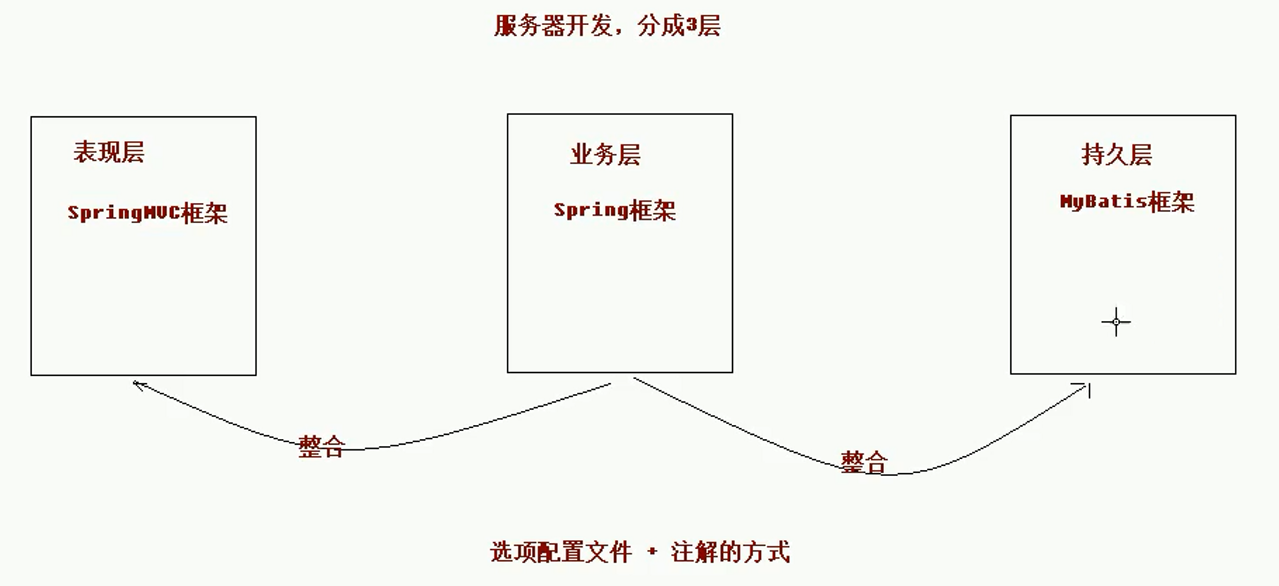
# SSM概述

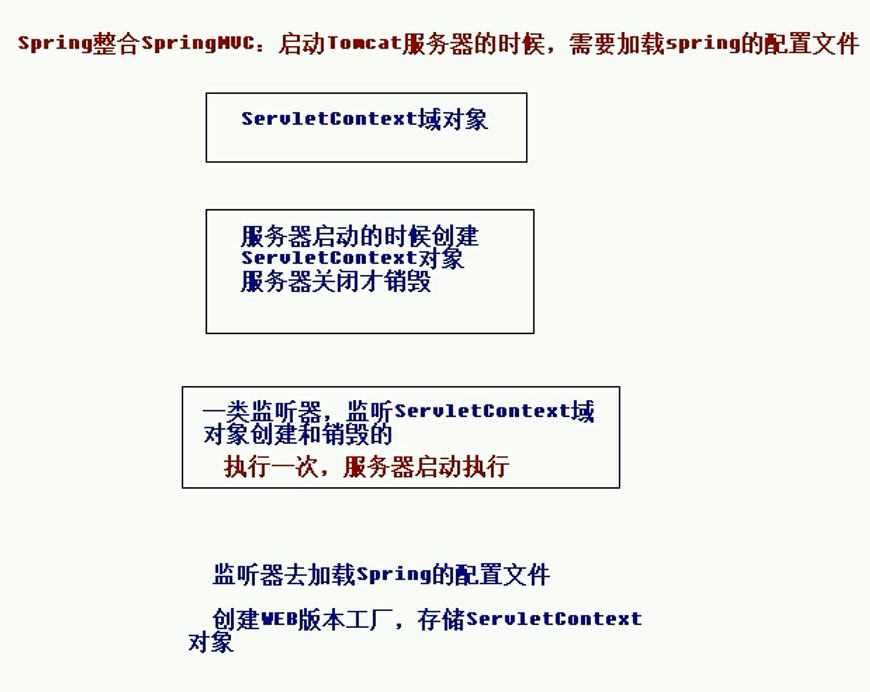
SSM：Spring + SpringMVC + Mybatis进行整合



整合方式：XML + 注解

整合步骤：

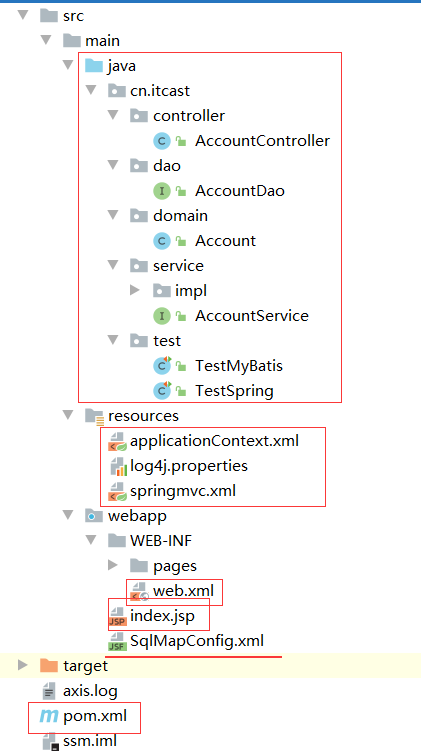
1. 搭建整合的Maven环境
2. 配置Spring
3. 整合SpringMVC



1. 整合Mybatis
2. 配置事务

# 文件

## 文件目录



## 配置文件

### 项目配置文件

**pom.xml（Maven项目配置文件）**

<?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"

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

<modelVersion>4.0.0</modelVersion>

<groupId>cn.itcast</groupId>

<artifactId>ssm</artifactId>

<version>1.0-SNAPSHOT</version>

<packaging>war</packaging>

<name>ssm Maven Webapp</name>

<!-- FIXME change it to the project's website -->

<url>http://www.example.com</url>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<maven.compiler.source>1.8</maven.compiler.source>

<maven.compiler.target>1.8</maven.compiler.target>

<spring.version>5.0.2.RELEASE</spring.version>

<slf4j.version>1.6.6</slf4j.version>

<log4j.version>1.2.12</log4j.version>

<mysql.version>5.1.6</mysql.version>

<mybatis.version>3.4.5</mybatis.version>

</properties>

<dependencies>

<!-- spring -->

<dependency>

<groupId>org.aspectj</groupId>

<artifactId>aspectjweaver</artifactId>

<version>1.6.8</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-aop</artifactId>

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

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</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-webmvc</artifactId>

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

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-test</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>junit</groupId>

<artifactId>junit</artifactId>

<version>4.12</version>

<scope>compile</scope>

</dependency>

<dependency>

<groupId>mysql</groupId>

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

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

</dependency>

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>servlet-api</artifactId>

<version>2.5</version>

<scope>provided</scope>

</dependency>

<dependency>

<groupId>javax.servlet.jsp</groupId>

<artifactId>jsp-api</artifactId>

