

# Installation et premiers pas avec Qt 5.5

## [Télécharger le pdf](#)

Pour un débutant, l'installation et la configuration d'un environnement de développement pour le C++ et Qt posent régulièrement des problèmes. Le processus est relativement simple et automatisé, mais encore faut-il avoir une idée de ce que l'on fait. Le but de cette série de tutoriels est de montrer pas à pas comment installer et tester l'installation de Qt.

Ce tutoriel est régulièrement mis à jour, en fonction des sorties des nouvelles versions de Qt.

La première partie est consacrée à l'installation de Qt et la création de projets par défaut, pour vérifier que l'installation s'est bien passée.

Dans la seconde partie, j'explique un peu plus en détail le fonctionnement de Qt et de son installation, pour aider ceux qui rencontrent des problèmes lors de l'installation.

Pour terminer, je décris comment utiliser la documentation de Qt, qui est très bien faite et très riche. En particulier, il existe des codes d'exemple pour la majorité des fonctionnalités de Qt. Quand vous souhaitez réaliser quelque chose, la première chose à faire est probablement d'étudier ces codes d'exemple.

1. [Désinstaller proprement Qt et Qt Creator](#)
2. [Télécharger l'installateur de Qt 5.5](#)
3. [Installer Qt 5.5 sous Windows](#)
4. [Installer Qt 5.5 sous Linux](#)
5. [Installer Qt 5.5 sous Mac OS X](#)
6. [Installer Qt 5.5 pour Android](#)
7. [Installer Qt 5.5 pour iOS](#)
8. [install qt55 freebox](#)
9. [Tester l'installation de Qt 5.5](#)

10. Configurer Qt Creator 3.4.2 pour Qt 5.5
11. Mettre à jour Qt 5.5
12. Déployer une application Qt
13. Savoir utiliser la documentation de Qt 5.5

Ce tutoriel est une reprise et une mise à jour des vidéos que j'avais faites pour [mon livre](#) sur la chaîne YouTube de mon éditeur, D-Booker, à l'occasion de la sortie de Qt 5.0 : [Installer le framework Qt 5](#). C'est également une reprise des vidéos que j'avais faites à l'occasion de la sortie de Qt 5.2, sur ma chaîne YouTube : [Installation et premier pas avec Qt 5.2 sur Windows](#) et [Savoir utiliser la documentation de Qt](#). Je préfère passer au format tutoriel classique, pour faciliter les mises à jour.

[Revenir à la page principale du tutoriel](#)

# Désinstaller proprement Qt et Qt Creator

Il peut sembler étrange de commencer un tutoriel sur l'installation de Qt en expliquant comment désinstaller. La raison est que beaucoup ont tenté d'installer Qt avant de lire ce tutoriel et il est dans ce cas nécessaire de bien désinstaller Qt avant de le réinstaller,

La désinstallation en elle-même ne pose généralement pas de problème. Mais si vous aviez un problème de configuration de Qt Creator qui bloquait la compilation, il faut penser à supprimer les fichiers de configuration. Ces fichiers ne se trouvent pas dans le répertoire d'installation de Qt, ce qui explique qu'il est facile de les oublier (d'autant plus qu'ils sont dans un répertoire caché).

Sous Windows, il faut aller dans le répertoire `C:\Users\VotreNom\AppData\Roaming` et supprimer les répertoires `Qt` et/ou `QtProject`. Sous Linux, il faut aller dans `/home/guillaume/.config` et supprimer `QtProject`. Ces répertoires sont cachés, donc pensez à afficher les répertoires cachés (Ctrl+H dans l'explorateur de fichiers sous Linux, dans le menu Option > "Afficher les fichiers cachés" sous Windows).

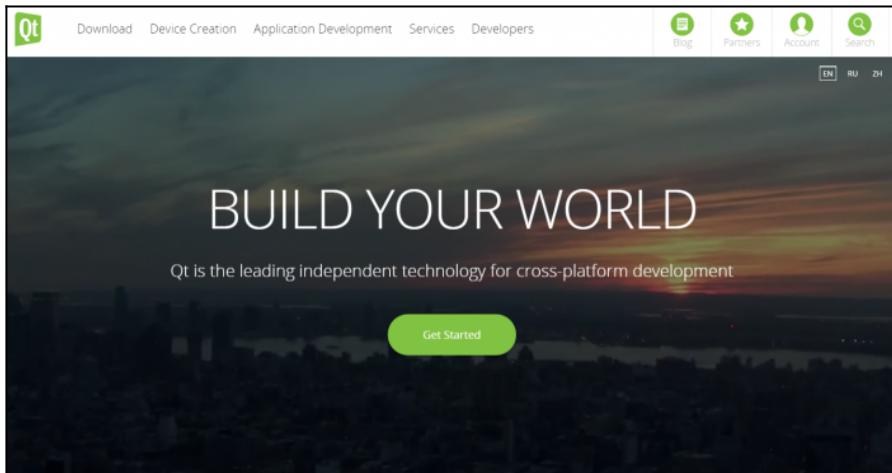
**Manque : sous MacOSX**

[Revenir à la page principale du tutoriel](#)

[Revenir à la page principale du tutoriel](#)

# Télécharger l'installateur de Qt 5.5

Le téléchargement de Qt se fait sur le site du projet Qt : <http://www.qt.io/>.



Cette page d'accueil contient une série de liens en haut, en particulier :

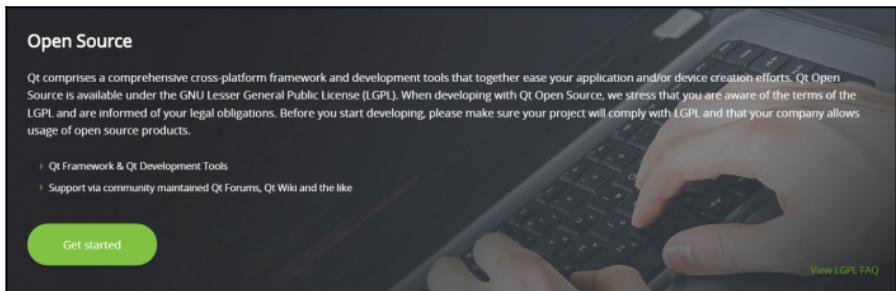
- pour télécharger Qt, allez sur le lien [Download](#) ;
- pour la documentation, allez sur le lien “Developers”, puis [Documentation](#).

La page de téléchargement contient plusieurs versions de Qt. Il faut savoir que Qt existe sous licence libre GPL/LGPL (qui permet de créer des programmes libres ou commerciaux, avec quelques conditions d'utilisation) ou sous licence commerciale (qui donnent plus de liberté).

Dans ce tutoriel, nous allons voir l'installation de la version gratuite de Qt, qui convient parfaitement pour débuter (et plus).

La page de téléchargement commence par une aide pour choisir la version de Qt à utiliser, puis les versions commerciales de Qt. Allez

directement à la version “Open-Source” un peu plus bas et cliquez sur “Get started”.



Cela permet d'aller sur la page de téléchargement : <http://www.qt.io/download-open-source/>.

Le téléchargement, le choix des outils et versions, l'installation et la mise à jour de Qt utilise un outil dédié, le *Qt Maintenance Tool*. Cet outil peut fonctionner selon deux modes :

- la version en ligne, qui contient uniquement l'installateur. Tous les outils seront installés après téléchargement en ligne ;
- la version hors ligne, qui contient l'installateur et une seule de version de Qt. Le fichier à télécharger est plus important, mais permet ensuite d'installer une version de Qt sans avoir de connexion Internet.

Dans tous les cas, une fois que vous avez installé une première fois Qt, vous pouvez directement installer de nouvelles versions de Qt ou en mettre à jour en lançant le *Qt Maintenance Tool*, qui se trouve dans le répertoire d'installation de Qt (par défaut : `C:\Qt\maintenancetool.exe` sous Windows et `/home/name/Qt/maintenancetool` sous Linux).

Dans ce tutoriel, nous allons utiliser l'installateur en ligne. La page de téléchargement propose par défaut l'installateur en ligne correspondant à votre système. Cliquez sur “Download Now” pour lancer le téléchargement (c'est relativement rapide pour cette étape).

The screenshot shows the Qt website's download section. At the top, there's a navigation bar with links for Product, Services, Qt in Use, Developers, Licensing, and Download. To the right of the navigation are icons for Blog, Partners, Sign in, and Search, along with language switches for EN, RU, and ZH. Below the navigation, a large banner features the text "Download Qt Community". Underneath the banner, there's a sub-navigation bar with links for Qt Online Installers, Qt 5.3, Qt Creator, Other Downloads, and Pre-releases. The main content area is titled "Recommended" and includes a message about detecting the operating system as Windows and recommending the "Qt Online Installer for Windows". A green "Download Now" button is present. Below this, there's a note about the online installer being a small executable that downloads content over the internet based on selections, providing all Qt 5.x binary & source packages and latest Qt Creator. It also links to the Developers page and All Downloads. To the right of the text, there are three images of devices displaying medical monitoring data (ECG, blood pressure, etc.) via the Qt interface.

Si vous souhaitez voir les autres téléchargements possibles, vous pouvez cliquer sur le lien “View All Downloads”. Cela permet d'afficher, dans l'ordre, les liens suivants :

- Les installateurs *online* pour tous les systèmes (linux 64b et 32b, Mac et Windows) ;
- Les installateurs *offline* pour Linux, Mac et Windows ;
- Les sources de Qt (Qt étant un projet libre, ses sources sont librement accessibles) ;
- L'éditeur Qt Creator ;
- Les autres outils :
  - les plugins pour Visual Studio ;
  - le [JOM](#) (outil de compilation similaire à nmake, dédié à Qt) ;
  - le [QBS](#) (Qt Build Suite, un outil de build, probable futur remplaçant de qmake) ;
  - [Qt Installer Framework](#) (pour créer vos propres installateurs pour vos programmes) ;
  - les dépôts Git de Qt ;

- les archives des anciennes versions de Qt (ce dernier lien permet en particulier de télécharger Qt 4 si nécessaire).

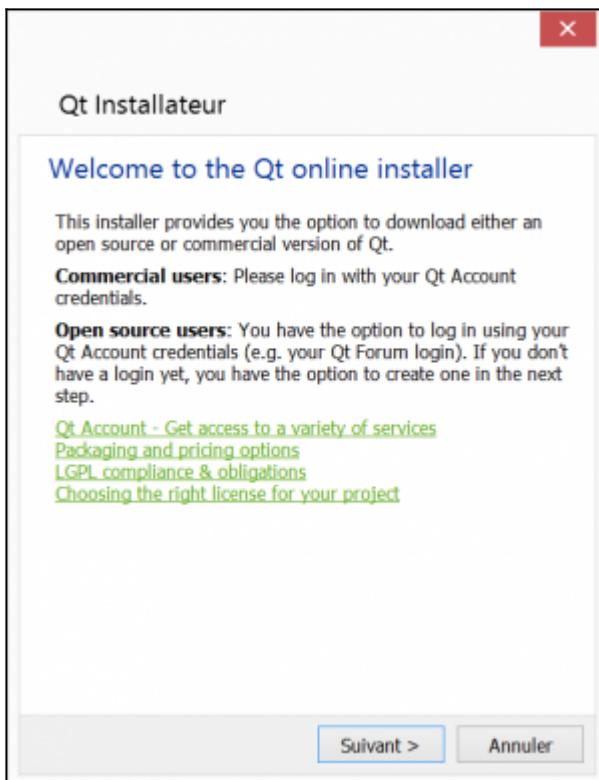
**[Revenir à la page principale du tutoriel](#)**

[Revenir à la page principale du tutoriel](#)

# Installer Qt 5.5 sous Windows

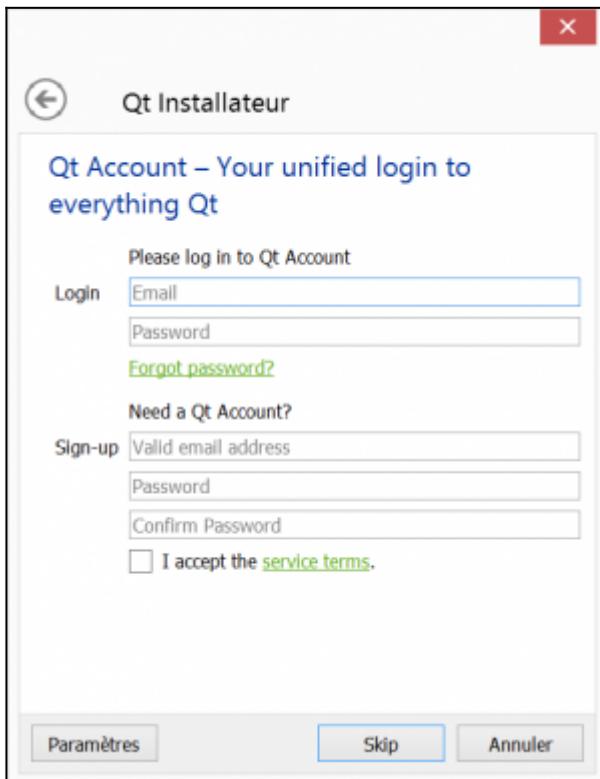
## Installer Qt 5.5 avec MinGW 4.9.2

L'installateur pour Windows s'appelle `qt-unified-windows-x86-2.0.2-1-online.exe`. Vous pouvez le lancer directement.



La première page permet de rappeler les différentes licences utilisables avec Qt (commerciale et open-source). Nous allons continuer avec la

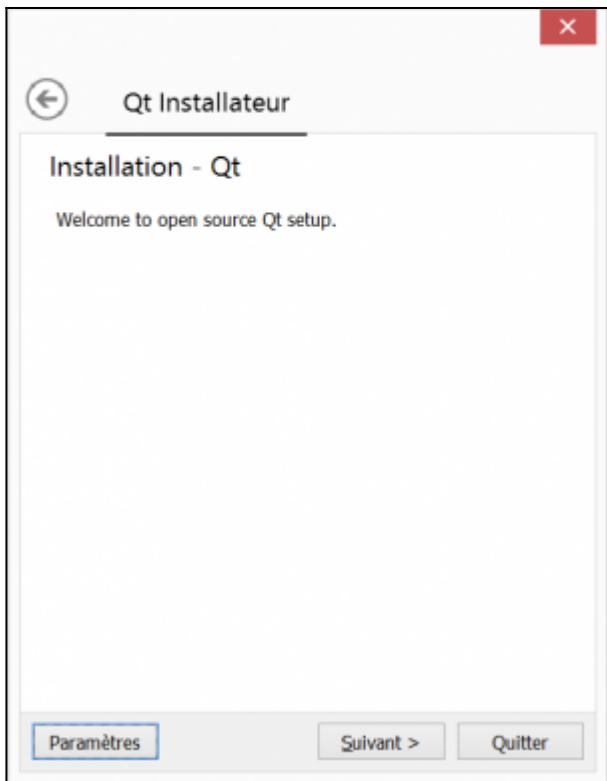
version open-source. Cliquez sur **Suivant**.



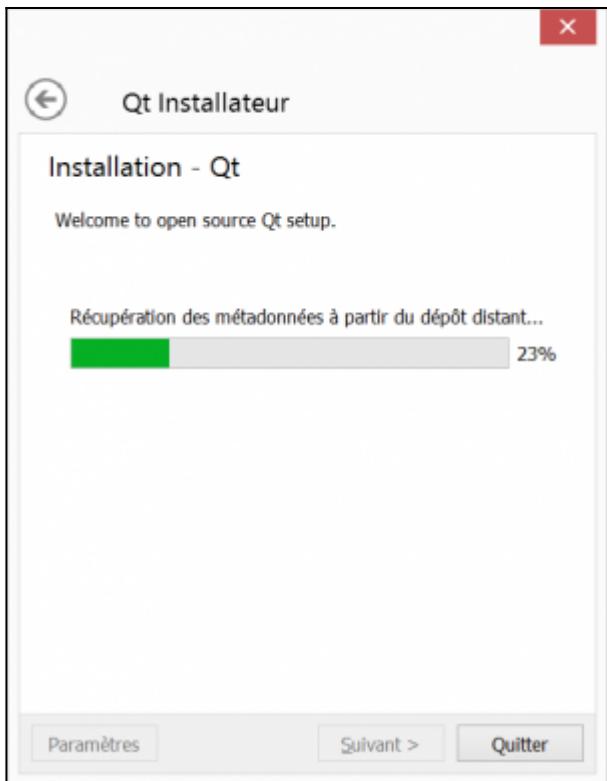
Le bouton **Paramètres** permet de configurer un serveur proxy si vous en utilisez un et d'ajouter des dépôts pour Qt. Normalement, vous n'avez pas besoin de modifier les dépôts, le dépôt principal de Qt est configuré par défaut. (Sauf si vous installez une version en cours de développement de Qt, avant sa sortie officielle).

Pour installer Qt, il est maintenant nécessaire de créer un compte. Si vous en avez déjà créé un sur le site <http://qt.io>, vous pouvez l'utiliser directement. Si ce n'est pas le cas, vous devez créer un compte en remplissant le formulaire.

Cliquez sur **Suivant** pour vous connecter.

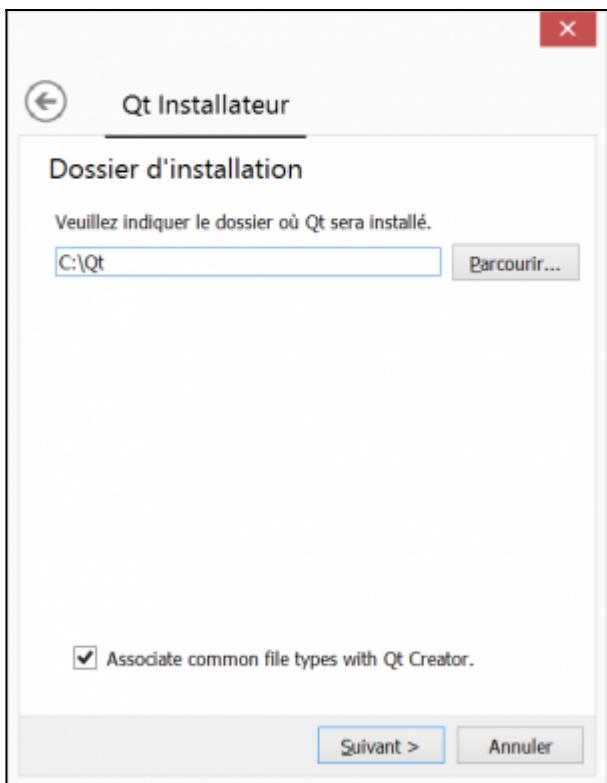


Cette page sert simplement à vous dire bienvenue... [Suivant.](#)



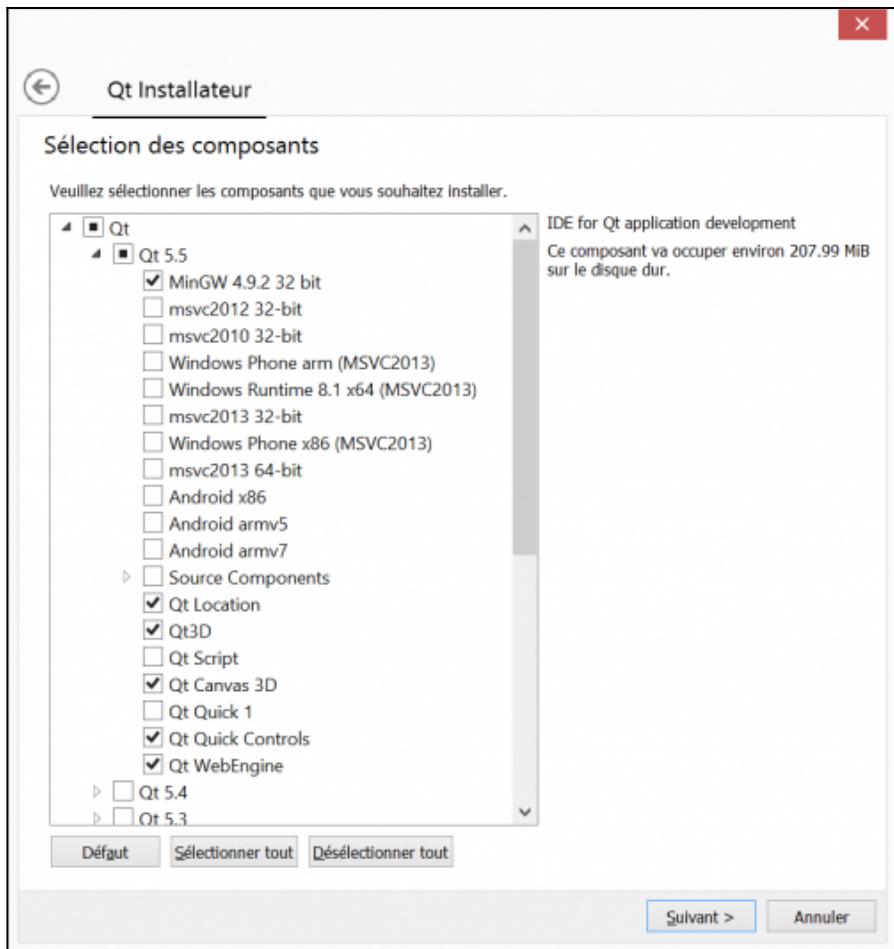
L'installateur recherche en ligne la liste des outils et versions que vous pouvez installer (cela peut durer de quelques dizaines de secondes à quelques minutes, en fonction de votre connexion internet).

Une fois que le téléchargement est fini, la page suivante s'affiche automatiquement. Cette page permet de choisir le dossier d'installation de Qt. Par défaut, le chemin est C:\Qt\.



Vous pouvez changer le répertoire d'installation si vous le souhaitez. Dans la suite de ce tutoriel, nous allons utiliser le chemin par défaut. Si vous changez de répertoire, pensez à adapter les chemins donnés dans la suite de ce tutoriel.

Cliquez sur Suivant.



La page suivante permet de sélectionner la liste des outils à installer. Pour programmer en C++ avec Qt, il faut installer au moins trois outils :

- un **éditeur**. Qt (le framework) est fourni avec un éditeur appelé Qt Creator (ne confondez pas l'éditeur et le framework). Cet éditeur est installé automatiquement et il n'est pas possible de le désactiver ;
- au moins une version de **Qt** ;
- un **compilateur** compatible avec la version de Qt installée.

Il existe sous Windows deux compilateurs principaux : MinGW (version Windows du compilateur [GCC](#) de Linux) et le compilateur de [Visual Studio](#) de Microsoft.

Pour des raisons de licences logicielles, l'installateur Qt ne fournit que le compilateur MinGW. Pour installer le compilateur de Visual Studio, il faudra installer séparément Visual Studio, directement à partir du site de Microsoft. La procédure est expliquée dans la partie “[Installer Qt 5.5 avec Visual Studio 2013](#)”.

L'installateur propose de nombreuses options, nous allons les détailler un peu.

Pour commencer, on peut voir que l'installateur permet d'installer plusieurs versions de Qt : 5.5, 5.4, 5.3, etc. Par défaut, nous allons utiliser la dernière version (5.5). Si vous devez travailler sur une version plus ancienne que Qt 5.5, vous pouvez l'installer à partir de cette page. (Remarque : il est encore possible d'installer Qt 4, mais la procédure est différente, nous n'allons pas voir cela.)

