

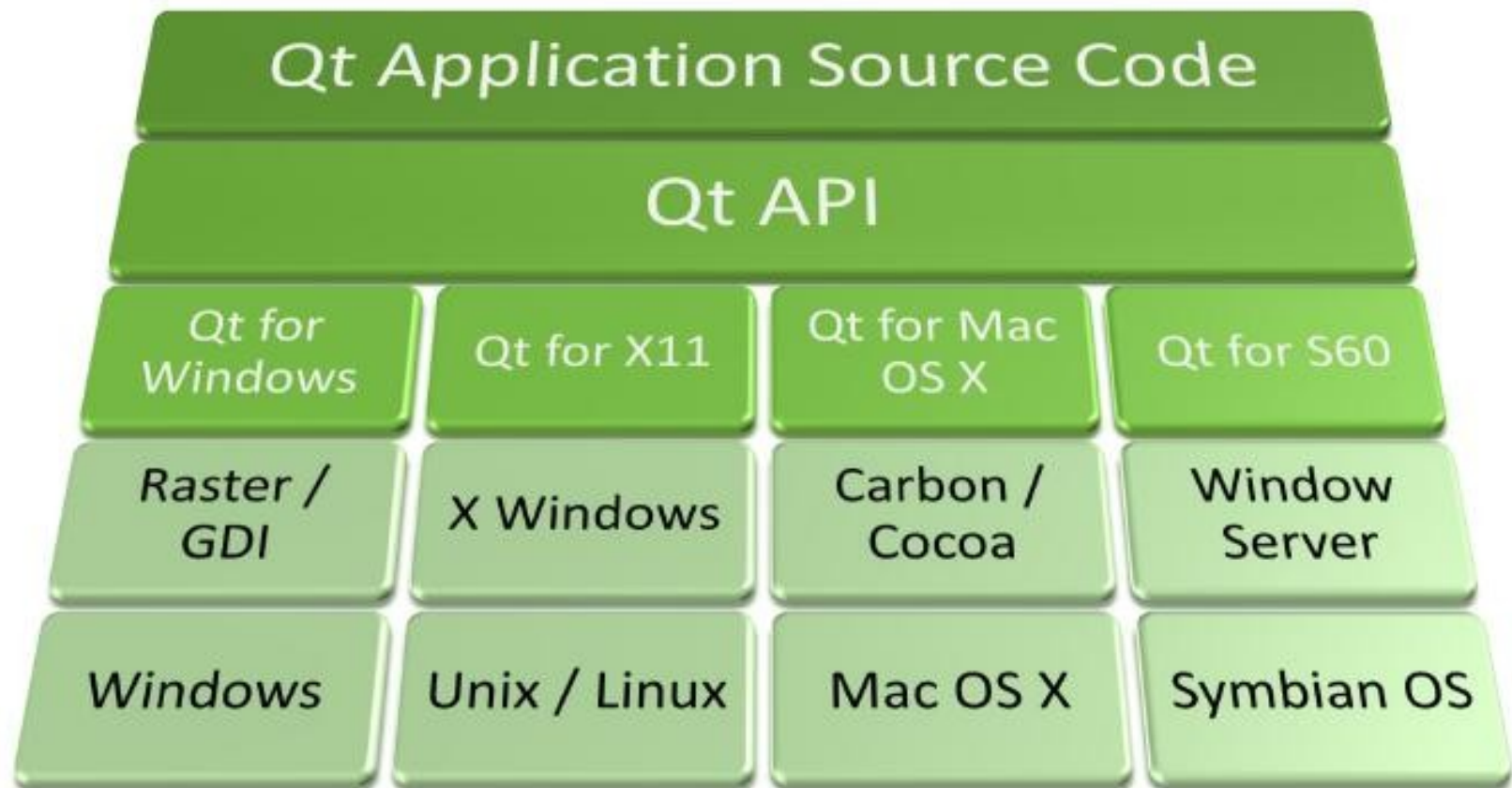
Qt pour Linux embarqué

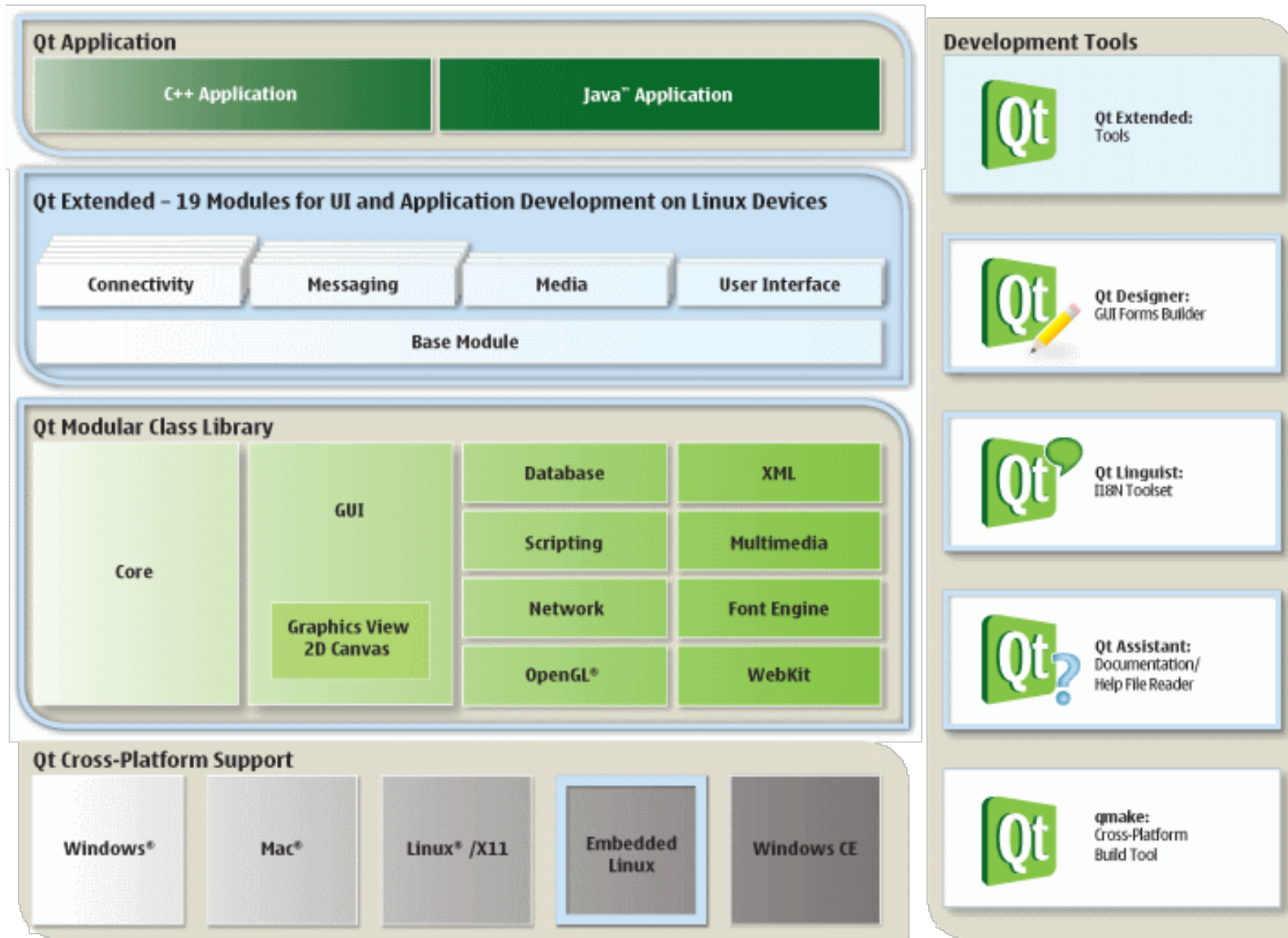
Pierre Ficheux (pierre.ficheux@openwide.fr)

