


TD 13 – Premiers Principes de Conceptions Orientée Objet

Cours 1 Interface Segregation Principle

1.1 Définition



- ➡ Généralement, ce principe revient à appliquer le principe de responsabilité unique au niveau de l'interface ou classe de base.

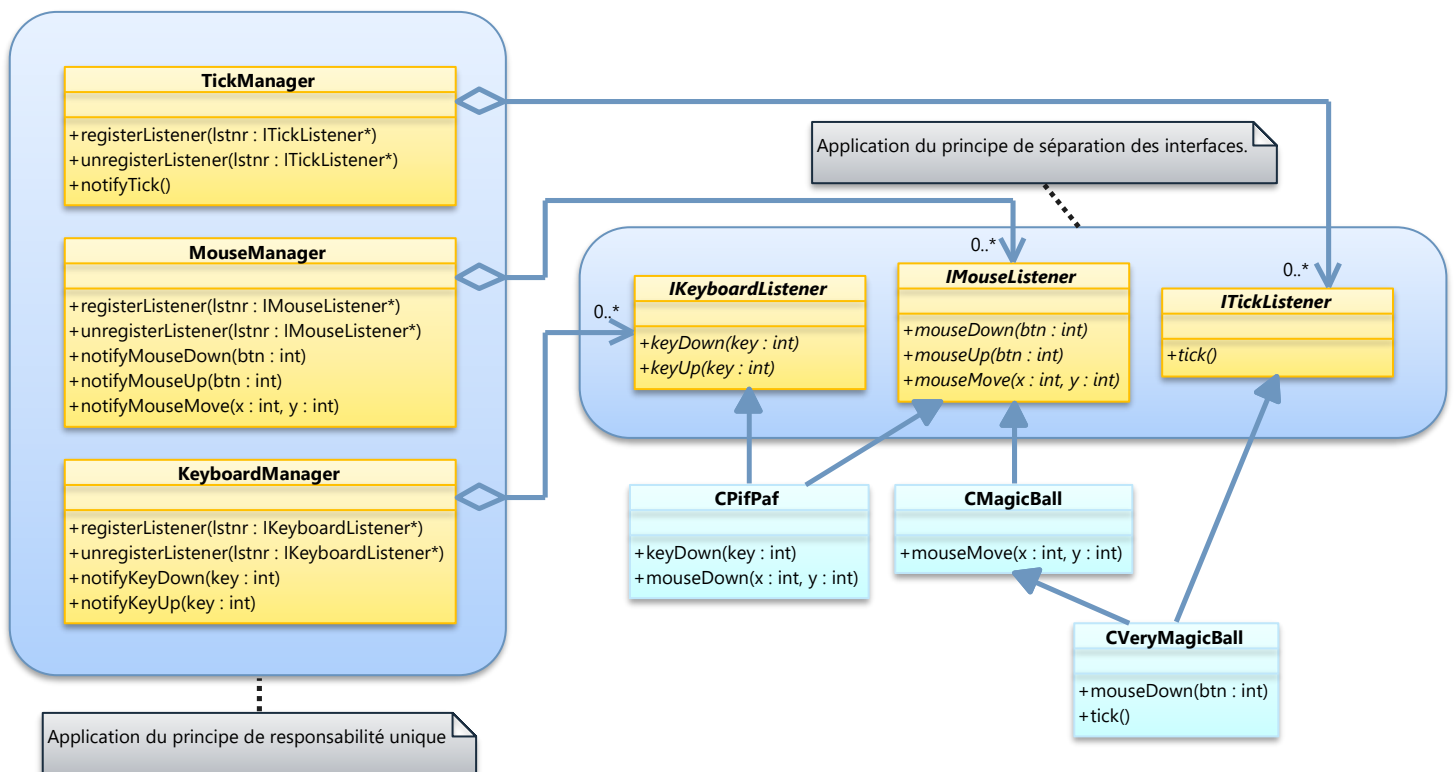
 LE SAVIEZ-VOUS ? Le découpage de Qt en différents modules (QtCore, QtWidgets, etc...) est en quelques sortes l'application de ce principe à un niveau supérieur.

