

**Qu’est-ce que c’est QT**

C’est une API orientée objet et développée en C++, conjointement par The Qt Company et Qt Project.

Elle est composé d’un ensemble d’outils de conception et développement.

Par certains aspects, elle ressemble à un framework lorsqu'on l'utilise pour concevoir des interfaces graphiques ou que l'on conçoit l'architecture de son application en utilisant les mécanismes des signaux et slots par exemple.

Qt permet la portabilité des applications qui n'utilisent que ses composants par simple recompilation du code source. Les environnements pris en charge sont les Unix (dont GNU/Linux) qui utilisent le système graphique X Window System ou Wayland, Windows, Mac OS X, Tizen et également Genode. Le fait d'être une bibliothèque logicielle multiplateforme attire un grand nombre de personnes qui ont donc l'occasion de diffuser leurs programmes sur les principaux OS existants.

**Historique**

C'est au Norwegian Institute of Technology à Trondheim que Haavard Nord (CEO de Trolltech) et Eirik Chambe-Eng (président de Trolltech) se rencontrent. En 1988, Haavard Nord est chargé par une entreprise suédoise de développer une bibliothèque logicielle en C++ pour gérer une interface graphique, c'est la première fois qu'il commence à aborder le sujet4. Deux ans plus tard, avec Chambe-Eng, ils développent une application multiplateforme (Unix, Macintosh et Windows) et commencent sérieusement à réfléchir à la conception d'une bibliothèque graphique multiplateforme généraliste.

En 1991, ils entament le développement de cette bibliothèque. L'année suivante, Chambe-Eng propose le principe des « signaux et slots », qui devient la pierre angulaire de Qt. Et en 1993, le noyau de Qt est prêt et permet aux informaticiens de développer leurs propres composants graphiques. C'est à la fin de cette année que Haavard Nord propose de créer une entreprise pour commercialiser leur bibliothèque.

**Versions**

Actuellement, on est à la version Qt5 sorti en [19](https://fr.wikipedia.org/wiki/19_d%C3%A9cembre) [décembre](https://fr.wikipedia.org/wiki/D%C3%A9cembre_2012) [2012](https://fr.wikipedia.org/wiki/2012_en_informatique). (26 May 2020. 5.12)

**Qt Quick** est un framework libre développé et maintenu par Digia faisant partie de la bibliothèque Qt. Il fournit la possibilité de créer des interfaces utilisateur personnalisables et dynamiques avec des effets de transition fluides de manière déclarative. Ce type d'interface dynamique est de plus en plus commune, notamment sur les smartphones. Qt Quick inclut un langage de script déclaratif appelé QML comparable au XAML créé par Microsoft pour sa bibliothèque WPF.

Qt Quick et QML sont officiellement fournis depuis Qt 4.7 (avec Qt Creator 2.1)

**Architecture**

L'API Qt est constituée de classes aux noms préfixés par Q et dont chaque mot commence par une majuscule (ex: QLineEdit), c'est la typographie camel case. Ces classes ont souvent pour attributs des types énumérés déclarés dans l'espace de nommage Qt20. Mis à part une architecture en pur objet, certaines fonctionnalités basiques sont implémentées par des macros (chaîne de caractères à traduire avec tr, affichage sur la sortie standard avec qDebug…).

Les conventions de nommage des méthodes sont assez semblables à celles de Java : le lower camel case est utilisé, c'est-à-dire que tous les mots sauf le premier prennent une majuscule (ex: indicatorFollowsStyle()), les modificateurs sont précédés par set, en revanche les accesseurs prennent simplement le nom de l'attribut (ex : text()) ou commencent par is dans le cas des booléens (ex : isChecked()).

**Licences utilisées dans Qt**

Le Framework et les outils Qt sont disponibles sous licences open source et commerciales.

**Arborescence des objets**

Les objets Qt (ceux héritant de QObject) peuvent s'organiser d'eux-mêmes sous forme d'arbre. Ainsi, lorsqu'une classe est instanciée, on peut lui définir un objet parent. Cette organisation des objets sous forme d'arbre facilite la gestion de la mémoire car avant qu'un objet parent ne soit détruit, Qt appelle récursivement le destructeur de tous les enfants.

Cette notion d'arbre des objets permet également de débugger plus facilement, via l'appel de méthodes comme QObject::dumpObjectTree() et Object::dumpObjectInfo()

**Arborescence des objets**

Le moc22 (pour Meta Object Compiler) est un préprocesseur qui, appliqué avant compilation du code source d'un programme Qt, génère des meta-informations relatives aux classes utilisées dans le programme. Ces meta-informations sont ensuite utilisées par Qt pour fournir des fonctions non disponibles en C++, comme les signaux et slots et l'introspection.

L'utilisation d'un tel outil additionnel démarque les programmes Qt du langage C++ standard. Ce fonctionnement est vu par Qt Development Frameworks comme un compromis nécessaire pour fournir l'introspection et les mécanismes de signaux. À la sortie de Qt 1.x, les implémentations des templates par les compilateurs C++ n'étaient pas suffisamment homogènes

**Signaux et slots**

Les signaux et slots sont une implémentation du patron de conception observateur. L'idée est de connecter des objets entre eux via des signaux qui sont émis et reçus par des slots. Du point de vue du développeur, les signaux sont représentés comme de simples méthodes de la classe émettrice, dont il n'y a pas d'implémentation. Ces « méthodes » sont par la suite appelées, en faisant précéder « emit », qui désigne l'émission du signal. Pour sa part, le slot connecté à un signal est une méthode de la classe réceptrice, qui doit avoir la même signature (autrement dit les mêmes paramètres que le signal auquel il est connecté), mais à la différence des signaux, il doit être implémenté par le développeur. Le code de cette implémentation représente les actions à réaliser à la réception du signal.