CTO Open Wide Ingénierie

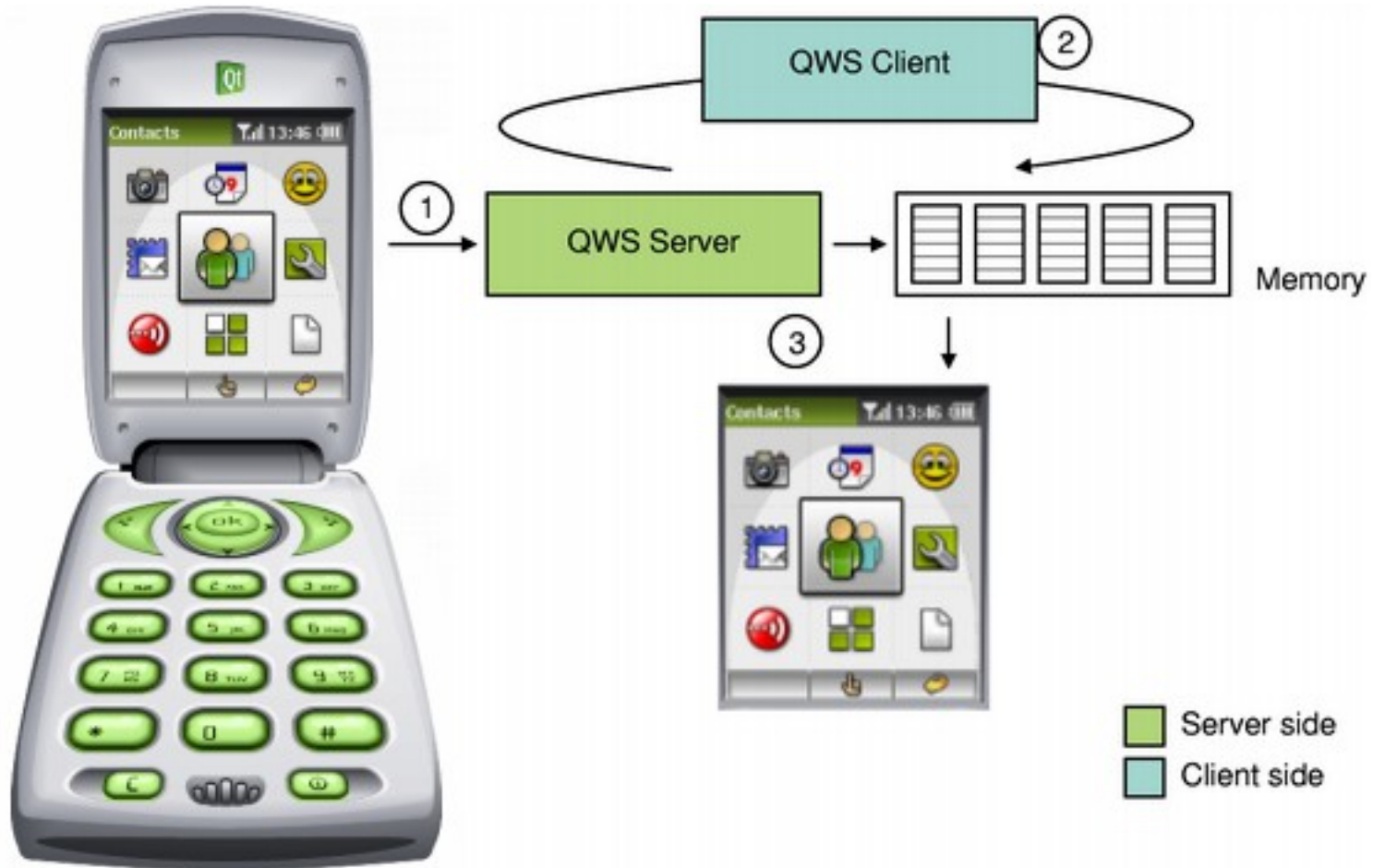
Octobre 2011

- Première version publiée par Trolltech en 1996
→ toolkit en C++
- Développé pour X11, 2 développeurs au départ, fondateurs de Trolltech
- Outil multi-plateforme (Unix, Windows, MacOS)
- Connu grâce à KDE !
- Qt2 en 2000 → Qtopia (PDA sous Linux « Zaurus » de SHARP)
- Jusqu'en 2008, double licence GPL/Propriétaire
- En 2008, achat par Nokia → LGPL
- Qt4: + Symbian, WinCE, Maemo, ...
- En 2011 rapprochement Nokia / M\$ → licence commerciale cédée à Digia PLC





- Client/serveur → QWS = Qt Window System
- Basé par défaut sur le framebuffer de Linux
- Peut utiliser un framebuffer « virtuel » (X11) avec QVFb
