SSM环境搭建（不使用Maven）

## 一、创建项目

* 1. Idea创建一个web项目
  2. 项目下创建resource文件夹并设置为源文件夹
  3. web文件夹下创建lib文件夹用来存放jar包，将lib文件夹添加为项目依赖

## 二、创建配置文件

在resource下创建

## 2.1 创建SqlMapConfig.xml

*<?***xml version="1.0" encoding="UTF-8"** *?>***<!DOCTYPE configuration PUBLIC "-//mybatis.org//DTD Config 3.0//EN"  
 "http://mybatis.org/dtd/mybatis-3-config.dtd"*>***<**configuration**>  
  
 *<!-- 1.Mybatis的配置 -->  
  
 <!-- 别名 -->* <**typeAliases**>  
 <**package name="com.enbuys.ssm.pojo"**/>  
 </**typeAliases**>  
  
</**configuration**>

## 2.2 创建jdbc.properties（数据库源）

**jdbc.driver**=**com.mysql.jdbc.Driver  
jdbc.url**=**jdbc:mysql://localhost:3306/crm?characterEncoding=utf-8  
jdbc.username**=**root  
jdbc.password**=**root**

## 2.3 创建applicationContext-dao.xml（mybatis的spring配置）

*<?***xml version="1.0" encoding="UTF-8"***?>*<**beans xmlns="http://www.springframework.org/schema/beans"  
 xmlns:context="http://www.springframework.org/schema/context" xmlns:p="http://www.springframework.org/schema/p"  
 xmlns:aop="http://www.springframework.org/schema/aop" xmlns:tx="http://www.springframework.org/schema/tx"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-4.0.xsd  
 http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-4.0.xsd  
 http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop-4.0.xsd http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-4.0.xsd  
 http://www.springframework.org/schema/util http://www.springframework.org/schema/util/spring-util-4.0.xsd"**>  
  
 *<!-- Spring与mybatis的配置 -->  
  
 <!-- 配置 读取properties文件 jdbc.properties -->* <**context:property-placeholder location="classpath:jdbc.properties"** />  
  
 *<!-- 配置 数据源 -->* <**bean id="dataSource" class="com.alibaba.druid.pool.DruidDataSource"**>  
 <**property name="driverClassName" value="${jdbc.driver}"** />  
 <**property name="url" value="${jdbc.url}"** />  
 <**property name="username" value="${jdbc.username}"** />  
 <**property name="password" value="${jdbc.password}"** />  
 </**bean**>  
  
 *<!-- 配置SqlSessionFactory -->* <**bean class="org.mybatis.spring.SqlSessionFactoryBean"**>  
 *<!-- 设置MyBatis核心配置文件 -->* <**property name="configLocation" value="classpath:SqlMapConfig.xml"** />  
 *<!-- 设置数据源 -->* <**property name="dataSource" ref="dataSource"** />  
 </**bean**>  
  
 *<!-- 配置Mapper扫描 -->* <**bean class="org.mybatis.spring.mapper.MapperScannerConfigurer"**>  
 *<!-- 设置Mapper扫描包 -->* <**property name="basePackage" value="com.enbuys.ssm.mapper"** />  
 </**bean**>  
</**beans**>

## 2.4 创建applicationContext-service（service的spring配置）

*<?***xml version="1.0" encoding="UTF-8"***?>*<**beans xmlns="http://www.springframework.org/schema/beans"  
 xmlns:context="http://www.springframework.org/schema/context" xmlns:p="http://www.springframework.org/schema/p"  
 xmlns:aop="http://www.springframework.org/schema/aop" xmlns:tx="http://www.springframework.org/schema/tx"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-4.0.xsd  
 http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-4.0.xsd  
 http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop-4.0.xsd http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-4.0.xsd  
 http://www.springframework.org/schema/util http://www.springframework.org/schema/util/spring-util-4.0.xsd"**>  
  
 *<!-- Service的spring配置 -->  
  
 <!-- 配置Service扫描 -->* <**context:component-scan base-package="com.enbuys.ssm.service"** />  
</**beans**>

## 2.5 创建springmvc.xml（SpringMVC配置）

*<?***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:p="http://www.springframework.org/schema/p"  
 xmlns:context="http://www.springframework.org/schema/context"  
 xmlns:mvc="http://www.springframework.org/schema/mvc"  
 xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-4.0.xsd  
 http://www.springframework.org/schema/mvc http://www.springframework.org/schema/mvc/spring-mvc-4.0.xsd  
 http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-4.0.xsd"**>  
  
 *<!-- SpringMVC的配置 -->  
  
 <!-- 配置Controller扫描 -->* <**context:component-scan base-package="com.enbuys.ssm.controller"** />  
  
 *<!-- 配置注解驱动 -->* <**mvc:annotation-driven** />  
  
</**beans**>

## 2.6 配置web.xml

*<?***xml version="1.0" encoding="UTF-8"***?>*<**web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app\_4\_0.xsd"  
 version="4.0"**>  
  
 *<!-- 配置spring -->* <**context-param**>  
 <**param-name**>contextConfigLocation</**param-name**>  
 <**param-value**>classpath:applicationContext-\*.xml</**param-value**>  
 </**context-param**>  
  
 *<!-- 配置监听器加载spring -->* <**listener**>  
 <**listener-class**>org.springframework.web.context.ContextLoaderListener</**listener-class**>  
 </**listener**>  
  
