

Design Patterns

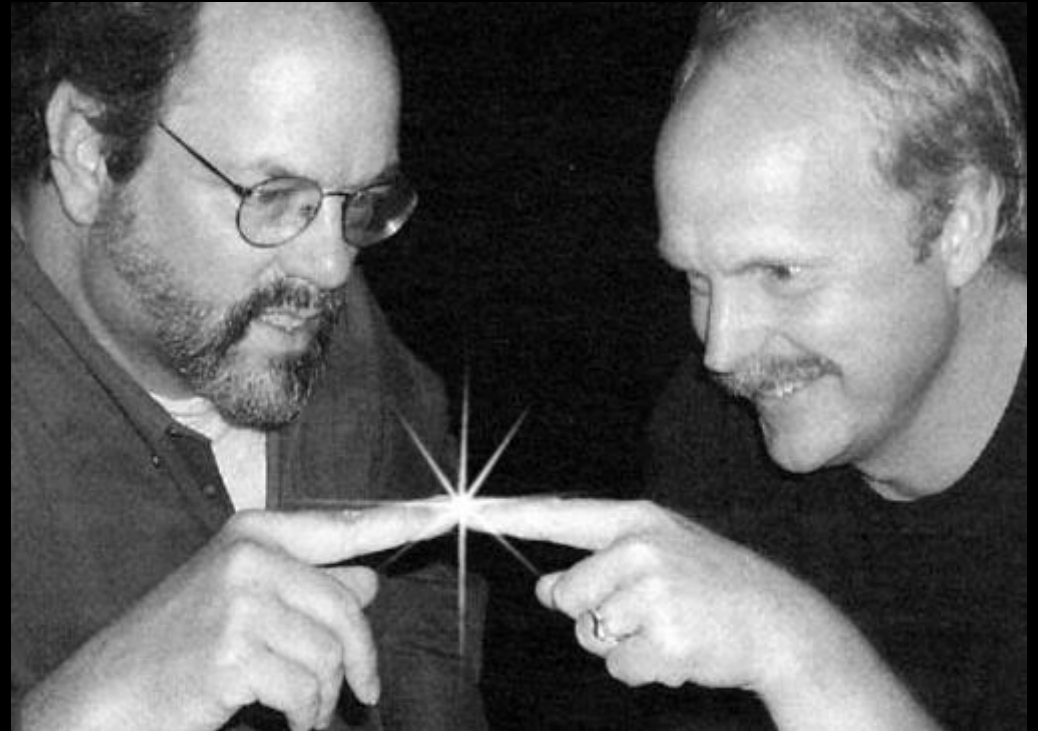
DÉCORATEUR

FAUCHER Loreena / BANET Xavier /
CHERIF Farès / PRODEL Pierre

D'OÙ VIENNENT LES PATTERNS ?



Christopher Alexander



Ward Cunningham et Kent Beck

QU'EST-CE QU'UN PATTERN ?



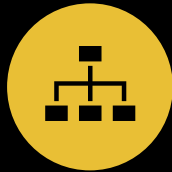
Un patron de conception ou design pattern, est une structure de classe qui utilise des interfaces.

Elle apporte une solution à un problème récurrent d'architecture logiciel.

QUELS SONT LES DIFFÉRENTS PATTERNS ?



Les patterns de création qui répondent aux problèmes concernant la création et la configuration d'objet.



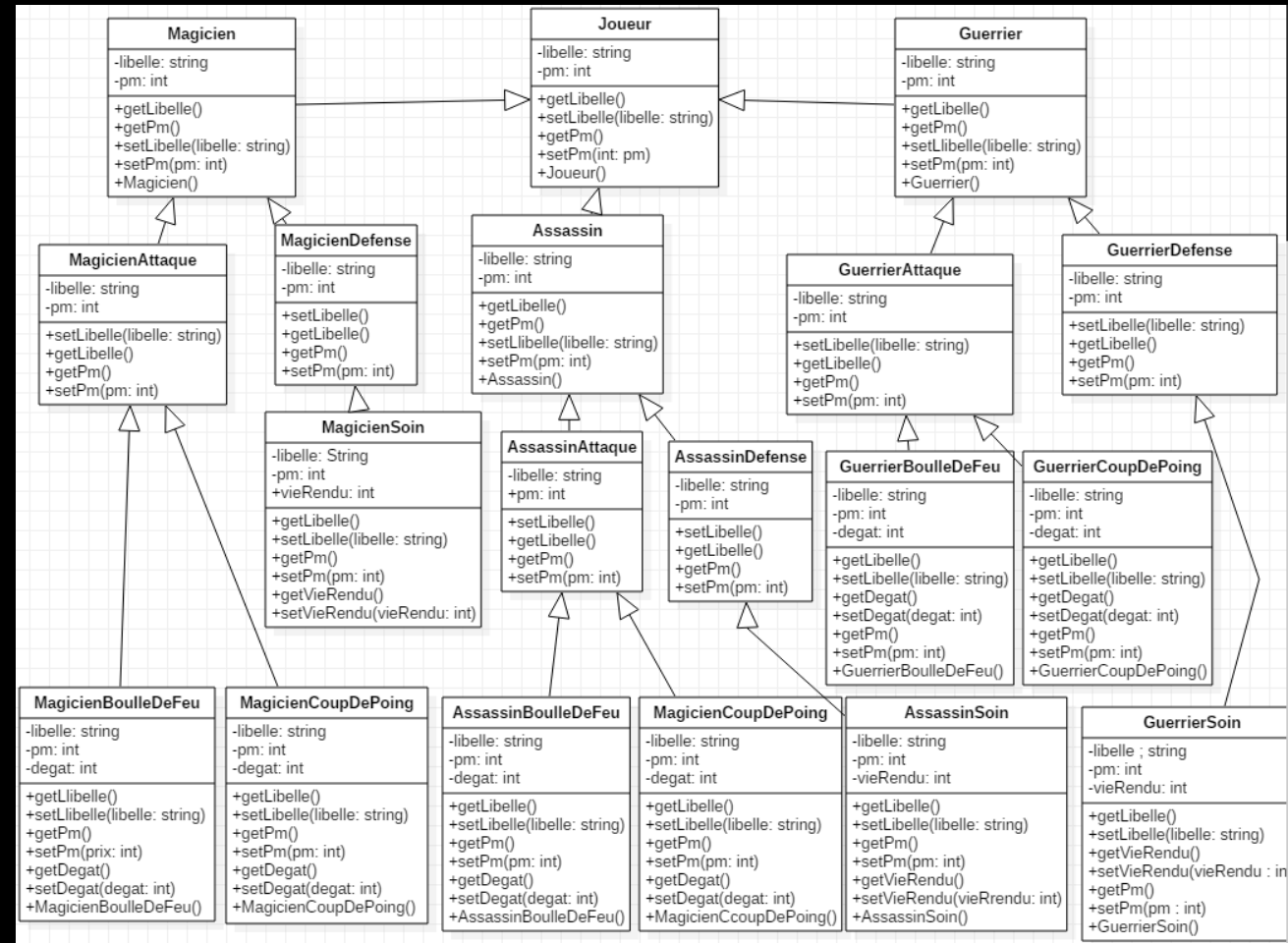
Les patterns de structure qui répondent aux problèmes sur la structure des classes et leurs interfaces



