

TP N°03 : Boites de Dialogue


Objectifs du TP


- ✓ A terme de ce TP, l'étudiant sera en mesure de créer des boites de dialogues ergonomiques, et de différencier les différents types de message.


Rappel


Il y a différents types de messages : **Confirmation, Information, Avertissement, et Erreur.**

Qtcreator nous permet d'afficher les boites de dialogue correspondantes grâce à la classe qui s'appelle **QMessageBox**. Elle possède un certain nombre de méthodes statiques qui permettent de choisir le type de message souhaité. A chaque type correspond une icône correspondante :

 **information()** : cette méthode ouvre une boite de dialogue avec un simple message informatif qu'il suffit juste de consulter.

 **question()** : cette méthode nous donne un message qui réclame un choix de notre part. Il faut alors proposer des boutons spécifiques en conséquence.

 **warning()** : cette méthode ouvre une boite de dialogue avec un message d'avertissement. Généralement, ce type de boite de message nous permet de valider un choix déjà fait.

 **critical()** : cette méthode ouvre une boite de dialogue avec cette fois-ci un message d'alerte. L'utilisateur doit alors être très attentif à ce type de message et prendre en compte l'avertissement proposé.

Remarque

En réalité, le choix de la méthode influence uniquement l'icône qui apparaîtra dans la boîte de dialogue. Avec n'importe quel type de boîte de dialogue, vous pouvez choisir le nombre de boutons que vous désirez avec l'intitulé souhaité.

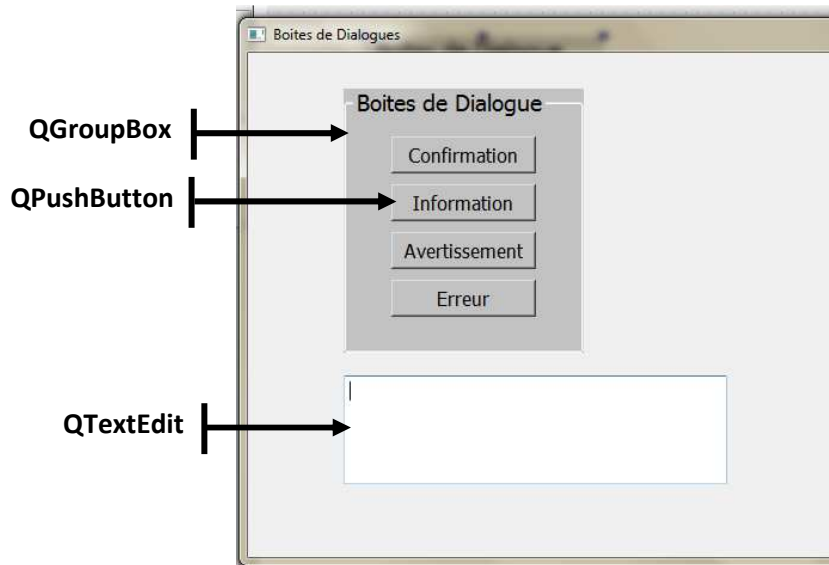
Chacune de ces méthodes renvoie une valeur entière qui correspond au choix effectué par l'utilisateur.

Si ce dernier clique sur le premier bouton de la boîte de dialogue, c'est la **valeur 0** qui est retournée.

Si l'utilisateur clique sur le deuxième bouton, c'est la **valeur 1** qui est retournée, etc.

Exercice 1

1. Lancer Qt creator, Ouvrir un nouveau projet
2. Mettre les widgets (Boutons de Boites de dialogue et textEdit seulement) comme le montre la figure ci-après.
3. Mettre le nom de TextEdit à **textEdithistorique**, pour les PushButton les noms sont identiques aux textes affichés.



4. Gestion des Signaux/Slots du Pushbutton « Confirmation »

- ✓ Cliquer avec le bouton droit de la souris sur ce bouton
- ✓ Cliquer sur Aller au Slot ...,
- ✓ Choisir la fonction clicked()
- ✓ Cliquer sur le bouton OK et compléter la fonction comme ci-dessous :

```
void MainWindow::on_Confirmation_clicked()
{
    int rep = QMessageBox::question(this, "Confirmation", "Voulez-vous
supprimer ce fichier ?", QMessageBox::Yes | QMessageBox::No |
QMessageBox::Cancel);

    ui->textEdithistorique->append( rep == QMessageBox::Yes ? "vous avez
choisi supprimer le fichier":rep == QMessageBox::No ? "vous avez
choisi de laisser le fichier":"vous avez choisi Annuler");
}
```

- ✓ Insérer la bibliothèque au début du fichier.cpp : #include <QMessageBox>
- ✓ Consulter le help (Mettre le curseur sur QMessageBox et taper F1)
- ✓ Compiler et Exécuter
- ✓ Critiquer cette boites de dialogue (titre, message, icônes, boutons).
- ✓ Quel est le style d'interaction utilisé dans cette boite de dialogue ?

5. Gestion des Signaux/Slots du Pushbutton « Information »

- ✓ Mêmes étapes que précédemment
- ✓ Cliquer sur le bouton OK et compléter la fonction comme ci-dessous :

```
void MainWindow::on_Information_clicked()
{
    QMessageBox::information(this, "Information", "Qtcreator a été
correctement désinstaller de cet ordinateur");
}
```

- ✓ Compiler et Exécuter
- ✓ Quelle est la différence entre les deux boites de dialogue « Confirmation et Information » ?

6. Gestion des Signaux/Slots du Pushbutton « Avertissement »

- ✓ Mêmes étapes que précédemment
- ✓ Cliquer sur le bouton OK et compléter la fonction comme ci-dessous :

```
void MainWindow::on_Avertissement_clicked()  
{  
    QMessageBox::warning(this, " Avertissement ", "Cette action ne peut  
pas être réalisée car le fichier est ou vert dans AdobeAcrobat  
9.0");  
}
```

- ✓ Compiler et Exécuter
- ✓ Quand on utilise cette boite de dialogue ?

7. Gestion des Signaux/Slots du Pushbutton « Erreur »

- ✓ Mêmes étapes que précédemment
- ✓ Cliquer sur le bouton OK et compléter la fonction comme ci-dessous :

```
void MainWindow::on_Erreur_clicked()  
{  
    QMessageBox::critical(this, " Erreur ", "une erreur critique !");  
}
```

- ✓ Compiler et Exécuter
- ✓ Critiquer ce message d'erreur. Est-il ergonomique ? justifier.