<version>2.0</version>

<scope>provided</scope>

</dependency>

<dependency>

<groupId>jstl</groupId>

<artifactId>jstl</artifactId>

<version>1.2</version>

</dependency>

<!-- log start -->

<dependency>

<groupId>log4j</groupId>

<artifactId>log4j</artifactId>

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

</dependency>

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-api</artifactId>

<version>${slf4j.version}</version>

</dependency>

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-log4j12</artifactId>

<version>${slf4j.version}</version>

</dependency>

<!-- log end -->

<dependency>

<groupId>org.mybatis</groupId>

<artifactId>mybatis</artifactId>

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

</dependency>

<dependency>

<groupId>org.mybatis</groupId>

<artifactId>mybatis-spring</artifactId>

<version>1.3.0</version>

</dependency>

<dependency>

<groupId>c3p0</groupId>

<artifactId>c3p0</artifactId>

<version>0.9.1.2</version>

<type>jar</type>

<scope>compile</scope>

</dependency>

</dependencies>

<build>

<finalName>ssm</finalName>

<pluginManagement><!-- lock down plugins versions to avoid using Maven defaults (may be moved to parent pom) -->

<plugins>

<plugin>

<artifactId>maven-clean-plugin</artifactId>

<version>3.0.0</version>

</plugin>

<!-- see http://maven.apache.org/ref/current/maven-core/default-bindings.html#Plugin\_bindings\_for\_war\_packaging -->

<plugin>

<artifactId>maven-resources-plugin</artifactId>

<version>3.0.2</version>

</plugin>

<plugin>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.7.0</version>

</plugin>

<plugin>

<artifactId>maven-surefire-plugin</artifactId>

<version>2.20.1</version>

</plugin>

<plugin>

<artifactId>maven-war-plugin</artifactId>

<version>3.2.0</version>

</plugin>

<plugin>

<artifactId>maven-install-plugin</artifactId>

<version>2.5.2</version>

</plugin>

<plugin>

<artifactId>maven-deploy-plugin</artifactId>

<version>2.8.2</version>

</plugin>

</plugins>

</pluginManagement>

</build>

</project>

**web.xml（web配置）**

<!DOCTYPE web-app PUBLIC

"-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"