Les patterns de comportement eux répondent aux problèmes d'interactions entre les classes.

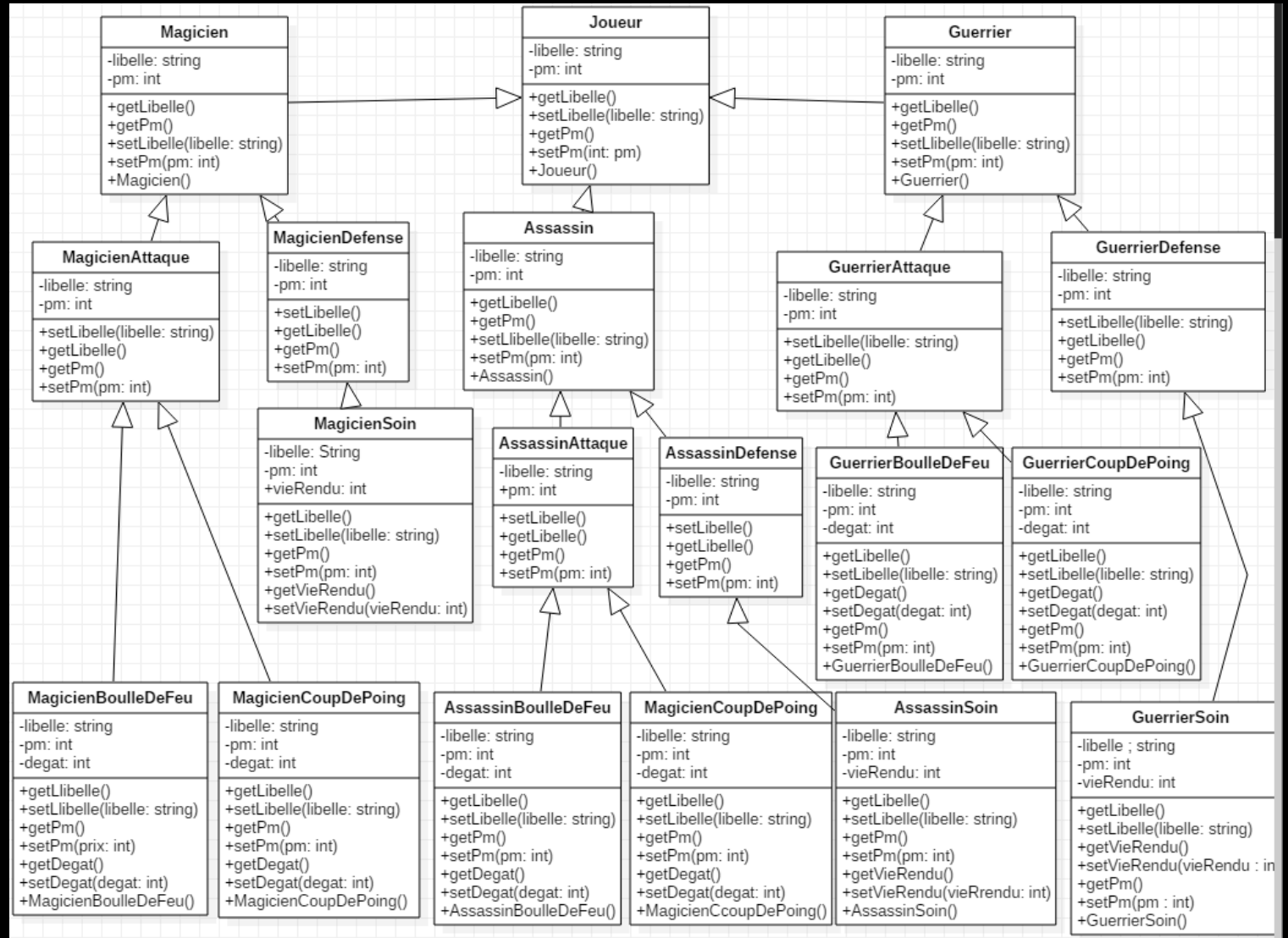
UN EXEMPLE CONCRET : LA CRÉATION DE PERSONNAGE

Une solution : l'héritage



Les problèmes de l'héritage

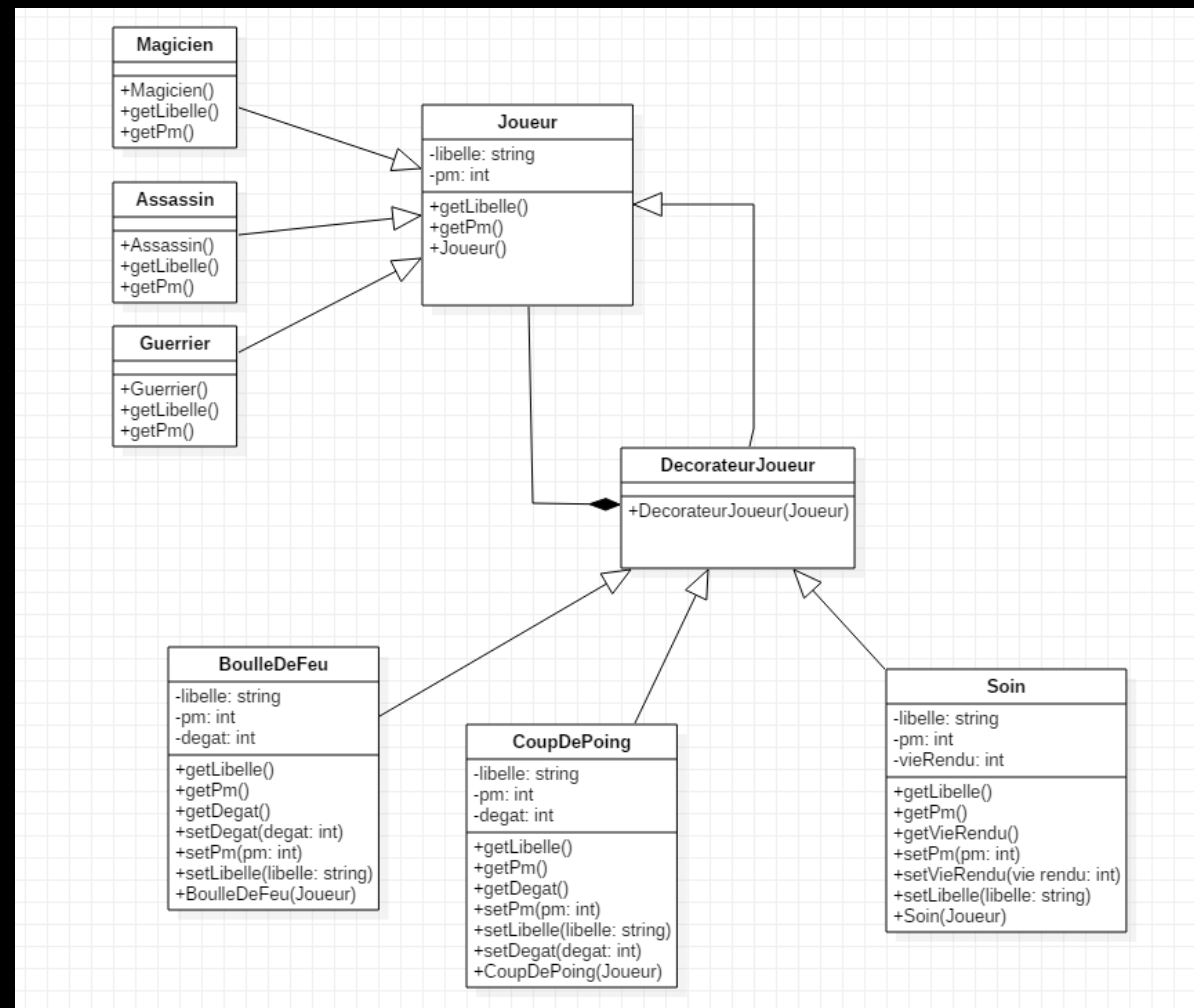
- redéfinition de fonction difficile
- Système complexe et confus
- modification impossible après compilation
- nombreuses redondances
- besoin d'imaginer tous les cas possibles



UNE AUTRE SOLUTION

