYING
-rw-r--r-- root/root        28 2010-04-17 14:24 ./TODO
drwxr-xr-x root/root         0 2010-04-17 14:24 ./usr/
drwxr-xr-x root/root         0 2010-04-17 14:24 ./usr/share/
drwxr-xr-x root/root         0 2010-04-17 14:24 ./usr/share/man/
drwxr-xr-x root/root         0 2010-04-17 14:24 ./usr/share/man/man1/
-rw-r--r-- root/root       489 2010-04-17 14:24
./usr/share/man/man1/qhelloworld.1.lzma
drwxr-xr-x root/root         0 2010-04-17 14:24 ./usr/bin/
-rwxr-xr-x root/root    387314 2010-04-17 14:24 ./usr/bin/qhelloworld

# ./setup.sh --check
Verifying archive integrity... MD5 checksums are OK. All good.
```

Annexe 1 : Qt

Qt est une bibliothèque logicielle orientée objet et développée en C++ par Qt Development Frameworks, filiale de Nokia. Elle offre des composants d'interface graphique (*widgets*), d'accès aux données, de connexions réseaux, de gestion des fils d'exécution, d'analyse XML, etc.

Qt permet la portabilité des applications qui n'utilisent que ses composants par simple recompilation du code source. Les environnements supportés sont les Unix (dont Linux) qui utilisent le système graphique X Window System, Windows et Mac OS X.

Qt est notamment connu pour être la bibliothèque sur laquelle repose l'environnement graphique KDE, l'un des environnements de bureau les plus utilisés dans le monde Linux.

Historique
Quasar Technologies est créé le 4 mars 1994 et renommé six mois plus tard en Troll Tech, puis Trolltech, puis Qt Software et enfin Qt Development Frameworks. Le 28 janvier 2008, Nokia lance une OPA amicale pour racheter Qt et Trolltech. Trolltech, renommé en Qt Software, devient une division de Nokia.

Il existe une version pour les systèmes embarqués, Qt/Embedded, connue depuis sous le nom de Qtopia, et publiée pour la première fois en 2000.

Depuis, Qt4 sépare la bibliothèque en modules :

- QtCore : pour les fonctionnalités non graphiques utilisées par les autres modules ;
- QtGui : pour les composants graphiques ;
- QtNetwork : pour la programmation réseau ;
- QtOpenGL : pour l'utilisation d'OpenGL ;
- QtSql : pour l'utilisation de base de données SQL ;
- QtXml : pour la manipulation et la génération de fichiers XML ;
- QtDesigner : pour étendre les fonctionnalités de Qt Designer, l'assistant de création d'interfaces graphiques ;
- QtAssistant : pour l'utilisation de l'aide de Qt [6];
- Qt3Support : pour assurer la compatibilité avec Qt 3.
- et de nombreux autres modules, etc.

Licences
Le projet d'environnement graphique KDE a dès le début utilisé la bibliothèque Qt. Mais avec le succès de cet environnement, une certaine partie de la communauté du logiciel libre a critiqué la licence de Qt qui était propriétaire et incompatible avec la GNU GPL utilisée par KDE. Ce problème fut résolu par la société Trolltech qui mit la version Unix/Linux de Qt sous licence GNU GPL lorsque l'application développée était également sous GNU GPL. Pour le reste, c'est la licence commerciale qui entre en application. Cette politique de double licence a été appliquée uniquement pour Unix dans un premier temps, mais depuis la version 4.0 de Qt, elle est appliquée pour tous les systèmes.
Le 14 janvier 2009, Trolltech annonce qu'à partir de Qt 4.5, Qt sera également disponible sous licence LGPL v2.1[12]. Cette nouvelle licence permet ainsi des développements de logiciels propriétaires, sans nécessiter l'achat d'une licence commerciale auprès de Qt Development Frameworks. Ce changement, voulu par Nokia pour faire en sorte que Qt soit utilisé par un maximum de projets, est rendu possible par le fait que Nokia peut se passer des ventes des licences commerciales, contrairement à Trolltech qui ne pouvait pas se passer de cette source de revenus.

Qt Designer est un logiciel qui permet de créer des interfaces graphiques Qt dans un environnement convivial. L'utilisateur, par glisser-déposer, place les composants d'interface graphique et y règle leurs propriétés facilement. Les fichiers d'interface graphique sont formatés en XML et portent l'extension .ui. Lors de la compilation, un fichier d'interface graphique est converti en classe C++ par l'utilitaire uic.

Environnement de développement intégré (EDI ou IDE)
Qt Creator est l'environnement de développement intégré dédié à Qt et facilite la gestion d'un projet Qt. Son éditeur de texte offre les principales fonctions que sont la coloration syntaxique, le complètement, l'indentation, etc... Qt Creator intègre en son sein les outils Qt Designer et Qt Assistant ;
Même si Qt Creator est présenté comme l'environnement de développement de référence pour Qt, il existe des modules Qt pour les environnements de développement Eclipse et Visual Studio. Il existe d'autres EDI dédiés à Qt et développés indépendamment de Nokia, comme QDevelop et Monkey Studio.

QDevelop est un environnement de développement intégré libre pour Qt. Le but de QDevelop est de fournir dans les environnements les plus utilisés, Linux, Windows et Mac OS X d'un outil permettant de développer en Qt de la même manière avec un IDE unique. Il intègre également les outils Qt-Designer pour la création d'interface graphique et Qt-Linguist pour le support de l'internationalisation.

KDevelop est un environnement de développement intégré (IDE) pour KDE. Il intègre également les outils Qt-Designer pour la création d'interface graphique et Qt-Linguist pour la gestion de l'internationalisation.

Autres bibliothèques généralistes multi-plateformes
Parmi les plus connus :

- **GTK+**, utilisée par l'environnement graphique GNOME
- **wxWidgets** (anciennement wxWindows)

Conclusion
De plus en plus de développeurs utilisent Qt, y compris parmi de grandes entreprises. On peut notamment citer : Google, Adobe Systems, Asus, Samsung, Philips, ou encore la NASA et bien évidemment Nokia.

Annexe 2 : Qmake

Qt se voulant un environnement de développement portable et ayant le MOC comme étape intermédiaire avant la phase de compilation/édition de liens, il a été nécessaire de concevoir un moteur de production spécifique. C'est ainsi qu'est conçu le programme qmake.

Ce dernier prend en entrée un fichier (avec l'extension .pro) décrivant le projet (liste des fichiers sources, dépendances, paramètres passés au compilateur, etc...) et génère un fichier de projet spécifique à la plateforme. Ainsi, sous les systèmes UNIX qmake produit un Makefile qui contient la liste des commandes à exécuter pour génération d'un exécutable, à l'exception des étapes spécifiques à Qt (génération des classes C++ lors de la conception d'interface graphique avec Qt Designer, génération du code C++ pour lier les signaux et les slots, ajout d'un fichier au projet, etc.).

Le fichier de projet est fait pour être très facilement éditable par un développeur. Il consiste en une série d'affectations de variables.

Manuel : http://doc.trolltech.com/4.3/qmake-manual.html

Annexe 3 : le script setup.sh

