Context and Dependency Injection

Pr. Youssef Saadi

Master Informatique Décisionnelle

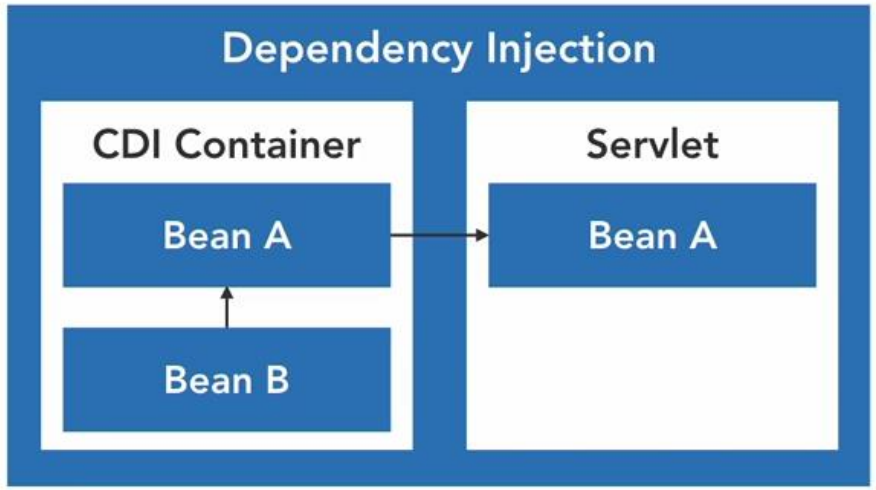
Faculté Des Sciences Et Techniques
Université Sultan Moulay Slimane Béni-Mellal

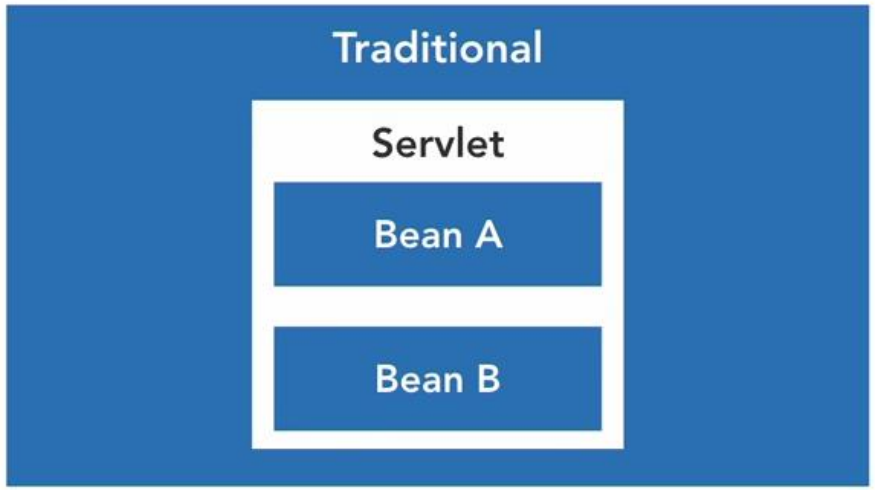
AU: 2019/2020

Introduction

- CDI (Context and Dependency Injection) est une spécification destinée à standardiser l'injection de dépendances et de contextes, au sein de la plateforme Java et plus particulièrement Java EE.
- Intégrée à la spécification Java EE 6, sa version 1.0 est sortie en décembre 2009 et a été suivie des versions 1.1 (mai 2013) et 1.2 (avril 2014). Son élaboration a été le fruit des JSR 299 et 346.
- Le but de cette spécification est de définir les API et les comportements associés en ce qui concerne ce qu'on appelle communément l'injection de dépendances (inversion de contrôle).
- Source : https://rmannibucau.developpez.com/tutoriels/cdi/introduction-cdi/#LVI

Injection de dépendance






```
@WebServlet("/Servlet")
public class Servlet extends HttpServlet {

@Inject
public BeanA beanA;
        /** GET/POST Omitted **/
}
```

Injection Point

```
public class BeanA {

        @Inject
        public BeanB beanB;
}
```

Injection Point

```
@WebServlet("/Servlet")
public class Servlet extends HttpServlet {

public BeanA beanA = new BeanA();
            /** GET/POST Omitted **/
}
```

```
public class BeanA {
            public BeanB beanB = new BeanB();
}
```

Source: Linkedin.com

Mode de découverte des beans

- **Explicite**
  - Dans le fichier beans.xml, l'on active l'entrée bean-discovery-mode=all
  - CDI version 1.1 et supérieur
- **Implicite** (par défaut)
  - Seulement les beans annotés de tels seront considérés.
  - Pas de beans.xml; bean-discovery-mode=annoted par défaut

Les portées des beans

- **@RequestScoped** : lié à la « requête ». Typiquement la requête HTTP, mais peut aussi être considéré comme lié à un ThreadLocal ;
- **@SessionScoped** : lié à la session HTTP ;
- **@ApplicationScoped** : lié à l'application (créé à la première invocation et détruit quand l'application est détruite) ;
- **@ConversationScoped** : lié à la conversation courante (sorte de sous-session démarrée/stoppée manuellement) ;
- **@Dependent** : crée une nouvelle instance pour l'injection en cours.

Les qualificateurs

- Il arrive que le type ne soit pas suffisant pour différencier deux beans. On peut avoir deux moteurs, mais un de marque **X1** et un autre de marque **X2**. Cependant les deux sont des « **Moteur** ».

```
Interface Moteur{}
class X1 implements Moteur
class X2 implements Moteur
class Voiture {
    @Inject
    Moteur x;
//
}
```

Les qualificateurs

- Pour garder le typage fort, CDI permet d'enrichir avec une information supplémentaire le type du bean : ce sont les qualificateurs.
- En pratique un qualificateur est une annotation décorée avec @javax.inject.**Qualifier**.
- Définir un qualificateur est aussi simple que définir une annotation décorée de @Qualifier.

Les intercepteurs

- Les intercepteurs permettent d'effectuer des actions avant et après les appels de méthodes et depuis CDI 1.1, après les appels de constructeurs.
- Pour faire cela, il y a trois étapes :
  - définir un @InterceptorBinding ;
  - définir son intercepteur ;
  - activer l'intercepteur dans le beans.xml.
- **Note** : les intercepteurs style EJB (@Interceptors) sont supportés, mais privilégiez les @InterceptorBinding qui permettent un faible couplage entre le bean et l'intercepteur.