- Une application serveur (-qws), les autres clientes
- Le serveur gère en général l'affichage mais l'application peut accéder directement à l'écran pour des raisons de performances
- Ajout d'un pilote accéléré en dérivant QScreen et QRasterPaintEngine
- <http://doc.qt.nokia.com/4.7/qt-embedded-accel.html>



- Basée sur qmake (sur-couche à make similaire à GNU/Autotools)
- Utilisation par défaut du fichier `mkspecs/qws/linux-arm-g++` → définition de la chaîne de compilation croisée

```
$ ./configure -embedded arm -xplatform qws/linux-arm-g++
```

```
$ make; make install
```
- Configuration spéciale

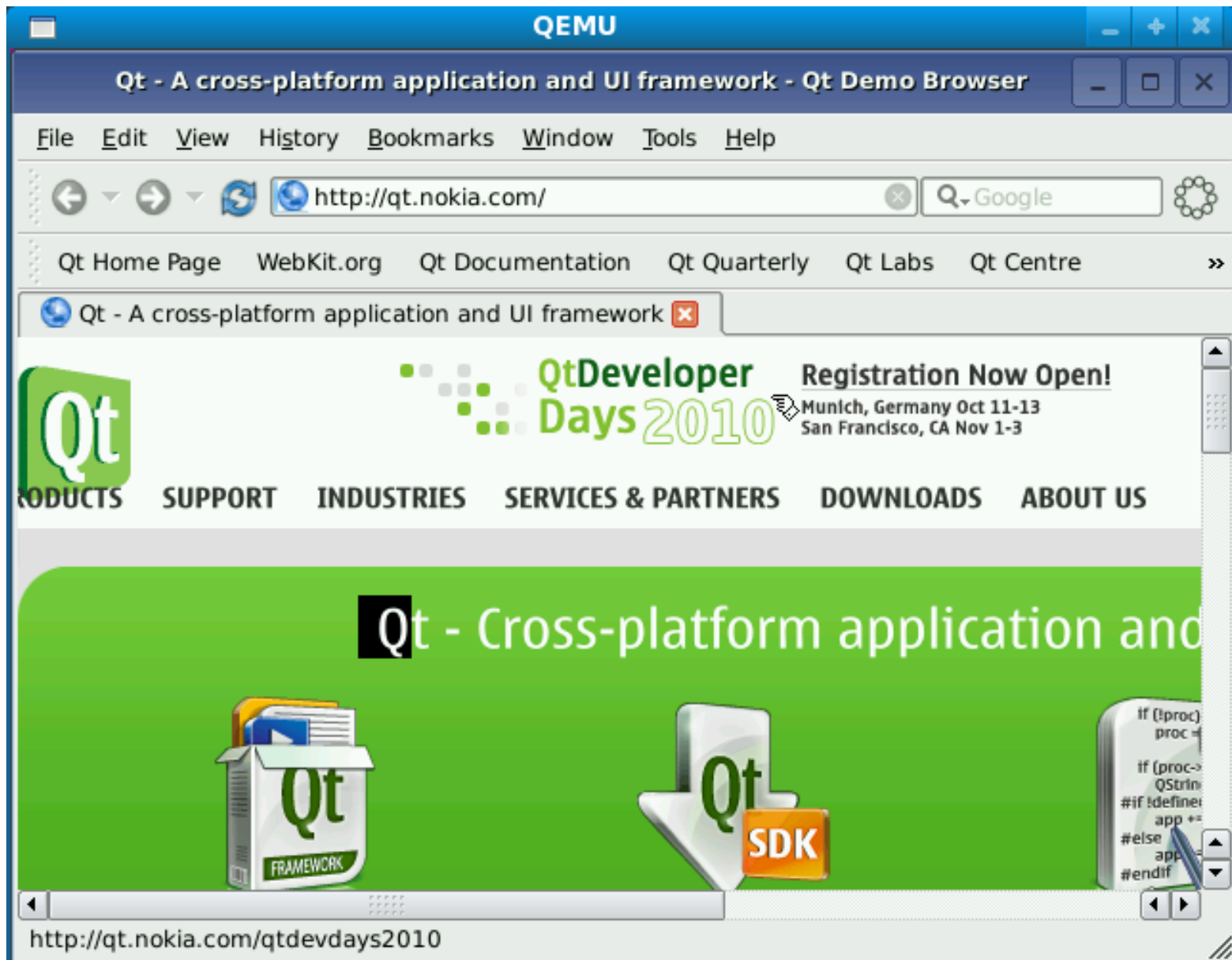
```
$ ./configure -embedded arm -xplatform qws/linux-myconfig-g++
```
- Intégré à Buildroot

