Qt 5

Guillaume Belz

1 Sortie de Qt 5 alpha

Reprise de mon article publié le 16 avril 2012

La première version majeure du Qt Project autonome se concentre sur les performances et les capacités graphiques

La version 5 de Qt vient de sortir en version alpha. Cette version est la première version majeure depuis que Qt est devenu autonome avec la création du Qt Project. Beaucoup de personnes ont contribué à cette nouvelle version, pas uniquement des développeurs de chez Nokia. Les différents modules ont été regroupés en deux catégories, les essentiels, installés par défaut, et les add-ons, installés à la demande. L'objectif de cette version alpha est de récupérer les retours des utilisateurs, principalement sur les modules essentiels.

Lars Knoll, le responsable en chef du projet Qt, a publié en mai dernier deux discussions sur les QtLabs pour présenter les approches choisies pour Qt 5 (voir les discussions Thoughts about Qt 5 et Responses to Qt 5). La pensée directrice est résumée dans les phrases suivantes :

« Qt 5 doit être le fondement d'une nouvelle façon de développer des applications. Tout en offrant la puissance de Qt natif en C++, l'accent sera mis sur un modèle où le C++ sera principalement utilisé pour implémenter des fonctionnalités modulaires d'arrière-plan pour Qt Quick », a déclaré Lars Knoll.

Neuf mois de travail, plusieurs centaines d'intervenants et plusieurs milliers de modifications du code ont été nécessaires pour aboutir à cette version alpha. Pour cette première version majeure, l'accent a été mis sur la partie embarquée, proche de la vision que Lars Knoll a décrite, mais il faudra attendre les versions 5.1 ou 5.2 pour que cette vision soit entièrement appliquée pour la version desktop.

Cette version alpha est l'aboutissement d'un travail important sur quatre points : QPA, la pile graphique, la modularité et le nettoyage de l’architecture en déplaçant les QWidgets dans les

modules add-ons.

1.1 Le Qt Platform Abstraction Layer (QPA)

Pour améliorer la portabilité de Qt, il a été nécessaire de restructurer l'architecture pour isoler toutes les fonctionnalités de bas-niveau qui sont spécifiques à une plate-forme. Ce travail a permis d'aboutir au QPA, facilitant le portage de Qt sur toutes nouvelles plate-formes. Cette abstraction a été introduite dans Qt 4.8 en remplacement de QWS pour les versions embarquées de Qt, mais elle est maintenant disponible pour toutes les éditions dans Qt 5. La meilleure preuve de l'efficacité de cette abstraction est que plusieurs portages sont en cours de développement : pour QNX, iOS et Android, par exemple.

1.2 La réorganisation de la pile graphique

Un autre objectif majeur pour Qt 5 est l'amélioration des performances graphiques, en particulier pour les versions embarquées. Pour ce faire, il a fallu réorganiser la pile graphique, pour bénéficier au maximum de l'accélération matérielle. Pour cela, l'accent a été mis sur l'utilisation d'OpenGL.

Par exemple, Qt Quick 2 a subi une réorganisation importante se basant sur le graphe de scène et utilisant OpenGL (GL ES 2 minimum) en arrière-plan. Qt Gui contient maintenant des classes QOpenGL à la place des classes QGL (maintenues dans le module QtOpenGL pour la compatibilité).

On note l'apparition de nouvelles classes :

- QGuiApplication, plus légère que QApplication (hérite de QCoreApplication et dérivée par QApplication) ;
- QWindow, pour manipuler les fenêtres de premier plan. QWidget et dérivées continuent de fonctionner, comme dans Qt 4, avec QPainter, bien que cet outil soit moins utilisé pour les autres piles graphiques (il est maintenant limité à la rastérisation logicielle sur écran, les images et les pixels, avec un backend OpenGL et un autre pour la génération de PDF et l'impression).