C'est le MOC qui se charge de générer le code C++ nécessaire pour connecter les signaux et les slots

**Concepteur d'interface**

Qt Designer est un logiciel qui permet de créer des interfaces graphiques Qt dans un environnement convivial. L'utilisateur, par glisser-déposer, place les composants d'interface graphique et y règle leurs propriétés facilement. Les fichiers d'interface graphique sont formatés en XML et portent l'extension

Lors de la compilation, un fichier d'interface graphique est converti en classe C++ par l'utilitaire uic. Il y a plusieurs manières pour le développeur d'employer cette classe

* l'instancier directement et connecter les signaux et slots
* l'agréger au sein d'une autre classe
* l'hériter pour en faire une classe mère et ayant accès ainsi à tous les éléments constitutifs de l'interface créée
* la générer à la volée avec la classe QUiLoader qui se charge d'interpréter le fichier XML .ui et retourner une instance de classe QWidget

**qmake**

Qt se voulant un environnement de développement portable et ayant le MOC comme étape intermédiaire avant la phase de compilation/édition de liens, il a été nécessaire de concevoir un moteur de production spécifique. C'est ainsi qu'est conçu le programme qmake.

Ce dernier prend en entrée un fichier (avec l'extension .pro) décrivant le projet (liste des fichiers sources, dépendances, paramètres passés au compilateur, etc.) et génère un fichier de projet spécifique à la plateforme. Ainsi, sous les systèmes UNIX qmake produit un Makefile qui contient la liste des commandes à exécuter pour génération d'un exécutable, à l'exception des étapes spécifiques à Qt (génération des classes C++ lors de la conception d'interface graphique avec Qt Designer, génération du code C++ pour lier les signaux et les slots, ajout d'un fichier au projet, etc.).

Le fichier de projet est fait pour être très facilement éditable par un développeur. Il consiste en une série d'affectations de variables. En voici un exemple pour un petit projet:

**Installation**

L’installation de qt peut faire sous trois forme :

Les programmes d'installation vous permettent de télécharger et d'installer les composants suivants:

Bibliothèques Qt, prédéfinies pour une plate-forme de développement particulière (système d'exploitation et compilateur)

Environnement de développement intégré (IDE) Qt Creator

Documentation de référence et exemples

Qt sous forme de code source (nécessaire uniquement si vous souhaitez créer vous-même le framework et les outils)

Composants complémentaires pertinents pour différentes plates-formes

**En ligne**

Dans le programme d'installation en ligne , vous pouvez sélectionner des versions commerciales ou open source de Qt, des outils et des composants complémentaires à installer. L'utilisation du programme d'installation en ligne nécessite une connexion à votre compte Qt . Le programme d'installation récupère la licence attachée au compte à partir d'un serveur et fournit une liste des composants disponibles correspondant à la licence.

Le programme d'installation en ligne est disponible à partir de la page Téléchargements de Qt

(<http://qt.io/download>)

**NB**: la version en ligne (online), qui contient uniquement l'installateur. Tous les outils seront installés après téléchargement en ligne

**Hors ligne**

Un programme d'installation hors ligne est un package unique qui contient tous les Qt et modules complémentaires pertinents pour une plate-forme cible. Les utilisateurs disposant d'une licence commerciale peuvent se connecter avec leur compte Qt ou avec leur clé de licence Qt pendant le processus d'installation. La clé de licence est disponible sur le portail Web du compte Qt.

Le programme d'installation hors ligne est disponible <https://www.qt.io/offline-installer>.

**NB**: la version hors ligne (offline), 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.

* **Installer le compilateur C++**

Pour écrire des programmes en C++ avec Qt, il est nécessaire d'installer différents outils. Dans de nombreux cas, l'installation de Qt permet d'installer tous les outils de base, mais il faudra parfois installer des outils supplémentaires.

Le premier (et seul) outil à installer pour programmer en C++ est un compilateur C++. Un compilateur est un outil qui prend du code C++ dans des fichiers textes et produit des binaires.

Certains compilateurs sont disponibles sur plusieurs systèmes, alors que d'autres compilateurs sont spécifique d'un système. Pour les plus connus sur Desktop.

* **SOUS WINDOWS**
* **Visual Studio (MSVC)**

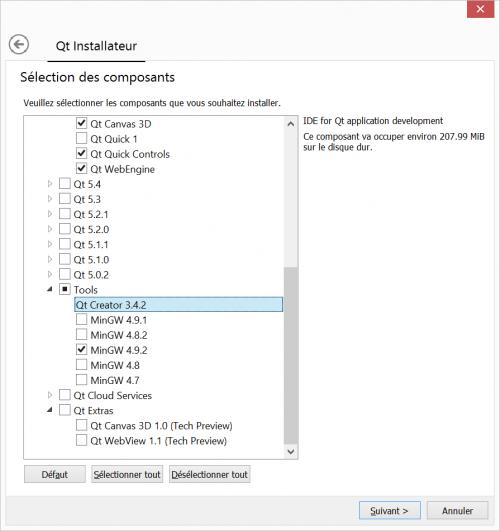
Visual Studio est l'outil de développement conçu par Microsoft (“MSVC” signifie “Microsoft Visual C++”). C'est donc naturellement l'outil conseillé pour compiler sur Windows. De nombreuses bibliothèques logicielles ne sont disponibles sur Windows uniquement pour Visual Studio. C'est en particulier le cas du module WebEngine de Qt, qui permet d'afficher des pages internet en utilisant le moteur de Chromium.

Les versions payantes sont relativement chères pour un particulier. Si votre entreprise possède des licences, utilisez-les. Sinon, la version Community de 2017 (<https://visualstudio.microsoft.com/fr/downloads/>) est suffisante.

* **Mingw32 (GCC)**

A l'origine, le compilateur GCC a été conçu pour Linux. Il existe des versions pour Windows (par exemple <http://www.equation.com/servlet/equation.cmd?fa=fortran>), mais on préfère souvent utiliser un compilateur dérivé de GCC pour Windows : MingW.

Le plus simple pour installer MingW sur Windows est de sélectionner cet outil dans l'installateur de Qt.



