# 实验六：依赖注入

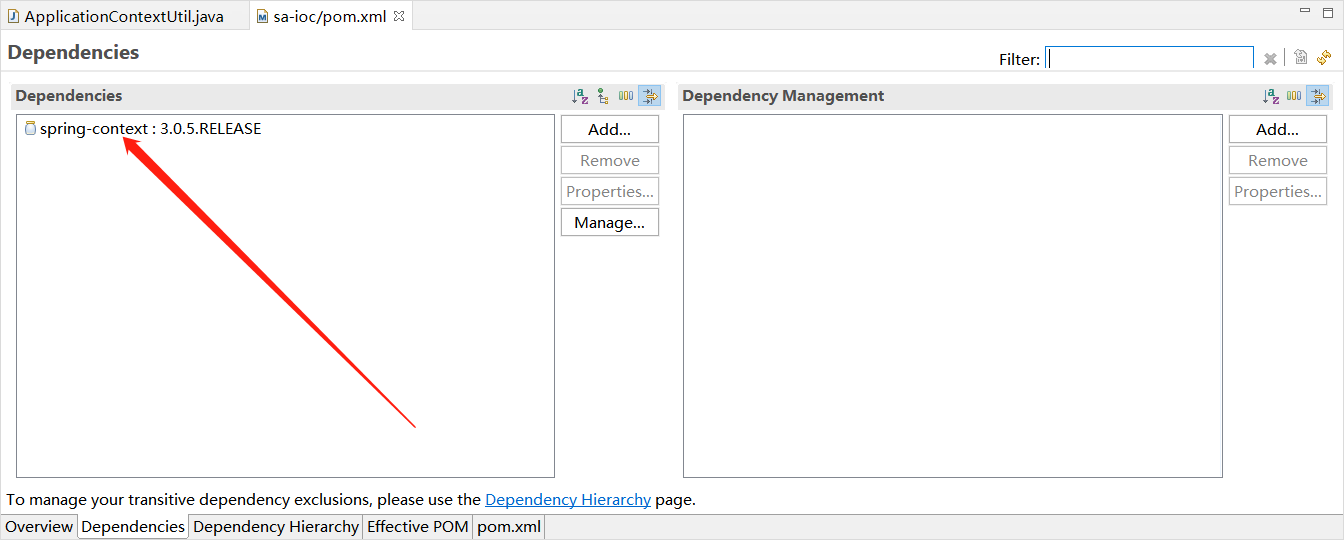
## 实验内容

同学们(Student)的体系结构课程的老师(Teacher)是“梁早清”。这些是同学们上课之前就确定了的。

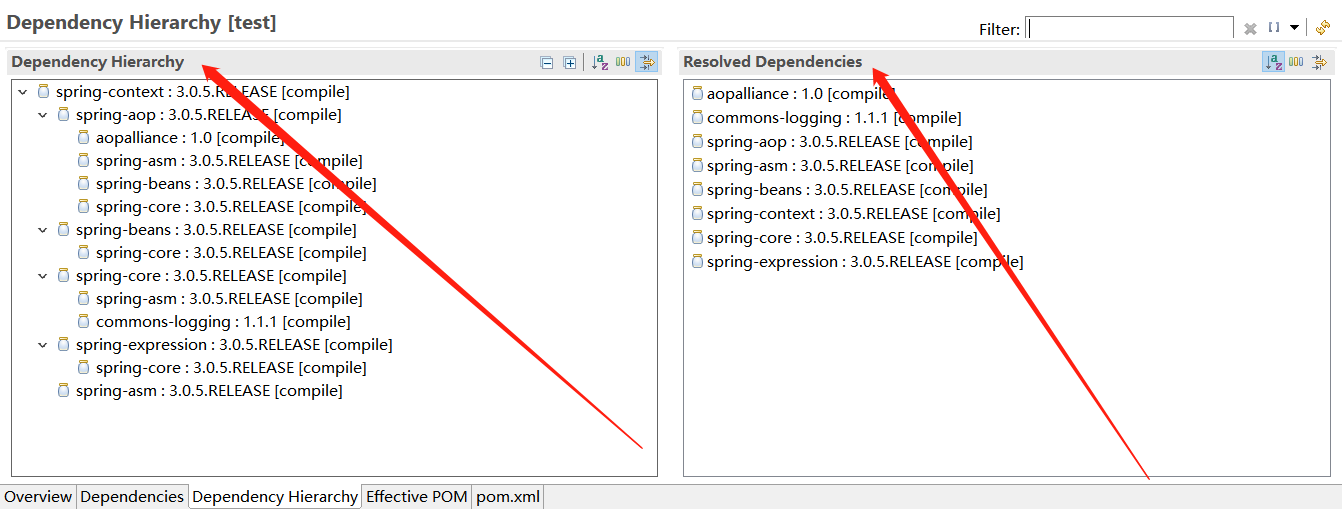
利用Spring框架，配置好以上的2个基础对象，其中学生是你自己，老师是梁早清。测试客户端输出你的体系结构课程老师的名字。

