



INF461

PATRONS DE CONCEPTIONS



redige par

Kamga Marilane

Sous la supervision de

M.Kemtho zidane

PRESENTATION DES PATRONS DE CONSTRUCTION

- Ils ont pour vocation d'abstraire les mécanismes de création d'objets.
- Organiser la création d'objets.
- Un système qui les utilise devient indépendant de la façon dont les objets sont créés, en particulier des mécanismes d'instanciation des classes concrètes.

LES PATRONS DE CONSTRUCTION

1

FACTORY

2

FACTORY ABSTRACT

3

SINGLETON

4

PROTOTYPE

5

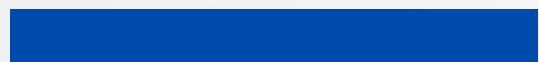
PROTOTYPE

LE PATRON **FACTORY**

Il permet de créer un objet dont le type dépend du contexte.

On l'implémente de deux manières :

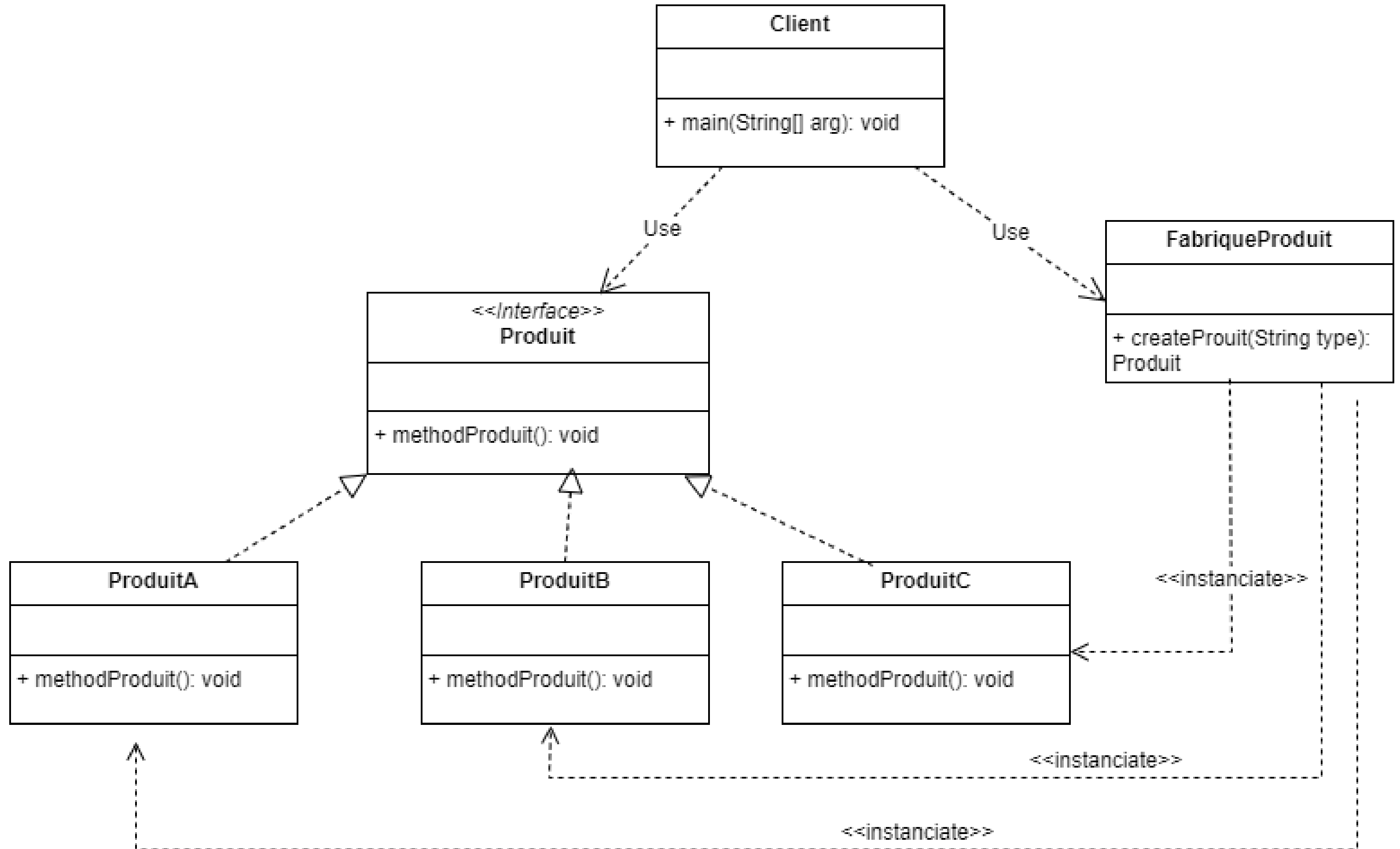
- **factory methode**
- **factory abstract**



