

# Spring Annotation base Configuration

## 1. Spring Annotation Base là gì

- Từ Spring 2.5 trở đi thì có thể config Dependency injection bằng cách dùng Annotations. Thay vì dùng XML để mô tả Bean thì ta có thể dời toàn bộ Bean config vào chung trong Class thực thi của Bean đó bằng cách dùng annotation ở level class, method, field

❖ Ví dụ:

- o Có 1 class Student.java như sau

```
public class Student {  
  
    private String name;  
  
    public void setName(String name) {  
        this.name = name;  
    }  
    public String getName() {  
        return name;  
    }  
}
```

- o Có 2 cách để định nghĩa class Student là **Bean**: dùng **XML** hoặc **Annotation**

- o Dùng 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-3.0.xsd  
http://www.springframework.org/schema/context  
http://www.springframework.org/schema/context/spring-context-3.0.xsd">  
    <context:annotation-config/>  
  
    <!-- Definition for student bean -->  
    <bean id = "student" class = "com.example.Student">  
        <property name = "name" />  
    </bean>  
  
</beans>
```

## ○ Dùng Annotation

```
@Configuration
public class ApplicationConfig {

    @Bean
    public Student student() {
        return new Student();
    }
}
```

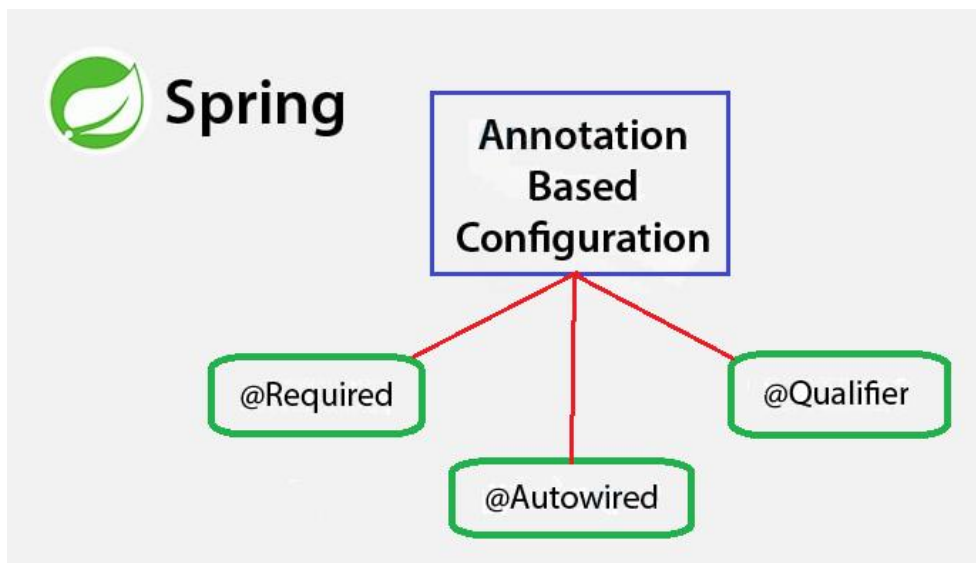
- Annotation injection được run trước XML injection. Do đó nếu như 1 bean được định nghĩa bằng cả 2 annotation và XML thì XML injection sẽ override lại Annotation injection
- Mặc định thì Annotation sẽ không được turn on, và nếu muốn sử dụng thì phải bật Annotation bằng cách chỉ định trong Spring configuration file như sau:

```
<?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-3.0.xsd
http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context-3.0.xsd">

    <context:annotation-config/>

</beans>
```

## 2. Vài Annotation quan trọng



## 2.1 @Required

Được dùng cho method **setter** của **Bean property**, chỉ định việc bắt buộc property của Bean phải được định nghĩa trong file XML config. Có nghĩa là object class Student có 1 property “name”, khi dùng @Required thì property “name” bắt buộc phải được định nghĩa trong file XML config. Nếu không có nó sẽ throw exception **BeanInitializationException**

Student.java

```
public class Student {  
  
    private String name;  
  
