Revenir à la page principale

Déployer une application Qt

J'ai l'impression d'avoir répondu à cette question une bonne dizaine de fois ces deux dernières semaines. J'en parle également dans ma vidéo Installation et premier pas avec Qt 5.2 sur Windows, mais je crois qu'il faut que je détaille un peu plus.

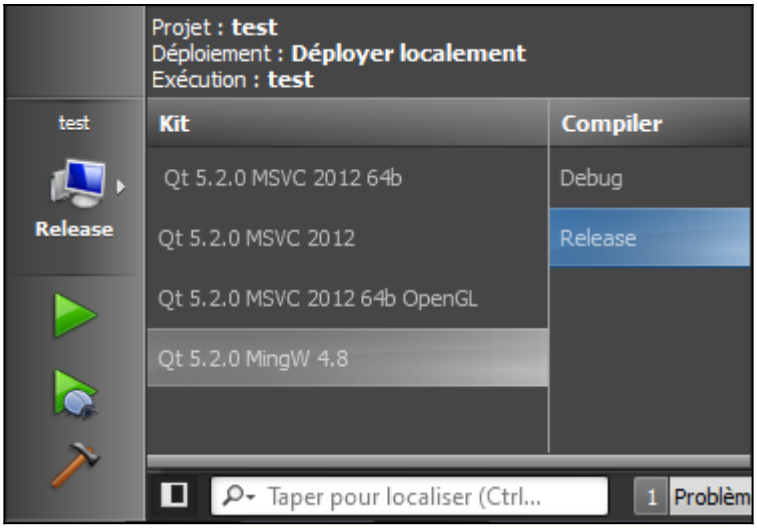
La problématique

Lorsque l'on lance l'exécution d'une application dans Qt Creator, celui-ci utilise le kit de compilation sélectionné pour lancer l'application. Cela nécessite aucune manipulation particulière. Par contre, si on lance l'application directement dans le navigateur de fichiers, on obtient un message d'erreur signalant qu'il manque des bibliothèques dynamiques (`.dll` dans Windows, `.so` dans Linux). Il faut donc fournir ces fichiers avec votre application pour pouvoir la lancer sur un autre ordinateur. Le plus simple est de copier (attention, “copier”, pas “déplacer” sinon vous ne pourrez plus exécuter vos applications dans Qt Creator) ces fichiers dans le répertoire de l'application. Il existe d'autres méthodes (paquets, dépôts, redistribuables, etc.), ça fera l'objet de plusieurs autres articles.

Pour commencer

La page d'aide de la documentation de Qt est la suivante : Deploying Qt Applications.

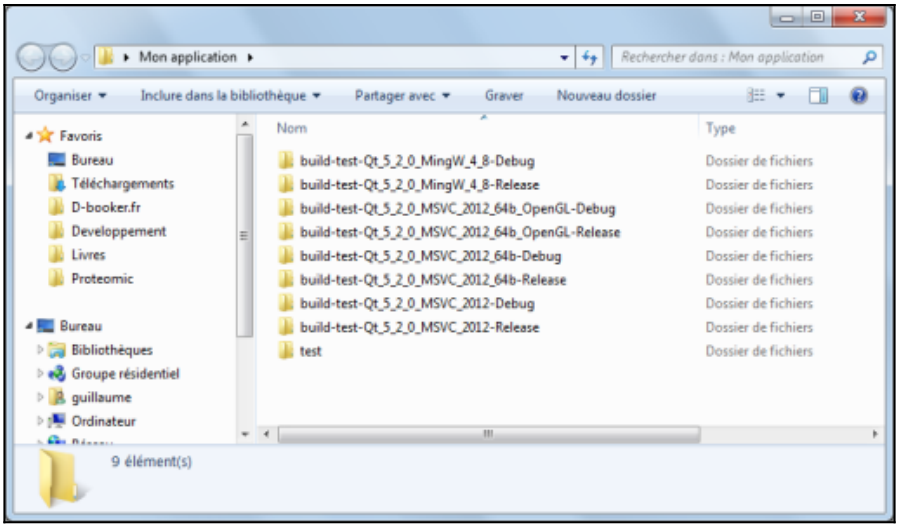
La première chose à faire est de compiler votre application en mode `Release`. Pour cela cliquez sur l'icône d'ordinateur en bas à gauche dans Qt Creator (au-dessus du triangle vert).



