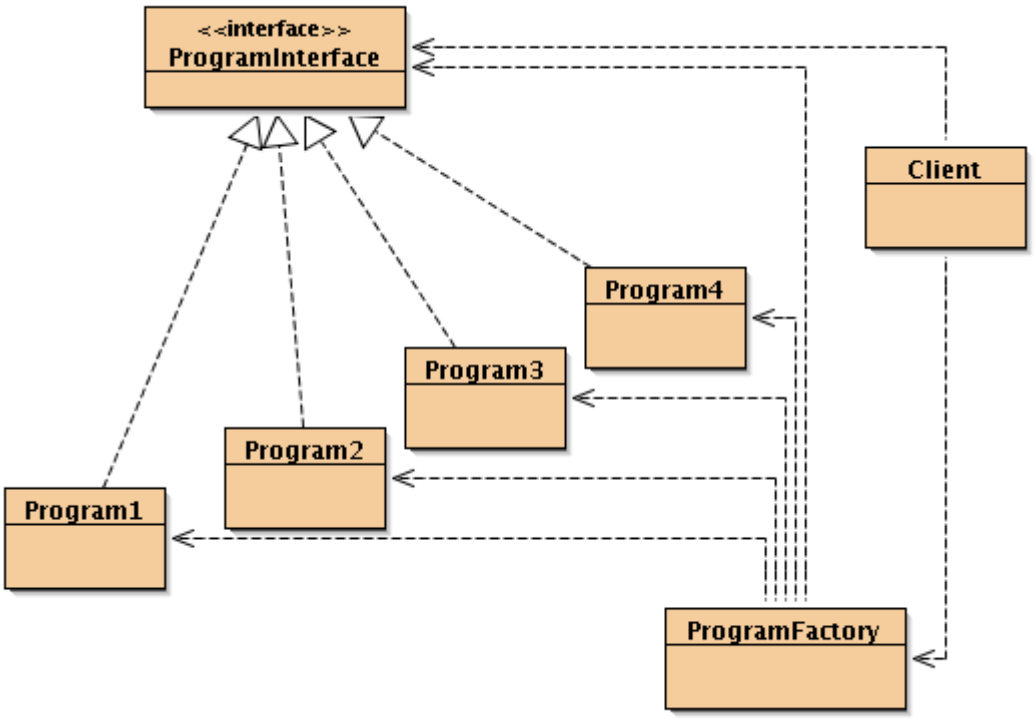
Design Pattern – Corrigé TD n°2

Ce corrigé se contente de résumer ce qui a été dit en TP. Lire simplement ce corrigé sans être venu en TP, ou sans rien avoir compris à ce qui s'est fait en TP, ne servira pas à grand chose. Le principal but de ce corrigé est de vous permettre de comparer votre code au corrigé (qui NE FAIT PAS référence). Les explications dans ce présent corrigé restent très succinctes et ne remplacent pas celles données en cours ou en TP.

LE PATTERN FACTORY

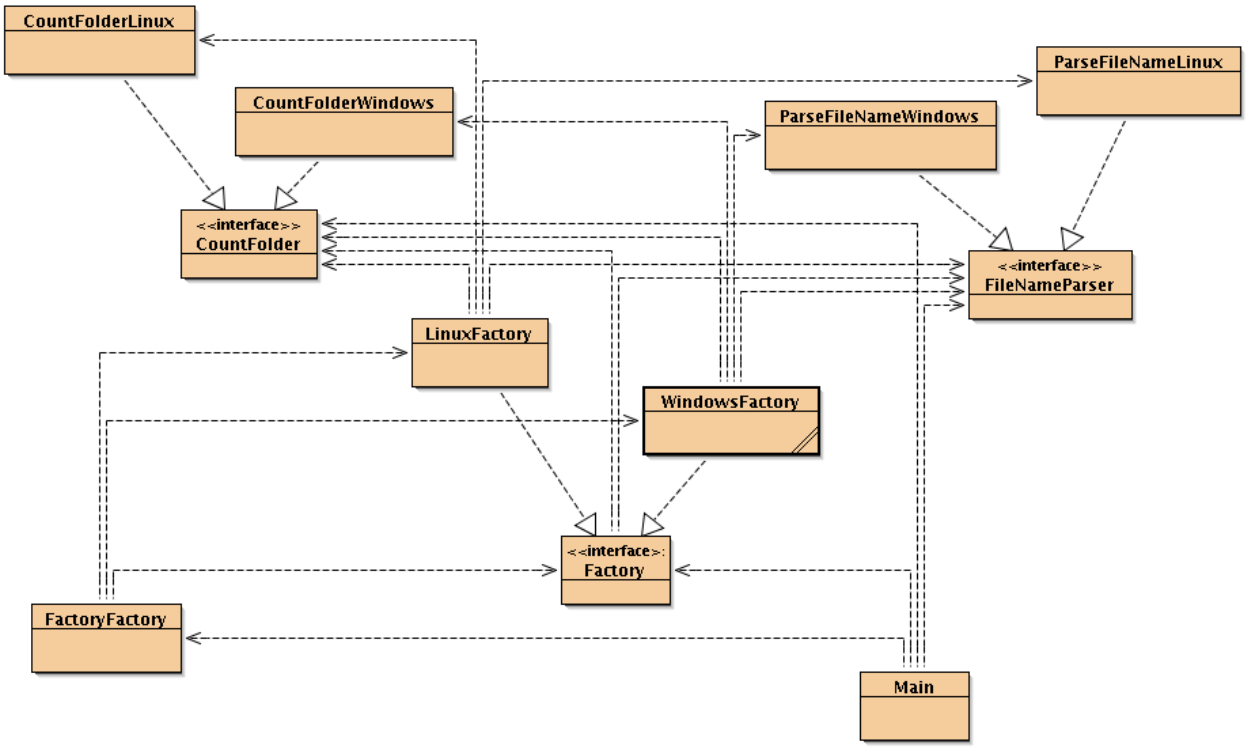
A la fin de l'exercice, vous devriez vous retrouver avec le diagramme de classe suivant.



La *ProgramFactory* est notre usine de création des *Program*. En regardant le diagramme de classe, vous vous rendez bien compte que le client n'a aucune idée de l'existence des *Program* : il « voit » seulement la *Factory* ainsi que le *ProgramInterface*.

> Télécharger le code source <

LE PATTERN ABSTRACT FACTORY



A la page précédente, vous avez le diagramme de classe que vous seriez sensé obtenir à la fin de la question 4 : on a deux factory, une *LinuxFactory* et une *WindowsFactory* qui permettent de créer des « ensembles » de programmes (*FileNameParser* et *CountFolder*) pour Windows ou Linux. Ceci est le pattern abstract factory, qui consiste à créer une *Factory* abstraite qui est ensuite implémentée par différentes factory concrètes.

La *FactoryFactory*, qui est une factory de factory est présente aussi : cette dernière ne fait pas partir du pattern abstract factory, mais permet, comme vous le voyez sur le diagramme de classe, de rendre le client (le *Main*) complètement insensible aux implémentations de nos interface. Le client ne voit que la *FactoryFactory*, et nos trois interfaces.

> Télécharger le code source <