**一、Spring 技术基础知识**

**1、Bean工厂（用来实例化对象）**

Spring中，存在一个 Bean工厂。我们把每一个java类当做是一个 bean（即豌豆），Spring就可以当做是一个factory（工厂），bean factory（豌豆工厂）的功能就是专门生产bean的。也就是说：Spring 可以去生产类的对象，也即 实例化类对象（new 类名();）。

**2、IOC/DI 控制反转（不用维护类与类之间的依赖性，交给Bean工厂去实例化）**

从新建一个Dynamic Web project开始，在工程项目中，引入 最新版本的 Spring jar包，配置使用Spring，熟悉Spring框架在项目中，所起到的作用。重点学习了解IOC/DI，即“控制反转、依赖注入”

简单的说清楚，IOC产生的原因，及解决的问题。

在“三层结构”(表示层、业务逻辑层、数据访问层)架构开发中，层与层之间，类有调用依赖的关系

表示层——>业务逻辑层——>数据访问层，表示层类中，需要调用业务逻辑层类的方法；业务逻辑层类中，需要调用数据访问层类的方法。

以用户登录、注册、用户个人资料维护为例：设计3个类 UserController， UserService， UserDao ，分别对应“三层结构”的表示层、业务逻辑层、数据访问层。

那么，在项目实际开发代码中，存在：

表示层类 UserController 中，要引用 UserService：



我们会发现 类与类之间，有很紧密的依赖关系，即：在代码里，一个类中，引用了另外一个类，并 new 了一个对象。

这样，也就意味着有很强的耦合性。而这样，是不建议的。

因为软件设计中，很强调的是设计出来的软件需要具有很好的“特性：“高内聚、松耦合”。

为了解耦，降低这种很强的依赖性，Spring 框架中，设计出了 IOC。

代码中，不去 维护类与类之间的依赖性，也即： UserController 中不去new UserService()

而是去这样写：



那么 实例化UserService，即 new UserService();在哪里操作？

交给Spring 的bean工厂，去实例化。

如何去实例化类对象，有哪些需要约定，这就需要一个 xml配置文件去记录。后来，可以使用annotation（注解）的方式去配置属性。

spring配置文件：



**3、AOP**

面向切面编程，主要解决横切性的问题。

什么是横切性的问题？比如开发中，有很多的类、很多的方法，类与类之间存在调用的依赖的关系，我们称之为“从上而下”的线性调用。在这些代码中，经常需要在很多位置，添加“打印日志”的代码。而这些，“打印日志”的代码，基本都是一样的，和“从上而下”的线性调用，没有什么直接的业务逻辑关系。我们可以称之为：横切到这个“从上而下”的线性中。就像一个“十字”、“垂直”、“正交”这样。

除了“日志”属于横切性问题，“事务”也属于。

AOP就是为了解决这种横切性的问题，通过配置，不让这些相同的代码，充斥在项目代码的各处。而是通过，很少的配置，把这些相似的横切性代码，配置到它们应该出现的位置。

AOP也需要了解一些，专门的术语，我们这里只是简单的介绍一下，AOP需要说清楚，还要写专门的文章，去举例和描述。

**4.IOC概念理解**

IoC 是一个概念，是一种思想，其实现方式多种多样。当前比较流行的实现方式有两种：

依赖注入和依赖查找。依赖注入方式应用更为广泛。

依赖注入 DI 是指程序运行过程中，若需要调用另一个对象协助时，无须在代码中创建

被调用者，而是依赖于外部容器，由外部容器创建后传递给程序。

5.AOP理解

面向切面编程，就是将交叉业务逻辑封装成切面，利用 AOP 容器的功能将切面织入到

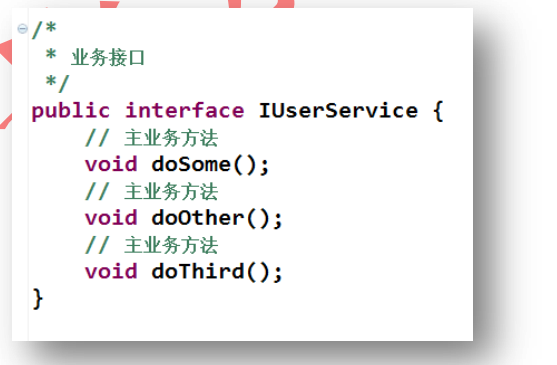
主业务逻辑中。所谓交叉业务逻辑是指，通用的、与主业务逻辑无关的代码，如安全检查、