Choix du kit et du mode de compilation

L'application sera créée par défaut (shadow build) dans un répertoire spécifique, contenu dans le même répertoire que les sources. Le nom du répertoire dépend du kit utilisé et contient le terme “release”.

Par exemple, pour une application nommée “test” utilisant Qt 5.2 et compilée avec MingW 4.8 en mode release, le répertoire sera : `build-test-Qt_5_2_0_MingW_4_8-Release`.



Répertoires créés par les compilations selon les kits

Où trouver ces fichiers ?

Vous pouvez trouver les fichiers demandés dans plusieurs répertoires. Par exemple, sous Windows, dans le système (`C:\windows` par exemple) ou dans le répertoire de Qt Creator (`C:\Qt\Tools\QtCreator\bin` pour une installation de Qt par défaut).

Il ne faut surtout pas utiliser ces fichiers !

Il faut utiliser les mêmes fichiers que ceux utilisés pour la compilation. Une compilation est définie par un kit et il existe plusieurs kits possible (selon votre installation de Qt). Un kit est défini par :

- le compilateur : MingW ou Microsoft Visual C++ (MSVC) sur Windows, GCC sur Linux ;
- la version de compilateur : 4.7 ou 4.8 pour MingW/GCC (ou antérieur si vous utilisez Qt 4), 2011 (MSVC 10) ou 2012 (MSVC 11) pour Visual C++ ;
- 32 ou 64 bits ;

- DirectX ou OpenGL (uniquement pour Visual C++).
- la version de Qt : Qt 5.2, Qt 5.1.1, etc.

Il faut bien identifier le kit que vous utilisez pour la compilation, les fichiers nécessaires pour l'exécution en dépend. Ces fichiers sont dans le répertoire suivant pour une installation de Qt par défaut : `C:\Qt\version_Qt\version_compilateur\bin`. Par exemple, pour Qt 5.2 avec MingW 4.8 32 bits, le répertoire est : `C:\Qt\5.2.0\mingw48_32\bin`.

Certains modules de Qt nécessite l'utilisation de plugins, par exemple pour les images (JPEG, GIF, SVG, etc.) ou les bases de données. Les fichiers se trouvent dans le répertoire `C:\Qt\version_Qt\version_compilateur\plugins\nom_module`. Par exemple, pour les images avec Qt 5.2 et MingW 4.8 32 bits, les fichiers sont dans `C:\Qt\5.2.0\mingw48_32\plugins\imageformats`.

Quels sont les fichiers à copier ?

Cela dépend également du kit et des modules Qt utilisés.

Avec MingW :

- `libgcc_s_dw2-1` ;
- `libstdc++-6` ;
- `libwinpthread-1` (pour Windows uniquement).

Avec Microsoft Visual C++ :

- `msvcr100.dll` (MSVC 10) ou msvcr110.dll (MSVC 11). Ces fichiers sont dans le système (C:\windows\system32), ils ne sont pas installés par Qt, mais par le SDK de Microsoft Visual C++. Pour des raisons de licences, il n'est pas autorisé de donner directement ce fichier (je crois), il faut donner le redistribuable de Visual C++ (pour 2010 ou 2012).

