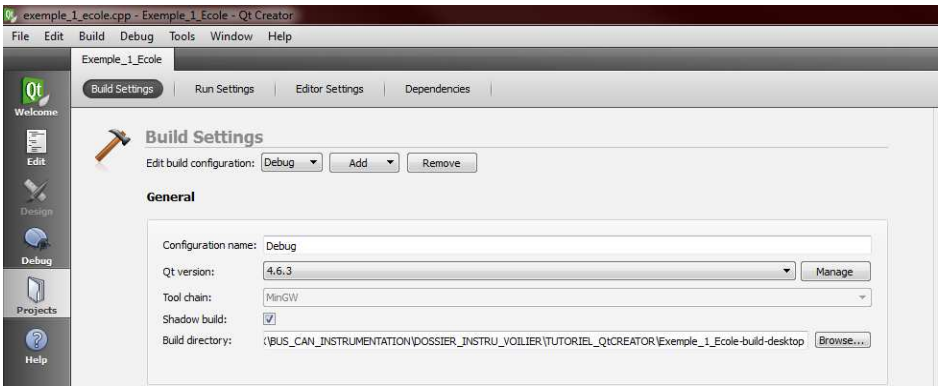
TUTORIEL Qt CREATOR

Dans ce document, vous trouverez deux écoles de conception sous Qt Creator. La première école consiste à placer les objets (boutons, zones de texte, afficheurs, etc...) où l'on veut sur une grille. Au contraire, la deuxième école utilise des fonctions pour un placement automatique de vos objets.

PREAMBULE : Ouvrir un projet Existant :

Ouvrir le fichier '.pro' avec Qt Creator

Aller dans l'onglet 'Projects' et redéfinir le 'Build directory'



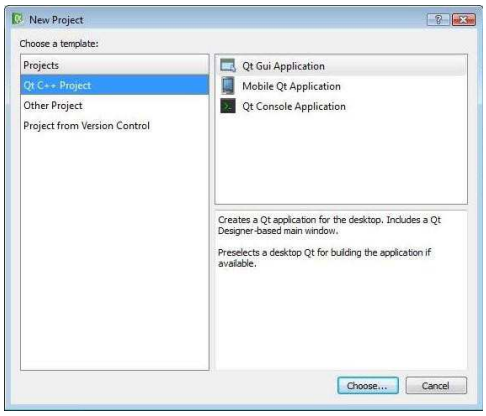
Il est alors possible de recompiler le projet et de l'exécuter

1 Première école

Tout d'abord, ouvrez le logiciel *Qt Creator* et suivez les instructions ci-dessous.

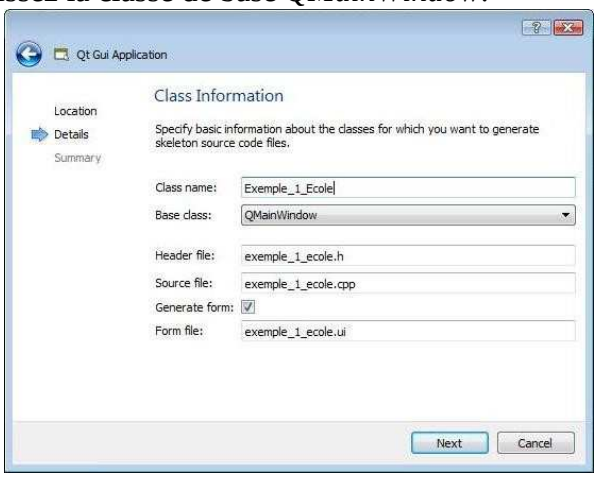
Sélectionnez *Create Project.*

Sélectionnez *Qt C++ Project* puis *Qt Gui Application.*



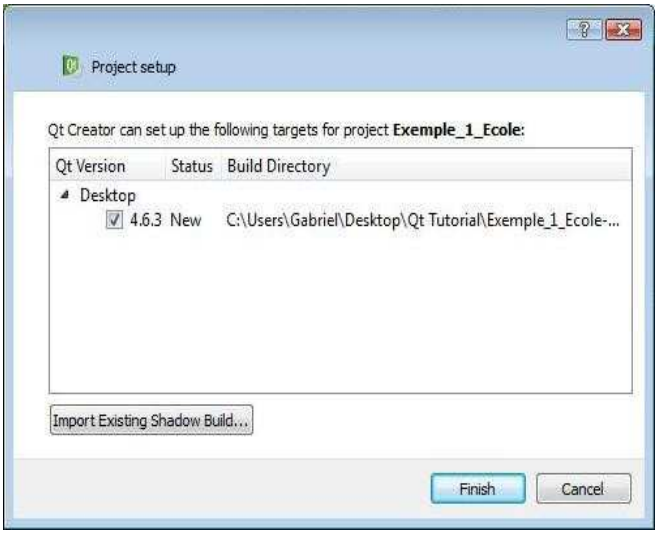
Renommez votre projet et choisissez le dossier cible.

Renommez la classe créée et choisissez la classe de base *QMainWindow*.

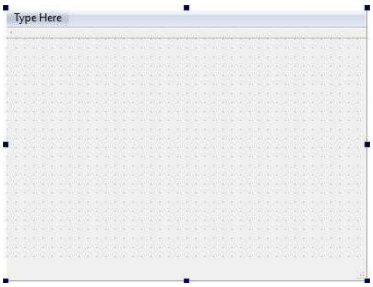


Cliquez sur Finish.

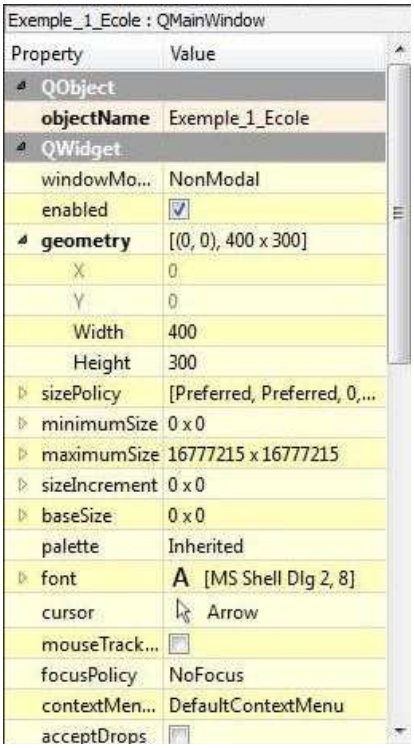
Cliquez sur Finish.



Votre projet vient d'être créé et vous tombez directement sur la fenêtre de *Form*.

