Conception Object

DIP : *Principe d'inversion des dépendances*

Alain Plantec

UBO

17 décembre 2012

Principe d'inversion des dépendances

Énoncé

Les modules de haut niveau ne doivent pas dépendre de modules de bas niveau. Tous deux doivent dépendre d'abstractions.
Les abstractions ne doivent pas dépendre de détails. Les détails doivent dépendre d'abstractions.

- Ce principe complète le LSP.
- Il faut concevoir des abstractions et les spécialiser ensuite.
- Principe utile pour utiliser le LSP afin de respecter l'OCP.

Exemple

Quel problème potentiel dans cet exemple ?

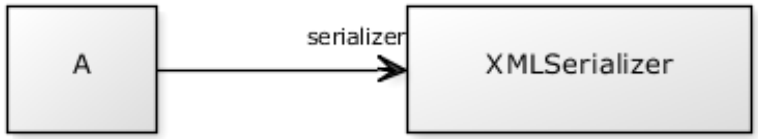
```
class A {
   XMLSerializer xmls;
   public void serialize() {
      xmls.serialize(this);
   }
}
class XMLSerializer {
   ...
}
```

Réponse :

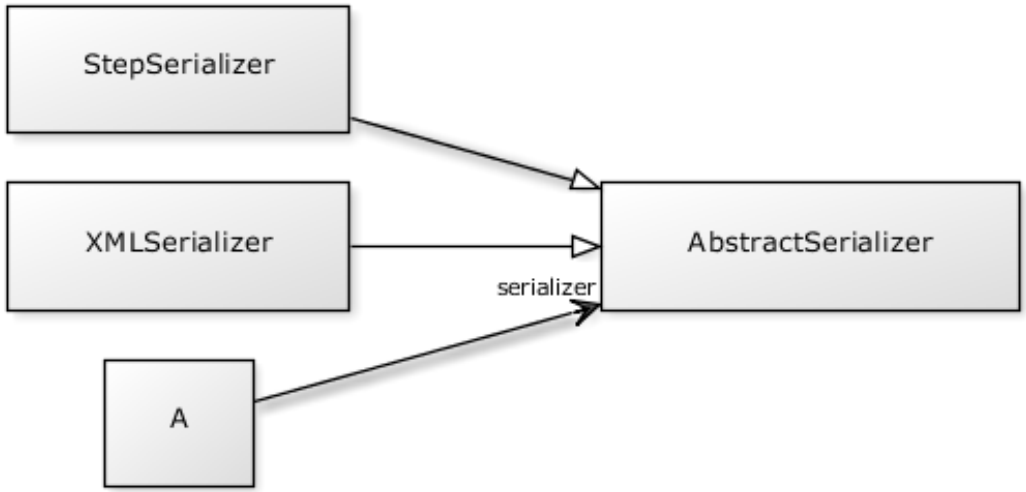
- On a figé une association avec *XMLSerializer* (couplage fort)
- La classe A dépend de XMLSerializer
- Comment faire pour utiliser un autre sérialiseur ?

Exemple

On a :

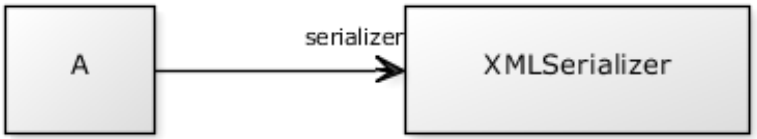


Pourquoi est-il préférable d'avoir la conception suivante ?



Exemple

XMLSerializer est une mise en œuvre concrète de sérialiseur pour XML. A dépend donc d'un module de bas niveau.



Ici, A dépend d'une abstraction, il est possible d'étendre à d'autres sérialiseurs (Par exemple STEPSerializer).