1.3 L'architecture modulaire

Objectif : flexibilité, possibilité de choisir ses modules pour les utilisateurs, meilleure intégration de Qt Mobility, faciliter les contributions en les incluant comme modules tiers. Il s'agit principalement de ménage interne, peu visible par les utilisateurs (toujours en cours).

1.4 Déplacer QWidget dans un module indépendant

Déplacer ces classes dans le module "widgets" permet de garantir la continuité des QWidget et dérivés, mais également l'évolution vers d'autres approches (QML et Qt Quick). Cela nettoie l'architecture sur le long terme.

1.5 Installation et compilation

Il y a plusieurs moyens d'installer Qt 5. Le plus simple est d'utiliser les binaires non officiels, régulièrement mis à jour :

- à partir des dépôts ppa (Linux) : https://launchpad.net/~forumnokia/+archive/fn-ppa ;
- à partir de Git : Building Qt 5 from Git ;
- à partir des sources Qt 5.0 Alpha release en différents format (7z, tar.bz2, tar.gz, tar.xz, zip) ;
- pour compiler : Qt 5 Alpha building instructions.

1.6 Passer de Qt 4 à Qt 5

Les changements importants pour conserver la compatibilité du code écrit pour Qt 4 avec Qt 5 sont d'intégrer le module widgets si on utilise des QWidget ou dérivés et de renommer le module Qt Quick en quick1. Voici un exemple de code dans le fichier .pro pour garantir la compatibilité :

```
greaterThan(QT_MAJOR_VERSION, 4)
{
    QT += widgets
    QT += quick1
} else {
    QT += declarative
}
```

Le script Perl qtbase/bin/fixqt4headers.pl met à jour les inclusions des fichiers d'en-tête.

Pour la création de plugins, les macros Q_EXPORT_PLUGIN et Q_EXPORT_PLUGIN2 sont dépréciées et doivent être remplacées par la macro Q_PLUGIN_METADATA, qui permet de lire les informations sans devoir charger le plugin avec la fonction dlopen().

1.7 Sources

- la mailing list pour les retours utilisateurs : http://lists.qt-

project.org/pipermail/development/ ;

- le bugtracker pour signaler des bugs : https://bugreports.qt-project.org/secure/Dashboard.jspa ;
- la documentation : http://qt-project.org/doc/qt-5.0/ ;
- la description de Qt 5 sur le wiki de qt-project : http://qt-project.org/wiki/Qt-5-Alpha ;
- l'annonce sur QtLabs : http://labs.qt.nokia.com/2012/04/03/qt-5-alpha/ ;
- l'annonce sur le blog de Qt : http://blog.qt.nokia.com/2012/04/03/...of-the-future/ ;
- l'annonce sur le blog de Digia : http://www.digia.com/en/Blogs/Qt-blo...ther-for-Qt-5/.

2 Les modules de Qt 5

Reprise de mon article publié le 4 septembre 2012

L'un des principaux changements que l'on trouvera dans Qt 5 est la réorganisation des modules. Les modules sont regroupés en deux groupes : les Essentials, installés automatiquement, et les Add-ons, installés à la demande. Puisque Qt 5 n'est pas encore en version finale, les informations données dans cet article sont susceptibles d'être modifiées.

2.1 Les modules Essentials

2.1.1 Le module Qt Core

Ce module fournit les fonctionnalités de base de Qt, excepté ce qui concerne l'interface graphique. Tous les autres modules sont liés à ce module. Voici la liste de quelques ajouts dans Qt 5 :

- QStandardPaths : permet de récupérer les répertoires par défaut en fonction de la plateforme. C'est une évolution de QDesktopServices avec plus de fonctionnalités, sur le modèle de KStandardDirs de KDE. Cela permet par exemple de faire une recherche de toutes les occurrences d'un fichier dans les différents répertoires ;
- support de JSON : permet de créer ou de lire un fichier JSON à partir d'une représentation binaire en mémoire ;
- extension prise en charge MIME : permet de déterminer le type mime d'un fichier ou de données en mémoire, en fonction de l'extension et/ou du contenu. Ce module utilise une base de données des types MIME par QMimeDatabase fourni par freedesktop.org shared-

mime-info project. Cette base de données est incluse par défaut dans le système sous Linux et fourni par Qt sur Windows et Mac OS X ;

