**Compte rendu 1 micro-services**

**1.1 Activité 1**

**Une image contenant texte, logiciel, Logiciel multimédia, Logiciel de graphisme

Le contenu généré par l’IA peut être incorrect.**

On a crée une interface appeler IDao dans le package dao qui définit une seulee méthode getValue()

Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Le contenu généré par l’IA peut être incorrect.Cette classe inmplemente l’interface IDao et définit une valeur « value »

Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Le contenu généré par l’IA peut être incorrect.

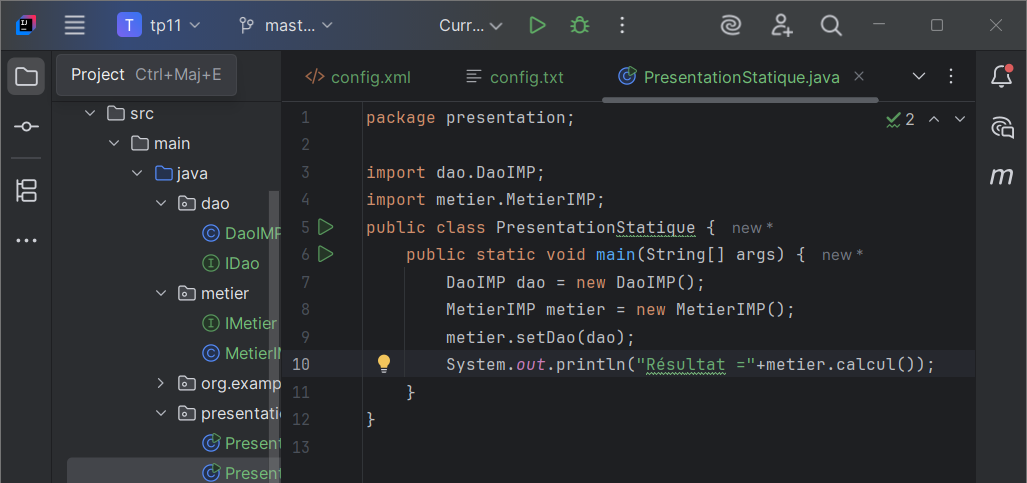
Cette interface declare la methode calcul

Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Le contenu généré par l’IA peut être incorrect.

Cette classe qui implémente l interface IDao telque « private IDao dao «  designe laa dépendance vers IDao

public void setDao(IDao dao) { this.dao = dao;} désigne l’injection de la dépendance via un setter cela permet de remplacer facilement le dao sans toucher au code métier ‘couplage faible’



La classe présentation statique utilisant l injection statique c à d le développeur qui relie les objets à la mainte lque les objets DaoIMP et MetierIMP sont créés directement dans le code

L’ instanciation statique se faitpar ‘new’ telque DaoIMP dao = new DaoIMP(); et MetierIMP metier = new MetierIMP();

On fait injection de dépendance manuellement par ‘metier.setDao(dao)’ , on dit que flexible (si on change de DAO, on doit recompiler)

Une image contenant texte, Logiciel multimédia, logiciel, Logiciel de graphisme

Le contenu généré par l’IA peut être incorrect.

Résultat = (valeur de dao=12.5) \* 7=37.5

Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.