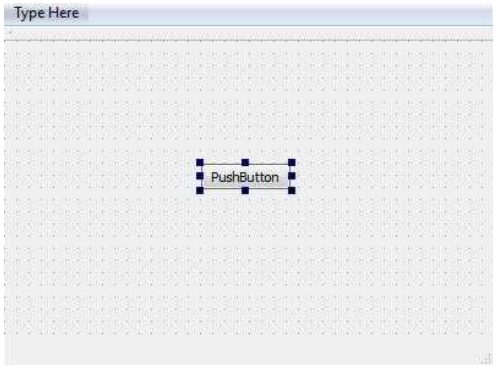


Cette grille permet de designer votre programme en y plaçant des boutons, des zones de textes, mais aussi des afficheurs LCD, des barres de chargement, des glissières, etc...
Si vous cliquez sur la fenêtre *Form*, vous pouvez voir les propriétés de la fenêtre.

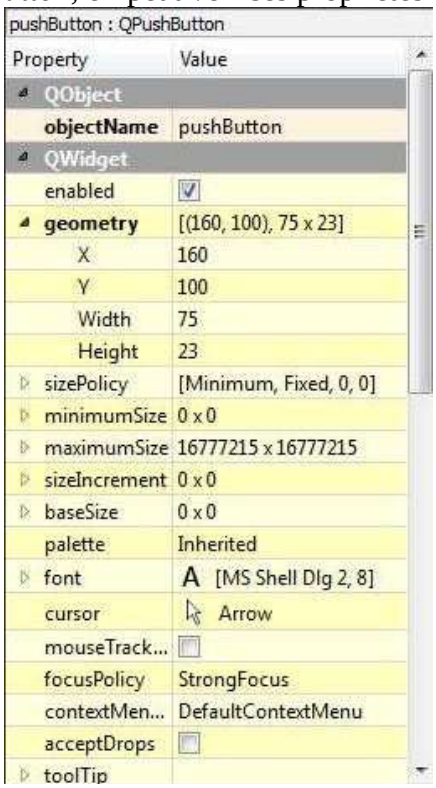


Sur le côté gauche de *Qt Creator* se trouvent une série d'objets.

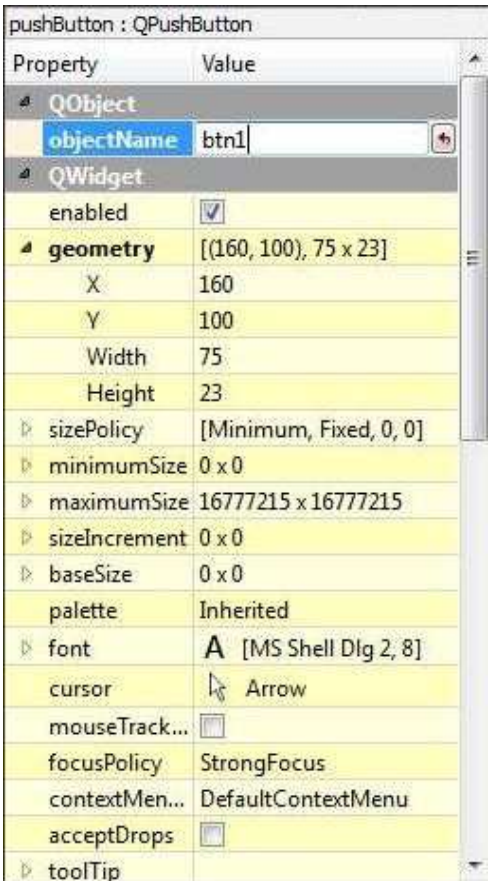
Faites glisser un *Push Button* dans *Form*.



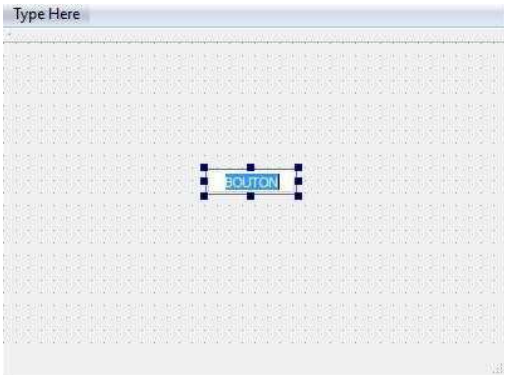
Lorsque que l'on clique sur l'objet *Push Button*, on peut voir ses propriétés en bas à droite de *Qt Creator*.



On peut voir le nom de l'objet et sa géométrie (position puis dimensions largeur, hauteur).
Maintenant renommons le bouton.

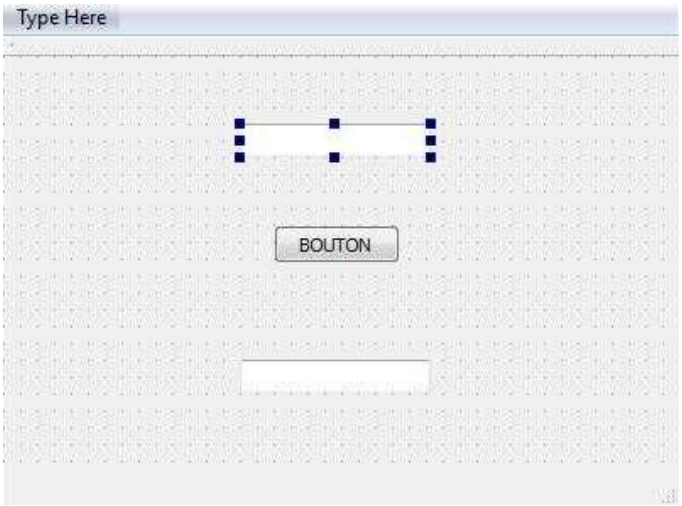


Changer le texte dans le bouton.

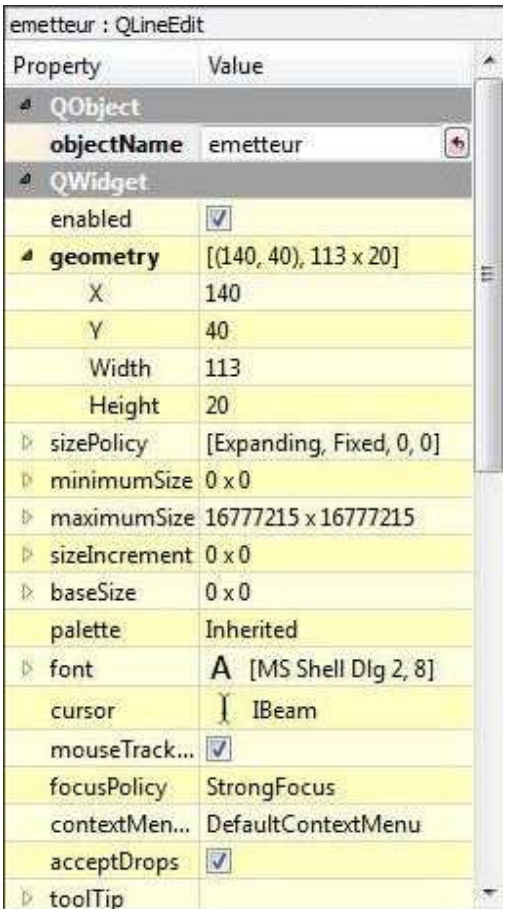


Une autre série d'objets.