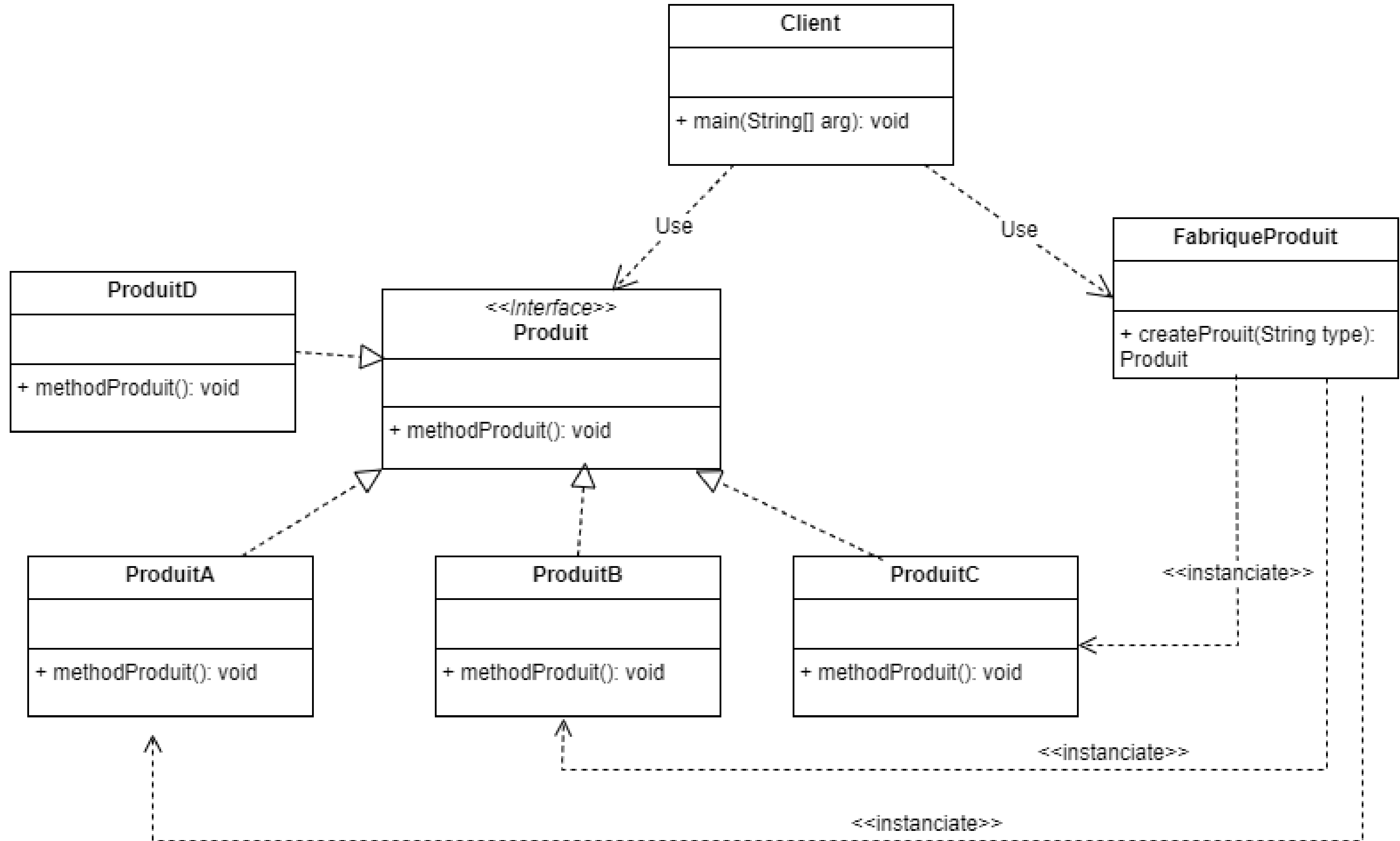
A - FACTORY METHODE

On declare une classe créatrice contenant une méthode de création. Cette méthode reçoit les informations nécessaires et décide quel type d'objet concret instancier, sans que le code client connaisse les détails.

