第3天: SpringMVC

一、目标

1. 掌握ssm整合(SpringMVC+Spring+Mybatis)

二、SSM整合

> 所谓SSM，即spring MVC + spring +mybatis，标准的MVC设计模式，将整个系统划分为表现层、controller层、service层、DAO层四层，因此，在搭建框架时项目的目录结构最好要包含这些，方便管理和查看。当然了，spring MVC、spring、mybatis各司其职，在整个框架中有着不同的作用

- Spring是一个轻量级的控制反转（IoC）和面向切面（AOP）的容器框架，主要实现 **业务对象管理**；
- Spring MVC框架，通过实现MVC模式将数据、业务与展现进行分离，主要负责 **请求的转发和视图管理**；
- MyBatis 是一个基于Java的持久层框架，作为 **数据对象的持久化** 引擎；

2.1 项目分层

- 控制层 (controller): SpringMVC
- 业务层 (service): Spring
- 持久层 (dao): Mybatis

2.2 环境准备

2.2.1 数据准备

```sql
CREATE TABLE `account` (
  `id` int(11) NOT NULL AUTO_INCREMENT COMMENT '主键',
  `uid` int(11) DEFAULT '1' COMMENT '用户编号',
  `money` decimal(10,2) DEFAULT '0.00' COMMENT '余额',
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=106 DEFAULT CHARSET=utf8;

-- ----------------------------
-- Records of account
-- ----------------------------
INSERT INTO `account` VALUES ('1', '1', '10.00');
INSERT INTO `account` VALUES ('2', '10', '0.00');
INSERT INTO `account` VALUES ('3', '24', '99.00');
```

2.2.2 项目环境

- 工程名称: mvc-day03-ssm
- 添加依赖: pom.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
    <parent>
        <artifactId>mvc3</artifactId>
        <groupId>com.itheima</groupId>
        <version>1.0-SNAPSHOT</version>
    </parent>
    <modelVersion>4.0.0</modelVersion>

    <artifactId>mvc-day03-ssm</artifactId>

    <dependencies>
        <!-- 1. Spring相关 依赖 -->
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-context</artifactId>
            <version>5.1.7.RELEASE</version>
        </dependency>
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-tx</artifactId>
            <version>5.1.7.RELEASE</version>
        </dependency>
        <dependency>
            <groupId>org.aspectj</groupId>
            <artifactId>aspectjweaver</artifactId>
            <version>1.8.13</version>
        </dependency>
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-jdbc</artifactId>
            <version>5.1.7.RELEASE</version>
        </dependency>
        <dependency>
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-java</artifactId>
            <version>8.0.17</version>
        </dependency>
        <dependency>
            <groupId>com.alibaba</groupId>
            <artifactId>druid</artifactId>
            <version>1.1.12</version>
        </dependency>
        <!-- 2. SpringMVC相关 依赖 -->
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-web</artifactId>
            <version>5.1.7.RELEASE</version>
        </dependency>
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-webmvc</artifactId>
            <version>5.1.7.RELEASE</version>
        </dependency>
        <!-- 3. Mybatis相关 依赖 -->
        <dependency>
            <groupId>org.mybatis</groupId>
```

```
        <dependency>
            <groupId>org.mybatis</groupId>
            <artifactId>mybatis</artifactId>
            <version>3.5.2</version>
        </dependency>
        <!-- 4. Jsp相关 依赖 -->
        <dependency>
            <groupId>javax.servlet</groupId>
            <artifactId>jsp-api</artifactId>
            <version>2.0</version>
            <scope>provided</scope>
        </dependency>
        <dependency>
            <groupId>javax.servlet</groupId>
            <artifactId>servlet-api</artifactId>
            <version>2.5</version>
            <scope>provided</scope>
        </dependency>
        <dependency>
            <groupId>jstl</groupId>
            <artifactId>jstl</artifactId>
            <version>1.2</version>
        </dependency>
        <!-- 5. SpringTest 依赖 -->
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-test</artifactId>
            <version>5.1.7.RELEASE</version>
        </dependency>
        <dependency>
            <groupId>junit</groupId>
            <artifactId>junit</artifactId>
            <version>4.12</version>
        </dependency>
        <!-- 6. Log4j 依赖 -->
        <dependency>
            <groupId>log4j</groupId>
            <artifactId>log4j</artifactId>
            <version>1.2.17</version>
        </dependency>
        <dependency>
            <groupId>org.slf4j</groupId>
            <artifactId>slf4j-api</artifactId>
            <version>1.7.26</version>
        </dependency>
        <dependency>
            <groupId>org.slf4j</groupId>
            <artifactId>slf4j-log4j12</artifactId>
            <version>1.7.7</version>
        </dependency>
    </dependencies>
</project>
```

- 创建实体: com.itheima.ssm.domain.Account