Introduire le supplément dans un autre objet

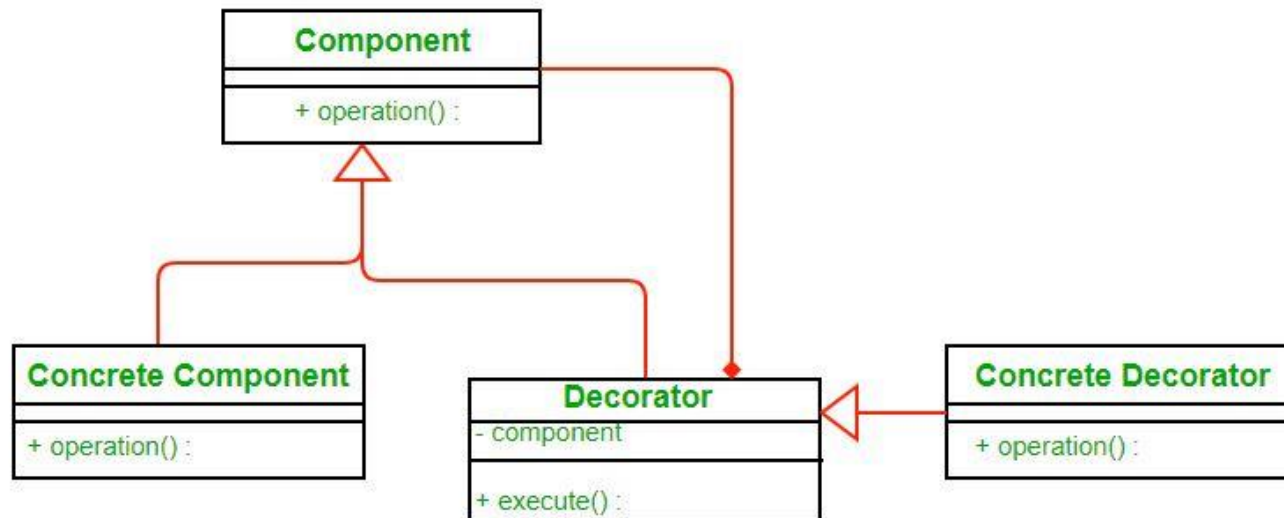
| Héritage | Nouvelle solution |
|--|--|
| - Système complexe et confus | - Système clair et lisible |
| redéfinition de fonction difficile | Redéfinition simple |
| modification impossible après compilation | des fonctionnalités peuvent être ajoutées de manière dynamique |
| - nombreuses redondances | Aucune redondance |
| - besoin d'imaginer tous les cas possibles | - seulement besoin de savoir les ajouts |



