

Tuto Qt



SOMMAIRE

1 Qu'est ce QT ?.....	3
2 Prérequis.....	3
3 Instalation.....	3
4 Préparation du projet.....	6
5 Choix du compilateur.....	11
6 Inclure des Dll + Librairies avec Qt.....	13
6.1 Inclure une .dll et une librairie compiler avec MinGW de type .dll + .a	13
6.2 Inclure une .dll et une librairie compiler avec MICROSOFT Visual Studio C/C++ de type .dll + .lib	16
7 Créer un bouton.....	17
8 Afficher un résultat.....	23
9 Fenêtre de saisie.....	27
10 Fermer la fenêtre.....	29
11 Portabilité.....	31

1 Qu'est ce QT ?

QT est un Framework qui permet de développer des bibliothèques ou des interfaces graphiques. L'avantage est que le logiciel est gratuit (en licence LGPL) et multiplateforme, avec des bibliothèques multiplateformes. A la base il a été développé par NOKIA pour les environnements PC et mobile.

2 licences de QT sont disponibles :

- 1 une licence propriétaire : <http://qt.digia.com/>
- 2 une licence LGPL : <http://qt-project.org/>

Qt est utilisée par : Adobe, Archos, Boeing, Google, Skype, la NASA...

2 Prérequis

Installer MICROSOFT Visual Studio C++ 2010 express Gratuit.

Ce tuto est réalisé avec les coupleurs sans-contact de la société ODALID <http://odalid.com> et les librairies pouvant communiquer avec les coupleurs.

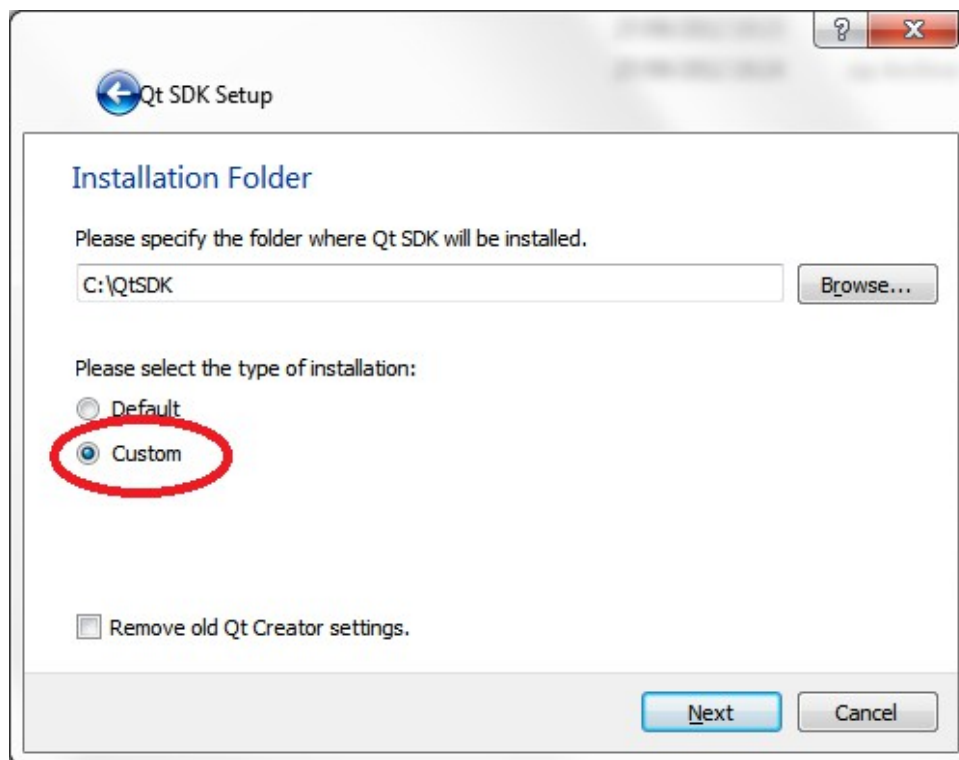
3 Instalation

Nous installerons le QtSDK, téléchargeable ici : http://www.developer.nokia.com/info/sw.nokia.com/id/da8df288-e615-443d-be5c-00c8a72435f8/Qt_SDK.html

Une fois téléchargé lancer : QtSdk-offline-win-x86-v1_2_1.exe
La fenêtre d'installation se lance.

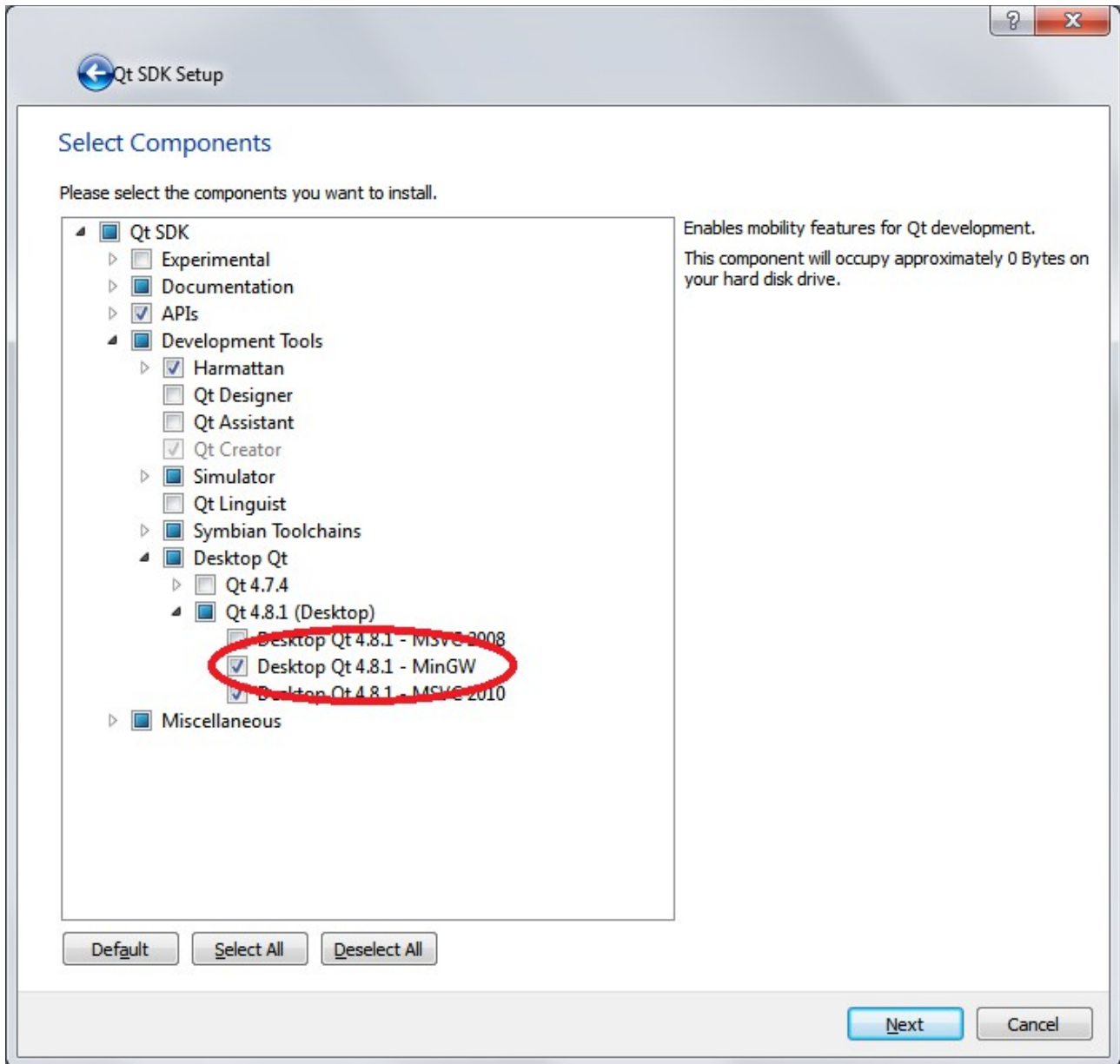
Puis arrive à cette fenêtre, choisir l'option **Custom** :

Tuto Qt		Page 3 on 31	
Lieu: DIJON / ODALID		Date : 18/12/2012	
Ref : 1.0		Ecrit par: Vincent THIVENT	



Cliquez sur Next :

Dans cette fenêtre nous choisirons d'utiliser le compilateur MinGW GNU, compilateur multiplateforme.



