



S02 Spring Core – 02 configuration Xml

02-02 Injection de dependances par setters

Méthode d'injection par setter(s)

- Prennons l'exemple d'un nouveau type de Musicien :

```
public class Batteur implements Musicien {  
  
    private PrepareService prepareService;  
  
    @Override  
    public String jouTaPartition() {  
  
        return "je joue les tambours du Bronx";  
    }  
  
    @Override  
    public String getPrepa() {  
        return prepareService.getPreparation();  
    }  
  
    public Batteur () {  
        System.out.println("Batteur : à l'intérieur du constructeur sans paramètre");  
    }  
  
    public void setPrepareService( PrepareService unService) {  
        System.out.println("Batteur : à l'intérieur de la méthode set – setPrepareService()");  
        this.prepareService= unService;  
    }  
  
}
```

Appelée par Spring
lors de l'injection
de la dépendance

Configurer la dependance dans le fichier de configuration xml

File: applicationContext.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:context="http://www.springframework.org/schema/context"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd
http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context.xsd">

  <bean id="unPrepareService"
        class="com.springdemo.ZenPreparationService" >
  </bean>

  <bean id="unMusicien" class="com.springdemo.Violoniste" >
    <!-- Définir l' Injection par constructeur -->
    <constructor-arg ref="unPrepareService"/>
  </bean>

  <bean id="unBatteur" class="com.springdemo.Batteur" >
    <!-- Définir l' Injection par setter -->
    <property name="prepareService" ref="unPrepareService" />
  </bean>

</beans>
```

Cela va appeler
public void setPrepareService(...)

Setter injection

En arrière plan

```
<bean id="unPrepareService"  
  class="com.springdemo.ZenPreparationService" >  
</bean>
```

```
<bean id="unMusicien" class="com.springdemo.Batteur" >  
  <property name ="prepareService" ref ="unPrepareService" />  
</bean>
```




```
ZenPrepareService zenPrepareService = new  
ZenPrepareService() ;
```

```
Batteur batteur = new Batteur() ;  
batteur.setPrepareService( zenPrepareService ) ;
```

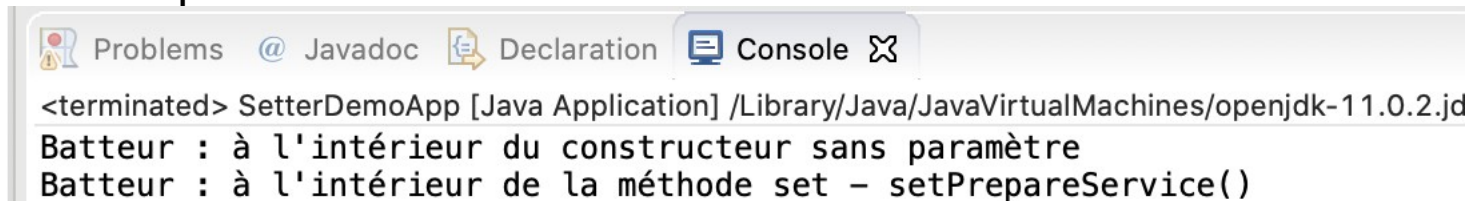
Une nouvelle méthode main (1/2)

```
public class SetterDemoApp {  
  
    public static void main(String[] args) {  
  
        // charger le context spring depuis le fichier de configuration xml  
        ClassPathXmlApplicationContext context = new  
            ClassPathXmlApplicationContext("applicationContext.xml");  
  
        //récupérer un bean depuis le container  
        Batteur batteur = context.getBean("unBatteur", Batteur.class);  
  
        // appeler une méthode sur le bean  
        // ...  
  
        // fermer le contexte  
        context.close();  
  
    }  
}
```



```
<bean id="unBatteur" class="com.springdemo.Batteur" >  
    <property name ="prepareService" ref ="unPrepareService" />  
</bean>
```

Exécuter ce main tel qu'il est ici.... =>

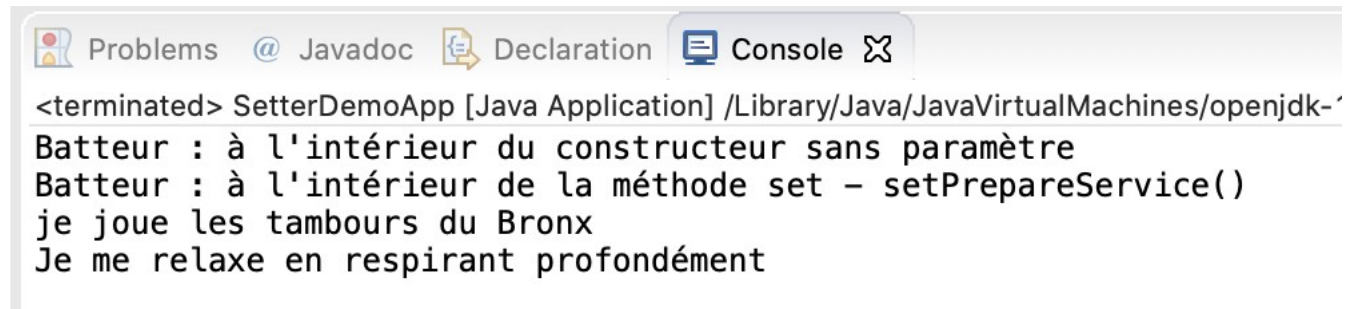


```
<terminated> SetterDemoApp [Java Application] /Library/Java/JavaVirtualMachines/openjdk-11.0.2.jd  
Batteur : à l'intérieur du constructeur sans paramètre  
Batteur : à l'intérieur de la méthode set - setPrepareService()
```

La méthode main (2/2)

```
public class SetterDemoApp {  
  
    public static void main(String[] args) {  
  
        // charger le context spring depuis le fichier de configuration xml  
        ClassPathXmlApplicationContext context = new  
            ClassPathXmlApplicationContext("applicationContext.xml");  
  
        //récupérer un bean depuis le container  
        Batteur batteur = context.getBean("unBatteur", Batteur.class);  
  
        // appeler une méthode sur le bean  
        System.out.println(batteur.joueTaPartition());  
        System.out.println(batteur.getPrepa());  
  
        // fermer le contexte  
        context.close();  
    }  
}
```

Exécution =>



```
<terminated> SetterDemoApp [Java Application] /Library/Java/JavaVirtualMachines/openjdk-  
Batteur : à l'intérieur du constructeur sans paramètre  
Batteur : à l'intérieur de la méthode set - setPrepareService()  
je joue les tambours du Bronx  
Je me relaxe en respirant profondément
```