LA NOUVELLE SOLUTION : LE PATTERN DÉCORATEUR

Autres avantages du décorateur :

- le code est bien plus facile à faire évoluer avec un décorateur



LE PATTERN DÉCORATEUR : UNE SOLUTION PARFAITE ?

Inconvénient :

- Multiplicité des objets à créer

Avantage :

- Système clair et lisible
- Redéfinition simple
- des fonctionnalités peuvent être ajoutées de manière dynamique
- Aucune redondance
- seulement besoin de savoir les ajouts
- le code est bien plus facile à faire évoluer avec un décorateur

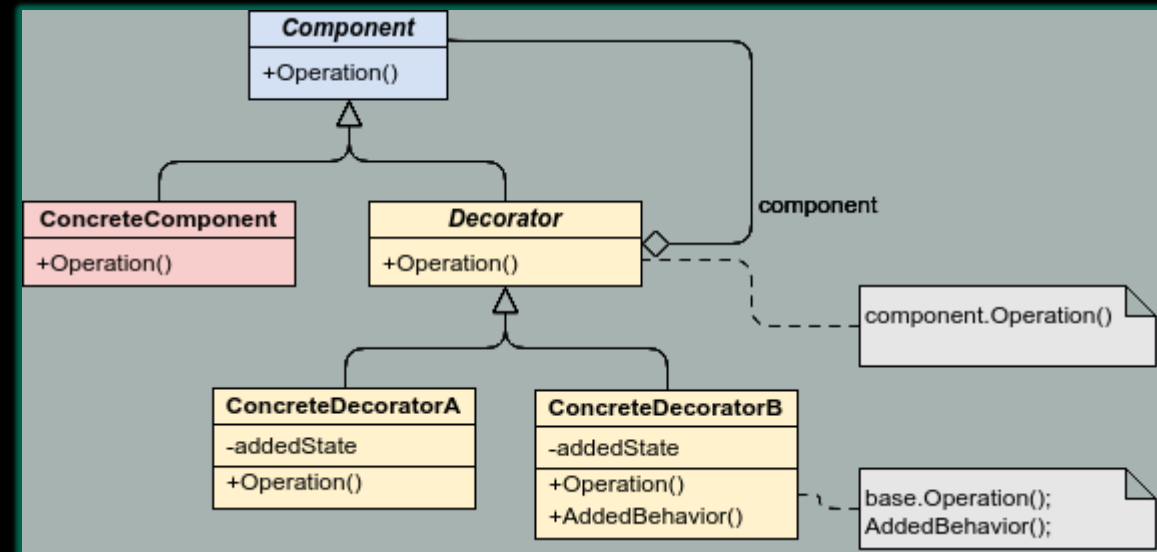
À QUOI SERT DONC CE PATTERN ?

