# Spring之旅

## 1.1 简化Java开发

为了降低java开发的复杂性，Spring采取了以下4中策略：

* 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

### 激发POJO的潜能

### 1.1.2 依赖注入

### 1.1.3 应用切面

面向切面编程(AOP)允许你把遍布各处的功能分离出来形成可重用的组件。

系统由许多不同的组件组成，每一个组件各负责一个特定的功能。除了实现自身核心的功能之外，这些组件还经常担任着额外的职责。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，因为他们会跨越系统的多个组件。

### 1.1.4 使用模板消除样板式代码

## 1.2 容纳你的Bean

容器是Spring框架的核心。Spring容器使用DI管理构成应用的组件，它会创建相互协作的组件之间的关联。

Spring容器并不是只有一个。Spring自带了多个容器实现，可以归为两种不同的类型。bean工程是最简单的容器，提供基本的DI支持。应用上下文基于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。

### 1.2.1 使用应用上下文

Spring自带了多种类型的应用上下文，下面几个是比较常用的：

* AnnotationConfigApplicationContext:从一个或多个基于java的配置类中加载Spring应用上下文。
* AnnotationConfigWebApplicationContext:从一个或多个基于java的配置类中加载Spring Web应用上下文。
* ClassPathXmlApplicationContext：从类路径中的一个或多个XML配置文件中加载上下文定义，把应用上下文的定义文件作为类资源。
* FileSystemXMLApplicationContext：从文件系统下的一个或多个XML配置文件中加载上下文定义。
* XmlWebApplicationContext：从Web应用下的一个或多个XML配置文件中加载上下文定义。

ApplicationContext context=new FileSystemXmlApplicationContext("c:/a.xml");

使用FileSystemXMLApplicationContext与使用ClassPathXMLApplicationContext的区别在于，前者在指定的文件系统路径下查找a.xml文件；而ClassPathXMLApplicationContext是在所有的类路径下查找a.xml文件。

如果你想在java配置中加载应用上下文，那么可以使用AnnotationConfigApplicationContext

ApplicationContext context = new AnnotationConfigApplicationContext(com.lizan.spring.exercise.Test.class);

应用上下文准备就绪之后，就可以调用上下文的getBean()方法从Spring容器中获取bean()。

### 1.2.2 bean的生命周期