# 实验6 bean的装配

## 实验目的

通过基于annotation的装配方式装配bean实例，来达到熟练掌握Spring中bean的装配方式的目的。

## 实验内容

定义一个学生类Student，在类中定义(sno,sname,major(专业)）三个属性，用三层架构模拟添加学生的操作，用注解方式完成bean的装配，在应用程序中获取bean对象，并调用方法完成模拟添加学生。

## 实验代码：

### applicationContext.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="com.shf"*/>  
  
</*beans*>*

Student.java

package com.shf.pojo;  
  
import lombok.AllArgsConstructor;  
import lombok.Data;  
import lombok.NoArgsConstructor;  
import org.springframework.beans.factory.annotation.Value;  
import org.springframework.context.annotation.Scope;  
import org.springframework.stereotype.Component;  
  
@Data  
@AllArgsConstructor  
@NoArgsConstructor  
@Component*(*"student"*)*@Scope*(*"singleton"*)*public class Student *{* @Value*(*"021312"*)* private Integer sno;  
 @Value*(*"张三"*)* private String sname;  
 @Value*(*"软件工程"*)* private String major;  
*}*

### StudentDao.java

package com.shf.dao;  
  
import com.shf.pojo.Student;  
  
public interface StudentDao *{* public void showStudentInfo*()*;  
  
 public void addStudent*(*Student student*)*;  
*}*

### StudentDaoImpl.java

package com.shf.dao.impl;  
  
import com.shf.dao.StudentDao;  
import com.shf.pojo.Student;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
import org.springframework.stereotype.Repository;  
  
@Repository*(*"StudentDao"*)*public class StudentDaoImpl implements StudentDao *{* @Override  
 public void showStudentInfo*() {* ClassPathXmlApplicationContext applicationContext = new ClassPathXmlApplicationContext*(*"applicationContext.xml"*)*;  
 Student s1 = *(*Student*)* applicationContext.getBean*(*"student"*)*;  
 System.*out*.println*(*s1*)*;  
 *}* @Override  
 public void addStudent*(*Student student*) {* ClassPathXmlApplicationContext applicationContext = new ClassPathXmlApplicationContext*(*"applicationContext.xml"*)*;  
 Student s2 = *(*Student*)* applicationContext.getBean*(*"student"*)*;  
 System.*out*.println*(*s2*)*;  
 *}  
}*

### StudentService.java

package com.shf.service;  
  
import com.shf.pojo.Student;  
  
public interface StudentService *{* public void showStudentInfo*()*;  
  
 public void addStudent*(*Student student*)*;  
*}*

### StudentServiceImpl.java

package com.shf.service.impl;  
  
import com.shf.dao.StudentDao;  
import com.shf.pojo.Student;  
import com.shf.service.StudentService;  
import org.springframework.stereotype.Service;  
  
import javax.annotation.Resource;  
  
@Service*(*"studentService"*)*public class StudentServiceImpl implements StudentService *{* @Resource*(*name="StudentDao"*)* private StudentDao studentDao;  
  
 @Override  
 public void showStudentInfo*() {* studentDao.showStudentInfo*()*;  
 *}* @Override  
 public void addStudent*(*Student student*) {* studentDao.addStudent*(*student*)*;  
 *}  
}*

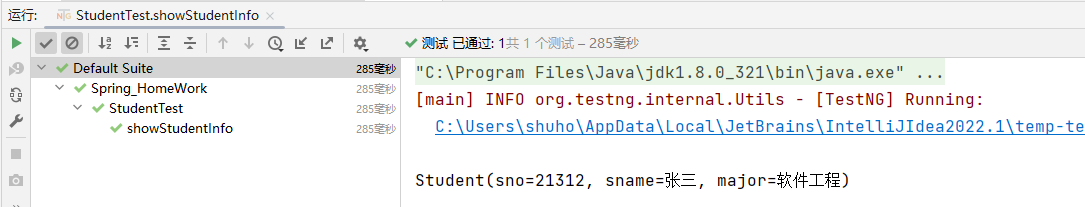
### StudentController.java

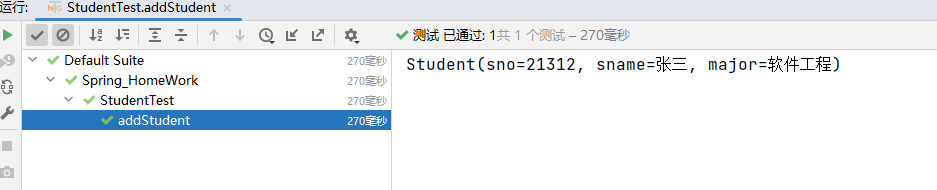
package com.shf.controller;  
  
import com.shf.pojo.Student;  
import com.shf.service.StudentService;  
import org.springframework.stereotype.Controller;  
  
import javax.annotation.Resource;  
  
@Controller*(*"StudentController"*)*public class StudentController *{* @Resource*(*name="studentService"*)* private StudentService studentService;  
  
 public void showStudentInfo*(){* studentService.showStudentInfo*()*;  
 *}* public void addStudent*(*Student student*){* studentService.addStudent*(*student*)*;  
 *}  
}*

### StudentTest.java

package com.shf.test;  
  
import com.shf.controller.StudentController;  
import com.shf.pojo.Student;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
import org.testng.annotations.Test;  
  
@Test  
public class StudentTest *{* @Test  
 public void showStudentInfo*(){* ClassPathXmlApplicationContext applicationContext = new ClassPathXmlApplicationContext*(*"applicationContext.xml"*)*;  
  
 StudentController studentController = *(*StudentController*)* applicationContext.getBean*(*"StudentController"*)*;  
  
 studentController.showStudentInfo*()*;  
 *}* @Test  
 public void addStudent*(){* ClassPathXmlApplicationContext applicationContext = new ClassPathXmlApplicationContext*(*"applicationContext.xml"*)*;  
  
 StudentController studentController = *(*StudentController*)* applicationContext.getBean*(*"StudentController"*)*;  
  
 Student student = new Student*(*20220418, "张三", "软件工程"*)*;  
  
 studentController.addStudent*(*student*)*;  
 *}  
}*

## 实验截图：





## 实验小结

通过本次实验我学会通过注解的方式将Bean注册到Spring容器中，并通过XML文件创建spring对象，通过spring对象对bean进行管理。