MyBatis 动态 SQL

- choose、when 标签

choose、when 标签和 if 标签用法很类似。

```
<select id="findByUser" parameterType="com.southwind.entity.User"
resultType="com.southwind.entity.User">
    select * from t_user
    <where>
        <choose>
            <when test="id!=0">
                id = #{id}
            </when>
            <when test="username!=null">
                and username = #{username}
            </when>
            <when test="password!=null">
                and password = #{password}
            </when>
            <when test="age!=0">
                and age = #{age}
            </when>
        </choose>
    </where>
</select>
```

- trim 标签

trim 标签中的 prefix 和 suffix 属性会被用于生成实际的 SQL 语句，会和标签内部的语句拼接，如果语句某个部分之前或之后遇到 prefixOverrides 或者 suffixOverrides 属性中指定的值，MyBatis 框架会自动将其删除。在指定多个值的时候，每个值后面都需要追加一个空格，保证不会和后面的 SQL 连接在一起。

```
<select id="findByUser" parameterType="com.southwind.entity.User"
resultType="com.southwind.entity.User">
    select * from t_user
    <trim prefix="where" prefixOverrides="and">
        <if test="id!=0">
            id = #{id}
        </if>
        <if test="username!=null">
            and username = #{username}
        </if>
        <if test="password!=null">
            and password = #{password}
        </if>
```

```
        <if test="age!=0">
            and age = #{age}
        </if>
    </trim>
</select>
```

- set 标签

set 标签用于 update 操作，会自动根据参数选择生成 SQL 语句。

```
<update id="update" parameterType="com.southwind.entity.User">
    update t_user
    <set>
        <if test="username!=null">
            username = #{username},
        </if>
        <if test="password!=null">
            password = #{password},
        </if>
        <if test="age!=0">
            age = #{age}
        </if>
    </set>
    where id = #{id}
</update>
```

- foreache

foreache 标签可以迭代生成一系列值，这个表情主要用于 SQL 的 in 语句。

修改 User 对象，添加一个 List 类型的属性 ids。

```
<select id="findByIds" parameterType="com.southwind.entity.User"
resultType="com.southwind.entity.User">
    select * from t_user
    <where>
        <foreach collection="ids" open="id in (" close=")" item="id"
separator=",">
            #{id}
        </foreach>
    </where>
</select>
```

SSM 框架整合

SSM：Spring + Spring MVC + MyBatis

Spring：Spring 是一个开源框架，于2003年兴起的一个轻量级的 Java 开源框架，它是为了解决企业级应用开发过于复杂而创建的。Spring 框架使用基本的 JavaBean 来完成之前只能由 EJB 完成的功能，Spring 框架的用途不仅限于服务端的开发，从简单性、可测试性和松耦合的角度来看，Java 应用都可以用 Spring 框架进行优化。简单来说，Spring 是一个轻量级的 IoC 和 AOP 的容器框架。

Spring MVC：Spring MVC 属于 Spring Framework 的一个子模块，已经整合到了 Spring Web 中，Spring MVC 分离了控制器、模型对象、处理器等角色，这种分离让它们更容易进行定制开发。

MyBatis：MyBatis 是 apache 的一个开源项目 iBatis，2010年更名为 MyBatis，MyBatis 是一个基于 Java 的持久层框架，MyBatis 提供的持久层框架包括 SQL Maps 和 Data Access Objects （DAO），MyBatis 框架消除了几乎所有的 JDBC 代码和参数的手工设置以及结果集的检索和解析，MyBatis 使用简单的 XML 或者注解用于配置和映射，将接口和 Java Bean 映射程数据库中的记录。

SSM 框架整合是当下比较主流的企业级项目技术选型，三个框架分别负责不同的功能，整合起来共同支持企业级项目的开发需求。

Spring MVC 负责 MVC 设计模式的实现，MyBatis 负责数据库持久层，Spring 的 IoC 容器来管理 Spring MVC 和 MyBatis 相关对象的创建注入，Spring 的 AOP 负责事务管理。