```
#!/bin/bash
# Copyleft (C) 2010 by <tv>

### Informations sur l'installation
NAME="qhelloworld"
VERSION="1.0.0"
SUMMARY="Affiche \"Hello world !\" dans une boite de dialogue"
LICENCE="GPL"
AUTEUR="Thierry Vaira <thierry.vaira@orange.fr>"

HAVE_MENU_KDE="yes"
ICON="qdevelop"
CATEGORY="Qt;Development;IDE;"
MIMETYPE="application/x-qdevelop;"
####################################

### liste des ressources à installer
BIN_FILES="${NAME}"
DOC_FILES="README NEWS COPYING AUTHORS TODO ChangeLog"
MAN_FILES="man1/*"
MAN_FILES="man1/qhelloworld.1.*"
EXTRA_FILES=""
####################################

### repertoires
BINDIR="usr/bin"
DATADIR="usr/share"
DOCDIR="${DATADIR}/doc/${NAME}"
MANDIR="${DATADIR}/man"
EXTRADIR="${DATADIR}/${NAME}"
TMPDIR="/tmp"
####################################

### outils
CAT="cat"
SED="sed"
INSTALL_FILE="install -m 644"
INSTALL_PROGRAM="install -m 755"
DEL_FILE="rm -f"
DEL_DIR="rmdir"
CHK_DIR_EXISTS="test -d"
CHK_FILE_EXISTS="test -f"
MKDIR="mkdir -p"
DIRMODE="755"
MKTEMP="mktemp"
MKTEMP_FILE="mktemp -q"
####################################

function interrupt_quit()
{
   $DEL_FILE $TMP_FILE > /dev/null 2>&1
      #echo -e "\nInstallation annulee !"
      #exit 1
}
```

