



KDE 4 Development Setup

How to save time

How to code faster

Pradepto K Bhattacharya
pradepto@kde.org

FOSS.IN/2007, Bengaluru



Topics covered

New in KDE4:

CMake

DBus

"*srcdir != builddir*" mandatory

Running kde4 programs in a kde3 session

Switching to kdesvn-build

=> many new opportunities for saving time...



Example paths

Qt: [src ==] build == inst

/d/qt/3/<dir>
/d/qt/4/<dir>

KDE: src != build != inst

/d/kde/src/3
/d/kde/src/4

/d/kde/build/3
/d/kde/build/4

/d/kde/inst/dbusdir
/d/kde/inst/kde3
/d/kde/inst/kde4

...



Why all as one user

/d : Development partition, entirely user-writable

one key to compile+link+install, no su/sudo needed

no risk of root-owned files in the builddir

safer if something goes wrong

(wrong prefix would overwrite the system kde3)

**make install is faster than "make && sudo make install"
since the Makefiles are parsed only once**



Directory navigation

```
cd 4
```

```
=> goes to /d/kde/src/4
```

```
export CDPATH=./d/kde/src
```

```
Back/forward history, with zsh:
```

```
setopt AUTO_PUSHD
```

```
alias p=popd
```

```
cd 4/kdelibs/kio
```

```
cd 3
```

```
p
```

```
cd /d/qt/4/qt-x11-commercial-src-4.1.2
```

```
cd 4/kdelibs/kdecore/tests
```

```
cd 4 3
```



Path-based env switch

cd 4 should also set KDE 4 environment

How? By redefining cd. With zsh:

```
function cd() {  
  builtin cd $1 $2  
  if test -f .my-setup; then  
    echo "Loading my setup"  
    source .my-setup  
  fi  
}
```

Better version (going up and caching) at
http://web.davidfaure.fr/scripts/cd_function
Uses <http://web.davidfaure.fr/scripts/findup>

Add "cd ." at the end of .zshrc



KDE 4 environment

```
export DBUSDIR=/d/kde/inst/dbusdir
export PATH=$DBUSDIR/bin:$PATH
export LD_LIBRARY_PATH=$DBUSDIR/lib:$LD_LIBRARY_PATH
export PKG_CONFIG_PATH=$DBUSDIR/lib/pkgconfig:$PKG_CONFIG_PATH
```

DBus

```
export QTDIR=/d/qt/4/qt-copy
export PATH=$QTDIR/bin:$PATH
export LD_LIBRARY_PATH=$QTDIR/lib:$LD_LIBRARY_PATH
export PKG_CONFIG_PATH=$QTDIR/lib:$PKG_CONFIG_PATH
```

Qt

```
export KDEDIR=/d/kde/inst/kde4
export PATH=$KDEDIR/bin:$PATH
export LD_LIBRARY_PATH=$KDEDIR/lib:$LD_LIBRARY_PATH
export QT_PLUGIN_PATH=$KDEDIR/lib/kde4/plugins
export KDEDIRS=$KDEDIR
```

KDE

```
unset XDG_DATA_DIRS
# set it, to avoid seeing kde3 files from /usr
export XDG_DATA_DIRS=$KDEDIR/share
unset XDG_CONFIG_DIRS
```

XD
G

```
export KDEHOME=$HOME/.kde4
export KDETMP=/tmp/$USER-kde4
mkdir -p $KDETMP
export KDEVARTMP=/var/tmp/$USER-kde4
```

U
ser



DBus, qt-copy

Install dbus from source

```
./autogen.sh --enable-maintainer-mode --disable-qt --disable-qt3 --prefix=/d/kde/inst/dbusdir
```

Starting the dbus daemon

Some distributions do it (but DBUS_SESSION_BUS_PID isn't exported!)

