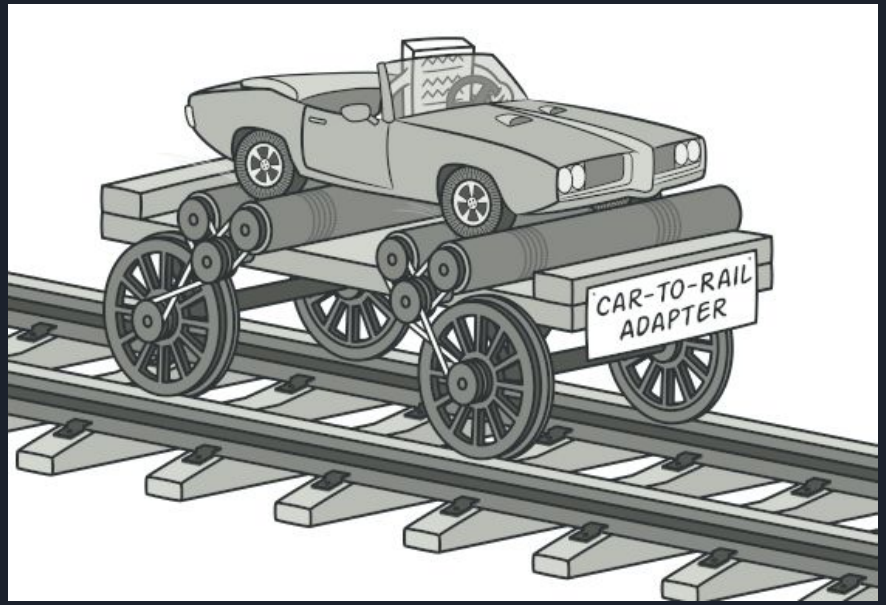
Adapter (Design Pattern)

KAING Thierry

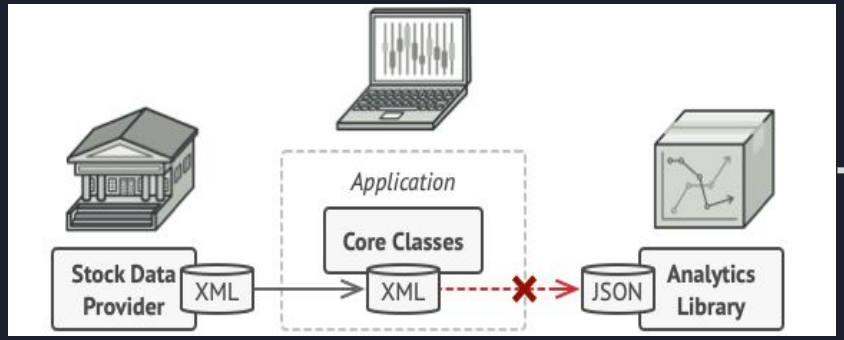
MENARD Dorian

Présentation
Adapter

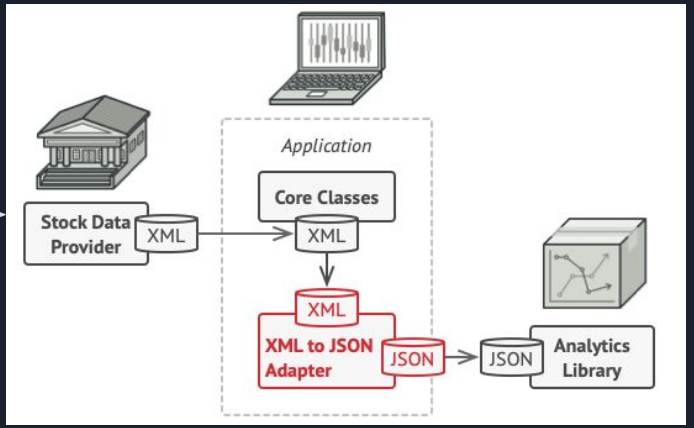


Le design pattern adapter est un design pattern qui permet de faire collaborer et communiquer des interfaces/class qui de base sont incompatible.

