Spring Core - annotation

Di - @Autowired on field

Autowiring on field

- On va pouvoir injecter une dépendance directement dans la propriété (même privée), dans classe où elle réside (= Java Reflection).
- On va continuer d'utiliser cet exemple déjà vu :
  - Injecter PrepareService dans une implémentation de Musicien
  - Spring scanne les packages , et parmi les classes annotées @Component il en cherche une qui implémente PrepareService.
  - S'il en trouve une , Spring l'injecte automatiquement (par exemple ZenPreparationService)

Démo – auto wire sur un attribut

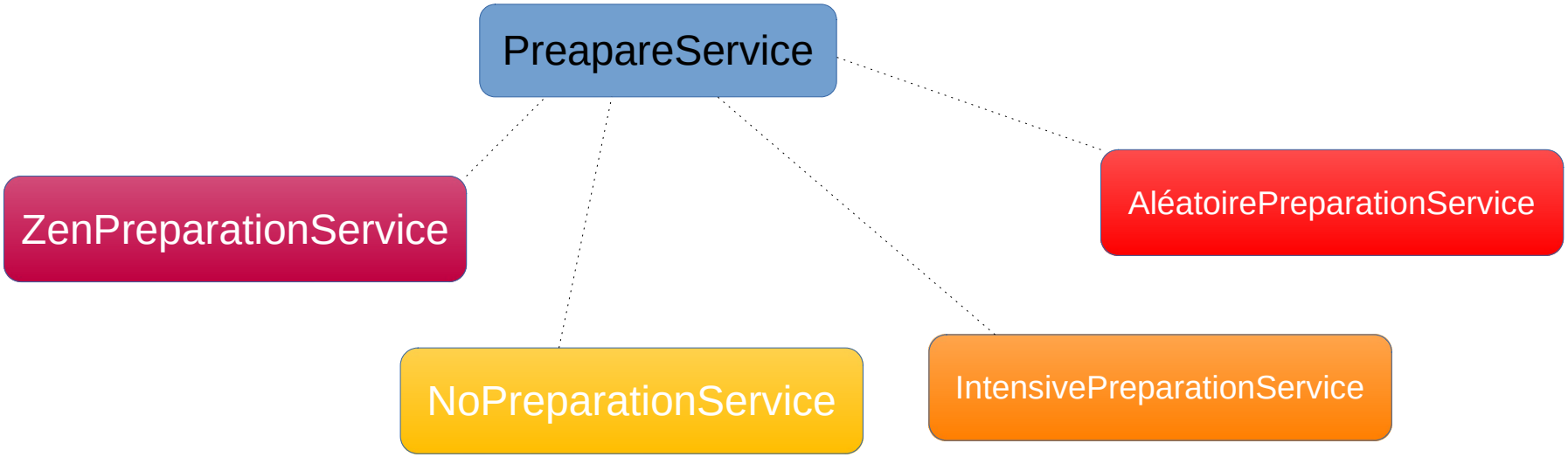
- A partir de la démo précédente , déplacez l' @Autowired depuis le setter vers la propriété prepareService
- supprimez constructeur et setter
- Live démo .java reflexivité..
- exécutez Run As > Java Application =>

Quel type d'injection choisir ?

- C'est votre décision , mais restez constants dans votre code
- Injection par constructeur, setter ou attribut répondent de manière équivalente au même besoin

@Qualifier

- Si il existe plusieurs implémentations de PrepareService ? Que se passe t il ?
- Quelle implémentation spring va t il choisir pour l'injection ?



Problème ... à résoudre.

- Une erreur est levée :

```
WARNING: Exception encountered during context initialization - cancelling refresh
attempt: org.springframework.beans.factory.UnsatisfiedDependencyException: Error
creating bean with name 'trompetiste': Unsatisfied dependency expressed through
field 'prepareService'; nested exception is
org.springframework.beans.factory.NoUniqueBeanDefinitionException: No qualifying
bean of type 'com.springdemo.PrepareService' available: expected single matching
bean but found 2: intensivePreparationService,zenPreparationService
Exception in thread "main"
org.springframework.beans.factory.UnsatisfiedDependencyException: Error creating
bean with name 'trompetiste': Unsatisfied dependency expressed through field
'prepareService'; nested exception is
org.springframework.beans.factory.NoUniqueBeanDefinitionException: No qualifying
bean of type 'com.springdemo.PrepareService' available: expected single matching
bean but found 2: intensivePreparationService,zenPreparationService
     at
org.springframework.beans.factory.annotation.AutowiredAnnotationBeanPostProcessor$
AutowiredFieldElement.inject(AutowiredAnnotationBeanPostProcessor.java:643)
```

La solution est @Qualifier

```java
@Component
public class Trompetiste implements Musicien {

    @Autowired
    @Qualifier("zenPreparationService")
    private PrepareService prepareService;

    public void setPrepareService(PrepareService prepareService) {
        System.out.println(">> Trompetiste: à l'intérieur du ...");
        this.prepareService = prepareService;
    }

    @Override
    public String joueTaPartition() {
        return "la trompête du toril";
    }

    @Override
    public String preparesToi() {
        return prepareService.getPreparation();
    }
}
```

Nom de bean par défaut particulier

- le nom par défaut attribué par spring utilise la convention de nommage : première lettre de la Classe en minuscule

  Par exemple : ZenPreparationService → zenPreparationService

- Cas particulier si la classe possède deux majuscules aux deux premières lettres : alors le default id sera le nom de la classe (sans transformation)

NB : @Autowired sur un constructeur

```
@Component
public class Trompetiste implements Musicien {

    private PrepareService prepareService;

    //default constructor
    public Trompetiste(  ) {
        System.out.println("Trompetiste : à l'intérieur du default constructor ");
    }

    @Autowired
    public Trompetiste(@Qualifier("intensivePreparationService")
    PrepareService prepareService) {
    System.out.println("Trompetiste : à l'intérieur du  constructor d'injection");
    this.prepareService = prepareService;
    }

    @Override
    public String joueTaPartition() {
        return "la trompête du toril";
    }

    @Override
    public String preparesToi() {
        return  prepareService.getPreparation();
    }
}
```