* **Pour Linux**

installation via les depots : clang et gcc

* **Pour Mac et iOS**

installation via XCode. Permet d'installer Clang et outils pour iOS

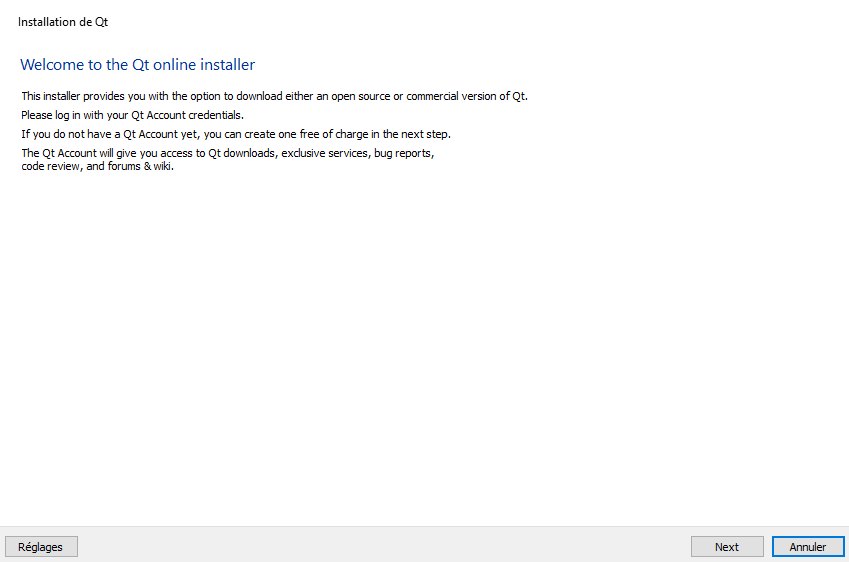
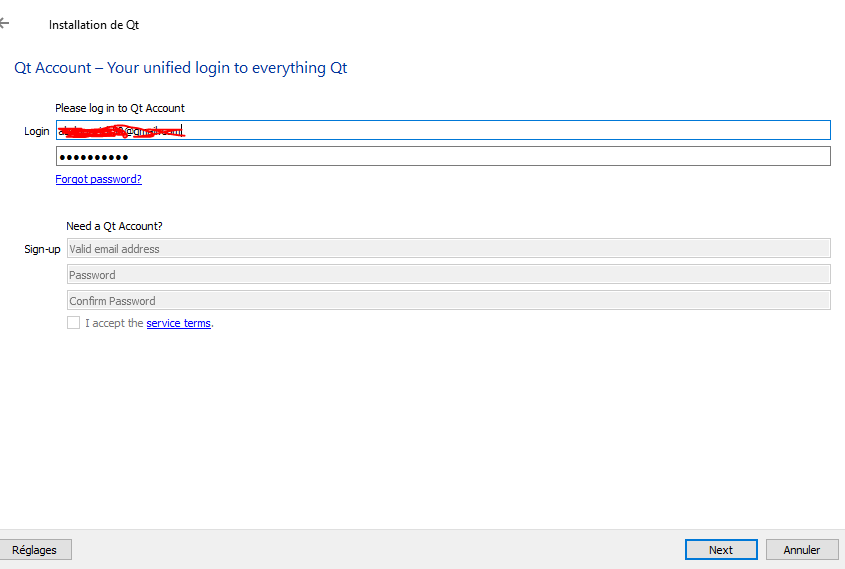
* **Pour Android**

Cross compile, sur Windows, linux et mac. Nécessite 2 outils :

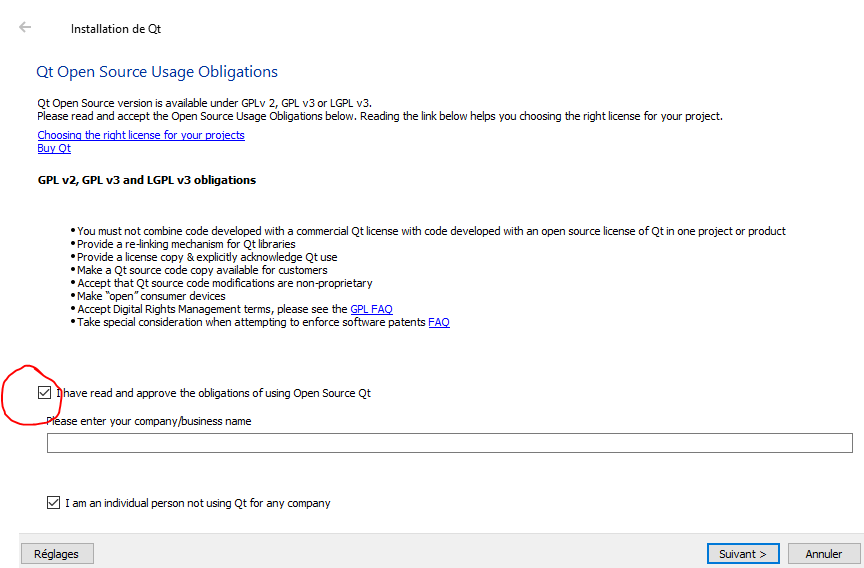
Android SDK pour programmer sur Android

Android NDK pour écrire des programmes C++

* **INSTALLATION DETAILLEE**

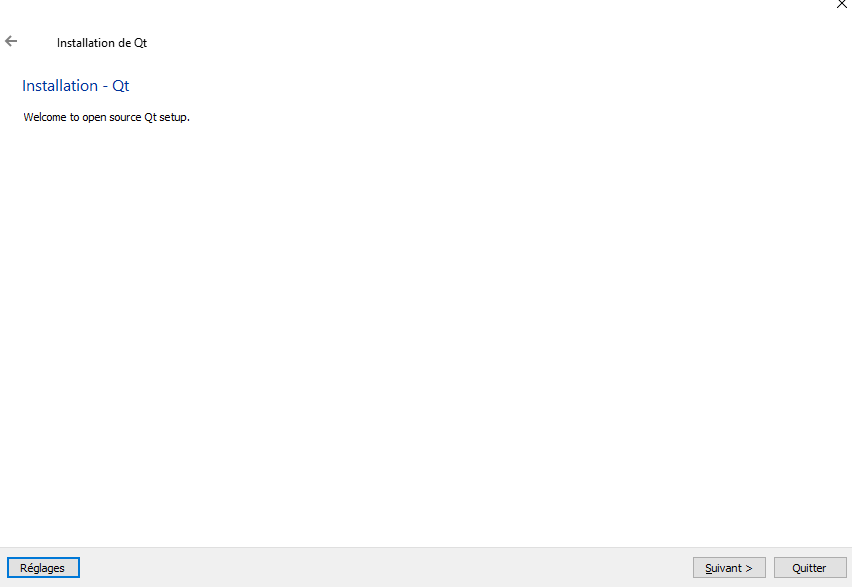
1. On clique sur le lien **Site official de Qt** (<https://www.qt.io/download>).
2. Sur la page d'accueil de Qt, je clique en haut à droite sur le bouton < **Download. Try. Buy.** >.
3. J'arrive sur la page "**Get Qt**" avec 2 colonnes, à gauche "**Commercial**" et à droite "**Open source**" et je choisi l'open source qui correspond à la version (L)GPL comme demandé dans le cours. Je cite: "**Qt est distribué sous deux licences, au choix : LGPL ou propriétaire. Celle qui nous intéresse est la licence LGPL car elle nous permet d'utiliser gratuitement Qt (et même d'avoir accès à son code source si on veut !)**". Donc, je clique en bas de la page à droite sur le bouton < **Go open source** >.
4. En bas de la page je clique en haut à droite sur le bouton  <**Download the Qt online installation.>**
5. Sur la page "Install Qt" je vois indiqué :
6. We detected your operating system as: **Windows**  
   Recommended download: **Qt Online Installer for Windows**
7. Une fois le téléchargement de l’exécutable terminer, je double clique là-dessus
8. **** Là, je clique sur “**Suivant**”.
9. ****