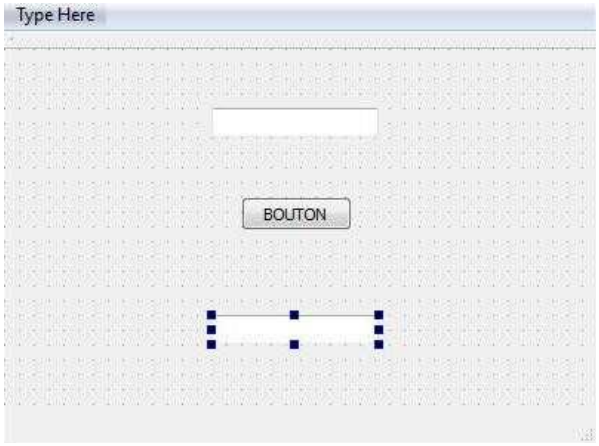
Faites glisser une *Line Edit* dans *Form*.



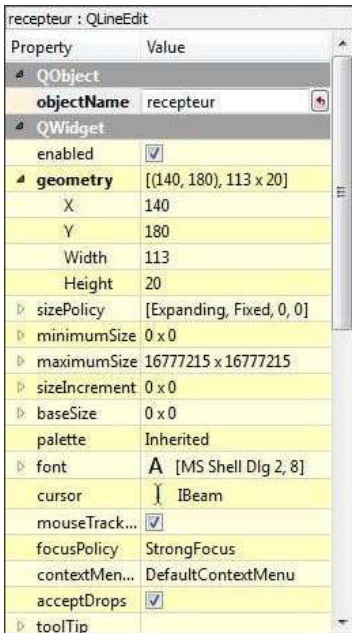
Voici les propriétés de la *Line Edit* renommée en *emetteur*.



Faites glisser une autre *Line Edit.*



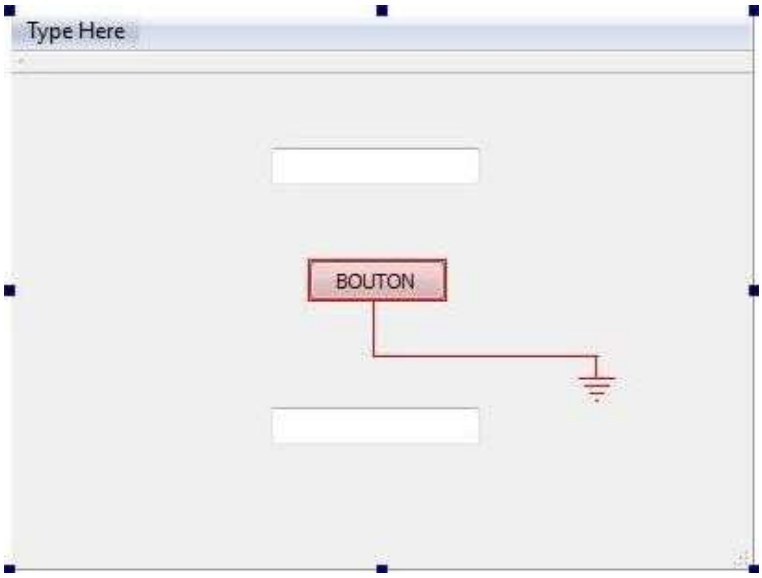
Voici les propriétés de cette autre *Line Edit* renommée en *recepteur*.



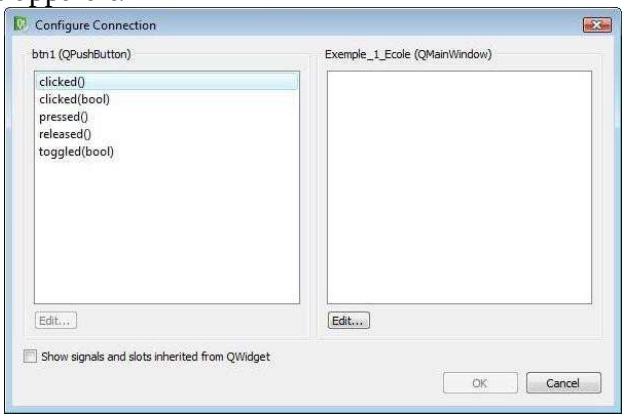
Maintenant, allons dans l'atelier d'édition des signaux et slots (deuxième bouton de la barre des ateliers).

Cet atelier va nous permettre de créer un slot (lien vers une fontion) lorsque que l'on déclanche le signal *clicked()* qui est émis lorsqu'on clique sur le bouton.

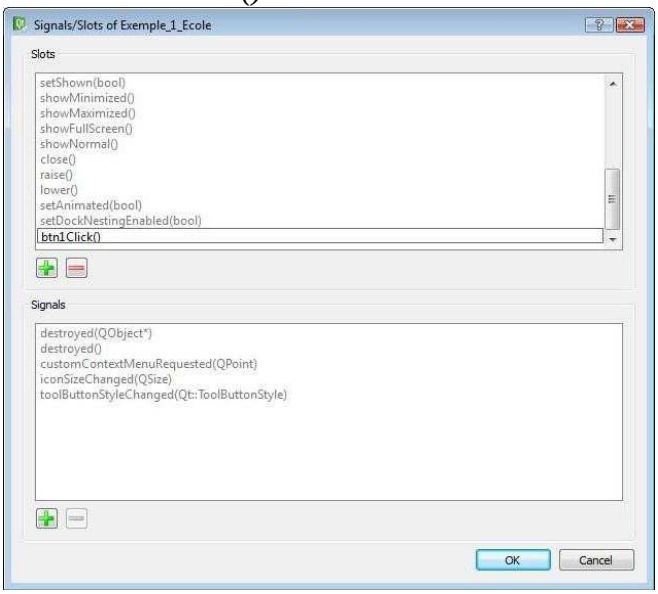
Cliquez sur le bouton et faites glisser.



