Navigation

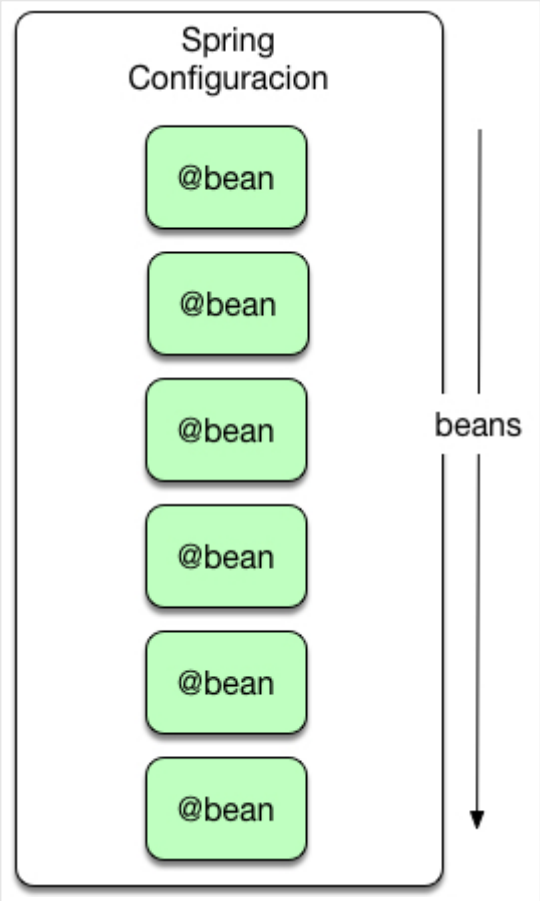
Spring @import , organizando Spring framework

por **Cecilio Álvarez Caules** sobre 9 Agosto, 2017 en **Spring Core**

Spring @import nos permite organizar de una mejor manera **la configuración de Spring Framework** . Normalmente en una configuración de Spring basada en anotaciones tenemos algo similar a lo siguiente:

```
1  package com.arquitecturajava.spring;
2
3  import org.springframework.context.annotation.Bean;
4  import org.springframework.context.annotation.ComponentScan;
5  import org.springframework.context.annotation.Configuration;
6
7  @Configuration
8  public class SpringConfiguracion {
9
10     @Bean
11     public static HelperPersistencia helper1() {
12
13         return new HelperPersistencia();
14     }
15
16     @Bean
17     public static HelperSeguridad helper2() {
18
19         return new HelperSeguridad();
20     }
21
22 }
```

En este caso el fichero es muy pequeño pero es bastante habitual q**ue tenga cientos de lineas de código registrando cada uno de los beans que son necesarios , seguridad ,persistencia etc.**



Podemos crear un programa principal y ejecutarlo:

```
1  package com.arquitecturajava.spring;
2
3  import org.springframework.context.annotation.AnnotationConfigApplicationContex
4
5
6
7  public class Principal {
8
9      public static void main(String[] args) {
10
11         AnnotationConfigApplicationContext contexto = new AnnotationConfigAppli
12         contexto.register(SpringConfiguracion.class);
13         contexto.refresh();
14
15         HelperSeguridad helper1=contexto.getBean(HelperSeguridad.class);
16         helper1.hola();
17         HelperPersistencia helper2=contexto.getBean(HelperPersistencia.class);
18         helper2.hola();
19
20     }
21
22 }
```

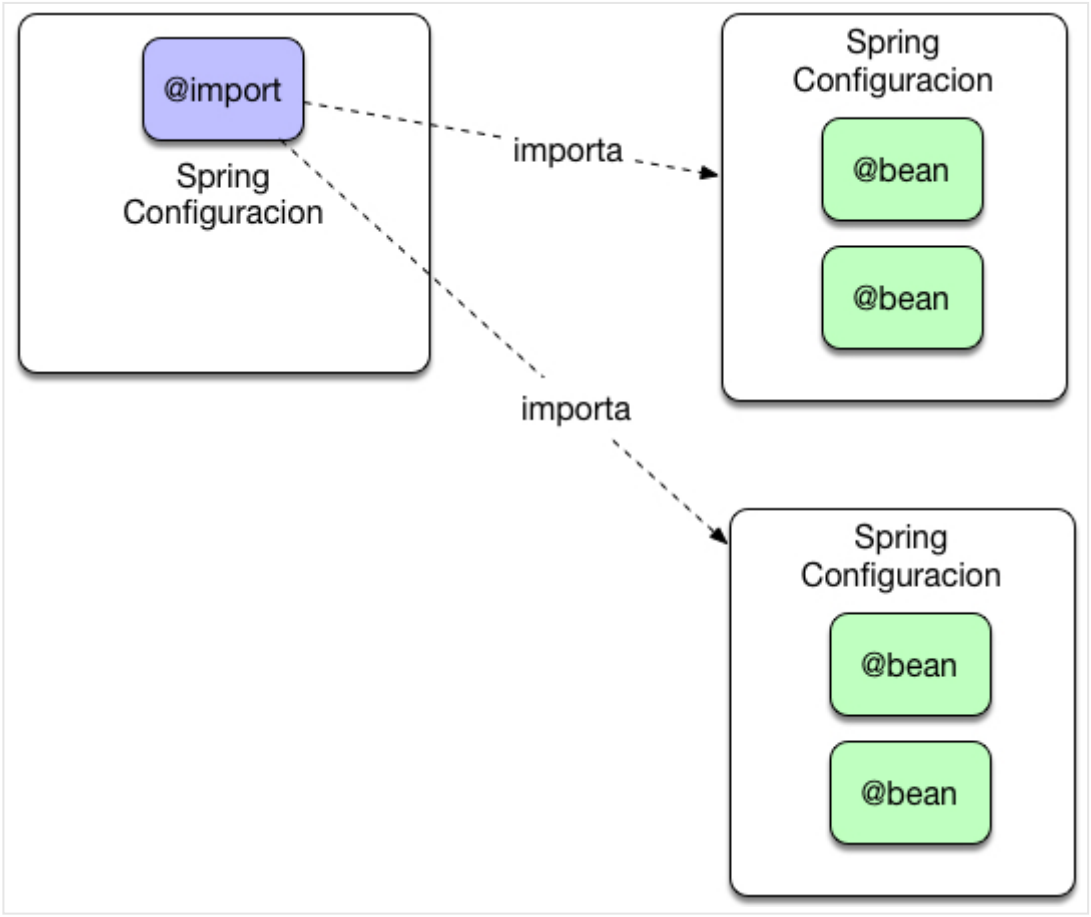
Veremos el resultado por la consola:

```
<terminated> Principal2 (2) [Java Application] /Library/Java/JavaVirtualMa
ago 09, 2017 10:32:05 AM org.springframework.context.anr
INFORMACIÓN: Refreshing org.springframework.context.annc
ago 09, 2017 10:32:05 AM org.springframework.beans.facto
INFORMACIÓN: Overriding bean definition for bean 'helper
ago 09, 2017 10:32:05 AM org.springframework.beans.facto
INFORMACIÓN: Overriding bean definition for bean 'helper
ago 09, 2017 10:32:05 AM org.springframework.beans.facto
INFORMACIÓN: Overriding bean definition for bean 'helper
bean que se usa en temas de seguridad
bean que se usa en JPA
```