Pour chaque version de Qt proposée, il existe plusieurs versions et modules, selon le compilateur.

**Remarque importante : vous choisissez ici les versions de Qt à installer, pas les compilateurs. Chaque version de Qt est identifiée par un nom de compilateur, mais cela ne veut pas dire que le compilateur correspondant sera installé. Les compilateurs sont installés séparément, voir plus bas.**

- **MinGW 4.9.2 32 bit** : version de Qt compatible avec le compilateur MinGW 4.9.2. C'est la dernière version de MinGW (GCC est en version 5.1, MinGW est un peu en retard, mais le support du C++11 est complet sur cette version, donc vous pouvez l'utiliser sans problème). Il n'existe pas de version 64 bits, mais la version 32 bit fonctionne parfaitement sous Windows 32 bit et 64 bit ;
- **MSVC 2012 32 bit** : compilateur de Visual Studio 2012 ;
- **MSVC 2010 32 bit** : compilateur de Visual Studio 2010 ;

- `MSVC 2013 32 bit` et `MSVC 2013 64 bit` : compilateur de Visual Studio 2013 ;
- `Windows Runtime 8.1 x64 (MSVC2013)` : version pour Windows RT 8.1, avec le compilateur de Visual Studio 2013.

Et les versions pour mobiles :

- `Windows Phone arm (MSVC2013)` et `Windows Phone x86 (MSVC2013)` : versions pour Windows Phone, avec le compilateur de Visual Studio 2013 ;
- `Android x86 (MSVC2013)`, `Android armv5 (MSVC2013)` et `Android armv7 (MSVC2013)` : versions pour Android, avec le compilateur de Visual Studio 2013.

Comme vous pouvez le voir, Qt supporte de nombreux systèmes d'exploitation, c'est un peu compliqué de s'y retrouver. Dans ce chapitre, nous allons voir uniquement l'installation de Qt pour les ordinateurs de bureau (Desktop). L'installation de Qt pour mobiles sera vu de dans chapitres spécifiques. De plus, nous n'allons voir que l'installation des dernières versions des compilateurs :

- `MinGW 4.9.2 32 bit`, qui sera vu dans la suite de ce chapitre ;
- `MSVC2013`, qui sera vu dans la partie “Installer Qt 5.5 avec Visual Studio 2013”.

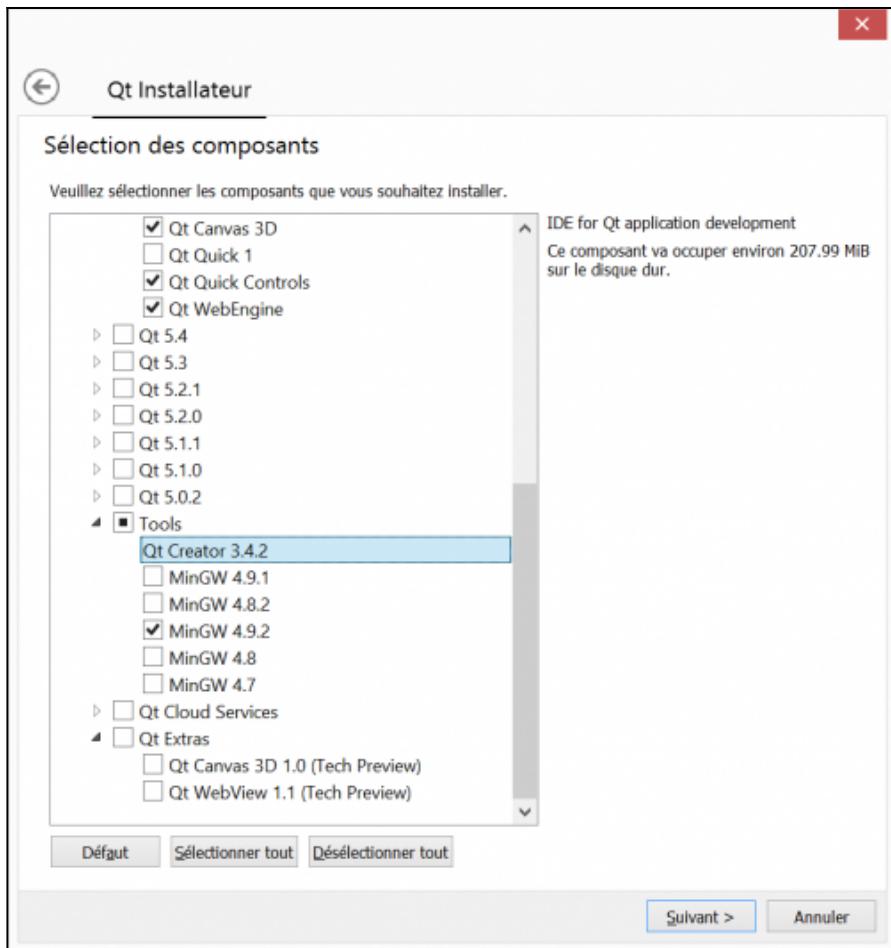
En dessous des différentes versions de Qt, vous pouvez choisir d'installer différents modules optionnels :

- `Source Components` : permet d'installer le code source de Qt. Si vous débutez, cela ne vous servira pas, mais si vous êtes un développeur C++/Qt un peu plus expérimenté, cela permet d'analyser comment Qt est conçu ;
- `Qt Location` : fonctionnalités de positionnement (GPS) ;
- `Qt 3D` : moteur 3D simple ;
- `Qt Script` : déprécié, ne pas utiliser (sauf besoin spécifique) ;
- `Qt Canvas 3D` : moteur 3D simple similaire à WebGL en JavaScript ;

- **Qt Quick 1** : déprécié, ne pas utiliser (sauf besoin spécifique) ;
- **Qt Quick Controls** : éléments d'interface graphique (bouton, dialogues, etc.) pour Qt Quick ;
- **Qt WebEngine** : moteur Web, pour afficher des pages internet.

Je vous conseille d'installer les modules correspondant à la capture d'écran précédente (Qt pour MinGW, Qt Location, Qt 3D, Qt Canvas 3D, Qt Quick Controls et Qt WebEngine. Vous n'aurez pas besoin de tous ces modules pour commencer, mais cela vous permettra d'explorer ce que propose Qt et de vous amuser un peu).

En dessous du choix des versions de Qt et des modules additionnels, vous pouvez sélectionner l'installation des outils externes.



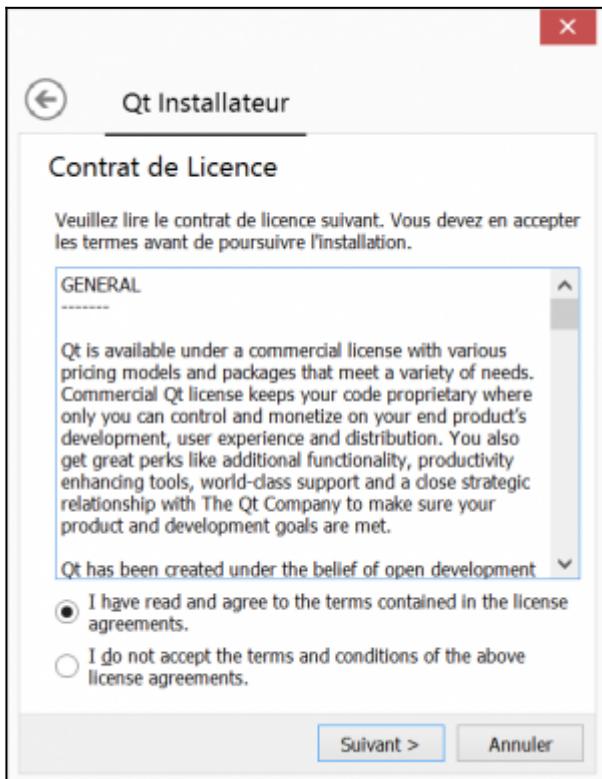
Qt Creator est l'éditeur fourni avec Qt. Il n'est pas possible de ne pas l'installer. (Même si vous souhaitez utiliser un autre éditeur, on utilisera Qt Creator dans ce tutoriel, pour tester si l'installation s'est bien passée).

Comme nous avons sélectionné Qt pour MinGW 4.9.2, il faut également installer le compilateur MinGW 4.9.2.

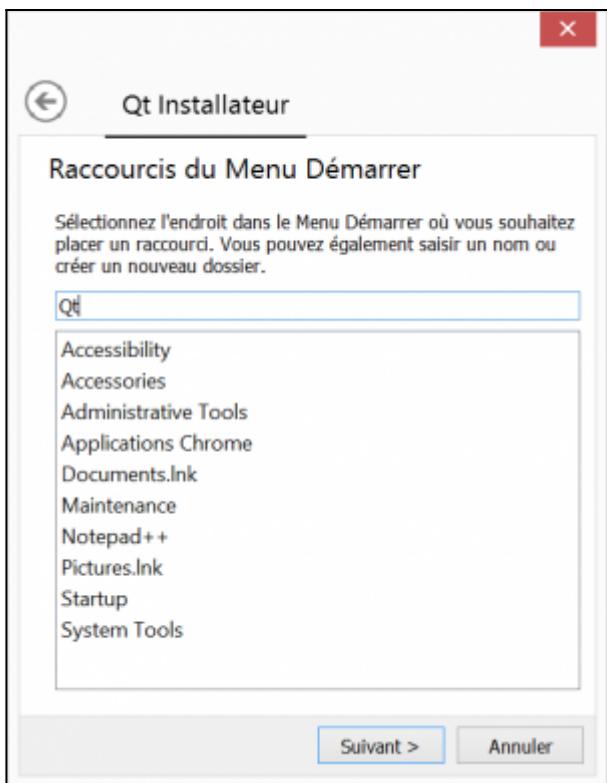
Il n'est pas nécessaire d'installer les modules **Qt Cloud Services** et **Qt Extras**.

Cliquez sur suivant lorsque vous avez sélectionné les modules qui vous

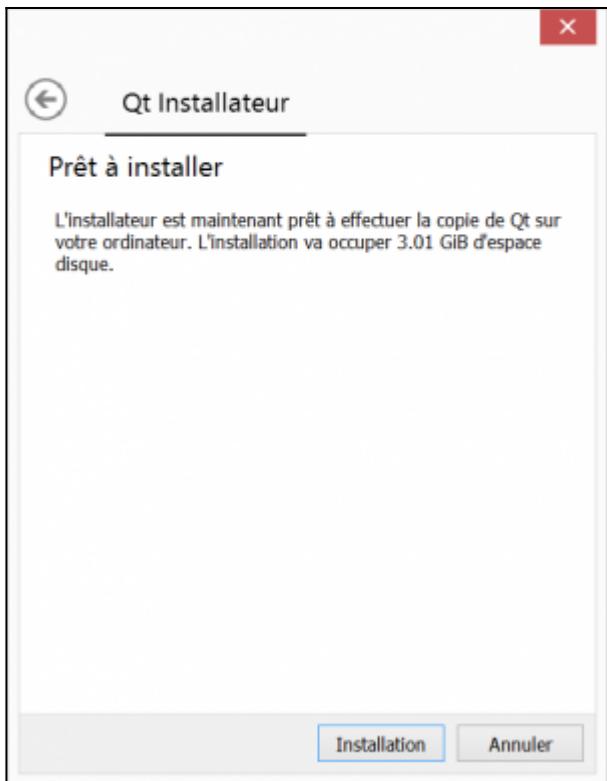
intéressent.



Cette page permet de valider les licences utilisateurs. Acceptez et cliquez sur suivant.

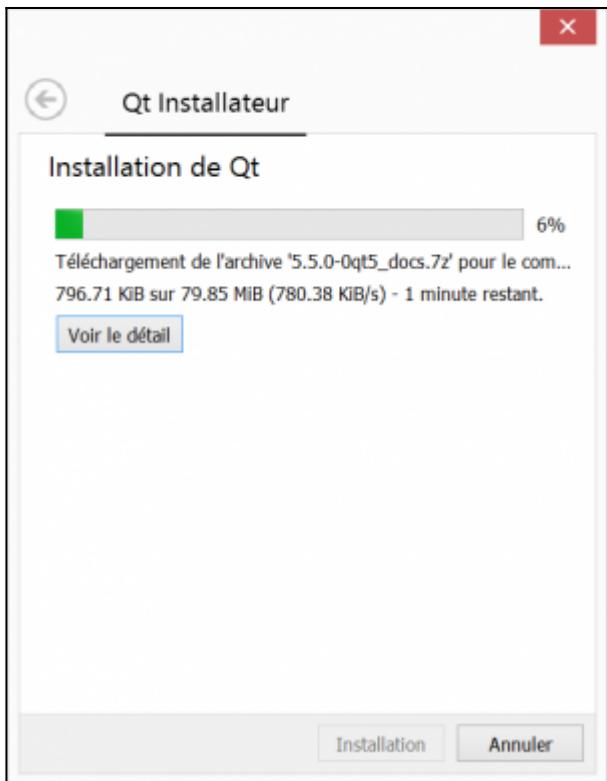


Cette page permet de choisir le répertoire dans le menu Démarrer.  
Cliquez sur suivant.

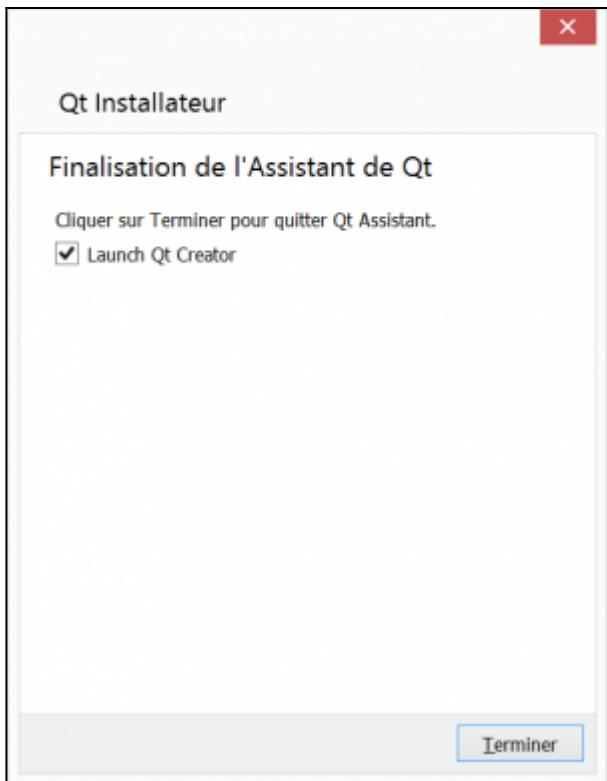


Une fois que tout cela est fait, l'installation est prête à démarrer. Cliquez sur [Installation](#).

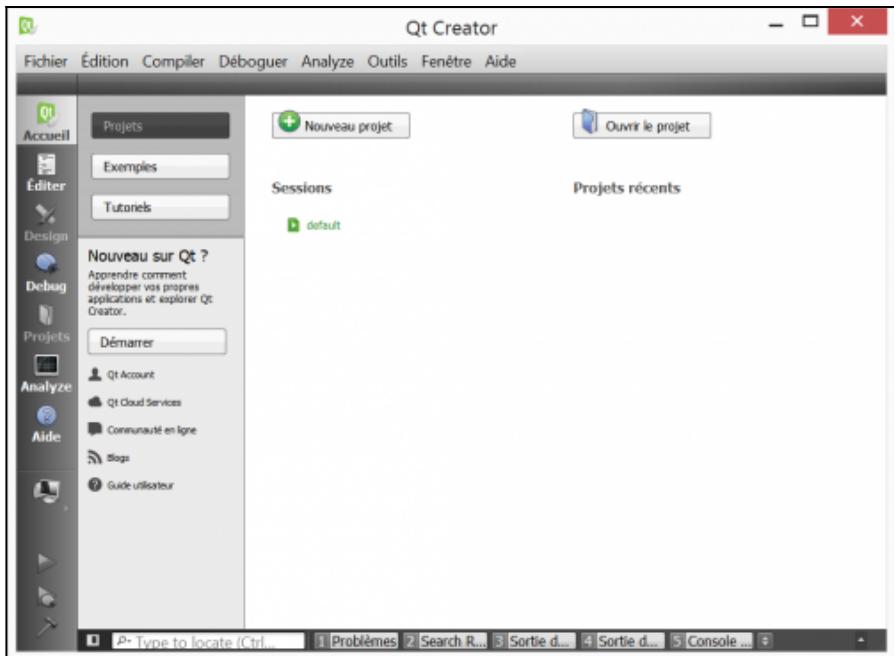
Le téléchargement puis l'installation se lancent. Selon votre connexion et le nombre de paquets que vous installez, cela peut prendre plusieurs minutes à plusieurs heures (si vous souhaitez installer beaucoup de paquets, il est probablement préférable de répéter l'installation plusieurs fois). L'installation sature le processeur, ne vous étonnez pas trop si Windows devient un peu lent pendant ce temps-là. Allez vous balader dehors, il fait beau (la pluie, c'est beau...).



Une fois que l'installation est terminée, la page suivante propose de lancer Qt Creator.



Cliquez sur **Terminer**.



Qt Creator s'ouvre et affiche la page d'accueil.

## Installer Qt 5.5 avec Visual Studio 2013

Il est également possible d'utiliser Qt avec le compilateur de Microsoft : Microsoft Visual Studio C++ (que l'on appelle en raccourci "MSVC"). L'installation de Qt pour Visual Studio est similaire à celle de MinGW, il faut simplement installer en plus :

- la version de Visual Studio correspondant à la version de Qt correspondante ;
- le plugin Qt dans Visual Studio (optionnel, il sert uniquement si vous souhaitez utiliser Visual Studio pour développer).

La première étape est donc d'installer la version de Qt correspondant à Visual Studio. Suivez les étapes décrites dans la première partie de ce tutoriel, jusqu'au choix des modules à installer. Dans la liste des versions de Qt, installez "msvc 2013 32-bit" et/ou "msvc 2013 64-bit". (Sachez

que la version 32-bit fonctionne sur Windows 32-bit et 64-bit, alors que la version 64-bit ne fonctionne que sur les Windows 64-bit. De plus, sauf contrainte particulière dans votre code, vous ne verrez pas de différence de performance entre les deux versions. Donc je vous conseille d'installer la version 32-bit). Terminez ensuite l'installation de façon classique.

La seconde étape consiste à installer Visual Studio. Il existe différentes versions de Visual Studio, identifiées par :

- la date de sortie : 2008, 2010, 2013, 2015 ;
- la licence d'utilisation : *Express* (version gratuite limitée), *Community* (version gratuite moins limitée) et les versions payantes (*Professional*, *Enterprise*, *Ultimate*...)

Les versions payantes sont relativement chères pour un particulier. Si votre entreprise possède des licences, utilisez-les. Sinon, la version *Community* est suffisante (nous allons utiliser cette version dans la suite de ce tutoriel).

La dernière version de Visual Studio est 2015, mais Qt n'est pas encore disponible pour ce compilateur (les binaires sont compatibles, mais il faut faire quelques manipulations. Autant attendre une version de Qt pour Visual Studio 2015, ça sera plus simple dans un premier temps).

Il faut rechercher un peu pour trouver le lien pour télécharger les anciennes versions de Visual Studio. Pour Visual Studio 2013 Community, vous pouvez utiliser ce lien : <http://go.microsoft.com/fwlink/?LinkId=517284>. Celui-ci permet de télécharger un installateur, qui s'appelle `vs_community.exe`. Vous pouvez le lancer.

*need screenshots...*

Suivez les instructions pour l'installation de Visual Studio. Cela peut prendre plusieurs minutes.

La dernière étape consiste à installer le plugin Qt dans Visual Studio. Cette étape est optionnelle, elle est utilisée que si vous utilisez l'éditeur de Visual Studio pour écrire vos programmes.

Pour bien comprendre ce que vous avez installé :

- vous avez installé une ou plusieurs versions de Qt (pour MinGW et/ou pour Visual Studio, pour Android, etc.) ;
- vous avez installé deux éditeurs de code : Qt Creator (qui est fourni dans Qt) et Visual Studio ;
- vous avez installé un ou plusieurs compilateurs (MinGW et/ou le compilateur de Visual Studio).

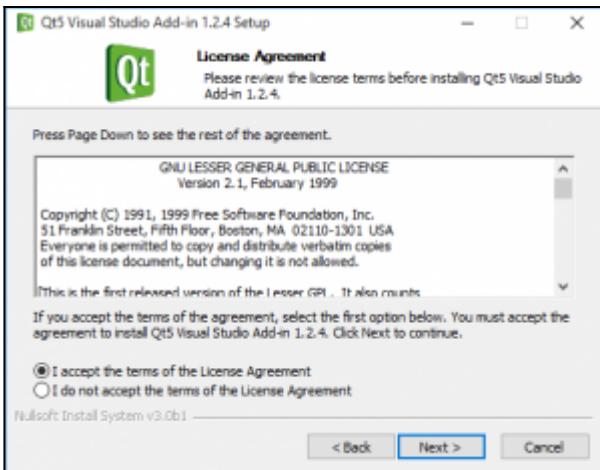
Pour utiliser Qt (avec MinGW et Visual Studio) dans Qt Creator, vous n'avez pas besoin d'installer le plugin Qt pour Visual Studio. Qt Creator reconnaît et configure automatiquement les différentes versions de Qt et les différents compilateurs installés. Le plugin ne sert que si vous utilisez l'éditeur Visual Studio.

(Sauf si vous avez besoin ou envie d'utiliser spécifiquement Visual Studio, je vous conseille d'utiliser Qt Creator. Il est spécifiquement conçu pour travailler avec Qt et permet de gérer facilement les différentes versions de Qt, en particulier les versions pour mobiles.)

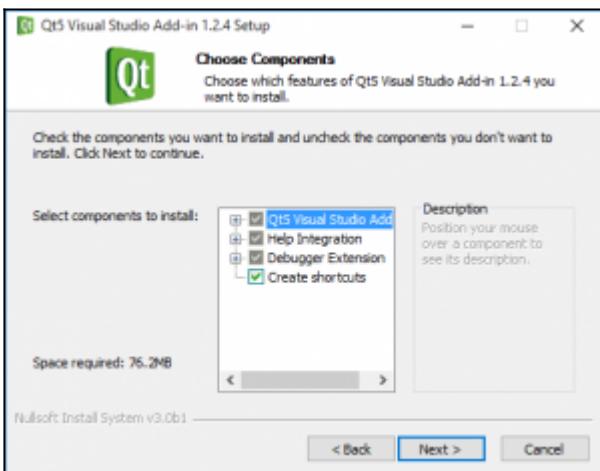
Commencez par télécharger le plugin Qt sur ce lien : [qt-vs-addin-1.2.4-opensource.exe](http://qt-vs-addin-1.2.4-opensource.exe) et lancez-le.



