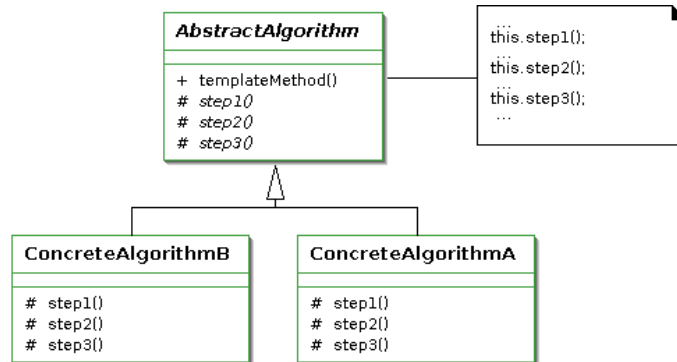


## Design Pattern : template method

### Intent

Define the skeleton of an algorithm in an operation, deferring some steps to subclasses. Template Method lets subclasses redefine certain steps of an algorithm without changing the algorithm's structure.

### Structure



### Eléments caractéristiques

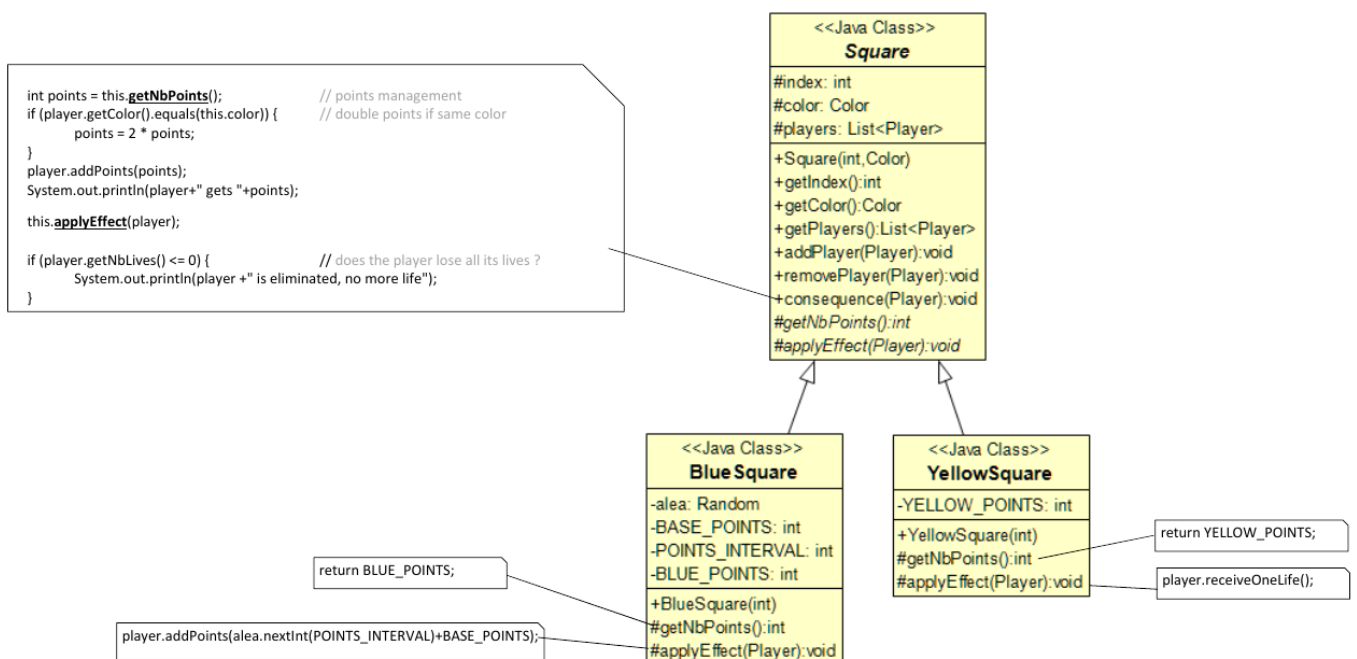
- une méthode décrivant un algorithme qui fait appel à des méthodes abstraites de la classe, ces méthodes sont définies dans les sous-classes, permettant à ces sous-classes de faire varier le comportement de l'algorithme.

**Remarque :** il peut être pertinent de déclarer `final` la *template method* afin de garantir que l'algorithme de base ne soit pas modifié dans les sous-classes mais simplement adapté dans les sous-classes par les méthodes des différentes étapes.

### Exemples rencontrés

#### Cours – SquareGame : gestion de la conséquence d'une case

Dans le « jeu » utilisé comme exemple en cours, pour la gestion de la conséquence de l'arrivée sur une case, la méthode `consequence()` s'appuie sur les méthodes abstraites `getNbPoints()` et `applyEffect()` qui sont spécialisées dans les sous-classes.



## TD – Transport : gestion de l'encombrement

Gestion de l'encombrement : la méthode `add()` de `Shipment` s'appuie sur les méthodes abstraites `quantityCriteria()` et `getQuantity()`.

