Revenir à la page principale
https://guillaumebelz.wordpress.com/category/android/

Cet article a été écrit en 2013 pour Qt 5.1, puis mis a jour régulièrement, en fonction des sorties de Qt. Il est actuellement a jour pour la version actuelle de Qt 5.5.

Installer Qt 5.5 pour Android

Vue d'ensemble

L'utilisation de Qt sur Android nécessite l'installation de plusieurs outils :

- la version de Qt 5.5 pour Android : ok, on s'y attendait un peu. Contient la bibliothèque complète et Qt Creator ;
- le kit de développement (SDK) pour Android : permet de créer des applications Java pour Android et un simulateur pour tester ses applications ;
- le kit de développement natif (NDK) pour Android : permet de créer des applications C++ pour Android ;
- l'environnement de développement de Java (JDK), qui contient également l'environnement d'exécution (JRE). Par exemple OpenJDK pour Linux ou Oracle Java pour Windows. Ces kits sont nécessaires pour faire tourner le SDK Android ;
- Apache Ant : pour faire beau. (bon ok, ce n'est pas que pour faire beau, ça sert au déploiement. Mais comme c'est un outil Java, je ne connais pas du tout).

Dans ce tutoriel, je vais montrer l'utilisation de Qt Creator, mais il doit être possible d'utiliser d'autres outils.

Installations

Le développement sur Android se fait classiquement en Java, en utilisant le kit de développement Android (*Android SDK*). Il est possible de développez en C++ sur Android en utilisant le kit de développement natif (*Android NDK*). C'est cette approche qui est utilisée par Qt pour fonctionner sous Android.

Pour simplifier le portage de Qt, celui-ci n'utilise pas ses propres outils de compilation et déploiement sur Android, mais utilise les outils fournit dans les kits de développement Android. Ce qui implique que pour développer avec Qt sur Android, il faut installer ces kits et le Java.

Installation de l'environnement de développement de Java

Pour commencer, il faut installer l'environnement de développement de Java, en version 6 ou plus récente. Vous pouvez tester si Java est installé avec la ligne de commande suivante :

```
java -version
```

Sur Ubuntu, j'ai utilisé les paquets fournis par la distribution. Il faut donc juste taper ces lignes de commande :

```
sudo apt-get install openjdk-8-jre
sudo apt-get install openjdk-8-jdk
```

Pour Windows, j'ai utilisé le paquet `jdk-8u51-windows-i586` téléchargé sur le site Oracle Java (attention de bien prendre la version correspondant à votre système : 32 ou 64 bits).

Pour faire fonctionner Ant correctement, il faut ajouter le chemin du JDK dans les variables d'environnement `PATH` et `JAVA_HOME` :

```
PATH : C:\Program Files\Java\jdk1.7.0_25\bin
```

```
JAVA_HOME : C:\Program Files\Java\jdk1.7.0_25
```

Installation d’Apache Ant

Sur Ubuntu, même méthode, j’utilise les paquets de la distribution :

```
sudo apt-get -u install ant
```

Pour tester l’installation d’Apache Ant, vous pouvez taper la ligne de commande suivante :

```
ant -version
```

Sur Windows, j'ai téléchargé le paquet `apache-ant-1.9.6-bin.zip` sur le site Apache Ant et je l'ai décompressé dans un répertoire de travail pour Android. Il faut ensuite ajouter le répertoire bin dans la variable d'environnement `PATH`. Par exemple :

```
PATH : D:\Developpement\Android\apache-ant-1.9.2\bin
```

Installation du SDK Android

Installation

L’installation du SDK Android est simple aussi… puisqu’il n’y a pas d’installation à faire. Vous devez simplement télécharger le kit de développement pour Android et décompresser l’archive. Ce SDK contient les outils de développement Android, des plateformes de compilation, l’éditeur Eclipse ADT (Android Developer Tools, configuré spécialement pour Android), un simulateur Android pour tester ses applications.

Pour installer d’autres outils ou faire les mises à jour (faites-le lors de la première utilisation, pour être sûr que tous les paquets sont à jour), vous

pouvez utiliser le SDK Manager. Celui-ci est disponible dans la racine du SDK, dans Eclipse ADT (dans Windows > Android SDK Manager) et sera disponible (après configuration) dans Qt Creator (dans Outils > Options... > Android > Démarrer le gestionnaire Android d'AVD).

Tester Eclipse

Comme je suis curieux et que j'ai envie de voir un peu comment fonctionne une application Java sur Android, j'ai testé Eclipse et Java sur Ubuntu.

Pour cela, j'ai simplement suivi le tutoriel : Building Your First App (il faut reconnaître que la documentation est très claire et simple à suivre, même pour un développeur C++ comme moi). Allez dans le répertoire d'installation du SDK Android puis dans le répertoire eclipse/. Celui-ci contient un binaire eclipse, que vous lancez.