Nous garderons aussi le compilateur MSVC (Microsoft Visual C++) qui nous permettra de faire des libraires compatibles avec Microsoft Visual C++.

Puis déroulez le processus normal d'installation.

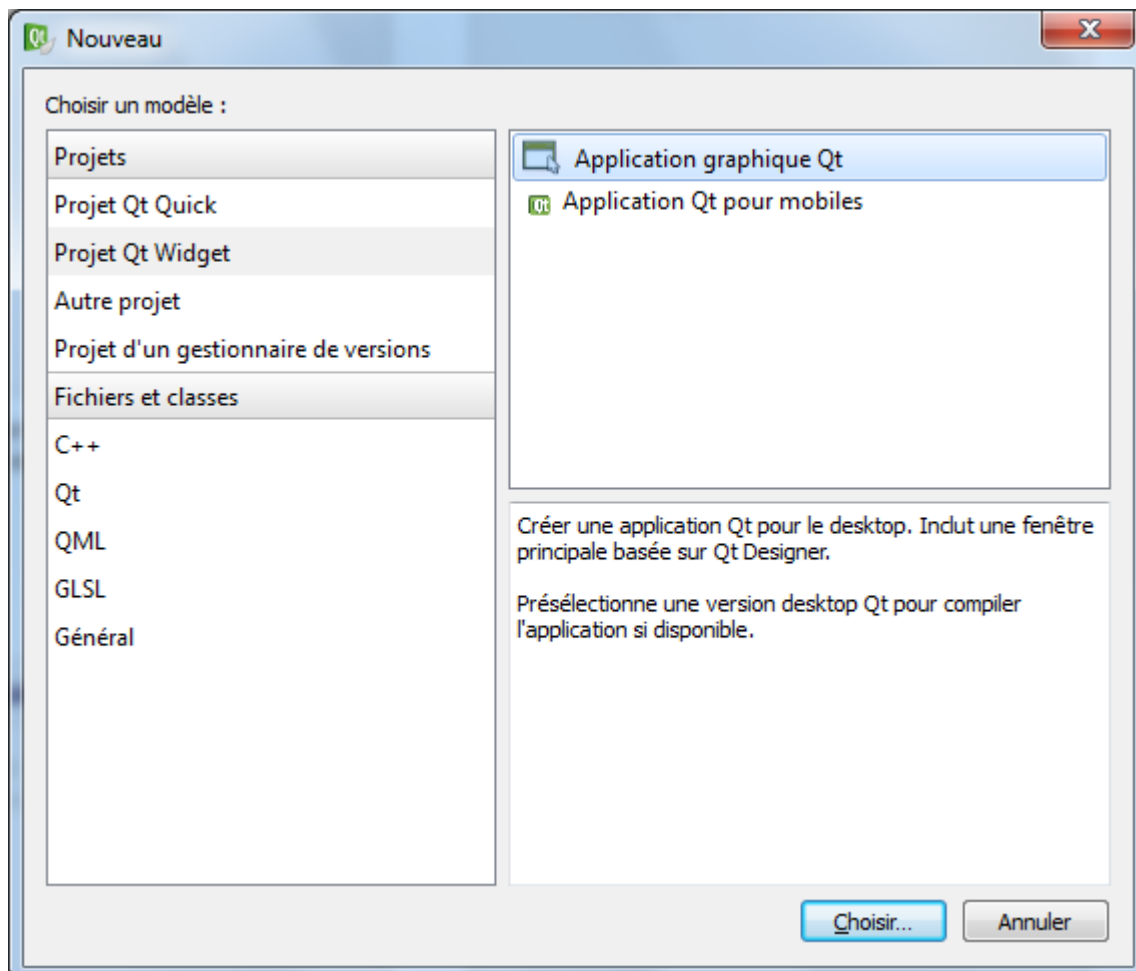
4 Préparation du projet

Pour créer un nouveau projet vous devez, lancer l'application Qtcreator

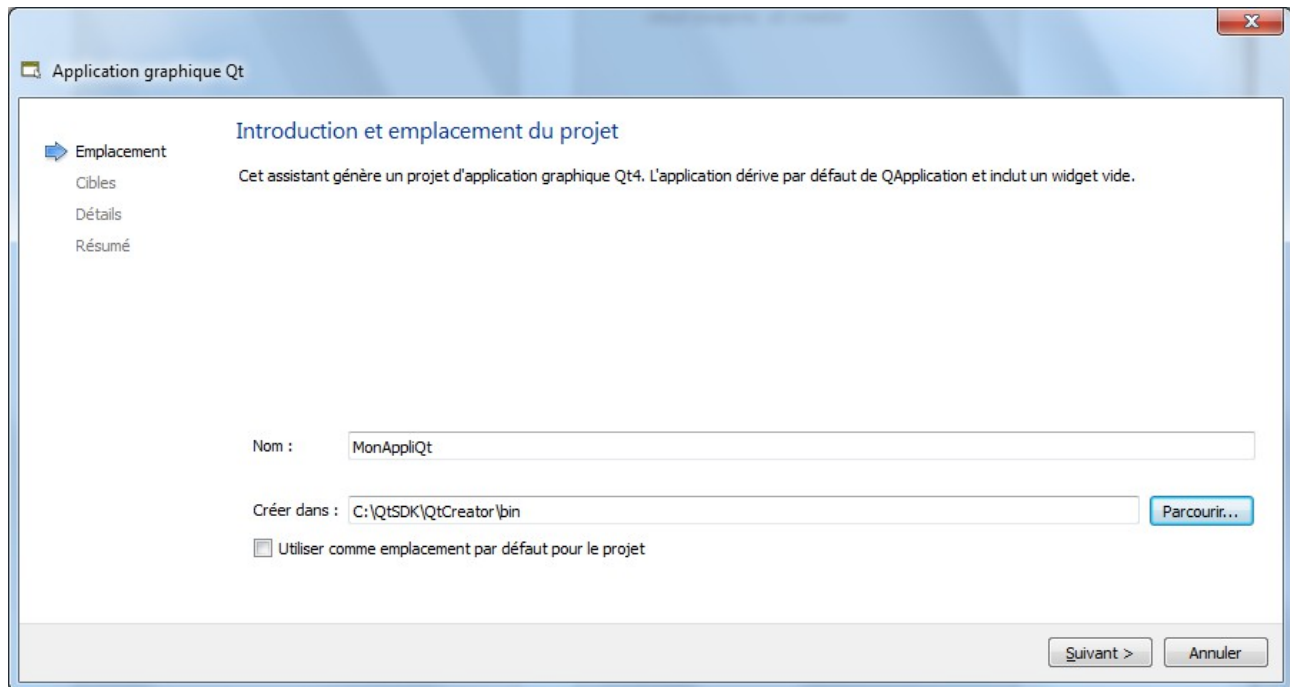


Puis cliquez sur fichier → Nouveau fichier ou projet...

