@@======INSTALLATION DE AETHERA======@@

//=====Pré requis=====//

~-Système Débian installé

~-Librairies suivantes installées:

~o libqpe-fb-dev
~o libaudio-dev
~o xlibs-dev
~o libtool

~-Outils suivants installés:

~o automake
~o bzip2

//=====Installation=====//

====QT3.3.3====

1. Se procurer les dernières paquetages récents sur
   «[[http://www.trolltech.com/download/index.html]] » : (qt-x11-free-3.3.3.tar.bz2)

Pour toute la suite de l'installation, ouvrir un Terminal et se mettre en superutilisateur (root).

2. Dans un terminal taper :
   %%> su -
   Password: le_mdp %%

3. Copier le paquetage dans /usr/share (avec la commande cp):
   %% > cp qt-x11-free-3.3.1.tar.bz2 /usr/share %%

4. Se placer dans /usr/share et dézipper le paquetage avec l'une des méthodes suivantes:
   %%> tar xvjf qt-x11-free-3.3.3.tar.bz2 %%
   ou
   %%> bunzip2 qt-x11-free-3.3.3.tar.bz2
> tar xvf qt-x11-free-3.3.3.tar %%

6. Toujours dans /usr/share, renommer le dossier qt-x11-free-3.3.3 obtenu après la décompression par qt333 avec la commande:
   %%> mv qt-x11-free-3.3.3 qt333 %%

//Qt est bien dans /usr/share mais n'es pas encore installé. (On aurait put le placer dans un autre répertoire comme /usr/local/).//

7. Nous allons configurer les variables d'environnements. Se placer dans le répertoire de root:
%%> cd /root %%

Editer le fichier caché .bashrc:
%%> vi .bashrc %%

//« Echap + i » permet de passer en mode console pour pouvoir modifier le fichier. (cf annexe 1)//

Ajouter les lignes suivantes:
%%
QTDIR=/usr/share/qt333 #(ou le répertoire de votre qt)
PATH=$QTDIR/bin:$PATH
MANPATH=$QTDIR/doc/man:$MANPATH
LD_LIBRARY_PATH=$QTDIR/lib:$LD_LIBRARY_PATH

export QTDIR PATH MANPATH LD_LIBRARY_PATH%%

//Quitter vi en enregistrant les modifications «Echap + wq»
pour vérifier votre environnement taper simplement « env » dans la console.//

8. Maintenant, on compile la bibliothède Qt. Lancer alors la configuration avec les options suivantes:

%%>cd /usr/share/qt333
> ./configure -qt-gif -system-nas-sound -thread -no-exceptions%%

- Si un problème ou pour recompiler, faire: %%> make confclean
> make clean
> make distclean%%, puis recompiler de nouveau.

- Si tout se passe bien, faire:
%%> make%%
Ne soyez pas étonné si la compilation prend du temps (entre 15 et 45 minutes)

puis faire
%%> make install%%

9. Une fois Qt installé, il faut ajouter les options suivantes dans ld.so.conf, Pour cela éditer ce fichier (avec vi par exemple):
%%> vi /etc/ld.so.conf%%
et ajouter les lignes:
%% /usr/local/qt333/lib
/usr/local/lib%%

Ensuite taper : %%> ldconfig%%

++Qt devrait maintenant correctement installé sur votre machine. Dans le répertoire / usr/share/qt333/bin/ se trouve les binaires des applications ""QtDesigner"" ""QtConfiguration"" ""QtLinguistic"".++

====AETHERA====

D'abord récuperer les sources de Aethera:

site officiel: « [[http://adullact.net/scm/?group_id=94]]»

et les placer dans un répertoire aethera dans votre /home/moi/aethera par exemple.

La méthode de récupération des sources CVS est expliqué sur ce même Wiki (annexe 2). Vous devez donc avoir un répertoire aethera comprenant les modules (sous formes de sous-répertoires) suivant :

{{colour c="blue" text="/aethera "}}
{{colour c="blue" text="/koplugin "}}
{{colour c="blue" text="/korelib "}}
{{colour c="blue" text="/minikde "}}
{{colour c="blue" text="/tkcbase "}}
{{colour c="blue" text="/tkcssl "}}

===Installation de Korelib:===

Si vous souhaitez récuperer la dernière version de Korelib «[[http://www.thekompany.com/projects/korelib]]», suivre les consignes d'installation (rubrique «Installation Instructions»). En cas d'erreur de compilation sur l'un des modules suivants, faire un tour dans la rubrique FAQ.