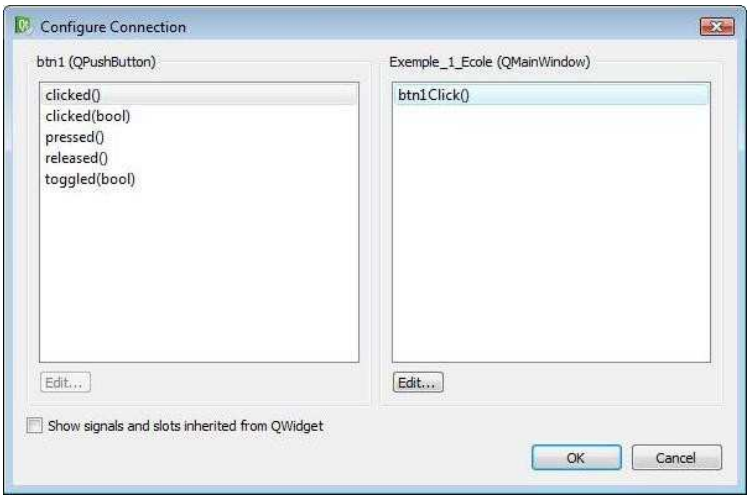
Relâchez. Une fenêtre de dialogue apparaît.



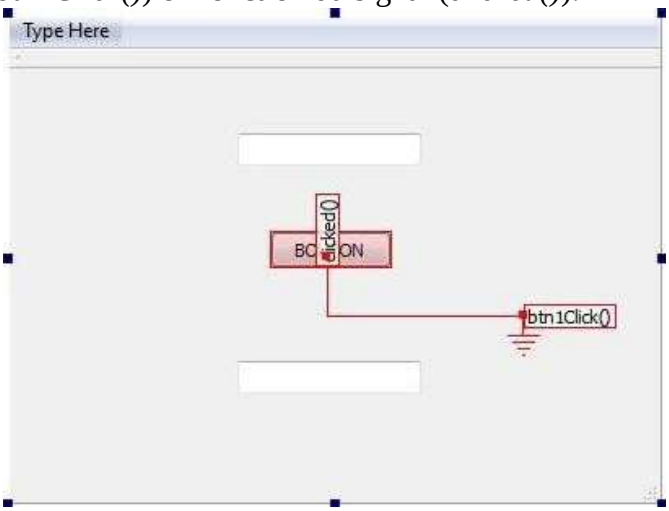
Sélectionner le signal *clicked()* puis cliquez sur *Edit* des slots. Une nouvelle fenêtre de dialogue apparaît. Appuyer sur le + des slots et créez le slot *btn1Click().*



Cliquez sur OK, sélectionnez *btn1Click()* puis cliquez sur OK.



Nous pouvons voir le slot créé (*btn1Click()*) en fonction du signal (*clicked()*).



Revenons dans l'atelier d'édition des *Widgets* (premier bouton de la barre des ateliers).

Maintenant, écrivons un peu de code. Allez dans le fichier header *Exemple_1_Ecole.h* et déclarez le slot que nous venons de créer.

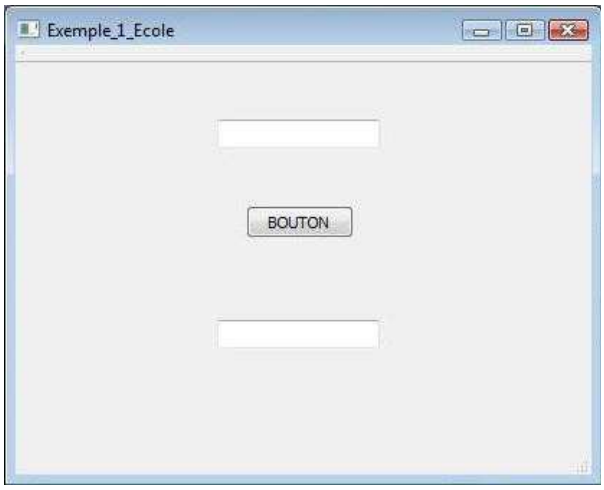
```
#ifndef EXEMPLE_1_ECOLE_H
#define EXEMPLE_1_ECOLE_H

#include <QMainWindow>

namespace Ui {
class Exemple_1_Ecole;
}

class Exemple_1_Ecole : public QMainWindow
{
Q_OBJECT

public:
```

```
explicit Exemple_1_Ecole(QWidget *parent = 0);
~Exemple_1_Ecole();

public slots:
void btn1Click();

private:
Ui::Exemple_1_Ecole *ui;
};

#endif // EXEMPLE_1_ECOLE_H
```

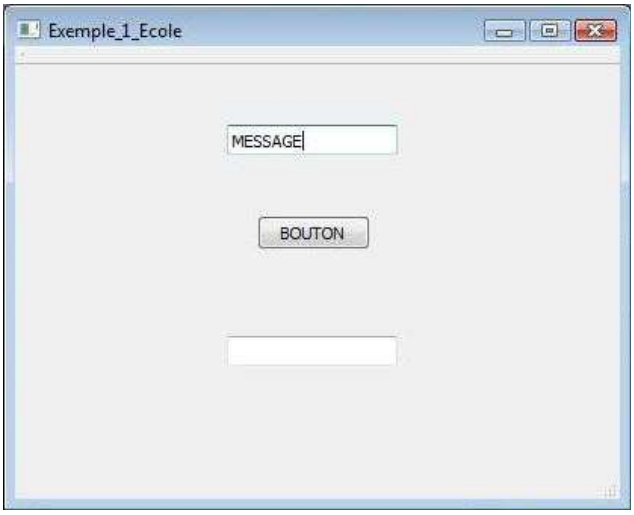
Puis allez dans le fichier source *Exemple_1_Ecole.cpp* et créez la fonction *void btn1Click()*.

```
#include "exemple_1_ecole.h"
#include "ui_exemple_1_ecole.h"

Exemple_1_Ecole::Exemple_1_Ecole(QWidget *parent) :
QMainWindow(parent),
ui(new Ui::Exemple_1_Ecole)
{
ui->setupUi(this);
}

Exemple_1_Ecole::~Exemple_1_Ecole()
{
delete ui;
}

void Exemple_1_Ecole::btn1Click()
{
ui->recepteur->setText(ui->emetteur->text());
ui->emetteur->clear();
}
```

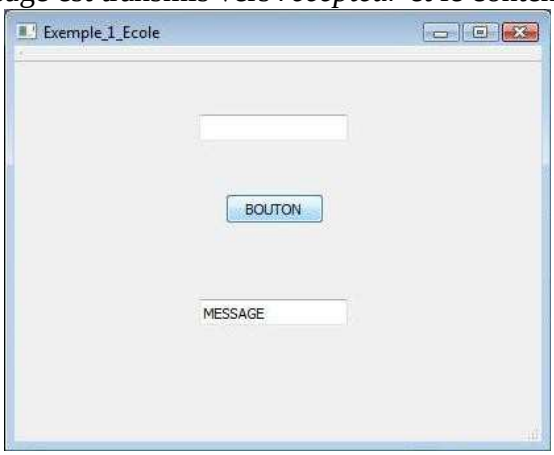
On a ainsi créé une fonction qui va récupérer le texte contenu dans *emetteur* et l'insérer dans *recepteur*, puis effacer le contenu de *emetteur*, tout ça lorsqu'on clique sur *btn1*.
Ainsi, on obtient ceci.