    @Required  
    public void setName(String name) {  
        this.name = name;  
    }  
    public String getName() {  
        return name;  
    }  
}
```

MainApp.java

```
public class MainApp {  
  
    public static void main(String[] args) {  
        ApplicationContext context = new  
        ClassPathXmlApplicationContext("Beans.xml");  
  
        Student student = (Student) context.getBean("student");  
        System.out.println("Name : " + student.getName() );  
    }  
}
```

## Beans.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-3.0.xsd
http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context-3.0.xsd">

  <context:annotation-config/>

  <!-- Definition for student bean -->
  <bean id = "student" class = "spring.example.Student">
  </bean>
</beans>
```

- ➔ Nếu không định nghĩa property "name" thì sẽ throw exception **BeanInitializationException**, bởi vì trong class **Student.java** đang định nghĩa **@Required** cho method **setName()**
- ➔ File **Bean.xml** sẽ fix như sau:

```
<?xml version = "1.0" encoding = "UTF-8"?>

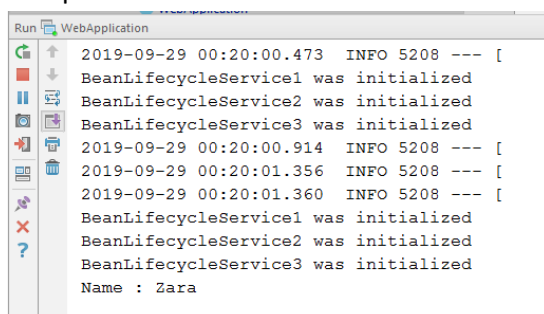
<beans xmlns = "http://www.springframework.org/schema/beans"
  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
  xmlns:context = "http://www.springframework.org/schema/context"
  xsi:schemaLocation = "http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context-3.0.xsd">

  <context:annotation-config/>

  <!-- Definition for student bean -->
  <bean id = "student" class = "spring.example.Student">
    <property name = "name" value = "Mr.Bean" />
  </bean>

</beans>
```

## Kết quả run:



```
Run WebApplication
2019-09-29 00:20:00.473 INFO 5208 --- [
BeanLifecycleService1 was initialized
BeanLifecycleService2 was initialized
BeanLifecycleService3 was initialized
2019-09-29 00:20:00.914 INFO 5208 --- [
2019-09-29 00:20:01.356 INFO 5208 --- [
2019-09-29 00:20:01.360 INFO 5208 --- [
BeanLifecycleService1 was initialized
BeanLifecycleService2 was initialized
BeanLifecycleService3 was initialized
Name : Zara
```

## 2.2 @Autowired

- @Autowired là cách inject tự động vào Bean, khác với @Required chỉ hỗ trợ setter method thì @Autowired được hỗ trợ rộng hơn: setter method, constructor, property

### @Autowired on Setter Methods

#### *TextEditor.java*

```
public class TextEditor {

    private SpellChecker spellChecker;

    @Autowired
    public void setSpellChecker( SpellChecker spellChecker ){
        this.spellChecker = spellChecker;
    }

    public SpellChecker getSpellChecker( ) {
        return spellChecker;
    }

    public void spellCheck() {
        spellChecker.checkSpelling();
    }
}
```

#### **SpellChecker.java:**

```
public class SpellChecker {

    public SpellChecker(){
        System.out.println("Inside SpellChecker constructor." );
    }

    public void checkSpelling(){
        System.out.println("Inside checkSpelling." );
    }
}
```

## MainApp.java

```
public class MainApp {  
  
    public static void main(String[] args) {  
  
        ApplicationContext context = new  
ClassPathXmlApplicationContext("Beans.xml");  
  
        TextEditor text = (TextEditor) context.getBean("textEditor");  
  
        text.spellCheck();  
    }  
}
```

## Beans.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:annotation-config/>  
  
    <!-- Definition for textEditor bean without constructor-arg -->  
    <bean id = "textEditor" class = " spring.example.TextEditor">  
    </bean>  
  
    <!-- Definition for spellChecker bean -->  
    <bean id = "spellChecker" class = "spring.example.SpellChecker">  
    </bean>  
  