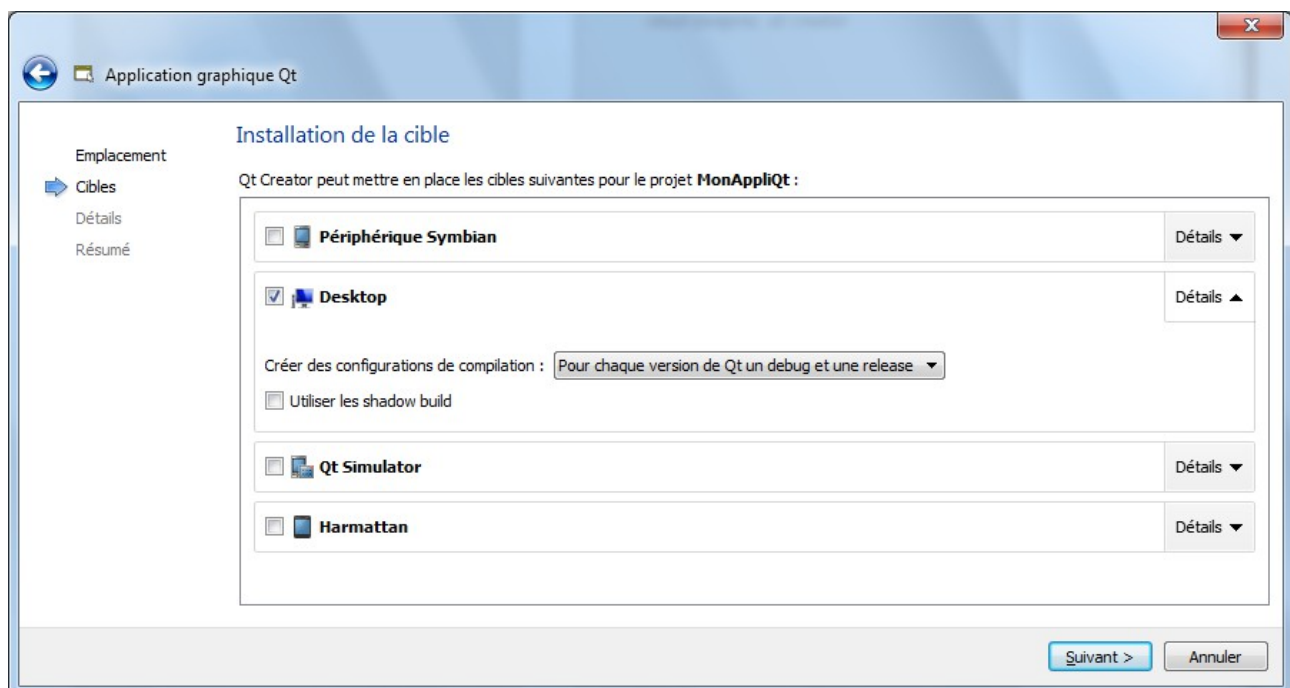
La fenêtre suivante s'ouvre :



Ensuite sélectionner Projet Qt Widget → Application graphique Qt

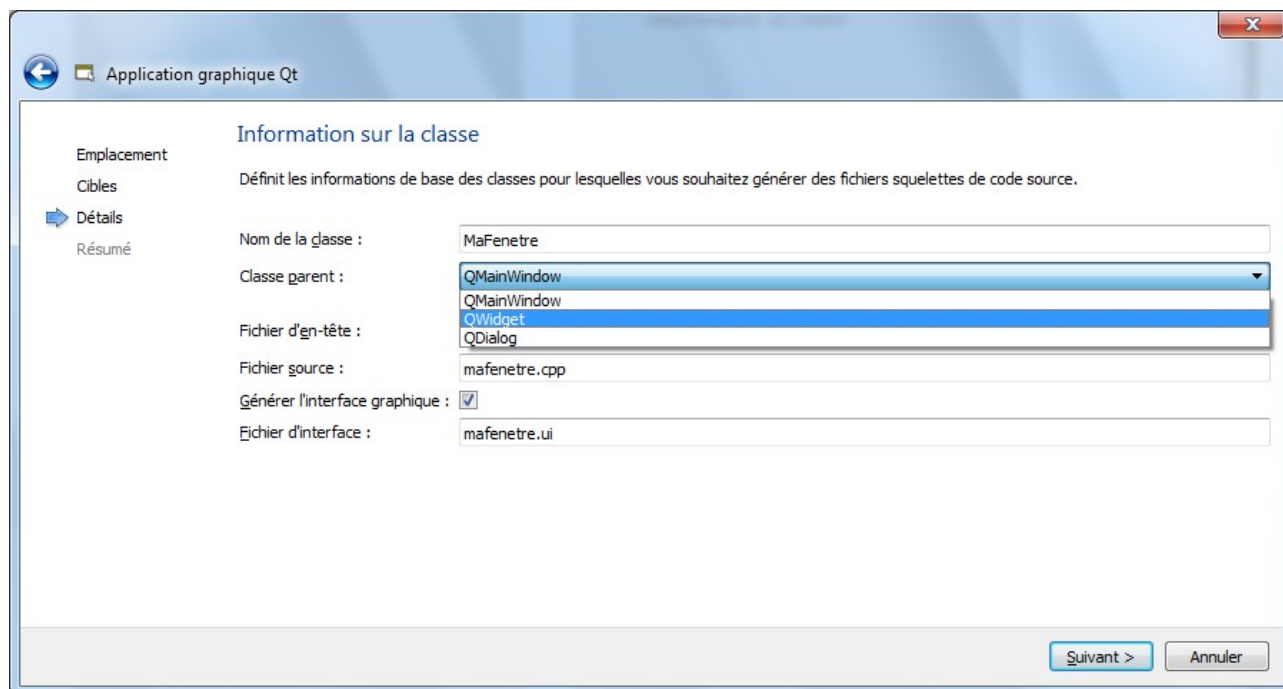


Entrez le nom de votre application puis sélectionnez le répertoire de travail.



Cette fenêtre vous permet de sélectionner le type de plateforme pour l'exécution de votre programme. Laissez les paramètres par défaut (Desktop → application pour ordinateur), puis cliquez sur suivant.

Cette fenêtre apparaît :



Application graphique Qt

Emplacement
Cibles
Détails
Résumé

Information sur la classe

Définit les informations de base des classes pour lesquelles vous souhaitez générer des fichiers squelettes de code source.

Nom de la classe : MaFenetre

Classe parent : QMainWindow
QMainWindow
QWidget
QDialog

Fichier d'en-tête : mafenetre.h

Fichier source : mafenetre.cpp

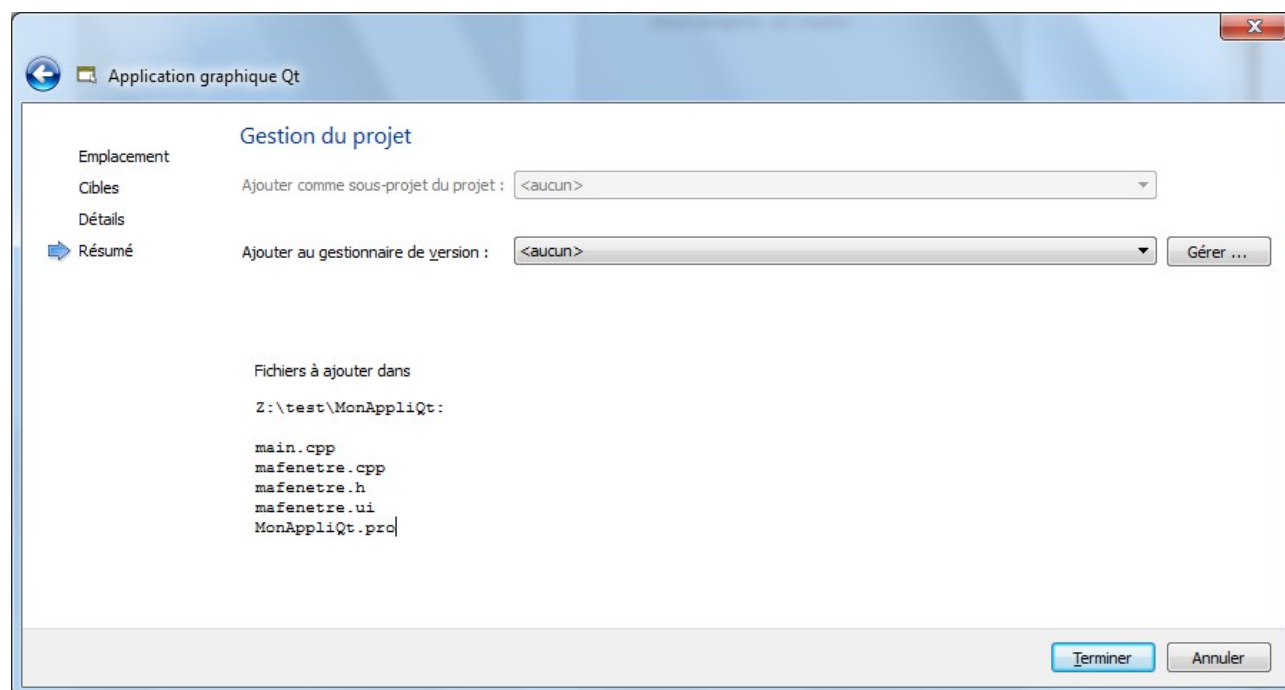
Générer l'interface graphique : ☒

Fichier d'interface : mafenetre.ui

Suivant > Annuler

Donner un nom à votre classe, ici on prendra « MaFenetre ». Puis sélectionner QWidget. Et cliquer sur suivant