```
function init()
{
   # temp file
   # mktemp installé ?
   if [ -x "`which ${MKTEMP} 2>&-`" ]
   then
      TMP_FILE="$(${MKTEMP_FILE} ${TMPDIR}/$0.XXXXXX)" || exit 1
   else
      TMP_FILE="${TMPDIR}/$(basename $0).$$"
      ! $CHK_FILE_EXISTS $TMP_FILE && $TOUCH $TMP_FILE > /dev/null  2>&1 || exit 1
   fi

   distribution=`${CAT} /etc/release 2> /dev/null | tr "[:upper:]" "[:lower:]" 2>
/dev/null`

   # by Bjarni R. Einarsson <bre@netverjar.is>
   GUESS_XTERMS="xterm dtterm rxvt kvt eterm konsole"
   #PREFER_X=yes
   PREFER_TTY=yes
   for a in $GUESS_XTERMS; do
      if which $a 2>/dev/null >/dev/null; then
         XTERM=$a
      fi
   done
   HAVE_TTY=
   INTERFACE=null
   if stty >/dev/null 2>/dev/null; then
      HAVE_TTY=1
      INTERFACE=tty
   fi
   if [ "$DISPLAY" != "" ]; then
      if [ "$HAVE_TTY" = "" -a "$XTERM_LOOP" != "$PREFER_TTY" ]; then
         export XTERM_LOOP="$PREFER_TTY"
         exec $XTERM -title $(basename $0) -e "$0" ${@+"$@"}
      fi
      [ "$PREFER_X" != "" -o "$HAVE_TTY" = "" ] && INTERFACE=X
   fi
   XTERM_LOOP=
   export HAVE_TTY XTERM_LOOP
   case $INTERFACE in
      X)
         if [ "$DISPLAY" = "" ]; then
            echo "Interface graphique non disponible (variable DISPLAY vide) !"
            exit 1
         fi
         if !which xmessage 2>/dev/null >/dev/null; then
            echo "Le programme xmessage n'est pas installé !"
            exit 1
         fi
         CAT="xmessage -buttons okay:0 -title setup.sh -center -default okay -file -"
         CLEARSCR=true
         ;;
      tty)
         CAT=cat
         CLEARSCR=clear
         ;;
      null)
         CAT=cat
         ;;
   esac
}
```

```
function query()
{
   QUESTION=$1
   DEFAULT=$2

   case $INTERFACE in
      tty)
         echo -n "$QUESTION"
         if [ 0 -eq $DEFAULT ]; then
            echo -n " [O/n/q] "
         else
            echo -n " [o/N/q] "
         fi
         read -s -n 1 R <&1
         echo

         [ "$R" = "O" -o "$R" = "o" ] && return 0
         [ "$R" = "N" -o "$R" = "n" ] && return 1
         [ "$R" = "Q" -o "$R" = "q" ] && echo -e "\x1B[0m" && exit 1
         return $DEFAULT
         ;;
      X)
         if [ 0 -eq $DEFAULT ]; then
            DL=oui
         else
            DL=non
         fi
         xmessage \
            -title $(basename $0) -center \
            -buttons "oui:0,non:1,quitter:2" \
            -default $DL "$QUESTION"

         R=$?
         [ $R -eq 2 ] && exit 0
         return $R
         ;;
      *)
         if [ 0 -eq $DEFAULT ]; then
            DEF=oui
         else
            DEF=non
         fi
         echo "$QUESTION ($DEF !)" | cat
         return $DEFAULT
         ;;
   esac
}

function root_check()
{
   # seul root peut executer ce script :)
   #id | grep "uid=0(root)" > /dev/null  2>&1
   #if [ $? != "0" ]
   #then
   #   echo -e "\x1B[49;31;1mERREUR: ce script ne peut etre execute que sous le
compte root !\x1B[0m"
   #   echo "";
   #   exit 1;
   #fi

   #$CLEARSCR

   echo -e "\x1B[49;31;1m"
   WARNING="\
```

```
WARNING:
    Installing programs as root can seriously damage your system!
    Unless you obtained it from a trusted source, you run the risk
    of installing trojan horses or virii - not to mention the fact
    that the installation sequence itself might inadvertantly harm
    your system.

    Installing as a normal user and examining the program (perhaps
    later re-installing it as root) is recommended for most
    applications."

   id | grep "uid=0(root)" > /dev/null  2>&1
   if [ $? != "0" ]; then
   #if [ "$UID" != "0" ]; then

      QUESTION="
You are not running $(basename $0) as root (the system administrator),
which means you cannot perform a system-wide installation of this
program.