Problématique
Adapter



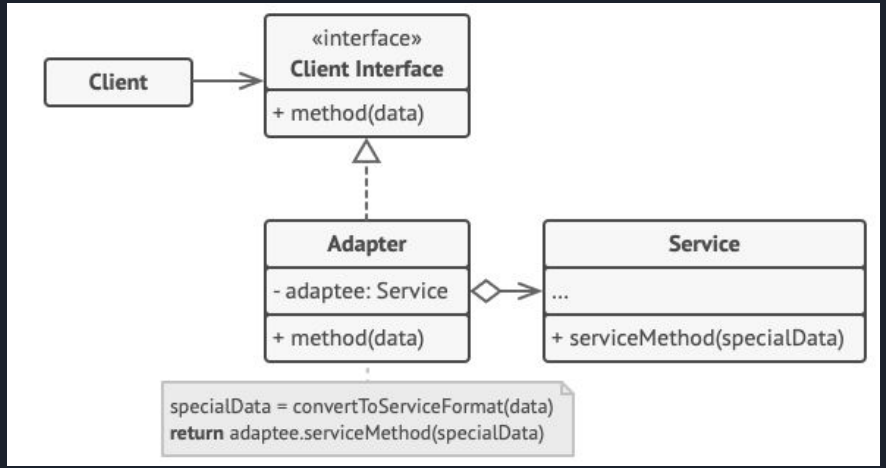
Adapter



Solution Object Adapter

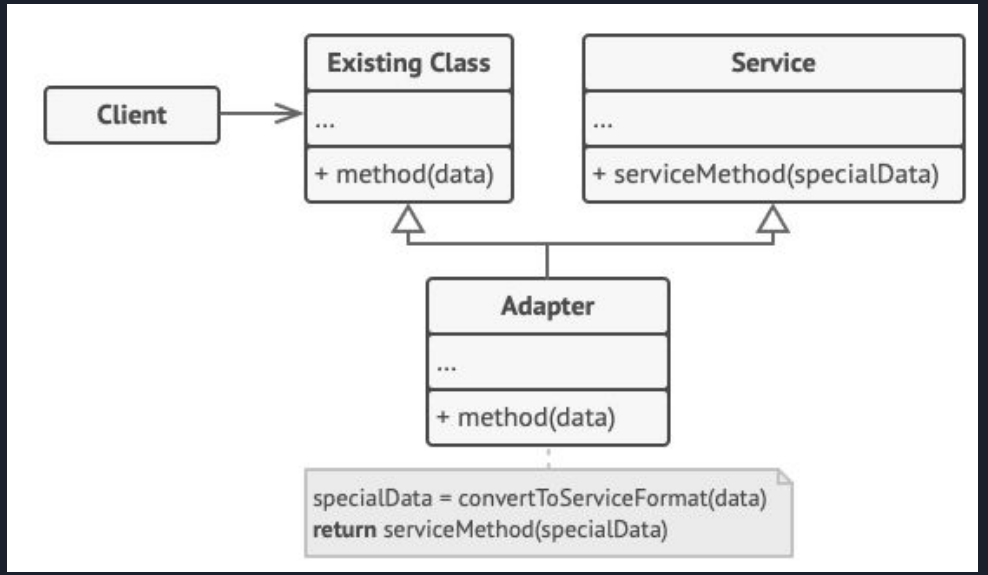
Adapter

Dans cette implémentation, l'Adapter implémente l'Interface d'un objet et encapsule le deuxième.



Solution Class Adapter
Adapter

- Cette implémentation utilise l'héritage. L'adapter hérite des deux objets en même temps.
- Implémentation utilisable quand dans des langages qui ont de l'héritage multiple.



Avantages et Inconvénients
Adapter

Avantages

- Responsabilité ou Isolation de chaque code pour une fonctionnalité similaire pour des comportements différents
  - Séparation du code entre la Classe principale et l'Interface (ou le code de conversion de données)
  - Éviter de casser la principale logique business initiale en implémentant d'autres logiques ailleurs dans des Adapters

Inconvénients

- Complexité du code
  - Difficulté de lecture du code
  - Compréhension plus profonde du code
  - Temps d'analyse de la logique du code ainsi que les différentes relations

Sources

https://refactoring.guru/design-patterns/adapter

https://blog.cellenza.com/developpement-specifique/le-design-pattern-adapter/