%%> automake
 > ./configure
 > make
 > make install%%

===Installation de tkcssl:===

%%
> cd /home/moi/aethera/tkcssl
> ./qmake.sh (chmod 755 si nécessaire)
> make
> make install%%

===Installation de tkcbase:===

%%
> cd /home/moi/aethera/tkcbase
> ./qmake.sh (chmod 755 si nécessaire)
> make
> make install%%

===Installation de aethera:===

%%
> cd /home/moi/aethera/aethera
> ./qmake.sh (chmod 755 si nécessaire)
> make
> make install%%

===Installation de webdav:===

%%
> cd /home/moi/aethera/webdav
> ./qmake.sh (chmod 755 si nécessaire)
> make
> make install%%

===Installation de koplugin:===

déplacer minikde.pro dans /koplugin/minikde
%%
> cd /home/moi/aethera/koplugin
> ./qmake.sh (chmod 755 si nécessaire)
> make
> make install%%

++La procédure d'installation de aethera est maintenant terminé. Le binaire du groupware aethera se trouve dans /usr/local/bin/aethera.bin.++

@@=====FAQ=====@@

==* lorsque je lance un des ./qmake.sh, il me renvoie permission denied :==

--> chmod 777 qmake.sh

==* lorsque je lance ./qmake.sh, il me renvoie : : bad interpreter: No such file or directory ==

--> les fichiers qmake et configure entre autres ont été formaté windows, c'est à dire qu'ils ont été converti en format texte de windows qui inclue des caractères invisible comme le retour a la ligne. Pour remédier à ce problème il faudra installer un des deux programmes suivants: dos2unix ou flip (> flip -u qmake.sh). Je vous conseil de reprendre entièrement le cvs de aethera sans passer par windows, car sinon il faudra tout convertir beaucoup trop de fichier et donc perdre beaucoup de temps.

==* lorsque je veux compiler le module tkcbase, il me manque les librairies palm qpe/...==

--> pour cela réinstaller les librairies de palm (debian: apt-get install libqpe-fb-dev)

==* installation de qt pose probleme lorsque je fais le make, il y a des problemes avec les librairies audio ==

--> installer la libraire libaudio-dev : %%> apt-get install libaudio-dev%%

==* Erreur; /usr/bin/ld: cannot find -lXt (librairie en rapport a X11)==

--> installer la bibliotheque xlibs-dev: %%> apt-get install xlibs-dev%%

==* Probleme de version qd je fé automake dans korelib:==

"automake requires `AM_CONFIG_HEADER', not `AC_CONFIG_HEADER'
aclocal.m4:421: required file `./$@)].in' not found? par exemple!"

Il manque libtool :

%%> apt-get install libtool%%

Si il y a un porbleme de conflit avec une autre version de automake, il faudra simplement supprimer la version la plus ancienne de automake:

apt-get remove automakevXXX

==* Compilation de aethera: config.h et hash_map no such file or directory==

Lorsque je lance make de aethera, j'obtiens l'erreur suivante:

?/usr/local/include/kore/metainfo.h:10:20: config.h: No such file or directory
/usr/local/include/kore/metainfo.h:14:20: hash_map: No such file or directory
make[2]: *** [.tmp/uipart.o] Error 1
make[2]: Leaving directory `/home/mzi/cvsaethera/aethera/libs/tino'
make[1]: *** [sub-tino] Error 2
make[1]: Leaving directory `/home/mzi/cvsaethera/aethera/libs'
make: *** [sub-libs] Error 2?

--> Pour cela, copier le fichier config.h se trouvant dans korelib et le mettre dans / usr/local/include/kore/

%%> cp /korelib/config.h /usr/local/include/kore/%%

==* erreur: cannot find -ltkcbase==

« /usr/bin/ld: cannot find -ltkcbase
collect2: ld returned 1 exit status
make[2]: *** [libaethera.so.1.0.1] Error 1 »

il faut déplacer soit même les libraires partagées!
Il y a seulement un lien dans /usr/local/lib
Le vrai ichier est manquant. Il faut juste copier tous les fichiers libtkcbase* dans /usr/local/lib et lancer ldconfig
---
==* korelib: « Makefile: Missing séparateur. Stop! »==

--> refaire: (ne pas oublié de faire) la procédure -automake -configure -make -make install
---
==* Il manque la libraire libkore.so lors du lancement de aethera: ==

-->erreur: « trying to load kore
Warning[kore]: libkore.so not found!
koreLib error: main components unavailable!"

Vous avez une version trop vieille de libtool sur votre système. L'astuce est de copier quelques fichiers de automake et de libtool dans votre répertoire korelib.

Chercher le répertoire d'installation de libtool et automakeXXX (XXX étant la version). Dans mon cas, /usr/share/libtool/libltdl/ et /usr/share/automake-1.6/.

D'abord et malheureusement, il est préférable de tout « décompiler » les modules avec les deux commandes: make clean puis make distclean

et recommencer la procédure d'installation de aethera depuis le début (Korelib/tkcssl/tkcbase/aethera/webdav/koplugin).