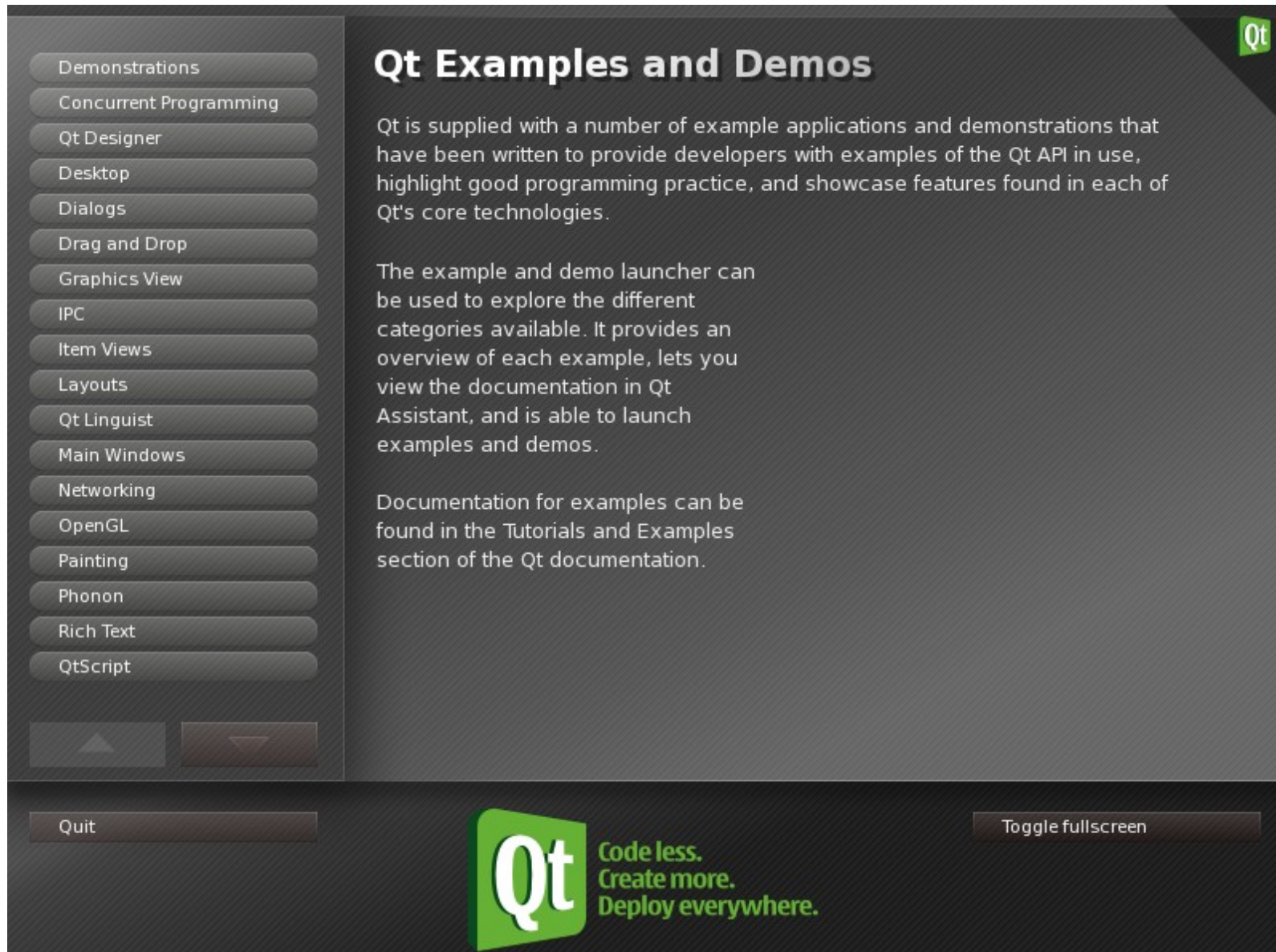
- Premier test possible avec QEMU ! (framebuffer non accéléré)

```
$ qemu-system-arm -M versatilepb -m 128  
-kernel zImage -initrd rootfs.gz  
-append "mem=128M"
```