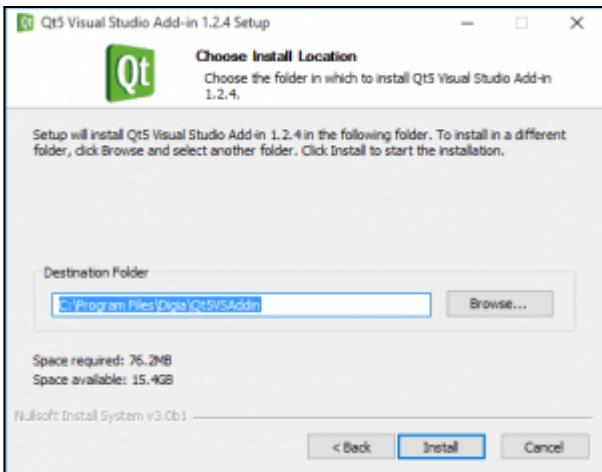
La première page est une simple page d'information. Cliquez sur **Next**.



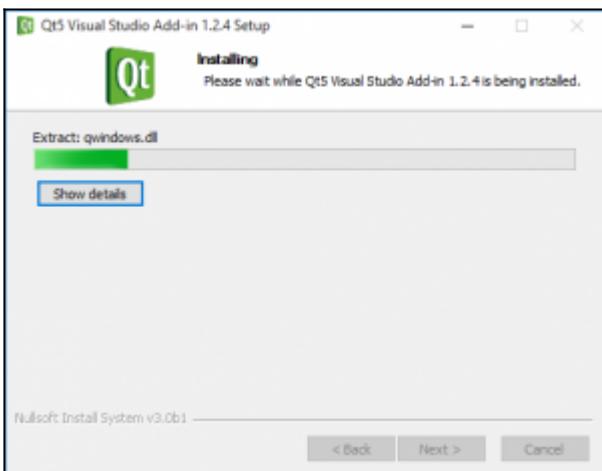
Cette page permet d'accepter la licence d'utilisation du plugin. Cliquez sur I accepte..., puis sur Next.



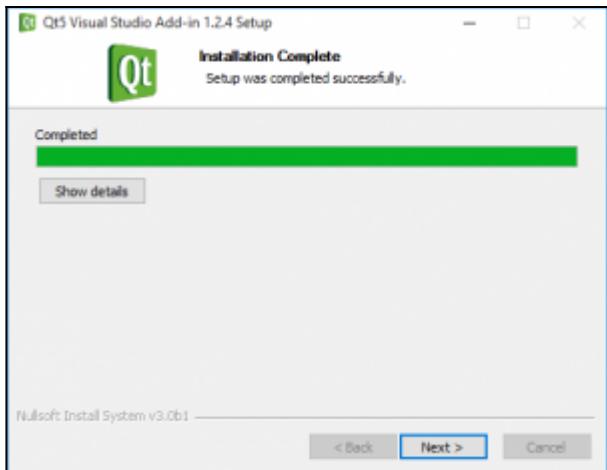
Cette page permet de choisir les versions de Visual Studio à installer. Choisissez les versions 2013 pour le plugin, l'aide et le débogage. Cliquez sur Next.



Cette page permet de choisir le dossier d'installation. Utilisez le répertoire par défaut et cliquez sur **Next**.



Le téléchargement, puis l'installation se lancent. Cela peut prendre quelques minutes, en fonction de votre connexion. (La taille des fichiers téléchargés est beaucoup plus petite que ceux téléchargés pour Qt, donc cela prend beaucoup moins de temps).



Une fois que l'installation des fichiers est terminée, cliquez sur **Next**.



L'installation est terminée, vous pouvez fermer l'installateur en cliquant sur **Finish**.

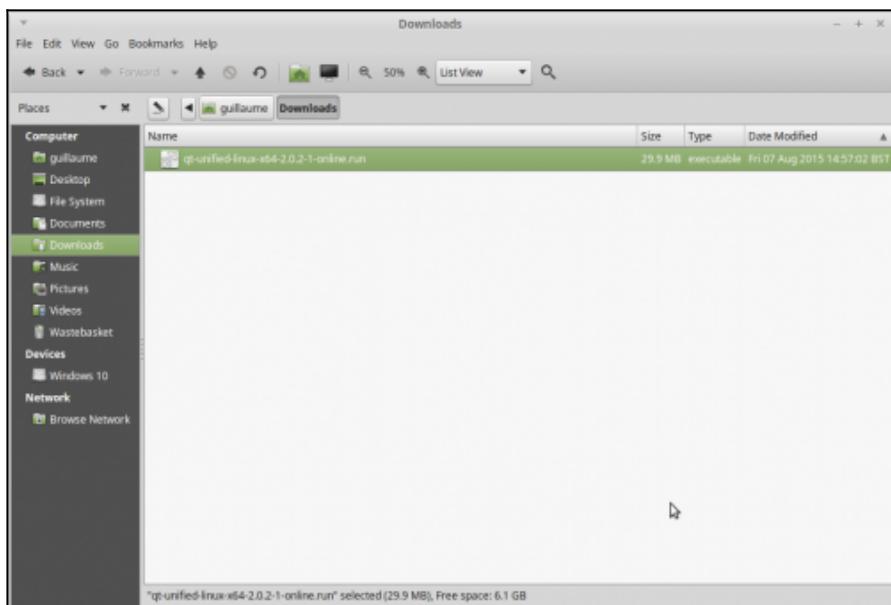
[\*\*Revenir à la page principale du tutoriel\*\*](#)

[Revenir à la page principale du tutoriel](#)

# Installer Qt 5.5 sous Linux

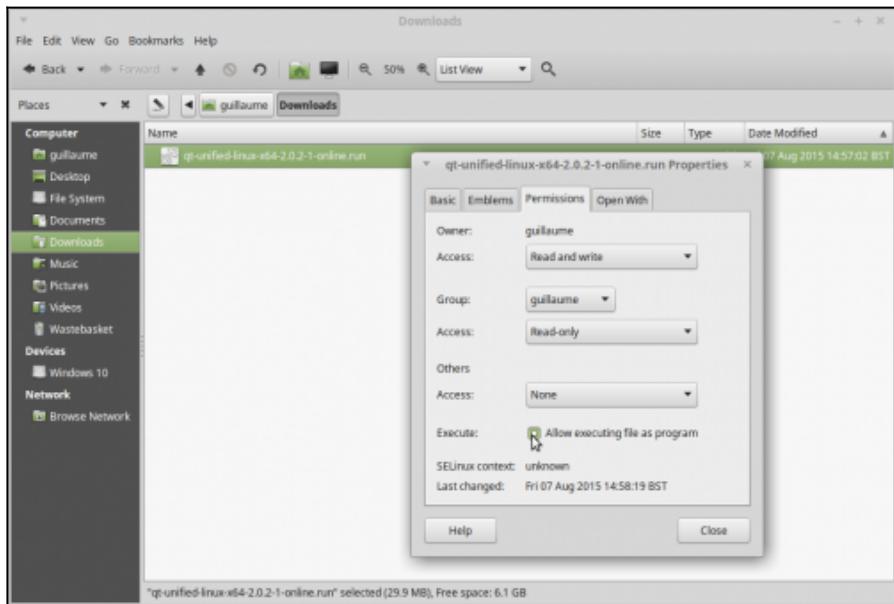
## Utiliser l'installateur

L'installateur sous Linux s'appelle `qt-unified-linux-x86-2.0.2-1-online.run` (pour la version 32-bit) ou `qt-unified-linux-x64-2.0.2-1-online` (pour la version 64-bit). Par défaut, ce programme n'est pas exécutable, la première chose à faire est donc de modifier ses propriétés.



Cliquez avec le bouton droit de la souris et choisissez "Propriétés". Dans l'onglet "Permission", sélectionner "Allow executing file as program" (ou équivalent si vous avez une version en français). Vous pouvez également utiliser `chmod` en ligne de commande :

```
chmod +x qt-unified-linux-x86-2.0.2-1-online.run
```



Vous pouvez ensuite lancer l'application.



La première page permet de rappeler les différentes licences utilisables avec Qt (commerciale et open-source). Nous allons continuer avec la version open-source. Cliquez sur **Suivant**.



Le bouton **Paramètres** permet de configurer un serveur proxy si vous en utilisez un et d'ajouter des dépôts pour Qt. Normalement, vous n'avez pas besoin de modifier les dépôts, le dépôt principal de Qt est configuré par défaut. (Sauf si vous installez une version en cours de développement de Qt, avant sa sortie officielle).

Pour installer Qt, il est maintenant nécessaire de créer un compte. Si vous en avez déjà créé un sur le site <http://qt.io>, vous pouvez l'utiliser directement. Si ce n'est pas le cas, vous devez créer un compte en remplissant le formulaire.

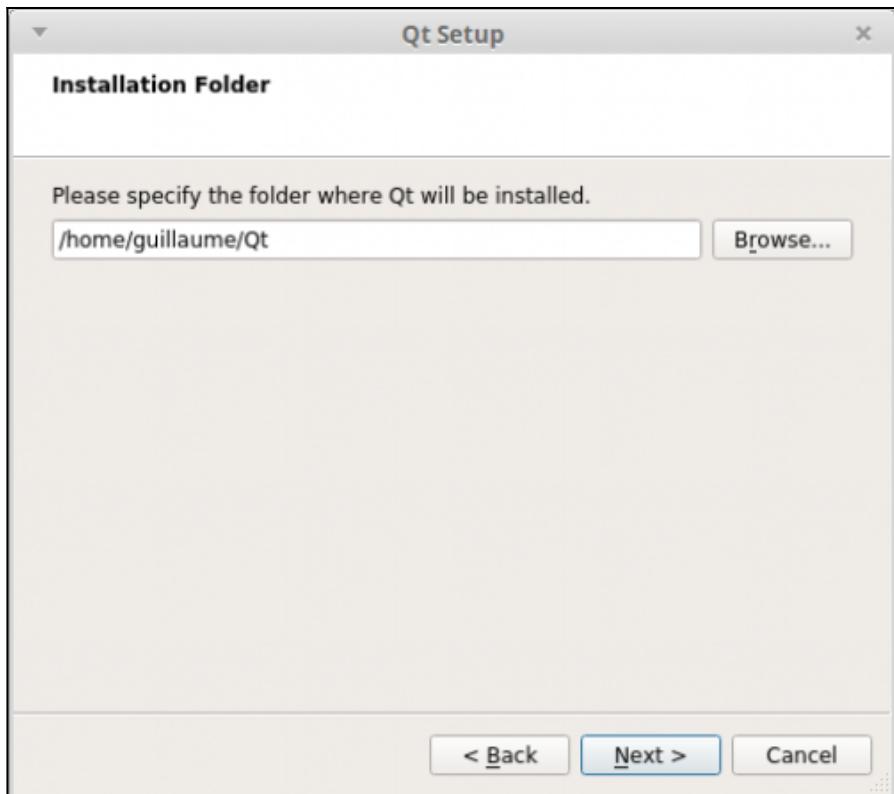
Cliquez sur **Suivant** pour vous connecter.



Cette page sert simplement à vous dire bienvenue... [Suivant](#).



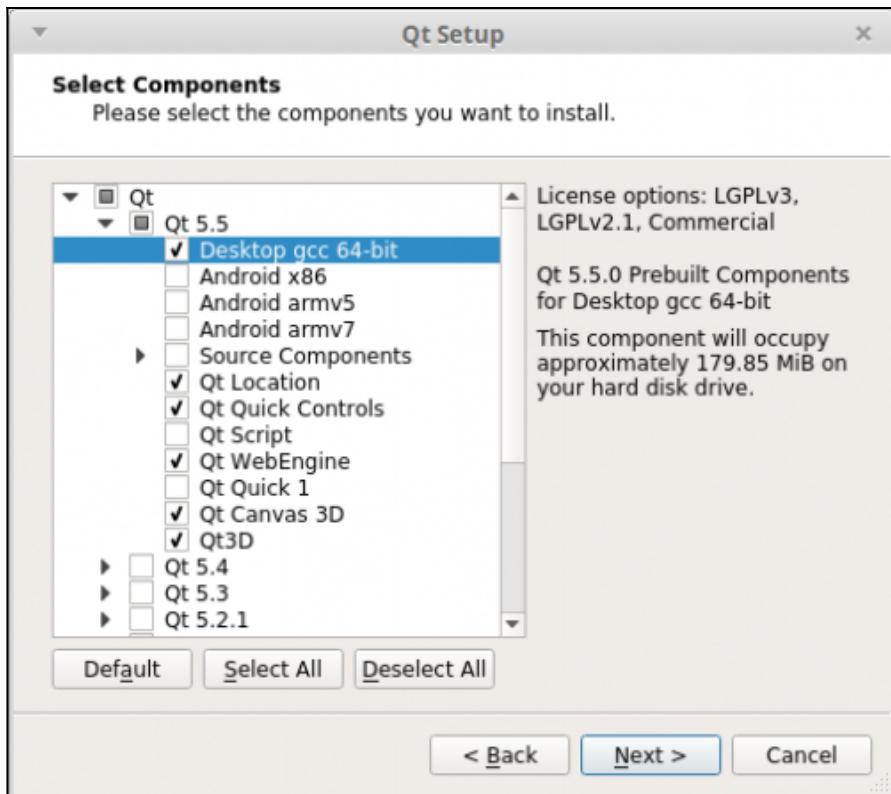
L'installateur recherche en ligne la liste des outils et versions que vous pouvez installer (cela peut durer de quelques dizaines de secondes à quelques minutes, en fonction de votre connexion internet).



Une fois que le téléchargement est fini, la page suivante s'affiche automatiquement. Cette page permet de choisir le dossier d'installation de Qt. Par défaut, le chemin est `/home/username/Qt`.

Vous pouvez changer le répertoire d'installation si vous le souhaitez. Dans la suite de ce tutoriel, nous allons utiliser le chemin par défaut. Si vous changez de répertoire, pensez à adapter les chemins donnés dans la suite de ce tutoriel.

Cliquez sur Suivant.



La page suivante permet de sélectionner la liste des outils à installer. Pour programmer en C++ avec Qt, il faut installer au moins trois outils :

- un **éditeur**. Qt (le framework) est fourni avec un éditeur appelé Qt Creator (ne confondez pas l'éditeur et le framework). Cet éditeur est installé automatiquement et il n'est pas possible de le désactiver ;
- au moins une version de **Qt** ;
- un **compilateur** compatible avec la version de Qt installée.

Il existe sous Linux deux compilateurs principaux : [GCC](#) et le compilateur de [Clang](#). La version de Qt pour GCC est compatible avec Clang.

Si vous installez les compilateurs en utilisant le gestionnaire de paquets de votre version de Linux, vous n'aurez pas forcément les dernières

versions. Pour bénéficier du C++11, il faut au moins GCC 4.9 (la version la plus récente est 5.2) et Clang 3.4 (la version la plus récente est 3.6.2).

L'installateur propose de nombreuses options, nous allons les détailler un peu.

Pour commencer, on peut voir que l'installateur permet d'installer plusieurs versions de Qt : 5.5, 5.4, 5.3, etc. Par défaut, nous allons utiliser la dernière version (5.5). Si vous devez travailler sur une version plus ancienne que Qt 5.5, vous pouvez l'installer à partir de cette page. (Remarque : il est encore possible d'installer Qt 4, mais la procédure est différente, nous n'allons pas voir cela.)

Pour chaque version de Qt proposée, il existe plusieurs versions et modules, selon le compilateur.

**Remarque importante : vous choisissez ici les versions de Qt à installer, pas les compilateurs. Chaque version de Qt est identifiée par un nom de compilateur, mais cela ne veut pas dire que le compilateur correspondant sera installé. Les compilateurs sont installés séparément, voir plus bas.**

- Desktop GCC 64 bit : version de Qt compatible avec les compilateurs GCC et Clang ;
- Android x86, Android armv5 et Android armv7 : versions pour Android.

Comme vous pouvez le voir, Qt supporte de nombreux systèmes d'exploitation, c'est un peu compliqué de s'y retrouver. Dans ce chapitre, nous allons voir uniquement l'installation de Qt pour les ordinateurs de bureau (Desktop). L'installation de Qt pour mobiles sera vu de dans chapitres spécifiques.

En dessous des différentes versions de Qt, vous pouvez choisir d'installer différents modules optionnels :

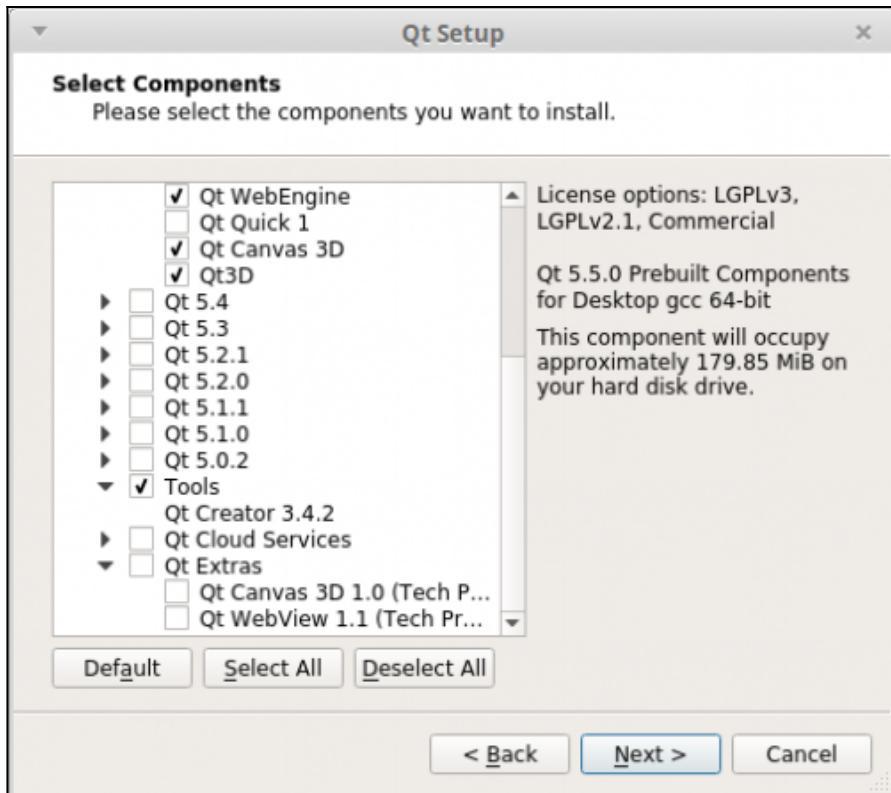
- Source Components : permet d'installer le code source de Qt. Si vous débutez, cela ne vous servira pas, mais si vous êtes un développeur C++/Qt un peu plus expérimenté, cela permet

d'analyser comment Qt est conçu ;

- `Qt Location` : fonctionnalités de positionnement (GPS) ;
- `Qt Quick Controls` : éléments d'interface graphique (bouton, dialogues, etc.) pour Qt Quick ;
- `Qt Script` : déprécié, ne pas utiliser (sauf besoin spécifique) ;
- `Qt WebEngine` : moteur Web, pour afficher des pages internet ;
- `Qt Quick 1` : déprécié, ne pas utiliser (sauf besoin spécifique) ;
- `Qt Canvas 3D` : moteur 3D simple similaire à WebGL en JavaScript ;
- `Qt 3D` : moteur 3D simple.

Je vous conseille d'installer les modules correspondant à la capture d'écran précédente (Qt pour GCC, Qt Location, Qt 3D, Qt Canvas 3D, Qt Quick Controls et Qt WebEngine. Vous n'aurez pas besoin de tous ces modules pour commencer, mais cela vous permettra d'explorer ce que propose Qt et de vous amuser un peu).

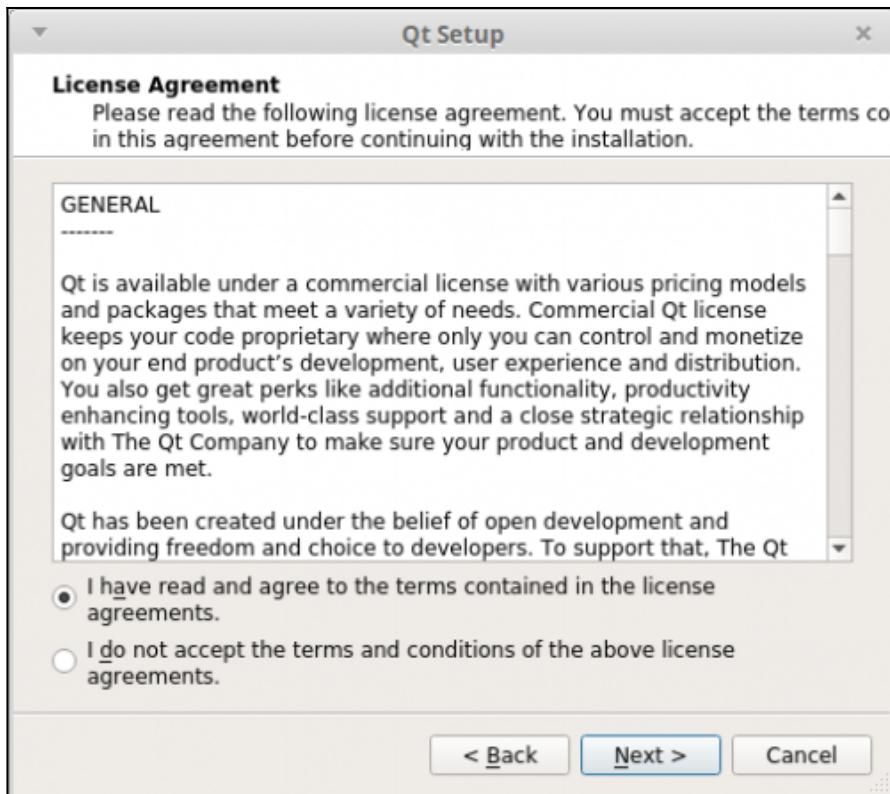
En dessous du choix des versions de Qt et des modules additionnels, vous pouvez sélectionner l'installation des outils externes.



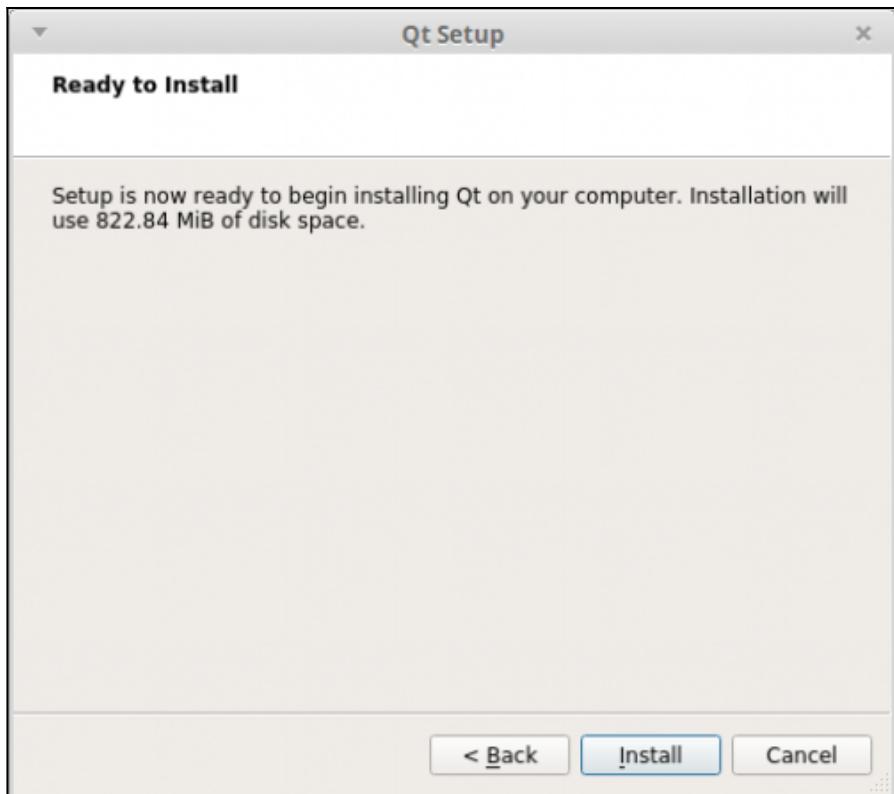
Qt Creator est l'éditeur fourni avec Qt. Il n'est pas possible de ne pas l'installer. (Même si vous souhaitez utiliser un autre éditeur, on utilisera Qt Creator dans ce tutoriel, pour tester si l'installation s'est bien passée).

