**ssm整合**

**1、本章面试题**

           说说ssm分别是什么框架，每个框架在项目分层的作用

普通web 自己new对象调用 servlet(m v c) jdbc(封装) jsp+servlet+BaseDao

SSM spring (使用IOC贯穿各层) srpingmvc（表现层） mybatis（持久层）

SSH spring structs2 hibernate

**2、知识点**

**2.1、课程回顾**

       spring mvc 运行原理

spring mvc 常用注解的用法 @Controller @RequestMapping

**2.2、本章重点**

         ssm框架的整合

**3、具体内容**

**3.1,新建maven项目，引入jar**

<!--spring版本号-->

<spring.version>4.2.4.RELEASE</spring.version>

<!-- mybatis版本号 -->

<mybatis.version>3.4.1</mybatis.version>

<!-- oracle驱动 -->

<ojdbc6.version>11.2.0.3.0</ojdbc6.version>

<!-- log4j日志文件管理包版本 -->

<log4j.version>1.2.17</log4j.version>

<!-- jackson版本号 -->

<jackson.version>2.8.3</jackson.version>

<!--servlet 版本号-->

<servlet.version>4.0.1</servlet.version>

<!--oracle驱动包-->

<dependency>

<groupId>oracle</groupId>

<artifactId>oracle-jdbc</artifactId>

<version>${ojdbc6.version}</version>

</dependency>

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>javax.servlet-api</artifactId>

<version>${servlet.version}</version>

</dependency>

<!-- spring核心包 -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-core</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-web</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-oxm</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-tx</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>org.springframework</groupId>

<artifactId>spring-aop</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-test</artifactId>

<version>${spring.version}</version>

</dependency>

<!-- mybatis核心包 -->

<dependency>

<groupId>org.mybatis</groupId>

<artifactId>mybatis</artifactId>

<version>${mybatis.version}</version>

</dependency>

<!-- mybatis/spring包 -->

<dependency>

<groupId>org.mybatis</groupId>

<artifactId>mybatis-spring</artifactId>

<version>1.3.0</version>

</dependency>

<!-- 导入java ee jar 包 -->

<dependency>

<groupId>javax</groupId>

<artifactId>javaee-api</artifactId>

<version>7.0</version>

</dependency>

<!-- 导入dbcp的jar包，用来在applicationContext.xml中配置数据库 -->

<dependency>

<groupId>commons-dbcp</groupId>

<artifactId>commons-dbcp</artifactId>

<version>1.4</version>

</dependency>

<!-- JSTL标签类 -->

<dependency>

<groupId>jstl</groupId>

<artifactId>jstl</artifactId>

<version>1.2</version>

</dependency>

<!-- 日志文件管理包 -->

<dependency>

<groupId>log4j</groupId>

<artifactId>log4j</artifactId>

<version>${log4j.version}</version>

</dependency>

<dependency>

<groupId>com.alibaba</groupId>

<artifactId>fastjson</artifactId>

<version>1.2.17</version>

</dependency>

<!-- 上传组件包 -->

<dependency>

<groupId>commons-fileupload</groupId>

<artifactId>commons-fileupload</artifactId>

<version>1.3.1</version>

</dependency>

<dependency>

<groupId>commons-io</groupId>

<artifactId>commons-io</artifactId>

<version>2.5</version>

</dependency>

<dependency>

<groupId>commons-codec</groupId>

<artifactId>commons-codec</artifactId>

<version>1.10</version>

</dependency>

<dependency>

<groupId>org.aspectj</groupId>

<artifactId>aspectjweaver</artifactId>

<version>1.6.12</version>

</dependency>

<!-- redis 开始 -->

<dependency>

<groupId>redis.clients</groupId>

<artifactId>jedis</artifactId>

<version>2.8.0</version>

</dependency>

<dependency>

<groupId>org.springframework.data</groupId>

<artifactId>spring-data-redis</artifactId>

<version>1.6.2.RELEASE</version>

</dependency>

<dependency>

<groupId>org.mybatis</groupId>

<artifactId>mybatis-ehcache</artifactId>

<version>1.0.0</version>

</dependency>

<!-- redis 结束-->

<!-- https://mvnrepository.com/artifact/mysql/mysql-connector-java -->

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>5.1.10</version>

</dependency>

**3.2，web.xm spring配置监听器和springmvc总调度器**

**修改头部版本：**

<web-app

version="3.0"

xmlns="http://java.sun.com/xml/ns/javaee"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd">

