QT/C++

- CIR2 -

ISEN ALL IS DIGITAL! NANTES | yncréa

TP1

Ayoub KARINE (ayoub.karine@isen-ouest.yncrea.fr)

Rappel

La structure standard d'une IHM QT/C++ est :

```
#include <QApplication>
#include <QPushButton>
int main(int argc, char **argv)
{
 QApplication app (argc, argv);
 QPushButton button ("Hello world !");
 button.show();
 return app.exec();
}
```

Le fichier CMakeLists.txt doit ressembler à :

```
cmake_minimum_required(VERSION 3.10)
project(code)
set(CMAKE_CXX_STANDARD 14)
set(CMAKE_AUTOMOC ON)
find_package(Qt5Widgets CONFIG REQUIRED)
add_executable(code main.cpp)
target_link_libraries(code Qt5::Widgets)
```

Exercice 1

Dans la fonction main :

1. Créer un bouton avec le constructeur par défaut de la class QPushButton
2. Modifier le texte de ce bouton en utilisant la méthode setText de la classe QPushButton
3. En mettant le curseur de la souris sur le bouton, on doit avoir un texte qui apparaît. Pour ce faire, utiliser la méthode setToolTip de la classe QPushButton
4. Modifier le style (Courier), la taille (en 16) et la bordure du texte de bouton (2). Pour ce faire, utiliser la méthode setFont de la classe QPushButton
5. Afficher l'IHM

Exercice 2

Dans la fonction main :

1. Créer un bouton contenant le texte "parent" en utilisant un constructeur de la classe QPushButton
2. Créer un deuxième bouton contenant le texte "fils" en utilisant un constructeur de la classe QPushButton. Ce bouton va être le fils du premier bouton. Il n'est pas demandé de programmer l'héritage.
3. Afficher l'IHM

Exercice 3

Dans la fonction main :

1. Créer un widget avec le constructeur par défaut de la classe QWidget
2. Modifier la taille du widget en (largeur=100, hauteur=50) en utilisant la méthode setFixedSize de la classe QWidget
3. Créer un bouton contenant le texte "fils" en utilisant un constructeur de la classe QPushButton. Ce bouton va être le fils du widget. Il n'est pas demandé de programmer l'héritage.
4. Modifier la géométrie du bouton en utilisant les valeurs suivantes : (x=10, y=10, w=80, h=30). Pour ce faire, utiliser la méthode setGeometry de la classe QPushButton
5. Afficher l'IHM

Exercice 4

Cet exercice est une amélioration de l'exercice 3. On va plus tout programmer dans la fonction main.

1. Dans un fichier Window.h
    a. Créer une classe Window qui hérite de la classe QWidget
    b. Ajouter le macro Q_OBJECT avant la déclaration des méthodes et des attributs
    c. Déclarer le constructeur suivant : Window(QWidget* parent=0);

d. Déclarer, dynamiquement, un attribut privé de la classe QPushButton

2. Dans un fichier Window.cpp, définir le constructeur de la classe Window qui :

    a. Initialise son seul attribut en utilisant le constructeur de QPushButton
    b. Modifier la taille du widget en (largeur=100, hauteur=50) en utilisant la méthode setFixedSize
    c. Modifier la géométrie du bouton en utilisant les valeurs suivantes : (x=10, y=10, w=80, h=30)
    d. Changer le titre de la fenêtre IHM en utilisant la méthode setWindowTitle de la classe QWidget

3. Dans un fichier WindowTest.cpp, en plus de la structure standard de la création des IHM QT

    a. Créer un objet de la classe Window
    b. Pour afficher l'IHM, appeler la méthode show() de cet objet. Réellement, la méthode show() est une méthode de QWidget qui est hérité par la classe Window