SPRING

# Introduction

* Injecteur de dépendances, un CDI (context dependency injection)
* Plus puissant qui existe
* Standard de fait, reconnu car extrêmement utilisé, mais pas standard pur oracle J2EE
* Permet de factoriser et de modifier de gros morceaux de codes ou de fonctionnalités, en ne modifiant aucun code mais un fichier type config.

# Conteneurs Spring

## BeanFactory

* Plus léger, moins puissant (smartphones)

## ApplicationContext

# Injection de dépendances et Inversion de contrôle

* Ce sont des **design patterns**
* Pratique car on gère juste des POJO simples, puis le framework s’occupe de la partie complexe (et rébarbative niveau code)

## Injection de dépendances

* Injection d’instances, dépendances de classes entre-elles, **gestion du cycle de vie des objets**

## Inversion de contrôle

* proxy par interfaces (spring)… Redirige les méthodes = **inversion du contrôle des flow d’exécution du programme.** On laisse faire le framework. Les méthodes appelées sont implémentées selon une demande (transaction, écriture…)

# Fonctionnement

* Avec Maven on look le repo/dep dans mvnrepository.com pour l’ajouter dans le POM.xml
* On créer le fichier applicationContext.xml dans main/resources, qui contient les déclarations des beans, config et paramétrages
* Dans le main, il faut démarrer Spring en premier : **ApplicationContext context = new ClassPathXmlApplicationContext(‘’applicationContext.xml’’)**
* Les beans spécifiés sont alors instanciés (new) en instance unique (par défaut)
* Puis on utilise avec (Class) context.getBean(‘‘nom’’) ou si unique context.getBean(ClassName.class)