Il n'est pas nécessaire d'installer les modules `Qt Cloud Services` et `Qt Extras`.

Cliquez sur suivant lorsque vous avez sélectionné les modules qui vous intéressent.

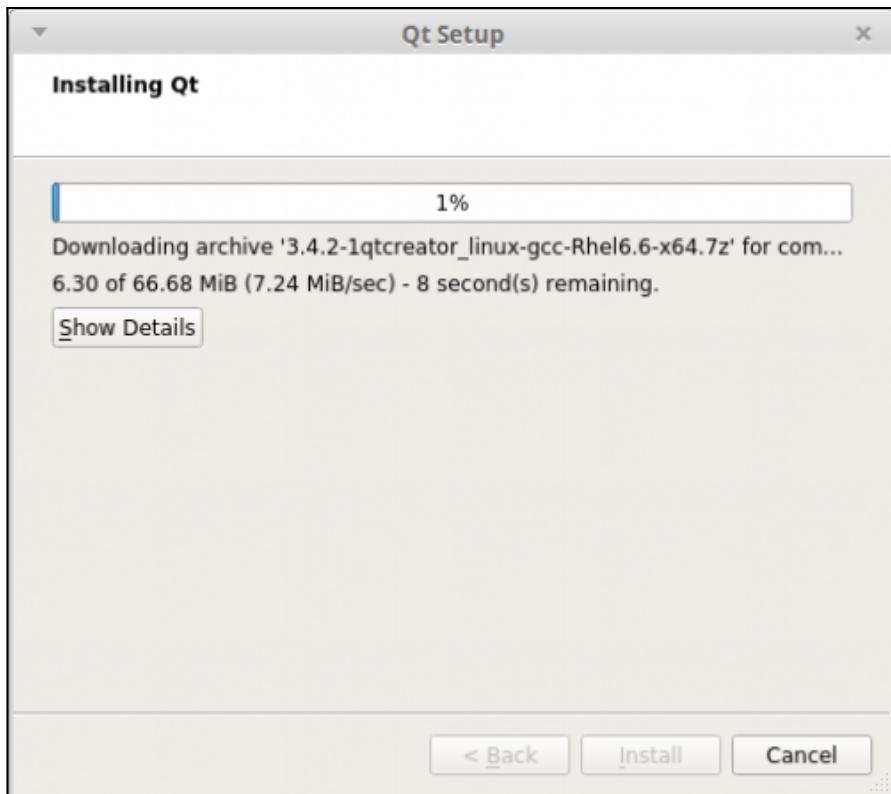


Cette page permet de valider les licences utilisateurs. Acceptez et cliquez sur suivant.



Une fois que tout cela est fait, l'installation est prête à démarrer. Cliquez sur Installation.

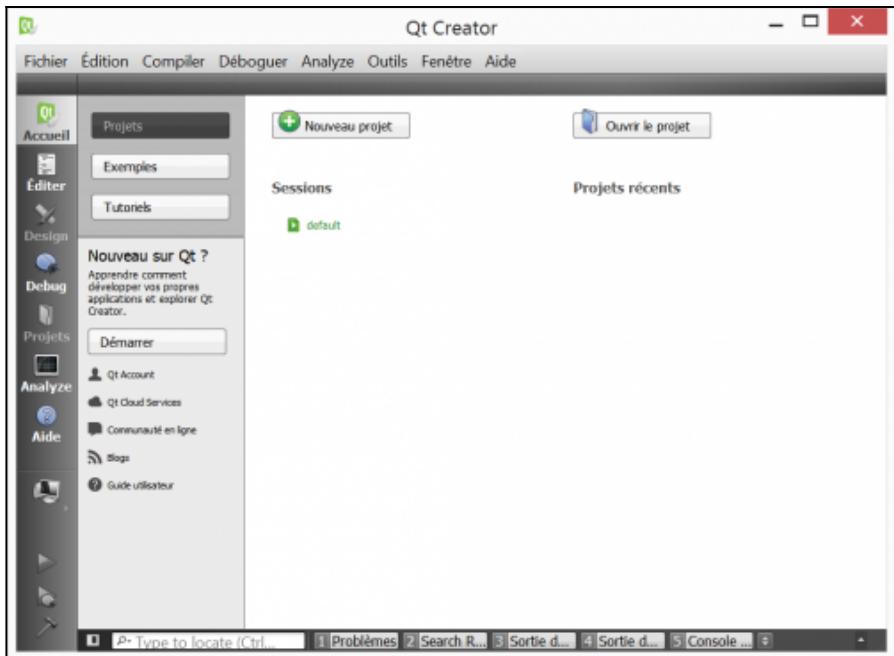
Le téléchargement puis l'installation se lancent. Selon votre connexion et le nombre de paquets que vous installez, cela peut prendre plusieurs minutes à plusieurs heures (si vous souhaitez installer beaucoup de paquets, il est probablement préférable de répéter l'installation plusieurs fois). L'installation sature le processeur, ne vous étonnez pas trop si Linux devient un peu lent pendant ce temps-là. Allez vous balader dehors, il fait beau (la pluie, c'est beau...).



Une fois que l'installation est terminée, la page suivante propose de lancer Qt Creator.

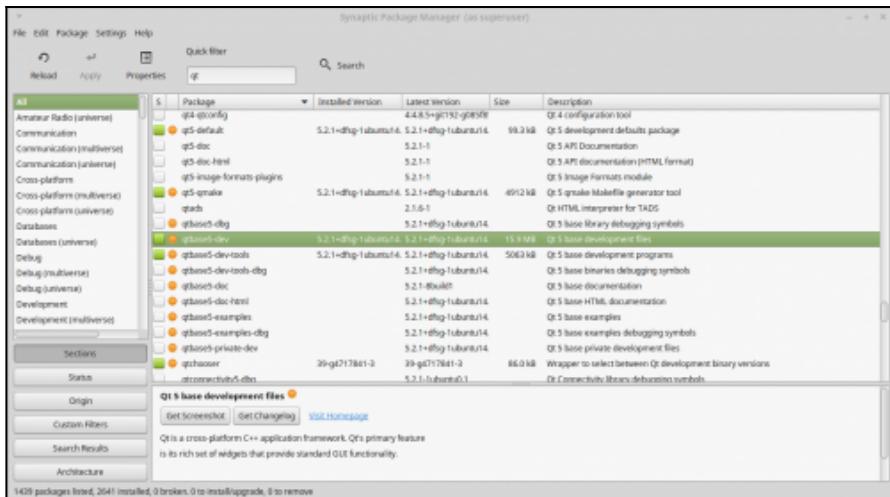


Cliquez sur Terminer. Qt Creator s'ouvre et affiche la page d'accueil.



## Installer en utilisant les dépôts

Beaucoup de distributions Linux proposent Qt 5 directement dans les dépôts, vous pouvez donc commencer par là. Par exemple, avec Ubuntu, lancez `Synaptic` et faites une recherche sur `qtbase5-dev` :



Comme vous pouvez le voir dans la copie d'écran, Ubuntu propose Qt 5, mais pas forcément Qt 5.5 (la version de Qt disponible dépendra de la distribution). Si vous n'avez pas besoin de la dernière version de Qt, vous pouvez installer celle des dépôts.

Vous pouvez également installer en ligne de commande, par exemple :

```
sudo apt-get install qtbase5-dev
```

Chaque module de Qt est dans un paquet spécifique. Regarder la liste des paquets dans [Synaptic](#) (`qtbase5-dev` contient les fonctionnalités de base, vous pouvez commencer juste avec lui).

En complément, vous aurez besoin d'installer un compilateur C++. Vous pouvez installer GCC ou/et CLang, via synaptic ou en ligne de commande. Il est également possible d'installer d'autres méta-paquets, qui contiennent plusieurs outils de développement, comme les paquets “build-essential” ou “ubuntu-sdk”.

Pour tester si vous avez un compilateur fonctionnel et quelle est sa version, vous pouvez taper l'une des lignes suivantes :

```
g++ -v
clang++-3.5 -v
```

Par exemple, chez moi, cela retourne pour GCC les lignes suivantes. Vous pouvez voir à la dernière ligne que c'est la version 4.9.1 de GCC.

```
Using built-in specs.
COLLECT_GCC=g++
COLLECT_LTO_WRAPPER=/usr/lib/gcc/x86_64-linux-gnu/4.9/lto-wr
apper
Target: x86_64-linux-gnu
Configured with: ../src/configure -v
--with-pkgversion='Ubuntu 4.9.1-16ubuntu6'
--with-bugurl=file:///usr/share/doc/gcc-4.9/README.Bugs
--enable-languages=c,c++,java,go,d,fortran,objc,obj-c++
--prefix=/usr --program-suffix=-4.9 --enable-shared
--enable-linker-build-id --libexecdir=/usr/lib
--without-included-gettext --enable-threads=posix
--with-gxx-include-dir=/usr/include/c++/4.9
--libdir=/usr/lib --enable-nls --with-sysroot=/
--enable-clocale-gnu --enable-libstdcxx-debug
--enable-libstdcxx-time=yes --enable-gnu-unique-object
--disable-vtable-verify --enable-plugin
--with-system-zlib --disable-browser-plugin
--enable-java-awt=gtk --enable-gtk-cairo
--with-java-home=/usr/lib/jvm/java-1.5.0-gcj-4.9-amd64/jre
--enable-java-home
--with-jvm-root-dir=/usr/lib/jvm/java-1.5.0-gcj-4.9-amd64
--with-jvm-jar-dir=/usr/lib/jvm-exports/java-1.5.0-gcj-4.9-a
md64 --with-arch-directory=amd64
--with-ecj-jar=/usr/share/java/eclipse-ecj.jar
--enable-objc-gc --enable-multiarch
--disable-werror --with-arch-32=i686 --with-abi=m64
--with-multilib-list=m32,m64,mx32
--enable-multilib --with-tune=generic
--enable-checking=release --build=x86_64-linux-gnu
--host=x86_64-linux-gnu --target=x86_64-linux-gnu
Thread model: posix
gcc version 4.9.1 (Ubuntu 4.9.1-16ubuntu6)
```

Pour Clang :

```
Ubuntu clang version 3.5.0-4ubuntu2 (tags/RELEASE_350/final)
(based on LLVM 3.5.0)
```

```
Target: x86_64-pc-linux-gnu
Thread model: posix
Found candidate GCC installation:
/usr/bin/../lib/gcc/i686-linux-gnu/4.9
Found candidate GCC installation:
/usr/bin/../lib/gcc/i686-linux-gnu/4.9.1
Found candidate GCC installation:
/usr/bin/../lib/gcc/x86_64-linux-gnu/4.8
Found candidate GCC installation:
/usr/bin/../lib/gcc/x86_64-linux-gnu/4.8.3
Found candidate GCC installation:
/usr/bin/../lib/gcc/x86_64-linux-gnu/4.9
Found candidate GCC installation:
/usr/bin/../lib/gcc/x86_64-linux-gnu/4.9.1
Found candidate GCC installation:
/usr/lib/gcc/i686-linux-gnu/4.9
Found candidate GCC installation:
/usr/lib/gcc/i686-linux-gnu/4.9.1
Found candidate GCC installation:
/usr/lib/gcc/x86_64-linux-gnu/4.8
Found candidate GCC installation:
/usr/lib/gcc/x86_64-linux-gnu/4.8.3
Found candidate GCC installation:
/usr/lib/gcc/x86_64-linux-gnu/4.9
Found candidate GCC installation:
/usr/lib/gcc/x86_64-linux-gnu/4.9.1
Selected GCC installation:
/usr/bin/../lib/gcc/x86_64-linux-gnu/4.9
Candidate multilib: .;@m64
Candidate multilib: 32;@m32
Candidate multilib: x32;@mx32
Selected multilib: .;@m64
```

[Revenir à la page principale du tutoriel](#)

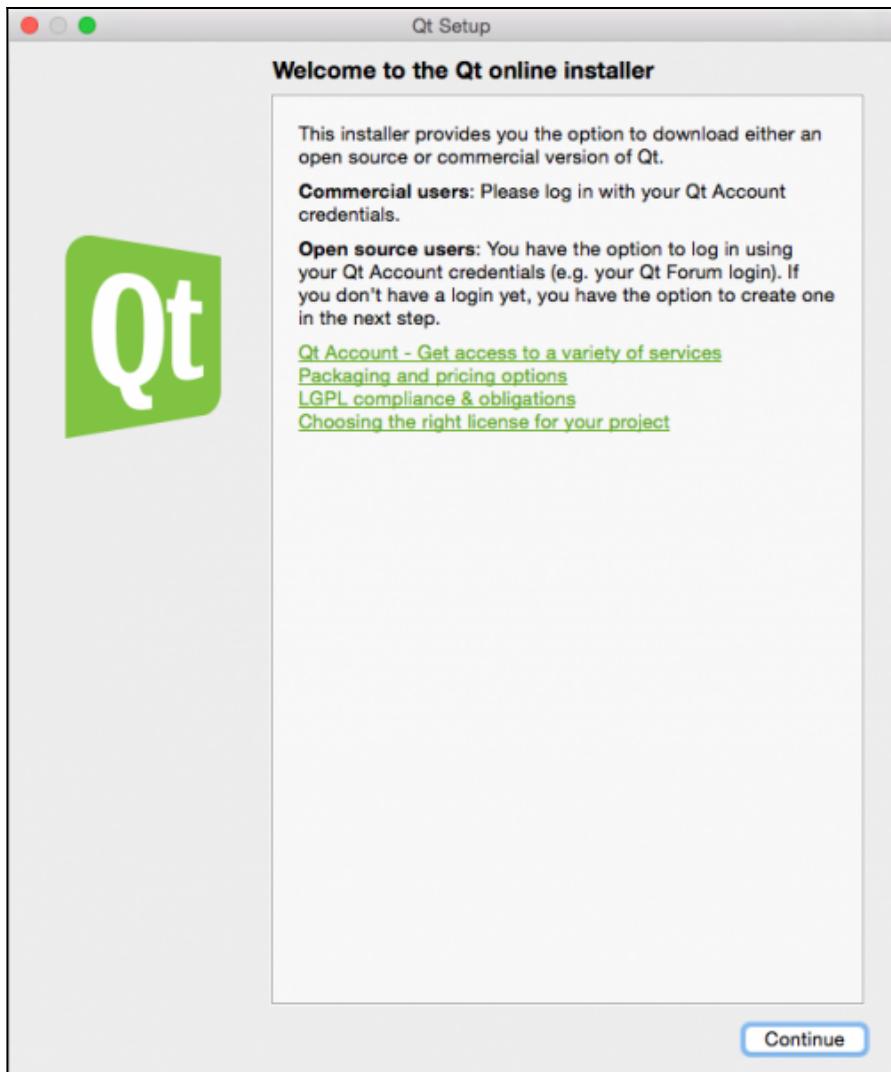
[Revenir à la page principale du tutoriel](#)

# Installer Qt 5.5 sous Mac OS X

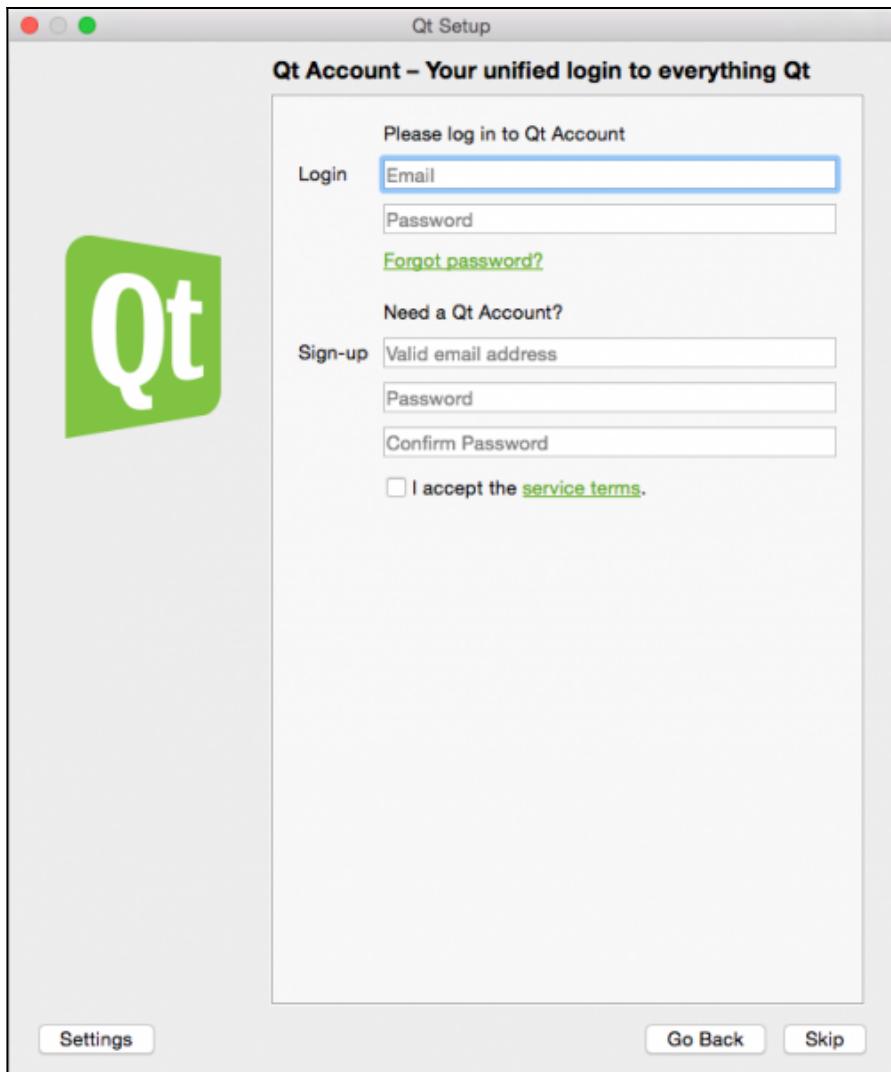
## Utiliser l'installateur

[http://download.qt.io/official\\_releases/online\\_installers/qt-unified-mac-x64-online.dmg](http://download.qt.io/official_releases/online_installers/qt-unified-mac-x64-online.dmg)

L'installateur sous Mac OS X s'appelle `qt-unified-mac-x64-online.dmg`, vous pouvez le lancer directement.



La première page permet de rappeler les différentes licences utilisables avec Qt (commerciale et open-source). Nous allons continuer avec la version open-source. Cliquez sur **Suivant**.

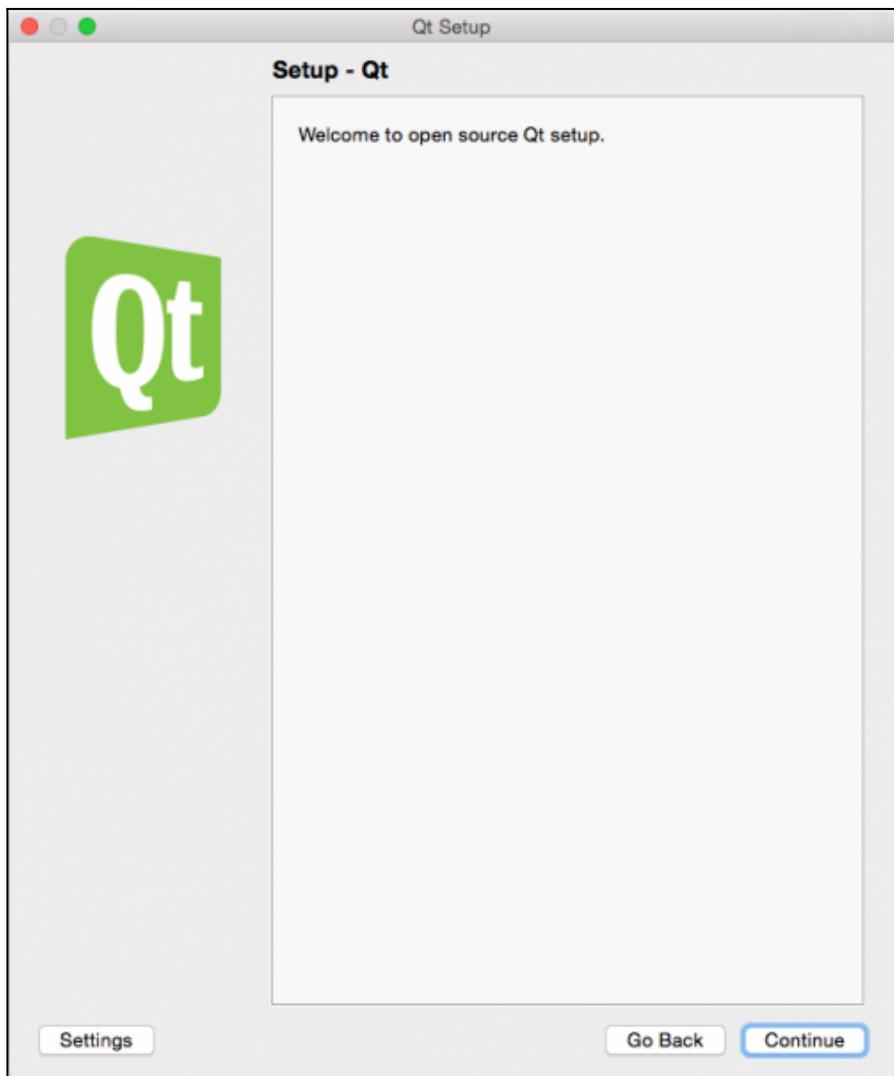


Le bouton **Paramètres** permet de configurer un serveur proxy si vous en utilisez un et d'ajouter des dépôts pour Qt. Normalement, vous n'avez pas besoin de modifier les dépôts, le dépôt principal de Qt est configuré par défaut. (Sauf si vous installez une version en cours de développement de Qt, avant sa sortie officielle).