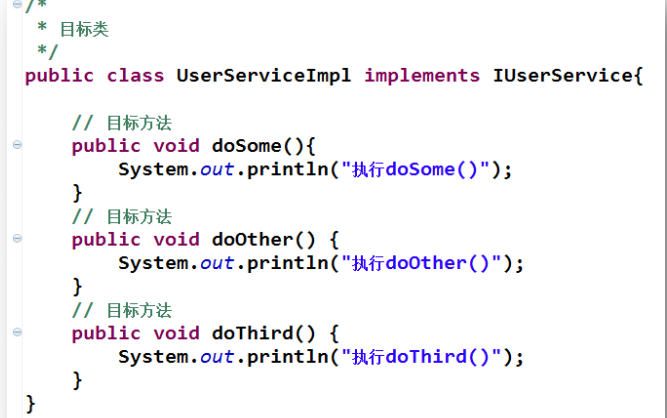
事务、日志等。

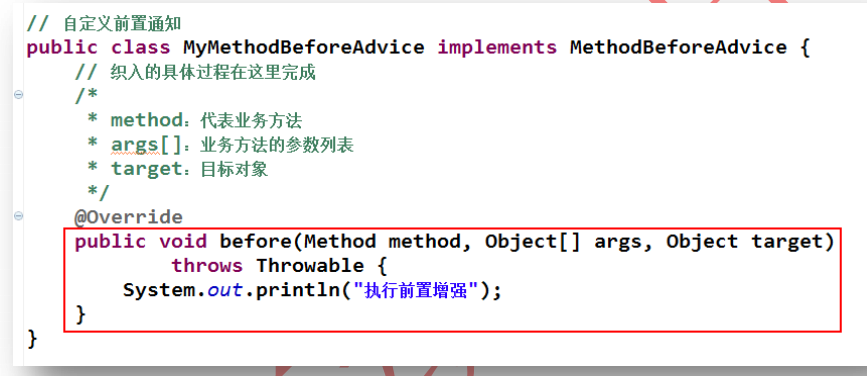
若不使用 AOP，则会出现代码纠缠，即交叉业务逻辑与主业务逻辑混合在一起。这样，

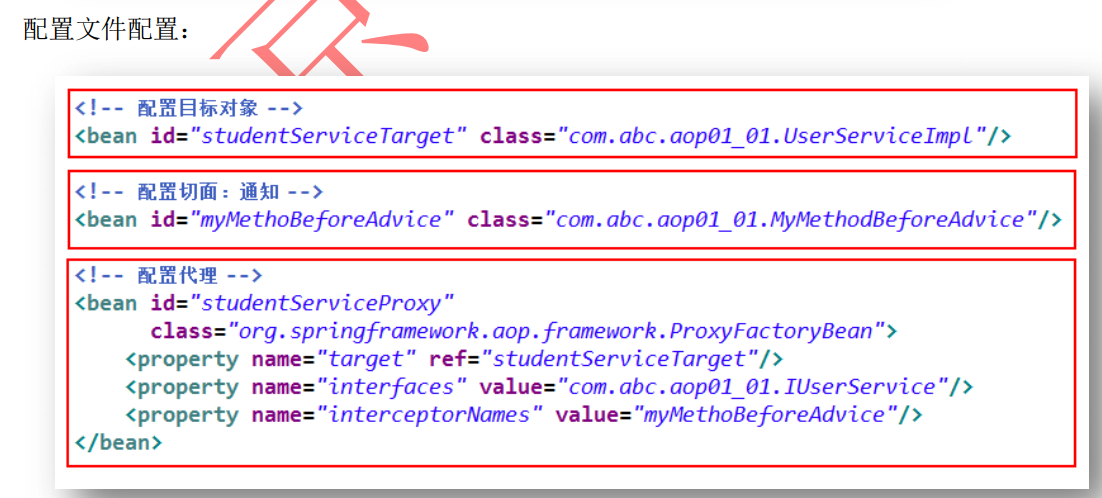
会使主业务逻辑变的混杂不清。

切面的两种实现方式：通知（Advice）和顾问（Advisor）











**二、Spring Boot四大核心**

Spring Boot将很多魔法带入了Spring应用程序的开发之中，其中最重要的是以下四个核心。  
❑ 自动配置：针对很多Spring应用程序常见的应用功能， Spring Boot能自动提供相关配置。  
❑ 起步依赖：告诉Spring Boot需要什么功能，它就能引入需要的库。  
❑ 命令行界面：这是Spring Boot的可选特性，借此你只需写代码就能完成完整的应用程序，无需传统项目构建。  
❑ Actuator：让你能够深入运行中的Spring Boot应用程序，一探究竟。

**三、Spring Boot项目结构**

如你所见，项目里基本没有代码，除了几个空目录外，还包含了如下几样东西。

①build.gradle： Gradle构建说明文件。如果选择Maven项目，这就会换成pom.xml。

②Application.java：一个带有main()方法的类，用于引导启动应用程序。

③ApplicationTests.java：一个空的JUnit测试类，它加载了一个使用Spring Boot自动

配置功能的Spring应用程序上下文。

④application.properties：一个空的properties文件，你可以根据需要添加配置属性。