"http://java.sun.com/dtd/web-app\_2\_3.dtd" >

<web-app>

<display-name>Archetype Created Web Application</display-name>

<!--配置Spring的监听器，默认只加载WEB-INF目录下的applicationContext.xml配置文件-->

<listener>

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

</listener>

<!--设置配置文件的路径-->

<context-param>

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

<param-value>classpath:applicationContext.xml</param-value>

</context-param>

<context-param>

<param-name/>

<param-value/>

</context-param>

<!--配置前端控制器-->

<servlet>

<servlet-name>dispatcherServlet</servlet-name>

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

<!--加载springmvc.xml配置文件-->

<init-param>

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

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

</init-param>

<!--启动服务器，创建该servlet-->

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

</servlet>

<servlet-mapping>

<servlet-name>dispatcherServlet</servlet-name>

<url-pattern>/</url-pattern>

</servlet-mapping>

<!--解决中文乱码的过滤器-->

<filter>

<filter-name>characterEncodingFilter</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>characterEncodingFilter</filter-name>

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

</filter-mapping>

</web-app>

### 三大框架配置文件

**applicationContext.xml（Spring配置文件）**

<?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:aop="http://www.springframework.org/schema/aop"

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

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

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

http://www.springframework.org/schema/aop/spring-aop.xsd

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

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

<!--开启注解的扫描，希望处理service和dao，controller不需要Spring框架去处理-->

<context:component-scan base-package="cn.itcast" >

<!--配置哪些注解不扫描-->

<context:exclude-filter type="annotation" expression="org.springframework.stereotype.Controller" />

</context:component-scan>

<!--Spring整合MyBatis框架-->

<!--配置连接池-->

<bean id="dataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource">

<property name="driverClass" value="com.mysql.jdbc.Driver"/>

<property name="jdbcUrl" value="jdbc:mysql:///ssm"/>

<property name="user" value="root"/>

<property name="password" value="root"/>

</bean>

<!--配置SqlSessionFactory工厂-->

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

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

</bean>

<!--配置AccountDao接口所在包-->

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

<property name="basePackage" value="cn.itcast.dao"/>

</bean>

<!--配置Spring框架声明式事务管理-->

<!--配置事务管理器-->

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

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

</bean>

<!--配置事务通知-->

<tx:advice id="txAdvice" transaction-manager="transactionManager">

<tx:attributes>

<tx:method name="find\*" read-only="true"/>

<tx:method name="\*" isolation="DEFAULT"/>

</tx:attributes>

</tx:advice>

<!--配置AOP增强-->

<aop:config>

<aop:advisor advice-ref="txAdvice" pointcut="execution(\* cn.itcast.service.impl.\*ServiceImpl.\*(..))"/>

</aop:config>

</beans>

**springmvc.xml（SpringMVC配置）**

<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:mvc="http://www.springframework.org/schema/mvc" xmlns:context="http://www.springframework.org/schema/context"

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.xsd

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

http://www.springframework.org/schema/mvc/spring-mvc.xsd

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

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

<!--开启注解扫描，只扫描Controller注解-->

<context:component-scan base-package="cn.itcast">

<context:include-filter type="annotation" expression="org.springframework.stereotype.Controller" />

</context:component-scan>

<!--配置的视图解析器对象-->

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

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

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

</bean>

<!--过滤静态资源-->

<mvc:resources location="/css/" mapping="/css/\*\*" />

<mvc:resources location="/images/" mapping="/images/\*\*" />

<mvc:resources location="/js/" mapping="/js/\*\*" />

<!--开启SpringMVC注解的支持-->

<mvc:annotation-driven/>

</beans>

**SqlMapConfig.xml（Mybatis配置文件）**

该文件已经被整合到spring的配置文件中！！！

**log4j.properties**

# Set root category priority to INFO and its only appender to CONSOLE.

#log4j.rootCategory=INFO, CONSOLE debug info warn error fatal

log4j.rootCategory=info, CONSOLE, LOGFILE

# Set the enterprise logger category to FATAL and its only appender to CONSOLE.

log4j.logger.org.apache.axis.enterprise=FATAL, CONSOLE

# CONSOLE is set to be a ConsoleAppender using a PatternLayout.

log4j.appender.CONSOLE=org.apache.log4j.ConsoleAppender

log4j.appender.CONSOLE.layout=org.apache.log4j.PatternLayout

log4j.appender.CONSOLE.layout.ConversionPattern=%d{ISO8601} %-6r [%15.15t] %-5p %30.30c %x - %m\n