Là, je renseigne mon compte clique ou je créé une si je n’en dispose pas puis je clique sur “**Suivant**”.

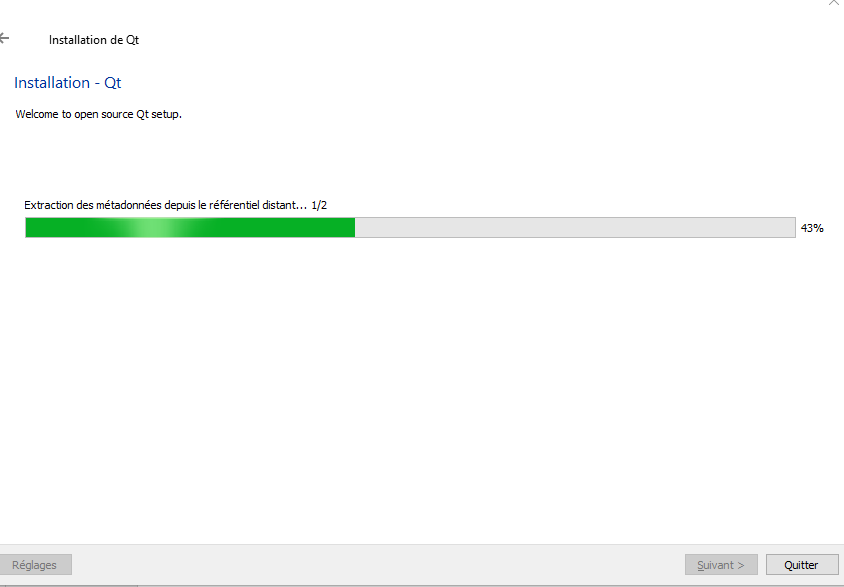
1. Comme je n'ai pas lieu de changer je clique sur "**Suivant**".
2. 

J’accepte les termes du contrat puis je clique sur “**Suivant**”

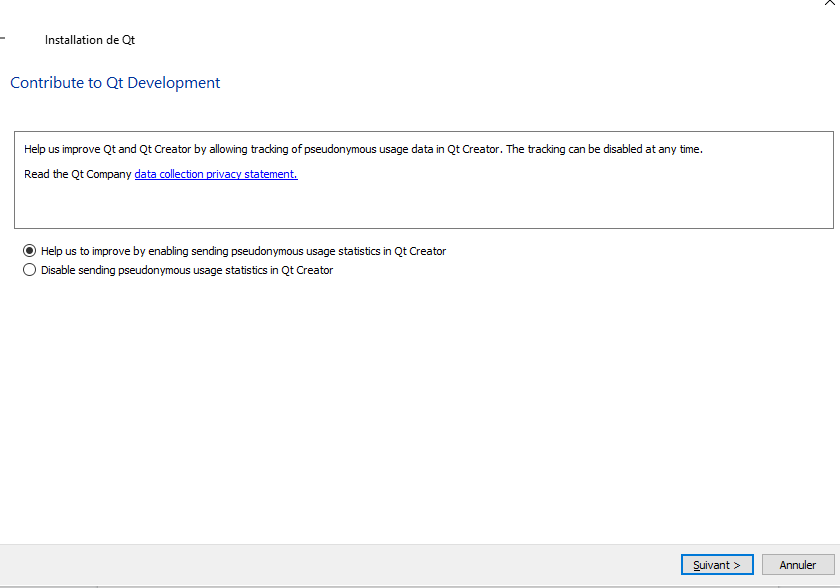
 Je laisse ce qui est coché car j'ai voulu faire d'autres essais mais en général il réclame trop de mémoire... au "**Suivant**".

**12)** ****

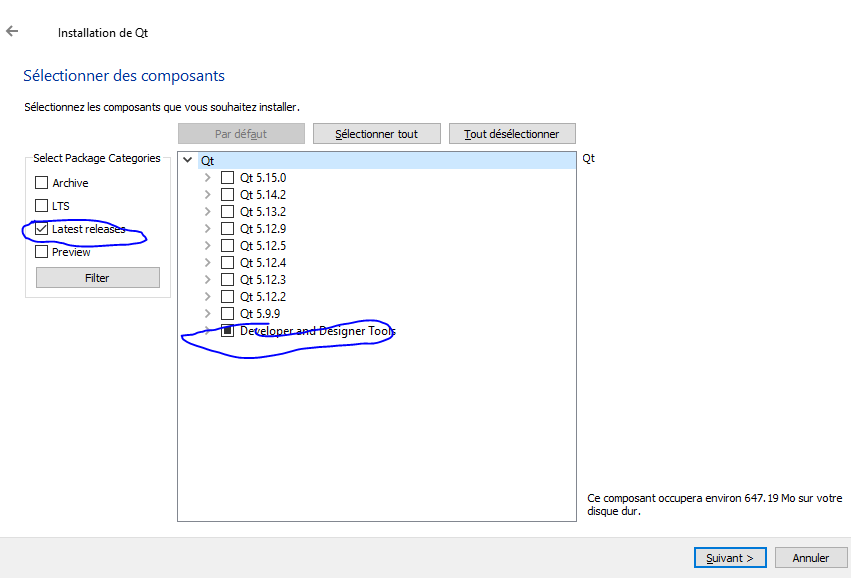
Sur cette page je ne vois pas ce que je peux faire alors au "**Suivant**".

1. 