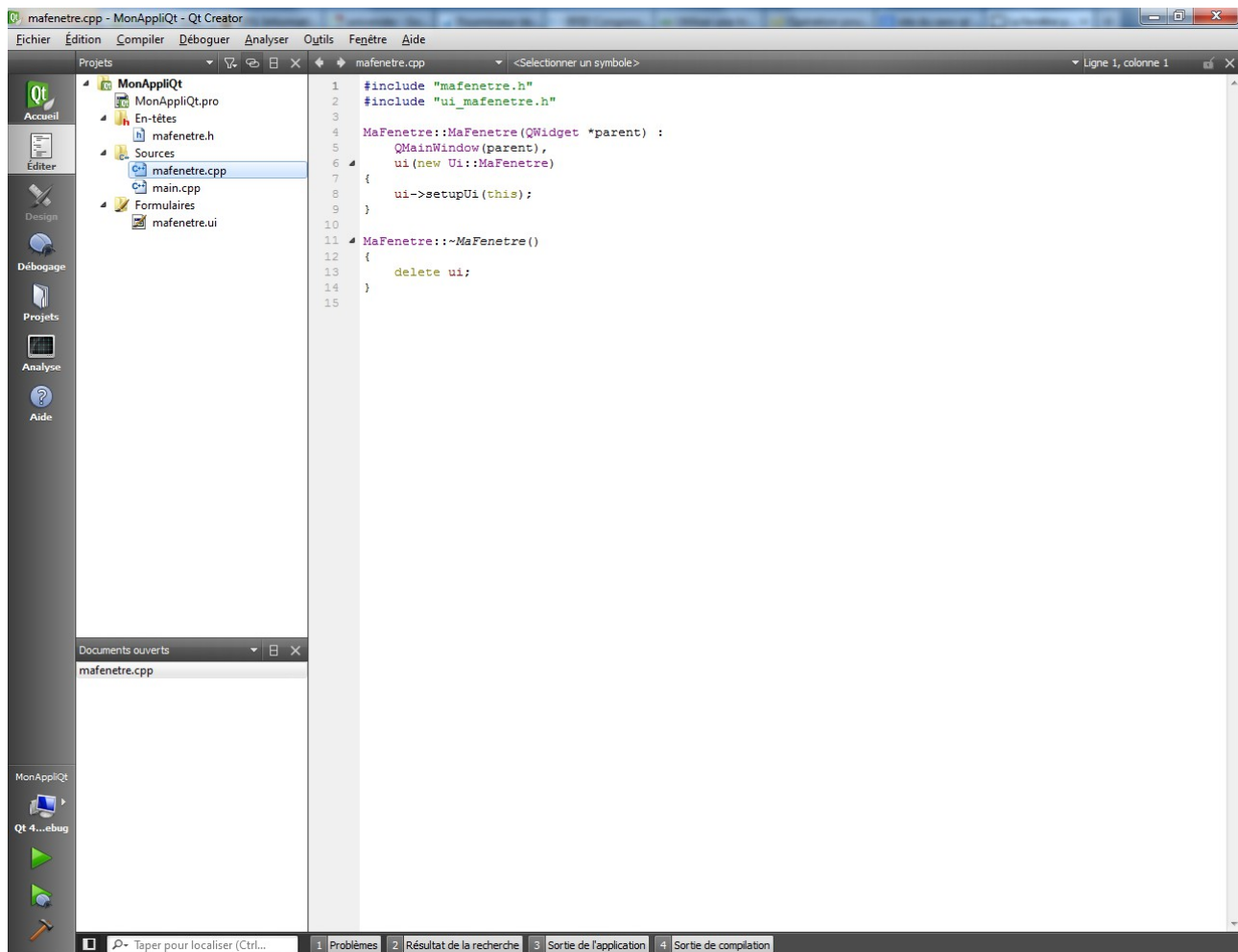
Cette fenêtre apparaît :



Cliquer sur terminer

Votre projet est prêt.

Il doit ressembler à cela :



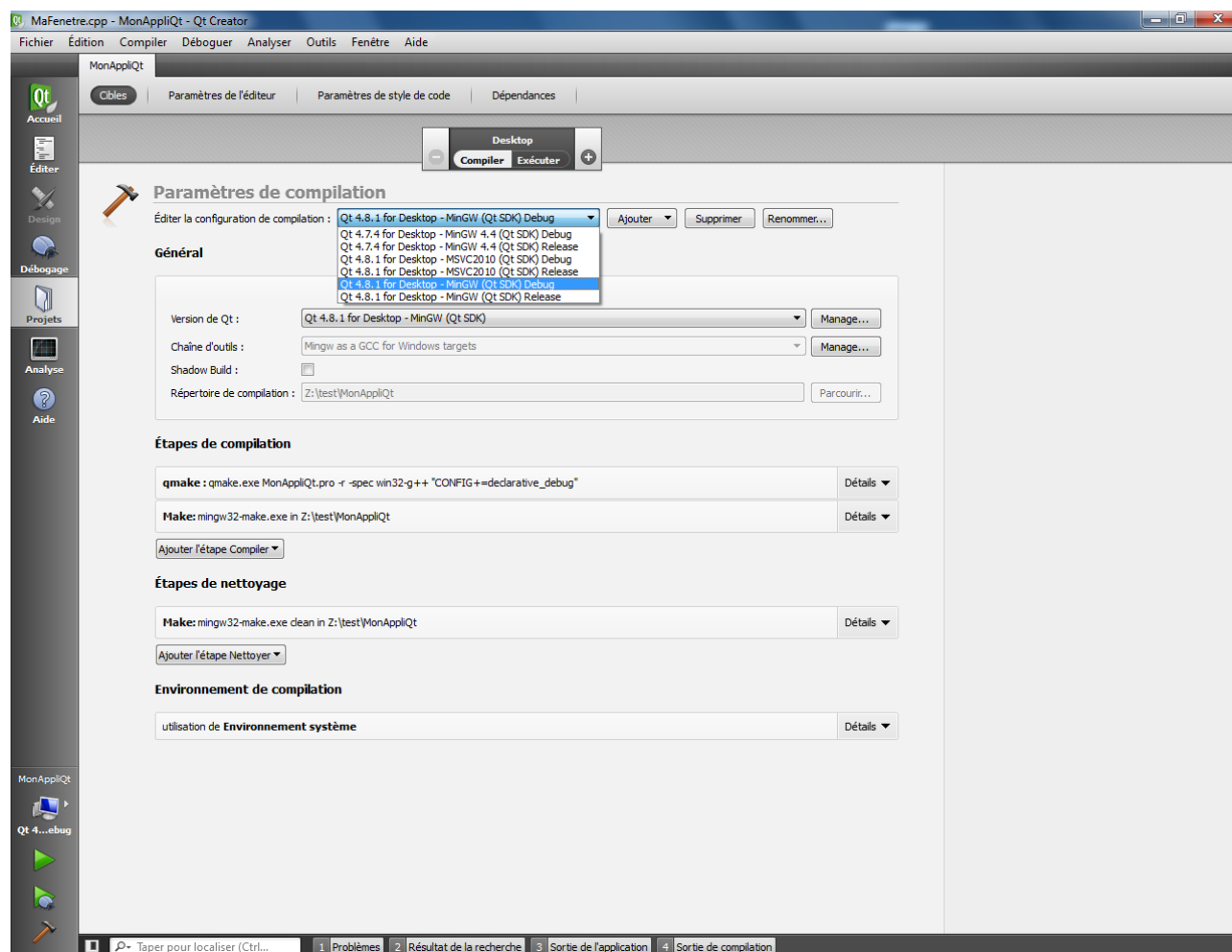
5 Choix du compilateur

Qt peut être utilisé avec différents compilateurs :