Factory Methode cours



FACTORY METHODE MODIFIE



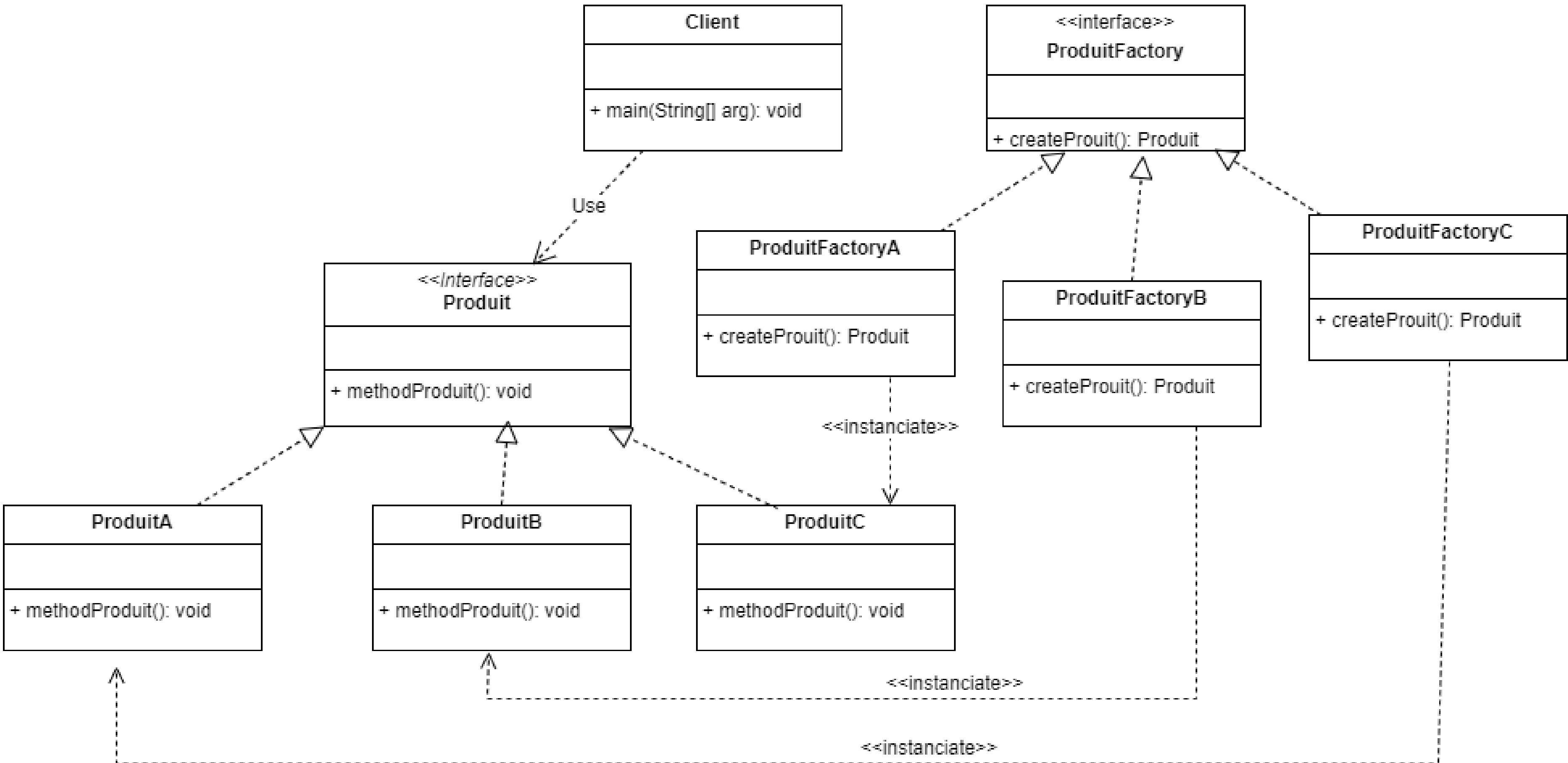
B - FACTORY ABSTRACT

Une classe définit une méthode de création, mais laisse les sous-classes décider quel objet concret construire.