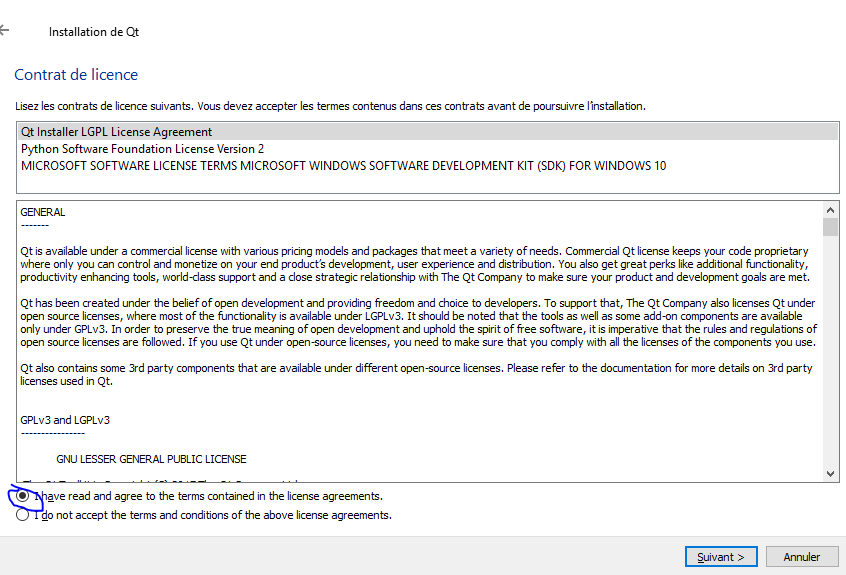
Puis sur "**Suivant**".

**13)** ****

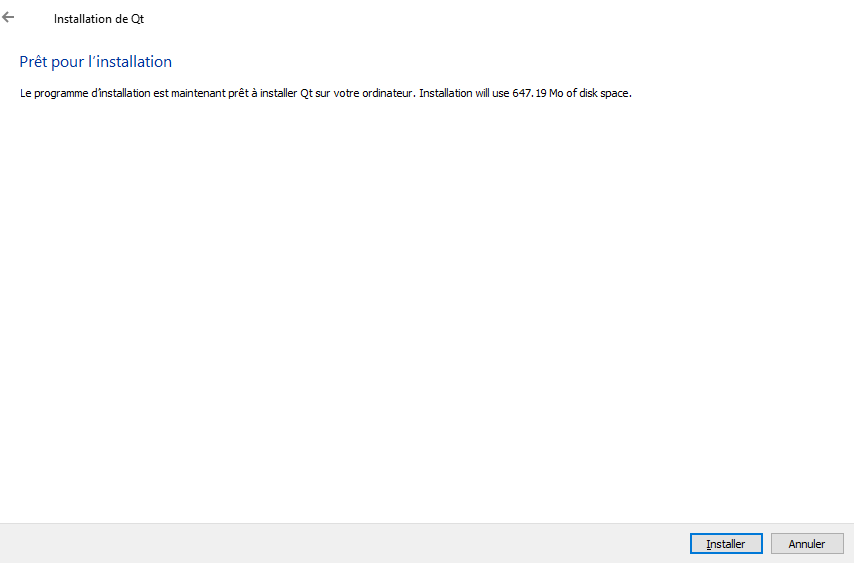
Choix pour contribuer au developpement du logiciel, apres je clique sur "**Suivant**".

**14)** ****

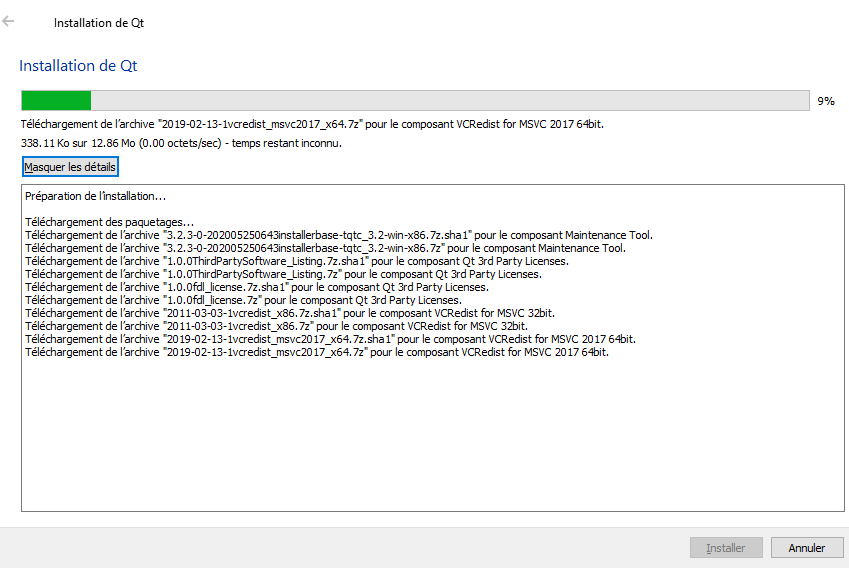
Choix des composants à installer puis je clique sur "**Suivant**".

**15)** ****

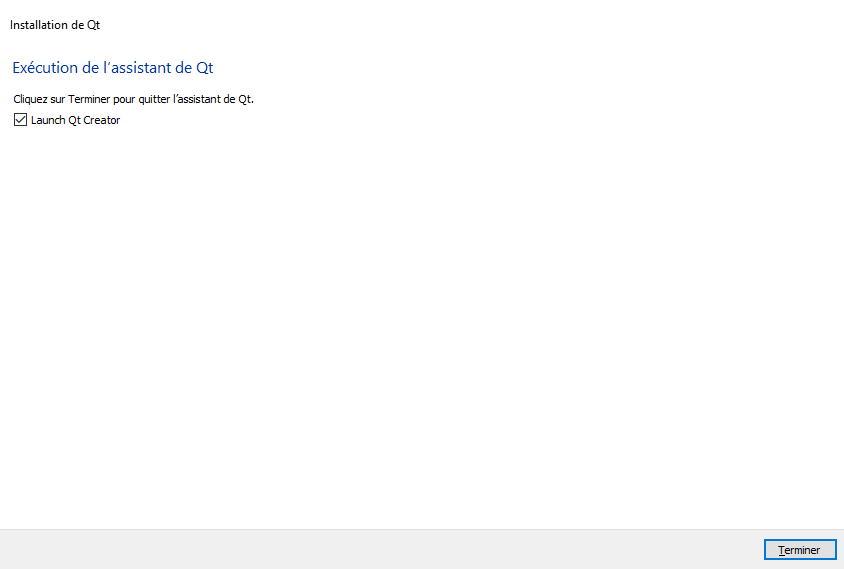
J’accepte les termes du licence pour les composants choisis puis je clique sur “**Suivant**”

**16)** ****

Et à la page suivante je clique sur "**Installer**".