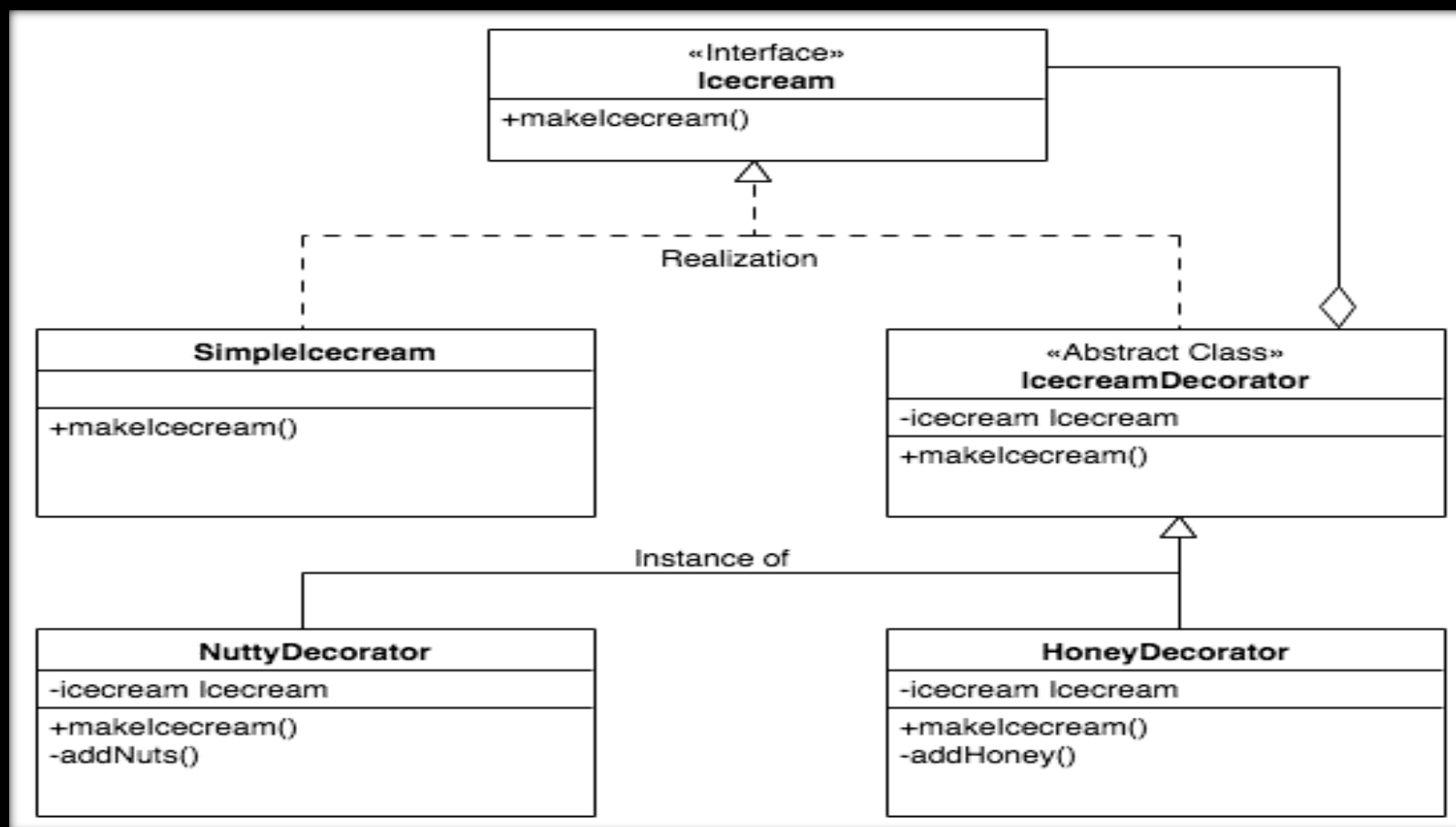


Erich Gamma, Richard Helm, Ralph Johnson et John Vlissides



Design Patterns:
Elements of Reusable
Object-Oriented
Software



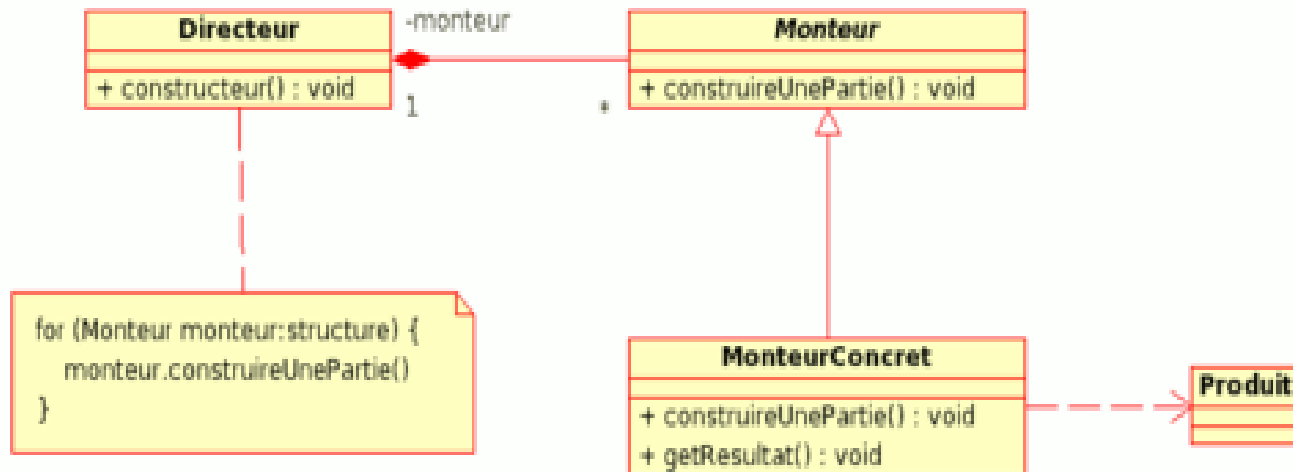


RAPPEL SOLID

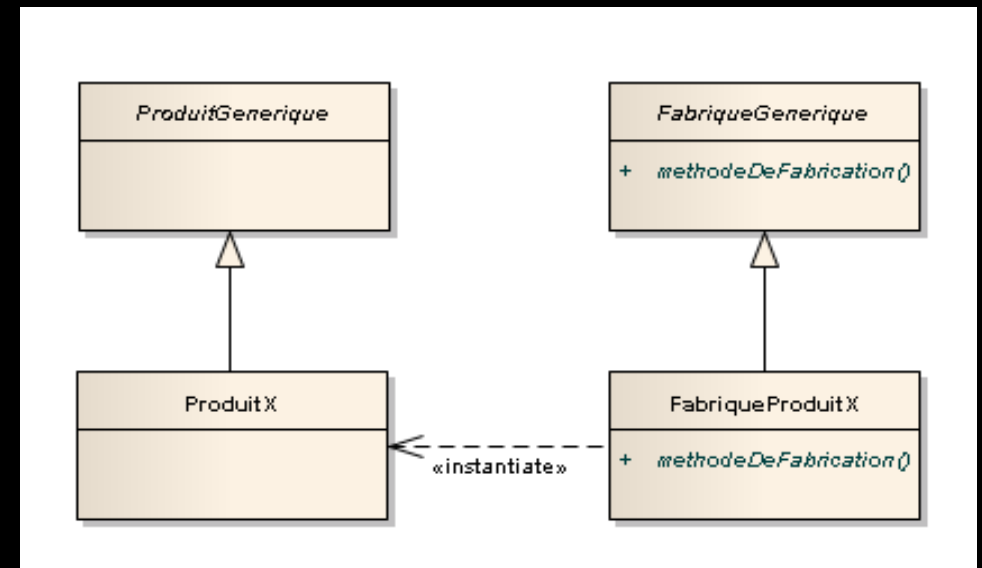
- *single responsibility principle*
- *open/closed principle*
- *Liskov substitution principle*
- *interface segregation principle*
- *dependency inversion principle*