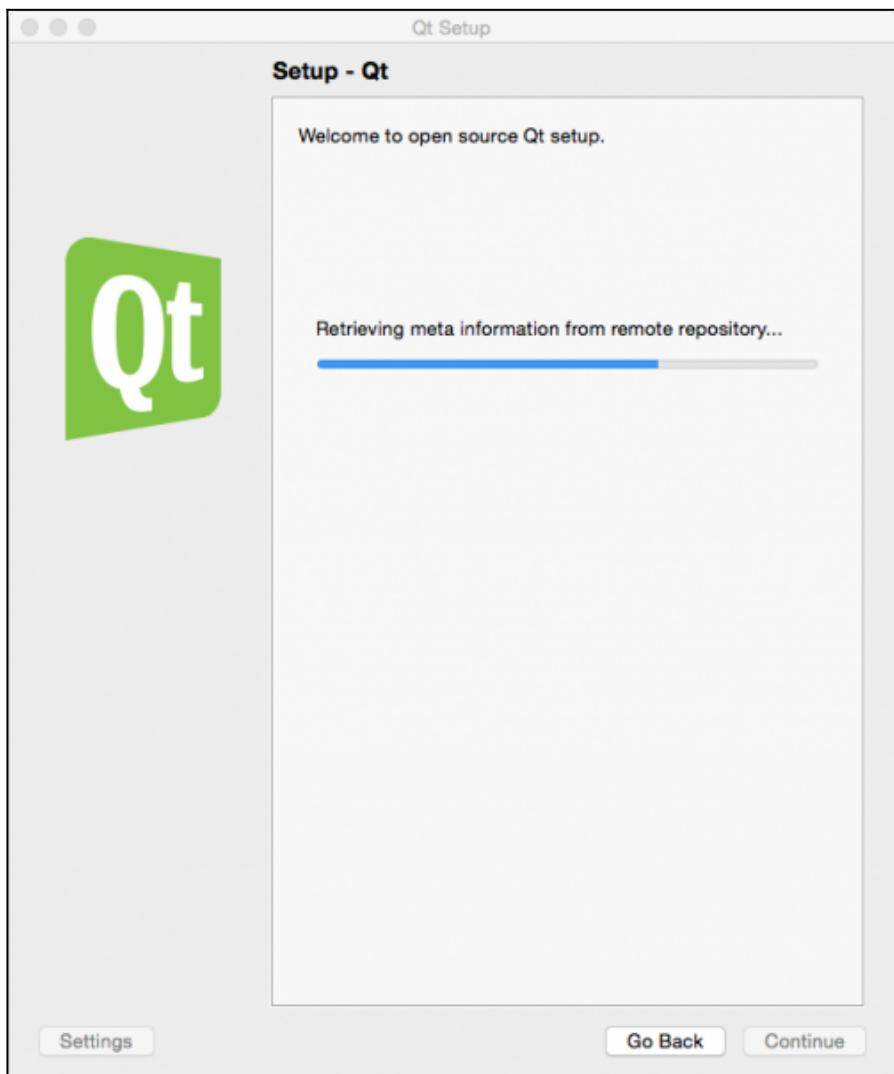
Pour installer Qt, il est maintenant nécessaire de créer un compte. Si

vous en avez déjà créé un sur le site <http://qt.io>, vous pouvez l'utiliser directement. Si ce n'est pas le cas, vous devez créer un compte en remplissant le formulaire.

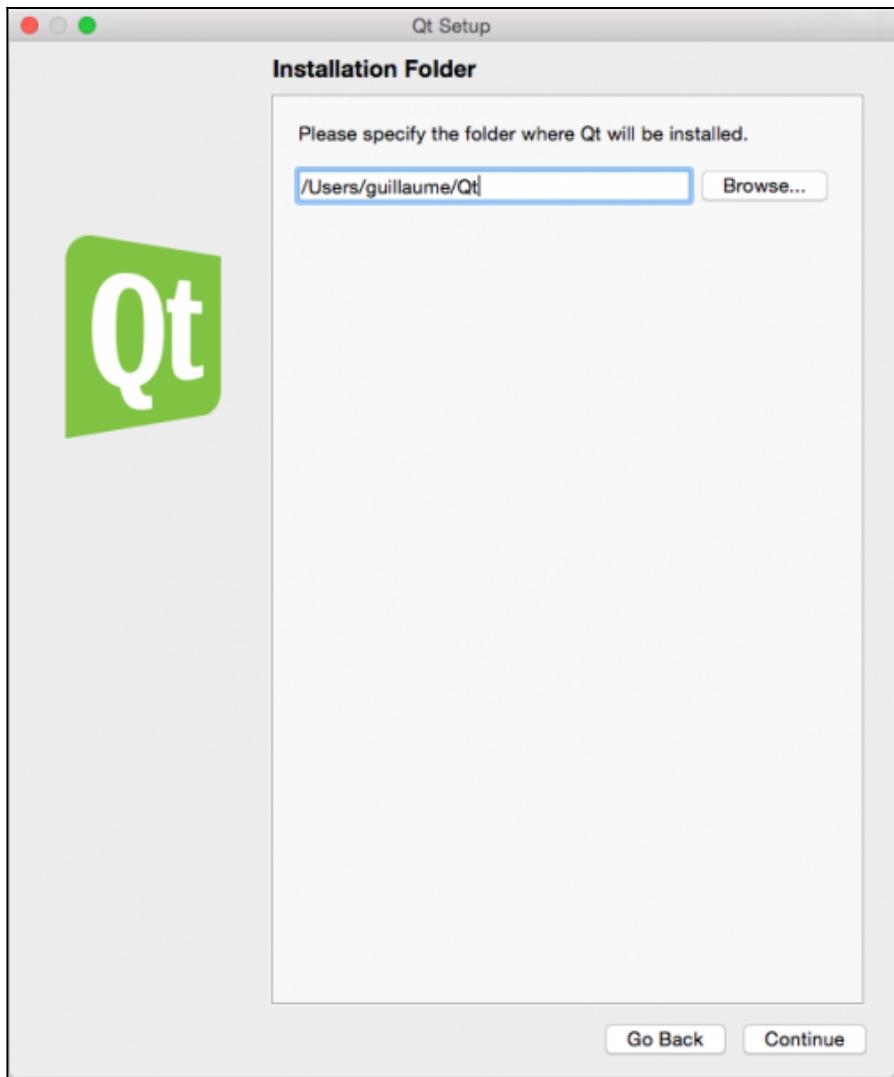
Cliquez sur **Suivant** pour vous connecter.



Cette page sert simplement à vous dire bienvenue... [Suivant.](#)



L'installateur recherche en ligne la liste des outils et versions que vous pouvez installer (cela peut durer de quelques dizaines de secondes à quelques minutes, en fonction de votre connexion internet).

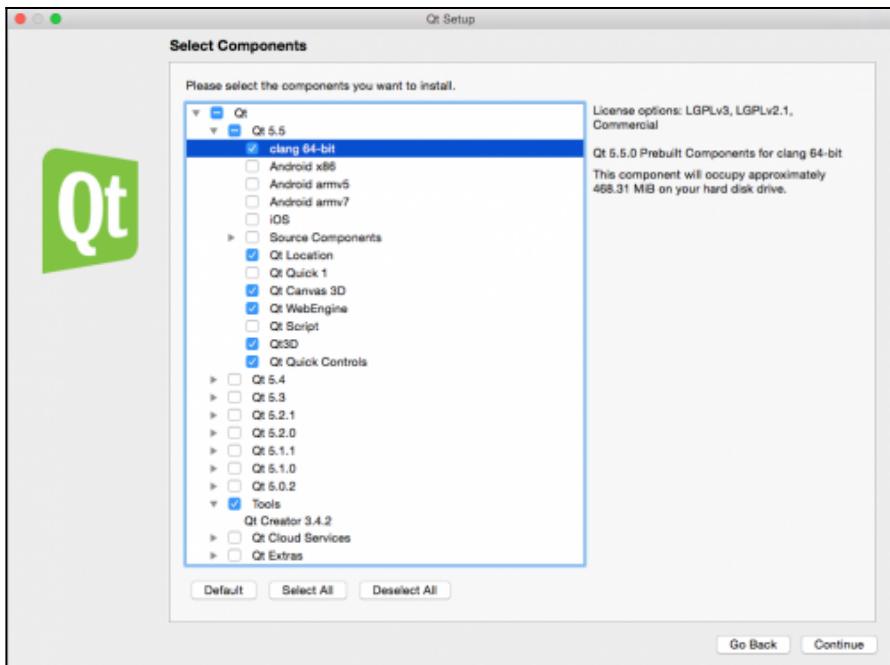


Une fois que le téléchargement est fini, la page suivante s'affiche automatiquement. Cette page permet de choisir le dossier d'installation de Qt. Par défaut, le chemin est `/home/username/Qt`.

Vous pouvez changer le répertoire d'installation si vous le souhaitez. Dans la suite de ce tutoriel, nous allons utiliser le chemin par défaut. Si vous changez de répertoire, pensez à adapter les chemins donnés dans

la suite de ce tutoriel.

Cliquez sur **Suivant**.



La page suivante permet de sélectionner la liste des outils à installer.  
Pour programmer en C++ avec Qt, il faut installer au moins trois outils :

- un **éditeur**. Qt (le framework) est fourni avec un éditeur appelé Qt Creator (ne confondez pas l'éditeur et le framework). Cet éditeur est installé automatiquement et il n'est pas possible de le désactiver ;
- au moins une version de **Qt** ;
- un **compilateur** compatible avec la version de Qt installée.

Apple fournit les compilateurs C++ pour Mac OS X et iOS.

Il existe sous Linux deux compilateurs principaux : **GCC** et le compilateur de **Clang**. La version de Qt pour GCC est compatible avec Clang.

Si vous installez les compilateurs en utilisant le gestionnaire de paquets de votre version de Linux, vous n'aurez pas forcément les dernières versions. Pour bénéficier du C++11, il faut au moins GCC 4.9 (la version la plus récente est 5.2) et Clang 3.4 (la version la plus récente est 3.6.2).

L'installateur propose de nombreuses options, nous allons les détailler un peu.

Pour commencer, on peut voir que l'installateur permet d'installer plusieurs versions de Qt : 5.5, 5.4, 5.3, etc. Par défaut, nous allons utiliser la dernière version (5.5). Si vous devez travailler sur une version plus ancienne que Qt 5.5, vous pouvez l'installer à partir de cette page. (Remarque : il est encore possible d'installer Qt 4, mais la procédure est différente, nous n'allons pas voir cela.)

Pour chaque version de Qt proposée, il existe plusieurs versions et modules, selon le compilateur.

**Remarque importante : vous choisissez ici les versions de Qt à installer, pas les compilateurs. Chaque version de Qt est identifiée par un nom de compilateur, mais cela ne veut pas dire que le compilateur correspondant sera installé. Les compilateurs sont installés séparément, voir plus bas.**

- Desktop GCC 64 bit : version de Qt compatible avec les compilateurs GCC et Clang ;
- Android x86, Android armv5 et Android armv7 : versions pour Android.

Comme vous pouvez le voir, Qt supporte de nombreux systèmes d'exploitation, c'est un peu compliqué de s'y retrouver. Dans ce chapitre, nous allons voir uniquement l'installation de Qt pour les ordinateurs de bureau (Desktop). L'installation de Qt pour mobiles sera vu de dans chapitres spécifiques.

En dessous des différentes versions de Qt, vous pouvez choisir d'installer différents modules optionnels :

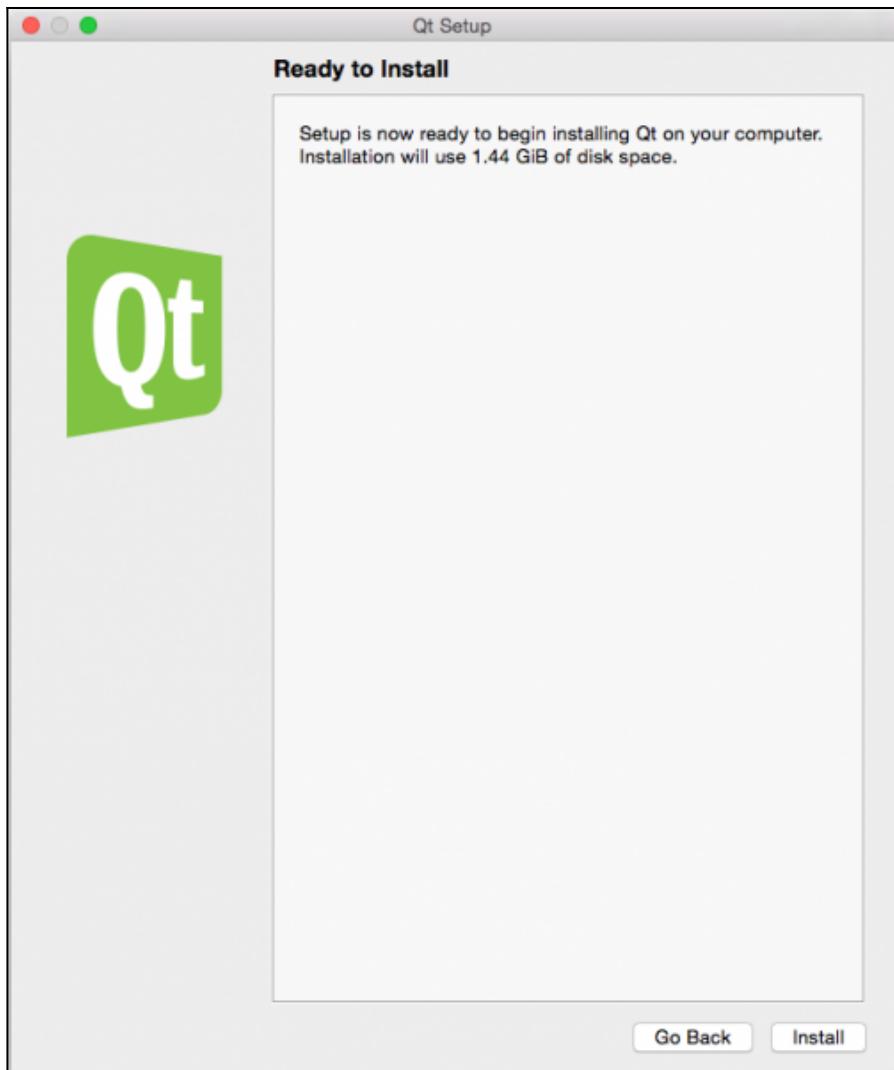
- Source Components : permet d'installer le code source de Qt. Si

vous débutez, cela ne vous servira pas, mais si vous êtes un développeur C++/Qt un peu plus expérimenté, cela permet d'analyser comment Qt est conçu ;

- `Qt Location` : fonctionnalités de positionnement (GPS) ;
- `Qt Quick Controls` : éléments d'interface graphique (bouton, dialogues, etc.) pour Qt Quick ;
- `Qt Script` : déprécié, ne pas utiliser (sauf besoin spécifique) ;
- `Qt WebEngine` : moteur Web, pour afficher des pages internet ;
- `Qt Quick 1` : déprécié, ne pas utiliser (sauf besoin spécifique) ;
- `Qt Canvas 3D` : moteur 3D simple similaire à WebGL en JavaScript ;
- `Qt 3D` : moteur 3D simple.

Je vous conseille d'installer les modules correspondant à la capture d'écran précédente (Qt pour GCC, Qt Location, Qt 3D, Qt Canvas 3D, Qt Quick Controls et Qt WebEngine. Vous n'aurez pas besoin de tous ces modules pour commencer, mais cela vous permettra d'explorer ce que propose Qt et de vous amuser un peu).

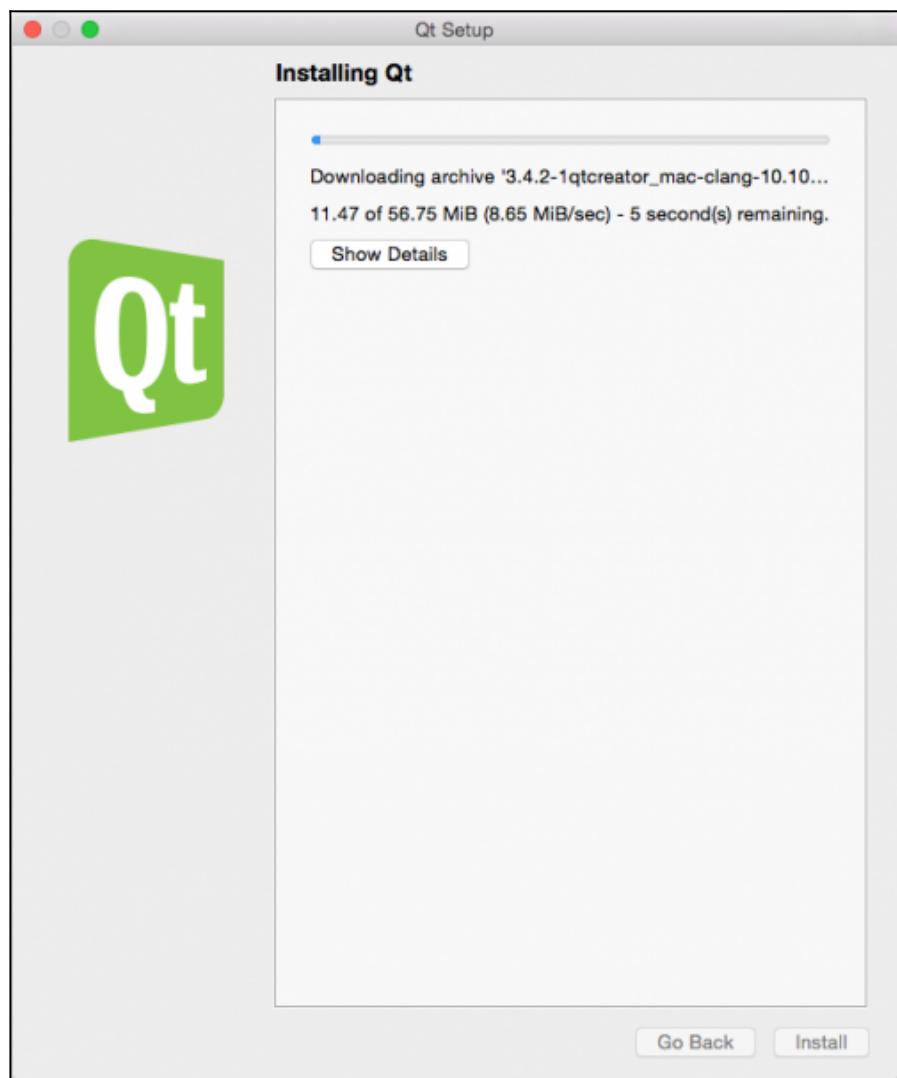
En dessous du choix des versions de Qt et des modules additionnels, vous pouvez sélectionner l'installation des outils externes.



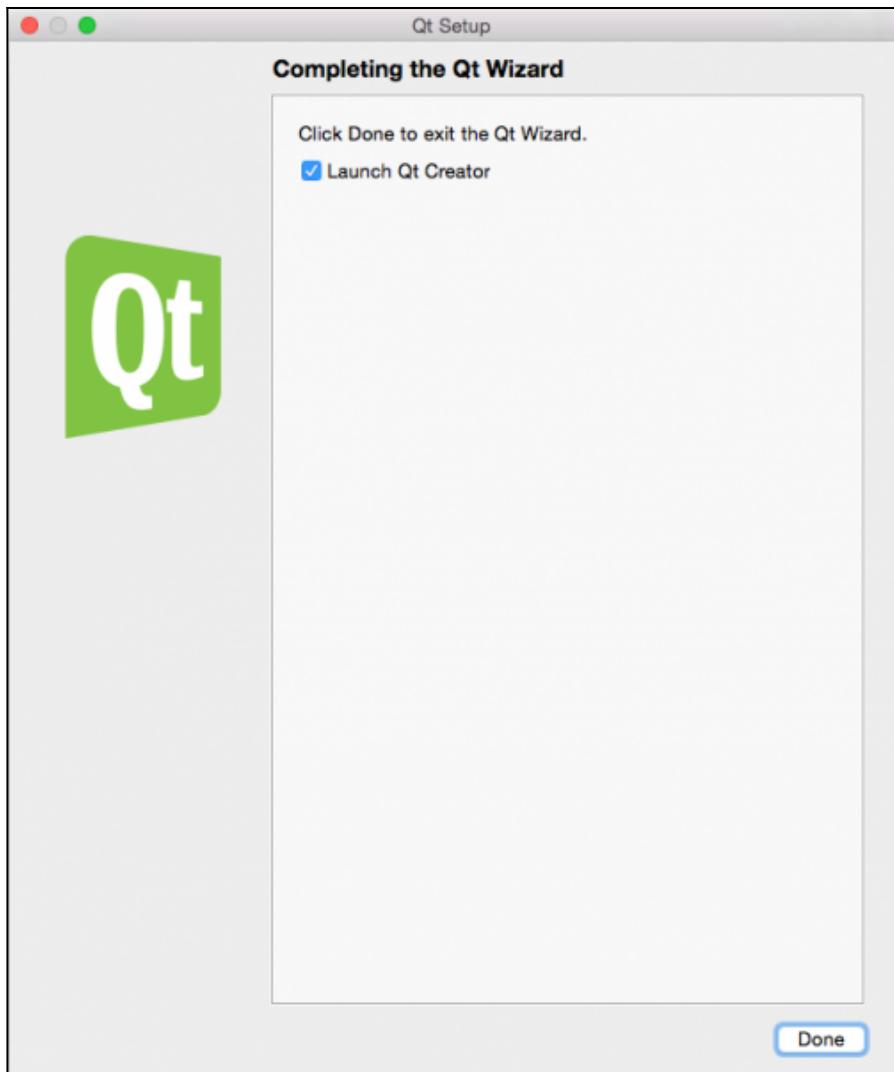
Qt Creator est l'éditeur fourni avec Qt. Il n'est pas possible de ne pas l'installer. (Même si vous souhaitez utiliser un autre éditeur, on utilisera Qt Creator dans ce tutoriel, pour tester si l'installation s'est bien passée).

Il n'est pas nécessaire d'installer les modules `Qt Cloud Services` et `Qt Extras`.

Cliquez sur suivant lorsque vous avez sélectionné les modules qui vous intéressent.



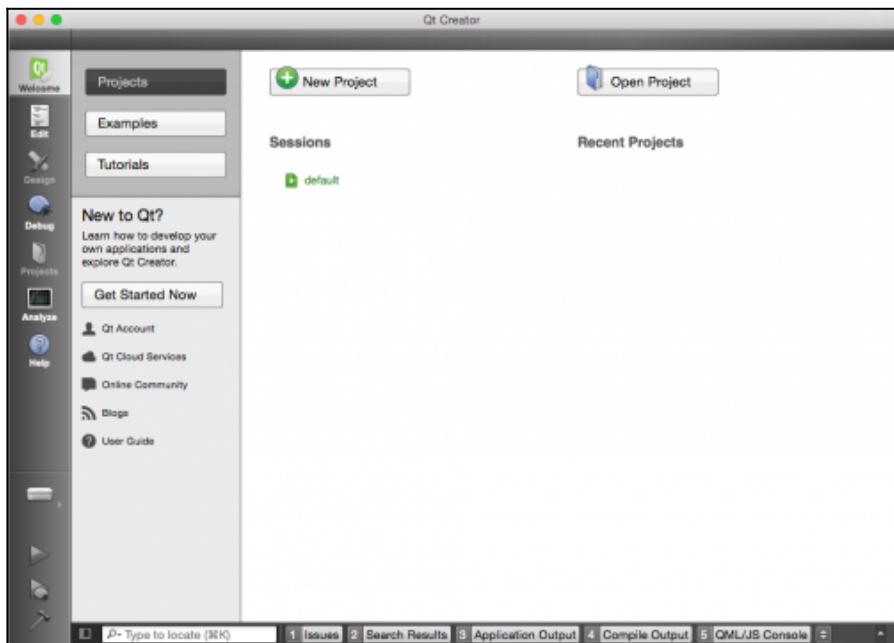
Cette page permet de valider les licences utilisateurs. Acceptez et cliquez sur suivant.