Installation en cours

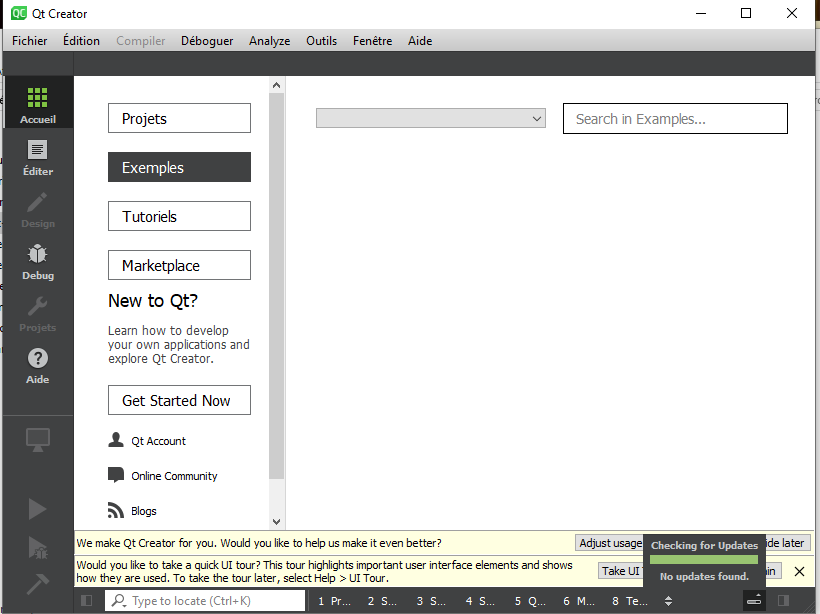
**17)** ****

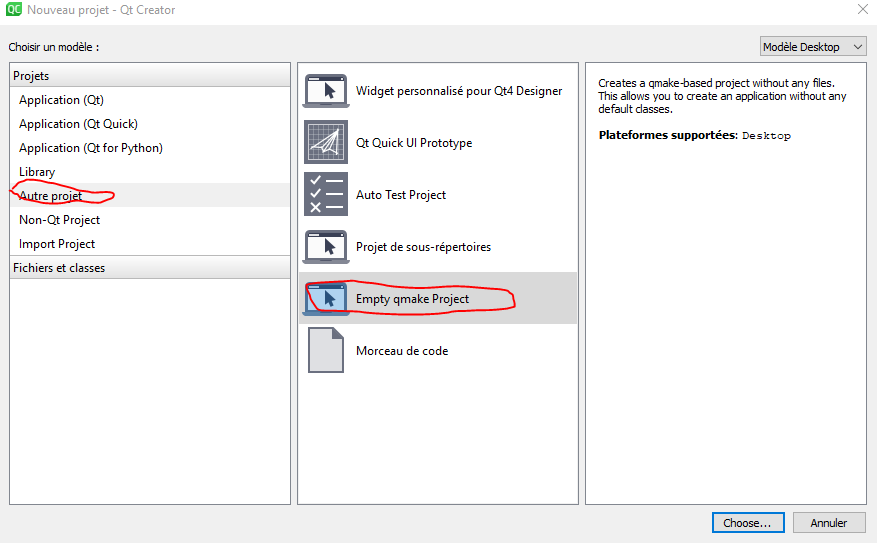
Enfin, lorsqu'il en a terminé de l'installation, je clique sur "**Terminer**".

**Utilisations de Qt**

* **Création d’un projet vide sur Qt Creator**

Fenetre de lancement du logiciel



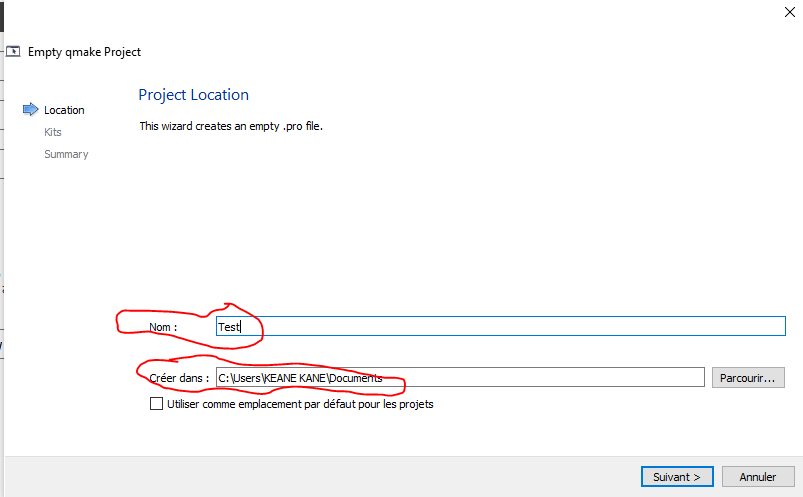
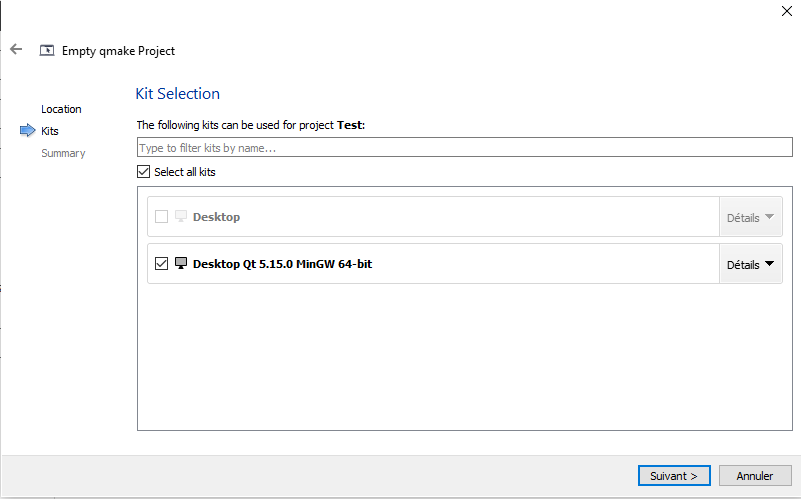
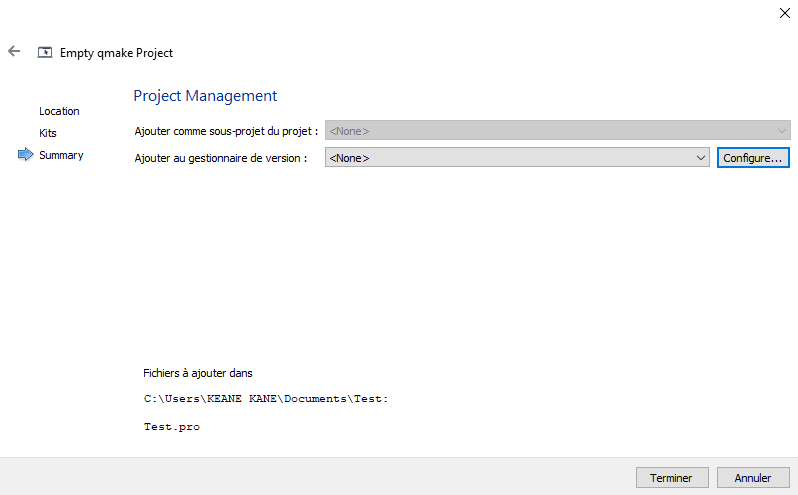
1. Fichier > Nouveau fichier ou projet (ou ctrl+N) 