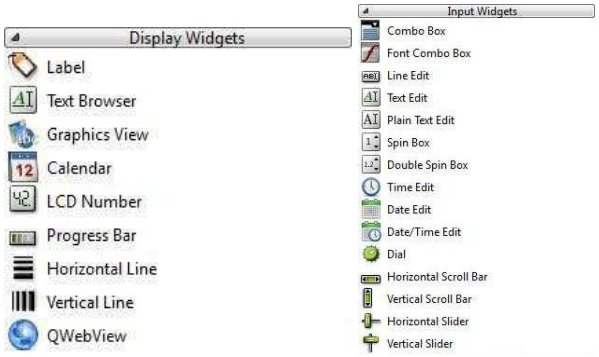
Ecrivez un message dans *emetteur*.



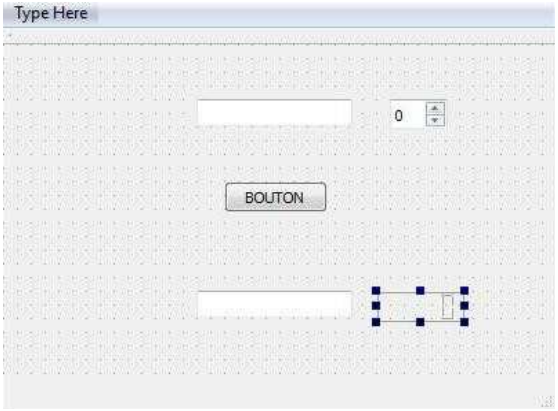
Appuyer sur le bouton *btn1*. Le message est transmis vers *recepteur* et le contenu de *emetteur* est effacé.



Maintenant, rajoutons une Spin Box et un LCD Number.



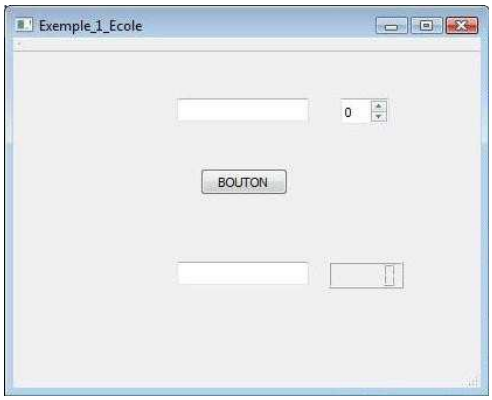
Form avec *Spin Box* renomée en *numero* et *LCD Number* renomé en *lcdNumero*.



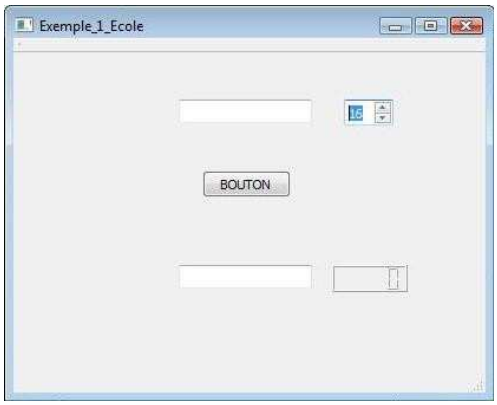
Retournons au code de la fonction *void btn1Click()*.

```
void Exemple_1_Ecole::btn1Click()
{
ui->recepteur->setText(ui->emetteur->text());
ui->emetteur->clear();
ui->lcdNumero->display(ui->numero->value());
ui->numero->setValue(0);
}
```

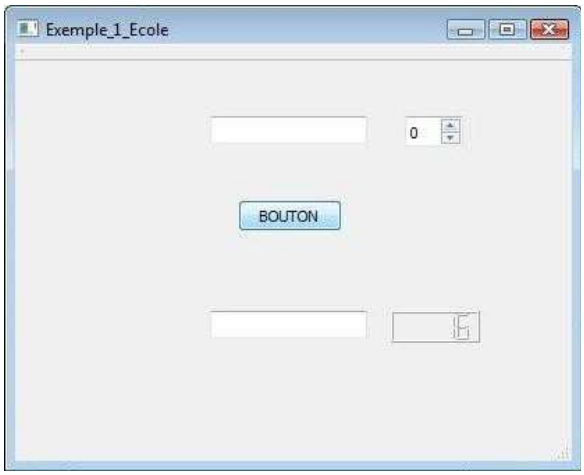
Nous avons rajouter du code pour que le valeur contenue dans *numero* soit transmise au *lcdNumero* et remettre à zéro la valeur de *numero*.
Soit le programme.



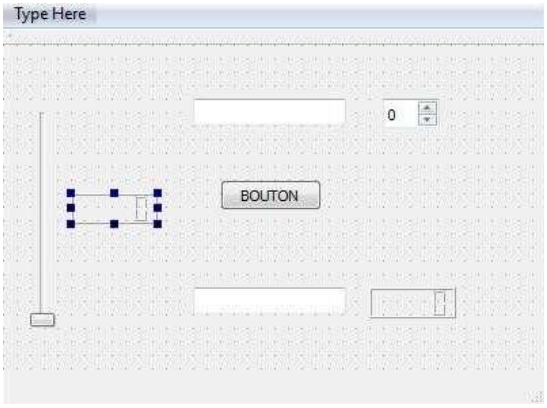
Mettre une valeur pour *numero*.



Cliquez sur le bouton *btn1*. La valeur de *numero* est transmise vers *lcdNumero* et *numero* est remis à zéro.



Pour finir, rajoutons un *Vertical Slider* et un *LCD Number*.
Renommons *Vertical Slider* en *verticalMobile* et *LCD Number* en *lcdVertical*.



Pour que notre valeur de *lcdVertical* corresponde à notre valeur de *verticalMobile*, on vient rajouter un *QTimer* qui va permettre d'exécuter cette fonction.
Dans *Exemple_1_Ecole.h*, ajouter.

