# 实验四 Spring的IoC反转控制

一、实验目的：

理解Spring的IoC容器工作原理，通过该容器管理各组件之间的依赖关系来降低组件之间的耦合度。

二、实验内容：

在MyEclipse环境下创建Java项目实现反转控制，调试运行程序。

三、实验要求：

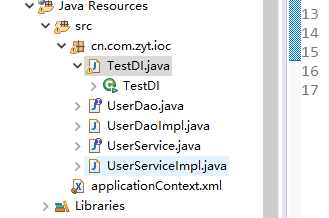
1. 熟悉并掌握创建Java项目的基本步骤。

2. 运用依赖注入方式实现反转控制。

3. 写出实验报告。

四、实验步骤：

1．进入MyEclipse环境，新建一个Java Project。



2. 导入Spring Java包。

3. 编写程序，并给出关键代码及注释。

public class TestDI {

public static void main(String[] args) {

//1.初始化spring容器，加载配置文件

ApplicationContext applicationContext =

new ClassPathXmlApplicationContext("applicationContext.xml");

//2.通过容器获取UserService实例

UserService userService =

(UserService) applicationContext.getBean("userService");

//3.调用实例中的say()方法

userService.say();

}

}

<?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-4.3.xsd"*>

<!-- 将指定类配置给Spring，让Spring创建其对象的实例 -->

<bean id=*"userDao"* class=*"cn.com.zyt.ioc.UserDaoImpl"* />

<!--添加一个id为userService的实例 -->

<bean id=*"userService"* class=*"cn.com.zyt.ioc.UserServiceImpl"*>

<!-- 将id为userDao的Bean实例注入到userService实例中 -->

<property name=*"userDao"* ref=*"userDao"* /> </bean>

</beans>

**package** cn.com.zyt.ioc;

**public** **class** UserServiceImpl **implements** UserService

{

// 声明UserDao属性

**private** UserDao userDao;

// 添加UserDao属性的setter方法，用于实现依赖注入

**public** **void** setUserDao(UserDao userDao)

{

**this**.userDao = userDao;

}

// 实现的接口中方法

**public** **void** say()

{

// 调用userDao中的say()方法，并执行输出语句

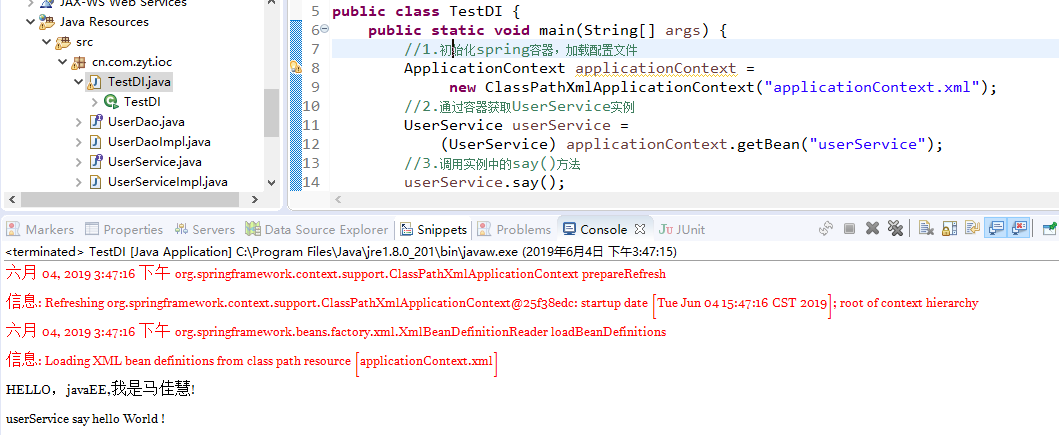
**this**.userDao.say();

System.***out***.println("userService say hello World !");

}

}

4. 程序运行截图。



五、心得体会：

这次的实验参考了老师课上给的案例源码，完成的比较顺利。之前上过云计算的课，也做过相似的联系，这次的实验更加巩固了基础，更进一步理解了Spring的一些原理，之后还得多练习，希望之后的大作业可以熟练运用。