### JAVA第三阶段—DAY11-JAVA案例

1. SSM整合案例

通过该案例讲解SSM整合的配置过程

1、添加依赖

<dependency>  
          <groupId>mysql</groupId>  
          <artifactId>mysql-connector-java</artifactId>  
          <version>8.0.16</version>  
      </dependency>  
​  
      <!-- https://mvnrepository.com/artifact/com.mchange/c3p0 -->  
      <dependency>  
          <groupId>com.mchange</groupId>  
          <artifactId>c3p0</artifactId>  
          <version>0.9.5.2</version>  
      </dependency>  
​  
      <!-- https://mvnrepository.com/artifact/javax.servlet/javax.servlet-api -->  
      <dependency>  
          <groupId>javax.servlet</groupId>  
          <artifactId>javax.servlet-api</artifactId>  
          <version>3.1.0</version>  
          <scope>provided</scope>  
      </dependency>  
​  
      <dependency>  
          <groupId>org.apache.taglibs</groupId>  
          <artifactId>taglibs-standard-impl</artifactId>  
          <version>1.2.5</version>  
      </dependency>  
​  
      <dependency>  
          <groupId>org.apache.taglibs</groupId>  
          <artifactId>taglibs-standard-compat</artifactId>  
          <version>1.2.5</version>  
      </dependency>  
​  
      <dependency>  
          <groupId>org.apache.taglibs</groupId>  
          <artifactId>taglibs-standard-jstlel</artifactId>  
          <version>1.2.5</version>  
      </dependency>  
​  
      <dependency>  
          <groupId>org.apache.taglibs</groupId>  
          <artifactId>taglibs-standard-spec</artifactId>  
          <version>1.2.5</version>  
      </dependency>  
​  
      <!-- https://mvnrepository.com/artifact/org.springframework/spring-context -->  
      <dependency>  
          <groupId>org.springframework</groupId>  
          <artifactId>spring-context</artifactId>  
          <version>5.2.8.RELEASE</version>  
      </dependency>  
​  
      <!-- https://mvnrepository.com/artifact/org.springframework/spring-context -->  
      <dependency>  
          <groupId>org.springframework</groupId>  
          <artifactId>spring-webmvc</artifactId>  
          <version>5.2.8.RELEASE</version>  
      </dependency>  
​  
      <!-- https://mvnrepository.com/artifact/org.springframework/spring-context -->  
      <dependency>  
          <groupId>org.springframework</groupId>  
          <artifactId>spring-test</artifactId>  
          <version>5.2.8.RELEASE</version>  
      </dependency>  
​  
      <!-- https://mvnrepository.com/artifact/org.springframework/spring-context -->  
      <dependency>  
          <groupId>org.springframework</groupId>  
          <artifactId>spring-jdbc</artifactId>  
          <version>5.2.8.RELEASE</version>  
      </dependency>  
​  
      <!-- https://mvnrepository.com/artifact/org.mybatis/mybatis -->  
      <dependency>  
          <groupId>org.mybatis</groupId>  
          <artifactId>mybatis</artifactId>  
          <version>3.5.5</version>  
      </dependency>  
​  
      <dependency>  
          <groupId>org.mybatis</groupId>  
          <artifactId>mybatis-spring</artifactId>  
          <version>2.0.5</version>  
      </dependency>

2、SpringMVC的配置 spring-mvc.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 https://www.springframework.org/schema/context/spring-context.xsd http://www.springframework.org/schema/mvc https://www.springframework.org/schema/mvc/spring-mvc.xsd">  
  <!--配置视图处理器-->  
  <bean id="viewResolver" class="org.springframework.web.servlet.view.InternalResourceViewResolver">  
      <property name="prefix" value="/WEB-INF/pages/"></property>  
      <property name="suffix" value=".jsp"></property>  
  </bean>  
​  
  <mvc:default-servlet-handler/>  
  <mvc:annotation-driven/>  
</beans>

3、MyBatis基础设置 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>  
  <settings>  
      <!-- 下划线转驼峰命名 -->  
      <setting name="mapUnderscoreToCamelCase" value="true"/>  
      <!-- 打印查询语句 -->  
      <setting name="logImpl" value="STDOUT\_LOGGING" />  
      <!--配置懒加载-->  
      <setting name="lazyLoadingEnabled" value="true"/>  
      <!--执行所有属性按需加载-->  
      <setting name="aggressiveLazyLoading" value="false"/>  
  </settings>  
</configuration>

4、Spring整合MyBatis配置 spring-mybatis.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"  
      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">  
​  
  <!--配置C3P0数据源-->  
  <bean id="dataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource">  
      <!--jdbc的配置-->  
      <property name="driverClass" value="com.mysql.cj.jdbc.Driver"></property>  
      <property name="jdbcUrl" value="jdbc:mysql://localhost:3306/book\_db?useUnicode=true&amp;characterEncoding=UTF-8&amp;serverTimezone=UTC&amp;useSSL=false"></property>  
      <property name="user" value="root"></property>  
      <property name="password" value="123456"></property>  
      <!--非必须-->  
      <property name="maxPoolSize" value="100"></property>  
      <property name="minPoolSize" value="10"></property>  
      <property name="initialPoolSize" value="10"></property>  
      <property name="maxStatements" value="200"></property>  
  </bean>  
  <!--配置会话工厂-->  
  <bean id="sqlSessionFactory" class="org.mybatis.spring.SqlSessionFactoryBean">  
      <!--配置数据源-->  
      <property name="dataSource" ref="dataSource"></property>  
      <!--包的别名-->  
      <property name="typeAliasesPackage" value="com.blb.bookms.entity"></property>  
      <!--MyBatis配置文件路径-->  
      <property name="configLocation" value="classpath:mybatis-config.xml"/>  
      <!--映射文件的路径-->  
      <property name="mapperLocations" value="classpath:mappers/\*.xml"></property>  
  </bean>  
  <!--配置映射接口的扫描器-->  
  <bean class="org.mybatis.spring.mapper.MapperScannerConfigurer">  
      <!--会话工厂名称-->  
      <property name="sqlSessionFactoryBeanName" value="sqlSessionFactory"></property>  
      <!--接口的包位置-->  
      <property name="basePackage" value="com.blb.bookms.dao"></property>  
  </bean>  
</beans>