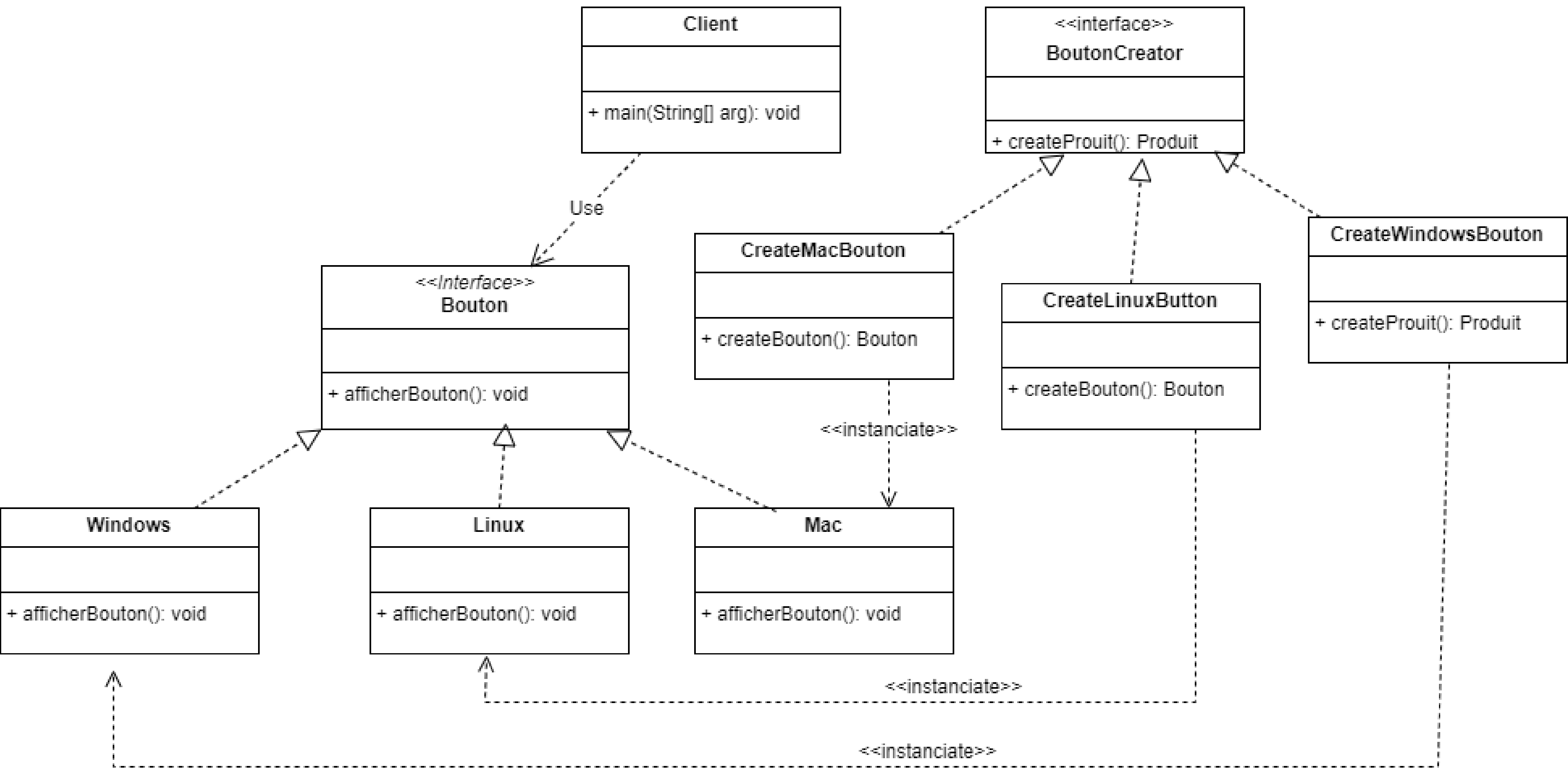
Participants

- Product : interface des produits.
- ConcreteProduct : produit concret.
- Creator : contient la factory method abstraite : `createProduct()`.
- ConcreteCreator : implémente `createProduct()` et retourne un ConcreteProduct.

FACTORY ABSTRACT COURS



FACTORY ABSTRACT MODIFIE





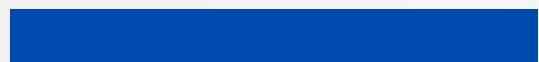
LE PATRON

ABSTRACT FACTORY

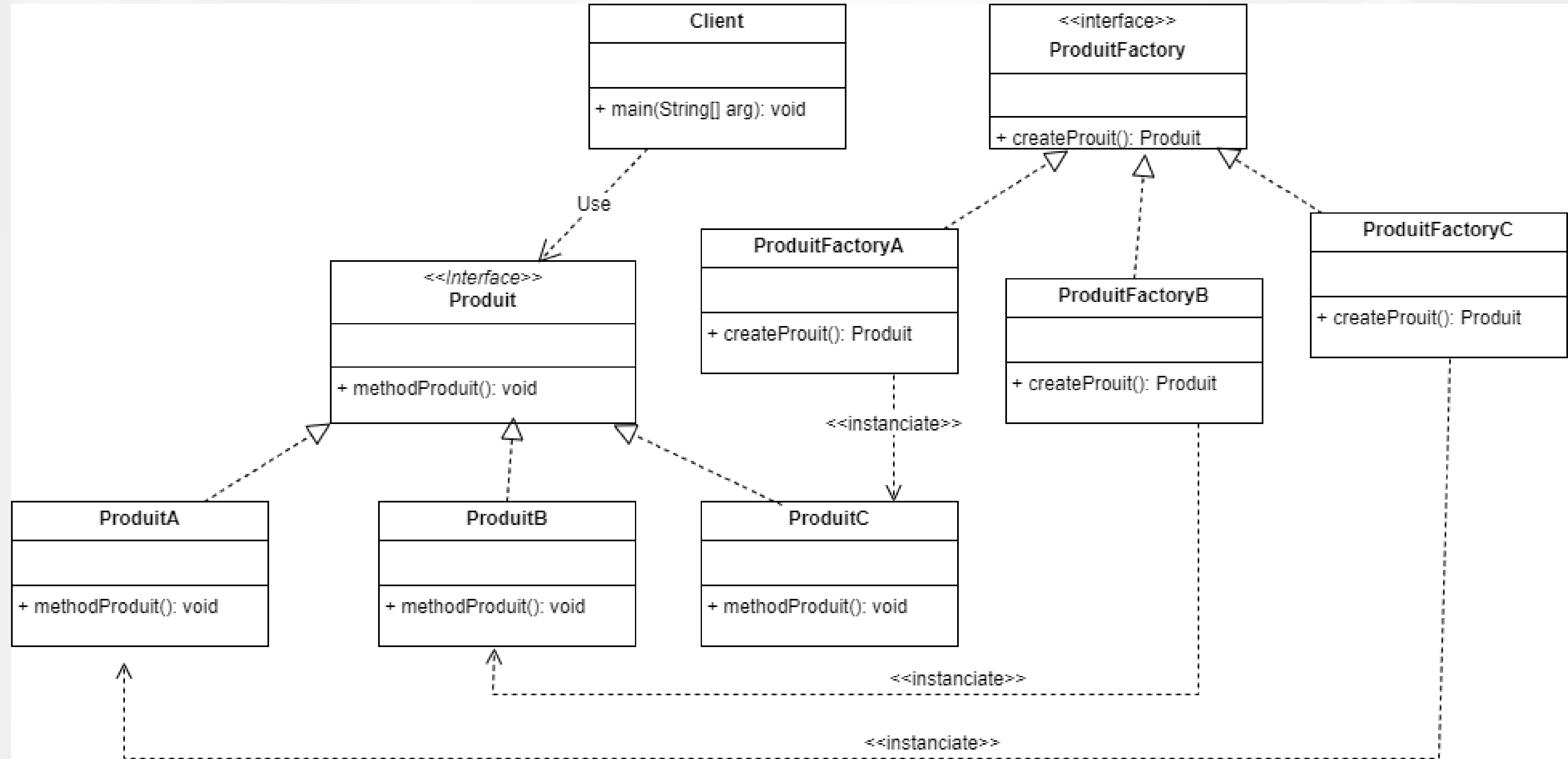
Permet de créer des familles entières d'objets compatibles entre eux .