```java
/**
 * 账户实体类.
 *
 * @author : Jason.lee
 * @version : 1.0
 */
public class Account {
    private Integer id;
    private Integer uid;
    private Double money;

    public Integer getId() {
        return id;
    }

    public void setId(Integer id) {
        this.id = id;
    }

    public Integer getUid() {
        return uid;
    }

    public void setUid(Integer uid) {
        this.uid = uid;
    }

    public Double getMoney() {
        return money;
    }

    public void setMoney(Double money) {
        this.money = money;
    }

    @Override
    public String toString() {
        return "Account{" +
                "id=" + id +
                ", uid=" + uid +
                ", money=" + money +
                '}';
    }
}
```

2.3 单独Spring环境

2.3.1 业务层

- com.itheima.ssm.service.AccountService

```java
package com.itheima.ssm.service;

import com.itheima.ssm.domain.Account;
import org.springframework.stereotype.Service;
```

```java
import java.util.List;

/**
 * 账户业务.
 *
 * @author : Jason.lee
 * @version : 1.0
 */
@Service
public class AccountService {


    /**
     * 查询所有账户
     */
    public List<Account> findAll(){
        System.out.println("查询所有账户业务..");
        return null;
    }

    /**
     * 保存账户
     */
    @Transactional
    public void save(Account account){
        System.out.println("保存账户业务..");
    }
}
```

2.3.2 添加配置

- applicationContext.xml

```xml
<?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"
       xmlns:tx="http://www.springframework.org/schema/tx"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
       http://www.springframework.org/schema/beans/spring-beans.xsd
       http://www.springframework.org/schema/context
       https://www.springframework.org/schema/context/spring-
context.xsd
       http://www.springframework.org/schema/tx
       http://www.springframework.org/schema/tx/spring-tx.xsd">

    <!-- Spring单独环境 -->
    <!-- 1. 扫描Spring组件注解 -->
    <context:component-scan base-package="com.itheima.ssm.service"/>
    <!-- 2. 开启Spring事务注解 -->
    <tx:annotation-driven/>
</beans>
```

2.3.3 单元测试

```xml
<?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"
       xmlns:tx="http://www.springframework.org/schema/tx"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
       http://www.springframework.org/schema/beans/spring-beans.xsd
       http://www.springframework.org/schema/context
       https://www.springframework.org/schema/context/spring-context.xsd
       http://www.springframework.org/schema/tx
       http://www.springframework.org/schema/tx/spring-tx.xsd">

    <!-- 数据库基础配置 -->
    <context:property-placeholder location="classpath:db.properties"/>
    <bean id="dataSource" class="com.alibaba.druid.pool.DruidDataSource">
        <property name="driverClassName" value="${db.driver}"/>
        <property name="url" value="${db.url}"/>
        <property name="username" value="${db.username}"/>
        <property name="password" value="${db.password}"/>
    </bean>


    <!-- Spring单独环境 -->
    <!-- 1. 扫描Spring组件注解 -->
    <context:component-scan base-package="com.itheima.ssm.service"/>
    <!-- 2. 开启Spring事务注解 -->
    <tx:annotation-driven/>
    <!-- 3. 配置事务管理器 -->
    <bean id="transactionManager" class="org.springframework.jdbc.datasource.DataSourceTransactionManager">
        <property name="dataSource" ref="dataSource"/>
    </bean>
</beans>
```

2.4 单独SpringMVC环境

2.4.1 控制层

- com.itheima.ssm.controller.AccountController2.4.2 添加配置
- web.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="http://java.sun.com/xml/ns/javaee"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
    version="2.5">


    <!-- 1. SpringMVC前端控制器 -->
    <!-- 1.1 处理的请求路径 -->
```

```
        <url-pattern>/</url-pattern>
    </servlet-mapping>
    <!-- 1.2 处理请求的控制器 -->
    <servlet>
        <!-- 1.2.1 定义控制器 -->
        <servlet-name>dispatcherServlet</servlet-name>
        <servlet-
class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
        <!-- 1.2.2 设置控制器的配置 -->
        <init-param>
            <param-name>contextConfigLocation</param-name>
            <param-value>classpath:springMVC.xml</param-value>
        </init-param>
    </servlet>
</web-app>
```