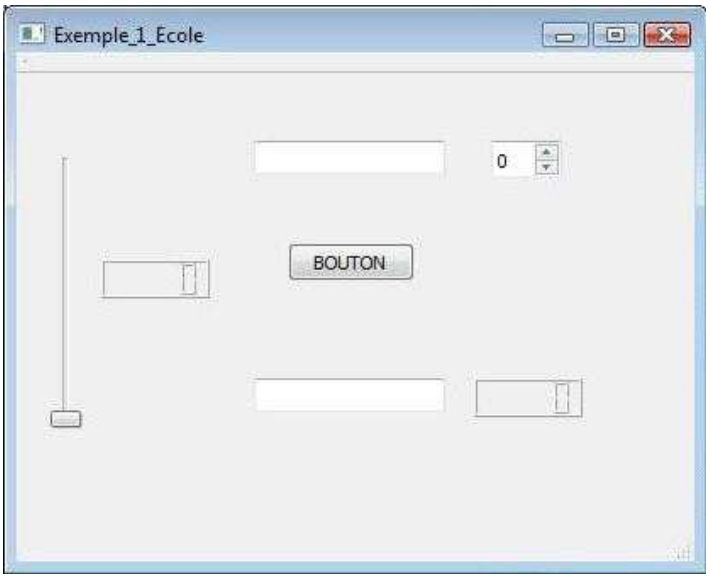
```
public slots:
void btn1Click();
void timerRead();
```

Dans *Exemple_1_Ecole.cpp*, ajouter.

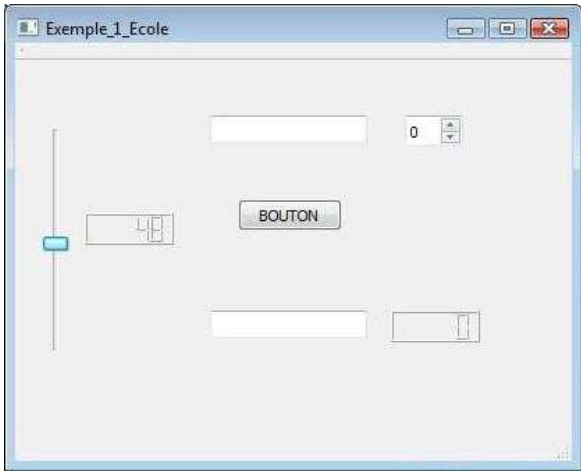
```
#include "exemple_1_ecole.h"
#include "ui_exemple_1_ecole.h"
#include <QTimer>

Exemple_1_Ecole::Exemple_1_Ecole(QWidget *parent) :
QMainWindow(parent),
ui(new Ui::Exemple_1_Ecole)
{
ui->setupUi(this);
QTimer *timer = new QTimer(this);
connect(timer, SIGNAL(timeout()), this, SLOT(timerRead()));
timer->setInterval(1);
timer->start();
}
```

```
Exemple_1_Ecole::~Exemple_1_Ecole()
{
delete ui;
}

void Exemple_1_Ecole::btn1Click()
{
ui->recepteur->setText(ui->emetteur->text());
ui->emetteur->clear();
ui->lcdNumero->display(ui->numero->value());
ui->numero->setValue(0);
}

void Exemple_1_Ecole::timerRead()
{
ui->lcdVertical->display(ui->verticalMobile->value());
}
```

Soit le programme.



Lorsque que la glissière *verticalMobile* est déplacée, la valeur de *lcdVertical* est immédiatement insérée.

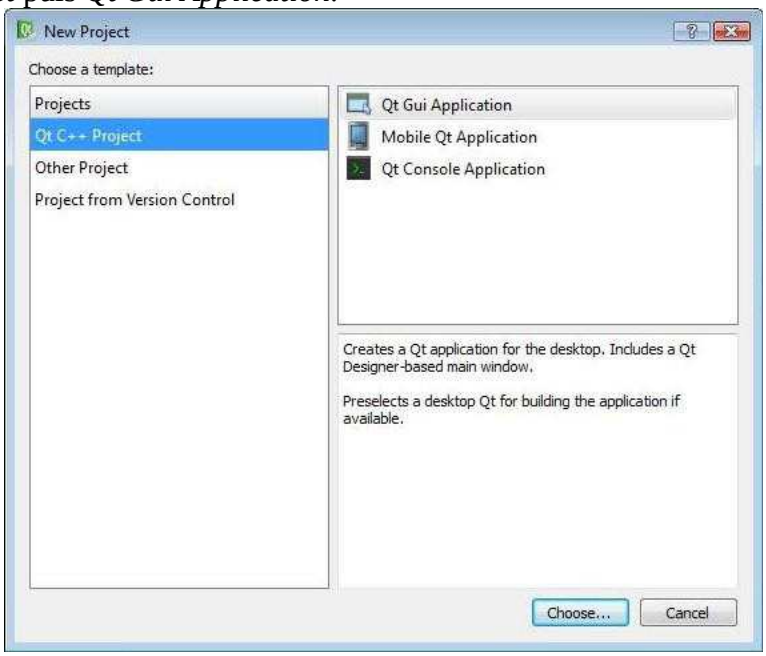


Voilà la fin de l'exemple pour la première école.

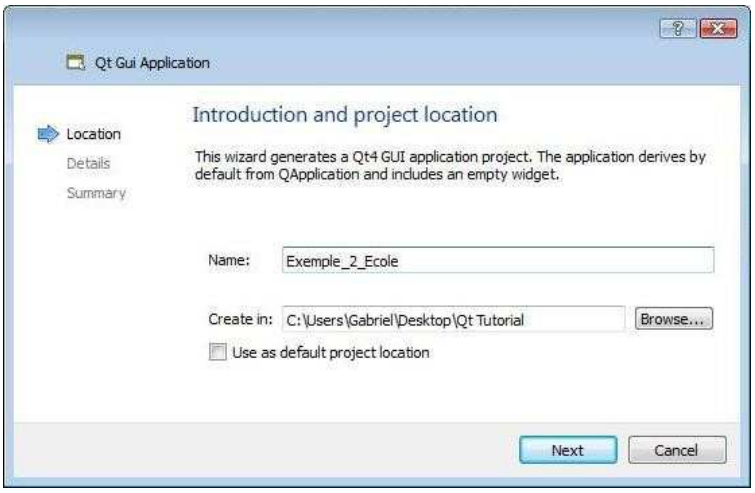
2 Deuxième école

Tout d'abord, ouvrez le logiciel *Qt Creator* et suivez les instructions ci-dessous.
Sélectionnez *Create Project*.

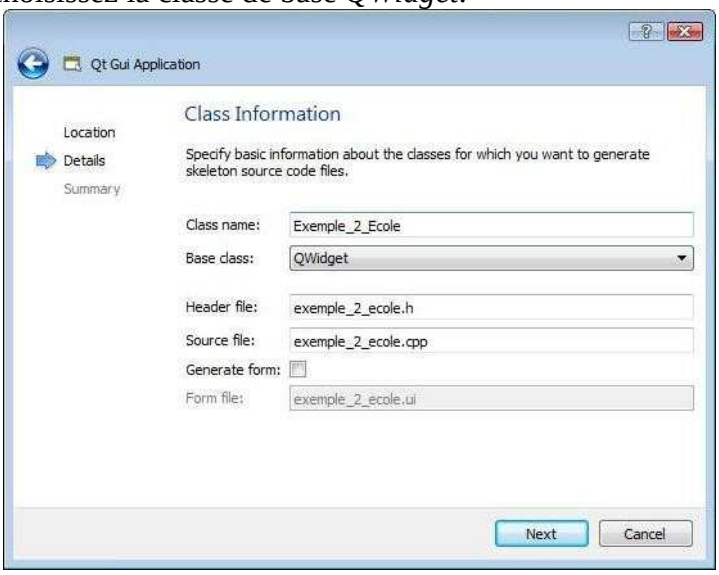
Sélectionnez *Qt C++ Project* puis *Qt Gui Application*.