- MinGW (4.7.4 et 4.8.1) : Compilateur GNU multi-plateforme ;
- MSVC2010 : Compilateur de MICROSOFT Visual Studio 2010;

Dans la barre de droite cliquez sur l'icône Projets :

Cette fenêtre apparaît :



Dans la fenêtre « editier la configuration de compilation » vous pouvez choisir entre les différents compilateurs en mode Debug ou Release.

Pour notre projet nous choisirons MinGW 4.8.1.

Dans les projets de développement C/C++ vous serez amené à inclure des DLL et des librairies propres aux matériels. Nous allons voir comment les inclure dans le projet afin de pouvoir utiliser les fonctions des librairies.

6 Inclure des Dll + Librairies avec Qt

Vous devez avoir 3 fichiers

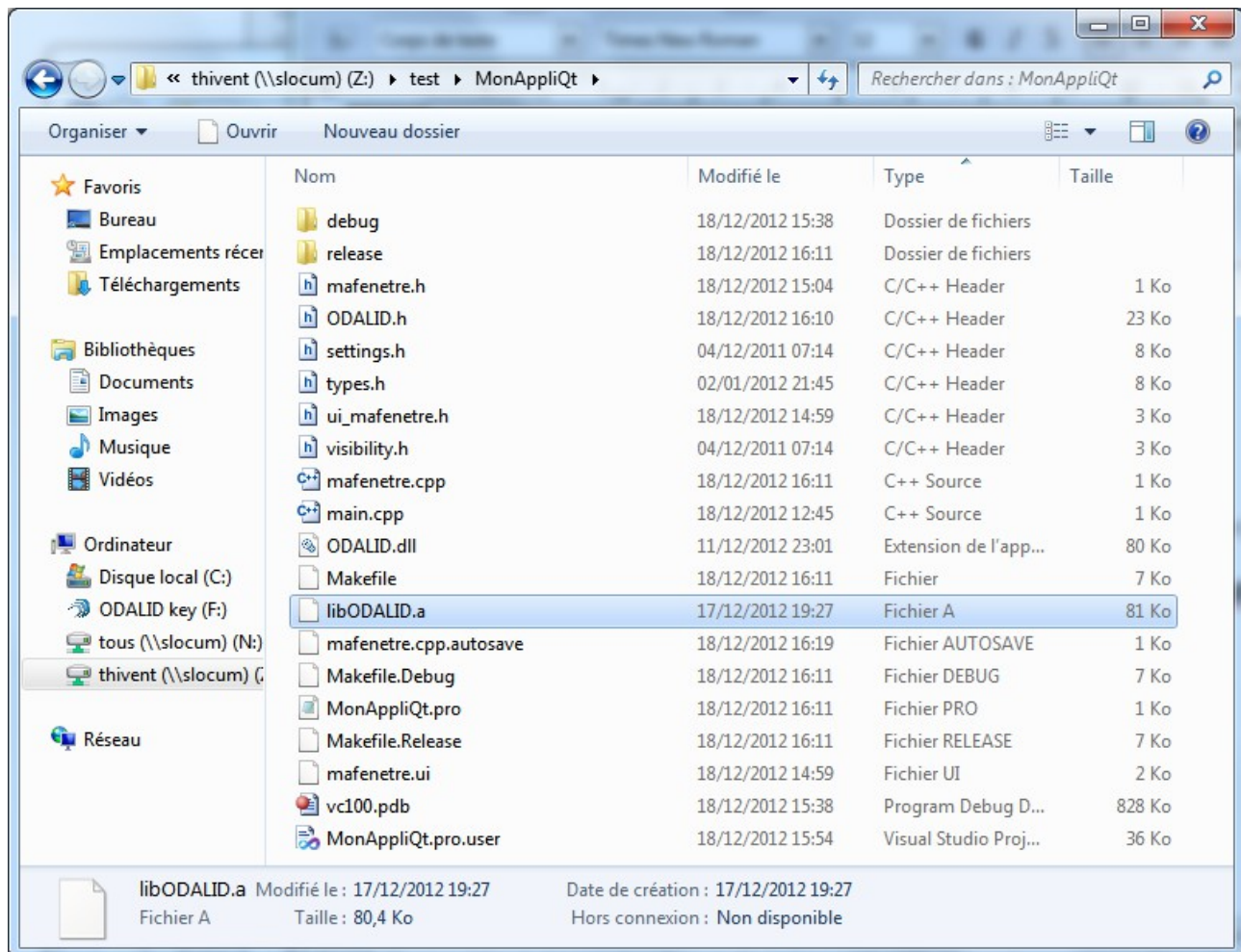
- .h exemple ODALID.h ;
- .dll exemple ODALID.dll ;
- .a ou .lib exemple libODALID.a ou ODALID.lib.

Ou uniquement le .dll et le .h mais cela ne sera pas présenter dans ce tuto.

6.1 Inclure une .dll et une librairie compiler avec MinGW de type .dll + .a

Copiez ces 3 fichiers au même niveau que votre projet :

- ODALID.h
- ODALID.dll
- libODALID.a



Puis il suffit d'ajouter la ligne suivante dans le fichier « MonAppliQt.pro »

```
LIBS += -LZ:\\test\\MonAppliQt -lODALID
```

LIBS += -L{chemin de la librairie libODALID.a} -l{nome de la librairie libODALID.a sans « lib » et sans « .a »}

Votre fichier doit ressembler à ça:

```
QT += core gui
```

```
TARGET = MonAppliQt
```

```
TEMPLATE = app
```

```
SOURCES += main.cpp\
          mafenetre.cpp
```

```
HEADERS += mafenetre.h
```

```
FORMS += mafenetre.ui
```

```
LIBS += -LZ:\\test\\MonAppliQt -lODALID
```

Puis la ligne suivante dans le fichier « mafenetre.cpp »

```
#include "ODALID.h"
```

Votre fichier doit ressembler à ça:

```
#include "mafenetre.h"
#include "ui_mafenetre.h"
#include "ODALID.h"
```

```
MaFenetre::MaFenetre(QWidget *parent) :
    QMainWindow(parent),
    ui(new Ui::MaFenetre)
{
    ui->setupUi(this);
}
```

