# TP N°01 : Prise en main du QT (Cute)
## 1ère partie

## Objectifs du TP

- ✓ Mise en œuvre de Qt Creator / Qt Designer
- ✓ Création d'un nouveau projet QT, utilisation de quelques widgets, personnalisation d'un widget et
- ✓ Mise en œuvre du mécanisme signal/slot.

## 1. Présentation du QT

- Qt (cute) est une **bibliothèque logicielle orientée objet** (API) développée en **C++** par Qt Development Frameworks, filiale de Digia. Qt est une plateforme de **développement d'interfaces graphiques GUI** (*Graphical User Interface*) fournie à l'origine par la société norvégienne Troll Tech, rachetée par Nokia en février 2008 puis cédée intégralement en 2012 à Digia (www.qt.io).

- De grandes entreprises utilisent Qt : Google, Adobe Systems, Asus, Samsung, Philips, ou encore la NASA et bien évidemment Nokia. Qt est notamment connu pour être la bibliothèque sur laquelle repose l'environnement graphique KDE, l'un des environnements de bureau les plus utilisés dans le monde Linux.

- Qt fournit un **ensemble de classes** décrivant des éléments graphiques (***widgets***) et des éléments non graphiques : accès aux données (fichier, base de données), connexions réseaux (*socket*), gestion du multitâche (*thread*), XML, etc.

- Qt permet la **portabilité des applications** (qui n'utilisent que ses composants) par simple recompilation du code source. Les environnements supportés sont les **Unix** (dont **Linux**), **Windows** et **Mac OS X**.

- Les ***widgets*** peuvent être utilisés pour créer ses propres **fenêtres** et **boîtes de dialogue** complètement prédéfinies (ouverture/enregistrement de fichiers, progression d'opération, etc).

- **Les interactions avec l'utilisateur** sont gérées par un mécanisme appelé ***signal/slot***. Ce mécanisme est la base de la programmation événementielle des applications basées sur Qt (Figure 1) :
  - ➢ un ***signal*** est émis lorsqu'un **événement** particulier se produit. Les classes de Qt possèdent de nombreux signaux prédéfinis mais vous pouvez aussi hériter de ces classes et leur ajouter vos propres signaux.

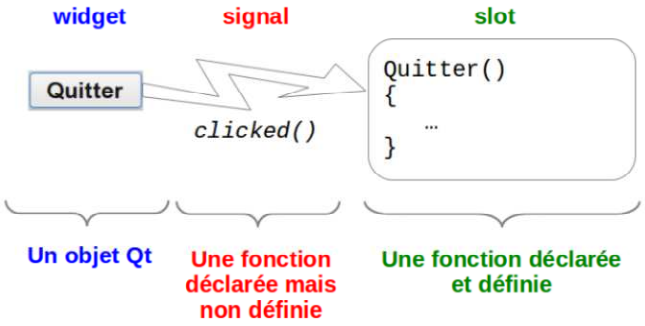



**Figure 1.Signal et slot.**

- un ***slot*** est une **fonction** qui va être appelée en réponse à un ***signal*** particulier. De même, les classes de Qt possèdent de nombreux *slots* prédéfinis, mais il est très courant d'hériter de ces classes et de créer ses propres *slots* afin de gérer les signaux qui vous intéressent.

## 2. Présentation du QT Creator

- **Qt Creator** est un environnement de développement intégré multiplate-forme faisant partie du framework Qt. Il est donc orienté pour la programmation en C++.
- L'éditeur de texte intégré permet l'auto-complétion ainsi que la coloration syntaxique.
- Qt Creator utilise sous Linux le compilateur gcc et **MinGW par défaut sous Windows**.
- Qt Creator intègre en son sein les outils **Qt Designer** et **Qt Assistant**. Il intègre aussi un mode débuggage. **Qt Designer** est un logiciel qui permet de créer des interfaces graphiques Qt dans un environnement convivial. L'utilisateur, par glisser-déposer, place les composants d'interface graphique et y règle leurs propriétés facilement. Les fichiers d'interface graphique sont formatés en XML et portent **l'extension .ui**.

