Qt5 & Yocto: SDK and app migration

Denys Dmytriyenko

LCPD, Arago Project

Texas Instruments

Agenda

- Qt history in OpenEmbedded
- Qt4/5 usage in OE/Yocto
- Building and packaging filesystem images
- Qt SDK basics
- Qt5 SDK usage
- App migration between Qt4 and Qt5

Qt history in OpenEmbedded

- Classic OpenEmbedded
  - qte, qtopia, Qt Extended, OPIE...
  - qt-x11-free – Qt 3 for X11
  - Qt 4 – X11 and Embedded
- OpenEmbedded-Core
  - qt4-x11 and qt4-embedded (in OE-Core, proposal to separate)
  - Qt3 in meta-qt3:
    - http://git.yoctoproject.org/cgit/cgit.cgi/meta-qt3
  - Qt5 in meta-qt5:
    - http://github.com/meta-qt5/meta-qt5

meta-qt5 layer

- A separate layer on GitHub
- Maintained by Martin Jansa and Otavio Salvador
- https://github.com/meta-qt5/meta-qt5

Using Qt4

- BBLAYERS += “.../openembedded-core/meta” in conf/bblayers.conf
- inherit qt4e or qt4x11
- Qt4 is monolithic and builds everything in single recipe
  - Application's build dependencies are handled automatically
  - May need to RRECOMMENDS or otherwise install plugins and other pieces on the target

Using Qt5

- Depends on openembedded-core/meta and meta-openembedded/meta-ruby
- In conf/bblayers.conf:

```
BBLAYERS += “ \
…/meta-qt5 \
…/meta-openembedded/meta-ruby \
…/openembedded-core/meta”
```

- inherit qmake5
- PACKAGECONFIG in qtbase controls “USE” flags and external dependencies of the build
- Very modular, so need to DEPENDS on necessary components, e.g. qtdeclarative, qtmultimedia, qtwebkit, etc.

Bundle Qt4 in images

- Qt4 build creates large number of packages – libraries, plugins, fonts etc.
  - Very granular and can reduce overall size of the image
  - May be tedious to list all the necessary packages for the image
  - With Debian naming enabled, most libraries are renamed, others are not:
    - libqt-embeddedcore4
    - libqt-embeddedmultimedia4
    - libqt-embeddedopengl4
    - qt4-embedded-qml-plugins
    - qt4-embedded-plugin-imageformat-jpeg
  - Library dependencies are handled by OE automatically, plugins and data need explicit manual listing in packagegroup or image

Bundle Qt5 in images

- Since Qt5 project is modular on its own, only required packages are built
- Still need to handle plugins and other data manually in packagegroup or image:
  - qtbase-plugins
  - qtwebkit-qmlplugins
  - qtwebkit-examples-examples

Qt SDK basics

- Set of host tools, target libraries and header files for cross-compiling applications on the host system outside of OpenEmbedded/Yocto to be used on the target
- OE-built SDK comes with environment-setup script to set all the environment variables necessary to use the provided sysroots and drive the cross-compilation

Qt4 SDK

- Standard meta-toolchain-qt and meta-toolchain-qte recipes, based on meta-toolchain for building and packaging toolchains/SDKs
- Alternatively, bitbake -c populate_sdk for the rootfs image will generate an SDK with corresponding *-dev and *-dbg packages

Qt5 SDK

- Mostly developed in meta-arago for TI SDK in late 2013
- Upstreamed to meta-qt5 layer in early 2014
  - Thanks to Otavio for provided reviews and help
- Similarly, supports bitbake -c populate_sdk as a main way of building and packaging SDK
- Legacy method of meta-toolchain-qt5 is also supported

Using Qt SDK

- Install a self-extracting *.sh file from a deploy/sdk directory on your host system
- Source the environment-setup script
- Run qmake helloworld.pro to generate a Makefile from Qt project file
- Run GNU make to cross-compile the application
- All the magic to use the correct cross-compilation toolchain, target libraries and headers is done behind the scene!

Application migration

- Arago Project comes with few sample Qt applications for demonstrating some of the capabilities.
- Need to re-use the same sample Qt apps on either Qt4 or Qt5 systems
- Implemented some mechanisms to migrate existing Qt app recipes to be buildable against Qt4 or Qt5 libraries
- Introduce and discuss qt-provider and qt-vars classes from meta-arago

Application migration (cont)

qt-provider.bbclass

- QT_PROVIDER variable selects which Qt version is being used - “qt5”, “qt4e”, “qt4x11” etc.
- Based on that, necessary classes are inherited and other setup steps performed – inherit qt4e etc.

qt-vars.bbclass

- Defines a set of variables to be used in DEPENDS and RDEPENDS statements
  - QT_DEPENDS_BASE is qtbase for qt5 and qt4-embedded for qt4e
  - QT_DEPENDS_WEBKIT is qtwebkit for qt5 and empty for qt4e
  - QT_RDEPENDS_FONTS is qtbase-fonts for qt5 and qt4-embedded-fonts for qt4e

Application migration (cont)

- There may be some sources and Makefile modifications required per Qt5 Migration Guide - http://qt-project.org/wiki/Transition_from_Qt_4.x_to_Qt5
- Replace QtGui include with QtWidgets:

```
-#include <QtGui>
+#include <QtWidgets>
```

- Replace QString fromAscii()/toAscii() with fromLatin()/toLatin():

```
-m_cityId = parseCityInfo(QString::fromAscii(data));
+m_cityId = parseCityInfo(QString::fromLatin1(data));
```

- May need to add to the .pro project file:

```
QT += widgets
```

Sample recipe

```
DESCRIPTION = "Qt Demo"
LICENSE = "BSD"
LIC_FILES_CHKSUM = "file://LICENSE;md5=93a105adb99011afa5baee932b560714"

inherit qt-provider

DEPENDS += "${QT_DEPENDS_SVG} ${QT_DEPENDS_SCRIPT}"
QT_DIFF = " \
file://0001-Replace-QtGui-with-QtWidgets-per-Qt5-migration-guide.patch \
file://0002-Replace-fromAscii-toAscii-with-fromLatin1-toLatin1-p.patch"

SRC_URI = "git://gitorious.org/qt-demo/qt-demo.git;protocol=git"

SRC_URI += "${@base_conditional('QT_PROVIDER','qt5',${QT_DIFF},'',d)}"
```

Thank you

Q&A