2.4.3 单元测试

- WEB-INF/jsp/account/list.jsp

```
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%--
  Created by IntelliJ IDEA.
  User: Jason
  Date: 2019/8/30
  Time: 13:03
  To change this template use File | Settings | File Templates.
--%>
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<html>
<head>
    <title>账户列表</title>
</head>
<body>
<h3>账户列表</h3>
<table>
    <tr>
        <th>账户编号</th>
        <th>用户编号</th>
        <th>账户余额</th>
    </tr>
    <c:forEach items="${list}" var="account">
        <tr>
            <td>${account.id}</td>
            <td>${account.uid}</td>
            <td>${account.money}</td>
        </tr>
    </c:forEach>
</table>
</body>
</html>
```

2.5 SpringMVC与Spring整合

- 希望整合Spring获取到IOC中的业务对象注入到控制层中
- com.itheima.ssm.controller.AccountController

```java
package com.itheima.ssm.controller;

import com.itheima.ssm.domain.Account;
import com.itheima.ssm.service.AccountService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.RequestMapping;

import java.util.List;

/**
 * 账户请求处理器(控制器).
 *
 * @author : Jason.lee
 * @version : 1.0
 */
@Controller
public class AccountController {


    /**
     * 希望整合Spring获取到IOC中的业务对象
     */
    @Autowired
    AccountService accountService;

    /**
     * 查询账户列表
     * @return 返回视图
     */
    @RequestMapping("list")
    public String list(Model model){
        List<Account> list = accountService.findAll();
        model.addAttribute("list", list);
        return "account/list";
    }
}
```

2.5.2 配置

- 配置Spring提供的监听器: 在容器启动时, 加载Spring配置文件启动IOC容器
- web.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="http://java.sun.com/xml/ns/javaee"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
    version="2.5">

```

```
    <!-- 1.1 处理的请求路径 -->
    <servlet-mapping>
        <servlet-name>dispatcherServlet</servlet-name>
        <url-pattern>/</url-pattern>
    </servlet-mapping>
    <!-- 1.2 处理请求的控制器 -->
    <servlet>
        <!-- 1.2.1 定义控制器 -->
        <servlet-name>dispatcherServlet</servlet-name>
        <servlet-
class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
        <!-- 1.2.2 设置控制器的配置 -->
        <init-param>
            <param-name>contextConfigLocation</param-name>
            <param-value>classpath:springMVC.xml</param-value>
        </init-param>
    </servlet>

    <!-- 2. Spring配置  -->
    <!-- 2.1 配置Spring提供的ServletContext监听器: Servlet容器启动时触发 -->
    <listener>
        <listener-
class>org.springframework.web.context.ContextLoaderListener</listener-
class>
    </listener>
    <!-- 2.2 设置监听器执行参数: Spring配置文件 -->
    <context-param>
        <param-name>contextConfigLocation</param-name>
        <param-value>classpath:applicationContext.xml</param-value>
    </context-param>
</web-app>
```

2.6 单独Mybatis环境

2.6.1 持久层

- com.itheima.ssm.dao.AccountDao

```
package com.itheima.ssm.dao;

import com.itheima.ssm.domain.Account;
import org.apache.ibatis.annotations.Insert;
import org.apache.ibatis.annotations.Select;

import java.util.List;

/**
 * 账户持久层.
 *
 * @author : Jason.lee
 * @version : 1.0
 */
public interface AccountDao {
```

```
     * 查询所有账户
     */
    @Select("select * from account")
    List<Account> findAll();

    /**
     * 保存账户
     */
    @Insert("insert into account(id, uid, money) values(#{id}, #{uid}, #{money})")
    void save(Account account);
}
```

2.6.2 添加配置

- 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. 加载数据库配置文件 -->
    <properties resource="db.properties"/>

    <settings>
        <!-- 开启二级缓存 -->
        <setting name="cacheEnabled" value="false"/>
        <!-- 设置超时时间 -->
        <setting name="defaultStatementTimeout" value="5"/>
        <!-- 使用驼峰命名法 -->
        <setting name="mapUnderscoreToCamelCase" value="true"/>
        <!-- 允许JDBC自动生成主键 -->
        <setting name="useGeneratedKeys" value="true"/>
    </settings>

    <!-- 2. 设置Mybatis环境 -->
    <environments default="dev">
        <environment id="dev">
            <transactionManager type="JDBC"></transactionManager>
            <dataSource type="pooled">
                <property name="driver" value="${db.driver}"/>
                <property name="url" value="${db.url}"/>
                <property name="username" value="${db.username}"/>
                <property name="password" value="${db.password}"/>
            </dataSource>
        </environment>
    </environments>

    <!-- 3. 给指定包下的接口动态生成持久层代理对象 -->
    <mappers>
        <package name="com.itheima.ssm.dao"/>
    </mappers>
</configuration>
```