 *<!-- 配置过滤器，解决post的乱码问题 -->* <**filter**>  
 <**filter-name**>encoding</**filter-name**>  
 <**filter-class**>org.springframework.web.filter.CharacterEncodingFilter</**filter-class**>  
 <**init-param**>  
 <**param-name**>encoding</**param-name**>  
 <**param-value**>UTF-8</**param-value**>  
 </**init-param**>  
 </**filter**>  
 <**filter-mapping**>  
 <**filter-name**>encoding</**filter-name**>  
 <**url-pattern**>/\*</**url-pattern**>  
 </**filter-mapping**>  
  
 *<!-- 配置SpringMVC -->* <**servlet**>  
 <**servlet-name**>boot-crm</**servlet-name**>  
 <**servlet-class**>org.springframework.web.servlet.DispatcherServlet</**servlet-class**>  
 <**init-param**>  
 <**param-name**>contextConfigLocation</**param-name**>  
 <**param-value**>classpath:springmvc.xml</**param-value**>  
 </**init-param**>  
 *<!-- 配置springmvc什么时候启动，参数必须为整数 -->  
 <!-- 如果为0或者大于0，则springMVC随着容器启动而启动 -->  
 <!-- 如果小于0，则在第一次请求进来的时候启动 -->* <**load-on-startup**>1</**load-on-startup**>  
 </**servlet**>  
 <**servlet-mapping**>  
 <**servlet-name**>boot-crm</**servlet-name**>  
 *<!-- 所有的请求都进入springMVC -->* <**url-pattern**>/</**url-pattern**>  
 </**servlet-mapping**>  
  
  
 <**display-name**>SSM\_CRM</**display-name**>  
 <**welcome-file-list**>  
 <**welcome-file**>index.html</**welcome-file**>  
 </**welcome-file-list**>  
  
</**web-app**>

**注意：一些写死的包名如果不同需要修改！**

# 三、所有配置完成，可以进行测试了

创建com.enbuys.ssm文件夹

## 3.1 创建pojo.User类

**package** com.enbuys.ssm.pojo;  
**public class** User {  
  
 **private int id**;  
 **private** String **username**;  
  
 **public int** getId() {  
 **return id**;  
 }  
  
 @Override  
 **public** String toString() {  
 **return "User{"** +  
 **"id="** + **id** +  
 **", username='"** + **username** + **'\''** +  
 **'}'**;  
 }  
  
 **public void** setUsername(String username) {  
 **this**.**username** = username;  
 }  
  
 **public void** setId(**int** id) {  
 **this**.**id** = id;  
 }  
  
 **public** String getUsername() {  
  
 **return username**;  
 }  
}

## 3.2 创建mapper.UserMapper接口

**public interface** UserMapper {  
  
 *//根据客户类别代码查询数据* **public** List<User> selectUserList();  
  
}

## 3.3 创建mapper.UserMapper.xml文件

*<?***xml version="1.0" encoding="UTF-8"** *?>***<!DOCTYPE mapper  
 PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"  
 "http://mybatis.org/dtd/mybatis-3-mapper.dtd"*>***<**mapper namespace="com.enbuys.ssm.mapper.UserMapper"**>  
  
 <**select id="selectUserList" resultType="User"**>  
 SELECT *\** from user  
 </**select**>  
  
</**mapper**>

## 3.4 创建service.UserService类

@Service  
**public class** UserService {  
  
 @Autowired  
 **private** UserMapper **userMapper**;  
  
 **public** List<User> selectUserList(){  
 **return userMapper**.selectUserList();  
 }  
}

## 3.5 创建controller.UserController类

@Controller  
**public class** MainController {  
  
 @Autowired  
 **private** UserService **userService**;  
  
 @RequestMapping(**"/list"**)  
 **public void** list(Model model){  
 List<User> userList = **userService**.selectUserList();  
 **for**(User user : userList)  
 System.***out***.println(user);  
 }  
}

## 3.6 创建数据库

CREATE DATABASE ssm;

use ssm;

CREATE TABLE `user` (

`userid` bigint(32) NOT NULL AUTO\_INCREMENT,

`username` varchar(32) NOT NULL,

PRIMARY KEY (`userid`)

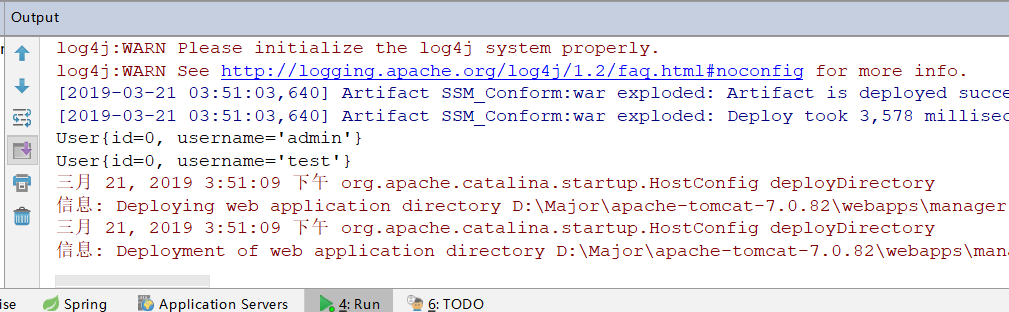
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

INSERT INTO `user` VALUES ('1','admin');

INSERT INTO `user` VALUES ('2','test');

SELECT \* from user;

## 3.7 配置Tomcat，运行输入127.0.0.1:8080/list进行测试



成功获取User信息