Notions fondamentales (sans Spring Boot)

Objectifs

- Comprendre l'inversion de contrôle (IoC)
- Comprendre l'injection de dépendances (DI)
- Définir des beans Spring en XML
- Démarrer un contexte Spring
- Organiser une application simple en couches

Inversion de contrôle (IoC)

- Principe : l'application ne crée plus directement ses objets.
- C'est le **contexte Spring** qui instancie et connecte les composants à notre place.
- Avantage : découplage du code, testabilité, maintenabilité.

Inversion de contrôle (IoC)

Exemple (sans Spring)

```
Service service = new Service(new Repository());
```

Avec Spring (IoC)

```
Service service = context.getBean("service", Service.class);
```

Injection de dépendances (DI)

- Spring fournit automatiquement les dépendances nécessaires à un bean.
- Plusieurs types d'injection : par constructeur, par setter, par champ (champ = déconseillé ici).
- Dans ce TP : injection **via setter** dans le fichier XML.

Le fichier applicationContext.xml

- C'est le cœur de la configuration dans ce TP.
- On y déclare les objets (beans) et leurs dépendances.

Syntaxe de base

```
<bean id="..." class="...">
    <property name="..." ref="..."/>
</bean>
```

Montrez qu'on peut gérer les dépendances sans annotation, tout est déclaré dans ce fichier XML.

Structure classique d'une application

- **Entity** : objets représentant les données manipulées
- **Repository** : gère la "persistance" ou l'accès aux données
- **Service** : logique métier, utilise les repositories
- **Main / Lanceur** : initialise le contexte et lance l'application

Démarrage de Spring

Chargement du contexte via la classe `ClassPathXmlApplicationContext` :

```
ApplicationContext context =
    new ClassPathXmlApplicationContext("applicationContext.xml");
```

Récupération d'un bean Spring :

```
Service service = context.getBean("service", Service.class);
```

Bonnes pratiques

- Garder une responsabilité claire pour chaque classe.
- Éviter que les services connaissent les détails d'implémentation des repositories.
- Penser en termes d'interfaces et d'abstraction.

Ce que vous devez savoir faire

- Créer une classe simple avec des getters/setters.
- Créer un repository avec une méthode "save".
- Créer un service qui dépend d'un repository.
- Définir les beans dans un fichier XML.
- Démarrer un contexte et appeler un service.

Attention

- Pas d’annotations dans ce TP
- Pas de Spring Boot
- Pas de base de données : repository simplifié
- Uniquement de la configuration XML