J

J

J

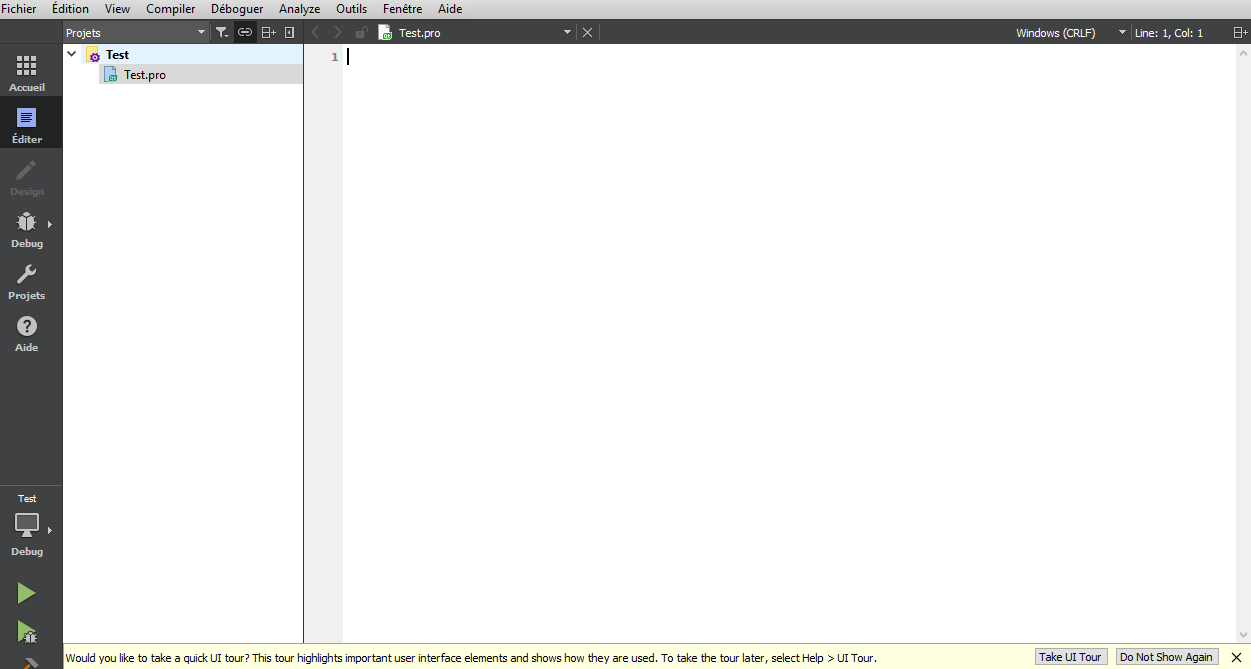
Je clique sur "**Choose**".

1. Je donne le nom du projet et son emplacement
2. Detection des **Kits (**le Qt et le compilateur installer**)**
3. On peut associer le projet à un gestionnaire de version (comme SVN, Git). C'est un outil très utile, notamment si on travaille à plusieurs sur un code source.  
   Pour le moment, j’en dispose pas alors je clique simplement sur « Terminer».