 Participants (modèle générique)

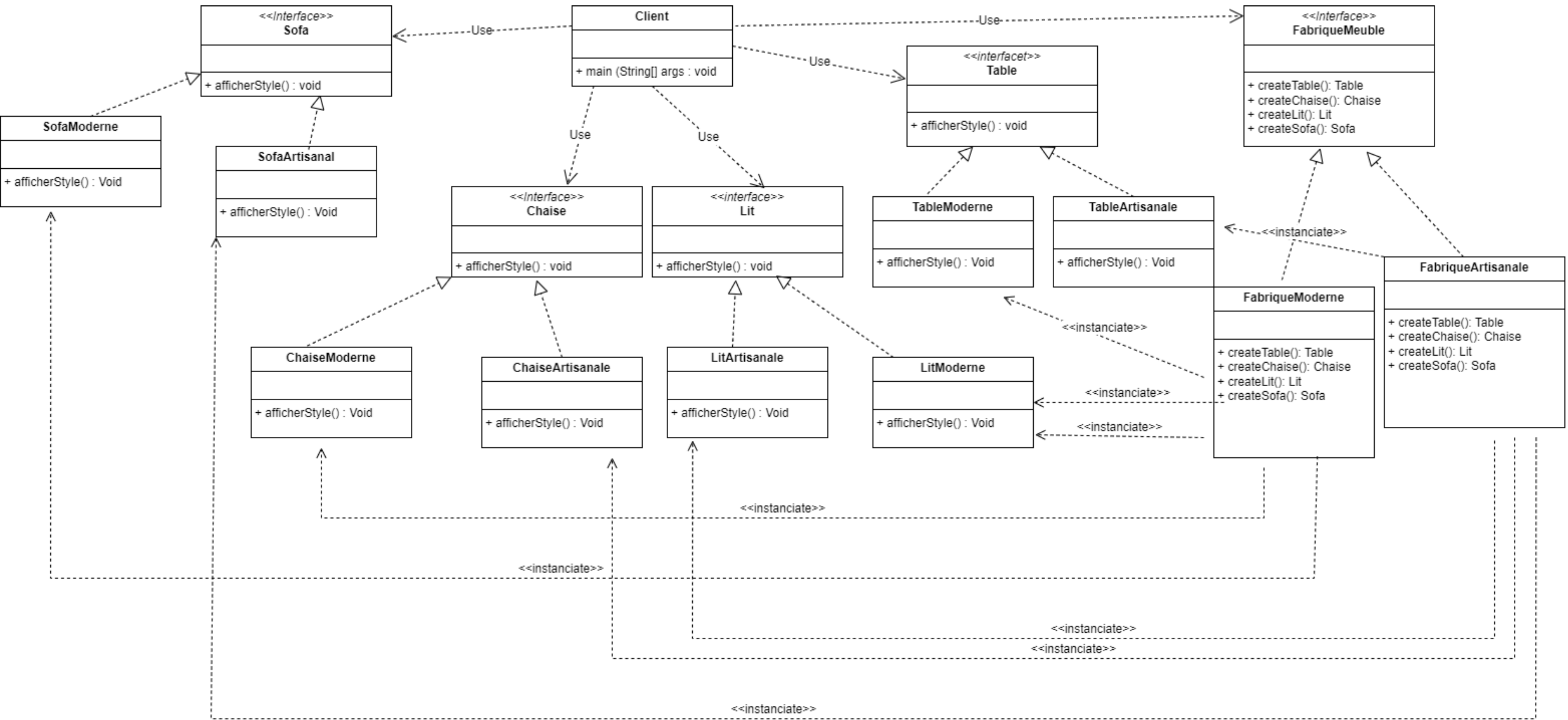
- AbstractFactory
 - interface avec plusieurs méthodes : createProductA(), createProductB(), ...
- ConcreteFactory
 - crée une famille cohérente de produits.
- AbstractProductA / AbstractProductB
 - interfaces des types de produits.
- ConcreteProductA1 / A2 / B1 / B2
 - implémentations concrètes pour chaque famille.



