DI с использованием аннотаций

Простая XML-конфигурация Spring

```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"
xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd">
</beans>
```

XML-конфигурация Spring с поддержкой аннотаций

```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">
    <context:component-scan base-package="by.htp.jd2.annotation" />
</beans>
```

Дескриптор <context: component-scan> сообщает Spring о необходимости сканирования кода на предмет внедряемых бинов, аннотированных с помощью @Component, @Controller, @Repository и @Service, а также поддерживающих аннотации @Autowired и @Inject в указанном пакете (и всех его внутренних пакетах). В дескрипторе <context: component-scan> можно определить множество пакетов, используя в качестве разделителя запятую, точку запятой или пробел. Кроме того, для более детализированного управления этот дескриптор поддерживает включение и исключение сканирования компонентов.

```xml
<context:component-scan base-package="com.apress.prospring4.ch3.annotation" >
    <context:exclude-filter type="assignable" expression="com.example.NotAService"/>
</context:component-scan>
```

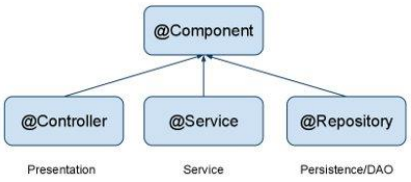
Для определения бина с помощью аннотаций следует использовать @Component, @Service, @Repository.

```java
package org.springframework.stereotype;

import java.lang.annotation.Documented;
import java.lang.annotation.ElementType;
import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;
import java.lang.annotation.Target;

@Target(ElementType.TYPE)
@Retention(RetentionPolicy.RUNTIME)
@Documented
@Indexed
public @interface Component {

    String value() default "";

}
```

```java
package org.springframework.stereotype;

import java.lang.annotation.Documented;
import java.lang.annotation.ElementType;
import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;
import java.lang.annotation.Target;

import org.springframework.core.annotation.AliasFor;

@Target({ElementType.TYPE})
@Retention(RetentionPolicy.RUNTIME)
@Documented
@Component
public @interface Service {

    @AliasFor(annotation = Component.class)
    String value() default "";

}
```

```java
package org.springframework.stereotype;

import java.lang.annotation.Documented;
import java.lang.annotation.ElementType;
import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;
import java.lang.annotation.Target;

import org.springframework.core.annotation.AliasFor;

@Target({ElementType.TYPE})
@Retention(RetentionPolicy.RUNTIME)
@Documented
@Component
public @interface Repository {

    @AliasFor(annotation = Component.class)
    String value() default "";

}
```

Разница между аннотациями @Component, @Repository, @Service



@Component – используется для указания класса в качестве компонента spring. При использовании поиска аннотаций, такой класс будет сконфигурирован как spring bean
@Controller – специальный тип класса, применяемый в MVC приложения. Обрабатывает запросы и чпсто используется с аннотацией @RequestMapping
@Repository – указывает, что класс используется для работы с поиском, получением и хранением данных. Аннотация может использоваться для реализации шаблона DAO
@Service – указывает, что класс является сервисом для реализации бизнес логики

Для указания контейнеру на класс-бин можно использовать любую из этих аннотаций. Но различные имена позволяют различать назначение того или иного класса.

spring-annotations-component
spring-annotations-service-and-default-name

Внедрение через методы установки.

Напоминалка (внедрение через проперти и пространтсов имен р

spring-annotations-setters-autowired

```
<bean id="myCricketCoach"
        class="by.htp.jd2.di_setters.CricketCoach">

        <property name="fortuneService" ref="myFortuneService" />
</bean>
```

Пространство имен р не определено в файле XSD и существует только в ядре Spring; следовательно, в атрибуте schemaLocation никакие файлы XSD не объявляются.

```
xmlns:p="http://www.springframework.org/schema/p"

<bean id="injectSimpleSpel"
                class="by.htp.jd2.start.spring_spel.InjectSimpleSpEL"
                p:name="#{injectSimpleConfig.name}"
                p:age="#{injectSimpleConfig.age + 1}"
                p:height="#{injectSimpleConfig.height}"
                p:programmer="#{injectSimpleConfig.programmer}"
                p:ageInSeconds="#{injectSimpleConfig.ageInSeconds}"
                p:pi="#{T(java.lang.Math).PI}"
                p:value="#{'Hello'}" />
```

```
<bean id="messageRenderer"
class="com.apress.prospring4.ch3.xml.StandardOutMessageRenderer"
p:messageProvider-ref="messageProvider"/>
```

При возникновении ситуации, когда контейнеру доступны несколько объектов, подходящий для внедрения, регерируется исключительная ситуация org.springframework.beans.factory.NoUniqueBeanDefinitionException

Внедрение через конструктор

spring-annotations-constructor-autowired

При возникновении ситуации конфликта – когда для внедрения доступны несколько контрукторов – сгенерируется исключительная ситуация org.springframework.beans.factory.BeanCreationException

```
<bean id="myCoach"   class="by.htp.jd2.di_constructor.TrackCoach">
                <constructor-arg ref="myFortuneService" />
</bean>
```

Пространство имен с не определено в файле XSD и существует только в ядре Spring; следовательно, в атрибуте schemaLocation никакие файлы XSD не объявляются.

```
<bean id="toyota" class="com.baeldung.spring.domain.Car">
        <constructor-arg index="0" ref="engine" />
        <constructor-arg index="1" ref="transmission" />
</bean>
```

```
<bean id="engine" class="com.baeldung.spring.domain.Engine">
        <constructor-arg index="0" value="v4" />
        <constructor-arg index="1" value="2" />
</bean>
```

```
xmlns:c="http://www.springframework.org/schema/c"

<bean id="messageProvider"
        class="com.apress.prospring4.ch3.xml.ConfigurableMessageProvider"
        c:message="This is a configurable message" />
```

Внедрение параметров в конструктор через @Value

spring-annotations-constructor-value
spring-annotations-constructor-without-value

Путаница с конструкторами
spring-di-constructor s-confusion

При внедрении через конструктор с использованием аннотаций такой путаницы можно избежать, применяя аннотацию непосредственно к целевому методу конструктора

spring-annotations-constructor s-confusion
spring-annotations-constructor s-confusion-without-value

Аннотация @Autowired может быть применена только к одному из методов конструкторов. В случае ее применения к нескольким методам конструкторов платформа Spring выдаст соответствующее сообщение во время начальной загрузки ApplicationContext.

Внедрение свойств

Использование @Autowired для внедрения свойств

spring-annotations-field-autowired

Использование @Value для внедрения свойств

spring-annotations-field-value

Использование SpEL для внедрения свойств

spring-annotations-field-spel

Внедрение через метод

spring-annotations-method-autowired

Внедрение файла properties через аннотации

spring-annotations-properties