06: ♥ Debugging NoSuchBeanDefinitionException in Spring

Posted on September 7, 2015 by Arulkumaran Kumaraswamipillai

#1. Adding the @Named annotation to the interface instead of the implementation

Adding the @Named, @Component, @Repository, etc annotation on the interface as opposed to the implementation class. The code below can throw "NoSuchBeanDefinitionException"

```
1 package com.mytutorial;
2
3 @Named
4 public interface SimpleDao {
5     abstract String getNameById(int id);
6 }
```

```
1 package com.mytutorial;
2
3 import javax.inject.Named;
```

```
4
5 public class SimpleDaoImpl implements SimpleDao {
6     public String getNameById(int id) {
7          return "test"
8     }
9 }
```

FIX: The @Named annotation is now in the implementation class.

```
1 package com.mytutorial;
2
3 public interface SimpleDao {
4     abstract String getNameById(int id);
5 }
```

```
1  package com.mytutorial;
2
3  import javax.inject.Named;
4
5  @Named
6  public class SimpleDaoImpl implements SimpleDao
7      public String getNameById(int id) {
8           return "test"
9      }
10 }
```

#2. Not configuring the component-scan in the XML or JavaConfig

NOT having the “component-scan” in the XML based configuration

```
1 <context:component-scan base-package="com.mytutor
```

or NOT having the “@ComponentScan” in the Java based configuration

```
1  package com.mytutorial;
2
3  import org.springframework.context.annotation.Co
4  import org.springframework.context.annotation.Co
5
6  @Configuration
7  @ComponentScan(basePackages={"com.mytutorial"})
8  public class AppConfig {
9
10 }
```


16 Technical Key Areas

80+ step by step Java Tutorials

Even if the base package is defined incorrectly you can get this exception.

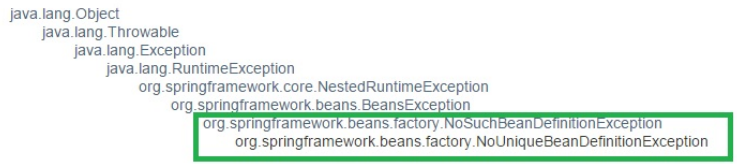
#3. Trying to @Inject or @Autowire a bean that has NOT been defined

For example, if "SimpleDaoImpl" bean was not defined with the @Commponent, @Repository or @Named annotations. Trying to inject a bean (e.g. SimpleDao) that has not been defined at all or annotated can throw this exception.

```
1  package com.mytutorial;
2
3  import javax.inject.Inject;
4  import javax.inject.Named;
5
6  @Named
7  public class SimpleServiceImpl implements Simple
8
9      @Inject
10     SimpleDao dao;
11
12     public String processUser(int id) {
13         String name = dao.getNameById(id);
14         return "Hello " + name;
15     }
16 }
```

#4. Having more than one bean being defined in the context throws "NoUniqueBeanDefinitionException"

which is a subclass of "NoSuchBeanDefinitionException". If you have more than one bean defined, then you need to use the "@Qualifier" annotation to specify which bean you need to use.



Spring NoSuchBeanDefinitionException

open all | close all
Setting up Tutorial (6)
Tutorial - Diagnosis (2
Akka Tutorial (9)
Core Java Tutorials (2
Hadoop & Spark Tuto
JEE Tutorials (19)
Scala Tutorials (1)
Spring & HIbernate Tu
Tools Tutorials (19)
Other Tutorials (45)

100+ Java pre-interview coding tests

open all | close all
Can you write code? (
♦ Complete the given
Converting from A to I
Designing your classe
Java Data Structures
Passing the unit tests
What is wrong with th
Writing Code Home A
Written Test Core Jav
Written Test JEE (1)

How good are your .....?

open all | close all
Career Making Know-
Job Hunting & Resum

FIX: Qualifying the beans with the "**@Qualifier**" annotation.

```
1  package com.mytutorial;
2
3  import javax.inject.Inject;
4  import javax.inject.Named;
5
6  import org.springframework.beans.factory.annotat
7
8  @Named
9  public class SimpleServiceImpl implements Simple
10
11     @Inject
12     @Qualifier(value="simpleDao")
13     SimpleDao dao;
14
15     public String processUser(int id) {
16         String name = dao.getNameById(id);
17         return "Hello " + name;
18     }
19 }
```

