实验12 Spring Boot实验

**【实验目的及要求】**

1. 创建和配置Spring Boot项目；
2. 在项目中使用Thymeleaf模板引擎；
3. 整合Spring Boot和MyBatis框架；
4. 要求所有回答的文本格式：**五号，宋体、1.5倍行距，保留段单元格背景。**

**【实验步骤】**

1. **Spring Boot和MyBatis框架整合（学生信息管理模块）**

（1）在IDEA中新建Spring Boot项目gms\_ex（Grade Management System）。

（2）在项目中添加依赖，内容如下。

|  |
| --- |
| <dependencies>  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-thymeleaf</artifactId>  </dependency>  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-web</artifactId>  </dependency>  <dependency>  <groupId>org.mybatis.spring.boot</groupId>  <artifactId>mybatis-spring-boot-starter</artifactId>  <version>2.3.0</version>  </dependency>  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-devtools</artifactId>  <scope>runtime</scope>  <optional>true</optional>  </dependency>  <dependency>  <groupId>com.mysql</groupId>  <artifactId>mysql-connector-j</artifactId>  <scope>runtime</scope>  </dependency>  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-test</artifactId>  <scope>test</scope>  </dependency>  <dependency>  <groupId>com.github.pagehelper</groupId>  <artifactId>pagehelper-spring-boot-starter</artifactId>  <version>1.4.3</version>  </dependency> </dependencies> |

（3）在applicaiton.properties全局配置文件中，添加如下配置。

|  |
| --- |
| spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver spring.datasource.url=jdbc:mysql://localhost:3306/student?useUnicode=true&userSSL=false&characterEncoding=utf8 spring.datasource.username=root spring.datasource.password=root mybatis.mapper-locations=classpath:mapper/\*.xml mybatis.type-aliases-package=com.javaee.gms\_ex.po mybatis.configuration.map-underscore-to-camel-case=true  pagehelper.helperDialect=mysql pagehelper.reasonable=true |

（4）配置项目的热部署。

（5）在main/java中，创建配置类com.javaee.gms\_ex.config.WebMvcConfig,代码如下。

|  |
| --- |
| package com.javaee.gms\_ex.config; import org.springframework.context.annotation.Configuration; import org.springframework.web.servlet.config.annotation.ResourceHandlerRegistry; import org.springframework.web.servlet.config.annotation.WebMvcConfigurer;  @Configuration public class WebMvcConfig implements WebMvcConfigurer {  public void addResourceHandlers(  ResourceHandlerRegistry registry) {  registry.addResourceHandler("/js/\*\*").  addResourceLocations("classpath:/static/js/");  } } |

（6）在main/resources/static/js中，添加jQuary库文件jquary.min.js。

（7）在main/java中，创建实体类com.javaee.gms\_ex.po.Student，代码如下。

|  |
| --- |
| package com.javaee.gms\_ex.po; public class Student {  private String sno;  private String sname;  private String ssex;  private String snative;  private int mno;  public String getSno() {  return sno;  }  public void setSno(String sno) {  this.sno = sno;  }  public String getSname() {  return sname;  }  public void setSname(String sname) {  this.sname = sname;  }  public String getSsex() {  return ssex;  }  public void setSsex(String ssex) {  this.ssex = ssex;  }  public String getSnative() {  return snative;  }  public void setSnative(String snative) {  this.snative = snative;  }   public int getMno() {  return mno;  }  public void setMno(int mno) {  this.mno = mno;  } } |

（8）在main/java中，创建com.javaee.gms\_ex.mapper.StudentMapper映射器接口，代码如下。

|  |
| --- |
| package com.javaee.gms\_ex.mapper; import com.javaee.gms\_ex.po.Student; import org.apache.ibatis.annotations.Mapper; import java.util.List; @Mapper public interface StudentMapper {  List<Student> findStudents();  Student findStudent(String sno);  int addStudent(Student student);  int deleteStudent(String sno);  int updateStudent(Student student); } |