- Nombreux programmes de test
- Ne pas oublier l'option `-qws`
- Utiliser la variable `QT_QWS_FONTDIR` pour les polices :

```
# export QT_QWS_FONTDIR=/usr/lib/fonts  
# /usr/examples/widgets/analogclock/analogclock -qws
```



A screenshot of the Qt Examples and Demos application window. The window has a dark grey background with a grid pattern. On the left side, there is a vertical list of buttons for different categories: Demonstrations, Concurrent Programming, Qt Designer, Desktop, Dialogs, Drag and Drop, Graphics View, IPC, Item Views, Layouts, Qt Linguist, Main Windows, Networking, OpenGL, Painting, Phonon, Rich Text, and QtScript. Below these buttons are two arrow buttons. The main content area on the right is titled 'Qt Examples and Demos' and contains three paragraphs of text. At the bottom of the window, there is a 'Quit' button on the left, a Qt logo with the slogan 'Code less. Create more. Deploy everywhere.' in the center, and a 'Toggle fullscreen' button on the right. A small Qt logo is also visible in the top right corner of the main content area.

Qt Examples and Demos

Qt is supplied with a number of example applications and demonstrations that have been written to provide developers with examples of the Qt API in use, highlight good programming practice, and showcase features found in each of Qt's core technologies.

The example and demo launcher can be used to explore the different categories available. It provides an overview of each example, lets you view the documentation in Qt Assistant, and is able to launch examples and demos.

Documentation for examples can be found in the Tutorials and Examples section of the Qt documentation.

Quit

Toggle fullscreen

Qt Code less.
Create more.
Deploy everywhere.

- LE point faible de Qt
- Consommation de l'exemple analogclock: 20 Mo
- Consommation de webkit: 50 Mo
- Consommation d'une démonstration DirectFB (df_andi) : 2 Mo
- Qt est destiné au développement d'applications « complexes » !

- Le même code source utilisable sur toutes les plateformes grâce à qmake :

```
$ qmake-qt4  
$ make  
$ file DemoNew
```

Linux/X11



```
DemoNew: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamically linked  
(uses shared libs), for GNU/Linux 2.6.18, not stripped
```


```
$ make distclean  
$ type qmake  
qmake est haché (/home/pierre/buildroot-2010.05/output/staging/usr/bin/qmake)  
$ qmake  
$ make  
$ file DemoNew
```

Buildroot (ARM)



```
DemoNew: ELF 32-bit LSB executable, ARM, version 1 (SYSV), dynamically linked (uses  
shared libs), for GNU/Linux 2.6.33, not stripped
```

- Qt est PLUS qu'un toolkit graphique !
- Abstraction pour sockets, threads, Unicode, SQL, ...
- Les objets dérivent de QObject mais ne sont pas uniquement des objets graphiques, ex: QThread
- Les « signaux/slots » remplacent avantageusement les *callbacks*
- Prise en compte I18N
- Chargement dynamique d'IHM

- Outil WYSIWYG de création d'interface
- IDE de développement semblable à Eclipse
 - Edition du code
 - Compilation (croisée)
 - Mise au point (GDB)
- Les fichiers d'IHM Qt (.ui) sont au format XML
- Permet de définir des cibles X11, embedded, etc.
- Moins agréable que  :-)

The screenshot shows the Qt Creator IDE interface for designing a Qt widget. The main canvas displays a button with the text "hello N900!". The left sidebar contains a widget palette with categories like Layouts, Spacers, Buttons, Item Views, Item Widgets, and Containers. The right sidebar shows the Object Inspector and Property Inspector for the selected QPushButton widget.