Appliquer ce principe poursuit toujours l'idéal d'écrire des applications souples, facile à maintenir et à faire évoluer.

Exercice 2 BAKt après séparation des interfaces

-  Appliquez ce principe à la conception de **BAKt**.

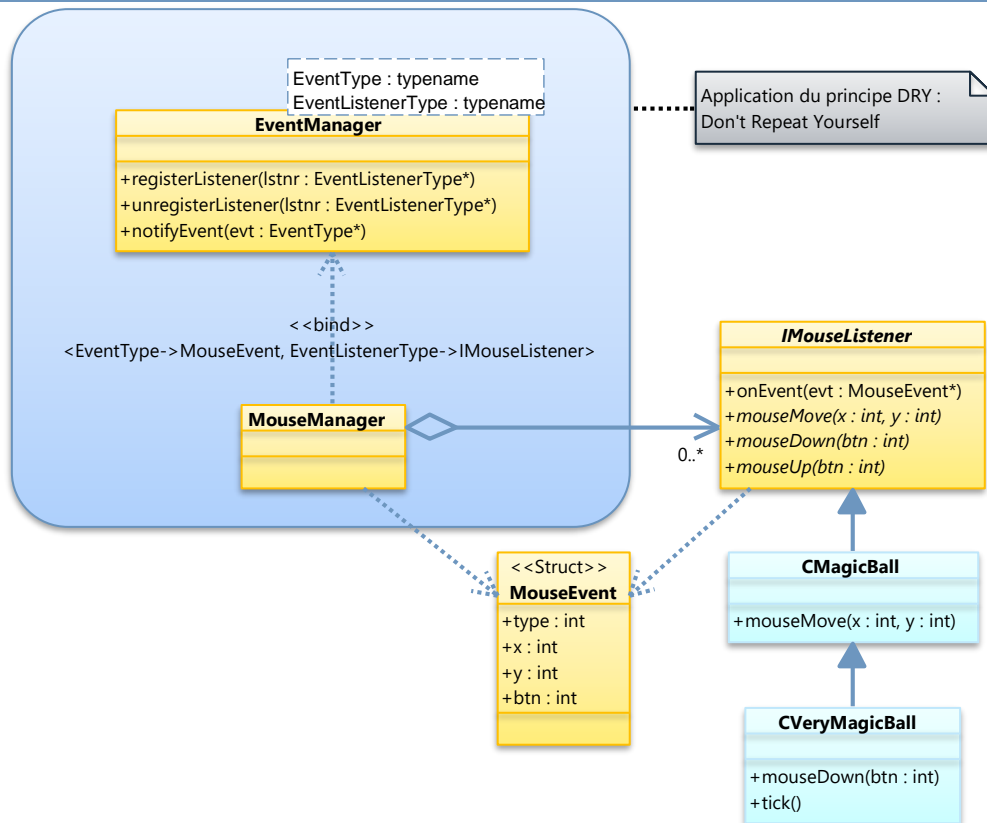
2.1 Solution proposée



🐼 **NOUVEAUTE** : bien que cela n'a pas été abordé en cours, il est possible en C++ de faire hériter une classe de plusieurs classes. Cela s'appelle de l'héritage multiple. La classe ainsi dérivée hérite des comportements de toutes les classes de base. Cela est très pratique dans le cas présent, mais peut devenir incompréhensible et ambiguë (y compris pour le compilateur) lorsque l'héritage multiple se fait à partir de classes de bases héritant elle-même d'une même classe ou proposant des méthodes identiques.

🐼 **Nota** : je pense qu'il n'est pas nécessaire d'épiloguer sur ce point.

C'est bien mais il y a beaucoup de redite de code dans les trois classes **XManager**. Cette conception ne respecte pas le principe **DRY** (*Don't Repeat Yourself*) et occasionne du code **WET** (*We Enjoy Typing*). Nous pouvons « assécher » cela en utilisant un template générique pour les gestionnaires d'événements qui transmettront aux **XListener** des objets concrets contenant les informations de l'événement. Voici ce que cela donne pour le cas des événements souris :



Et voici le diagramme complet :