（9）在main/resources/mapper中，创建映射文件StudentMapper.xml，内容如下。

|  |
| --- |
| *<?*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.javaee.gms\_ex.mapper.StudentMapper">  <select id="findStudents" resultType="student">  select *\** from stu  </select>  <select id="findStudent" parameterType="String" resultType="Student" >  select *\** from stu where sno=#{sno};  </select>  <insert id="addStudent" parameterType="Student">  insert into stu(sno,sname,ssex,snative,mno) values(#{sno},#{sname},#{ssex},#{snative},#{mno});  </insert>  <update id="updateStudent" parameterType="Student">  update stu set sname=#{sname},ssex=#{ssex},snative=#{snative},mno=#{mno} where sno=#{sno};  </update>  <delete id="deleteStudent" parameterType="String">  delete from stu where sno=#{sno};  </delete> </mapper> |

（10）在main/java中，添加com.javaee.gms\_ex.service.StudentService业务层接口，代码如下。

|  |
| --- |
| package com.javaee.gms\_ex.service;  import com.javaee.gms\_ex.po.Student;  import java.util.List;  public interface StudentService {  List<Student> findStudents();  Student findStudent(String sno);  int addStudent(Student student);  int deleteStudent(String sno);  int updateStudent(Student student); } |

（11）在main/java中，业务层接口的实现类com.javaee.gms\_ex.service.impl. StudentServiceImpl，代码如下。

|  |
| --- |
| package com.javaee.gms\_ex.service.impl;  import com.javaee.gms\_ex.mapper.StudentMapper; import com.javaee.gms\_ex.po.Student; import com.javaee.gms\_ex.service.StudentService; import org.springframework.beans.factory.annotation.Autowired; import org.springframework.stereotype.Service;  import java.util.List;  @Service public class StudentServiceImpl implements StudentService {  @Autowired  private StudentMapper studentMapper;  @Override  public List<Student> findStudents() {  return studentMapper.findStudents();  }   @Override  public Student findStudent(String sno) {  return studentMapper.findStudent(sno);  }   @Override  public int addStudent(Student student) {  return studentMapper.addStudent(student);  }   @Override  public int deleteStudent(String sno) {  return studentMapper.deleteStudent(sno);  }   @Override  public int updateStudent(Student student) {  return studentMapper.updateStudent(student);  } } |

（12）在main/java中，创建com.javaee.gms\_ex.controller.StudentController控制器，代码如下。

|  |
| --- |
| package com.javaee.gms\_ex.controller;  import com.github.pagehelper.PageInfo; import com.javaee.gms\_ex.po.Student; import com.javaee.gms\_ex.service.StudentService; 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 org.springframework.web.bind.annotation.ResponseBody;  import java.util.List;  @Controller public class StudentController {  @Autowired  public StudentService studentService;   @RequestMapping("/student\_list")  public String studentList(Model model){  List<Student> students = studentService.findStudents();  PageInfo<Student> pageInfo = new PageInfo<>(students);  model.addAttribute("pageInfo", pageInfo);  return "student/studentList";  }  @RequestMapping("/to\_add\_student")  public String toAddStudent(Model model){  model.addAttribute("studentOper", "0");  return "student/add\_student";  }  @RequestMapping("/add\_student")  public String addStudent(Student student, Model model){  studentService.addStudent(student);  return "redirect:/student\_list";  }  @RequestMapping("/to\_edit\_student")  public String toEditStudent(String sno, Model model){  Student student = studentService.findStudent(sno);  model.addAttribute("student", student);  model.addAttribute("studentOper", "1");  return "student/add\_student";  }  @RequestMapping("/edit\_student")  public String editStudent(Student student, Model model){  studentService.updateStudent(student);  return "redirect:/student\_list";  }  @RequestMapping("/delete\_student")  @ResponseBody  public String deleteStudent(String sno){  studentService.deleteStudent(sno);  return "SUCCESS";  } } |

（13）在main/resources/templates目录中添加首页index.html，提供进入学生信息管理的超链接，内容如下。

|  |
| --- |
| <!DOCTYPE html> <html lang="en"> <head>  <meta charset="UTF-8">  <title>成绩管理系统（GMS）</title> </head> <body> <a href="student\_list">学生信息管理</a>  </body> </html> |

