**Spring实战**

**第一章**

1、Enterprise Java Bean：EJB，企业级Java Bean

Java Data Object：JDO，Java数据对象

Plain Old Java Object：POJO，简单老式Java对象

Dependency Injection：DI，依赖注入

Aspect-Oriented Programming：AOP，面向切面编程

JavaBean来表示应用组件

2、为了降低Java开发的复杂性，Spring采取了以下4种关键策略：

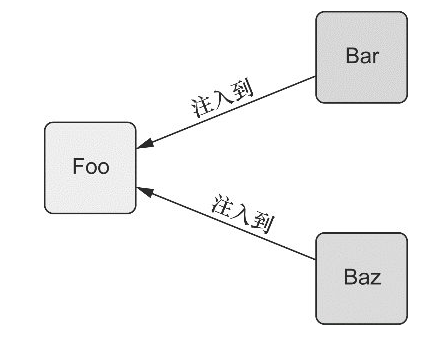
基于POJO的轻量级和最小侵入性编程；

通过依赖注入和面向接口实现松耦合；

基于切面和惯例进行声明式编程；

通过切面和模板减少样板式代码。

3、通过DI，对象的依赖关系将由系统中负责协调各对象的第三方组件在创建对象的时候进行设定。对象无需自行创建或管理它们的依赖关系，如图1.1所示，依赖关系将被自动注入到需要它们的对象当中去。

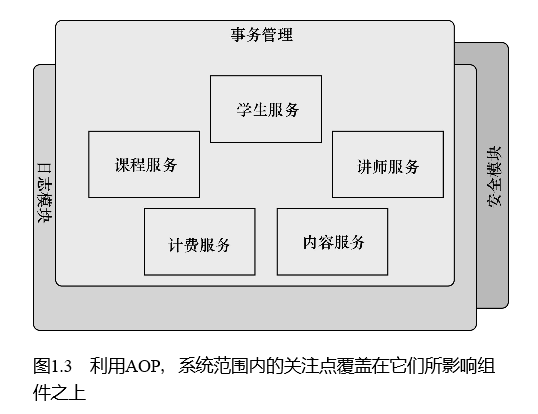
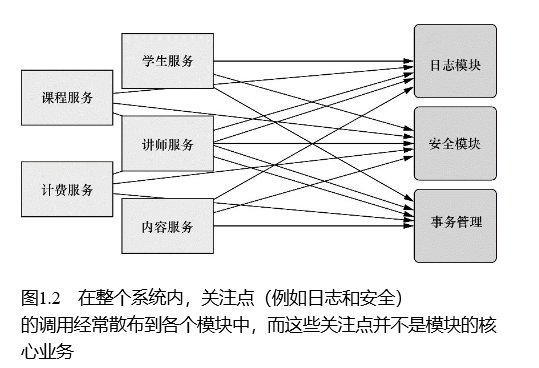


4、创建应用组件之间协作的行为通常称为装配（wiring）。Spring有多种装配bean的方式，采用XML是很常见的一种装配方式。

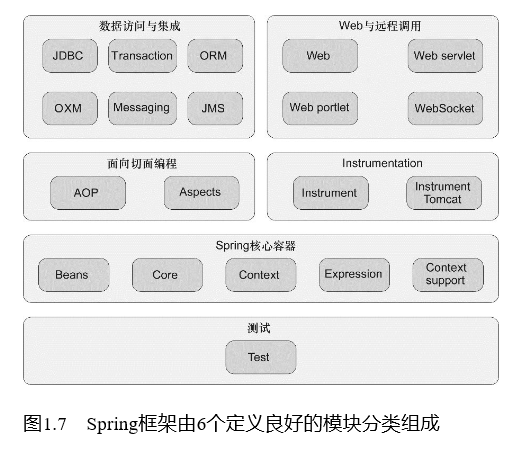
5、Spring通过应用上下文（Application Context）装载bean的定义并把它们组装起来。Spring应用上下文全权负责对象的创建和组装。Spring自带了多种应用上下文的实现，它们之间主要的区别仅仅在于如何加载配置。XML和Java配置。

6、DI能够让相互协作的软件组件保持松散耦合，而面向切面编程（aspect-oriented programming，AOP）允许你把遍布应用各处的功能分离出来形成可重用的组件。

面向切面编程往往被定义为促使软件系统实现关注点的分离一项技术。系统由许多不同的组件组成，每一个组件各负责一块特定功能。除了实现自身核心的功能之外，这些组件还经常承担着额外的职责。诸如日志、事务管理和安全这样的系统服务经常融入到自身具有核心业务逻辑的组件中去，这些系统服务通常被称为**横切关注点**，因为它们会跨越系统的多个组件。



7、让我们再了解下Spring容器，这是应用中的所有bean所驻留的地方。在基于Spring的应用中，你的应用对象生存于Spring容器（container）中。如图1.4所示，Spring容器负责创建对象，装配它们，配置它们并管理它们的整个生命周期，从生存到死亡（在这里，可能就是new到finalize()）。Spring自带了多个容器实现，可以归为两种不同的类型。bean工厂（由org.springframework.beans.factory.BeanFactory接口定义）是最简单的容器，提供基本的DI支持。应用上下文（由org.springframework.context.ApplicationContext接口定义）基于BeanFactory构建，并提供应用框架级别的服务，例如从属性文件解析文本信息以及发布应用事件给感兴趣的事件监听者。



**第2章　装配Bean**

创建应用对象之间关联关系的传统方法（通过构造器或者查找）通常会导致结构复杂的代码，这些代码很难被复用也很难进行单元测试。

1、创建应用对象之间协作关系的行为通常称为装配（wiring），这也是依赖注入（DI）的本质。Spring容器负责创建应用程序中的bean并通过DI来协调这些对象之间的关系。

2、当描述bean如何进行装配时，Spring具有非常大的灵活性，它提供了三种主要的装配机制：

在XML中进行显式配置。

在Java中进行显式配置。

隐式的bean发现机制和自动装配。

**2.2　自动化装配bean**

1、Spring从两个角度来实现自动化装配：

组件扫描（component scanning）：Spring会自动发现应用上下文中所创建的bean。

自动装配（autowiring）：Spring自动满足bean之间的依赖。

2、@Component注解。这个简单的注解表明该类会作为组件类，并告知Spring要为这个类创建bean。

3、简单来说，自动装配就是让Spring自动满足bean依赖的一种方法，在满足依赖的过程中，会在Spring应用上下文中寻找匹配某个bean需求的其他bean。为了声明要进行自动装配，我们可以借助Spring的@Autowired注解。