



Core – java

configuration du container

Configurer le container en java

- Aujourd'hui on n'utilise plus de fichiers xml dans notre configuration

Full XML configuration

```
...
<!-- Définir les beans ici -->
<!-- Définir les Dépendances -->
<bean id="unPrepareService"

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

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

XML Component Scan

```
...
<context:component-scan base-package="com.springdemo"/>
```

Java Configuration

```
package com.springdemo.conf;

import org.springframework.context.annotation.ComponentScan;
import org.springframework.context.annotation.Configuration;

@Configuration
@ComponentScan("com.springdemo")
public class MusiConfig {

}
```

Pas d'XML !



Etape 1 : créer une classe de Configuration

```
package com.springdemo.conf;

import
org.springframework.context.annotation.Configuration;

@Configuration
public class MusiConfig {
    // ajouter du contenu
}
```

Etape 2 (optionnelle)

@ComponentScan

- Pour l'instant on va simplement ajouter l'annotation pour indiquer les packages à scanner
- On y reviendra plus tard

```
import org.springframework.context.annotation.ComponentScan;
import org.springframework.context.annotation.Configuration;

@Configuration
@ComponentScan("com.springdemo")
public class MusiConfig {

}
```



```
<context:component-scan base-package="com.springdemo"/>
```

Etape 3 & 4: Lire la configuration, accéder au bean.

- A partir de la méthode main, pour "ouvrir" le contexte on va changer la classe et la méthode utilisée . Avec la configuration par code java on utilisera
- Récupérer le bean depuis le contexte

```
import org.springframework.context.annotation.AnnotationConfigApplicationContext;
import com.springdemo.conf.MusiConfig;

public class JavaCodeConfigDemoApp {

    public static void main(String[] args) {
        //lire la classe de configuration java
        AnnotationConfigApplicationContext context = new AnnotationConfigApplicationContext (MusiConfig.class);

        // accéder au bean géré par le spring container
        Trompetiste musicien= context.getBean("trompetiste", Trompetiste.class);
        ...etc
    }
}
```

Dernière étape : récupérer les beans depuis le container

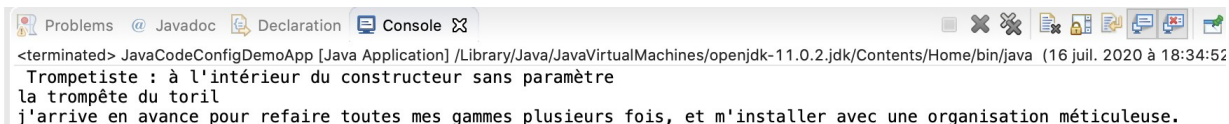
Ainsi le bean est fonctionnel on peut s'en servir normalement :

```
// utiliser les methode de musicien
System.out.println(musicien.joueTaPartition());
System.out.println(musicien.preparesToi());

// fermer le context
context.close();
}
```

Run As > Java Application =>

Il n'y a plus de fichier de configuration xml mais tout fonctionne de la même façon



```
<terminated> JavaCodeConfigDemoApp [Java Application] /Library/Java/JavaVirtualMachines/openjdk-11.0.2.jdk/Contents/Home/bin/java (16 juil. 2020 à 18:34:52)
Trompetiste : à l'intérieur du constructeur sans paramètre
la trompète du toril
j'arrive en avance pour refaire toutes mes gammes plusieurs fois, et m'installer avec une organisation méticuleuse.
```