- vérification des connexions signaux/slots à la compilation : vérifie l'existence du signal et du receveur et que les arguments sont compatibles. Cette fonctionnalité utilise les templates et est compatible avec C++11. Il est possible de connecter un signal à des fonctions lambda, des fonctions membres ou des fonctions statiques, sans avoir besoin de déclarer comme slots. Voir l'article détaillé sur le sujet : Les signaux et slots dans Qt 5 ;
- QRegularExpression : nouveau moteur d'expressions régulières compatible Perl, plus puissante et rapide que QRegExp, avec plus de fonctionnalités (lazy and possessive quantifiers, lookbehinds, named capturing groups and iteration of matches) ;
- amélioration des performances, en particulier pour les structures de données ;
- amélioration du support C++11 quand c'est possible (mais compatibilité avec C++98) ;
- support des boutons supplémentaires sur les souris (souris pour joueurs), jusque 27 boutons pour XCB, XLIB ou DirectFB, jusque 16 pour Wayland, Evdev ou OS-X, jusque 8 pour BlackBerry/QNX et 5 sur Windows (limitation due au système).

2.1.2 Le module Qt Gui

Ce module fournit les fonctionnalités de base pour créer une interface utilisateur. Les anciennes classes QWidget et dérivées sont séparées dans un module indépendant. Ce module contient de nouvelles classes comme QWindow, QScreen, QSurfaceFormat permettant le support de fonctionnalités de base et est surtout destiné à être utilisé par les autres modules (QWidget, QQuickView, Qt 3D View, etc.).

En particulier, ce module fournit les classes QOpenGLxxx (QOpenGLFramebufferObject, QOpenGLShaderProgram, QOpenGLFunctions, QOpenGLContext, etc.) permettant de fournir l'accélération matérielle pour tous les modules graphiques (widgets traditionnels, Qt Quick). La classe QOpenGLContext est plus générique que QGLContext et est découplée de QWindow, pour permettre d'utiliser un contexte commun pour plusieurs affichages. QOpenGLPaintDevice permet d'utiliser directement QPainter sur un contexte OpenGL sans devoir dériver de QWindow ou QOpenGLFramebufferObject.

Voir l'article OpenGL dans Qt 5 pour plus de détails.

2.1.3 Le module Qt QML

Ce module permet d'utiliser le langage de script déclaratif QML grâce au QML engine. Il présente des améliorations des performances et des ajouts de fonctionnalités par rapport celui inclus dans Qt 4.

2.1.4 Le module Qt Js backend (JavaScript)

Ce module fournit un interpréteur JavaScript, permettant de scripter les applications écrites en C++ et en QML. Il utilise un nouveau moteur JS v8 plus rapide. Il inclut de nouvelles classes (QJSEngine, QJSValue), le support de nouveaux types (QColor avec les propriétés r, g, b et a, QVector4D constructible avec Qt.vector4d()). Il est possible d'ajouter des fonctionnalités dans un namespace avec la fonction qmlRegisterModuleApi et d'importer du QML et du JS directement dans un fichier JS.

2.1.5 Le module Qt Quick

Ce module permet de créer des interfaces dynamiques riches, en utilisant les modules QML et le JS. Cette nouvelle version de Qt Quick correspond au module « qtquick2″ alors que l'ancienne version correspond au module « qtquick1″. L'interface graphique de Qt Quick 2 se base maintenant sur scenegraph et permet l'accélération matérielle en utilisant les classes QOpenGLxxx de Qt Gui.