# LOGFILE is set to be a File appender using a PatternLayout.

log4j.appender.LOGFILE=org.apache.log4j.FileAppender

log4j.appender.LOGFILE.File=d:\axis.log

log4j.appender.LOGFILE.Append=true

log4j.appender.LOGFILE.layout=org.apache.log4j.PatternLayout

log4j.appender.LOGFILE.layout.ConversionPattern=%d{ISO8601} %-6r [%15.15t] %-5p %30.30c %x - %m\n

## java文件

### java/domain

public class Account implements Serializable{

private Integer id;

private String name;

private Double money;

public Integer getId() {

return id;

}

public void setId(Integer id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public Double getMoney() {

return money;

}

public void setMoney(Double money) {

this.money = money;

}

@Override

public String toString() {

return "Account{" +

"id=" + id +

", name='" + name + '\'' +

", money=" + money +

'}';

}

}

### java/dao

/\*\*

\* 帐户dao接口

\*/

@Repository

public interface AccountDao {

// 查询所有账户

@Select("select \* from account")

public List<Account> findAll();

// 保存帐户信息

@Insert("insert into account (name,money) values (#{name},#{money})")

public void saveAccount(Account account);

}

### java/service

AccountService.java

public interface AccountService {

// 查询所有账户

public List<Account> findAll();

// 保存帐户信息

public void saveAccount(Account account);

}

AccountServiceImpl.java

@Service("accountService")

public class AccountServiceImpl implements AccountService{

@Autowired

private AccountDao accountDao;

public List<Account> findAll() {

System.out.println("业务层：查询所有账户...");

return accountDao.findAll();

}

public void saveAccount(Account account) {

System.out.println("业务层：保存帐户...");

accountDao.saveAccount(account);

}

}

### java/controller

/\*\*

\* 帐户web

\*/

@Controller

@RequestMapping("/account")

public class AccountController {

@Autowired

private AccountService accountService;

@RequestMapping("/findAll")

public String findAll(Model model){

System.out.println("表现层：查询所有账户...");

// 调用service的方法

List<Account> list = accountService.findAll();

model.addAttribute("list",list);

return "list";

}

/\*\*

\* 保存

\* @return

\*/

@RequestMapping("/save")

public void save(Account account, HttpServletRequest request, HttpServletResponse response) throws IOException {

accountService.saveAccount(account);

response.sendRedirect(request.getContextPath()+"/account/findAll");

return;

}

}

### test

public class TestSpring {

@Test

public void run1(){

// 加载配置文件

ApplicationContext ac = new ClassPathXmlApplicationContext("classpath:applicationContext.xml");

// 获取对象

AccountService as = (AccountService) ac.getBean("accountService");

// 调用方法

as.findAll();

}

}

public class TestMyBatis {

/\*\*

\* 测试查询

\* @throws Exception

\*/

@Test

public void run1() throws Exception {

// 加载配置文件

InputStream in = Resources.getResourceAsStream("SqlMapConfig.xml");

// 创建SqlSessionFactory对象

SqlSessionFactory factory = new SqlSessionFactoryBuilder().build(in);

// 创建SqlSession对象

SqlSession session = factory.openSession();

// 获取到代理对象

AccountDao dao = session.getMapper(AccountDao.class);

// 查询所有数据

List<Account> list = dao.findAll();

for(Account account : list){

System.out.println(account);

}

// 关闭资源

session.close();

in.close();

}

/\*\*

\* 测试保存

\* @throws Exception

\*/

@Test

public void run2() throws Exception {

Account account = new Account();

account.setName("熊大");

account.setMoney(400d);

// 加载配置文件

InputStream in = Resources.getResourceAsStream("SqlMapConfig.xml");

// 创建SqlSessionFactory对象

SqlSessionFactory factory = new SqlSessionFactoryBuilder().build(in);

// 创建SqlSession对象

SqlSession session = factory.openSession();

// 获取到代理对象

AccountDao dao = session.getMapper(AccountDao.class);

// 保存

dao.saveAccount(account);

// 提交事务

session.commit();

// 关闭资源

session.close();

in.close();

}

}

### jsp页面

index.jsp

<%--

Created by IntelliJ IDEA.

User: Administrator

Date: 2018/5/6

Time: 0:22

To change this template use File | Settings | File Templates.

--%>

<%@ page contentType="text/html;charset=UTF-8" language="java" %>

<html>

<head>

<title>Title</title>

</head>

<body>

<a href="account/findAll">测试查询</a>

<h3>测试包</h3>

<form action="account/save" method="post">

姓名：<input type="text" name="name" /><br/>

金额：<input type="text" name="money" /><br/>

<input type="submit" value="保存"/><br/>

</form>

</body>

</html>