On change le coefficient la résultat obtenu = 87.5

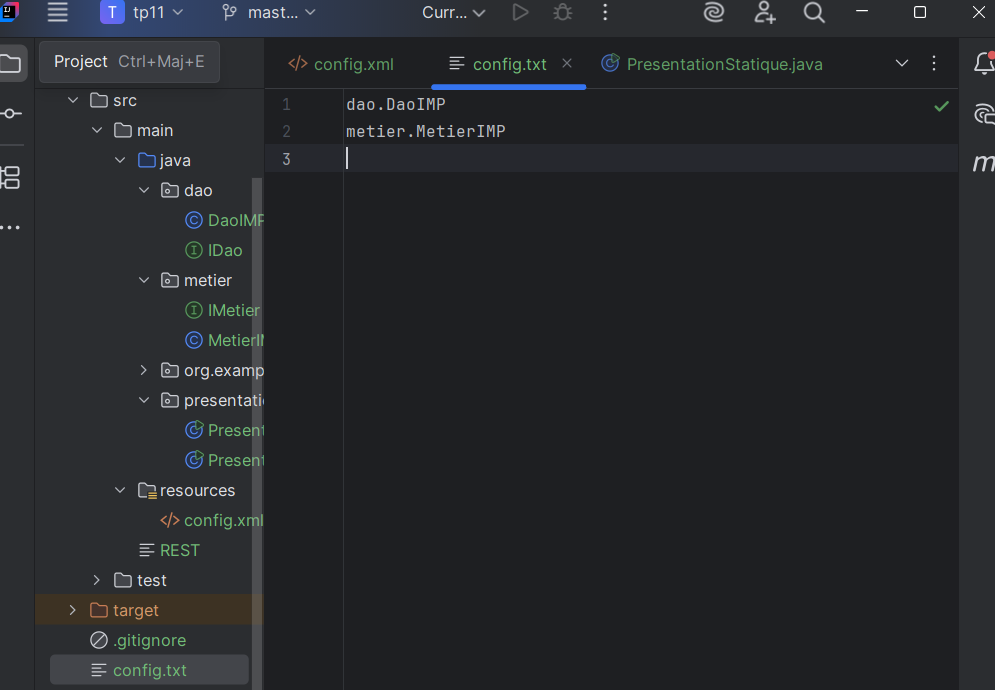
Une image contenant texte, capture d’écran, Police, logiciel

Le contenu généré par l’IA peut être incorrect.

Conclusion

Dans la version statique, presentation statique dépend directement des classes concrètes, ce qui rend le code flexible. En utilisant des interfaces et l’injection de dépendances, la classe ne dépend plus des détails d’implémentation ce qui réduit le couplage et le code plus flexible et facile à maintenir.

**1. Activité 2**

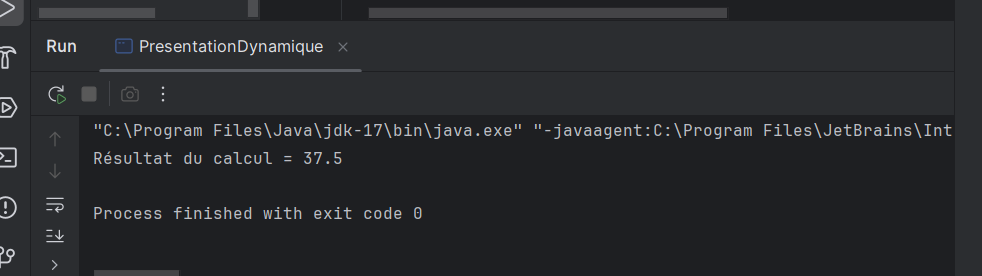


Le programme lit les noms des classes depuis config.txt

BufferedReader reader = new BufferedReader(new FileReader("config.txt"));  
et il charge ces classes avec Class.forName() puis il crée les objets dynamiquement avec newInstance() : IDao dao = (IDao) daoClass.getDeclaredConstructor().newInstance();  
IMetier metier = (IMetier) metierClass.getDeclaredConstructor().newInstance();

on fait l injection dynalmique via setter metier.getClass().getMethod("setDao", IDao.class).invoke(metier, dao);

6. ici, MetierIMP n’instancie pas directement DaoIMP et DaoIMP est créé à l’extérieur et injecté dans MetierIMP



7.conclusion

Dans PresentationDynamique l’objet DaoIMP est créé à l’extérieur et injecté dans MetierIMP. Cela illustre l’injection de dépendances par instanciation dynamique : la classe métier ne crée pas elle-même ses dépendances, ce qui réduit le couplage et rend le code plus flexible et facile à maintenir.