## 3. Téléchargement et installation du QT

- Les liens de téléchargement **open-source** sont disponibles à l'adresse :
  https://www.qt.io/download-open-source

- Cliquer sur le bouton vert, une nouvelle page s'ouvre comme le montre la figure ci-dessous. Vous pouvez choisir l'installation « **Online** » ou le téléchargement « **offline des paquets** » :

  - Pour windows 10 et 11, l'installation « Online » de Qt vous permettra de choisir les paquets que vous souhaitez installer avant leur téléchargement. Ainsi, la durée de l'installation est considérablement diminuée. **Je vous recommande la version 5.12.12**
  - **NB. Cocher le compilateur qui convient à votre OS**

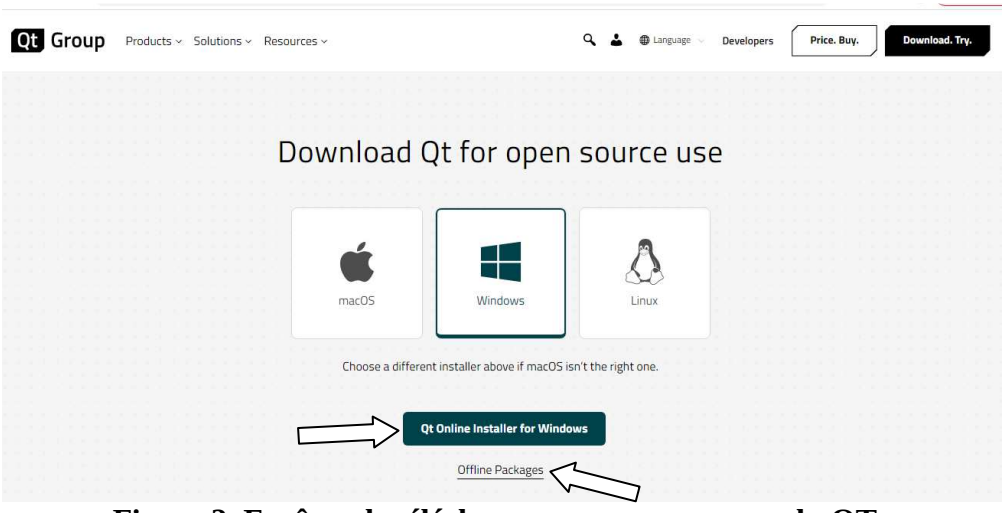



**Figure 2. Fenêtre de téléchargement open-source du QT.**

  - Pour les autres versions de windows (windows 10, windows 8.1, windows 7 (architectures x86 et x86_64)) cliquer sur **Offline Packages**, vous obtiendrez la page de la figure ci-dessous.
  - cliquer sur **5.12.x Offline Installers** ensuite cliquer sur **Qt 5.12.12 for windows**. Le téléchargement du fichier **qt-opensource-windows-x86-5.12.12.exe** commence.

- **Lancer le fichier exécutable et suivre les instructions ; cocher le compilateur MinGw 32 ou 64 bit selon votre OS.**

