/

```
#include <QApplication>
#include "mainwindow.h"
int main(int argc, char **argv)
{
      QApplication app(argc, argv); // un objet QApplication
      MaFenetre maFenetre; // un objet fenêtre
      maFenetre.show(); // affiche la fenêtre
      int ret = app.exec(); // exécute la boucle principale d'évènement
      return ret;
}
```

```
#include "mainwindow.h"
MaFenetre::MaFenetre( QWidget *parent ) : QDialog( parent )
{
      // 1. Instancier les widgets
      valeur = new QLineEdit(this);
      resultat = new QLabel(this);
      unite = new QLabel(this);
      choixConversion = new QComboBox(this);
      bConvertir = new QPushButton(QString::fromUtf8("Convertir"),
this);
      bQuitter = new QPushButton(QString::fromUtf8("Quitter"), this);
      // 2. Personnaliser les widgets
      valeur->setStyleSheet("color: #ffffff; background-color:
#000000;");
      valeur->clear();
      QFont font("Liberation Sans", 12, QFont::Bold);
      choixConversion->setFont(font);
      choixConversion->addItem(QString::fromUtf8("Celcius ->
Farenheit"));
      choixConversion->addItem(QString::fromUtf8("Farenheit ->
Celcius"));
      resultat->setStyleSheet("color: #0a214c;");
      unite->setStyleSheet("color: #0a214c;");
      // 3. Instancier les layouts
      QHBoxLayout *hLayout1 = new QHBoxLayout;
      QHBoxLayout *hLayout2 = new QHBoxLayout;
      QVBoxLayout *mainLayout = new QVBoxLayout;
      // 4. Positionner les widgets dans les layouts
      hLayout1->addWidget(valeur);
      hLayout1->addWidget(choixConversion);
      hLayout1->addWidget(resultat);
      hLayout1->addWidget(unite);
```

```
	hLayout2->addWidget(bConvertir);
	hLayout2->addWidget(bQuitter);
	mainLayout->addLayout(hLayout1);
	mainLayout->addLayout(hLayout2);
	setLayout(mainLayout);
	// 5. Connecter les signaux et slots
	connect(bConvertir, SIGNAL(clicked()), this, SLOT(convertir()));
	connect(this, SIGNAL(actualiser()), this, SLOT(convertir()));
	connect(choixConversion, SIGNAL(currentIndexChanged(int)),
this,SLOT(permuter()));
	connect(bQuitter, SIGNAL(clicked()), qApp, SLOT(quit()));
	connect(valeur, SIGNAL(textChanged(const QString &)), this,
SLOT(convertir()));
	// 6. Personnaliser la fenêtre
	setWindowTitle(QString::fromUtf8("Convertisseur de
températures"));
	adjustSize();
	// on lance une conversion
	//convertir();
	// ou :
	emit actualiser();
}
// 7. Définir les slots
void MaFenetre::convertir()
{
	QString temperature = valeur->text();
	if (temperature.isEmpty())
	{
	resultat->setText(QString::fromUtf8("--.--"));
	afficherUnite();
	return;
	}
	switch (choixConversion->currentIndex())
	{
	case CELCIUS_FARENHEIT:
		resultat->setText(QString::fromUtf8("%1").arg(9
*temperature.toDouble() / 5 + 32, 0, 'f', 2));
	break;
	case FARENHEIT_CELCIUS:
		double farenheit = 5 * (temperature.toDouble() - 32 ) / 9;
		resultat->setText(QString::number(farenheit, 'f', 2));
		break;
	}
}
void MaFenetre::permuter()
{
```

```
        if(resultat->text() != "--.--")
        {
        valeur->setText(resultat->text());
        emit actualiser();
        }
        afficherUnite();
}
// 8. Définir les méthodes
void MaFenetre::afficherUnite()
{
        switch (choixConversion->currentIndex())
        {
        case CELCIUS_FARENHEIT:
        unite->setText(QString::fromUtf8(" °F"));
        break;
        case FARENHEIT_CELCIUS:
        unite->setText(QString::fromUtf8(" °C"));
        break;
        }
}
```

```
#include <QApplication>
#define CELCIUS_FARENHEIT 0
#define FARENHEIT_CELCIUS 1

#if QT_VERSION >= 0x050000
#include <QtWidgets> /* tous les widgets de Qt5 */
#else
#include <QtGui> /* tous les widgets de Qt4 */
#endif
class MaFenetre : public QDialog
{
        Q_OBJECT
        // Membre(s) public(s)
public:
        MaFenetre( QWidget *parent = 0 );
        // Membre(s) privé(s)
private:
        // Les widgets
        QLineEdit *valeur;
        QLabel *resultat;
        QLabel *unite;
        QComboBox *choixConversion;
        QPushButton *bConvertir;
```

```
      QPushButton *bQuitter;
      QDoubleValidator *doubleValidator;
      void afficherUnite();
      // Mécanisme(s) Qt
signals:
      void actualiser();
private slots:
      void convertir();
      void permuter();

} ;
```