At this point you may (if you know the root password) complete the
installation as root, or just continue with your normal privileges
and install the program in your home directory.

$WARNING

Continue as root ?"

      if query "$QUESTION" 1; then
         if [ "$HAVE_TTY" = "" -a "$DISPLAY" != "" ]; then
            : exec $XTERM -title $(basename $0) -e su -c "$0 -i $INTERFACE --asroot"
         else
            exec su -c "$0 -i $INTERFACE --asroot"
         fi
      fi
      echo -e "\x1B[0m"
      echo "";
      exit 1;
      #ou on continue ? TODO
      export INSTALL_IN_HOME=yes
   else
      QUESTION="
$WARNING

Continue as root ?"

      if ! query "$QUESTION" 1; then
         echo -e "\x1B[0m"
         exit 0
      fi
   fi
   echo -e "\x1B[0m"
   return 0
}

function say_hello()
{
   $CLEARSCR

   echo -e "\x1B[49;34;1m
======================================================
Copyleft (C) 2010 <tv>

Script d'installation pour Mandriva Linux (bts iris)
```

```
=====================================================\x1B[0m\n"

   echo -n "Vous utilisez une distribution "
   echo -e "\x1B[49;34;1m$distribution\x1B[0m\n"

   MESSAGE="Vous etes sur le point d'installer

Nom: $NAME
Version: $VERSION
Description: $SUMMARY
Auteur: $AUTEUR
Licence: $LICENCE

Continuer ?"

   if ! query "$MESSAGE" 0; then
      exit 1
   fi
}

function verif_size()
{
   DESTDIR="$1"
   shift
   BASEDIR="$1"
   shift
   basedir="${BASEDIR-pwd}"
   totalSizeKB=0
   for file in ${*}
   do
        fileSizeKB=0
        name=$(echo $basedir/$file | $SED -e 's/[^[:alnum:]\.\_\-\/]//g')
        $CHK_FILE_EXISTS $name && fileSizeKB=`ls -l $name | awk '{print $(NF-3)}' |
tail -1`
        totalSizeKB=`expr $totalSizeKB + $fileSizeKB`
   done
   totalSizeKB=`expr $totalSizeKB / 1024`

   echo "Espace total requis : $totalSizeKB kB"
   echo -n "Espace disponible   : "
   freespace=`df -k $DESTDIR | tail -1 | awk '{print $(NF-2)}'`
   echo "$freespace kB"
   if [ $totalSizeKB -gt $freespace ] ; then
      return 1
   fi
   return 0
}

function install_dir()
{
   # from mkinstalldirs by Noah Friedman <friedman@prep.ai.mit.edu>
   dirmode=$1
   shift
   for f
   do
      if test -d "$f"; then
         shift
      else
         break
      fi
   done
   case $# in
   0) return 0 ;;
   esac
```

```
   case $dirmode in
  '')
    if $MKDIR --version . >/dev/null 2>&1 && test ! -d ./--version; then
      #echo "${MKDIR} -- $*"
      $MKDIR -- "$@"
    else
      # On NextStep and OpenStep, the `mkdir' command does not
      # recognize any option.  It will interpret all options as
      # directories to create, and then abort because `.' already
      # exists.
      test -d ./-p && $DEL_DIR ./-p
      test -d ./--version && $DEL_DIR ./--version
    fi
    ;;
  *)
    if mkdir -m "$dirmode" -p --version . >/dev/null 2>&1 &&
       test ! -d ./--version; then
      #echo "mkdir -m $dirmode -p -- $*"
      mkdir -m "$dirmode" -p -- "$@"
    else
      # Clean up after NextStep and OpenStep mkdir.
      for d in ./-m ./-p ./--version "./$dirmode";
      do
        test -d $d && $DEL_DIR $d
      done
    fi
    ;;
   esac
      echo ""
   return 0
}

