TP1 *Création de notre première interface en Qt*

Le but de ce TP est de réaliser une petite application simple permettant d'effectuer une opération arithmétique entre deux nombres et d'afficher le résultat. Les opérations à gérer sont : '+', '-', '*', '/'. Vous devez obtenir une interface ressemblant au dessin suivant :

Exercice 1 : Découverte de l'environnement Qt

Dans un premier temps, vous devrez identifier les différents composants qui ont servis à réaliser cette interface puis les placer pour obtenir le même comportement que celui qui apparaît sur le dessin ci-dessus. Enfin il faudra créer et connecter les divers `slots` et `signaux Qt` nécessaires au bon fonctionnement de l'application.

Pour réaliser cet exercice, vous devez utilisez les composants `Qt` à votre disposition mais vous devez exécuter toutes les phases de compilation manuellement. Nous rappelons que pour compiler une application `Qt` il faut rajouter comme arguments :

- Pour la phase de compilation : `-I/usr/include/qt4` où le répertoire en argument est le répertoire d'installation des entêtes de `Qt4`.
- Pour la phase de linkage : `-lQtCore` et `-lQtGui` pour indiquer qu'il faut lier avec ces deux bibliothèques. Il est parfois nécessaire de rajouter d'autres bibliothèques avec lesquels lier par exemple lors de l'utilisation de composants réseaux ou openGl.
- De plus il est nécessaire de générer un fichier `.cpp` supplémentaire pour toute classe (déclarée dans un fichier `.h`) qui définit sa propre section `signals` et/ou `slots`. Par exemple : `moc -o moc_calculator.cpp calculator.h` génère le fichier `moc_calculator.cpp` à partir du fichier `calculator.h` qui contient une classe définissant sa propre section `slots` et sa propre section `signals`. Il ne faut pas oublier de compiler ce fichier et de le lier avec tous les autres fichiers issus des phases de compilation.

Il s'agit d'un premier contact avec la librairie `Qt` prenez bien le temps de comprendre ce que vous écrivez ! Pensez à utiliser `assistant-qt4` qui contient toute la documentation des composants nécessaires à la réalisation de ce TP.

Exercice 2 : Séparation de l'interface du moteur

C'est une bonne pratique que de dissocier le code concernant l'interface graphique de celui du "moteur" (i.e. : la partie algorithmique). En effet, en cas de changement de librairie d'interface graphique, vous n'avez pas à changer le code source du moteur. De plus, sur une application conséquente, cette dissociation peut faciliter le travail collaboratif : une personne se charge de l'implémentation du moteur pendant qu'une autre se charge de l'implémentation de l'interface.

Vous allez maintenant refaire l'exercice 1 en pensant à dissocier le code concernant l'interface graphique de celui du "moteur". Pour cela, vous aurez besoin d'une classe qui fera la liaison (la traduction) entre l'interface graphique et le moteur. Vous aurez donc au final 3 classes :

- Vue : affiche les composant graphique et "communique" avec Traducteur.
- Traducteur : fait la liaison entre Vue et Moteur.
- Moteur : effectue les calculs.

La classe Moteur **ne doit pas faire appel à la librairie Qt**, elle doit être implémentée uniquement avec du C++ standard.