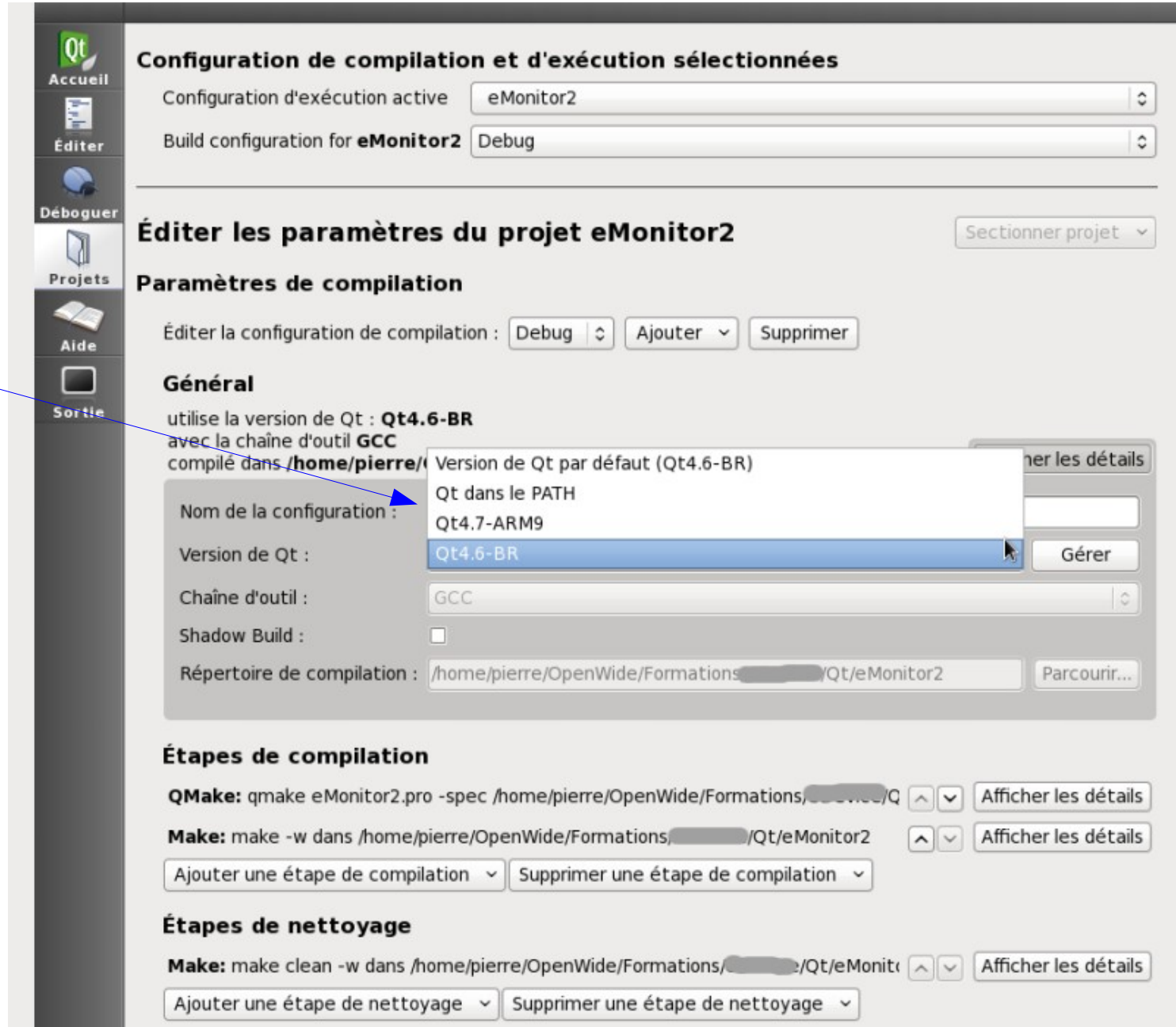
Object Inspector:

Object	Class
MainWindow	QMainWindow
centralWidget	QWidget
pushButton	QPushButton

Property Inspector:

Property	Value
layoutDirection	LeftToRight
autoFillBackg...	<input type="checkbox"/>
styleSheet	
locale	Polish, Poland
inputMethod...	ImhNone
QAbstractButton	
text	hello N900!
icon	
iconSize	16 x 16
shortcut	
checkable	<input type="checkbox"/>
checked	<input type="checkbox"/>
autoRepeat	<input type="checkbox"/>
autoExclusive	<input type="checkbox"/>
autoRepeatD...	300
autoRepeatI...	100

Sélection SDK



Configuration de compilation et d'exécution sélectionnées

Configuration d'exécution active: eMonitor2
Build configuration for eMonitor2: Debug

Éditer les paramètres du projet eMonitor2 Sectionner projet ▾

Paramètres de compilation

Éditer la configuration de compilation: Debug ▾ Ajouter ▾ Supprimer

Général

utilise la version de Qt : **Qt4.6-BR**
avec la chaîne d'outil **GCC**
compilé dans **/home/pierre/**

Nom de la configuration :
Version de Qt : Qt4.6-BR
Chaîne d'outil : GCC
Shadow Build :
Répertoire de compilation : /home/pierre/OpenWide/Formations/Qt/eMonitor2 Parcourir...

Étapes de compilation

QMake: qmake eMonitor2.pro -spec /home/pierre/OpenWide/Formations/Qt/eMonitor2 ^ ▾ Afficher les détails
Make: make -w dans /home/pierre/OpenWide/Formations/Qt/eMonitor2 ^ ▾ Afficher les détails