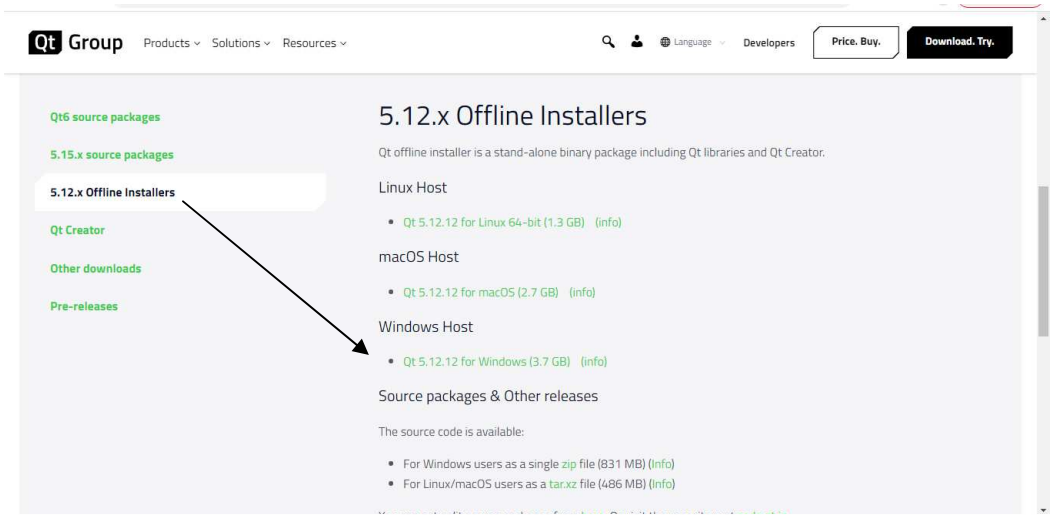



**Figure 3. Fenêtre de téléchargement offline du QT 5.12.12**

## 5.1 Téléchargement/ installation du QT creator 5.0.2

- Le lien de téléchargement du qt **creator 5.0.2** est disponible à cette adresse :
  - https://www.npackd.org/p/qt-creator64/5.0.2
- **qt creator 5.0.2** est pour windows 10, windows 8.1, windows 7 (architectures x86 et x86_64).
- Une fois le téléchargement terminé, exécuter le fichier et suivre les instructions.

La figure ci-dessous présente l'interface de Qt creator.

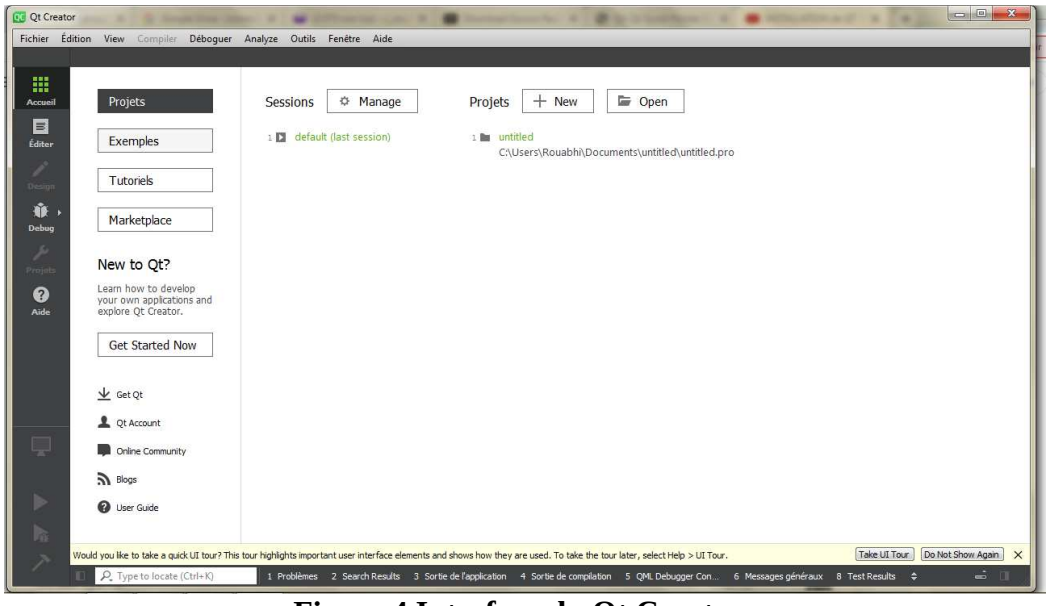
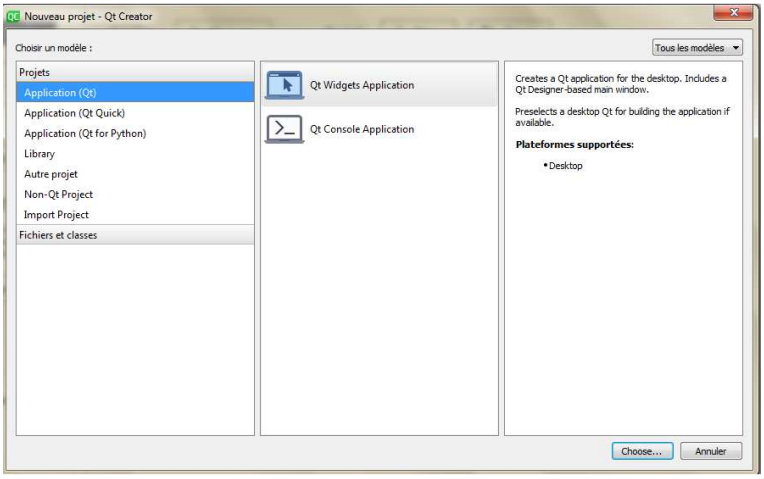