LES LIMITES/SOLUTIONS

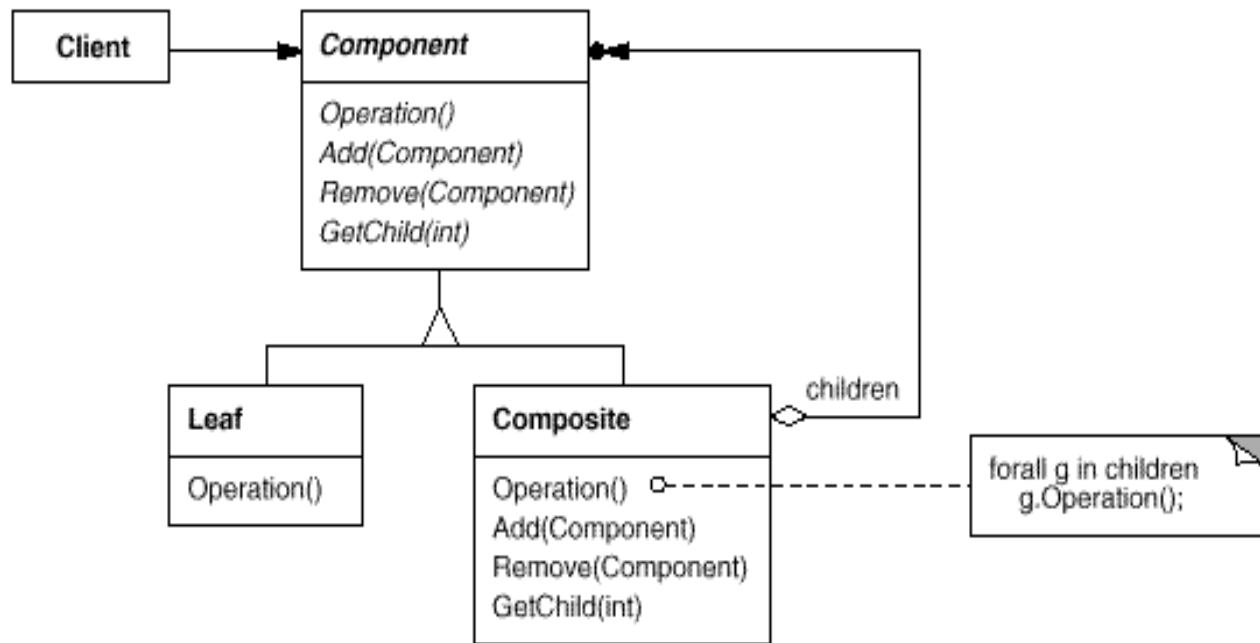
Monteur



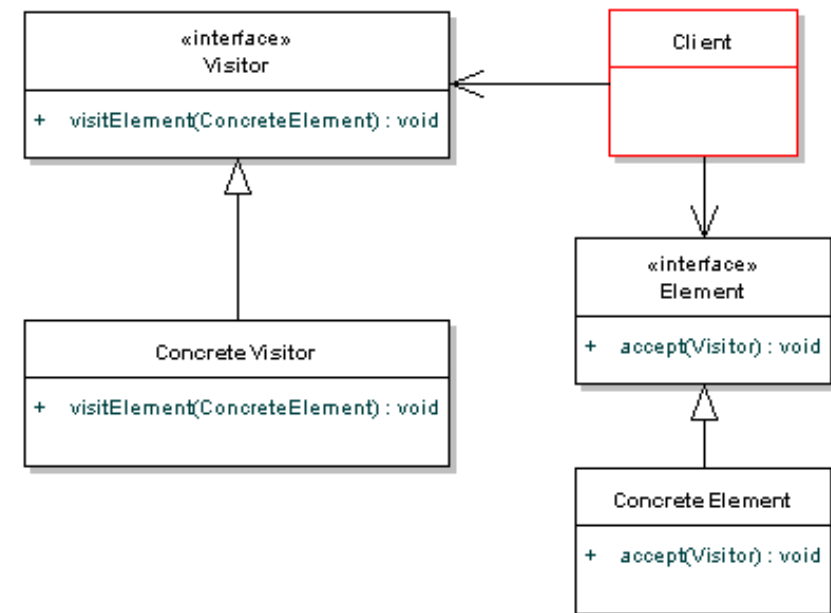
Fabrique



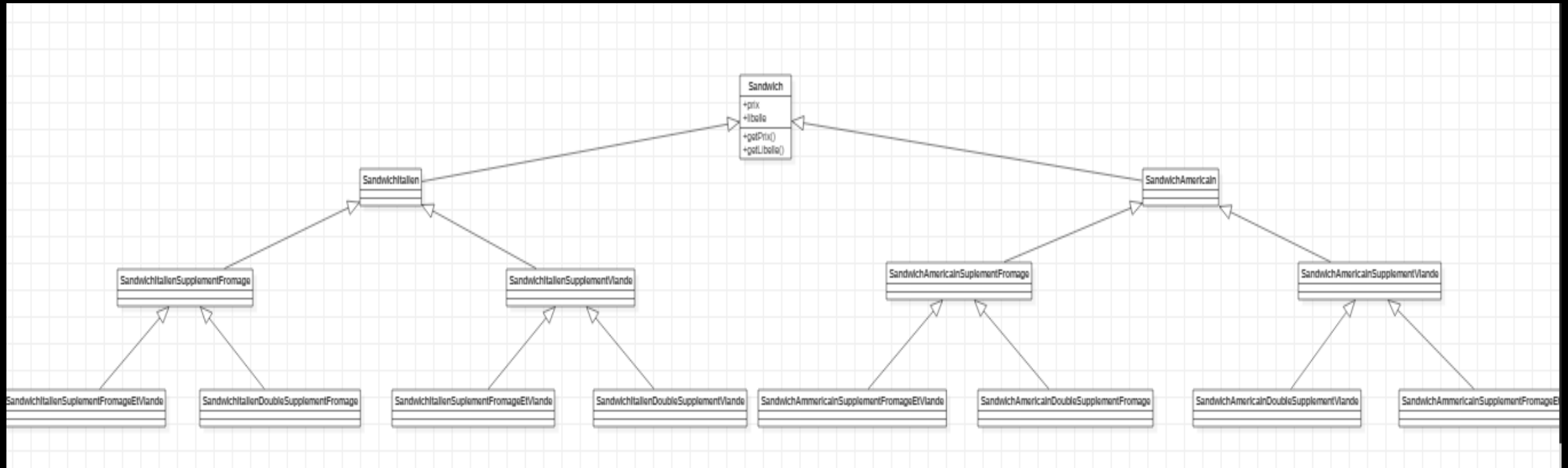
Composite



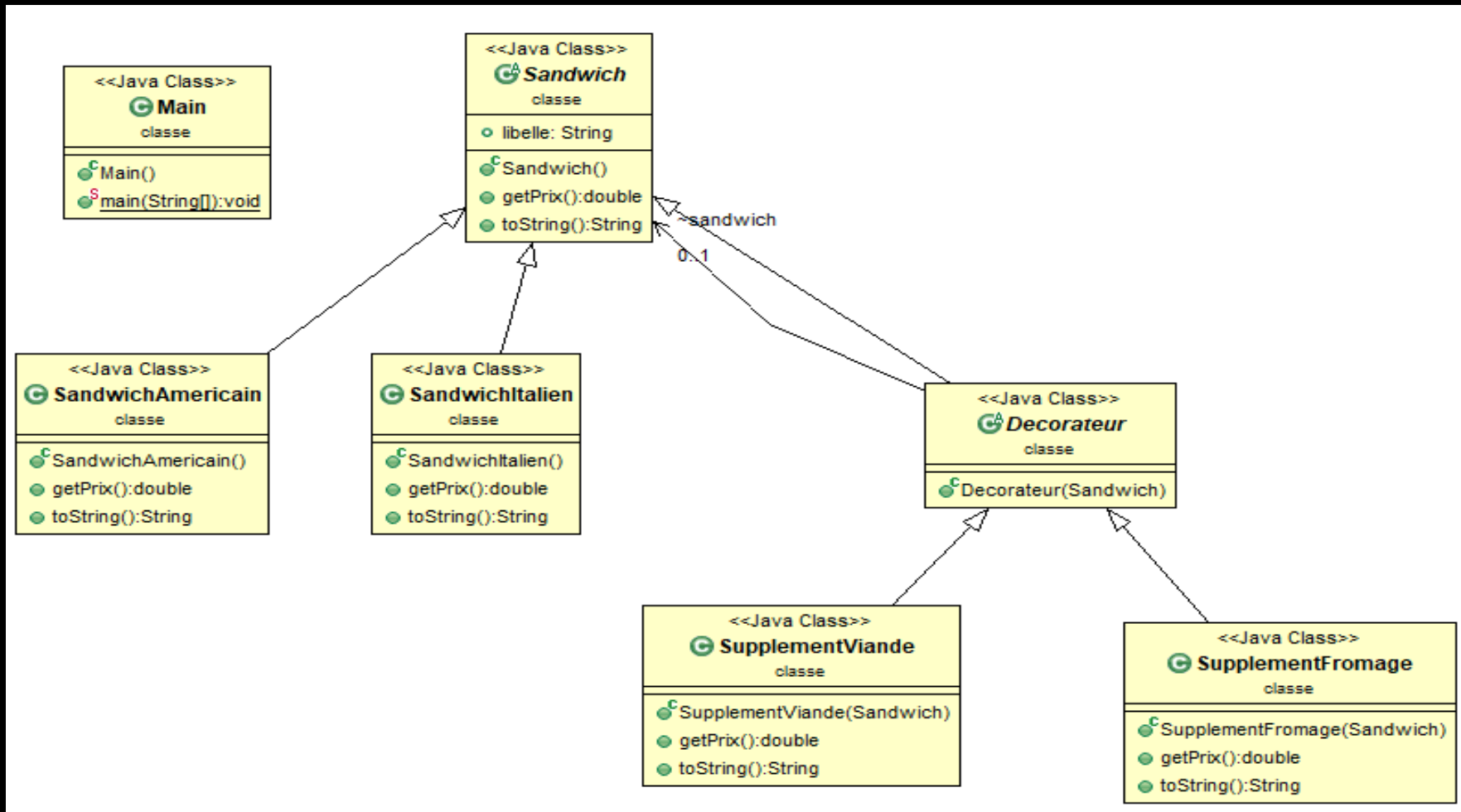
Visiteur



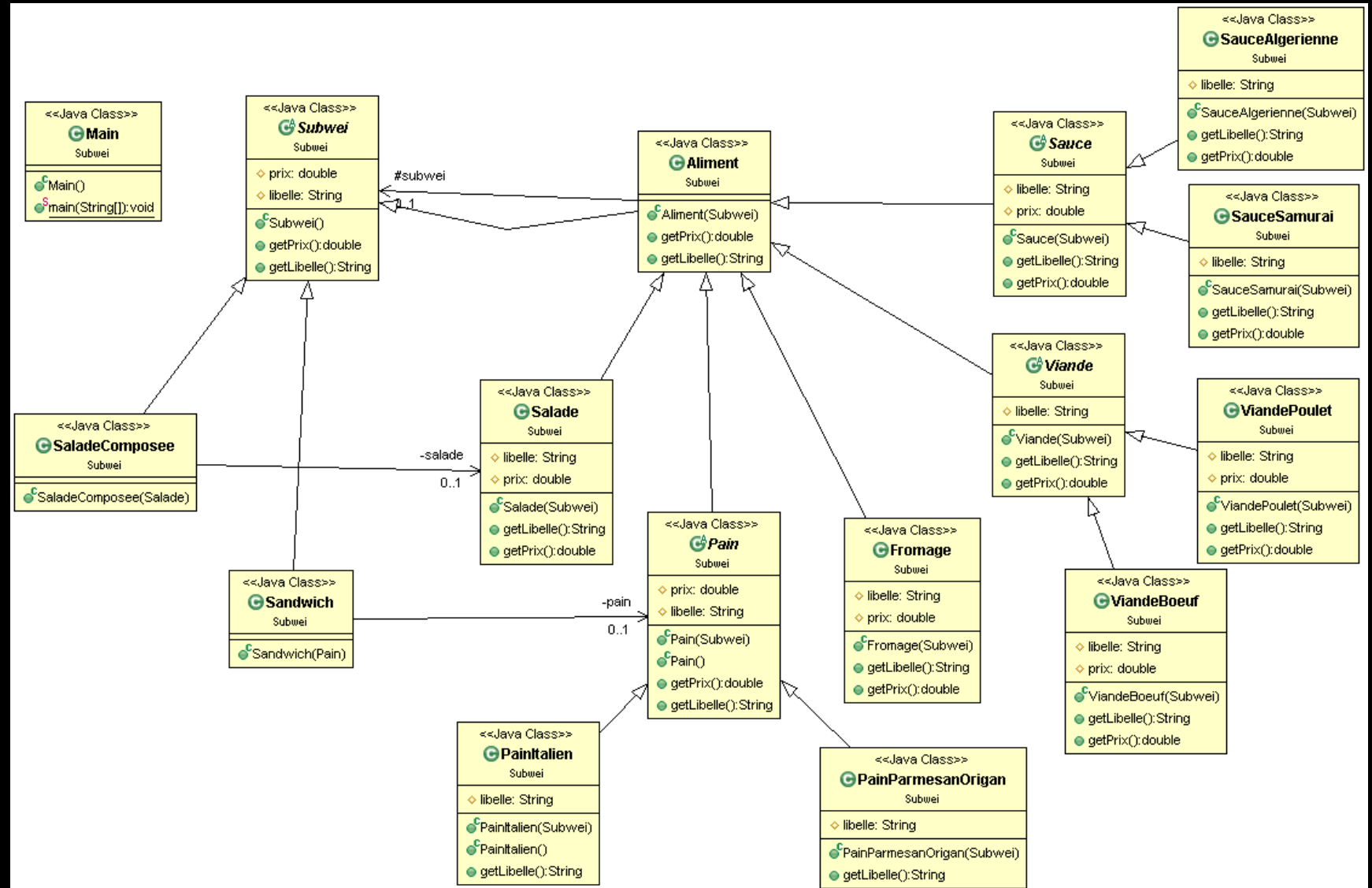
Sans décorateur



Avec décorateur



EXEMPLE SUBWEI :



SITOGRAFIE :

Medium.com :

<https://medium.com/@armandfardeau/quest-ce-qu-un-design-pattern-cac63a3fa642>

Elaio.com :

<https://blog.elao.com/fr/dev/design-pattern-decorator/>

Cellenza.com :

<https://blog.cellenza.com/archi-patterns-bp/le-design-pattern-decorator-decorateur/>

Wikipédia :

[https://fr.wikipedia.org/wiki/Décorateur \(patron de conception\)](https://fr.wikipedia.org/wiki/D%C3%A9corateur_(patron_de_conception))