**配置spring监听器,作用启动Web容器时，自动装配ApplicationContext的配置信息。**

<!--spring监听器 开始 -->

<listener>

<listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>

</listener>

<!--监听文件-->

<context-param>

<param-name>contextConfigLocation</param-name>

<param-value>classpath:spring-mybatis.xml</param-value>

</context-param>

<!--spring监听器 结束 -->

**配置springmvc总调度器**

<!--配置总调度器 开始-->

<servlet>

<servlet-name>springmvc</servlet-name>

<servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>

<!--加载自定义配置文件-->

<init-param>

<param-name>contextConfigLocation</param-name>

<param-value>classpath:springmvc-config.xml</param-value>

</init-param>

</servlet>

<servlet-mapping>

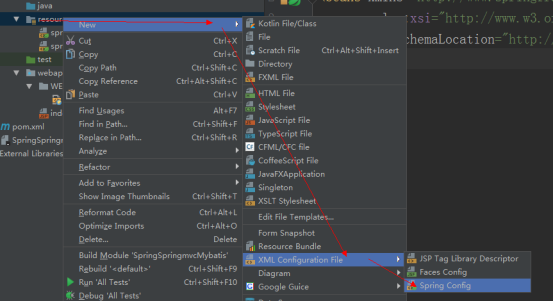
<servlet-name>springmvc</servlet-name>

<url-pattern>\*.do</url-pattern>

</servlet-mapping>

<!--配置总调度器 结束-->

**3.3，spring整合mybatis的spring-mybatis-config.xml配置文件**



**dbcp.properties：**

jdbc.driver=oracle.jdbc.driver.OracleDriver

jdbc.url=jdbc:oracle:thin:@localhost:1521:orcl

jdbc.username=scott

jdbc.password=tiger

#初始连接数

initialSize=10

#最大活跃数

maxActive=5

#最大空闲连接

maxIdle=1

#最小idle数

minIdle=1

#最长等待时间(毫秒)

maxWait=1000

**spring-mybatis-config.xml**

<!-- 开启包扫描 -->

<context:component-scan base-package="com.aaa.ssm.service"></context:component-scan>

<!-- 加载数据源配置文件 -->

<bean class="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer">

<property name="location" value="classpath:dbcp.properties"></property>

</bean>

<!-- dbcp连接池配置 -->

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

<property name="driverClassName" value="${jdbc.driver}"></property>

<property name="url" value="${jdbc.url}"></property>

<property name="username" value="${jdbc.username}"></property>

<property name="password" value="${jdbc.password}"></property>

<property name="initialSize" value="${initialSize}"></property>

<property name="maxActive" value="${maxActive}"></property>

<property name="maxIdle" value="${maxIdle}"></property>

<property name="minIdle" value="${minIdle}"></property>

<property name="maxWait" value="${maxWait}"></property>

</bean>

<!-- 创建SqlSessionFactory -->

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

<!-- 加载数据源 -->

<property name="dataSource" ref="dataSourceA"></property>

<!-- 加载mapper文件 -->

<property name="mapperLocations" value="classpath:mapper/\*.xml"></property>

</bean>

<!-- 利用配置的mybatis接口和配置文件 把dao实例化 -->

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

<property name="basePackage" value="com.aaa.ssm.dao"></property>

<property name="sqlSessionFactoryBeanName" value="sqlSessionFactory"></property>

</bean>

<!-- 定义事务管理器 -->

<bean id="transactionManager" class="org.springframework.jdbc.datasource.DataSourceTransactionManager">

<property name="dataSource" ref="dataSourceA"/>

</bean>

<!-- 注解驱动，启动注解 -->

<tx:annotation-driven transaction-manager="transactionManager"/>

**3. 4，springmvc-coinfig.xml配置文件**

<context:component-scan base-package="com.aaa.ssm.controller"></context:component-scan>

<!--mvc:annotation-driven 作用：

Spring会默认帮我们注册处理请求，参数和返回值的类

会自动注册RequestMappingHandlerMapping、RequestMappingHandlerAdapter 与ExceptionHandlerExceptionResolver 三个bean。