</beans>
```

## Kết quả

```
Inside SpellChecker constructor.  
Inside checkSpelling.
```

## @Autowired on Property

### TextEditor.java

```
public class TextEditor {

    @Autowired
    private SpellChecker spellChecker;

    public TextEditor() {
        System.out.println("Inside TextEditor constructor." );
    }

    public SpellChecker getSpellChecker( ){
        return spellChecker;
    }

    public void spellCheck(){
        spellChecker.checkSpelling();
    }

}
```

### Beans.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:annotation-config/>

    <!-- Definition for textEditor bean -->
    <bean id = "textEditor" class = "spring.example.TextEditor">
    </bean>

    <!-- Definition for spellChecker bean -->
    <bean id = "spellChecker" class = "spring.example.SpellChecker">
    </bean>

</beans>
```

### Kết quả

```
Inside TextEditor constructor.
Inside SpellChecker constructor.
Inside checkSpelling.
```

## @Autowired on Constructor

## TextEditor.java

```
public class TextEditor {
    private SpellChecker spellChecker;

    @Autowired
    public TextEditor(SpellChecker spellChecker){
        System.out.println("Inside TextEditor constructor." );
        this.spellChecker = spellChecker;
    }
    public void spellCheck(){
        spellChecker.checkSpelling();
    }
}
```

## Beans.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:annotation-config/>

    <!-- Definition for textEditor bean without constructor-arg -->
    <bean id = "textEditor" class = "spring.example.TextEditor">
    </bean>

    <!-- Definition for spellChecker bean -->
    <bean id = "spellChecker" class = "spring.example.SpellChecker">
    </bean>

</beans>
```

## Kết Quả

```
Inside SpellChecker constructor.
Inside TextEditor constructor.
Inside checkSpelling.
```



## 2.2 @Qualifier

Trường hợp khi tạo nhiều Bean có cùng 1 kiểu dữ liệu, thì khi autowired sẽ có vấn đề phát sinh là Không biết wiring vào Bean nào. Khi đó phải dùng **@Qualifier** để xác định Bean theo name

### Student.java

```
public class Student {  
  
    private Integer age;  
    private String name;  
  
    public void setAge(Integer age) {  
        this.age = age;  
    }  
    public Integer getAge() {  
        return age;  
    }  
    public void setName(String name) {  
        this.name = name;  
    }  
    public String getName() {  
        return name;  
    }  
}
```

### Profile.java

```
public class Profile {  
  
    @Autowired  
    @Qualifier("student1")  
    private Student student;  
  
    public Profile(){  
        System.out.println("Inside Profile constructor." );  
    }  
    public void printAge() {  
        System.out.println("Age : " + student.getAge() );  
    }  
    public void printName() {  
        System.out.println("Name : " + student.getName() );  
    }  
}
```

## MainApp.java

```
public class MainApp {  
    public static void main(String[] args) {  
        ApplicationContext context = new  
        ClassPathXmlApplicationContext("Beans.xml");  
  
        Profile profile = (Profile) context.getBean("profile");  
        profile.printAge();  
        profile.printName();  
    }  
}
```

## Beans.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-3.0.xsd  
http://www.springframework.org/schema/context  
http://www.springframework.org/schema/context/spring-context-3.0.xsd">  
  
    <context:annotation-config/>  
  
    <!-- Definition for profile bean -->  
    <bean id = "profile" class = "com.tutorialspoint.Profile"></bean>  
  
    <!-- Definition for student1 bean -->  
    <bean id = "student1" class = "com.tutorialspoint.Student">  
        <property name = "name" value = "Zara" />  
        <property name = "age" value = "11"/>  
    </bean>  
  
    <!-- Definition for student2 bean -->  
    <bean id = "student2" class = "com.tutorialspoint.Student">  
        <property name = "name" value = "Nuha" />  
        <property name = "age" value = "2"/>  
    </bean>  
  
</beans>
```

### Kết quả

```
Inside Profile constructor.  
Age : 11  
Name : Zara
```

### ❖ Tài liệu tham khảo

<https://docs.spring.io/spring/docs/3.0.0.M3/reference/html/ch04s11.html>