On trouve de nouvelles classes (QQuickView, QQuickCanvas, QQuickItem et QQuickPaintedItem qui remplacent les classes équivalentes de QDeclarative), de nouveaux items (Canvas permet le support de l'API Context2D de HTML 5, le rendu est réalisé dans Canvas.Image et Canvas.FramebufferObject, avec support multithread en arrière-plan). Le moteur de particules 2D Qt Quick.Particles 2.0 et la collection d'effets de shaders qui étaient avant des projets séparés dans Qt Labs sont maintenant inclus dans Qt.

2.1.6 Le module Qt 3D

Le module Qt 3D est également un ancien projet provenant de Qt Labs et est inclus dans Qt 5. Il a permis dans Qt 4 l'ajout de nombreuses fonctionnalités de calculs 3D comme les classes QMatrix4×4, QGLShaderProgram et QVector3D. Il utilise en interne le module Qt QML et le support OpenGL de Qt Gui. Ce module contient deux bibliothèques : Qt 3D (pour utiliser directement la 3D en C++) et Qt 3D Quick (pour l'utilisation dans Qt Quick).

Plusieurs fonctionnalités sont ajoutées :

- gestion de scènes 3D, avec rendu en OpenGL ;
- lecture de fichiers 3D (par exemple .obj et .3ds) ;
- gestion des lumières, des meshs, des textures, des matériaux, des animations, des caméras, des vues ;
- ajout de shader directement ou par fichier dans les propriétés QML ;

2.1.7 Le module Qt Location

Ce module est un ajout dans Qt 5, mais il existait déjà depuis des années comme sous-ensemble de Qt Mobility. Il fournit les services nécessaires pour la localisation : GPS, cartographie, etc.

Il inclut une fonctionnalité permettant d'afficher des cartes avec MapQuickItem. L'affichage se base sur une approche modèle/vue et bénéficie de l'accélération OpenGL dans scenegraph. Les gestuelles pour les zooms et les panoramas dynamiques, le routage et le géocodage, l'ajout de repères sur les cartes sont pris en charge.

2.1.8 Le module Qt Network

Ce module fournit une interface portable pour utiliser les réseaux. Parmi les évolutions :

- amélioration du support IPv6 et des réseaux utilisant les types d'adresse IP, de manière transparente par défaut. En réception, QTcpServer et QUdpSocket lancés avec QHostAddress::Any permet de recevoir dans les deux modes ; avec QHostAddress::AnyIPv4 et QHostAddress::AnyIPv6 permet de travailler sur un mode uniquement. En émission, QNetworkAccessManager tente d'utiliser les deux modes et garde le premier qui réussit ;
- QTcpSocket peut maintenant être attaché à un socket existant avant de lancer une connexion, pour limiter les connexions dans un environnement multi-hôte ;
- QDnsLookup permet de rechercher des enregistrements DNS. Il ne remplace pas QHostInfo, qui permet de résoudre les noms en adresse IP, mais permet principalement d'utiliser les autres types d'enregistrements DNS : SRV, TXT et MX ;
- les classes QFtp et QHttp ne sont pas conservées dans ce module, mais restent disponibles dans un module indépendant pour la compatibilité. Elles sont remplacées par QNetworkAccessManager.
- extensions et vérifications des certificats SSL : prise en charge des extensions des certificats.

La vérification des certificats ne se fait plus uniquement lors de la connexion à un serveur ;

- support des clés privées masquées : permet de lire une clé privée à partir d'un périphérique, par exemple un dongle PKCS#11.

2.1.9 Les autres modules

- Qt Multimedia : fournit les fonctionnalités de base pour lire l'audio, la vidéo, la radio et gérer les caméras ;
- Qt SQL : fournit une prise en charge portable des bases de données SQL ;
- Qt Test : fournit les outils nécessaires pour implémenter des tests unitaires ;
- Qt WebKit : basé sur WebKit 2, mais sans changement de l'API C++. Ce module continue l'amélioration de prise en charge HTML 5 et des performances.

