# Java spring mvc配置

1. 安装spring插件, 百度 spring tool suite 下载
2. 新建Maven Project, 其中一个next步骤选择filter为maven-archetype-webapp.
3. 修改pom.xml导入mvc相关jar包. 添加版本节点, 值可根据对应版本要求进行修改(注意命名空间)

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4\_0\_0.xsd">

<properties>

<org.springframework.version>4.2.6.RELEASE</org.springframework.version>

</properties>

<dependencies>

<dependency>

<groupId>javax.servlet</groupId>

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

<version>3.1.0</version>

<scope>provided</scope>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>${org.springframework.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-jdbc</artifactId>

<version>${org.springframework.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-aspects</artifactId>

<version>${org.springframework.version}</version>

</dependency>

</dependencies>

1. 修改项目配置文件WEB-INF文件夹下的web.xml, 添加如下节点, 注意spring上下文中的xml配置指向的是自己的配置bean文件

<?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\_3\_1.xsd"

version="3.1">

<context-param>

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

<param-value>classpath\*:/dispatcher-servlet.xml, classpath\*:/beans/\*.xml</param-value>

</context-param>

<listener>

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

</listener>

<servlet>

<servlet-name>dispatcher</servlet-name>

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

<load-on-startup>1</load-on-startup>

</servlet>

<servlet-mapping>

<servlet-name>dispatcher</servlet-name>

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

</servlet-mapping>

</web-app>

Servlet名称决定了配置文件的名称, 如上面配置的名称为dispatcher, 则在WEB-INF文件夹中添加的配置文件为dispatcher-servlet.xml, 同时注意修改要扫描的base-package, 文件内容如下

<?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"

xsi:schemaLocation="http://www.springframework.org/schema/beans

http://www.springframework.org/schema/beans/spring-beans-2.5.xsd

http://www.springframework.org/schema/context

http://www.springframework.org/schema/context/spring-context-2.5.xsd">

<!-- ①：对web包中的所有类进行扫描，以完成Bean创建和自动依赖注入的功能 -->

<context:component-scan base-package="your controller package"/>

<!-- ②：对模型视图名称的解析，即在模型视图名称添加前后缀 -->

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

p:prefix="/WEB-INF/jsp/" p:suffix=".jsp"/>

</beans>

至此配置已完成

1. 修改项目的jre运行时版本, Java Resources 🡪 Libraries; 添加tomcat运行时, 以便可以在eclipse中运行, 右键项目 🡪 build path 🡪 Libraries 🡪 Add Library 🡪 选择Server Runtime 🡪 选择汤姆猫
2. 将所有source输出目录改为WEB-INF下的classes文件夹.
3. 添加公共配置(这个不是spring mvc必须, 为后面数据访问做准备), 在WEB-INF文件夹下添加common-beans.xml文件, 内容如下:

<?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"

xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd

http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-4.2.xsd">

<!-- properties配置连接字符串 -->

<context:property-placeholder location="classpath:application.properties" />

<context:component-scan base-package="daos,entities,services"></context:component-scan>

<!-- 数据库映射 -->

<!-- class="org.apache.tomcat.dbcp.dbcp.BasicDataSource" -->

<!-- class="org.springframework.jdbc.datasource.DriverManagerDataSource" -->

<bean id="dataSource"

class="org.springframework.jdbc.datasource.DriverManagerDataSource">

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

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

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

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

</bean>

</beans>

在resource下添加application.properties文件, 内容如下:

# jdbc.oracle

#jdbc.driverClassName=oracle.jdbc.driver.OracleDriver

#jdbc.url=jdbc:oracle:thin:@192.168.77.176:1521:utm40

#jdbc.user=ibm\_v7

#jdbc.password=ibm\_v7

#jdbc.testSql=select 1 from dual

# jdbc.mysql

#jdbc.driverClassName=com.mysql.jdbc.Driver

#jdbc.url=jdbc:mysql://localhost/toolsitemvc4j

#jdbc.user=root

#jdbc.password=root

#jdbc.testSql=select 1

# jdbc.sql server

jdbc.driverClassName=com.microsoft.sqlserver.jdbc.SQLServerDriver

jdbc.url=jdbc:sqlserver://PC201404190064\\SQLEXPRESS; DatabaseName=ToolSiteMvc4J

#jdbc.url=jdbc:sqlserver://ZJY; DatabaseName=ToolSiteMvc4J

jdbc.user=sa

jdbc.password=1024

jdbc.testSql=select 1

# hibernate.X

hibernate.dialect=org.hibernate.dialect.SQLServerDialect

hibernate.show\_sql=true

hibernate.hbm2ddl.auto=create-drop

# 其它配置

1. 导出war包

* 右键项目选择export 🡪 War file, 选择运行时, 勾上Export source files

1. 添加全局乱码修复filter

<filter>

<filter-name>EncodingFilter</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>

<init-param>

<param-name>forceEncoding</param-name>

<param-value>true</param-value>

</init-param>

</filter>

<filter-mapping>

<filter-name>EncodingFilter</filter-name>

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

</filter-mapping>

1. Filter 过滤掉js问题, 在web.xml里添加如下的配置

<servlet-mapping>

<servlet-name>default</servlet-name>

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

</servlet-mapping>

<servlet-mapping>

<servlet-name>default</servlet-name>

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

</servlet-mapping>

<servlet-mapping>

<servlet-name>default</servlet-name>

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

</servlet-mapping>

<servlet-mapping>

<servlet-name>default</servlet-name>

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

</servlet-mapping>

1. 修复返回json乱码的问题

<mvc:annotation-driven>

<mvc:message-converters register-defaults="true">

<bean class="org.springframework.http.converter.StringHttpMessageConverter">

<property name="supportedMediaTypes" value="application/json;charset=utf-8" />

</bean>

</mvc:message-converters>

</mvc:annotation-driven>