- pom.xml 中添加依赖

```
<dependencies>

    <!-- Spring MVC -->
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-webmvc</artifactId>
        <version>5.0.11.RELEASE</version>
    </dependency>

    <!-- MyBatis -->
    <dependency>
        <groupId>org.mybatis</groupId>
        <artifactId>mybatis</artifactId>
        <version>3.4.5</version>
    </dependency>

    <!-- MyBatis 整合 Spring 的适配包 -->
    <dependency>
        <groupId>org.mybatis</groupId>
        <artifactId>mybatis-spring</artifactId>
        <version>1.3.1</version>
    </dependency>

    <!-- MySQL 驱动 -->
    <dependency>
```

```
        <groupId>mysql</groupId>
        <artifactId>mysql-connector-java</artifactId>
        <version>8.0.11</version>
    </dependency>

    <!-- C3P0 数据源 -->
    <dependency>
        <groupId>c3p0</groupId>
        <artifactId>c3p0</artifactId>
        <version>0.9.1</version>
    </dependency>

    <!-- JSTL -->
    <dependency>
        <groupId>jstl</groupId>
        <artifactId>jstl</artifactId>
        <version>1.2</version>
    </dependency>

    <!-- ServletAPI -->
    <dependency>
        <groupId>javax.servlet</groupId>
        <artifactId>javax.servlet-api</artifactId>
        <version>3.1.0</version>
    </dependency>

</dependencies>

<build>
    <finalName>0219</finalName>

    <resources>
      <resource>
        <directory>src/main/java</directory>
        <includes>
          <include>**/*.xml</include>
        </includes>
      </resource>
    </resources>

    <pluginManagement><!-- lock down plugins versions to avoid using Maven
defaults (may be moved to parent pom) -->
      <plugins>
        <plugin>
          <artifactId>maven-clean-plugin</artifactId>
          <version>3.1.0</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>3.8.0</version>
        </plugin>
        <plugin>
          <artifactId>maven-surefire-plugin</artifactId>
          <version>2.22.1</version>
        </plugin>
        <plugin>
          <artifactId>maven-war-plugin</artifactId>
          <version>3.2.2</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>
```

- web.xml 配置启动 Spring、Spring MVC，字符编码过滤器、加载静态资源（Spring MVC 会拦截所有请求，导致 JSP 页面中对 js 和 css 的引用失效）

```
<!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>

  <!-- 启动 Spring -->
  <context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>classpath:spring.xml</param-value>
  </context-param>
  <listener>
    <listener-
class>org.springframework.web.context.ContextLoaderListener</listener-class>
  </listener>

  <!-- Spring MVC 前置控制器 -->
```

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

  <servlet-mapping>
    <servlet-name>dispatcherServlet</servlet-name>
    <url-pattern>/</url-pattern>
  </servlet-mapping>

  <!-- 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>

  <!-- 加载静态资源 -->
  <servlet-mapping>
    <servlet-name>default</servlet-name>
    <url-pattern>*.js</url-pattern>
  </servlet-mapping>

  <servlet-mapping>
    <servlet-name>default</servlet-name>
    <url-pattern>*.css</url-pattern>
  </servlet-mapping>

</web-app>
```

- SSM 框架的整合是通过设置各自的配置文件来完成的，配置文件存放在 resources 目录下。

spring.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:mvc="http://www.springframework.org/schema/mvc"
       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/mvc
       http://www.springframework.org/schema/mvc/spring-mvc-3.2.xsd">

    <!-- 配置 C3P0 数据源 -->
    <bean id="dataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource">
        <property name="user" value="root"></property>
        <property name="password" value="root"></property>
        <property name="driverClass" value="com.mysql.cj.jdbc.Driver">
</property>
        <property name="jdbcUrl" value="jdbc:mysql://localhost:3306/mybatis?
useUnicode=true&amp;characterEncoding=UTF-8"></property>
        <property name="initialPoolSize" value="5"></property>
        <property name="maxPoolSize" value="10"></property>
    </bean>

    <!-- 配置 MyBatis 的 SqlSessionFactory -->
    <bean id="sqlSessionFactory"
class="org.mybatis.spring.SqlSessionFactoryBean">
        <!-- 指定 MyBatis 数据源 -->
        <property name="dataSource" ref="dataSource"></property>
        <!-- 指定 MyBatis mapper 文件的位置 -->
        <property name="mapperLocations"
value="classpath:com/southwind/repository/*.xml"></property>
        <!-- MyBatis 全局配置文件的位置 -->
        <property name="configLocation" value="classpath:config.xml">
</property>
    </bean>

    <!-- 扫描 MyBatis 的 Mapper 接口 -->
    <bean class="org.mybatis.spring.mapper.MapperScannerConfigurer">
        <!-- 扫描所有的 Mapper 接口 -->
        <property name="basePackage" value="com.southwind.repository">
</property>
    </bean>

</beans>
```

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>
    <settings>
        <!-- 打印SQL-->
        <setting name="logImpl" value="STDOUT_LOGGING" />
    </settings>

    <typeAliases>
        <package name="com.southwind.entity"></package>
    </typeAliases>
</configuration>
```

springmvc.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:mvc="http://www.springframework.org/schema/mvc"
       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/mvc
       http://www.springframework.org/schema/mvc/spring-mvc-3.2.xsd">

    <!-- 启用 Spring MVC 的注解驱动 -->
    <mvc:annotation-driven></mvc:annotation-driven>

    <!-- 自动扫描 -->
    <context:component-scan base-package="com.southwind">
</context:component-scan>

    <!-- 视图解析器 -->
    <bean
class="org.springframework.web.servlet.view.InternalResourceViewResolver">
        <property name="prefix" value="/"></property>
        <property name="suffix" value=".jsp"></property>
    </bean>

</beans>
```