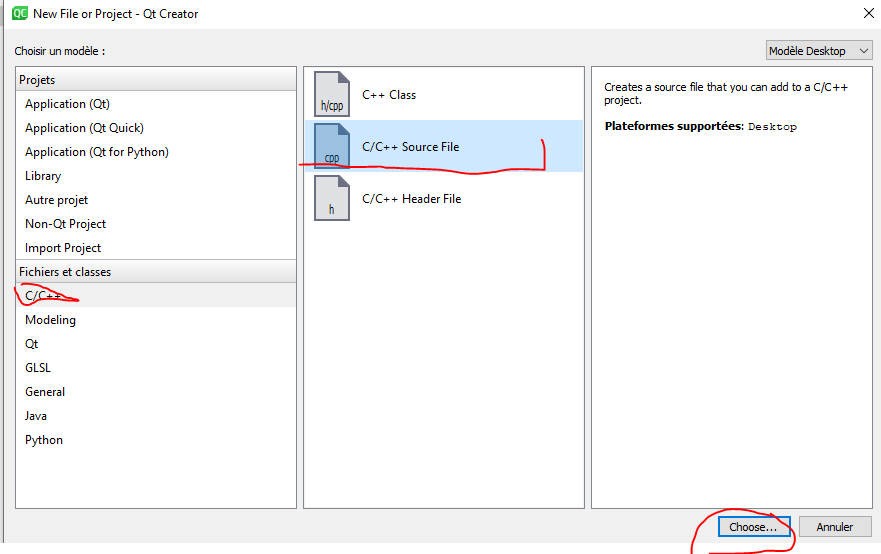
* **Compiler mon premier fenetre**

1. Notre projet vide est créé :

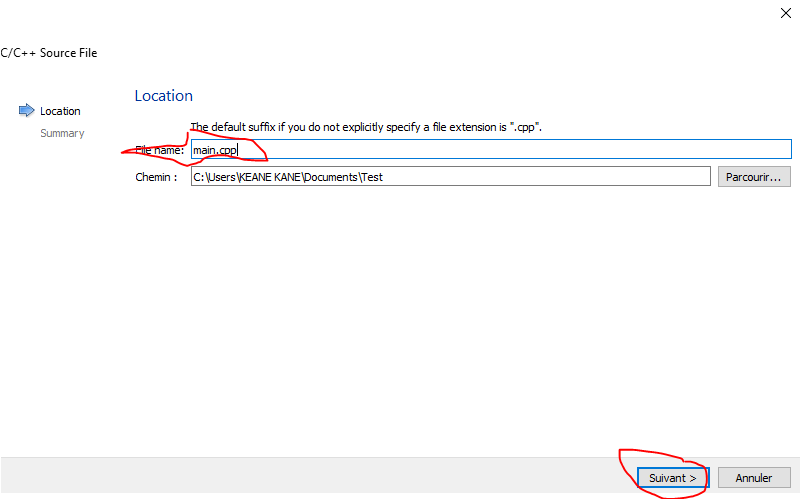
Le projet est constitué seulement d'un fichier.pro. Ce fichier, propre à Qt, sert à configurer le projet au moment de la compilation.

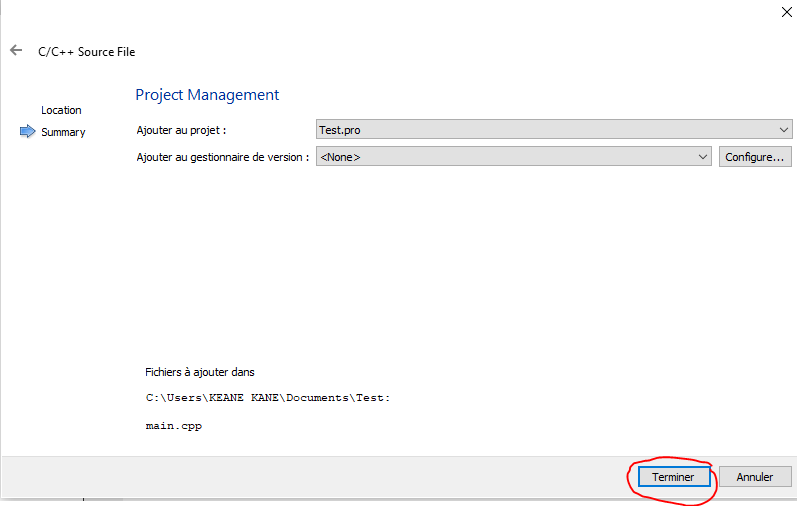


1. Je créer le fichier ‘**main.cpp’**

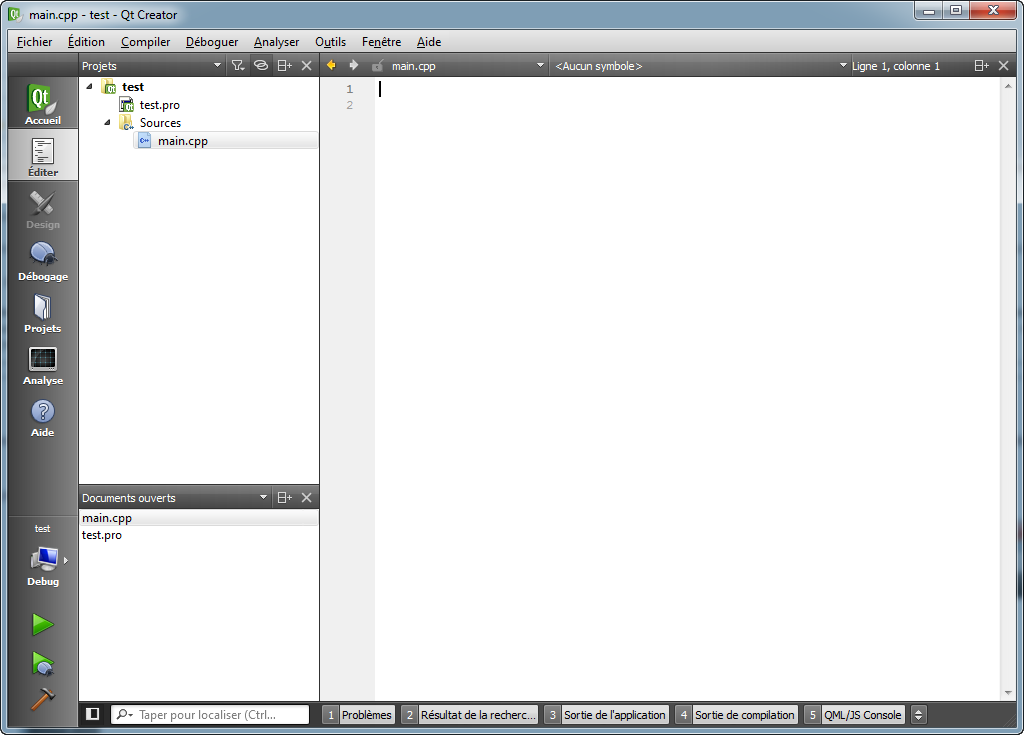
Fichier > Nouveau fichier ou projet et je sélectionne C++ > Fichier source C++

Je saisis le nom du fichier ici « **main.cpp** »



Idem, Pour le moment, j’en dispose pas alors je clique simplement sur «Terminer » 

J’ai cette fenétre qui s’affiche, de la je peux taper mon code



**Voici le code minimal pour créer une fenetre sous Qt**

#include <QApplication>

int main(int argc, char \*argv[])

{

QApplication app(argc, argv);

return app.exec();

}

**Sources**

<https://www.qt.io/download>

<https://doc.qt.io/>

<https://fr.wikipedia.org/wiki/Qt>

<https://openclassrooms.com/forum/sujet/a-lire-absolument-pour-installer-qt>

<https://guillaumebelz.github.io/qtinstall/>

<https://openclassrooms.com/fr/courses/1894236-programmez-avec-le-langage-c/1899081-compilez-votre-premiere-fenetre-qt>