1. Spring+Mybaties ：Spring配置文件中加载jdbc.properties中的属性时，加载进来的按字符串进行了处理，并不能在

<bean id="dataSource" class="org.apache.commons.dbcp.BasicDataSource"

destroy-method="close"

p:driverClassName="${jdbc.driverClassName}"

p:url="${jdbc.url}"

p:username="${jdbc.username}"

p:password="${jdbc.password}"/>

中直接使用。

**原因：在装配Mybaties时，我们在spring的配置文件中加入了**

**<bean class="org.mybatis.spring.mapper.MapperScannerConfigurer"**

**p:sqlSessionFactory-ref="sqlSessionFactory"**

**p:basePackage="org.ncu.dao"/>将dao.xml映射接口dao直接转化为spring容器中的bean，这样就可以在Service注入映射接口的bean， 可能会导致PropertyPlaceholderConfigurer失效，也就是用${jdbc.username}这样之类的表达式，将无法获取到properties文件里的内容。 导致这一原因是因为，MapperScannerConigurer实际是在解析加载bean定义阶段的，这个时候要是设置sqlSessionFactory的话，会导致提前初始化一些类，这个时候，PropertyPlaceholderConfigurer还没来得及替换定义中的变量，导致把表达式当作字符串复制了。**

**修改方法：将**

**p:sqlSessionFactory-ref="sqlSessionFactory"**

**改成**

**p:sqlSessionFactoryBeanName="sqlSessionFactory"**

1. Spring+MyBities：将properties中的属性注入后，报错：Access denied for user '##.##'@'localhost' (using password: YES)。

**原因：properties中属性配置格式引起注入歧义。password这些名字可能与注入spring容器中的其他bean的name相同导致注入歧义。**

**原格式：**

**driverClassName=com.mysql.jdbc.Driver**

**url=jdbc:mysql://localhost:3306/signdb**

**username=root**

**password=Lxb090288**

**修改方法：在properties中的属性名字前加入jdbc.XXXX来和其他的那么区分开来**

**现在格式：**

**jdbc.driverClassName=com.mysql.jdbc.Driver**

**jdbc.url=jdbc:mysql://localhost:3306/signdb**

**jdbc.username=root**

**jdbc.password=Lxb090288**

1. spring+springMVC+Mybities：用idea2017工程部署完成后，在spring+Mybities下用TESTNG做单元测试时成功，添加springMVC后，tomcat运行报错：在web中配置的监听启动失败，找不到Class文件。

**原因：idea2017在maven普通项目中添加web文件后没有自动生成lib文件，也就是没有web工程所需要的war包，导致项目启动失败。**

**解决办法：在idea2017中打开File->ProjectStructer->Artifacts,在该对话框下进行手动jar包导入web项目**

1. ssm框架：spring MVC的配置org.springframework.web.servlet.DispatcherServlet启动失败

**原因：web.xml通过contextConfigLocation的属性来为web域启动spring容器，将spring的属性注入范围扩大到web，而springMVC的配置文件的加载则通过名字匹配来完成，文件名字格式为XXX-servlet.xml，XXX与在web.xml中启动spring的servlet拦截器的名字相同（servlet-name）**

**<servlet>**

**<servlet-name>webOffice</servlet-name>**

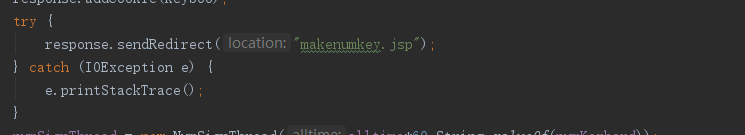
**解决方法：将springMVC配置文件名由原来的applicationContext改为webOffice-servlet.xml**