function install_file()
{
   echo "Etape $1 : Installation des fichiers"
   # installation : bin
   if [ x"$BIN_FILES" != x ]; then
      verif_size "/${BINDIR}" "${BINDIR}" $BIN_FILES
      if [ $? -eq 1 ] ; then
         echo -e "\x1B[49;31;1mErreur : manque d'espace disque !\x1B[0m"
         return 1
      fi
      echo ""
      install_dir $DIRMODE "/${BINDIR}"
      if [ $? -eq 1 ] ; then
         echo -e "\x1B[60G\x1B[49;31;1mIErreur : impossible de créer le répertoire "/
${BINDIR}"\x1B[0m"
         return 1
      fi
      for file in $BIN_FILES
         do
         echo -n "Installation du fichier $file dans /${BINDIR}"
         INSTALL_CMD="${INSTALL_PROGRAM} "${BINDIR}/$file" "/${BINDIR}/$file""
         $CHK_DIR_EXISTS "/${BINDIR}" && $INSTALL_CMD > /dev/null 2>&1
         $CHK_FILE_EXISTS "/${BINDIR}/$file" || echo -e
"\x1B[60G\x1B[49;31;1m[ERREUR]\x1B[0m"
         $CHK_FILE_EXISTS "/${BINDIR}/$file" && echo -e
"\x1B[60G\t\t\x1B[49;32;1m[OK]\x1B[0m"
         $CHK_FILE_EXISTS "/${BINDIR}/$file" && echo "$DEL_FILE /${BINDIR}/$file" >>
$TMP_FILE
      done
      echo ""
   fi
```

```
   # installation : doc
   if [ x"$DOC_FILES" != x ]; then
      install_dir $DIRMODE "/${DOCDIR}"
      if [ $? -eq 1 ] ; then
         echo -e "\x1B[60G\x1B[49;31;1mIErreur : impossible de créer le répertoire "/
${DOCDIR}"\x1B[0m"
         return 1
      fi
      verif_size "/${DOCDIR}" "." $DOC_FILES
      if [ $? -eq 1 ] ; then
         echo -e "\x1B[49;31;1mErreur : manque d'espace disque !\x1B[0m"
         return 1
      fi
      echo ""
      for file in $DOC_FILES
         do
         echo -n "Installation du fichier $file dans /${DOCDIR}"
         INSTALL_CMD="${INSTALL_FILE} $file "/${DOCDIR}/${file}""
         $CHK_DIR_EXISTS "/${DOCDIR}" && $INSTALL_CMD > /dev/null 2>&1
         $CHK_FILE_EXISTS "/${DOCDIR}/$file" || echo -e
"\x1B[60G\x1B[49;31;1m[ERREUR]\x1B[0m"
         $CHK_FILE_EXISTS "/${DOCDIR}/$file" && echo -e
"\x1B[60G\t\t\x1B[49;32;1m[OK]\x1B[0m"
         $CHK_FILE_EXISTS "/${DOCDIR}/$file" && echo "$DEL_FILE /${DOCDIR}/$file" >>
$TMP_FILE
      done
      $CHK_DIR_EXISTS "/${DOCDIR}" && echo "$DEL_DIR /${DOCDIR}" >> $TMP_FILE
      echo ""
   fi
   # installation : man
   if [ x"$MAN_FILES" != x ]; then
      verif_size "/${MANDIR}" "${MANDIR}" $MAN_FILES
      if [ $? -eq 1 ] ; then
         echo -e "\x1B[49;31;1mErreur : manque d'espace disque !\x1B[0m"
         return 1
      fi
      echo ""
      install_dir $DIRMODE "/${MANDIR}"
      if [ $? -eq 1 ] ; then
         echo -e "\x1B[60G\x1B[49;31;1mIErreur : impossible de créer le répertoire "/
${MANDIR}"\x1B[0m"
         return 1
      fi
      for file in "${MANDIR}/${MAN_FILES}"
         do
         echo -n "Installation des pages man dans /${MANDIR}"
         for f in $file
         do
            dstdir=`echo $f | $SED -e 's,[^/]*$,,;s,/$,,;s,^$,.,'`
         done
         INSTALL_CMD="${INSTALL_FILE} ${file} "/${dstdir}""
         $CHK_DIR_EXISTS "/${dstdir}" && $INSTALL_CMD > /dev/null 2>&1
         if [ $? -eq 1 ] ; then
            echo -e "\x1B[60G\x1B[49;31;1m[ERREUR]\x1B[0m"
         else
            echo -e "\x1B[60G\t\t\x1B[49;32;1m[OK]\x1B[0m"
            echo "$DEL_FILE /${file}" >> $TMP_FILE
         fi
      done
      echo ""
   fi
```