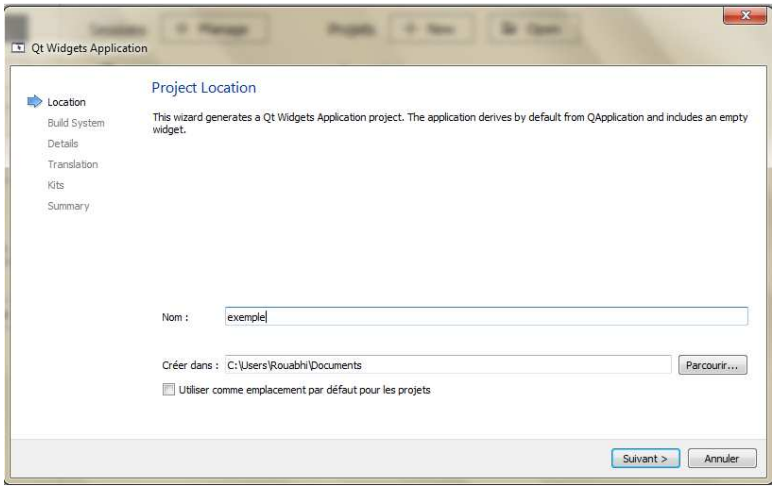

**Figure 4.Interface du Qt Creator.**

## 5.2 Création d'un nouveau projet Qt

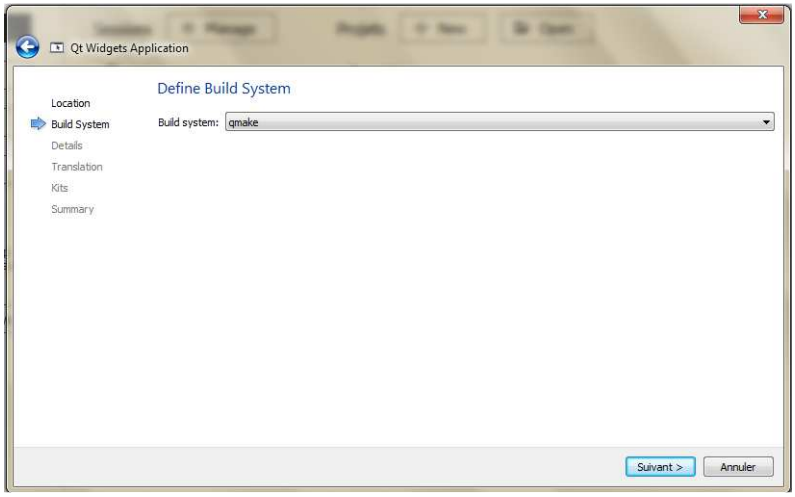
**Étape 1.** Lancer **QtCreator** et "Créer un projet ..."(Projets->New)
**Étape 2.** Choisir de développer une application Qt basée sur des *widgets* pour un PC (*desktop*).