Ensuite, l'astuce est de copier les fichiers install.sh et missing de automake et les coller à la racine de korelib

cp /usr/share/automake-1.6/install.sh /home/moi/aethera/korelib/
cp /usr/share/automake-1.6/missing /home/moi/aethera/korelib/

et les fichiers config.guess config.sub ltmain.sh de libtool et les coller également à la racine de korelib

cp /usr/share/libtool/libltdl/config.guess /home/moi/aethera/korelib/
cp /usr/share/libtool/libltdl/config.sub /home/moi/aethera/korelib/
cp /usr/share/libtool/libltdl/ltmain.sh /home/moi/aethera/korelib/

Réfaire automake dans le module korelib, puis ./configure , make et make install. Maintenant, il devrait y avoir libkore.so/libkore.so.0/libkore.so.0.8.0 dans /usr/local/lib :).

@@====ANNEXE====@@

1.===Comment utiliser vi:===

Exemple, pour éditer le fichier: ld.so.conf, taper: vi /etc/ld.so.conf. Puis, appuyer sur les touches « Echap + i » (ou inser du clavier numérique), pour pouvoir éditer et naviguer avec les touches "down" et "up", "droite" et "gauche".
Pour quitter, appuyer sur la touche Echap puis taper :wq pour quitter vi en enrégistrant ou taper : q! pour quitter vi sans enregistrer.

2.==Comandes CVS de base==

CVS (Concurrent Versions System) est un outil d'aide au développement de logiciels.
Il permet une gestion efficace et riche des différentes versions pour un projet logiciel. Pour celà, cette méthode met en place un suivi, et par conséquent d'un historique, pour l'ensemble des fichiers appartenant au projet.
Il faut bien noter que la gestion de versions se fait autant au niveau de l'ensemble du projet qu'au niveau de chaque modules (ou même fichier) pris séparément.

Les commandes CVS se présentent sous la forme suivante :

cvs options_globales commande options_de_la_commande.

Les commandes de base pour la communication avec le serveur et le transfert de fichiers sont :

%%cvs add

cvs checkout (ou ses synonymes cvs co et cvs get)
cvs commit (ou ses synonymes cvs ci et cvs com)
cvs update (ou ses synonymes cvs up et cvs upd) %%

* ++Cvs add++

Cette commande est nécessaire lorsqu'on crée un nouveau fichier et qu'on souhaite l'ajouter au module. Après exécution, seules des modifications sur le poste local auront été faites : il faudra encore mettre à jour le serveur en utilisant la commande cvs commit .

* ++Cvs checkout++

Un checkout permet de demander au serveur de renvoyer les fichiers d'un module. Ainsi, la commande cvs checkout monmodule va extraire dans le répertoire courant des fichiers obtenus à partir du répertoire monmodule et ces fichiers seront prêts à être modifiés.

* ++Cvs commit++

La commande commit demande au serveur de valider et d'intégrer les changements qui ont pu être faits sur les fichiers en local. Le serveur mettra donc à jour les fichiers modifiés et leur attribuera un nouveau numéro de révision. Le commit peut se faire sur l'ensemble des fichiers modifiés, ou sur un fichier précis.

* ++Cvs update++

La commande update a pour effet de mettre à jour les fichiers locaux à partir de versions situées sur le serveur, ou de consulter le statut des fichiers locaux par rapport à ceux sur le serveur.

===Récupération de aethera sur le dépôt adullact:===

La première étape est d'installer CVS sur la machine si ce n'es déjà fait.

%%> apt-get install cvs%%

Normalement laisser les valeurs par défaut qui sont propres à l'OS à moins que vous vouliez personnaliser la configuration.

Puis:
%%export CVS_RSH=ssh
cvs -z3 -d:ext:developername@cvs.adullact.net:/cvsroot/epnadmin co modulename %%

===Récupération à partir du dépôt CVS de thekompany:===

Il faut créer la variable d'environement CVSROOT pour pouvoir utiliser les commandes cvs de bases:

export CVSROOT=":ext:login_utilisateur@cvs.thekompany.com:/home/cvs/public"

finalement faire un checkout comme suit:

cvs -z9 co nom_module

{{colour c="blue" text="co : contraction de « checkout »"}}
{{colour c="blue" text="-z9 : taux de compression maximum"}}

Après la commande checkout, vous devez obtenir le résultat suivant en fonction du module mise à jour (ici tkcssl).

RSA key fingerprint is XX:XX:XX:XX:XX:XX:XX:XX:XX:XX:XX:XX:XX:XX:XX:XX.
Are you sure you want to continue connecting (yes/no)? yes
Password:
cvs checkout: Updating tkcssl
U tkcssl/.cvsignore
U tkcssl/defines.pri
U tkcssl/library.cpp
U tkcssl/library.h
(...)
cvs checkout: Updating tkcssl/qt
U tkcssl/qt/qcleanuphandler.h
U tkcssl/qt/qdns.cpp
U tkcssl/qt/qdns.h

Récupération des sources grâve au CVS:

Si vous avez effectué des modifications à partir d'une révision et que quelqu'un a créé une nouvelle révision entre temps, l'appel de cvs update vous permettra de mettre à jour votre version locale en intégrant les changements. Il faudrat éventuellement résoudre des conflits. Vos modifications sont alors intégrées en local à la révision la plus récente, vous pouvez mettre à jour le fichier sur le serveur avec cvs commit.