Renommez votre projet et choisissez le dossier cible.



Renommez la classe créée et choisissez la classe de base *QWidget*.



Nous allons voir ensemble comment créer une fenêtre d'application à partir de rien.
Tout d'abord, il faut inclure les classes des objets que nous utilisons [*#include*], ainsi que déclarer les objets (les nommer [*private*]).
Dans le fichier header *Exemple_2_Ecole.h*, ajouter.

```
#ifndef EXEMPLE_2_ECOLE_H
#define EXEMPLE_2_ECOLE_H

#include <QtGui/QWidget>
#include <QPushButton>
#include <QLineEdit>
#include <QSpinBox>
#include <QLCDNumber>
#include <QSlider>
#include <QTimer>

class Exemple_2_Ecole : public QWidget
{
Q_OBJECT

public:
```

```
Exemple_2_Ecole(QWidget *parent = 0);
~Exemple_2_Ecole();

private:
QPushButton* btn1;
QLineEdit* emetteur;
QLineEdit* recepteur;
QSpinBox* numero;
QLCDNumber* lcdNumero;
QSlider* horizontalSlider;
QLCDNumber* lcdSlider;
QTimer* timer;
};

#endif // EXEMPLE_2_ECOLE_H
```

Ensuite dans le fichier source *Exemple_2_Ecole*, il faut initialiser les objets que nous avons créer.

```
#include "exemple_2_ecole.h"

Exemple_2_Ecole::Exemple_2_Ecole(QWidget *parent)
: QWidget(parent)
, btn1(new QPushButton)
, emetteur(new QLineEdit)
, recepteur(new QLineEdit)
, numero(new QSpinBox)
, lcdNumero(new QLCDNumber)
, horizontalSlider(new QSlider)
, lcdSlider(new QLCDNumber)
, timer(new QTimer)
{
timer->start();
}
```

Maintenant, nous allons voir ensemble comment arranger le programme avec des mises en place semi-automatique. Premièrement, il faut choisir la forme générale du programme (c'est-à-dire comment vont être disposé les différentes zones d'objets). Nous prendrons vertical [*#include <QVBoxLayout>*]. Et pour les zones en elles-mêmes, nous prendrons un arrangement horizontal [*#include <QHBoxLayout>*].
Dans le fichier header, déclarer le fonction *void createUI()* qui nous permettra de créer la fenêtre.

```
#ifndef EXEMPLE_2_ECOLE_H
#define EXEMPLE_2_ECOLE_H

#include <QtGui/QWidget>
#include <QPushButton>
#include <QLineEdit>
#include <QSpinBox>
#include <QLCDNumber>
#include <QSlider>
#include <QTimer>
```

```
class Exemple_2_Ecole : public QWidget
{
Q_OBJECT

public:
Exemple_2_Ecole(QWidget *parent = 0);
~Exemple_2_Ecole();
void createUI();

private:
QPushButton* btn1;
QLineEdit* emetteur;
QLineEdit* recepteur;
QSpinBox* numero;
QLCDNumber* lcdNumero;
QSlider* horizontalSlider;
QLCDNumber* lcdSlider;
QTimer* timer;
};

#endif // EXEMPLE_2_ECOLE_H
```

Deuxièmement, dans le fichier source, il faut créer la fontion *void createUI()*.

```
#include "exemple_2_ecole.h"
#include <QVBoxLayout>
#include <QHBoxLayout>

Exemple_2_Ecole::Exemple_2_Ecole(QWidget *parent)
: QWidget(parent)
, btn1(new QPushButton)
, emetteur(new QLineEdit)
, recepteur(new QLineEdit)
, numero(new QSpinBox)
, lcdNumero(new QLCDNumber)
, horizontalSlider(new QSlider)
, lcdSlider(new QLCDNumber)
, timer(new QTimer)
{
timer->start();
createUI();
}

Exemple_2_Ecole::~Exemple_2_Ecole()
{
}

void Exemple_2_Ecole::createUI()
{
QHBoxLayout *layout1(new QHBoxLayout);
layout1->addWidget(emetteur);
```

```
layout1->addWidget(btn1);
layout1->addWidget(recepteur);

QHBoxLayout *layout2(new QHBoxLayout);
layout2->addWidget(numero);
layout2->addWidget(lcdNumero);

QHBoxLayout *layout3(new QHBoxLayout);
horizontalSlider->setOrientation(Qt::Horizontal);
layout3->addWidget(horizontalSlider);
layout3->addWidget(lcdSlider);

QVBoxLayout *mainLayout = new QVBoxLayout;
mainLayout->addLayout(layout1);
mainLayout->addLayout(layout2);
mainLayout->addLayout(layout3);
setLayout(mainLayout);
setWindowTitle("Exemple_2_Ecole");
}
```

Nous venons donc de créer trois couches horizontales d'objets (layout1, layout2, layout3) ainsi que l'arrangement vertical général (mainLayout).
Dans layout1 se trouve le bouton et les deux champs de texte, dans layout2 se trouve le compteur ainsi que l'afficheur LCD et dans layout3 se trouve le mobile horizontal et un autre afficheur LCD.
Avant de finir, il faut connecter le bouton au slot *btn1Click()* et le *timer* au slot *timerRead()*.
Donc, dans le fichier header, il faut ajouter.

```
#ifndef EXEMPLE_2_ECOLE_H
#define EXEMPLE_2_ECOLE_H

#include <QtGui/QWidget>
#include <QPushButton>
#include <QLineEdit>
#include <QSpinBox>
#include <QLCDNumber>
#include <QSlider>
#include <Qtimer>

class Exemple_2_Ecole : public QWidget
{
Q_OBJECT

public:
Exemple_2_Ecole(QWidget *parent = 0);
~Exemple_2_Ecole();
void createUI();
void connectUI();

public slots:
void btn1Click();
void timerRead();
```

