Spring @Qualifier 注释

可能会有这样一种情况，当你创建多个具有相同类型的 bean 时，并且想要用一个属性只为它们其中的一个进行装配，在这种情况下，你可以使用 **@Qualifier** 注释和 **@Autowired** 注释通过指定哪一个真正的 bean 将会被装配来消除混乱。下面显示的是使用 @Qualifier 注释的一个示例。

示例

让我们使 Eclipse IDE 处于工作状态，请按照下列步骤创建一个 Spring 应用程序：

步骤	描述
1	创建一个名为 *SpringExample* 的项目，并且在所创建项目的 **src** 文件夹下创建一个名为 *com.tutorialspoint* 的包。
2	使用 *Add External JARs* 选项添加所需的 Spring 库文件，就如在 *Spring Hello World Example* 章节中解释的那样。
3	在 *com.tutorialspoint* 包下创建 Java 类 *Student*，*Profile* 和 *MainApp*。
4	在 **src** 文件夹下创建 Beans 配置文件 *Beans.xml*。
5	最后一步是创建所有 Java 文件和 Bean 配置文件的内容，并且按如下解释的那样运行应用程序。

这里是 **Student.java** 文件的内容：

```
package com.tutorialspoint;
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;
   }
}
```

这里是 **Profile.java** 文件的内容：

```
package com.tutorialspoint;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Qualifier;
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() );
   }
}
```

下面是 **MainApp.java** 文件的内容：

```
package com.tutorialspoint;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
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();
   }
}
```

考虑下面配置文件 **Beans.xml** 的示例：

```
<?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>
```

一旦你在源文件和 bean 配置文件中完成了上面两处改变，让我们运行一下应用程序。如果你的应用程序一切都正常的话，这将会输出以下消息：

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