因为Spring的ApplicationContext对象是一个大对象，并且需要保证应用中使用单例，采用ApplicationContextUtil类。

项目的Spring框架依赖建议采用Mavan管理，本实验只需要org.springframework的spring-context的jar文件即可，Maven会自动导入相关的其他jar文件，关于maven，请同学们自行参考相关文档。



项目的依赖jar文件



项目的自动导入的所有依赖jar文件

## 依赖注入

### 核心代码

Bean的配置

<?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:p="http://www.springframework.org/schema/p" xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-3.0.xsd">

<bean id="teacherBean" class="cn.edu.scau.sec.lingjielin.Teacher">

<property name="name" value="梁早清"></property>

</bean>

<bean id="studentBean" class="cn.edu.scau.sec.lingjielin.Student">

</bean>

</beans>

Student核心代码

public class Student {

public Teacher teacher;

private String name;

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public Student(){}

public Student setTeacher(Teacher teacher) {

this.teacher = teacher;

return this;

}

public void displayTeacher() {

System.out.println(this.name + "的老师名字是" + this.teacher.name);

}

}

Teacher核心代码

public class Teacher {

String name;

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

}

客户端核心代码

public class Client{

public static void main(String[] args) {

ApplicationContext context = ApplicationContextUtil.getInstance();

Teacher techaerBeanTeacher = (Teacher)context.getBean("teacherBean");

Student studentBeaStudent = (Student)context.getBean("studentBean");

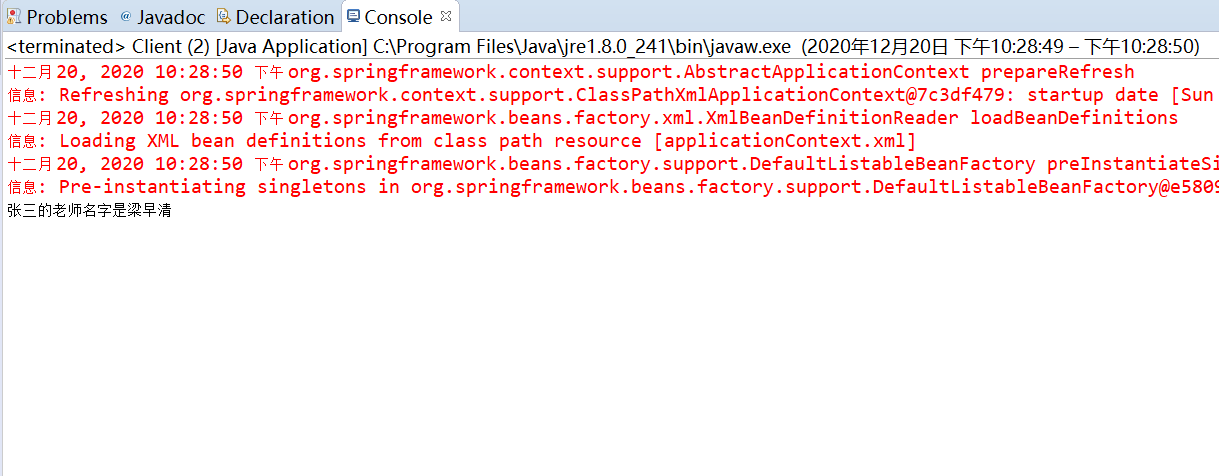
studentBeaStudent.setName("张三");

studentBeaStudent.setTeacher(techaerBeanTeacher).displayTeacher();;

}

}

### 实现效果



## 实验体会

1. 为什么Spring框架在Java应用中被如此广泛地使用？

帮助我们专注于核心任务，而不是与之相关的样板；以设计模式的形式汇集了多年的智慧；帮助我们遵守行业和监管标准；降低应用程序的总体拥有成本，简化了Java的开发工作。

1. 在客户端中为什么要使用ApplicationContextUtil获取ApplicationContext单例，而不是直接采用new ApplicationContext()的方式创建？

因为我们要经常使用这个ApplicationContext的实例化对象，假如每使用一次就new一个对象，就会造成内存的浪费，这个对象本身就是可以复用的，我们直接采用单例模式提供一个唯一的实例化对象更加符合操作逻辑。

1. 在应用中集成使用Spring框架，最主要的需要考虑什么？

便于项目的管理和开发，例如A要使用到B类，我们通常时new 一个B类的对象，但是这种方式不利于扩展，采用Spring的依赖注入就可以解决这个问题。使得我们可以专注核心业务，提高开发的效率。