（14）在main/resources/templates/student目录下，新建studentlist.html文件，提供学生信息的列表显示，以及学生信息的添加、删除和修改操作，内容如下。

|  |
| --- |
| <!DOCTYPE html> <html lang="en" xmlns:th="http://www.thymeleaf.org"> <head>  <meta charset="UTF-8">  <title>学生信息管理</title> </head> <body> <h3>学生信息管理</h3> <table width="100%" border="1" cellspacing="0" cellpadding="0" style="border-collapse: collapse;border-color: #238FE7">  <tbody><tr><td>学号</td><td>姓名</td><td>性别</td><td>籍贯</td><td>专业号</td><td>操作</td></tr>  <tr th:each="student:${pageInfo.list}" th:id="${student.sno}">  <td th:text="${student.sno}"></td>  <td th:text="${student.sname}"></td>  <td th:text="${student.ssex}"></td>  <td th:text="${student.snative}"></td>  <td th:text="${student.mno}"></td>  <td>  <div>  <a th:href="@{to\_edit\_student(sno=${student.sno})}">修改</a>  <a href="#" th:onclick="deleteStudent([[${student.sno}]]);">删除</a>  </div>  </td>  </tr>  </tbody></table> <p>当前 <span th:text="${pageInfo.pageNum}"></span> 页,总<span th:text="${pageInfo.pages}"></span> 页,共<span th:text="${pageInfo.total}"></span> 条记录</p> <a th:href="@{publisher\_list\_page}">首页</a> <a th:href="@{publisher\_list\_page(pageNum=${pageInfo.hasPreviousPage}?${pageInfo.prePage}:1)}">上一页</a> <a th:href="@{publisher\_list\_page(pageNum=${pageInfo.hasNextPage}?${pageInfo.nextPage}:${pageInfo.pages})}">下一页</a> <a th:href="@{publisher\_list\_page(pageNum=${pageInfo.pages})}">尾页</a> <br>*<!--<button onclick="addStudent()">添加</button>-->*<a th:href="@{to\_add\_student}">添加</a>  <script th:src="@{/js/jquery/jquery.min.js}"></script> <script>  function *deleteStudent*(sno) {  if (!*confirm*("确定要删除吗？"))  return;  $.post("/delete\_student",{"sno":sno},  function(data){  if(data == "SUCCESS"){  $("#" + sno).remove();  }else{  *alert*("删除学生失败！");  }  });  } </script> </body> </html> |

（15）在main/resources/templates/student目录下，新建add\_student.html文件，提供学生信息的添加和修改表单，内容如下。

|  |
| --- |
| <!DOCTYPE html> <html lang="en" xmlns:th="http://www.thymeleaf.org"> <head>  <meta charset="UTF-8">  <title th:text="${studentOper=='1'}?'编辑学生信息':'添加学生信息'"></title> </head> <body> <form th:action="${studentOper=='1'}?'/edit\_student':'/add\_student'">  学号：<span th:if="${studentOper=='0'}"><input type="text" name="sno" placeholder="学号"></span>  <span th:if="${studentOper=='1'}"><span th:text="${student.sno}"></span> <input type="hidden" name="sno" th:value="${student.sno}"></span><br>  姓名：<input type="text" name="sname" placeholder="姓名" th:value="${studentOper=='1'}?${student.sname}:''"><br>  性别：<select name="ssex" id="select">  <option value="男" selected>男</option>  <option value="女">女</option>  </select><br>  籍贯：<input type="text" name="snative" placeholder="籍贯" th:value="${studentOper=='1'}?${student.snative}:''"><br>  专业编号：<input type="number" name="mno" placeholder="专业编号" th:value="${studentOper=='1'}?${student.mno}:''"><br>  <input type="submit" th:value="${studentOper=='1'}?'修改':'添加'"> </form> </body> </html> |

（16）测试程序，确保gms\_ex程序能够正常运行。

1. **实现课程信息管理模块**

（1）模仿上述过程，在gms\_ex项目中，添加课程信息管理模块，主要内容如下。

|  |
| --- |
| courselist.html文件内容： |
| add\_course.jsp文件内容： |
| 实体类Course代码： |
| 业务层接口CourseService代码： |
| 接口CourseService的实现类CourseServiceImpl代码： |
| 映射文件CourseMapper.xml内容： |
| 接口CourseMapper代码： |
| 控制器CourseController代码： |

（2）在项目中添加捕捉处理器异常的全局异常处理类，要求能够处理ArithmeticException和Exception类型的异常。

|  |
| --- |
| // 全局异常处理类代码 |

（3）根据上述实验步骤，归纳总结在Spring Boot+MyBatis框架中，实现Web系统各模块功能的基本思路。

|  |
| --- |
|  |