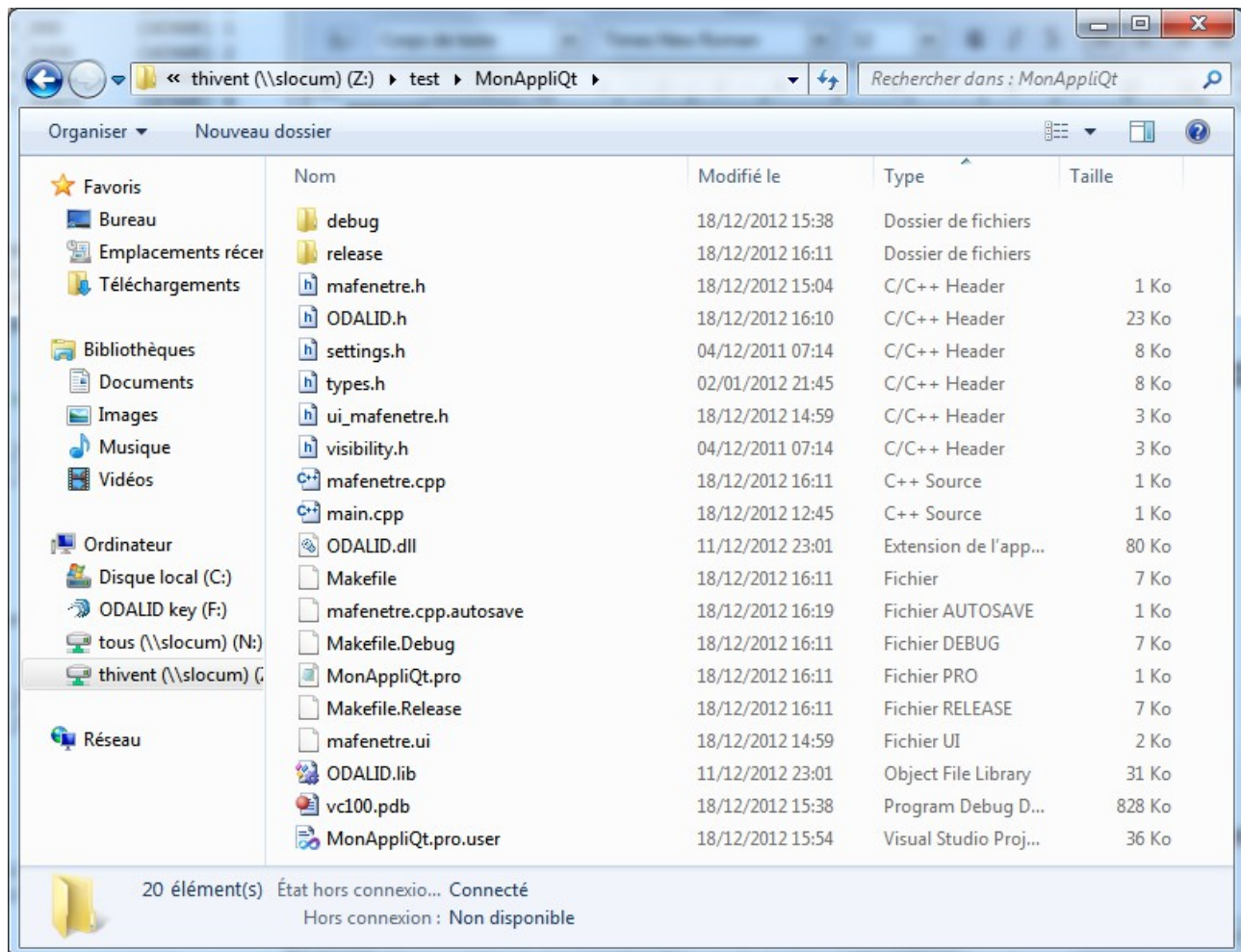
```
MaFenetre::~MaFenetre()
{
    delete ui;
}
```

Pour la compilation, utiliser le compilateur MinGW

6.2 Inclure une .dll et une librairie compiler avec MICROSOFT Visual Studio C/C++ de type .dll + .lib

Copiez ces 3 fichiers au même niveau que votre projet :

- ODALID.h
- ODALID.dll
- ODALID.lib



Puis il suffit d'ajouter les 2 lignes suivantes dans le fichier « mafenetre.cpp »

```
#pragma comment(lib, "ODALID.lib")
#include "ODALID.h"
```


Votre fichier doit ressembler à ça:

```
#pragma comment(lib, "ODALID.lib")
#include "mafenetre.h"
#include "ui_mafenetre.h"
#include "ODALID.h"

MaFenetre::MaFenetre(QWidget *parent) :
    QMainWindow(parent),
    ui(new Ui::MaFenetre)
{
    ui->setupUi(this);
}

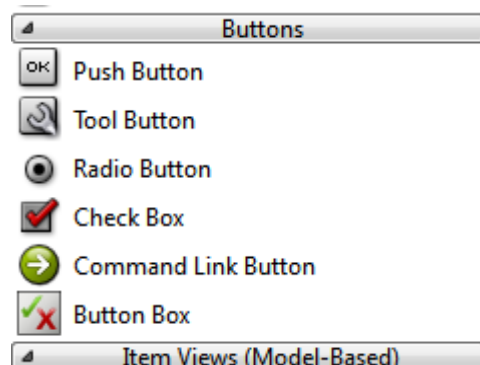
MaFenetre::~MaFenetre()
{
    delete ui;
}
```

Pour la compilation, utiliser le compilateur MICROSOFT Visual Studio C/C++

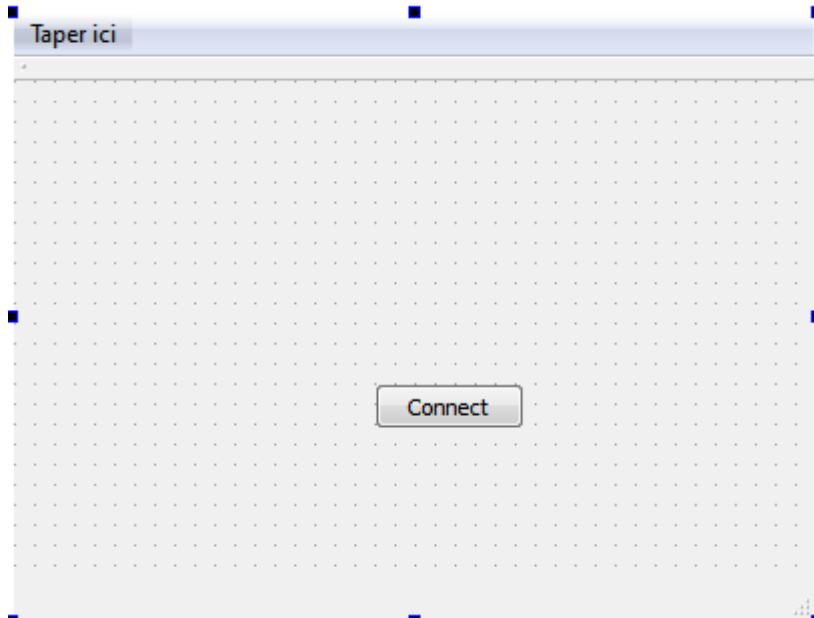
7 Créer un bouton

Pour créer un bouton cliquez sur le fichier « mafenetre.ui »

Ajoutez un bouton en cliquant sur l'icône « Push Button » du menu « Buttons »

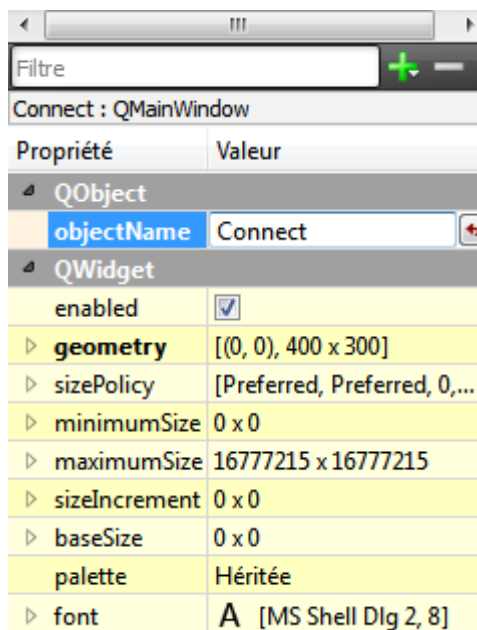


puis placer le bouton sur votre fenêtre comme ceci :



Vous pouvez changer le contenu du bouton en cliquant dessus

Dans la fenêtre en bas à droite, donnez lui un nom « Connect »



Dans le fichier « mafenetre.h » ajouter la ligne suivante

```
private slots:
    void on_Connect_clicked();
```