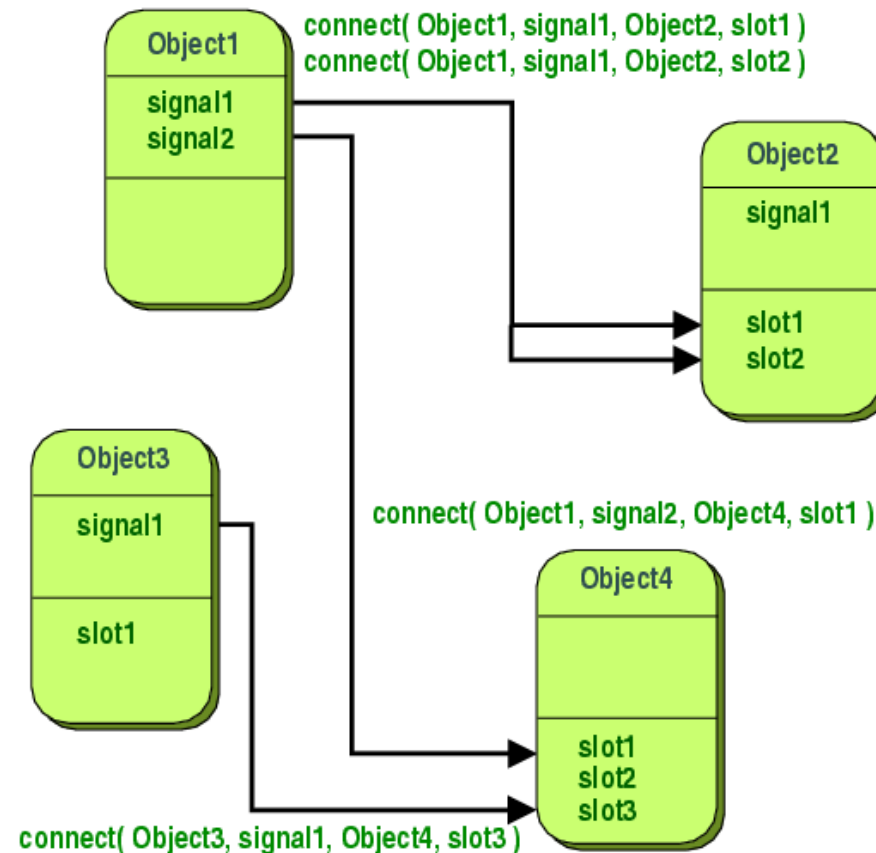
Ajouter une étape de compilation ▾ Supprimer une étape de compilation ▾

Étapes de nettoyage

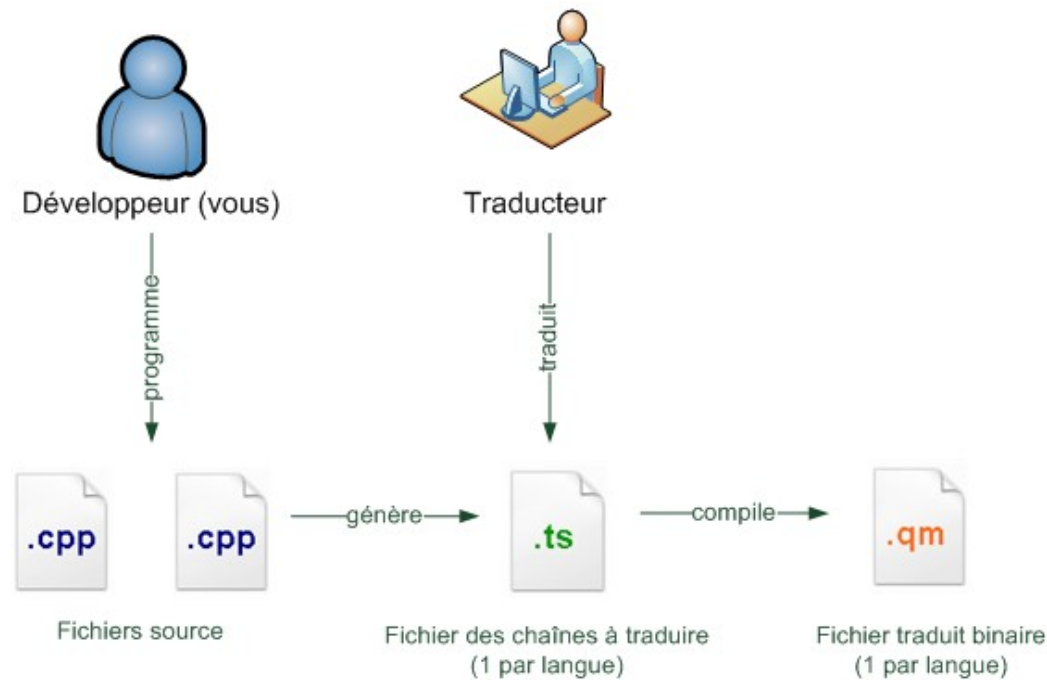
Make: make clean -w dans /home/pierre/OpenWide/Formations/Qt/eMonitor2 ^ ▾ Afficher les détails