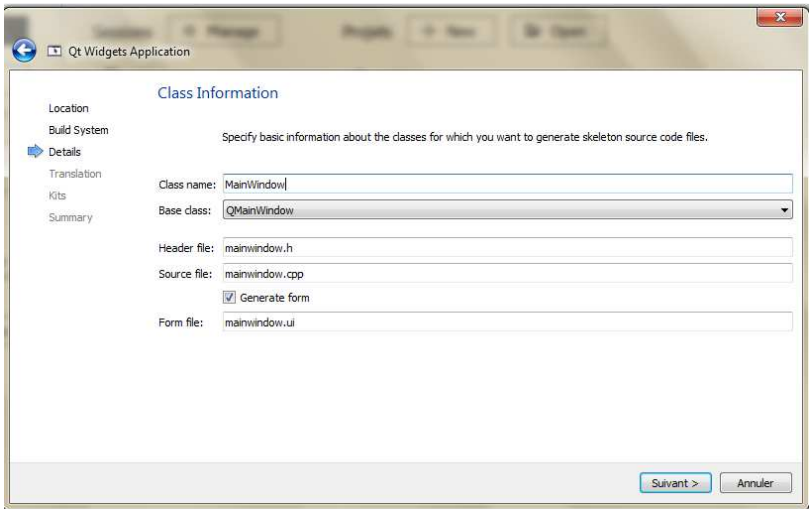

**Étape 3.** Donner un nom à votre projet et choisissez un emplacement.



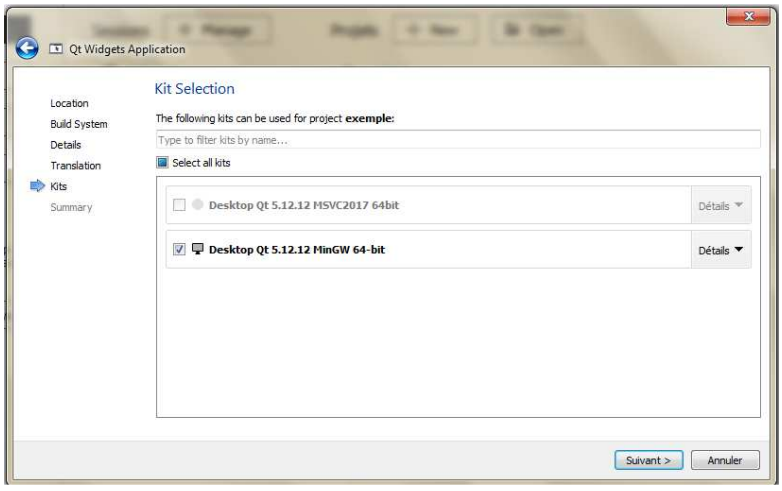

**Étape 4.** Choix de l'utilitaire de construction.



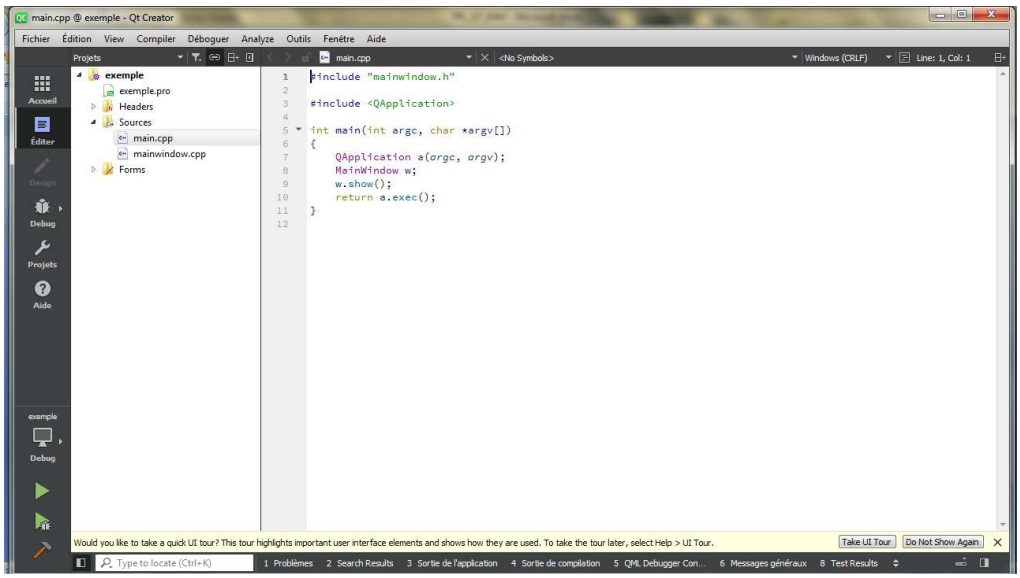

**Étape 5.** Donner un nom à votre classe principale et choisissez la classe de base (QMainWindow ou QDialog ou QWidget). Ces informations vont être utilisées par Qt pour générer les squelettes de votre application.



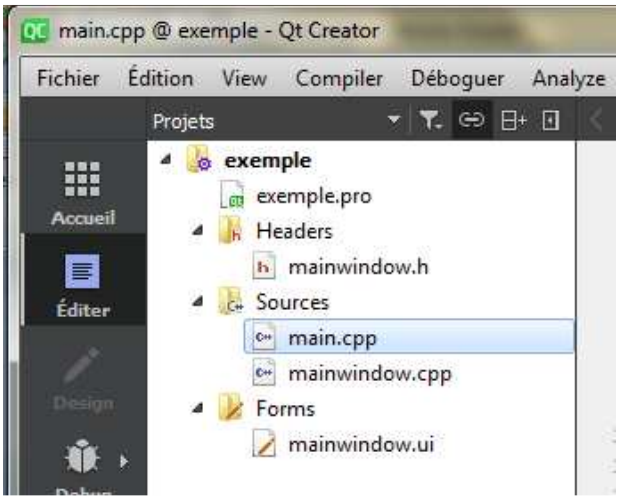

**Étape 6.** Selectionner le kit.




**Étape 7.** Terminer l'assistant de création de projet. La fenêtre ci-dessous s'affiche ; tous les fichiers sont dans le dossier **exemple** (nom du projet qu'on vient de créer).