Votre fichier doit ressembler à ça:

```
#ifndef MAFENETRE_H
#define MAFENETRE_H

#include <QMainWindow>

namespace Ui {
class MaFenetre;
}

class MaFenetre : public QMainWindow
{
    Q_OBJECT

public:
    explicit MaFenetre(QWidget *parent = 0);
    ~MaFenetre();

private slots:
    void on_Connect_clicked();

private:
    Ui::MaFenetre *ui;
};

#endif // MAFENETRE_H
```

Dans le fichier « mafenetre.cpp » ajouter la ligne suivante

```
void MaFenetre::on_Connect_clicked()
{
}
```

Votre fichier doit ressembler à ça:

```
#include "mafenetre.h"
#include "ui_mafenetre.h"
#include "ODALID.h"
```

```
MaFenetre::MaFenetre(QWidget *parent) :
    QMainWindow(parent),
    ui(new Ui::MaFenetre)
{
    ui->setupUi(this);
}
```

```
MaFenetre::~MaFenetre()
{
    delete ui;
}
```

```
ReaderName MonLecteur;
```

```
char pszHost[] = "192.168.1.4";
```

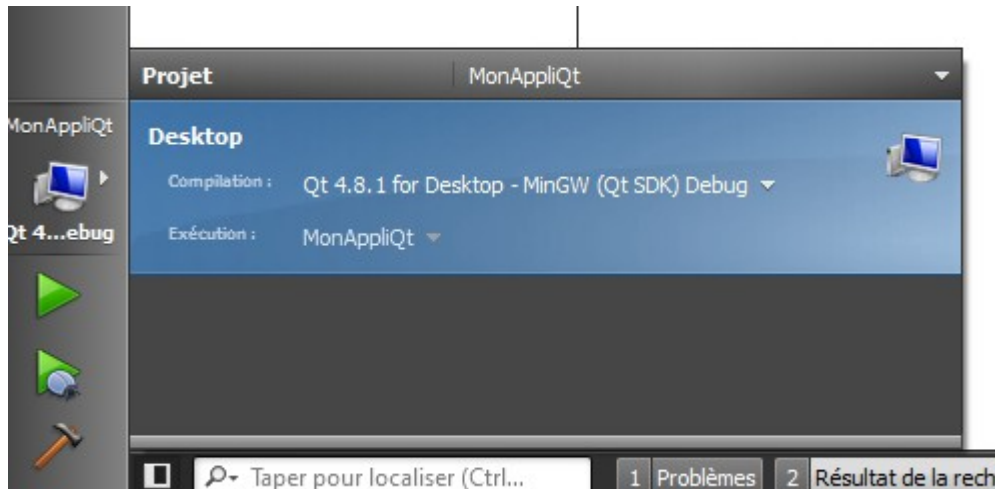
```
void MaFenetre::on_Connect_clicked()
{
}
```

Pour tester le programme, nous allons ajouter une fonction de la librairie.

```
#include <QtGui>
```

```
void MaFenetre::on_Connect_clicked()
{
    uint16_t status = 0;
    //MonLecteur.Type = ReaderTCP;
    //strcpy(MonLecteur.IPReader, pszHost);
    MonLecteur.Type = ReaderCDC;
    MonLecteur.device = 0;
    status = OpenCOM1(&MonLecteur);
    qDebug() << "OpenCOM1" << status;
}
```

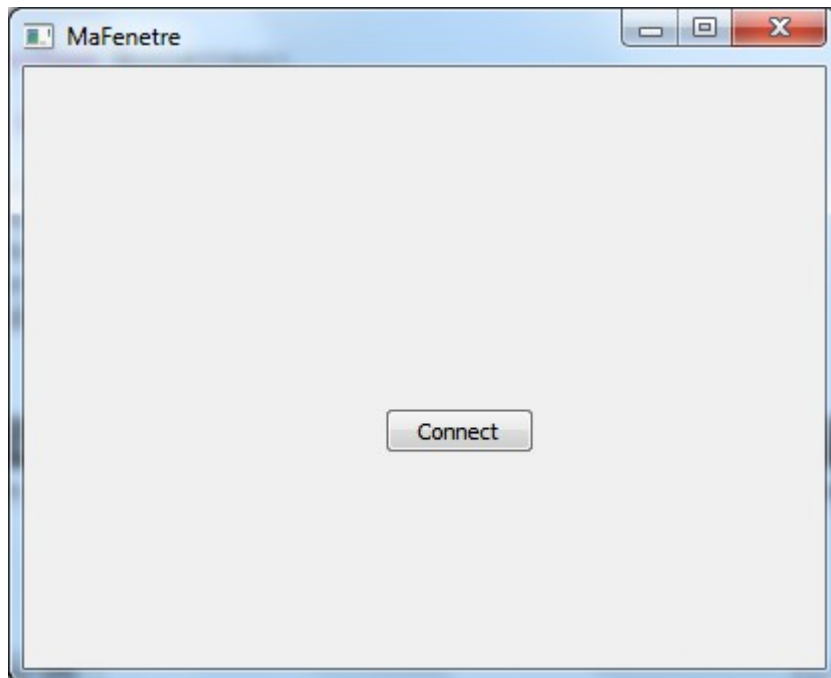
Cliquez sur l'icone en bas à gauche au dessus des flèches vertes puis choisissez le compilateur MinGW en mode Debug



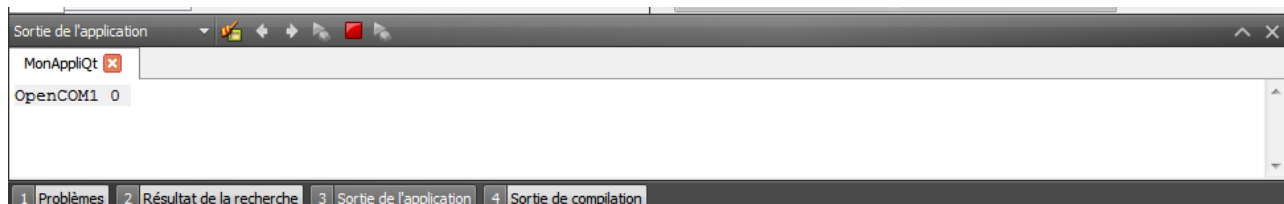
Puis cliquez sur la flèche verte de debugage :



La fenêtre suivante s'ouvre

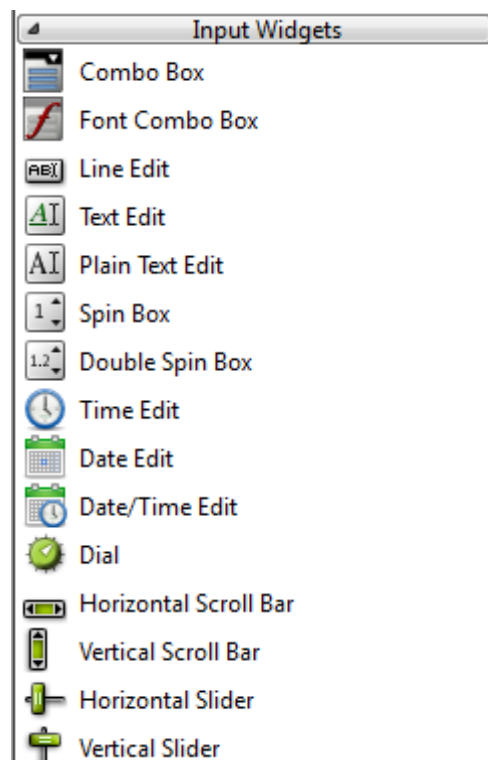


Connecter un lecteur sans contact à votre ordinateur, puis lorsque vous cliquez sur « Connect », le message suivant s'affiche dans la fenêtre « sortie d'application » de Qt