5、Spring配置文件，导入SpringMVC和MyBatis配置

<?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 https://www.springframework.org/schema/context/spring-context.xsd">  
​  
  <context:component-scan base-package="com.blb.bookms"></context:component-scan>  
​  
  <import resource="spring-mvc.xml"></import>  
  <import resource="spring-mybatis.xml"></import>  
</beans>

6、web.xml

<?xml version="1.0" encoding="UTF-8"?>  
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
        xmlns="http://xmlns.jcp.org/xml/ns/javaee"  
        xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app\_3\_1.xsd"  
        id="WebApp\_ID" version="3.1">  
<display-name>Archetype Created Web Application</display-name>  
   
<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:spring.xml</param-value>  
  </init-param>  
  <load-on-startup>1</load-on-startup>  
</servlet>  
<servlet-mapping>  
  <servlet-name>dispatcherServlet</servlet-name>  
  <url-pattern>/</url-pattern>  
</servlet-mapping>  
​  
<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>  
</web-app>

编写实体类

package com.blb.bookms.entity;  
​  
/\*\*  
\* 书籍  
\*/  
public class Book {  
​  
  private Integer id;  
  private String bookName;  
  private float price;  
  private Integer typeId;  
  private String author;  
  private String publishOrg;  
  private String publishTime;  
  private Integer state;  
  private String bookImage;  
  ...  
}

3、编写DAO接口

package com.blb.bookms.dao;  
​  
public interface IBookDAO {  
​  
  List<Book> selectAllBooks();  
}

4、编写Service接口

package com.blb.bookms.service;  
​  
public interface IBookService {  
​  
  List<Book> findAllBooks();  
}

5、编写Service实现类

package com.blb.bookms.service.impl;  
​  
@Service  
public class BookServiceImpl implements IBookService {  
​  
  @Autowired  
  private IBookDAO bookDAO;  
​  
  @Override  
  public List<Book> findAllBooks() {  
      return bookDAO.selectAllBooks();  
  }  
}

6、编写控制器

package com.blb.bookms.controller;  
​  
/\*\*  
\* 书籍控制器  
\*/  
@Controller  
@RequestMapping("/book")  
public class BookController {  
​  
  @Autowired  
  private IBookService bookService;  
​  
  @GetMapping("/findAllBooks")  
  public String findAllBooks(Model model){  
      List<Book> books = bookService.findAllBooks();  
      model.addAttribute("books",books);  
      return "book";  
  }  
}

7、编写webapp/WEB-INF/pages/book.jsp页面

<%@ page contentType="text/html;charset=UTF-8" language="java" isELIgnored="false" %>  
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<html>  
<head>  
  <title>图书管理</title>  
  <link rel="stylesheet" href="/layui/css/layui.css">  
</head>  
<body>  
  <div class="layui-container">  
      <div class="layui-row">  
          <div class="layui-col-md9">  
              <table class="layui-table" lay-size="sm">  
                  <thead>  
                  <tr>  
                      <th>编号</th>  
                      <th>书名</th>  
                      <th>价格</th>  
                      <th>类型</th>  
                      <th>作者</th>  
                      <th>出版社</th>  
                      <th>出版日期</th>  
                      <th>状态</th>  
                      <th>图片</th>  
                      <th>操作</th>  
                  </tr>  
                  </thead>  
                  <tbody>  
                  <c:forEach var="book" items="${books}">  
                      <tr>  
                          <td>${book.id}</td>  
                          <td>${book.bookName}</td>  
                          <td>${book.price}</td>  
                          <td>${book.typeId}</td>  
                          <td>${book.author}</td>  
                          <td>${book.publishOrg}</td>  
                          <td>${book.publishTime}</td>  
                          <td>${book.state}</td>  
                          <td>  
                              <img src="/images/${book.bookImage}">  
                          </td>  
                      </tr>  
                  </c:forEach>  
                  </tbody>  
              </table>  
              <div id="page"></div>  
          </div>  
      </div>  
  </div>  
  </body>  
</html>

1. 事务案例

用转账案例介绍事务

账户表：id、username、balance

1. SSM整合完成对余额的修改
2. Service层转账方法不加事务，结果出现了数据不一致的情况

Dao层更新转出账户余额

抛出异常

Dao层更新转入账户余额

3）添加事务

事务的配置

<!--配置事务管理器-->

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

<!--配置数据源-->

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

</bean>

<!--配置启动声明式事务-->

<tx:annotation-driven transaction-manager="transactionManager"/>

方法上添加@Transactional注解，出现异常后发生回滚，没有异常才能提交