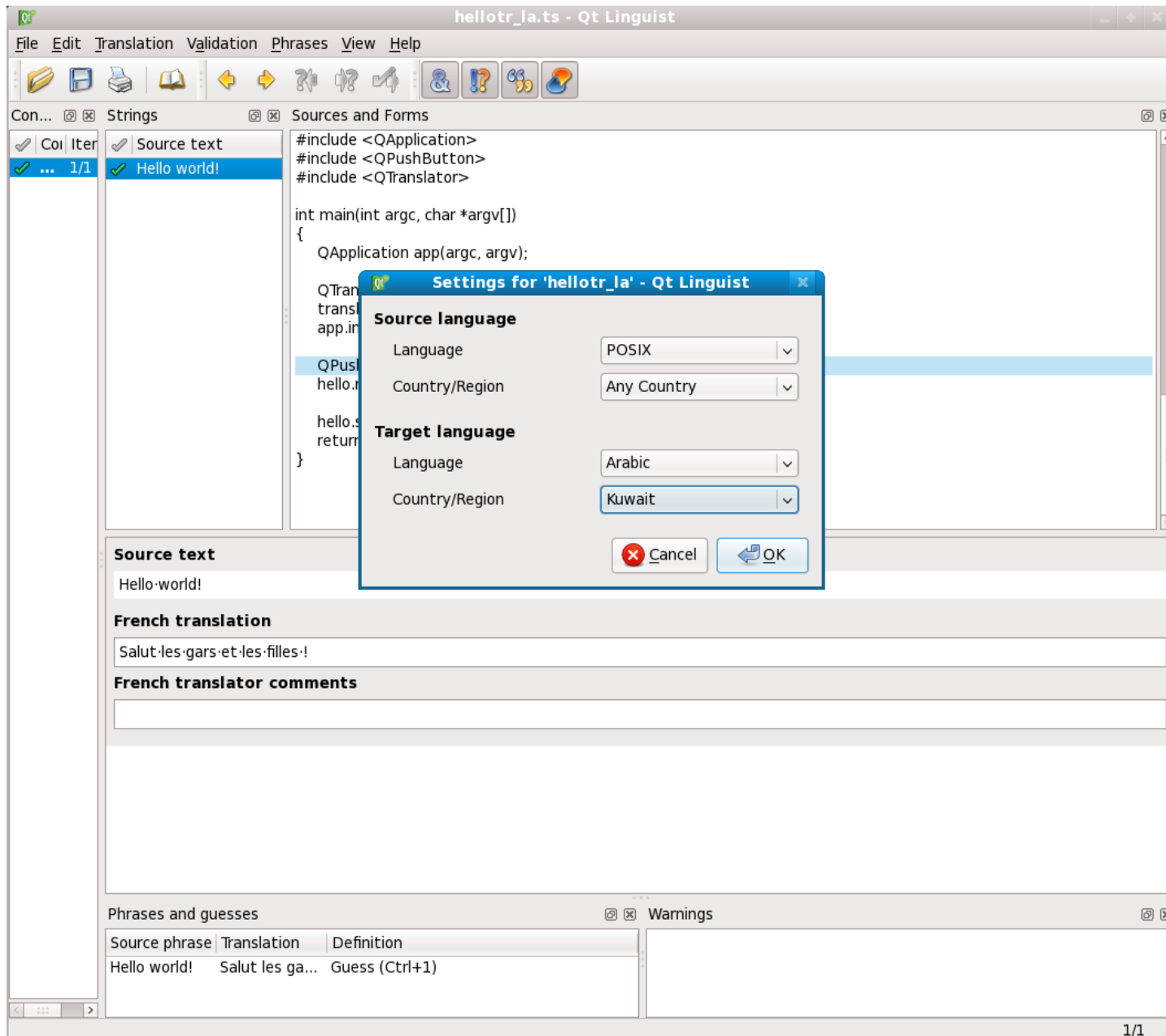
Ajouter une étape de nettoyage ▾ Supprimer une étape de nettoyage ▾

- Callback \rightarrow un événement \Leftrightarrow une fonction
- Qt : Un signal (ex: clic bouton) est reçu par plusieurs « slots » après connexion



- Message extraits du code source (.cpp) → .ts (XML)
- Traduction avec Qt Linguist en compilé en .qm





hellotr_la.ts - Qt Linguist

File Edit Translation Validation Phrases View Help

Con... Strings Sources and Forms

Source text: Hello world!

French translation: Salut les gars et les filles!

French translator comments:

Phrases and guesses: Source phrase Translation Definition
Hello world! Salut les ga... Guess (Ctrl+1)

Warnings:

1/1

Settings for 'hellotr_la' - Qt Linguist

Source language

Language: POSIX

Country/Region: Any Country

Target language

Language: Arabic

Country/Region: Kuwait

Cancel OK

- Chargement d'un .ui au démarrage → pas de re-compilation des .cpp
- Utilisation de la classe QUiLoader

Fichier créé avec QtCreator

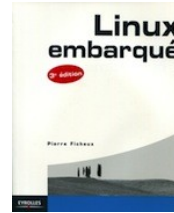
```
QUiLoader loader;
QFile file("form.ui");
file.open(QFile::ReadOnly);
QWidget *widget = loader.load(&file);
```

- Disponible sur la version 4.7
- Programmation dynamique d'interface basée sur le langage QML (Qt Modeling/Meta Language)
- Intégré à QtCreator
- Intégrable au code Qt existant
- Fonctionnellement « semblable » à HTML5 ?

- GTK+ : encore mal adapté au framebuffer
- WxWidgets : idem
- EFL : très bon outil (plus léger que Qt) mais pas d'outil de construction.

REMARQUE :

La puissance des cartes embarquées augmentant, il est de plus en plus fréquent d'utiliser X11



- Linux embarqué version 3 : <http://www.eyrolles.com/Informatique/Livre/linux-embarque-9782212124521>
- <http://www.slideshare.net/qtbynokia/qt-on-real-time-operating-systems>
- <http://www.docstoc.com/docs/56800911/directfb-introppt---Report-of-W>
- <http://qt.developpez.com/doc/4.6/requirements-embedded-linux/>
- <http://doc.qt.nokia.com/4.7/qt-embedded-accel.html>
- <http://doc.qt.nokia.com/4.7-snapshot/qt-embedded-crosscompiling.html>
- <http://doc.qt.nokia.com/4.7-snapshot/qt-embedded-linux.html>

