第十七章

SSM框架整合

www.51zxw.net 版权所有 盗版必究

我要自學網 WWW.51ZXW.NET www.51zxw.net 版权所有 盗版必究

17.1 整合环境搭建

我要自學網 WWW.51ZXW.NET www.51zxw.net 版权所有 盗版必究

17.1.1 整合思路

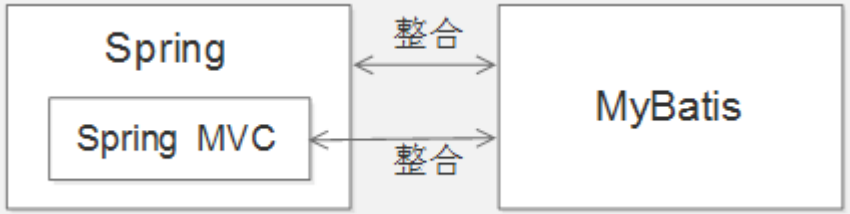
如何进行SSM框架整合？

由于Spring MVC是Spring框架中的一个模块，所以Spring MVC与Spring之间不存在整合的问题，只要引入相应JAR包就可以直接使用。

因此SSM框架的整合就只涉及到了Spring与MyBatis的整合，以及Spring MVC与MyBatis的整合。

我要自學網 WWW.51ZXW.NET www.51zxw.net 版权所有 盗版必究

SSM框架整合图如下所示：



如何确定SSM框架整合成功？

在第10章讲解Spring与MyBatis框架的整合时，我们是通过Spring实例化Bean，然后调用实例对象中的查询方法来执行MyBatis映射文件中的SQL语句的，如果能够正确查询出数据库中的数据，那么我们就认为Spring与MyBatis框架整合成功。同样，整合之后，如果我们可以通过前台页面来执行查询方法，并且查询出的数据能够在页面中正确显示，那么我们也可以认为三大框架整合成功。

我要自學網 WWW.51ZXW.NET www.51zxw.net 版权所有 盗版必究

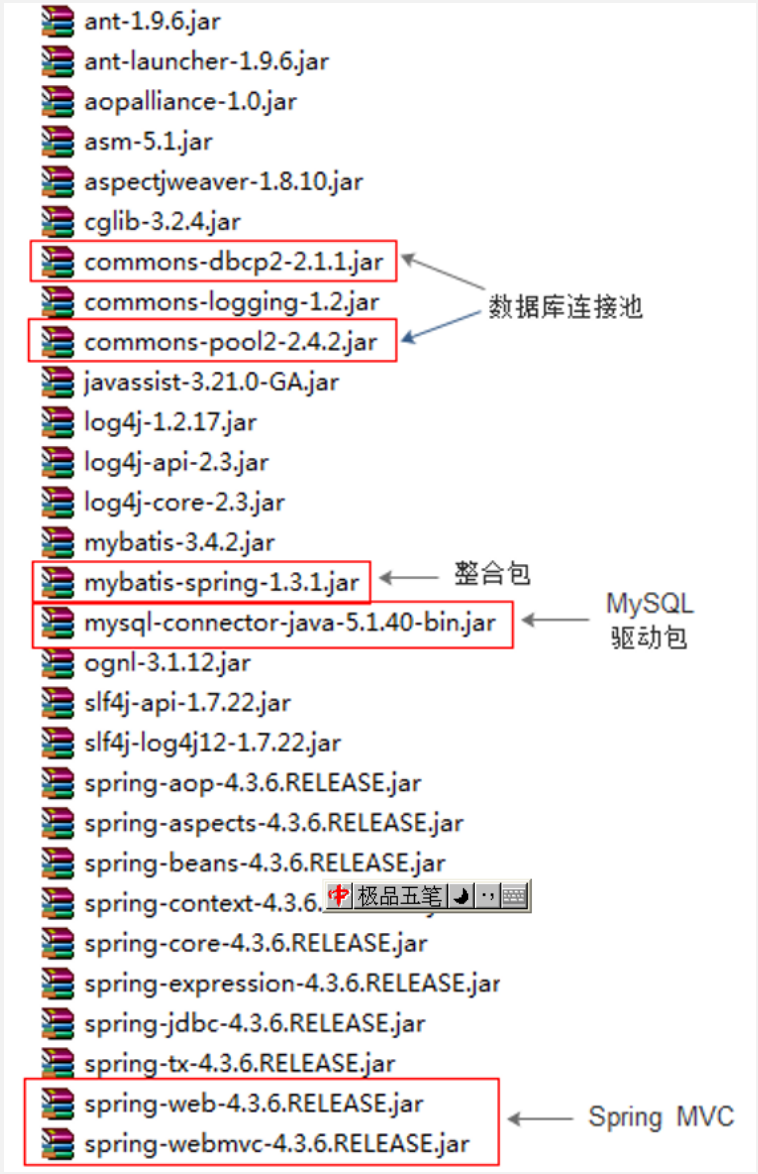
17.1.2 准备所需JAR包

要实现SSM框架的整合，首先要准备这三个框架的JAR包，以及其他整合所需的JAR包。在第10章讲解Spirng与MyBatis框架整合时，已经介绍了Spring与MyBatis整合所需要的JAR包，这里只需要再加入Spring MVC的相关JAR包即可，具体如下：

