Spring & Springboot

Table of Contents

Injection de dependance (IOC) utilisant XML

TIP	@ComponentScan, @Component, @Autowired, @Qualifier

Dans Sring @ComponentScan representte la configuration de base de `Srping` dans `Spring` l'annotation vamapper `@ComponentScan(basePackages ={"spring.bean"}` ce qui est remplacer par `application.yml ou.properties` dans `Springboot`.

- @ComponentScan(basePackages ={"spring.bean"} recherche des @Bean pour être utilisé dans la classe `main`.
  - @Component : est une classe que va chercher componentScan. Le @component seront des "@bean" dans le `main`
    - @Autowired : injection dans un service.
    - @Qualifier : moyen pour donner un nom a un component (le nom de la class) qui va être son ID
- pacquet `org.springframework.stereotype` est composé de
  - @Component: une classe candidat dans le but est de scanner les composant de Spring
  - @Controller: utilisé dans une application Spring MVC
  - @Repository: utilisé pour définir une classe DAO (Data Access Object)
  - @Service: utilisé pour définir le bisiness Services

IMPORTANT	Meta-Annotations sont des méthodes de classe mère exemple @Transactionnal

- @Autowired l'annotation fait de l'injection de dependance, elle peut être appelé sur un constructeur, une classe, une methode. exemple sur une classe DAO
  - classe de configuration de la base de données (DAO) coit être annoté pas @Component — l'annotation @Autowired peut être appelé dans cette classe, sur les constructeurs et les méthodes

Question: comment on fait si on veut injecter dans un champ private ? - on utilise @Autowird ou @Ressource Question: qu'est ce qu'un prefix? - Est utile pour mapper et accéder à une ressource. Dans `Spring` on peut utiliser des préfix sur le `classpath`,`file`,`http`,`(none)`

Injection de dependance (IOC) en utilisant @Configuration

`@Configuration` permet d'eviter de rediger un `fichier xml`. La configuration peut directement être écrit dans le main annoté @Bean. exemple:

@Bean public DataBaseService data(){ DataBaseService data = new DataBaseService(); return data;

} ---

Spring & Springboot (~/ressouces/)

- abstract=true bean sert a regrouper des propiété de class dans <bean></bean> Question: pourquoi on ne peut pas mettre @Bean sur une methodes final
- toutes les classes annotées @Configuration utilisé CGLIB de JavaConfig, en consequence toutes ses occurences ne peuvet être marquées en final ou private, donc @Bean n'ai pas pas accepté non plus en raison de CGLIB.

PropertyPlaceHolder

C'est un moyen de resoudre `${}` dans un fichier XML ou system ou variable d'environnement.

```
1) déclarer en tant que @Bean dans main
2) creer un fichier de configuration.properties
3) mettre les variables dans ce fichier
4) annoter la classe l'utilisant avec l'annotation @PropertySource
```

Scopes (application-context.xml)

Les scopes sont définis das la balise <bean .. scope="xxx"> soit par singleton, prototype, session, request, global_session, application.

- Singleton: est une et une seule instance d'un objet dans le Springframework
- Prototype: exemple sur le cas de login, au lieu de redefinir plusieurs fois une configuration, on va écrire dans application.xml la configuration et on l'annote @Prototype
- Session: environment web dont une instance va être créer à chaque session HTTP
- Request: environment web dont une instance va être créer à chaque requette
- Global session:

Profil en Spring (@Profil)

@Profile peut être defini dans un fichier XML ou dans une class. Ca consiste a definir un profil basé sur une configuration. exemple: On a deux @bean methode de connexion à une base de données.

```
1) on peut definir nommé un profil dessus
2) on peut l'appelé sur une `class` avec @ActiveProfil("nomProfil")
```