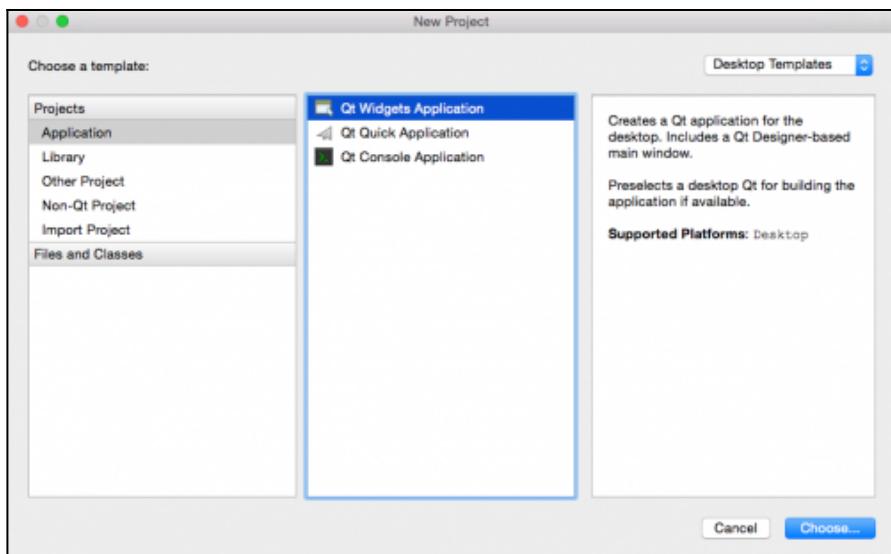
Une fois que tout cela est fait, l'installation est prête à démarrer. Cliquez sur [Installation](#).

Le téléchargement puis l'installation se lancent. Selon votre connexion et le nombre de paquets que vous installez, cela peut prendre plusieurs minutes à plusieurs heures (si vous souhaitez installer beaucoup de paquets, il est probablement préférable de répéter l'installation plusieurs

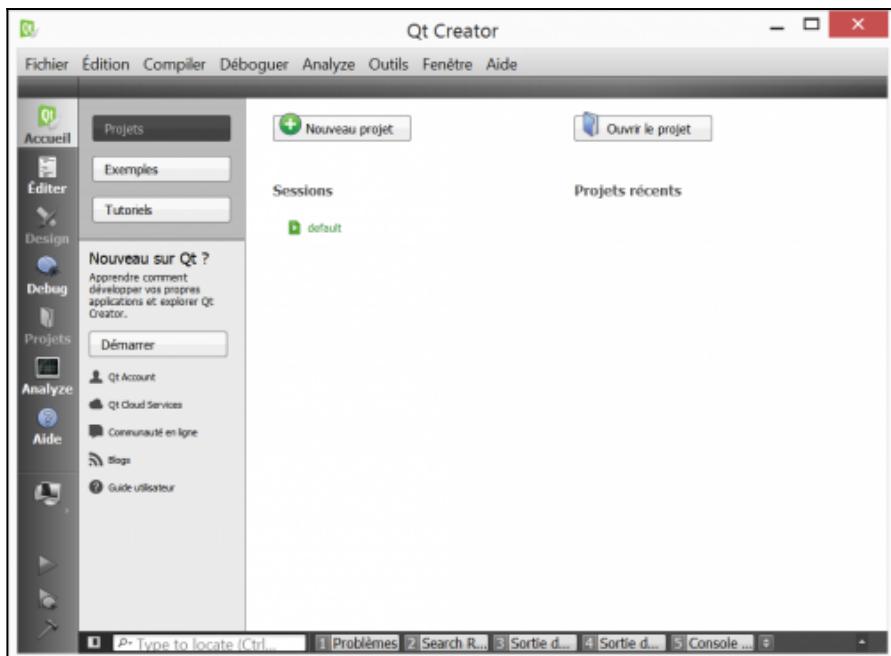
fois). L'installation sature le processeur, ne vous étonnez pas trop si Linux devient un peu lent pendant ce temps-là. Allez vous balader dehors, il fait beau (la pluie, c'est beau...).



Une fois que l'installation est terminée, la page suivante propose de lancer Qt Creator.

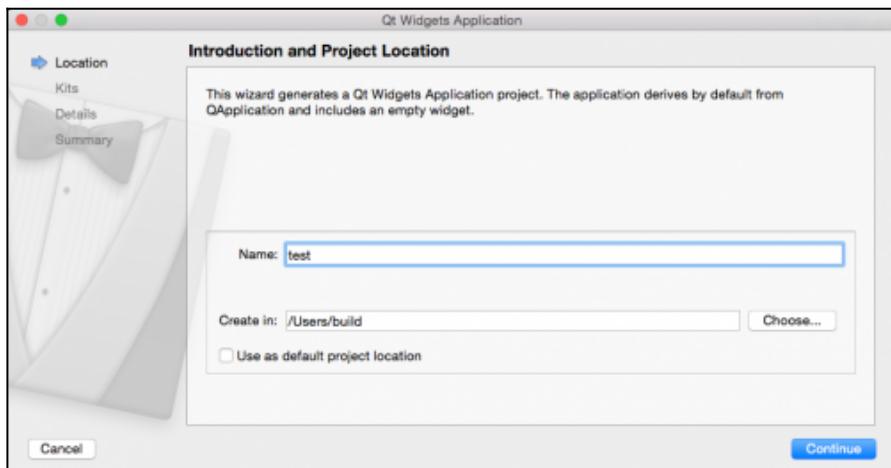


Cliquez sur **Terminer**. Qt Creator s'ouvre et affiche la page d'accueil.



## Installer en utilisant les dépôts

Beaucoup de distributions Linux proposent Qt 5 directement dans les dépôts, vous pouvez donc commencer par là. Par exemple, avec Ubuntu, lancez **Synaptic** et faites une recherche sur **qtbase5-dev** :



Comme vous pouvez le voir dans la copie d'écran, Ubuntu propose Qt 5, mais pas forcément Qt 5.5 (la version de Qt disponible dépendra de la distribution). Si vous n'avez pas besoin de la dernière version de Qt, vous pouvez installer celle des dépôts.

Vous pouvez également installer en ligne de commande, par exemple :

```
sudo apt-get install qtbase5-dev
```

Chaque module de Qt est dans un paquet spécifique. Regarder la liste des paquets dans **Synaptic** (**qtbase5-dev**) contient les fonctionnalités de base, vous pouvez commencer juste avec lui).

En complément, vous aurez besoin d'installer un compilateur C++. Vous pouvez installer GCC ou/et CLang, via synaptic ou en ligne de commande. Il est également possible d'installer d'autres méta-paquets, qui contiennent plusieurs outils de développement, comme les paquets

“build-essential” ou “ubuntu-sdk”.

Pour tester si vous avez un compilateur fonctionnel et quelle est sa version, vous pouvez taper l'une des lignes suivantes :

```
g++ -v  
clang++-3.5 -v
```

Par exemple, chez moi, cela retourne pour GCC les lignes suivantes. Vous pouvez voir à la dernière ligne que c'est la version 4.9.1 de GCC.

```
Using built-in specs.  
COLLECT_GCC=g++  
COLLECT_LTO_WRAPPER=/usr/lib/gcc/x86_64-linux-gnu/4.9/lto-wr  
apper  
Target: x86_64-linux-gnu  
Configured with: ../src/configure -v  
--with-pkgversion='Ubuntu 4.9.1-16ubuntu6'  
--with-bugurl=file:///usr/share/doc/gcc-4.9/README.Bugs  
--enable-languages=c,c++,java,go,d,fortran,objc,obj-c++  
--prefix=/usr --program-suffix=-4.9 --enable-shared  
--enable-linker-build-id --libexecdir=/usr/lib  
--without-included-gettext --enable-threads=posix  
--with-gxx-include-dir=/usr/include/c++/4.9  
--libdir=/usr/lib --enable-nls --with-sysroot=/  
--enable-clocale-gnu --enable-libstdcxx-debug  
--enable-libstdcxx-time=yes --enable-gnu-unique-object  
--disable-vtable-verify --enable-plugin  
--with-system-zlib --disable-browser-plugin  
--enable-java-awt=gtk --enable-gtk-cairo  
--with-java-home=/usr/lib/jvm/java-1.5.0-gcj-4.9-amd64/jre  
--enable-java-home  
--with-jvm-root-dir=/usr/lib/jvm/java-1.5.0-gcj-4.9-amd64  
--with-jvm-jar-dir=/usr/lib/jvm-exports/java-1.5.0-gcj-4.9-a  
md64 --with-arch-directory=amd64  
--with-ecj-jar=/usr/share/java/eclipse-ecj.jar  
--enable-objc-gc --enable-multiarch  
--disable-werror --with-arch-32=i686 --with-abi=m64  
--with-multilib-list=m32,m64,mx32  
--enable-multilib --with-tune=generic  
--enable-checking=release --build=x86_64-linux-gnu
```

```
--host=x86_64-linux-gnu --target=x86_64-linux-gnu
Thread model: posix
gcc version 4.9.1 (Ubuntu 4.9.1-16ubuntu6)
```

Pour Clang :

```
Ubuntu clang version 3.5.0-4ubuntu2 (tags/RELEASE_350/final)
(based on LLVM 3.5.0)
Target: x86_64-pc-linux-gnu
Thread model: posix
Found candidate GCC installation:
/usr/bin/../lib/gcc/i686-linux-gnu/4.9
Found candidate GCC installation:
/usr/bin/../lib/gcc/i686-linux-gnu/4.9.1
Found candidate GCC installation:
/usr/bin/../lib/gcc/x86_64-linux-gnu/4.8
Found candidate GCC installation:
/usr/bin/../lib/gcc/x86_64-linux-gnu/4.8.3
Found candidate GCC installation:
/usr/bin/../lib/gcc/x86_64-linux-gnu/4.9
Found candidate GCC installation:
/usr/bin/../lib/gcc/x86_64-linux-gnu/4.9.1
Found candidate GCC installation:
/usr/lib/gcc/i686-linux-gnu/4.9
Found candidate GCC installation:
/usr/lib/gcc/i686-linux-gnu/4.9.1
Found candidate GCC installation:
/usr/lib/gcc/x86_64-linux-gnu/4.8
Found candidate GCC installation:
/usr/lib/gcc/x86_64-linux-gnu/4.8.3
Found candidate GCC installation:
/usr/lib/gcc/x86_64-linux-gnu/4.9
Found candidate GCC installation:
/usr/lib/gcc/x86_64-linux-gnu/4.9.1
Selected GCC installation:
/usr/bin/../lib/gcc/x86_64-linux-gnu/4.9
Candidate multilib: .;@m64
Candidate multilib: 32;@m32
Candidate multilib: x32;@mx32
Selected multilib: .;@m64
```

[Revenir à la page principale du tutoriel](#)

[Revenir à la page principale](#)

<https://guillaumebelz.wordpress.com/category/android/>

Cet article a été écrit en 2013 pour Qt 5.1, puis mis à jour régulièrement, en fonction des sorties de Qt. Il est actuellement à 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 développer 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 fournis dans les kits de développement Android. Ce qui implique que pour développer avec Qt sur Android, il faut installer ces kits et 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) et 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. Une boîte de 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 l'avez déjà installé, lancez l'outil MaintenanceTool (qui se trouve dans le répertoire d'installation de Qt. Le processus d'installation est classique, vérifiez 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 dévelopeur de Qt Creator 2.8, compilée 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” Crédit à l'auteur 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 en 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\*\*](#)

[Revenir à la page principale](#)

# Installer Qt 5.5 pour iOS

[Revenir à la page principale](#)

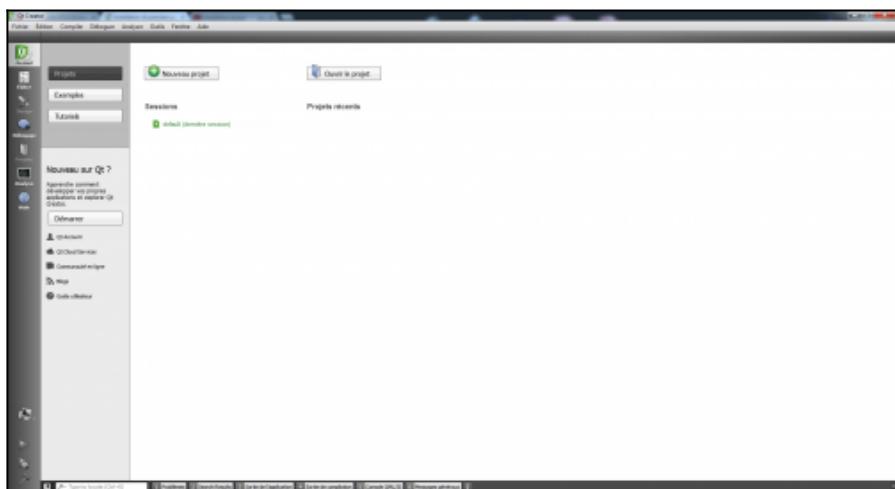
[Revenir à la page principale](#)

# Tester l'installation de Qt 5.5

## Avec Qt Creator

La première chose à faire après avoir installé Qt est de tester si l'installation s'est correctement déroulée. Pour cela, nous allons simplement créer un programme par défaut et l'exécuter. Si tout se passe bien, le programme se lancera et affichera une fenêtre.

Lorsque vous lancez l'éditeur Qt Creator, vous arrivez sur la page d'accueil suivante :



Quelques éléments de vocabulaire relatifs à Qt Creator, pour bien comprendre les choses. Les icônes à gauche en haut permettent de choisir le mode :

- “Accueil”, la page actuelle ;
- “Éditer”, lorsque vous éditerez un fichier ;

- “Design”, pour les éditeurs graphiques de Qt ;
- “Débogage”, pour corriger les programmes ;
- “Projet”, pour éditer les paramètres de compilation et d'exécution des projets ;
- “Analyse”, pour les outils d'analyse de performances ;
- “Aide”, pour les pages d'aide de Qt.

En dessous des icônes de mode (toujours dans la barre à gauche), une série d'icônes (actuellement griseée, puisqu'aucun projet n'est ouvert) permettent, de haut en bas :

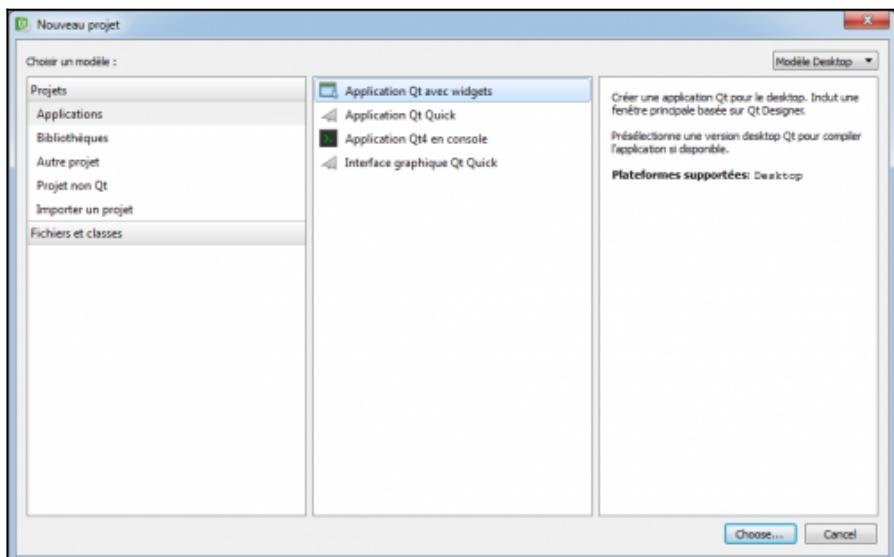
- de choisir le kit à utiliser pour la compilation et l'exécution (voir la suite pour les explications sur les kits, comment les configurer et les utiliser) ;
- de lancer le programme en mode normal (le triangle) ;
- de lancer le programme en mode Debug (le triangle avec un insecte - *bug* en anglais) ;
- de simplement compiler le programme, sans le lancer (bien sûr, les boutons précédents pour lancer le programme le compilent dans un premier temps, avant de le lancer).

En bas, une série d'onglets permet d'ouvrir des fenêtres de messages. Lors de la compilation ou lorsqu'il y a un problème, les messages s'affichent dans ces fenêtres. De gauche à droite :

- le localisateur (Ctrl+K) pour rechercher des classes, variables ou fonctions dans le projet ;
- la fenêtre de problèmes, qui affiche les messages d'erreur ;
- la fenêtre de recherche, pour afficher le résultat des recherches (Ctrl+F pour une recherche dans le fichier courant, Ctrl+Shift+F pour rechercher dans plusieurs fichiers) ;
- la sortie d'application, qui affiche les messages de l'application (par exemple avec `std::cout` ou `qDebug()`) ;
- la sortie de compilation, qui affiche les commandes lancées lors de la compilation. Cette fenêtre vous sera particulièrement utile en cas de problème de configuration du projet (fichier non trouvé

- par exemple) ;
- la console pour le QML et le JavaScript ;
- la fenêtre de messages généraux.

Pour créer un nouveau projet par défaut, vous pouvez aller dans le mode “Accueil” puis cliquer sur “Nouveau projet” ou aller dans le menu “Fichier” puis “Nouveau fichier ou projet...”. Un assistant vous permet de sélectionner le type de projet :



Il est possible de créer beaucoup de types de projet différents, il suffit de lire la description à droite pour savoir à quoi cela correspond.

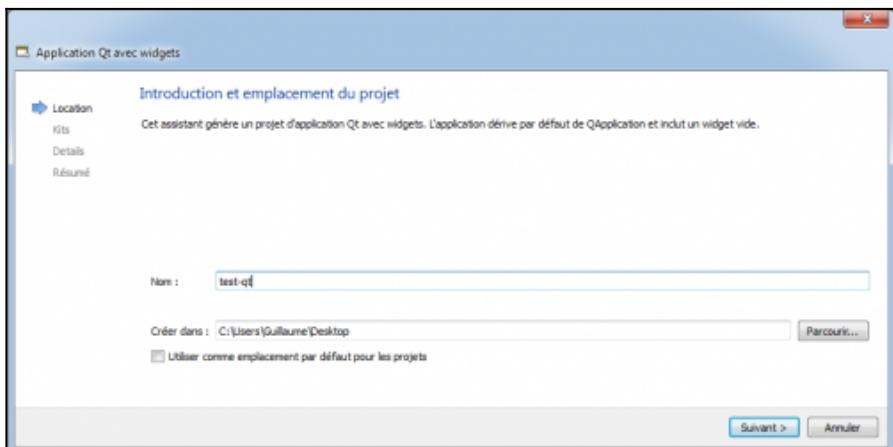
Pour les plus importants :

- “Application” puis “Application Qt avec widgets” pour les applications graphiques Qt classiques ;
- “Application” puis “Application Qt Quick” pour les applications graphiques Qt utilisant le nouveau langage de Qt : le QML ;
- “Application” puis “Application Qt console” pour les applications non graphiques Qt ;

- “Projet non Qt” puis “Projet C++” pour les applications C++ sans Qt ;
- “Importer un projet” pour créer un clone d'un projet existant dans un gestionnaire de versions (CVS, SVN, Git, etc.).

Choisissez “Application Qt avec widgets” pour ce premier test.

La page suivante permet de choisir le nom du projet que l'on souhaite créer et l'emplacement sur le disque. Remarque : ne mettez pas vos projets dans “C:\Qt”, créez un répertoire dédié pour cela, par exemple dans vos documents ou votre répertoire de travail.

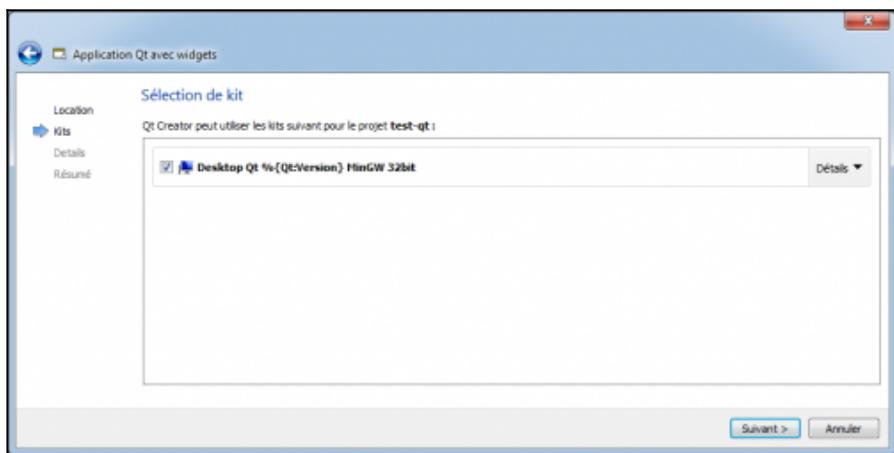


La page suivante permet de sélectionner les kits à utiliser pour compiler le programme. Il est possible de sélectionner plusieurs kits (voir la suite de ce tutoriel pour les explications sur les kits), pour le moment (si vous avez suivi les instructions de ce tutoriel et que c'est la première fois que vous installez Qt), vous n'avez qu'un seul kit disponible : “Qt MinGW”.

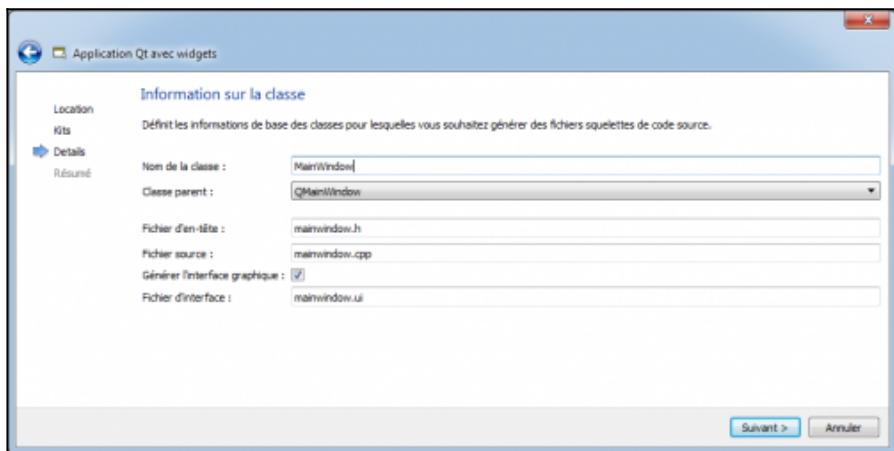
Si vous n'avez aucun kit disponible dans cette page, c'est qu'il y a eu un problème lors de l'installation (Qt Creator n'a pas réussi à trouver une version de Qt et un compilateur compatibles ensemble). Voir la suite de ce tutoriel pour les explications sur les kits.

Remarque : Qt 5.4 n'est pas encore totalement finalisé, il est possible

que vous ayez une erreur dans le message affiché, comme c'est le cas sur la copie d'écran. Rien de grave.