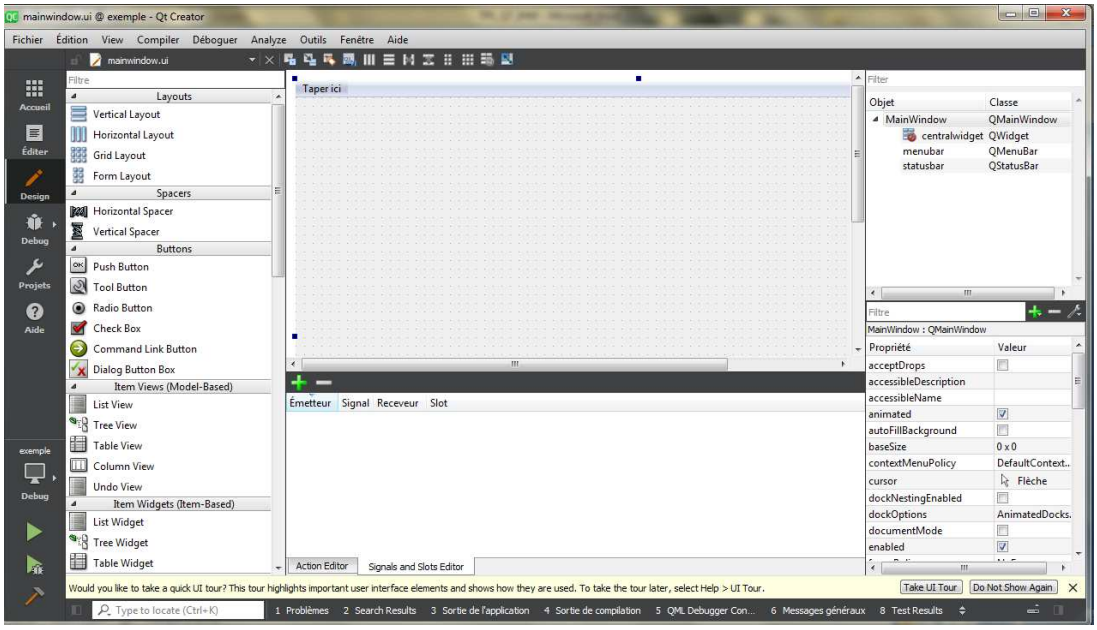
- Le **projet Qt** crée contient plusieurs dossiers comme le montre la figure ci-dessous. Il est défini par un fichier d'extension **.pro** décrivant la liste des fichiers sources, les dépendances, les paramètres passés au compilateur, etc...
- Pour contrôler ses propres utilitaires (moc, uic, ...), Qt fournit un moteur de production spécifique : le programme **qmake**. qmake prend en entrée un fichier de projet **.pro** et génère en sortie un fichier de fabrication spécifique à la plateforme.




**Étape 8.** Fabriquer et tester le squelette. Vous pouvez compiler et lancer l'application directement avec le **raccourci clavier Ctrl-R** ou la **petite flèche verte**.

**Étape 9.** Lancer **Qt Designer** (fenêtre ci-dessous). Il vous suffit de double-cliquer sur la fiche (dans notre exemple mainwindow.ui). Vous pouvez maintenant déposer les widgets (objets) que vous désirez avec un simple **glisser-déposer.**