1. ssm框架：Mybaties对数据库操作有多个参数时：

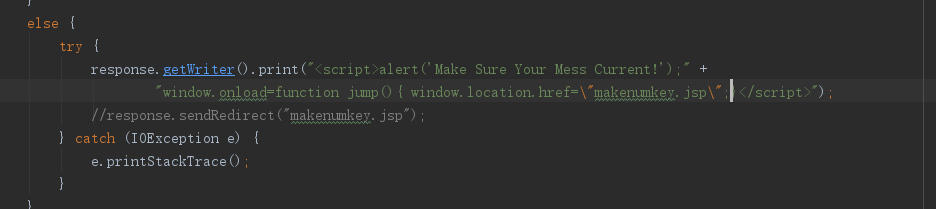
**eg：将<select id="XXX" resultType="int" parameterType="int">中的parameter属性去掉，在where条件中通过#{index}的方式来匹配入参，index从0开始**

1. SpringMVC 中返回值为void类型函数使用：

**void方法不定义HttpServletResponse类型的入参，Spring MVC会认为@RequestMapping注解中指定的路径就是要返回的视图name**



**通过response.sendRedirect()方法做页面跳转可以解决刷新页面重复提交表单的问题。**



**通过response写js是写到requestMapping的value页面中，如果自己再写页面跳转的话js不会生效。**

1. ssm框架：idea maven项目搭建ssm框架

Tip1.

<bean id="sqlSessionFactory class="org.mybatis.spring.SqlSessionFactoryBean"

p:dataSource-ref="dataSource"

p:configLocation="classpath:myBatisConfig.xml"

p:mapperLocations="classpath:org/ncu/dao/\*.xml"/>

将对数据源的映射放在myBatisConfig.xml中，将对数据的操作放在\*.xml中，org/ncu/dao/\*.xml要放在redources目录中，与java（sources）目录中的\*DAO.java文件通过名字一一对应（通过spring配置中的

<bean class="org.mybatis.spring.SqlSessionTemplate">

<constructor-arg ref="sqlSessionFactory"/>

</bean>完成）

Tip2.框架jarbao导入：

<dependencies>  
 <dependency>  
 <groupId>org.mybatis</groupId>  
 <artifactId>mybatis</artifactId>  
 <version>3.4.0</version>  
 </dependency>  
 <dependency>  
 <groupId>org.mybatis</groupId>  
 <artifactId>mybatis-spring</artifactId>  
 <version>1.3.0</version>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-beans</artifactId>  
 <version>${spring.version}</version>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-context</artifactId>  
 <version>${spring.version}</version>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-context-support</artifactId>  
 <version>${spring.version}</version>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-jdbc</artifactId>  
 <version>${spring.version}</version>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-webmvc</artifactId>  
 <version>${spring.version}</version>  
 </dependency>  
  
 <dependency>  
 <groupId>commons-dbcp</groupId>  
 <artifactId>commons-dbcp</artifactId>  
 <version>${commons-dbcp.version}</version>  
 </dependency>  
 <dependency>  
 <groupId>mysql</groupId>  
 <artifactId>mysql-connector-java</artifactId>  
 <version>${mysql.version}</version>  
 </dependency>  
 <dependency>  
 <groupId>javax.servlet</groupId>  
 <artifactId>servlet-api</artifactId>  
 <version>${servlet.version}</version>  
 <scope>provided</scope>  
 </dependency>  
 <dependency>  
 <groupId>javax.servlet</groupId>  
 <artifactId>jstl</artifactId>  
 <version>1.2</version>  
 </dependency>

<dependency>  
 <groupId>org.testng</groupId>  
 <artifactId>testng</artifactId>  
 <version>${testng.version}</version>  
 <scope>test</scope>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-test</artifactId>  
 <version>${spring.version}</version>  
 <scope>test</scope>  
 </dependency>  
</dependencies>

<properties>

<spring.version>4.2.2.RELEASE</spring.version>

<slf4j.version>1.7.5</slf4j.version>

<mysql.version>5.1.29</mysql.version>

<aspectj.version>1.8.1</aspectj.version>

<servlet.version>3.0-alpha-1</servlet.version>

<aopalliance.version>1.0</aopalliance.version>

<commons-dbcp.version>1.4</commons-dbcp.version>

<jetty.version>8.1.8.v20121106</jetty.version>

<testng.version>6.8.7</testng.version>

</properties>