Otherwise, add this in ~/.kde/env/dbus.sh :

```
eval `dbus-launch --sh-syntax --exit-with-session`
```

```
export DBUS_SESSION_BUS_PID
```

startkde starts the dbus daemon if DBUS_SESSION_BUS_PID unset

Checkout qt-copy

```
./apply_patches
```

```
./configure --prefix=$PWD ...
```

saves time when adding debug output in Qt itself



srcdir != builddir

Convenient in kde3, mandatory in kde4: out-of-source builds
Easy to rebuild from scratch
Easy to remove all generated files - even old ones
Easier look at the source files (grep, ls etc.)
Multiple build trees with different settings, e.g. debug and release

Example: /d/kde/src/4/kdelibs -> /d/kde/build/4/kdelibs

```
alias cb='cd /kde/src /kde/build'
```

```
alias cs='cd /kde/build /kde/src'
```

```
Non-zsh: alias cb='cd `pwd | sed -e s,/kde/src,/kde/build,`'
```

```
export OBJ_REPLACEMENT='s#/kde/src/#/kde/build/#'
```

used by makeobj, create_makefiles (kde3), pruneemptydirs etc.

```
alias make=makeobj to type make from the source dir (editor)
```



CMake and subdirs

```
cd kdelibs/kio/kio ; make => nothing
```

Make your editor use a wrapper script instead

```
#!/bin/sh
if test "$1" = "-k"; then shift; fi
cmake_in_parent=0
if test -f CMakeLists.txt; then
  if ! grep -q kde4_add CMakeLists.txt; then
    cmake_in_parent=1
    cd ..
  fi
fi
if test $# -gt 0; then
  arg="$1"
  if test "$arg" != install -a $cmake_in_parent -eq 1; then
    arg=`basename $PWD`/"$arg"
  fi
fi

jvalue=1
echo "calling makeobj -j $jvalue $arg"
makeobj -j $jvalue $arg
http://web.davidfaure.fr/scripts/makefromemacs
```



CMake and dependencies

```
touch kdeccore/kapplication.h  
cd kdeui ; make  
=> recompiles kdeccore first
```

```
touch kdeccore/kapplication.h  
cd kdeui ; make kdeui/fast  
=> recompiles only kdeui
```

```
make install/fast
```

```
Normal mode is also useful:  
kdelibs/kio/kio/kdirmodel.cpp  
kdelibs/kio/tests/kdirmodeltest.cpp  
from kdelibs/kio/tests: make kdirmodeltest
```



Compiling a KDE module

Manually:

```
svn co ....kdelibs
```

```
cb ; mkdir kdelibs ; cd kdelibs
```

```
cmake -DKDE4_BUILD_TESTS=TRUE -DCMAKE_BUILD_TYPE=debugfull \  
-DCMAKE_INSTALL_PREFIX=/d/kde/inst/kde4 /d/kde/src/4/kdelibs
```

```
make
```

Note: never rm CMakeCache.txt and then type make; always re-run cmake with the correct options first.



Using kdesvn-build

All-in-one: checking-out/updating, configuring, make, make install

Global options:

binpath - don't forget the path to icecream, \$PATH doesn't count
qtdir, svn-server, source-dir, build-dir, kdedir

Per-module options:

```
module qt-copy
    # kdesvn-build sets the prefix to $qtdir
    configure-flags [...]
    apply-qt-patches true
    make-options -j3 sub-src sub-tools
end module

module kdelibs
    cmake-options -DKDE4_BUILD_TESTS=TRUE
    -DCMAKE_BUILD_TYPE=debugfull
end module
```



Subversion tricks

Recursive grep hits subversion's own copy:

```
./svn/text-base/kapplication.cpp.svn-base: [...]  
./kapplication.cpp: [...]
```

Solution:

```
alias rgrep='wgrep -r'  
comes from subversion-1.4.0/contrib/client-side/wgrep
```

Update to subversion-1.4: much faster!



KDE4 programs in kde3

```
Xephyr -screen 1240x768 -ac :4 &  
export DISPLAY=:4  
sh -x startkde 2>&1 | tee kde4.log
```

```
> ./list-kde4-binaries  
/d/kde/inst/kde4/bin/kwin, pid 6756  
/d/kde/inst/kde4/bin/kpersonalizer, pid 6757
```

```
> ./kill-kde4-binaries  
Killing /d/kde/inst/kde4/bin/kwin, pid 6756  
Killing /d/kde/inst/kde4/bin/kpersonalizer, pid 6757
```

K menu, execute, konsole

Happy bugfixing! :)



Questions?

Questions?