- spring-web-4.3.6.RELEASE.jar
- spring-webmvc-4.3.6.RELEASE.jar

我要自學網 WWW.51ZXW.NET www.51zxw.net 版权所有 盗版必究

根据上述整合需求，SSM框架整合所需要的基本JAR包如下图所示：



我要自學網 WWW.51ZXW.NET www.51zxw.net 版权所有 盗版必究

17.1.3 编写配置文件

1 在Idea中，创建一个名为ssmchapter17的Maven项目，将整合所需的JAR包添加到pom文件中。

2 在Java目录下，创建一个名为resources的源文件夹，在该文件夹中分别创建数据库常量配置文件db.properties、Spring配置文件applicationContext.xml，以及MyBatis的配置文件mybatis-config.xml。

db.properties

```
jdbc.driver=com.mysql.jdbc.Driver
jdbc.url=jdbc:mysql://localhost:3306/mybatis
jdbc.username=root
jdbc.password=root
jdbc.maxTotal=30
jdbc.maxIdle=10
jdbc.initialSize=5
```

我要自學網 WWW.51ZXW.NET www.51zxw.net 版权所有 盗版必究

applicationContext.xml

```
<context:property-placeholder location="classpath:db.properties"/>
<bean id="dataSource" class="org.apache.commons.dbcp2.BasicDataSource">
        <property name="driverClassName" value="${jdbc.driver}" />
        ...
</bean>
<bean id="transactionManager" class="org.springframework.jdbc.datasource.DataSourceTransactionManager">
        <property name="dataSource" ref="dataSource" /></bean>
 <tx:annotation-driven transaction-manager="transactionManager"/>
 <bean id="sqlSessionFactory" class="org.mybatis.spring.SqlSessionFactoryBean">
        <property name="dataSource" ref="dataSource" />
        <property name="configLocation" value="classpath:mybatis-config.xml" /></bean>
 <bean class="org.mybatis.spring.mapper.MapperScannerConfigurer">
        <property name="basePackage" value="com.ssm.dao"/></bean>
 <context:component-scan base-package="com.ssm.service" />
```

数据源配置

配置事务管理器

开启事务注解

配置MyBatis工厂

配置mapper文件扫描器

Service层注解扫描

我要自學網 WWW.51ZXW.NET www.51zxw.net 版权所有 盗版必究

mybatis-config.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>
        <typeAliases>
                <package name="com.ssm.po" />
        </typeAliases>
</configuration>
```

由于在Spring中已经配置了数据源信息以及mapper接口文件扫描器，所以在MyBatis的配置文件中只需要根据POJO类路径进行别名配置即可。

我要自學網 www.51zxw.net 版权所有 盗版必究

3 在resources文件夹中，创建Spring MVC的配置文件springmvc-config.xml：

```
<context:component-scan base-package="com.ssm.controller" />
<mvc:annotation-driven />
<bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">
    <property name="prefix" value="/WEB-INF/jsp/" />
    <property name="suffix" value=".jsp" />
</bean>
```

Controller层注解扫描

加载注解驱动

配置视图解析器

我要自學網 www.51zxw.net 版权所有 盗版必究

4 在web.xml中，配置Spring的文件监听器、编码过滤器以及Spring MVC的前端控制器等信息。

```
<context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>classpath:applicationContext.xml</param-value>
</context-param>
<listener>
    <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
</listener>
<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>*.action</url-pattern>
</filter-mapping>
<servlet>
     <servlet-name>springmvc</servlet-name>
     <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
     <init-param><param-name>contextConfigLocation</param-name>
     <param-value>classpath:springmvc-config.xml</param-value></init-param>
     <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
     <servlet-name>springmvc</servlet-name>
     <url-pattern>/</url-pattern>
</servlet-mapping>
```

配置Spirng文件监听器

配置编码过滤器

配置Spirng MVC前端控制器

我要自學網 WWW.51ZXW.NET www.51zxw.net 版权所有 盗版必究

17.2 整合应用测试

上一小节已经完成了SSM框架整合环境的搭建工作，可以说完成了这些配置后，就已经完成了这三个框架大部分的整合工作。接下来，同样以查询客户信息为例，来讲解下SSM框架的整合开发。

我要自學網 WWW.51ZXW.NET www.51zxw.net 版权所有 盗版必究

1 在Java目录下，创建com.ssm.po包，并在包中创建持久化类Customer：

```
public class Customer {
        private Integer id;          // 主键id
        private String username; // 客户名称
        private String jobs;         // 职业
        private String phone;      // 电话
       //..省略getter/setter方法
}
```

我要自學網 WWW.51ZXW.NET www.51zxw.net 版权所有 盗版必究

2 在Java目录下，创建一个com.ssm.dao包，并在包中创建接口文件CustomerDao以及对应的映射文件CustomerDao.xml，编辑后分别如下所示：

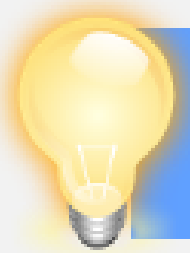
CustomerDao.java