```
   # installation : extra
   if [ x"$EXTRA_FILES" != x ]; then
      verif_size "/${EXTRADIR}" "${EXTRADIR}" $EXTRA_FILES
      if [ $? -eq 1 ] ; then
         echo -e "\x1B[49;31;1mErreur : manque d'espace disque !\x1B[0m"
         return 1
      fi
      echo ""
      install_dir $DIRMODE "/${EXTRADIR}"
      if [ $? -eq 1 ] ; then
         echo -e "\x1B[60G\x1B[49;31;1mIErreur : impossible de créer le répertoire "/
${EXTRADIR}"\x1B[0m"
         return 1
      fi
      for file in $EXTRA_FILES
         do
         echo -n "Installation du fichier $file dans /${EXTRADIR}"
         INSTALL_CMD="${INSTALL_FILE} $file "/${EXTRADIR}/${file}""
         CHK_DIR_EXISTS "/${EXTRADIR}" && $INSTALL_CMD > /dev/null 2>&1
         $CHK_FILE_EXISTS "/${EXTRADIR}/$file" || echo -e
"\x1B[60G\x1B[49;31;1m[ERREUR]\x1B[0m"
         $CHK_FILE_EXISTS "/${EXTRADIR}/$file" && echo -e
"\x1B[60G\t\t\x1B[49;32;1m[OK]\x1B[0m"
         $CHK_FILE_EXISTS "/${EXTRADIR}/$file" && echo "$DEL_FILE /${EXTRADIR}/$file"
>> $TMP_FILE
      done
      echo ""
   fi

   return 0
}

function install_menu()
{
   if [ x"$HAVE_MENU_KDE" != x ]; then
         echo -n "Etape $1 : Installation du menu dans KDE"
         $CHK_DIR_EXISTS "./${DATADIR}/applications" || $MKDIR "./$
{DATADIR}/applications" > /dev/null 2>&1
      cat > "./${DATADIR}/applications/mandriva-${NAME}.desktop" <<EOF
[Desktop Entry]
Encoding=UTF-8
Name=${NAME}
GenericName=Application
Comment=${SUMMARY}
Exec=/${BINDIR}/${NAME}
Icon=${ICON}
Terminal=false
Type=Application
StartupNotify=true
MimeType=${MIMETYPE}
Categories=${CATEGORY}
EOF

   $CHK_FILE_EXISTS "./${DATADIR}/applications/mandriva-${NAME}.desktop" && desktop-
file-install \
--add-category="${CATEGORY}" \
--add-mime-type="${MIMETYPE}" \
--add-only-show-in="KDE" \
--dir /${DATADIR}/applications ./${DATADIR}/applications/* > /dev/null 2>&1
      if [ $? -eq 1 ] ; then
         echo -e "\x1B[60G\x1B[49;31;1m[ERREUR]\x1B[0m"
      else
         echo -e "\x1B[60G\t\t\x1B[49;32;1m[OK]\x1B[0m"
         echo "$DEL_FILE /${DATADIR}/applications/mandriva-${NAME}.desktop" >>
```

```
$TMP_FILE
      fi
   fi
   return 0
}

function create_uninstaller()
{
   UNINST_PATH="/usr/local/sbin"
   UNINST_SH="uninstall-$NAME-$VERSION.sh"
   install_dir $DIRMODE "/${UNINST_PATH}"
   UNINST="$UNINST_PATH/$UNINST_SH"

   cat <<-tac > "$UNINST"
#!/bin/sh
#
if ! stty >/dev/null 2>/dev/null; then
   if [ x"\$DISPLAY" != x -a x"\$XTERM_LOOP" = x ]; then
      export XTERM_LOOP=1
      exec $XTERM -title "\$0" -e "\$0"
   fi
fi

echo "Ce script desinstalle $NAME $VERSION"
echo
echo "Appuyez sur une touche pour continuer ou CTRL-C pour quitter"
read -s -n 1 TOUCHE <&1
tac
      $CAT $TMP_FILE >> "$UNINST"
      echo "$DEL_FILE $UNINST" >> "$UNINST"
      chmod +x "$UNINST"
      echo
      echo "Un desinstalleur a ete cree dans $UNINST"
}

##########################################################################################
#     Main
##########################################################################################

# Initialisation
trap interrupt_quit 1 2 3 9 15 EXIT ERR

init

say_hello

root_check

# Demarrage de l'installation
(( etape = 1 ))
# Etape : installe les fichiers
install_file $etape && (( etape += 1 ))

# Etape : creation du menu dans KDE
install_menu $etape && (( etape += 1 ))

# Fin de l'installation
create_uninstaller

echo " "
echo "Installation terminee !"
exit 0
```