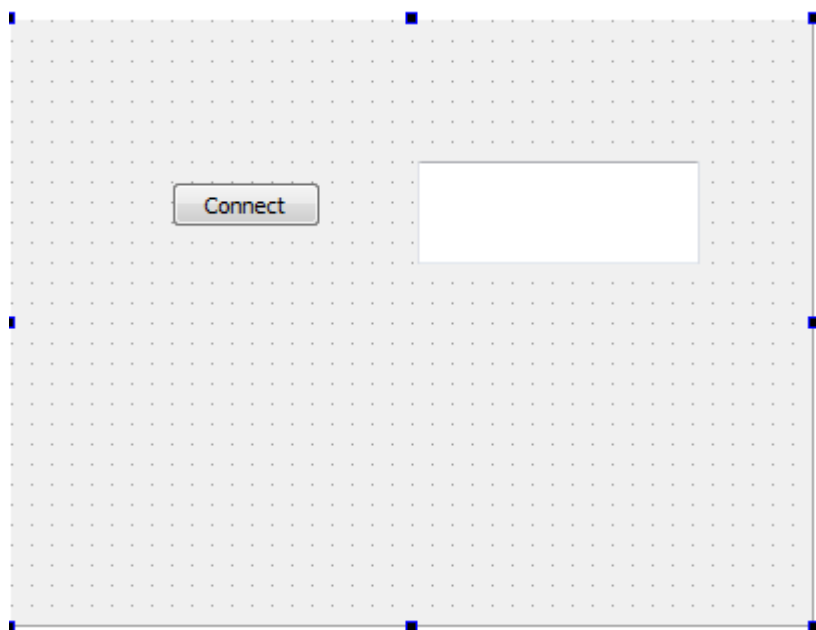


8 Afficher un résultat

Pour afficher un résultat, retournez dans l'interface de Design et cliquez sur « Text Edit » de la fenêtre « Inputs Widget » :



Puis placer le widget sur la fenêtre :



Donnez lui un nom comme « Affichage » :

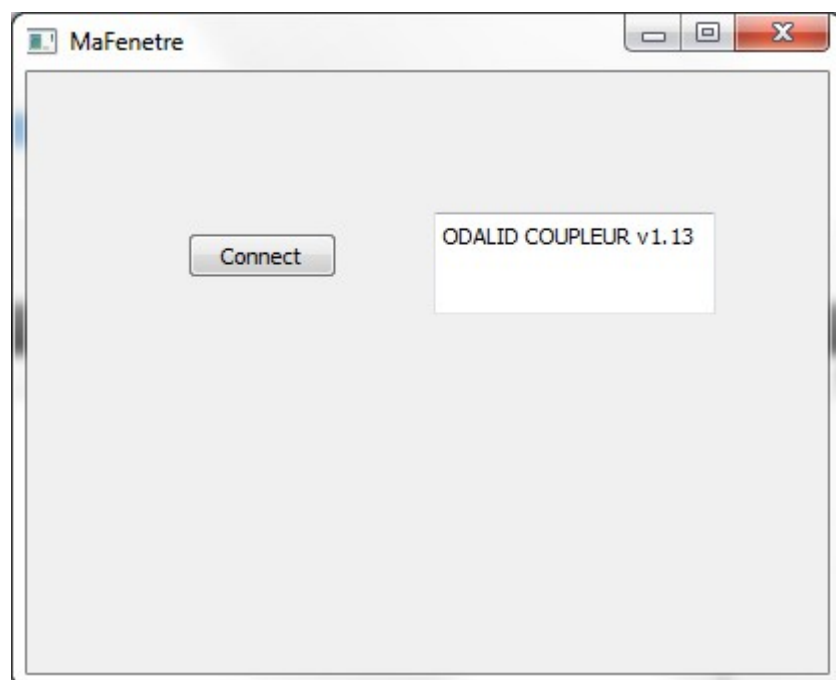
Affichage : QTextEdit	
Propriété	Valeur
QObject	
objectName	Affichage
QWidget	
enabled	<input checked="" type="checkbox"/>
geometry	[(203, 70), 141 x 51]
sizePolicy	[Expanding, Expanding,...]
minimumSize	0 x 0
maximumSize	16777215 x 16777215
sizeIncrement	0 x 0
baseSize	0 x 0
palette	Héritée
font	A [MS Shell Dlg 2, 8]
cursor	Flèche
mouseTrack...	<input type="checkbox"/>
focusPolicy	WheelFocus
contextMen...	DefaultContextMenu
acceptDrops	<input checked="" type="checkbox"/>
toolTip	
statusTip	
whatsThis	
accessibleN...	
accessibleD...	
layoutDirect...	LeftToRight

Dans le fichier « mafenetre.cpp » ajouter les lignes suivantes :

```
char version[30];
uint8_t serial[4];
char stackReader[20];

status = Version(&MonLecteur, version, serial, stackReader);
ui->Affichage->setText(version);
ui->Affichage->update();
```

Voici le résultat :



9 Fenêtre de saisie

Créez un nouveau bouton « saisie » puis une fenêtre de saisie avec le widget « Text Edit »

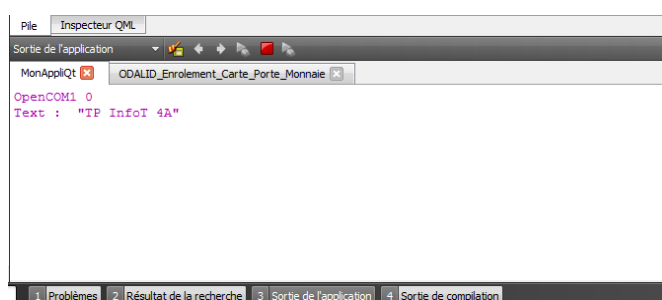
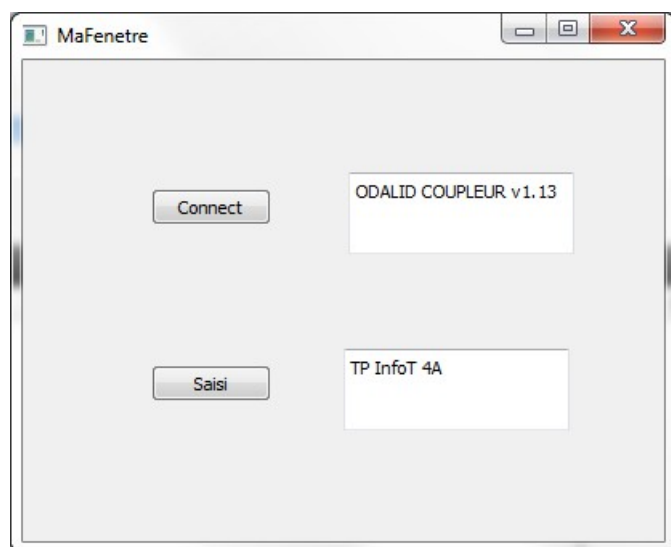
Dans le fichier « mafenetre.cpp » ajouter les lignes suivantes :

```
QString Text = ui->fenetre_saisi->toPlainText();
qDebug() << "Text : " << Text;
```

Lancez le programme en mode Debug.

Saisissez un texte dans la fenêtre de saisie, puis cliquez sur le bouton saisi.

La chaîne de caractère apparaît dans la fenêtre sortie d'application de Qt.



10 Fermer la fenêtre

Pour fermer une application, ajoutez un bouton Quitter à votre application.

Dans le fichier « mafenetre.cpp » ajouter les lignes suivantes :

```
void MaFenetre::on_Quitter_clicked()
{
    int16_t status = 0;
    RF_Power_Control(&MonLecteur, FALSE, 0);
    status = LEDBuzzer(&MonLecteur, LED_OFF);
    status = CloseCOM1(&MonLecteur);
    qApp->quit();
}
```

11 Portabilité

Pour pouvoir exécuter vos applications, vous devez mettre au même niveau que votre exécutable les fichiers suivants :

- ODALID.dll propre à la librairie chargée ;
 - mingwm10.dll et libgcc_s_dw2-1.dll propre à MinGW
- et
- QtCored4.dll et QtGuid4.dll propre à Qt en mode Debug
 - ou QtCore4.dll et QtGui4.dll propre à Qt en mode Release