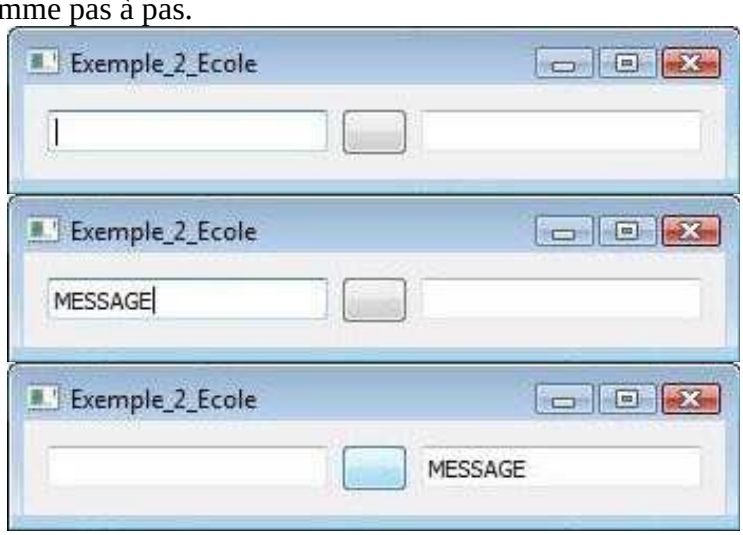
```
private:
QPushButton* btn1;
QLineEdit* emetteur;
QLineEdit* recepteur;
QSpinBox* numero;
QLCDNumber* lcdNumero;
QSlider* horizontalSlider;
QLCDNumber* lcdSlider;
QTimer* timer;
};

#endif // EXEMPLE_2_ECOLE_H
```

Pour finir, dans le fichier source, créer la fonction *void connectUI()* ainsi que les deux fonctions appartenant aux slots (identiques à l'exemple première école).

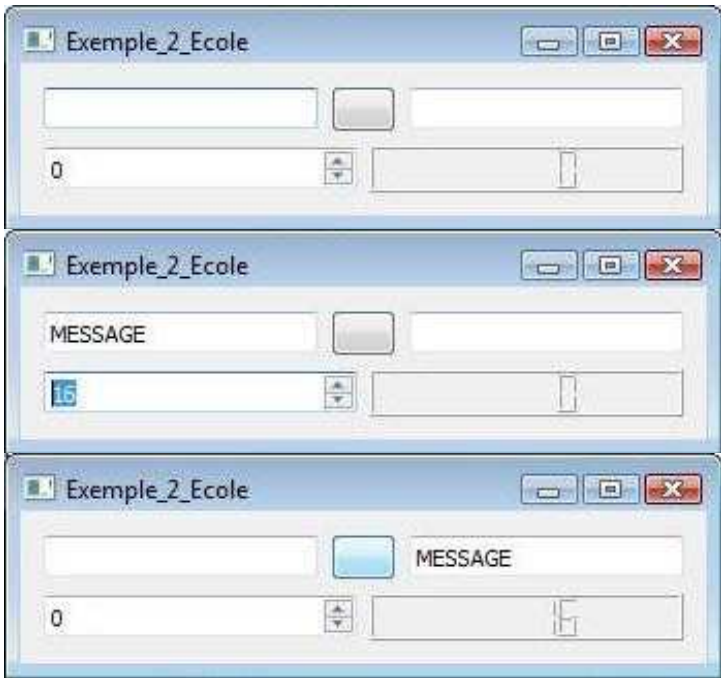
```
#include "exemple_2_ecole.h"
#include <QVBoxLayout>
#include <QHBoxLayout>

Exemple_2_Ecole::Exemple_2_Ecole(QWidget *parent)
: QWidget(parent)
, btn1(new QPushButton)
, emetteur(new QLineEdit)
, recepteur(new QLineEdit)
, numero(new QSpinBox)
, lcdNumero(new QLCDNumber)
, horizontalSlider(new QSlider)
, lcdSlider(new QLCDNumber)
, timer(new QTimer)
{
timer->start();
createUI();
connectUI();
}
Exemple_2_Ecole::~Exemple_2_Ecole()
{
}

void Exemple_2_Ecole::createUI()
{
QHBoxLayout *layout1(new QHBoxLayout);
layout1->addWidget(emetteur);
layout1->addWidget(btn1);
layout1->addWidget(recepteur);
QHBoxLayout *layout2(new QHBoxLayout);
layout2->addWidget(numero);
layout2->addWidget(lcdNumero);
QHBoxLayout *layout3(new QHBoxLayout);
horizontalSlider->setOrientation(Qt::Horizontal);
```

```
layout3->addWidget(horizontalSlider);
layout3->addWidget(lcdSlider);
QVBoxLayout *mainLayout = new QVBoxLayout;
mainLayout->addLayout(layout1);
mainLayout->addLayout(layout2);
mainLayout->addLayout(layout3);
setLayout(mainLayout);
setWindowTitle("Exemple_2_Ecole");
}

void Exemple_2_Ecole::connectUI()
{
connect(btn1, SIGNAL(clicked()), this, SLOT(btn1Click()));
connect(timer, SIGNAL(timeout()), this, SLOT(timerRead()));
}

void Exemple_2_Ecole::btn1Click()
{
recepteur->setText(emetteur->text());
emetteur->clear();
lcdNumero->display(numero->value());
numero->setValue(0);
}

void Exemple_2_Ecole::timerRead()
{
lcdSlider->display(horizontalSlider->value());
}
```

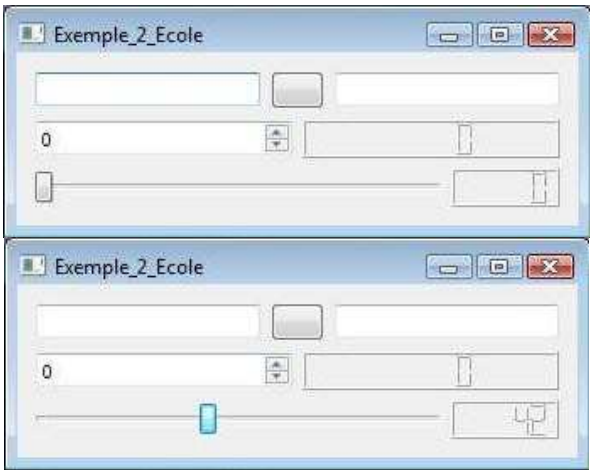
Voilà ce que donne le programme pas à pas.



Nous voyons ici la première zone d'objets (layout1).


Exemple_2_Ecole
0
Exemple_2_Ecole
MESSAGE
15
Exemple_2_Ecole
MESSAGE
0

Ici, nous pouvons voir l'insertion de la deuxième zone d'objets (layout1 puis layout2).



Cette dernière image nous montre le logiciel avec les trois zones d'objets (layout1 puis layout2 puis layout3).

Le travail s'achève, vous venez de créer un programme de deux manières différentes mais pour arriver au même résultat, à vous de choisir votre préférence.