# Spring

## spring配置文件

### spring的namespace

|  |
| --- |
| <?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:context="http://www.springframework.org/schema/context" xmlns:jdbc="http://www.springframework.org/schema/jdbc"   xmlns:jee="http://www.springframework.org/schema/jee" xmlns:tx="http://www.springframework.org/schema/tx"  xmlns:util="http://www.springframework.org/schema/util" xmlns:task="http://www.springframework.org/schema/task" xsi:schemaLocation="  http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-4.1.xsd  http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-4.1.xsd  http://www.springframework.org/schema/jdbc http://www.springframework.org/schema/jdbc/spring-jdbc-4.1.xsd  http://www.springframework.org/schema/jee http://www.springframework.org/schema/jee/spring-jee-4.1.xsd  http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-4.1.xsd  http://www.springframework.org/schema/util http://www.springframework.org/schema/util/spring-util-4.1.xsd  http://www.springframework.org/schema/task http://www.springframework.org/schema/task/spring-task-4.1.xsd"  default-lazy-init="true">  </beans> |

### 组件扫描

|  |
| --- |
| <context:component-scan base-package="com.scm">  <context:exclude-filter type="regex" expression="com.scm.bean"></context:exclude-filter> </context:component-scan> |

## Spring IOC功能

### 创建一个maven项目

### 修改对应的pom文件

|  |
| --- |
| <dependencies>  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-context</artifactId>  <version>4.3.9.RELEASE</version>  </dependency> </dependencies> |

### 添加配置文件

a)创建一个spring的xml配置文件

b)然后添加namespace

c)添加组件扫描配置(见一、2)

### 编写一个bean类

|  |
| --- |
| package com.scm.domain;  import org.springframework.stereotype.Repository;  */\*\*  \* Created by admin on 2017-07-21.  \*/* @Repository public class TestBean {  private String username;  private String password;   public String getUsername() {  return username;  }   public void setUsername(String username) {  this.username = username;  }   public String getPassword() {  return password;  }   public void setPassword(String password) {  this.password = password;  }   @Override  public String toString() {  return "TestBean{" +  "username='" + username + '\'' +  ", password='" + password + '\'' +  '}';  } } |

### 编写一个测试类

|  |
| --- |
| package com.scm.test;  import org.junit.Test; import org.springframework.context.support.ClassPathXmlApplicationContext; import com.scm.domain.TestBean;  */\*\*  \* Created by admin on 2017-07-21.  \*/* public class TestDemo1 {  @Test  public void test() {  ClassPathXmlApplicationContext context = new ClassPathXmlApplicationContext("springContext.xml");  TestBean bean = context.getBean(TestBean.class);  bean.setPassword("123");  bean.setUsername("zhagnbenxin");  System.*out*.println(bean);  } } |

简单来说，我对IOC的理解就是，首先对要交给spring管理的bean类进行标记，然后在配置文件中进行注解扫描。

## Spring中获取bean对象

### IOC，使用@Autowired注入

### spring中手动创建对象 SpringContextHolder.getBean()

**Private static** CacheManager *cacheManager* = SpringContextHolder.*getBean*(CacheManager.**class**);

|  |
| --- |
|  |

### ClassPathXmlApplication,加载spring配置文件，getBean()方法获取对应的类，见[二.5](#_编写一个测试类)。

## spring依赖注入

其实就是IOC(控制反转)的另一种说法，**[spring](http://lib.csdn.net/base/javaee" \o "Java EE知识库" \t "http://blog.csdn.net/lishuangzhe7047/article/details/_blank)**框架为我们提供了三种注入方式，分别是set注入，构造方法注入，接口注入。

### Set注入









<prop>标签中的key值是.properties属性文件中的名称

注意：

  无论给什么赋值，配置文件中<property>标签的name属性值一定是和对象中名称一致。

### 构造方法注入





### 接口注入

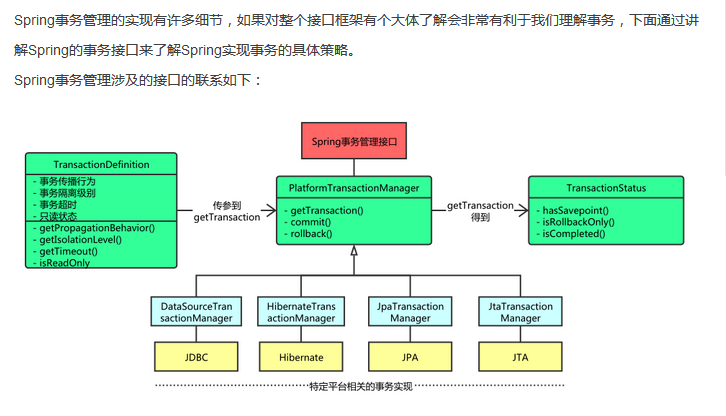
参考：<http://ghl116.iteye.com/blog/1474250>

## Spring AOP

参考：<http://blog.csdn.net/lishuangzhe7047/article/details/21300317>

## Spring事务管理

参考：<http://www.mamicode.com/info-detail-1248286.html>



## Spring实战