ABSTRACT FACTORY COURS




ABSTRACT FACTORY MODIFIE



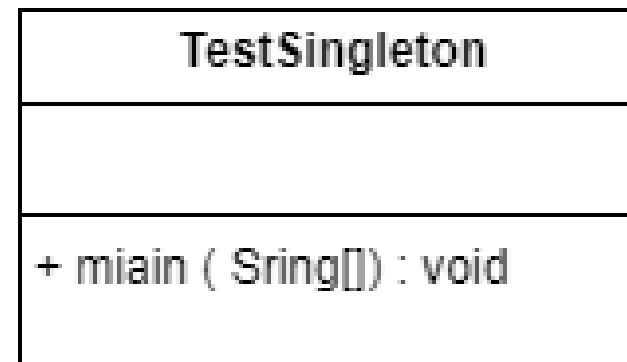
PATRON **SINGLETON**

Garantie qu'une seule instance d'une classe existe pendant toute l'exécution du programme,
et fournir un point d'accès global à cette instance.

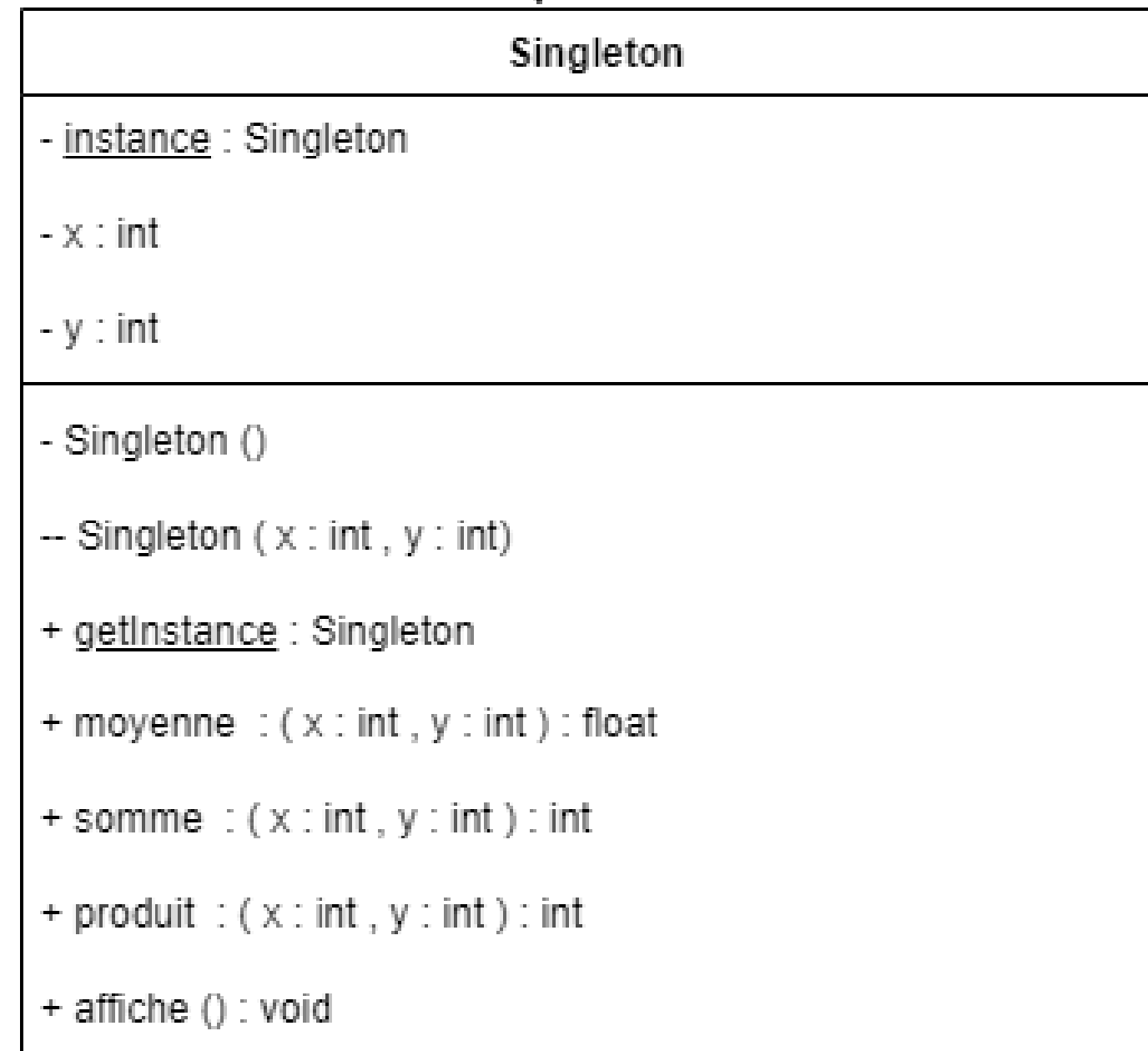
 Participants du modèle générique

- Singleton :
 - contient l'instance unique (statique)
 - possède un constructeur privé
 - fournit `getInstance()` pour récupérer l'unique objet
- Client :
 - demande l'instance via `Singleton.getInstance()`
 - utilise l'objet unique (ex. connexion BD, logger...)