Los beans se ejecutan sin problema:

Spring @import

Sin embargo si tenemos decenas de beans ,esto acaba siendo problemáti**co ya que es difícil clarificar que es todo lo que tenemos configurado**. Para solventar este tipo de problemas Spring soporta la anotación **@import** . Con **Spring @import** podemos partir el fichero en varias partes y organizarlo todo mucho mejor.



Vamos a verlo en código:

```
1  package com.arquitecturajava.spring;
2
3  import org.springframework.context.annotation.Bean;
4  import org.springframework.context.annotation.ComponentScan;
5  import org.springframework.context.annotation.Configuration;
6
7  @Configuration
8  public class SpringConfiguracionSeguridad {
9
10
11     @Bean
```

```
    public static HelperSeguridad helper2() {

        return new HelperSeguridad();
    }

}
```

```
package com.arquitecturajava.spring;

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

@Configuration

public class SpringConfiguracionPersistencia {

    @Bean
    public static HelperPersistencia helper1() {

        return new HelperPersistencia();
    }
}
```

```
package com.arquitecturajava.spring;

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

@Configuration
@Import({SpringConfiguracionSeguridad.class,SpringConfiguracionPersistencia.cla
public class SpringConfiguracionMega {


}
```

Acabamos de configurar Spring Framework apoyándonos en tres ficheros (SpringConfiguracion, SpringConfiguracionPersistencia, SpringConfiguracionSeguridad) . De esta forma es mucho más fácil acceder a cada bloque de configuración organizado en las categorías que nosotros necesitemos.

Otros artículos relacionados:

1. Spring Boot AOP y rendimiento
2. Introducción a Spring Data y JPA
3. Spring Singleton vs Prototype
4. @Import anotación

Related Posts:

Java Stream String y Java 8
JPA DTO (Data Transfer Object) y JPQL

JPA Criteria API , un enfoque diferente
JPA SQL Injection y sus problemas
Java Stream forEach y colecciones

‹ Java Stream String y Java 8

2 Responses to *Spring @import , organizando Spring framework*

Antuan 10 Agosto, 2017 at 11:39 #

Muchas gracias por tus píldoras super útiles

RESPONDER

Cecilio Álvarez Caules 10 Agosto, 2017 at 11:53 #

de nada 🙂 , me alegro que te sea útil 😉

RESPONDER

Deja un comentario

Name (required)

Email (will not be published) (required)

Website

Submit Comment

Buscar

Search...

Translate:

Seleccionar idioma

Con la tecnología de Google **Traductor de Google**

POPULAR

Mis Cursos de Java para desarrolladores
27 JULIO, 2017

JPA Composite Key y business objects
28 ABRIL, 2017

Command Pattern en Java y la gestion de tareas
26 MAYO, 2017

Java LinQ con JinQ y Java Persistence API
17 MAYO, 2017

JPA Database Schema y automatización
31 MAYO, 2017

Java 8 interface static methods y reutilizacion
20 ABRIL, 2017

Java 8 Factory Pattern y su implementación
26 JUNIO, 2017

El concepto de EJB Event y como desacoplar servicios
25 JULIO, 2017

Java Stream forEach y colecciones
15 JUNIO, 2017

Java value vs reference y sus curiosidades


17 ABRIL, 2017


Java Multiple Inheritance con default methods
19 MAYO, 2017


Eclipse Pull up , Pull down y refactorings
5 MAYO, 2017

Redes Sociales

Síguenos en Linkedin y Twitter o subscríbete al RSS.

E-mail Submit

Contacto

contacto@arquitecturajava.com