Si c'est la première utilisation d'Eclipse, il va falloir faire quelques réglages de configuration. En premier le Workspace (l'espace de travail d'Eclipse, dans lequel il range les fichiers). Personnellement, j'ai accepté toutes les valeurs par défaut.

Une fois qu'Eclipse est lancé, créez un nouveau projet de type « Android Application Project » dans Fichier puis Créer un projet. Un dialogue permet de donner un nom à votre application (c'est le nom qui apparaîtra sur le téléphone), la version minimale d'Android à prendre en charge et la version Android de destination. J'ai laissé par défaut. Vous pouvez ensuite choisir l'icône de l'application et le type d'activité. Choisissez BlanckActivity, qui va créer une activité « hello world ». Après quelques minutes (je ne vais rien dire sur la lenteur d'Eclipse...), un projet est créé, il contient quelques fichiers et répertoires. Les plus importants sont :

- le fichier AndroidManifest.xml : contient les informations sur l'application, en particulier les versions minimales et ciblées de la plateforme ;
- le répertoire res/layout/ : contient les interfaces de l'application,

décrites dans des fichiers XML ;

le répertoire src/ : contient les sources Java du projet. Dans ce projet par défaut, il lance simplement une activité, qui affiche l'interface décrite dans le fichier XML de layouts/ ;

- les répertoires res/drawable-xxx/ : contiennent les images de l'application.

Comme c'est une application Android, vous ne pouvez pas simplement la lancer comme une application de Bureau classique. Il existe deux méthodes : utiliser un simulateur ou déployer sur un téléphone.

Tester l'application sur simulateur

Pour cela, il faut dans un premier temps créer un simulateur. Le SDK Android est fourni avec un gestionnaire, permettant de créer différentes configurations pour le simulateur et de tester l'application sur plusieurs types de téléphone. Allez dans le menu Windows puis Android Virtual Device Manager. Dans le dialogue, créez un nouveau périphérique en cliquant sur « New », puis configurez selon le type de téléphone que vous voulez tester.

Par exemple, comme j'ai un téléphone Samsung Galaxy S1, j'ai choisi un écran 4″ en 480*800. Après la création du périphérique, lancez-le. Lancez ensuite l'application en ouvrant le fichier java (dans src/) puis en cliquant sur le bouton Run. Un dialogue « Run As… » s'ouvre pour choisir comment exécuter l'application. Choisissez Android Application. Normalement, l'application devrait se lancer.

Tester l'application sur téléphone

Bon, un simulateur, c'est bien, mais on aimerait avoir l'application sur un vrai téléphone pour tester. Ce n'est pas très compliqué non plus, mais il va falloir faire une manipulation sur le téléphone. Celle-ci permet

d’activer les fonctionnalités de développement du téléphone, en particulier le Debug Mode USB et de déployer des applications via un câble USB.

Pour Android 4.2, allez dans les paramètres puis dans le menu « À propos » du téléphone. Cliquez plusieurs fois sur Numéro de build pour activer le mode développeur. Vous allez avoir un message confirmant le passage en mode Debug. Revenez ensuite à l’écran précédent et allez dans les options de développeur. Activez le mode Debug USB.

Pour les autres versions d’Android, vous pouvez consulter la page suivante : Run on a Real Device. Pour terminer, il suffit de brancher le téléphone sur l’ordinateur avec un câble USB puis de lancer l’application comme précédemment. L’application devrait se lancer sur le téléphone.

Installation du NDK Android

Rien de compliqué ici non plus, il n’y a pas d’installation à faire, il faut simplement décompresser l’archive.

Installation de Qt Android

Installation des binaires de Qt

Si vous n'avez pas encore installé Qt, le plus simple est d’utiliser l'installateur online de Qt 5. Si vous avez déjà installé, lancez l'outil MaintenanceTool (qui se trouve dans le répertoire d'installation de Qt. Le processus d’installation est classique, vérifier simplement que tous les binaires nécessaires pour Android seront installés (Android versions ARM ou x86).

Choix des paquets lors de l'installation de Qt (MaintenanceTool.exe)

Installation de Qt Creator 2.8

Pour Qt Creator, plusieurs versions sont disponibles

- le Qt SDK est fourni avec Qt Creator 2.7.2 ;
- les paquets en ligne (MaintenanceTool.exe) fournissent Qt Creator 2.7.0 ;
- Qt Creator 2.8 est disponible dans un binaire séparé sur la page de téléchargement de Qt.

Sous Ubuntu, lors de mes premiers tests, j'avais utilisé une version développeur de Qt Creator 2.8, compilé moi-même. Sous Windows, j'ai testé la version fournie avec le SDK (2.7.2) dans un premier temps. Cependant, lorsque j'entrais les informations sur les Android SDK et NDK, Qt Creator devenait très très lent. Si cela vous arrive et que vous souhaitez continuer à utiliser Qt Creator 2.7.2, vous pouvez supprimer la configuration d'Android, en modifiant le fichier de configuration de Qt Creator (situé dans C:\Users\votre_login\AppData\Roaming\QtProject\QtCreator.ini) et en recherchant "[AndroidConfigurations]". Si vous souhaitez continuer à tester le portage de Qt pour Android, je vous conseille d'installer Qt Creator 2.8 manuellement. Le problème de lenteur sera réglé.