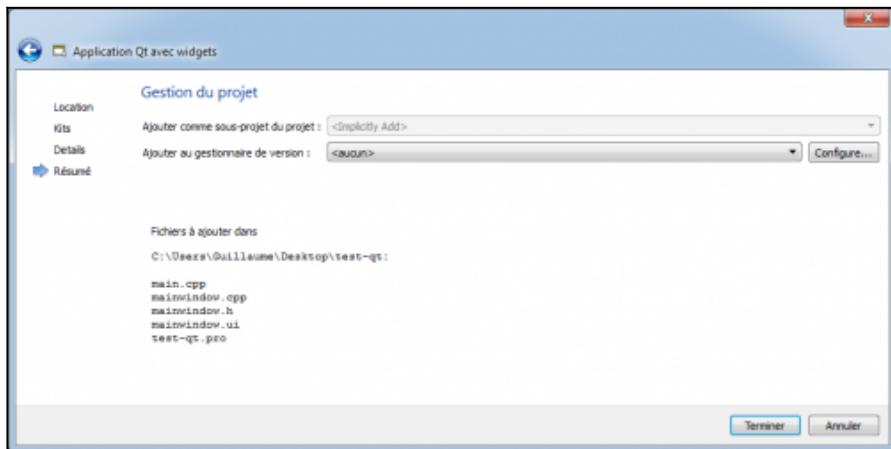


Le projet par défaut propose de créer une classe *MainWindow* (fenêtre principale). On va être gentil, on ne va pas le contrarier et le laisser faire. Cliquez sur Suivant.



Pour terminer, il est possible d'ajouter le projet dans un gestionnaire de versions. Cela n'est pas nécessaire pour ce test, mais n'hésitez pas à utiliser un tel gestionnaire, c'est très pratique et utile.

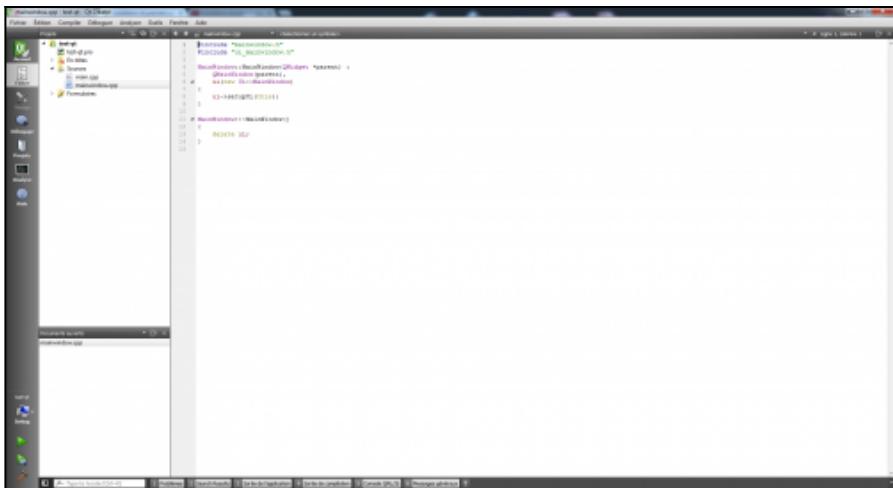
Cliquez sur “Terminer” pour créer le projet.



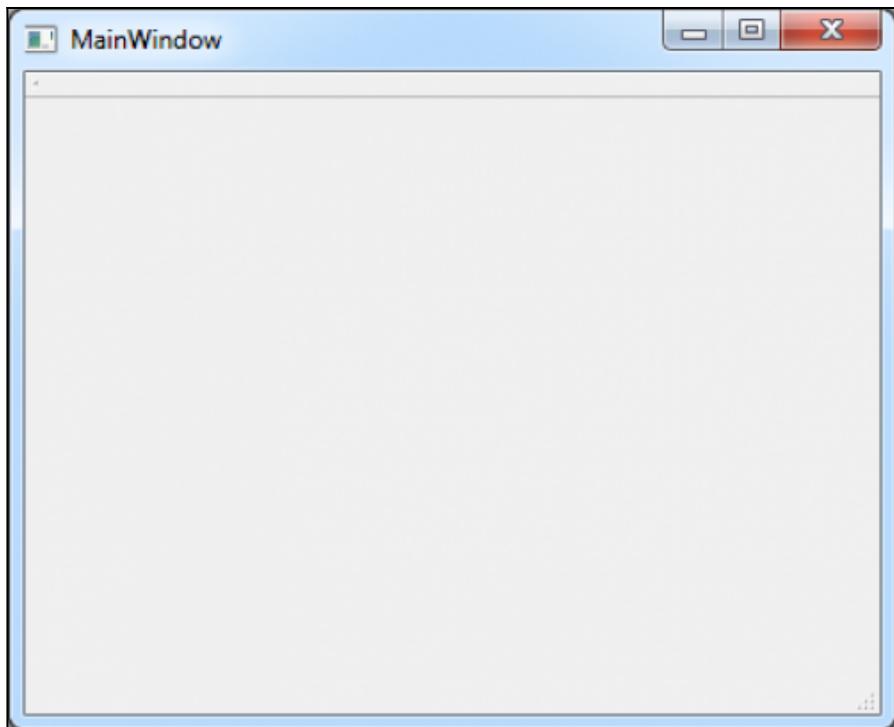
Qt Creator crée plusieurs fichiers et passe en mode “Éditer” pour afficher le contenu des fichiers. Un projet Qt contient généralement les fichiers suivants (cela peut changer en fonction du type de projet) :

- un fichier de projet `.pro` ou `.qmlproj`, qui contient les informations sur le projet (en particulier la liste des fichiers et les modules Qt à utiliser) ;
- les fichiers C++ d'en-tête (`.h`) et d'implémentation (`.cpp`). En particulier, le projet contient le fichier `main.cpp`, qui est le point de démarrage du programme ;
- les fichiers de formulaire `.ui`.

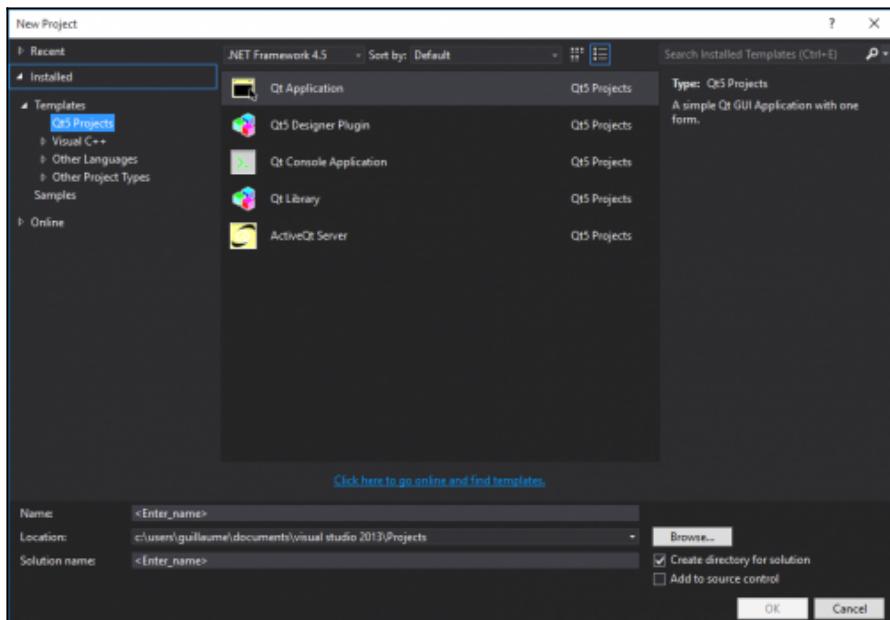
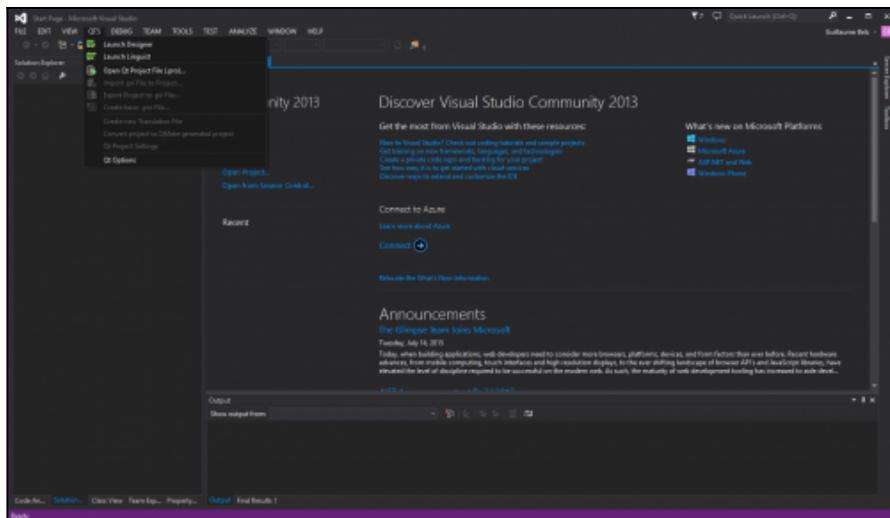
Cliquez sur les différents fichiers pour voir comment Qt Creator les affiche. Par exemple, les fichiers `.h` et `.cpp` sont affichés dans un éditeur de texte avec coloration syntaxique. Les fichiers `.ui` sont affichés en utilisant un éditeur graphique (mode Design).



Cliquez sur le triangle vert en bas à gauche, dans le menu “Compiler” puis “Exécuter” ou appuyez sur Ctrl+R pour lancer le programme. Si tout s'est bien passé, une fenêtre “MainWindow” devrait s'ouvrir.



## Avec Visual Studio 2013





## Welcome to the Qt5 GUI project wizard

This wizard generates a Qt5 GUI application project. The application derives by default from QApplication and includes an empty widget.

### Overview

Project Settings

Generated Class

These are currently selected Qt modules:

Core, GUI, Widgets

Click **Finish** from any window to accept the current settings.

< Previous

Next >

Finish

Cancel

Help



## Welcome to the Qt5 GUI project wizard

This wizard generates a Qt5 GUI application project. The application derives by default from QApplication and includes an empty widget.

### Overview

#### Project Settings

##### Generated Class

Select the modules you want to include in your project. The recommended modules for this project are selected by default.

- |   |  |   |
|---|--|---|
| <input type="checkbox"/> 3D                 | <input type="checkbox"/> Multimedia Widgets    | <input type="checkbox"/> SQL                |
| <input type="checkbox"/> ActiveQt container | <input type="checkbox"/> Network               | <input type="checkbox"/> SVG                |
| <input type="checkbox"/> ActiveQt server    | <input type="checkbox"/> OpenGL                | <input type="checkbox"/> System Info        |
| <input type="checkbox"/> Bluetooth          | <input type="checkbox"/> Organizer             | <input type="checkbox"/> Test               |
| <input type="checkbox"/> Concurrent         | <input type="checkbox"/> Phonon                | <input type="checkbox"/> Versit             |
| <input type="checkbox"/> Contacts           | <input type="checkbox"/> Print Support         | <input type="checkbox"/> WebKit             |
| <input checked="" type="checkbox"/> Core    | <input type="checkbox"/> Publish and Subscribe | <input type="checkbox"/> Webkit Widgets     |
| <input type="checkbox"/> Declarative        | <input type="checkbox"/> QML                   | <input checked="" type="checkbox"/> Widgets |
| <input checked="" type="checkbox"/> GUI     | <input type="checkbox"/> Quick                 | <input type="checkbox"/> XML                |
| <input type="checkbox"/> Help               | <input type="checkbox"/> Script                | <input type="checkbox"/> XML Patterns       |
| <input type="checkbox"/> Location           | <input type="checkbox"/> Sensors               |   |
| <input type="checkbox"/> Multimedia         | <input type="checkbox"/> Service Framework     |   |

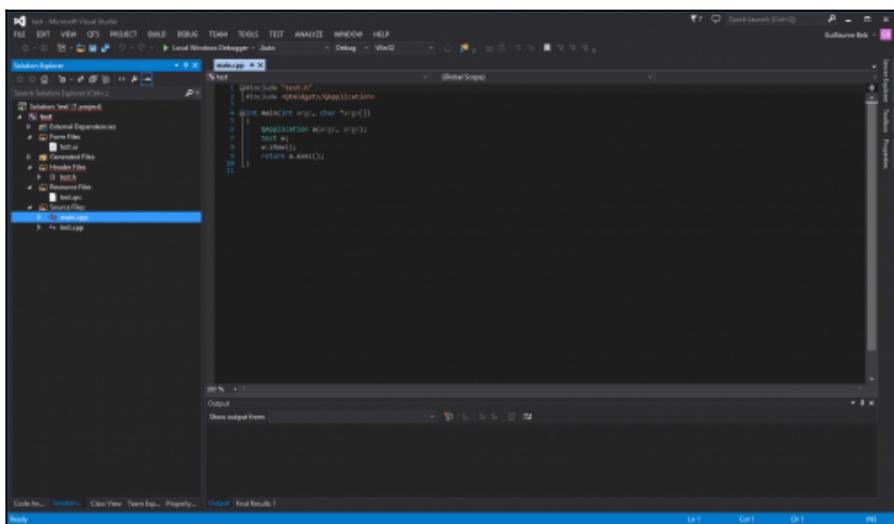
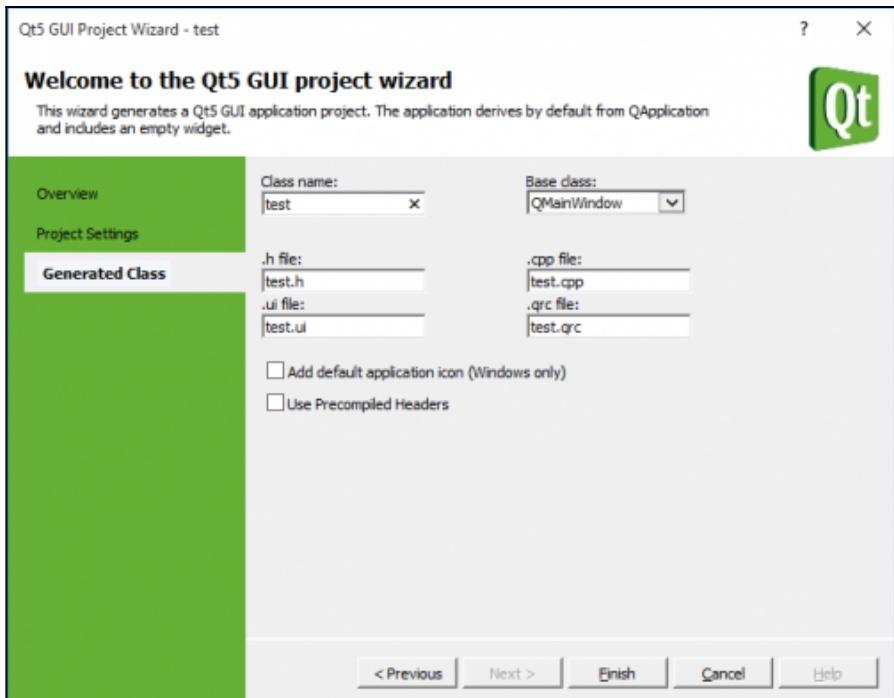
&lt; Previous

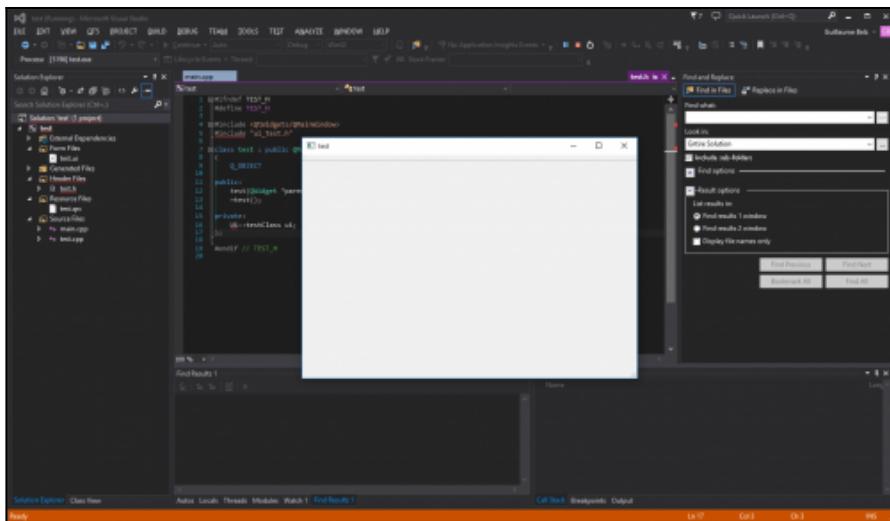
Next &gt;

Finish

Cancel

Help





[Revenir à la page principale](#)

# Configurer Qt Creator 3.4.2 pour Qt 5.5

## Quelques points de vocabulaire

Juste pour faciliter les discussions sur les forums, voici quelques notions à avoir :

- **langage de programmation** : un programme est écrit dans un langage de programmation. Avec Qt, ce langage sera le C++ ou parfois Python (pour PyQt). Qt propose aussi deux autres langages : le QML et le JavaScript.
- **bibliothèque** (et pas “librairie”, traduction incorrecte du terme anglais “library”) : une bibliothèque est un ensemble d’outils pour étendre les fonctionnalités d’un langage. Pour le C++, vous connaissez probablement la STL (la bibliothèque standard du C++). Qt est aussi une bibliothèque (voire même un ensemble de bibliothèques, on parle alors de framework).
- **compilateur** : c'est le programme utilisé pour convertir votre code en programme exécutable. Le compilateur (ou plus précisément, l'ensemble des outils de compilation) seront généralement appelés automatiquement par l'IDE. Mais sachez qu'il est possible d'appeler soi-même ces outils.
- **éditeur** : c'est le logiciel que vous utilisez pour éditer vos fichiers. Un éditeur avancé proposera au moins la coloration syntaxique (afficher le code selon un code de couleurs, pour faciliter la lecture) et l'auto-complétion (proposer des syntaxes correspondant à ce que vous êtes en train d'écrire).
- **IDE** (ou EDI, selon l'humeur des gens, pour “Integrated Development Environment” ou “Environnement de

Développement Intégré") est un éditeur "intelligent", qui propose des outils pour faciliter le développement. Qt Creator est l'IDE fourni avec Qt, il permet en particulier de lancer directement la compilation, d'accéder à l'aide (en appuyant sur F1), etc.

- **SDK** (ou kit de développement) est un ensemble d'outils pour développer. Par exemple, lorsque vous téléchargez Qt, vous téléchargez en réalité le "Qt SDK", qui contient en particulier le framework Qt, le compilateur MinGW ou GCC, l'IDE Qt Creator.

Donc, pour résumer, Qt n'est pas un langage. Ce n'est pas non plus un compilateur ou un éditeur. C'est simplement un framework. Ne confondez pas les choses et ne dites pas que vous avez écrit un programme en Qt (puisque ce n'est pas un langage - il faudrait dire "écrit un programme en C++ avec Qt").

## Kits, compilateurs et versions de Qt

Donc, pour créer un programme, il faut :

- un compilateur qui convertit le code C++ en programme ;
- une version de Qt compatible avec le compilateur ;
- accessoirement, un débogueur ("accessoirement" parce que généralement, le débogueur est fourni avec le compilateur, pas parce que le débogueur est accessoire...).

L'IDE Qt Creator permet de gérer plusieurs compilateurs et versions de Qt en même temps. Cela est particulièrement utile si vous faites de la compilation vers Android ou iOS. Si vous avez suivi la procédure d'installation de Qt décrite ci-dessus, Qt Creator a normalement pu configurer correctement votre environnement de compilation et vous devriez pouvoir compiler directement une application.

Pour pouvoir gérer plusieurs configurations différentes, Qt Creator utilise un système de kits. Un kit est tout simplement l'association d'un compilateur, d'un débogueur et d'une version de Qt. Un kit est valide

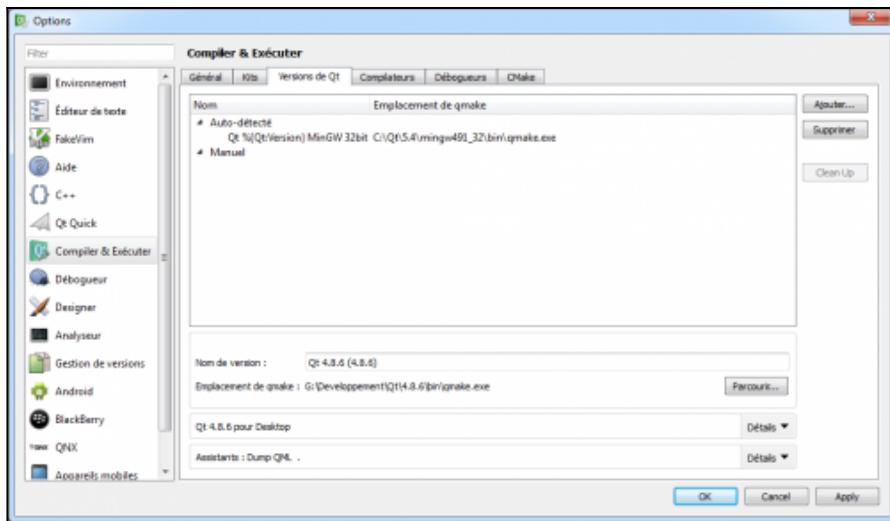
lorsque tous ces éléments sont compatibles. Si Qt Creator ne vous propose pas de kit lors de la création d'un projet, c'est peut-être parce que vous n'avez pas de kit valide (cela arrive souvent lorsque les gens installent Qt pour Microsoft Visual C++, mais sans installer ce compilateur).

Pour connaître les kits utilisables sur votre ordinateur et les configurer, allez dans le menu "Outils" puis "Options..." puis "Compiler & Exécuter". Ce dialogue possède des onglets qui vont nous intéresser :

- Kits ;
- Versions de Qt ;
- Compilateurs ;
- Débogeur.

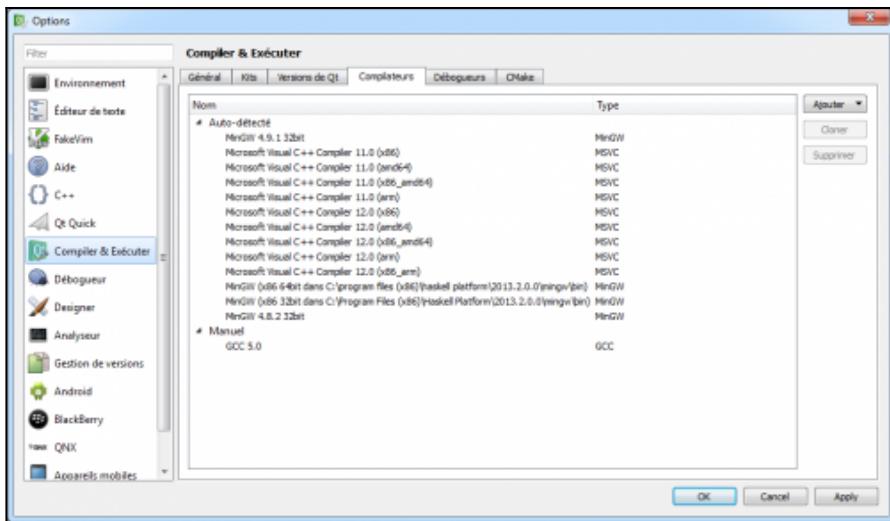
Comme vous l'avez sûrement compris, chacun de ces onglets permet de configurer les différents outils utilisés pour la compilation.