支持使用 ConversionService 实例对表单参数进行类型转换；

支持使用 @NumberFormat annotation、@DateTimeFormat；

支持使用 @RequestBody 和 @ResponseBody 注解

-->

<mvc:annotation-driven/>

<!--映射处理器，处理适配器不配置不带表没有，使用默认的了-->

<!--配置视图解析器-->

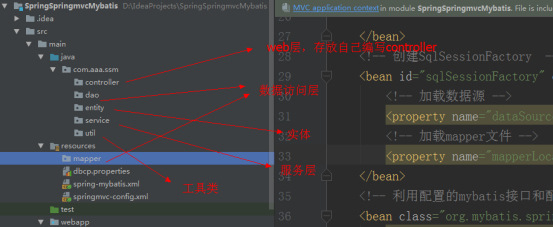
<bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">

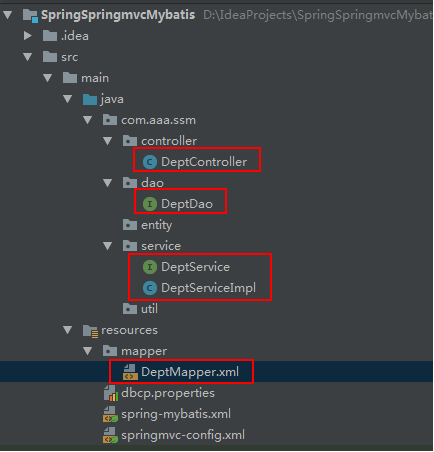
<property name="prefix" value="/WEB-INF/jsp/"></property>

<property name="suffix" value=".jsp"></property>

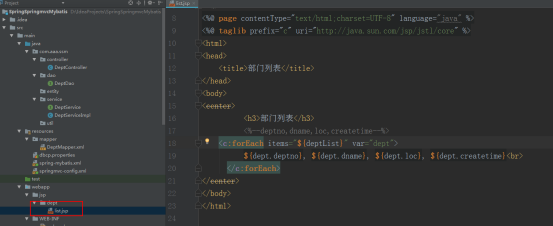
</bean>

**3.5，创建项目分层，编写各层代码**





**3.6，创建页面**

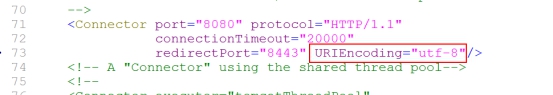


**3.7，运行测试**

**3.8 其他注意：**

**解决中文乱码:**

get： 如果是tomcat7:在 apache-tomcat7\conf\server.xml中修改配置：



post：

<!--post提交中文乱码解决 开始 -->

    <filter>

        <filter-name>charEncode</filter-name>

        <filter-class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>

        <!--初始化编码方式 -->

        <init-param>

            <param-name>encoding</param-name>

            <param-value>utf-8</param-value>

        </init-param>

    </filter>

    <filter-mapping>

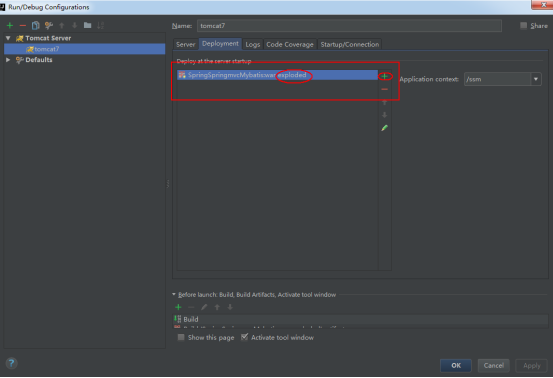
        <filter-name>charEncode</filter-name>

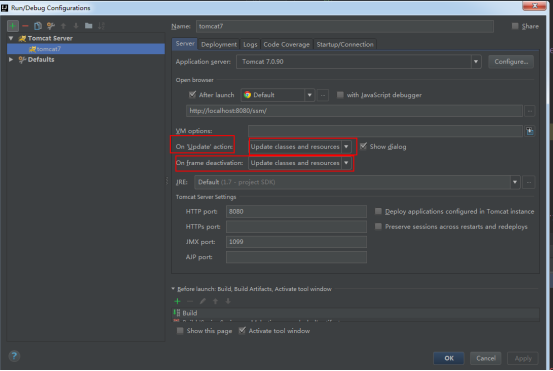
        <url-pattern>/\*</url-pattern>

    </filter-mapping>

    <!--post提交中文乱码解决 结束  -->

**idea下tomcat热部署：**





**使用debug方式启动**

**4、本章总结**

**4.1 总结本章知识点**

**4.2 面试题答案**

**4.3 预习下一章重点**