Wikibooks :

[https://fr.wikibooks.org/wiki/Patrons de conception](https://fr.wikibooks.org/wiki/Patrons_de_conception)

Efreidoc :

<http://www.efreidoc.fr/M1/COO%20avanc%C3%A9/Cours/20XX-XX/20XX-XX.cours.design-patterns.coo.pdf>

Design-patterns.fr :


<http://design-patterns.fr/decorateur-en-java>

Gaudry.be :

<http://www.gaudry.be/pattern-decorator.html>



QCM :



QUELS SONT LES ATTRIBUTS OBLIGATOIRE DE LA CLASSE DÉCORATEUR ?

1. Les mêmes attributs que la classe dont elle hérite.
2. Un objet du type de la classe dont elle hérite.
3. Les attributs ajoutés par les suppléments.



QU'EST-CE QU'UN PATTERN DE PROGRAMMATION?

1. Une sorte de moule pour créer des objets.
2. Un motifs abstrait pour faire un nuage de mot réussi.
3. Une structure abstraite à appliquer à notre code afin de traiter un problème récurrent.



A QUOI SERT LE PATTERN MONTEUR ?

1. À diminuer le nombre d'objet créés par le pattern décorateur.
2. À augmenter le nombre de classe que l'on peut implémenter.
3. A modifier une méthode.



A QUOI SERT UN PATTERN DÉCORATEUR ?

1. À pouvoir ajouter des fonctions dynamiquement.
2. À rajouter des fioritures dans son code.
3. À rajouter des couleurs sur les lignes de code.



QUEL EST L'UN DES GROS PROBLÈMES DU PATTERN DÉCORATEUR ?

1. La longueur du code à écrire
2. La multiplicité des objets à créer
3. La complexité du code