As you can see the the qualifier value is "simpleDao", and the dao beans will be defined with @Named values as shown below.

```
1  package com.mytutorial;
2
3  import javax.inject.Named;
4
5  @Named("simpleDao")
6  public class SimpleDaoImpl implements SimpleDao
7
8      public String getNameById(int id) {
9          //......
10     }
11
12 }
```

and the another one as

```
1  package com.mytutorial;
2
3  import javax.inject.Named;
4
5  @Named("anotherSimpleDao")
6  public class AnotherSimpleDaoImpl implements Sim
7
8      public String getNameById(int id) {
9          //....
10     }
11 }
```

as you can see the both implement the same type "SimpleDao", but have the different @Named values "simpleDao" and "anotherSimpleDao".

#5. Opposite of #1 when a bean in the context is proxied using the JDK Dynamic Proxy mechanism

The Spring documentation states that

"Spring recommends that you only annotate concrete classes (and methods of concrete classes) with the @Transactional annotation, as opposed to annotating interfaces. You certainly can place the @Transactional annotation on an interface (or an interface method), but this works only as you would expect it to if you are using interface-based proxies."

When you see a class with a name like **$Proxy31**, it's indicating that that Spring has auto-generated a proxy object for one of your beans say "SimpleDao". This uses java.lang.reflect.Proxy to generate the proxy objec t. This proxy object will implement the same interface(s) as the class that's being proxied, but it will not be **type compatible** with the target class itself. So, in the is scenario, if a bean is injected by an interface, it will be correctly wired in. If a bean is injected by the actual class, then Spring will not find the bean definition, and throws "NoSuchBeanDefinitionException".

Popular Member Posts

♦ 11 Spring boot interview questions & answers

904 views

♦ Q11-Q23: Top 50+ Core on Java OOP Interview Questions & Answers

816 views

001A: ♦ 7+ Java integration styles & patterns interview questions & answers

427 views

18 Java scenarios based interview Questions and Answers

408 views

♦ 7 Java debugging interview questions & answers

323 views

01b: ♦ 13 Spring basics Q8 – Q13 interview questions & answers

311 views

01: ♦ 15 Ice breaker questions asked 90% of the time in Java job interviews with hints

303 views

♦ 10 ERD (Entity-Relationship Diagrams) Interview Questions and Answers

301 views

♦ Q24-Q36: Top 50+ Core on Java classes, interfaces and generics interview questions & answers

251 views

001B: ♦ Java architecture & design concepts interview questions & answers

209 views

Bio | Latest Posts

Arulkumaran Kumaraswamipillai

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via Amazon.com in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site.**945+** paid members. join my LinkedIn Group. **Reviews**

About Arulkumaran Kumaraswamipillai

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via Amazon.com in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site.**945+** paid members. join my LinkedIn Group. **Reviews**

‹ 11 FAQ XSD interview questions and answers

CDI annotations @Named and @Inject Dependency Injection in Spring 3.0 Tutorial ›

Posted in Java Debugging, Spring

Empowers you to open more doors, and fast-track

Technical Know Hows

☀ Java generics in no time ☀ Top 6 tips to transforming your thinking from OOP to FP ☀ How does a HashMap internally work? What is a hashing function?
☀ 10+ Java String class interview Q&As ☀ Java auto un/boxing benefits & caveats ☀ Top 11 slacknesses that can come back and bite you as an experienced Java developer or architect

Non-Technical Know Hows

☀ 6 Aspects that can motivate you to fast-track your career & go places ☀ Are you reinventing yourself as a Java developer? ☀ 8 tips to safeguard your Java career against offshoring ☀ My top 5 career mistakes

Prepare to succeed

☀ Turn readers of your Java CV go from “Blah blah” to “Wow”? ☀ How to prepare for Java job interviews? ☀ 16 Technical Key Areas ☀ How to choose from multiple Java job offers?

Select Category ▼

© Disclaimer

 ↑