2.2 Les modules Add-ons

2.2.1 Le module Qt Widget

Ce module fournit l'ensemble des classes QWidget et dérivées pour la compatibilité avec Qt 4. Il utilise la nouvelle architecture Qt Platform Abstraction (QPA).

2.2.2 Le module Qt Quick 1

Ce module permet d'utiliser la version de Qt Quick disponible dans Qt 4, pour compatibilité. Pour utiliser ce module, il suffit d'ajouter dans le .pro :

```
QT += quick1
```

Et d'inclure les fichiers d'en-têtes :

```
#include QtQuick1/QDeclarativeView
#include QtQuick1/QDeclarativeItem
```

2.2.3 Les autres modules

- Qt Script : permet de rendre les applications scriptables, compatibilité avec Qt 4. Il utilise les classes QJSxxx du module Qt Js ;
- Qt Bluetooth : prise en charge du Bluetooth ;
- Qt D-Bus : inter-processus communication avec D-Bus ;
- Qt Graphical Effects : collection d'effets graphiques ;

- Qt Image Formats : prise en charge de formats d'images supplémentaires (TIFF, MNG, TGA, WBMP) ;
- Qt OpenGL : module 3D OpenGL compatible avec Qt 4 ;
- Qt Print Support : support pour l'impression ;
- Qt Publish and Subscribe ;
- Qt Script Tools : outils supplémentaires pour scripter ;
- Qt Sensor : gestion des capteurs (accéléromètre, détecteur de lumière ambiante, compas, etc.) ;
- Qt Service Framework : permet de fournir des services en ligne ;
- Qt SVG : prise en charge du format d'image SVG (image vectorielle) ;
- Qt System Info : informations sur le système (profile utilisateur, batterie, stockage, etc.) ;
- Qt Tools : outils divers (Qt Designer, Qt Help, etc.) ;
- Qt Pim : contacts, organiseur, vCard, etc. ;
- Qt WebKit Widgets : version de wekbit 1, pour compatibilité avec Qt 4 ;
- Qt XML : fichier XML avec SAX et DOM. Ce module est déprécié, il faut maintenant utiliser QXmlStreamReader/Writer ;
- Qt XML Patterns : support pour XPath, XQuery, XSLT et XML Schema validation.

2.3 Les modules accessoires

Le support de ces modules n'est pas encore déterminé.

- Active Qt : support des ActiveX et Com (Windows) ;
- Qt Feedback ;
- Qt JSON DB ;
- Phonon : support du framework phonon pour la vidéo et audio ;
- Qt QA : auto test, pour gestion automatique des tests ;
- Qt QLALR : interpréteur LALR.

2.4 Commentaires et remerciements

Merci à ClaudeLELOUP pour sa relecture orthographique.

3 Vidéos d'installation de Qt 5

Reprise de mon article publié le 11 mai 2013

Bonjour à tous

Un problème que l'on revoit souvent sur le forum est l'installation de Qt 5. J'ai réalisé, dans le cadre du livre sur Qt auquel je participe (Créer des applications avec Qt 5 – Les essentiels), plusieurs vidéos pour expliquer comment installer et tester l'installation.

Il y a quatre vidéos pour le moment :

- installation sur Windows avec Microsoft Visual C++ 2010 ;
- installation sur Windows avec MinGW ;
- installation sur Linux en utilisant les binaires ;
- installation sur Ubuntu en utilisant les dépôts.

Les vidéos sont disponibles sur le site de l'éditeur : D-booker.fr (YouTube).

Je n'ai pas fait de prise son, donc penser à activer les sous-titres pour avoir les explications. J'essaierais de proposer prochainement d'autres vidéos d'installation (sur Mac, sur Raspberry Pi) ou des démos.

Bonne visualisation (et bonne lecture à ceux qui liront le livre)