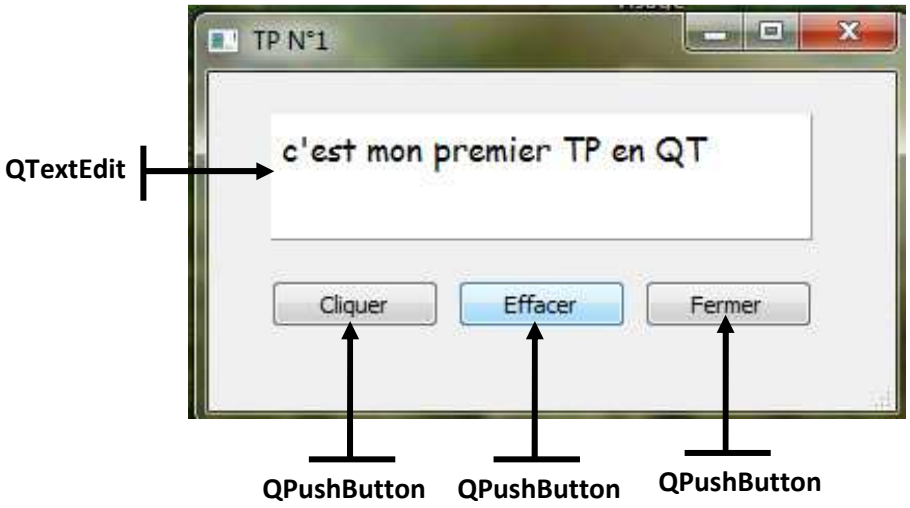
- Le filtre, dans le volet à gauche, pour un accès rapide aux différents widgets,
- Le volet à droite pour modifier les propriétés des objets
- La barre d'outils pour basculer entre le mode « Editer » et le mode « Design », pour l'execution, etc.



# TP N°01 : Prise en main du QT (Cute)
# 2ème partie

## Exercice 1

1. Lancer Qt et ouvrir un nouveau projet…
2. Déposer les différents widgets (objets graphiques) sur la fiche comme le montre la figure ci-dessous.
3. Modifier les propriétés des widgets comme suit :




**QTextEdit :**

- frameShape : Panel
- geometry : X :30, Y : 20, Largeur :262 et Hauteur :71
- objectName : textEdit
- readOnly : vrai

**QPushButton (Effacer):**

- enabled : faux
- text : Effacer

**4. Gestion des Signaux/Slots du Pushbutton « Fermer »**

- ✓ 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_pushButton_2_clicked()
{
    close();
}
```

- ✓ Compiler et Exécuter (petite flèche verte ou le raccourcis clavier Ctrl+R)

5. **Gestion des Signaux/Slots du Pushbutton « Cliquer »**
   - ✓ Même étapes que précédemment et
   - ✓ Compléter la fonction comme ci-dessous :

```
void MainWindow::on_pushButton_clicked()
{
    ui->textEdit->setText("c'est mon premier TP en QT");
    ui->pushButton_3->setEnabled(1);
    ui->pushButton->setEnabled(0);
}
```

   - ✓ Compiler et Exécuter pour vérifier.

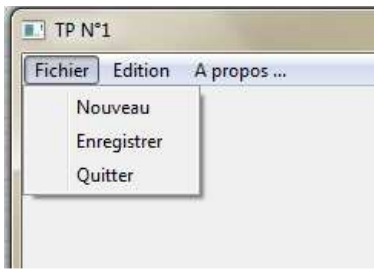
6. **Gestion des Signaux/Slots du Pushbutton « Effacer »**
   - ✓ Même étapes que précédemment et
   - ✓ Cliquer sur le bouton OK et compléter la fonction comme ci-dessous :