```
package com.ssm.dao;
import com.ssm.po.Customer;
public interface CustomerDao {
        public Customer findCustomerById(Integer id);
}
```

我要自學網 WWW.51ZXW.NET www.51zxw.net 版权所有 盗版必究

CustomerDao.xml

```
<?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.ssm.dao.CustomerDao">
      <select id="findCustomerById" parameterType="Integer" resultType="Customer">
                  select * from t_customer where id = #{id}
      </select>
</mapper>
```



小提示：在前面环境搭建时，已经在配置文件**applicationContext.xml**中使用包扫描的形式加入了扫描包**com.ssm.dao**，所以在这里完成**DAO**层接口及映射文件开发后，就不必再进行映射文件的扫描配置了。

我要自學網 WWW.51ZXW.NET www.51zxw.net 版权所有 盗版必究

3 在Java目录下，创建com.ssm.service包，然后在包中创建接口文件CustomerService，并在CustomerService中定义通过id查询客户的方法：

```
package com.ssm.service;
import com.ssm.po.Customer;
public interface CustomerService {
        public Customer findCustomerById(Integer id);
}
```

我要自學網 WWW.51ZXW.NET www.51zxw.net 版权所有 盗版必究

4 在Java目录下，创建一个com.ssm.service.impl包，并在包中创建CustomerService接口的实现类CustomerServiceImpl：

```
@Service
@Transactional
public class CustomerServiceImpl implements CustomerService {
        //注解注入CustomerDao
        @Autowired
        private CustomerDao customerDao;
        //查询客户
        public Customer findCustomerById(Integer id) {
                return this.customerDao.findCustomerById(id);
        }
}
```

我要自學網 WWW.51ZXW.NET www.51zxw.net 版权所有 盗版必究

5 在Java目录下，创建一个com.ssm.controller包，并在包中创建用于处理页面请求的控制类CustomerController：

```
@Controller
public class CustomerController {
        @Autowired
        private CustomerService customerService;
        @RequestMapping("/findCustomerById")
        public String findCustomerById(Integer id,Model model) {
                Customer customer = customerService.findCustomerById(id);
                model.addAttribute("customer", customer);
                return "customer";
        }
}
```

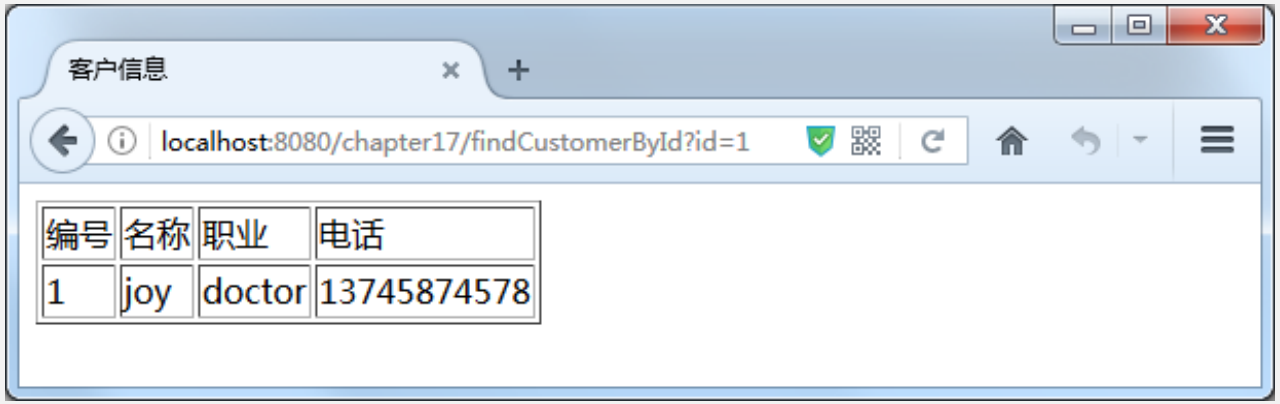
我要自學網 WWW.51ZXW.NET www.51zxw.net 版权所有 盗版必究

6 在WEB-INF目录下，创建一个jsp文件夹，在该文件夹下创建一个用于展示客户详情的页面文件customer.jsp：

```
<table border=1>
    <tr>
        <td>编号</td>
        <td>名称</td>
        <td>职业</td>
        <td>电话</td>
    </tr>
    <tr>
        <td>${customer.id}</td>
        <td>${customer.username}</td>
        <td>${customer.jobs}</td>
        <td>${customer.phone}</td>
    </tr>
</table>
```

我要自學網 WWW.51ZXW.NET www.51zxw.net 版权所有 盗版必究

7 将项目发布到Tomcat服务器并启动，在浏览器中访问地址http://localhost:8080/chapter17/findCustomerById?id=1，显示效果如下所示：



从上图可以看出，通过浏览器已经成功查询出了t_customer表中id为1的客户信息，这也就说明SSM框架整合成功。

本篇结束

希望大家学有所成

www.51zxw.net 版权所有 盗版必究

END