Si vous utilisez DirectX (c'est-à-dire les versions Visual C++ non

OpenGL), il faut la bibliothèque ANGLE :

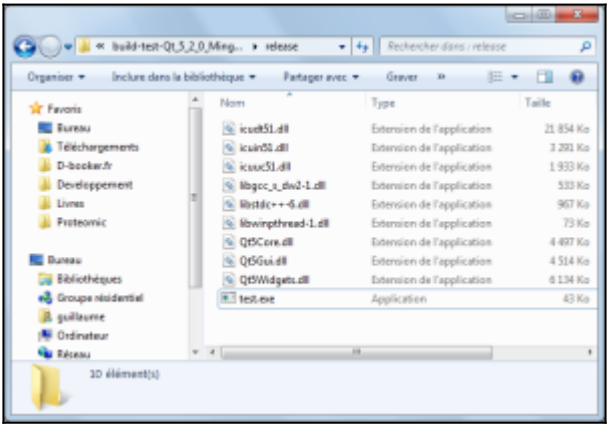
- `libEGL.dll` ;
- `libGLESv2.dll` ;
- `d3dcompiler_XX.dll` (dans `C:\windows\system32`, sera fournit par Windows en général).

Le support de l'Unicode avec ICU (version 51 chez moi) :

- `icudt51` ;
- `icuin51` ;
- `icuuc51`.

Pour les modules Qt, il faudra :

- `Qt5Core` au minimum ;
- `Qt5Gui` si vous créez une application graphique ;
- `QtWidgets` pour une application C++/widgets ;
- `QtNetwork`, `Qt5Qml` et `Qt5Quick` pour les applications Qt Quick ;
- pour chaque module Qt : `Qt5Xxx`, avec Xxx le nom du module (`Qt5Multimedia` pour le module multimédia, `Qt5OpenGL` pour la 3D avec OpenGL, `Qt5Webkit` pour Webkit, etc.)



Liste finale des fichiers nécessaires

Les fichiers de plugins

L'une des plus grosses évolutions de Qt 5 par rapport à Qt 4 est la création du Qt Platform Abstraction (QPA, voir là ou là). Ce changement, peu visible pour l'utilisateur lambda puisqu'il s'agit d'un changement d'architecture interne à Qt, a permis de rassembler toutes les fonctionnalités spécifiques à une plateforme dans un plugin. Tout le reste du code de Qt est donc indépendant de la plateforme. Cette approche permet de porter facilement Qt sur différentes plateformes, il suffit “simplement” de créer un nouveau plugin.

Si j'en parle dans cet article, c'est que cela va avoir une importance pour le déploiement. Il va falloir fournir les fichiers du plugin correspondant à votre plateforme. Ces fichiers se trouvent dans le répertoire `C:\Qt\version_Qt\version_compilateur\plugins\platforms`. Il faut mettre ces fichiers dans un sous-répertoire `platforms` dans le répertoire de votre application.

Pour Windows, il faudra copier le fichier qwindows.dll. Pour Linux et Android, il n'y a pas d'autres fichiers à copier. Pour les autres plateformes, je ne sais pas, n'hésitez pas à tester et à vérifier dans la documentation.

Si vous utiliser des formats d'images, il faudra aussi vérifier que vous n'avez pas besoin d'ajouter les plugins correspondant à ces formats. Ces fichiers se trouvent dans le répertoire `plugins/imageformats`. C'est le cas par exemple si vous utiliser des images au format JPEG ou SVG.

Plus généralement, il faudra vérifier dans le répertoire `plugins` s'il ne faut pas ajouter les fichiers correspondant aux plugins que vous utilisez dans votre application. Ça sera peut-être le cas si vous utiliser une base de données (sous-répertoire `sqldrivers`), les capteurs (dans `sensors`), l'audio, la vidéo, etc.

La difficulté avec les plugins est que si vous en oubliez un, cela ne produit pas un message d'erreur clair, du type “il manque tel plugin”. Vous aurez un simple message indiquant que le programme s'est terminé de façon inattendu.

Pour terminer

Vous pouvez supprimer les fichiers temporaires de compilation (`xxx.o` et `moc_xxx.cpp`).

Si vous rencontrez des problèmes avec le déploiement, vous pouvez expliquer vos difficultés sur le forum QtFr.org, en précisant :

- le kit utilisé ;
- où vous avez trouvé vos fichiers ;
- les modules Qt utilisés.

En complément : The Windows Deployment Tool.

Revenir à la page principale