```
void MainWindow::on_pushButton_3_clicked()
{
   ui->textEdit->clear();
   ui->pushButton_3->setEnabled(0);
   ui->pushButton->setEnabled(1);
}
```

   - ✓ Compiler et Exécuter pour vérifier.

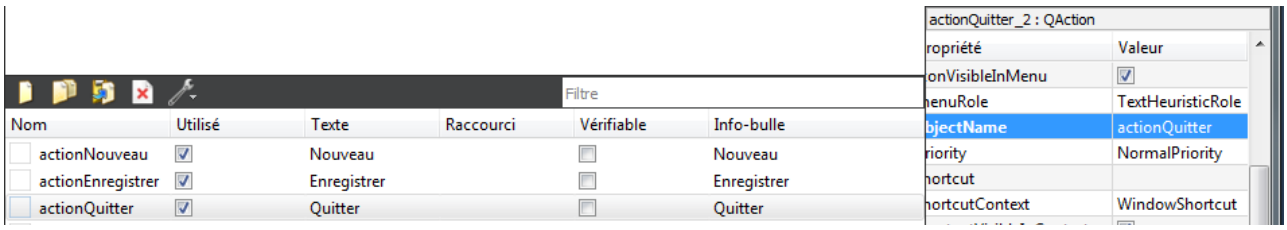
## Exercice 2

1. Lancer Qt et ouvrir un nouveau projet…
2. Modifier le titre de la fenêtre principale en modifiant la propriété **windowTitle : TP N°1**
3. Concevoir la barre de menu (figures ci-dessous) :

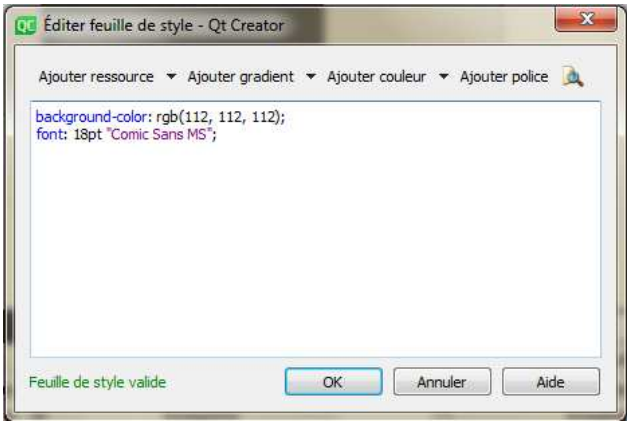





- ✓ Pour rajouter des entrées de menu, il suffit de taper les noms pour les menus et sous menus. L'ajout d'une entrée de menu rajoute également une **action visible dans la zone inférieure de QT Designer.**



4. Mettre le widget « **label** » sur la fenêtre principale avec ces propriétés :
   - textFormat : RichText
   - styleSheet : cliquer sur les « … », la fenêtre ci-dessous s'ouvre :
     - ✓ sélectionner dans « **Ajouter couleur » :** Background color et choisir la couleur gris foncé,
     - ✓ cliquer sur « **Ajouter police** » : choisir la taille 14pt et la police Comic Sans MS.
     - ✓ Cliquer sur le bouton OK.




5. **Gestion du Signal/Slot de la rubrique « Quitter » du menu « Fichier»**
   - ✓ Cliquer avec le bouton droit de la souris sur **« Action Editor** » -> actionQuitter ->Aller au Slot
   - ✓ Sélectionner la fonction **triggered**()
   - ✓ Cliquer sur le bouton OK et compléter la fonction comme ci-dessous :

```
void MainWindow::on_actionQuitter_triggered()
{
    close();
}
```

   - ✓ Compiler et Exécuter.

6. **Gestion du Signal/Slot du menu « A propos … »**
   - ✓ Mêmes étapes que précédemment et
   - ✓ Cliquer sur le bouton OK et compléter la fonction comme ci-dessous :

```
void MainWindow::on_actionA_propos_triggered()
{
    ui->label->setVisible(1);
    ui->label->setText("TP du module IHM 2023/2024");
}
```

   - ✓ Insérer la ligne en gras dans cette fonction :

```
MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
{
    ui->setupUi(this);
    ui->label->setVisible(0);
}
```

   - ✓ Compiler et Exécuter.

7. **Gestion du Signal/Slot de la rubrique « Gris » du menu « Edition»**

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

```
void MainWindow::on_actionGris_triggered()

    this->setStyleSheet("background-color: rgb(129, 129, 129);");
}
```

- ✓ Compiler et Exécuter.

8. **Gestion du Signal/Slot de la rubrique « Blanc » du menu « Edition»**

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

```
void MainWindow::on_actionBlanc_triggered()
{
   this->setStyleSheet("background-color: rgb(255, 255, 255);");
}
```

- ✓ Compiler et Exécuter.

## Exercice 3

1. Déposer sur la fiche deux objets LineEdit ;
2. Modifier le menu **Edition** en ajoutant les items suivants : Afficher, Copier et Coller ;
3. Faire les modifications nécessaires pour afficher le mot « Bonjour» dans le 1$^{er}$ lineEdit en cliquant sur « Afficher », le copier et le coller dans le 2$^{ème}$ lineEdit.