L'onglet **Versions de Qt** affiche les différentes versions de Qt installées. Une version de Qt est identifiée par un numéro de version de Qt (actuellement Qt 5.5, mais vous pouvez également avoir Qt 5.4, 5.3, etc.) et par le compilateur utilisé pour compiler Qt (MinGW 4.9.1, MSVC 2013 32b OpenGL, etc.) Il faut compiler votre programme avec le même compilateur utilisé pour compiler Qt.



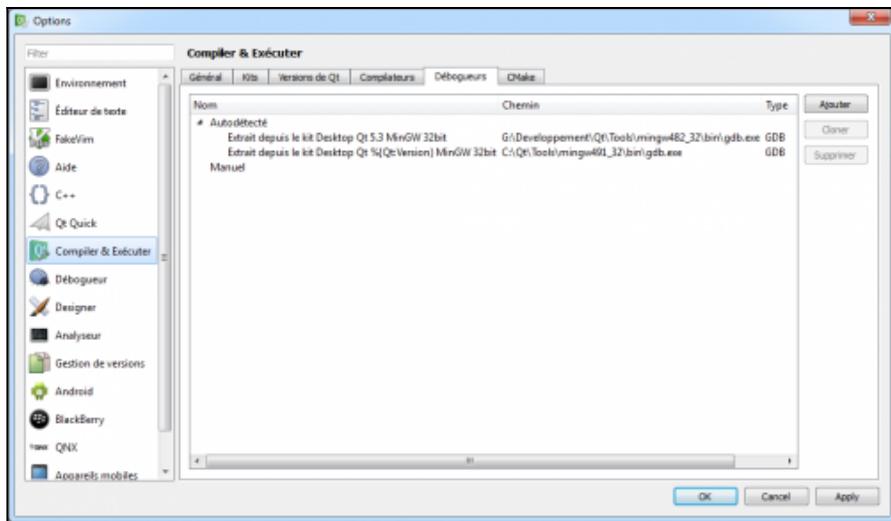
S'il manque une version de Qt, vous pouvez cliquer sur le bouton “Ajouter” et aller dans le répertoire de la version de Qt manquante, puis dans le sous-répertoire **bin**, sélectionner **qmake**. Par exemple, si vous avez suivi la procédure d'installation décrite ci-dessus et que Qt n'est pas reconnu, il faudra ajouter `C:\Qt\5.4\mingw491_32\bin`.

L'onglet **Compilateurs** affiche la liste des compilateurs connus. Il faut bien sûr au moins un compilateur valide pour compiler un programme. Qt Creator trouvera les compilateurs installés dans les répertoires par défaut. Si ce n'est pas le cas, cliquez sur le bouton “Ajouter” et allez chercher l'application g++ (pour MinGW et GCC), clang++ (pour LLVM/Clang) et cl.exe pour MSVC.



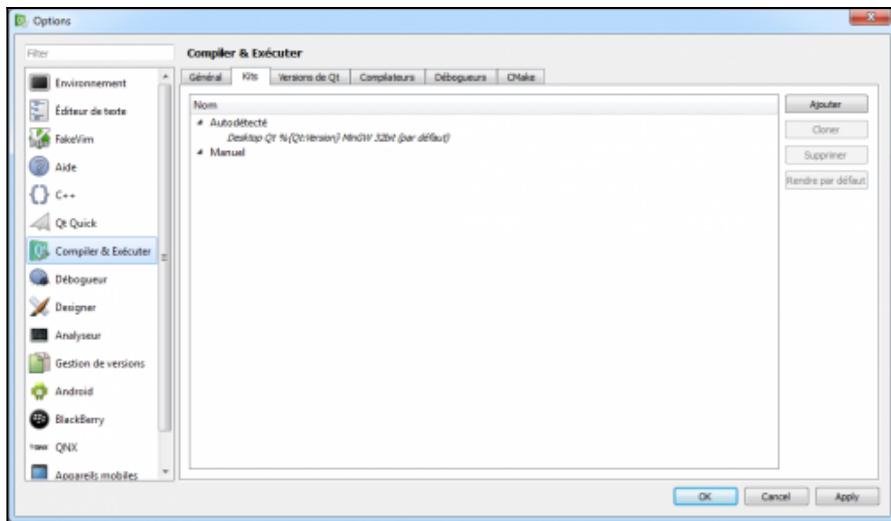
Les compilateurs se déclinent en plusieurs versions, il faudra bien choisir la version correspondant à la version de Qt utilisée. Le nom du compilateur correspondant à une version de Qt est indiqué dans la version de Qt.

Pour le **Débogueur**, si vous avez installé les compilateurs MinGW, GCC ou Clang, le débogueur est inclus dedans et devrait être directement reconnu (s'il est installé dans un répertoire par défaut). Pour MSVC, il faut installer le Windows SDK en complément.



Un débogueur ne peut fonctionner qu'avec le compilateur correspondant. Si vous utilisez MSVC, vous ne pouvez pas utiliser GDB par exemple.

Une fois que vous avez vérifié que tous les outils sont individuellement reconnus par Qt Creator, vous pouvez configurer les kits. De la même façon que pour les autres outils, Qt Creator créera différents kits par défaut, s'il arrive à trouver des outils compatibles entre eux.



Si un kit n'est pas valide, il y aura un triangle rouge sur le kit et un message indiquera le problème. Il faut obligatoirement au moins un kit valide pour compiler. Pour qu'un kit soit valide, il faut utiliser un compilateur compatible avec une version de Qt.

L'une des erreurs les plus courantes est d'installer une version de Qt pour MSVC sans installer MSVC (l'erreur vient du cours C++ de OpenClassroom).

Remarque : si on vous demande sur un forum avec quelle version de Qt et quel compilateur vous avez compilé votre programme, il faut donner les versions du kit que vous utilisez. Certains font parfois l'erreur d'aller dans le menu "Aide" puis "À propos de Qt Creator..." et donnent la version de Qt utilisée pour compiler Qt Creator, pas leur programme.

[Revenir à la page principale](#)

# Configurer Qt Creator 3.4.2 pour Qt 5.5

## Quelques points de vocabulaire

Juste pour faciliter les discussions sur les forums, voici quelques notions à avoir :

- **langage de programmation** : un programme est écrit dans un langage de programmation. Avec Qt, ce langage sera le C++ ou parfois Python (pour PyQt). Qt propose aussi deux autres langages : le QML et le JavaScript.
- **bibliothèque** (et pas “librairie”, traduction incorrecte du terme anglais “library”) : une bibliothèque est un ensemble d’outils pour étendre les fonctionnalités d’un langage. Pour le C++, vous connaissez probablement la STL (la bibliothèque standard du C++). Qt est aussi une bibliothèque (voire même un ensemble de bibliothèques, on parle alors de framework).
- **compilateur** : c'est le programme utilisé pour convertir votre code en programme exécutable. Le compilateur (ou plus précisément, l'ensemble des outils de compilation) seront généralement appelés automatiquement par l'IDE. Mais sachez qu'il est possible d'appeler soi-même ces outils.
- **éditeur** : c'est le logiciel que vous utilisez pour éditer vos fichiers. Un éditeur avancé proposera au moins la coloration syntaxique (afficher le code selon un code de couleurs, pour faciliter la lecture) et l'auto-complétion (proposer des syntaxes correspondant à ce que vous êtes en train d'écrire).
- **IDE** (ou EDI, selon l'humeur des gens, pour “Integrated Development Environment” ou “Environnement de

Développement Intégré") est un éditeur "intelligent", qui propose des outils pour faciliter le développement. Qt Creator est l'IDE fourni avec Qt, il permet en particulier de lancer directement la compilation, d'accéder à l'aide (en appuyant sur F1), etc.

- **SDK** (ou kit de développement) est un ensemble d'outils pour développer. Par exemple, lorsque vous téléchargez Qt, vous téléchargez en réalité le "Qt SDK", qui contient en particulier le framework Qt, le compilateur MinGW ou GCC, l'IDE Qt Creator.

Donc, pour résumer, Qt n'est pas un langage. Ce n'est pas non plus un compilateur ou un éditeur. C'est simplement un framework. Ne confondez pas les choses et ne dites pas que vous avez écrit un programme en Qt (puisque ce n'est pas un langage - il faudrait dire "écrit un programme en C++ avec Qt").

## Kits, compilateurs et versions de Qt

Donc, pour créer un programme, il faut :

- un compilateur qui convertit le code C++ en programme ;
- une version de Qt compatible avec le compilateur ;
- accessoirement, un débogueur ("accessoirement" parce que généralement, le débogueur est fourni avec le compilateur, pas parce que le débogueur est accessoire...).

L'IDE Qt Creator permet de gérer plusieurs compilateurs et versions de Qt en même temps. Cela est particulièrement utile si vous faites de la compilation vers Android ou iOS. Si vous avez suivi la procédure d'installation de Qt décrite ci-dessus, Qt Creator a normalement pu configurer correctement votre environnement de compilation et vous devriez pouvoir compiler directement une application.

Pour pouvoir gérer plusieurs configurations différentes, Qt Creator utilise un système de kits. Un kit est tout simplement l'association d'un compilateur, d'un débogueur et d'une version de Qt. Un kit est valide

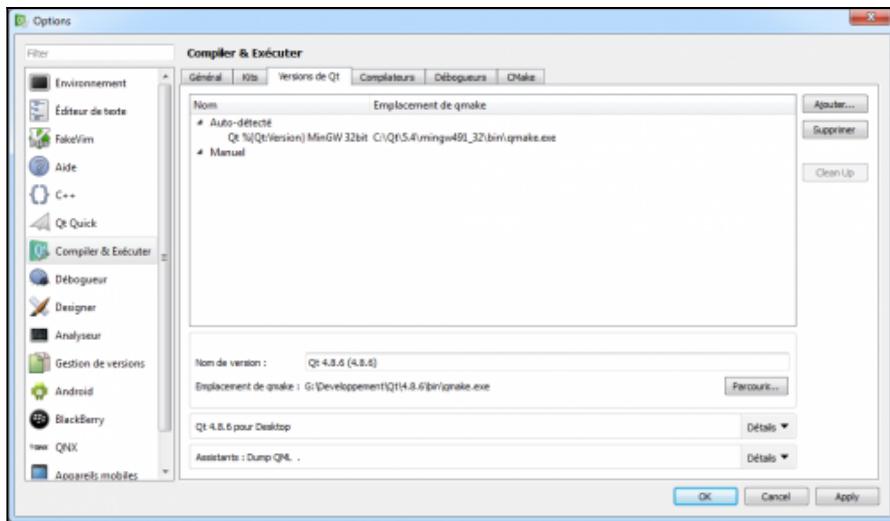
lorsque tous ces éléments sont compatibles. Si Qt Creator ne vous propose pas de kit lors de la création d'un projet, c'est peut-être parce que vous n'avez pas de kit valide (cela arrive souvent lorsque les gens installent Qt pour Microsoft Visual C++, mais sans installer ce compilateur).

Pour connaître les kits utilisables sur votre ordinateur et les configurer, allez dans le menu "Outils" puis "Options..." puis "Compiler & Exécuter". Ce dialogue possède des onglets qui vont nous intéresser :

- Kits ;
- Versions de Qt ;
- Compilateurs ;
- Débogeur.

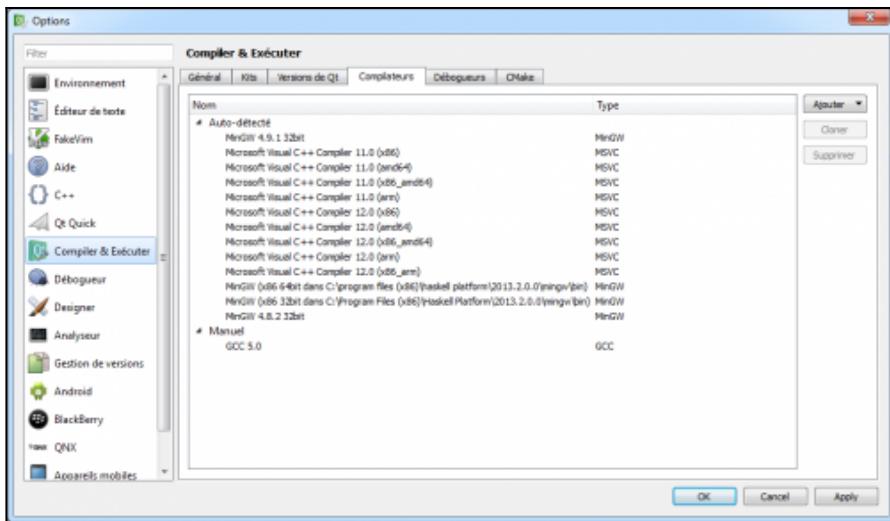
Comme vous l'avez sûrement compris, chacun de ces onglets permet de configurer les différents outils utilisés pour la compilation.

L'onglet **Versions de Qt** affiche les différentes versions de Qt installées. Une version de Qt est identifiée par un numéro de version de Qt (actuellement Qt 5.5, mais vous pouvez également avoir Qt 5.4, 5.3, etc.) et par le compilateur utilisé pour compiler Qt (MinGW 4.9.1, MSVC 2013 32b OpenGL, etc.) Il faut compiler votre programme avec le même compilateur utilisé pour compiler Qt.



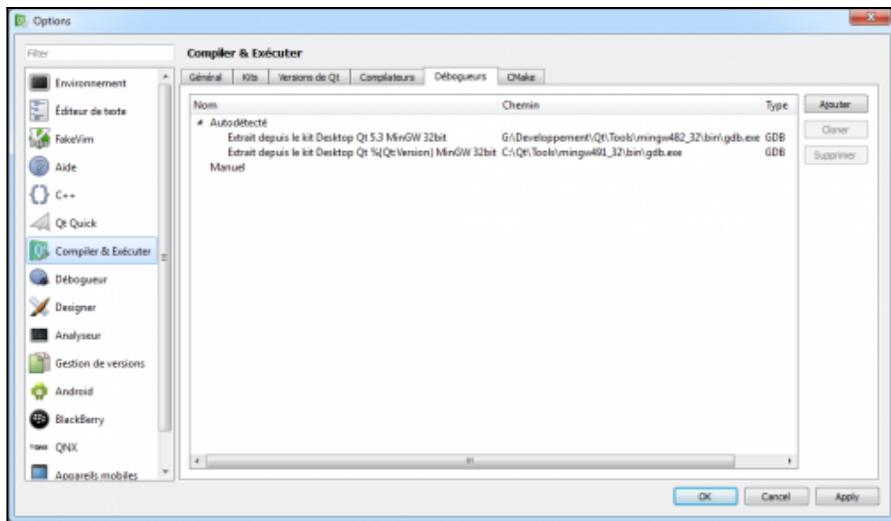
S'il manque une version de Qt, vous pouvez cliquer sur le bouton “Ajouter” et aller dans le répertoire de la version de Qt manquante, puis dans le sous-répertoire **bin**, sélectionner **qmake**. Par exemple, si vous avez suivie la procédure d'installation décrite ci-dessus et que Qt n'est pas reconnue, il faudra ajouter `C:\Qt\5.4\mingw491_32\bin`.

L'onglet **Compilateurs** affiche la liste des compilateurs connus. Il faut bien sûr au moins un compilateur valide pour compiler un programme. Qt Creator trouvera les compilateurs installés dans les répertoires par défaut. Si ce n'est pas le cas, cliquez sur le bouton “Ajouter” et allez chercher l'application g++ (pour MinGW et GCC), clang++ (pour LLVM/Clang) et cl.exe pour MSVC.



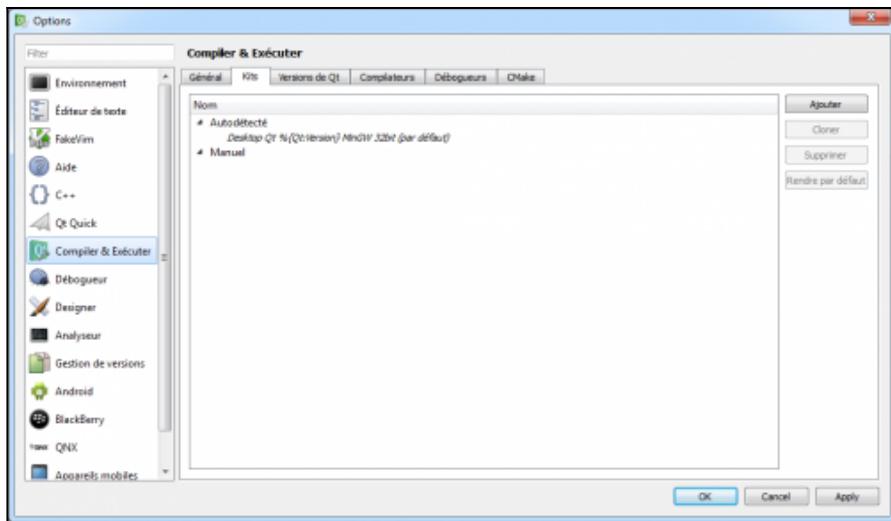
Les compilateurs se déclinent en plusieurs versions, il faudra bien choisir la version correspondant à la version de Qt utilisée. Le nom du compilateur correspondant à une version de Qt est indiqué dans la version de Qt.

Pour le **Débogueur**, si vous avez installé les compilateurs MinGW, GCC ou Clang, le débogueur est inclus dedans et devrait être directement reconnu (s'il est installé dans un répertoire par défaut). Pour MSVC, il faut installer le Windows SDK en complément.



Un débogueur ne peut fonctionner qu'avec le compilateur correspondant. Si vous utilisez MSVC, vous ne pouvez pas utiliser GDB par exemple.

Une fois que vous avez vérifié que tous les outils sont individuellement reconnus par Qt Creator, vous pouvez configurer les kits. De la même façon que pour les autres outils, Qt Creator créera différents kits par défaut, s'il arrive à trouver des outils compatibles entre eux.



Si un kit n'est pas valide, il y aura un triangle rouge sur le kit et un message indiquera le problème. Il faut obligatoirement au moins un kit valide pour compiler. Pour qu'un kit soit valide, il faut utiliser un compilateur compatible avec une version de Qt.

L'une des erreurs les plus courantes est d'installer une version de Qt pour MSVC sans installer MSVC (l'erreur vient du cours C++ de OpenClassroom).

Remarque : si on vous demande sur un forum avec quelle version de Qt et quel compilateur vous avez compilé votre programme, il faut donner les versions du kit que vous utilisez. Certains font parfois l'erreur d'aller dans le menu "Aide" puis "À propos de Qt Creator..." et donnent la version de Qt utilisée pour compiler Qt Creator, pas leur programme.

[Revenir à la page principale](#)

[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.

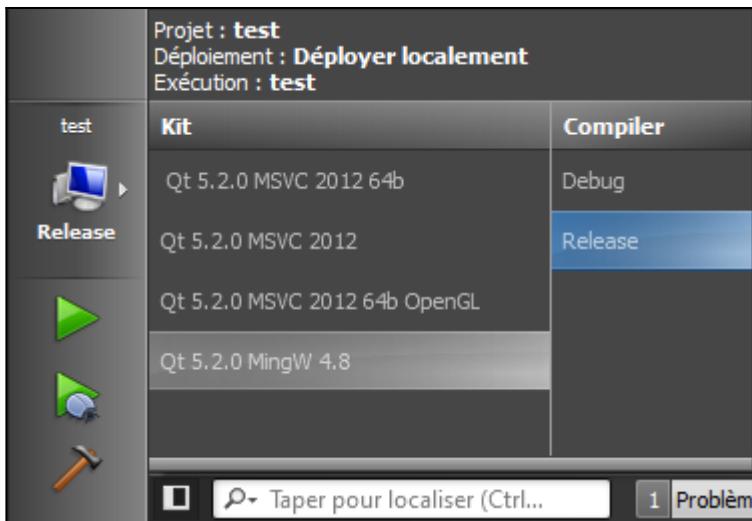
## La problématique

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 ne 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

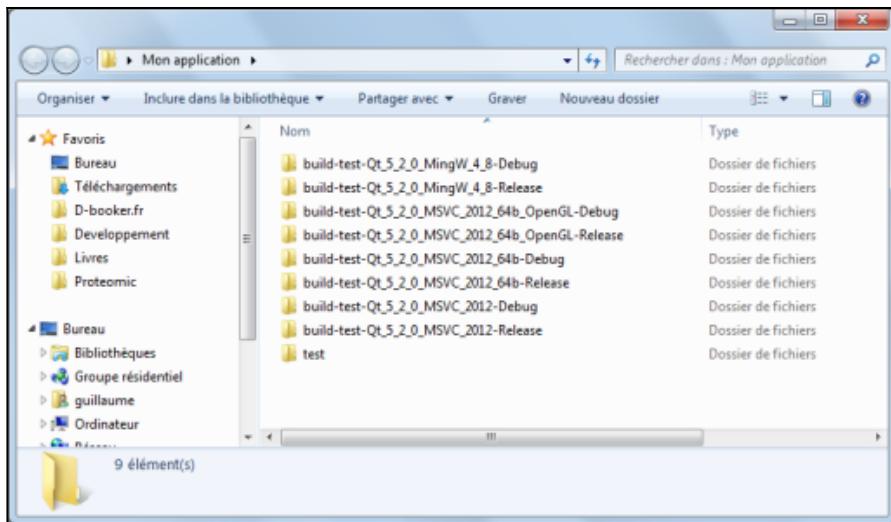
La page d'aide de la documentation de Qt est la suivante : [Deploying Qt Applications](#).

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).



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](#).



## 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 nécessitent 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 licence, 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` ;

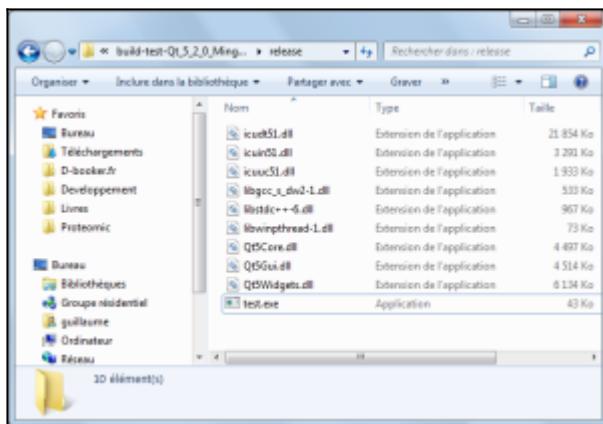
- `libGLESv2.dll` ;
- `d3dcompiler_XX.dll` (dans `C:\windows\system32`, sera fourni par Windows en général).

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 utilisez 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 utilisez 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 utilisez 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 inattendue.

## 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](#)

[Revenir à la page principale du tutoriel](#)

# Savoir utiliser la documentation de Qt 5.5

<https://www.youtube.com/watch?v=qKWtfDNfbzA>

[Revenir à la page principale du tutoriel](#)