- SsmTests

```
import com.itheima.ssm.dao.AccountDao;
import com.itheima.ssm.domain.Account;
import com.itheima.ssm.service.AccountService;
import org.apache.ibatis.io.Resources;
import org.apache.ibatis.session.SqlSession;
import org.apache.ibatis.session.SqlSessionFactory;
import org.apache.ibatis.session.SqlSessionFactoryBuilder;
import org.apache.ibatis.session.SqlSessionManager;
import org.junit.Test;
import org.junit.runner.RunWith;
import org.mybatis.spring.SqlSessionFactoryBean;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.test.context.ContextConfiguration;
import org.springframework.test.context.junit4.SpringJUnit4ClassRunner;

import java.io.InputStream;
import java.util.List;

/**
 * SSM整合单元测试代码.
 *
 * @author : Jason.lee
 * @version : 1.0
 */
@RunWith(SpringJUnit4ClassRunner.class)
@ContextConfiguration(locations = "classpath:applicationContext.xml")
public class SsmTests {

    @Autowired
    AccountService accountService;

    @Test
    public void testSpring (){
        accountService.findAll();
        Account account = new Account();
        account.setId(10);
        account.setUid(10);
        account.setMoney(10D);
        accountService.save(account);
    }

    @Test
    public void testMybatis (){
        // 1. 获取配置文件输入流
        InputStream in =
this.getClass().getClassLoader().getResourceAsStream("sqlMapConfig.xml"
);
        // 2. 创建SqlSessionFactoryBuilder
        SqlSessionFactory factory = new
SqlSessionFactoryBuilder().build(in);
        // 3. 获取SqlSession
        SqlSession sqlSession = factory.openSession();
        // 4. 获取持久层动态代理对象
```

```
        List<Account> list = accountDao.findAll();
        Account account = new Account();
        account.setId(10);
        account.setUid(10);
        account.setMoney(10D);
        accountDao.save(account);
        // 6. 事务提交
        sqlSession.commit();
        sqlSession.close();
    }
}
```

2.7 Spring与Mybatis整合

2.7.1 需求

- 希望整合Spring获取到IOC中的业务对象注入到业务层中
- com.itheima.ssm.service.AccountService

```
package com.itheima.ssm.service;

import com.itheima.ssm.dao.AccountDao;
import com.itheima.ssm.domain.Account;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;

import java.util.List;

/**
 * 账户业务.
 *
 * @author : Jason.lee
 * @version : 1.0
 */
@Service
public class AccountService {

    /**
     * 希望整合Spring获取到IOC中的持久层对象
     */
    @Autowired
    AccountDao accountDao;


    /**
     * 查询所有账户
     */
    public List<Account> findAll(){
        return accountDao.findAll();
    }

    /**
     * 保存账户
```

```
            accountDao.save(account);
        }
    }
```

2.7.2 配置

- applicationContext.xml

```
<?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"
       xmlns:tx="http://www.springframework.org/schema/tx"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
       http://www.springframework.org/schema/beans/spring-beans.xsd
       http://www.springframework.org/schema/context
       https://www.springframework.org/schema/context/spring-context.xsd
       http://www.springframework.org/schema/tx
       http://www.springframework.org/schema/tx/spring-tx.xsd">

    <!-- 数据库基础配置 -->
    <context:property-placeholder location="classpath:db.properties"/>
    <bean id="dataSource" class="com.alibaba.druid.pool.DruidDataSource">
        <property name="driverClassName" value="${db.driver}"/>
        <property name="url" value="${db.url}"/>
        <property name="username" value="${db.username}"/>
        <property name="password" value="${db.password}"/>
    </bean>


    <!-- Spring单独环境 -->
    <!-- 1. 扫描Spring组件注解 -->
    <context:component-scan base-package="com.itheima.ssm.service"/>
    <!-- 2. 开启Spring事务注解 -->
    <tx:annotation-driven/>
    <!-- 3. 配置事务管理器 -->
    <bean id="transactionManager" class="org.springframework.jdbc.datasource.DataSourceTransactionManager">
        <property name="dataSource" ref="dataSource"/>
    </bean>

    <!-- Spring整合Mybatis -->
    <!-- 1. 创建SqlSessionFactory -->
    <bean class="org.mybatis.spring.SqlSessionFactoryBean">
        <property name="dataSource" ref="dataSource"/>
    </bean>
    <!-- 2. 创建持久层动态代理对象 -->
    <bean class="org.mybatis.spring.mapper.MapperScannerConfigurer">
        <property name="basePackage" value="com.itheima.ssm.dao"/>
    </bean>
</beans>
```

- 浏览器操作