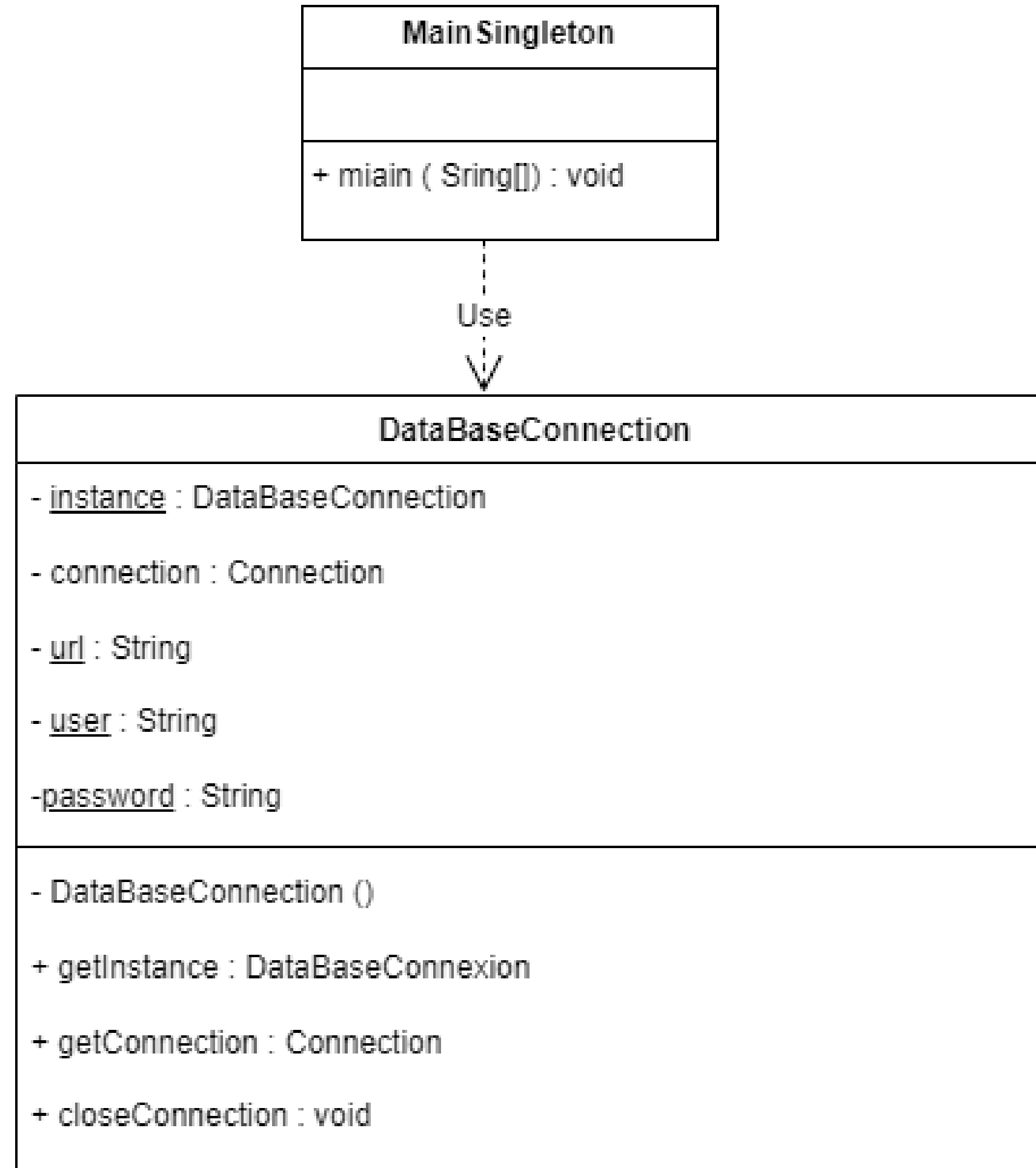
A-SINGLETON COURS



Use



B-SINGLETON MODIFIE



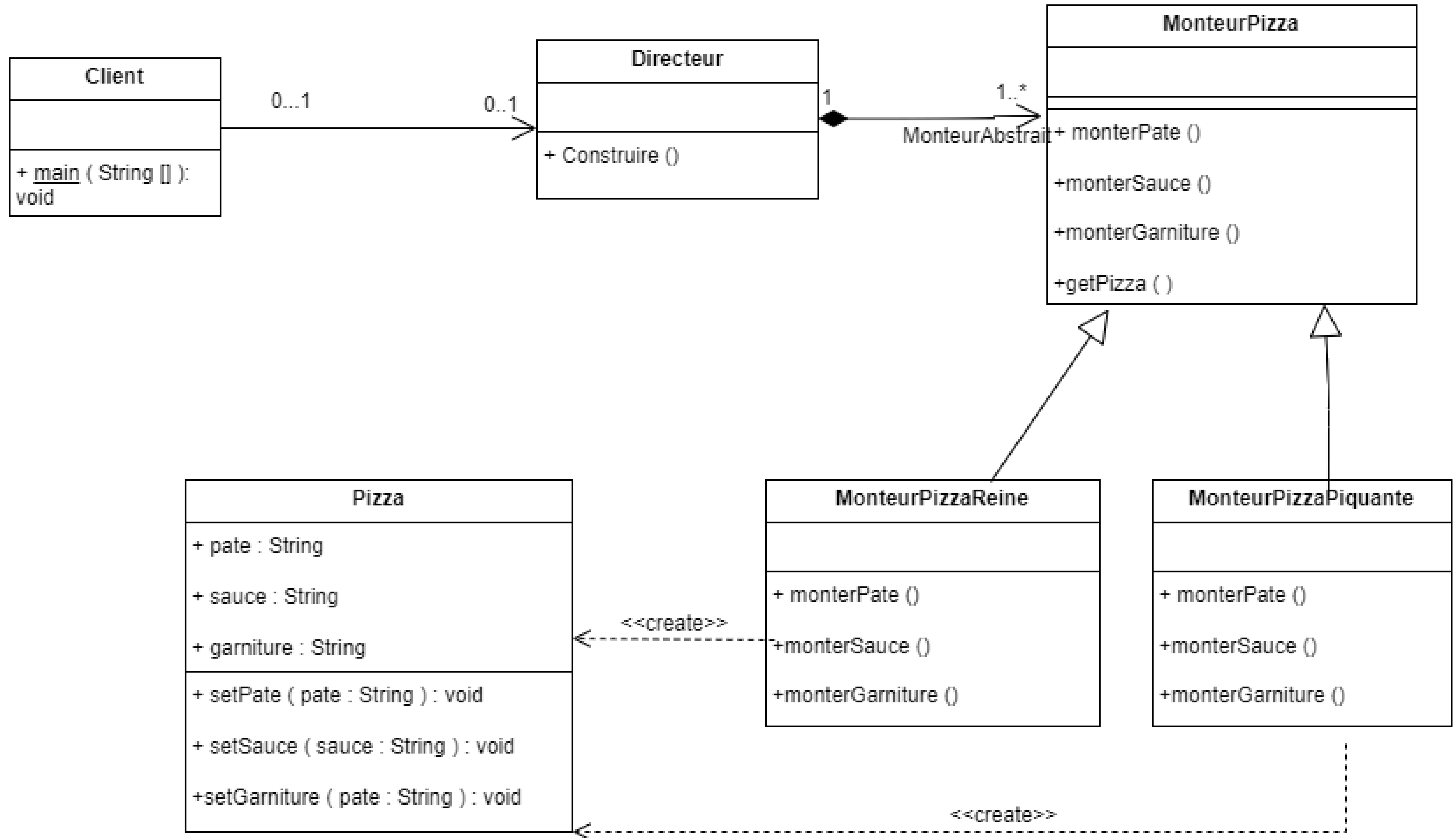
4 PATRON BUILDER

Permet de construire un objet complexe étape par étape,

 Participants du modèle générique

- Builder (interface/abstrait)
 - définit les étapes de construction
 - ex : buildPartA(), buildPartB()...
- ConcreteBuilder
 - implémente les étapes
 - assemble les éléments
 - retourne l'objet final
- Director
 - orchestre les étapes de construction dans un ordre précis
- Product
 - l'objet complexe construit
 - contient les différentes parties assemblées
- Client
 - utilise le directeur pour construire des objets

A-BUILDER COURS



B-BUILDER MODIFIE