Configuration de Qt Creator

Une fois l'installation finie, lancez Qt Creator. La première étape va être de configurer Android. Pour cela, allez dans le menu Outils puis Options. Dans la liste de gauche, allez dans Android, puis configurez les répertoires du SDK et du NDK. Cochez la case pour laisser Qt Creator créer (normalement) automatiquement les kits de compilation. Vérifiez sur les captures d'écran que vous avez bien la bonne configuration.

Exemple de configuration d'Android sur Ubuntu Exemple de configuration d'Android sur Ubuntu Exemple de configuration d'Android sur Windows

Exemple de configuration d'Android sur Windows Exemple de configuration des compilateurs sur Windows Exemple de configuration des compilateurs sur Windows Exemple de configuration des versions de Qt sur Windows Exemple de configuration des versions de Qt sur Windows Exemple de configuration des kits sur Windows Exemple de configuration des kits sur Windows

Il faut ensuite créer un périphérique pour le simulateur si ce n'est pas encore fait. Pour cela, cliquez sur le bouton Start Android AVD Manager.

avd manager Liste des périphériques virtuels disponibles

Cliquez sur New pour créer un nouveau périphérique. Choisissez le type de Device (pour simuler mon Galaxy S, j'ai donc choisi un 4″ en 480*800). N'oubliez pas d'activer la prise en charge du GPU (Use Host GPU) pour que Qt puisse utiliser OpenGL ES.

new device Exemple de configuration d'un nouveau périphérique virtuel (AVD)

Pour créer une application Qt, allez dans le menu Fichier puis Nouveau projet. Choisissez un projet compatible avec Android, par exemple « Application Graphique Qt » ou « Application Qt Quick 2 (élément de base) ». Vérifiez que le type de projet Qt choisi supporte bien Android (voir dans le cadre de droite "Plateforme supportée")

Création d'un nouveau projet "Qt Quick 2 Application" Création d'un nouveau projet "Qt Quick 2 Application"

Vous pouvez tester l'application créée avec Qt Desktop (donc sans utiliser le simulateur Android). Vous pouvez choisir la version du kit en cliquant bas à gauche sur l'icône d'ordinateur (juste au dessus du triangle vert).

Choix du kit Qt à utiliser pour compiler Choix du kit Qt à utiliser pour compiler

Pour lancer l'application, cliquez sur le triangle vert (Run) en bas à gauche.

new project Application par défaut, lancée en version Desktop

Choisissez ensuite un kit Qt pour Android, puis lancez l'application en cliquant sur Run. Dans un premier temps, le simulateur se lance (cela peut prendre quelques dizaines de secondes).

simulator Fenêtre d'accueil du simulateur Android

À la fin du déploiement, l'application se lance.

run simulator Application par défaut, lancée sur le simulateur Android Pour déployer sur un téléphone (préalablement passé en mode Debug USB, voir au début), il suffit simplement de le brancher, Qt Creator lancera alors l'application dessus.

photo Application par défaut, lancée sur un Galaxy S1

Conclusion

Rien de très compliqué finalement. Le déploiement est relativement simple une fois que les outils sont configurés. Reste plus qu'à tester les performances. ;)

Pour ceux qui sont intéressés par Qt Quick, je rappelle le livre auquel je participe : Créer des applications avec Qt 5 – Les essentiels. J'ai en particulier participé à la rédaction de Qt Quick. C'est encore une pré-version du livre final, mais il y a déjà pas mal de choses sur Qt Quick. Et n'hésitez pas à me dire s'il y a des points que vous souhaiteriez voir abordés dans ce livre.

Si vous êtes intéressé par Qt Quick sur mobile, j'ai fait une présentation le 5 juin à Paris : Introduction à Qt (YouTube).

Merci à prgasp77 et Winjerome pour leur relecture orthographique. Merci à rotoOm pour ses remarques sur la première version de ce tutoriel.

Mises à jour

- 23 mai 2013 : première version du tutoriel avec Qt 5.1 beta sur Ubuntu 13.04.
- 13 juillet 2013 : mise à jour pour Qt 5.1 (version finale) et Windows 7.
- 15 juillet 2013 : ajout précisions sur les variables d'environnement pour Java et Ant
- 19 juillet 2013 : complément d'article Qt Quick Controls sur Android
- 02 janvier 2015 : si un périphérique connecté est trouvé, mais que la version est “unknown”, voir Eclipse: Android Device Chooser - Unknown Android 2.3.4 Device
- 20 juillet 2015 : mise a jour pour Qt 5.5.0.

Revenir à la page principale