# 实现用户登录

**【案例目标】**

通过所学的Spring数据库编程知识，实现学生管理系统的登录功能。本案例要求学生在控制台输入用户名密码，如果用户账号密码正确则显示用户所属班级，如果登录失败则显示登录失败。

实现用户登录项目运行成功后控制台效果如图9-1。



1. 实现用户登录项目运行成功后控制台效果图

**【思路分析】**

根据学生管理系统及其登录要求，可以分析案例的实现步骤如下。

1. 为了存储学生信息，需要创建一个数据库。
2. 为了程序连接数据库并完成对数据的增删改查操作，需要在XML配置文件中配置数据库连接和事务等信息。
3. 在Dao层实现查询用户信息的方法。
4. 在Controller层处理业务逻辑，如判断用户输入的用户名与密码是否正确。

**【实现步骤】**

**1.创建数据库**

在MySQL中的spring数据库中创建一个名为student的表，student表的结构如表9-1所示。

1. student表的结构

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **类型** | **长度** | **是否主键** | **说明** |
| id | int | 11 | 是 | 学生编号 |
| username | varchar | 255 | 否 | 学生姓名 |
| password | varchar | 255 | 否 | 学生密码 |
| course | varchar | 255 | 否 | 学生班级 |

**2.编写实体类**

在chapter09项目的com.itheima包中创建entity包，在该包下创建Student类，在该类中定义id、username、password和course属性，以及属性对应的getter/setter方法。Student类具体实现如文件9-1所示。

1. Student.java
2. package com.itheima.entity;
3. public class Student {
4. //学生ID
5. private Integer id;
6. //学生姓名
7. private String username;
8. //学生密码
9. private String password;
10. //学生班级
11. private String course;
12. public Integer getId() {
13. return id;
14. }
15. public void setId(Integer id) {
16. this.id = id;
17. }
18. public String getUsername() {
19. return username;
20. }
21. public void setUsername(String username) {
22. this.username = username;
23. }
24. public String getPassword() {
25. return password;
26. }
27. public void setPassword(String password) {
28. this.password = password;
29. }
30. public String getCourse() {
31. return course;
32. }
33. public void setCourse(String course) {
34. this.course = course;
35. }
36. }

**3.编写配置文件**

在chapter09项目的resources文件夹下，创建配置文件applicationContext-student.xml，在该文件中配置id为dataSource的数据源Bean和id为jdbcTemplate的JDBC模板Bean，并将数据源注入到JDBC模板中。applicationContext-student.xml配置如文件9-2所示。

1. applicationContext-student.xml
2. <?xml version="1.0" encoding="UTF-8"?>
3. <beans xmlns="http://www.springframework.org/schema/beans"
4. xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5. xmlns:tx="http://www.springframework.org/schema/tx"
6. xsi:schemaLocation="http://www.springframework.org/schema/beans
7. http://www.springframework.org/schema/beans/spring-beans-4.3.xsd
8. http://www.springframework.org/schema/tx
9. http://www.springframework.org/schema/tx/spring-tx-4.3.xsd">
10. <!-- 1.配置数据源 -->
11. <bean id="dataSource"
12. class="org.springframework.jdbc.datasource.DriverManagerDataSource">
13. <!--数据库驱动 -->
14. <property name="driverClassName" value="com.mysql.jdbc.Driver" />
15. <!--连接数据库的url -->
16. <property name="url"
17. value="jdbc:mysql://localhost/spring?useSSL=false" />
18. <!--连接数据库的用户名 -->
19. <property name="username" value="root" />
20. <!--连接数据库的密码 -->
21. <property name="password" value="root" />
22. </bean>
23. <!-- 2.配置JDBC模板 -->
24. <bean id="jdbcTemplate"
25. class="org.springframework.jdbc.core.JdbcTemplate">
26. <!-- 默认必须使用数据源 -->
27. <property name="dataSource" ref="dataSource" />
28. </bean>
29. <!-- 3.定义id为accountDao的Bean -->
30. <bean id="studentDao" class="com.itheima.dao.impl.StudentDaoImpl">
31. <!-- 将jdbcTemplate注入到AccountDao实例中 -->
32. <property name="jdbcTemplate" ref="jdbcTemplate" />
33. </bean>
34. <!-- 4.事务管理器，依赖于数据源 -->
35. <bean id="transactionManager" class=
36. "org.springframework.jdbc.datasource.DataSourceTransactionManager">
37. <property name="dataSource" ref="dataSource" />
38. </bean>
39. <!-- 5.注册事务管理器驱动 -->
40. <tx:annotation-driven transaction-manager="transactionManager"/>
41. </beans>

**4.编写Dao层方法**

在com.itheima包下创建dao包，在该包下创建StudentDao接口，在StudentDao接口中声明查询所有用户信息的方法。StudentDao接口具体代码如文件9-3所示。

1. StudentDao.java
2. package com.itheima.dao;
3. import com.itheima.entity.Student;
4. import java.util.List;
5. public interface StudentDao {
6. //查询所有账户
7. public List<Student> findAllStudent();
8. }

**5.实现Dao层方法**

在com.itheima.dao包下创建Impl包，在该包下创建StudentDaoImpl实现类，在StudentDaoImpl类中实现StudentDao接口中的findAllStudent()方法。StudentDaoImpl类代码如文件9-4所示。

1. StudentDaoImpl.java
2. package com.itheima.dao.impl;
3. import com.itheima.dao.StudentDao;
4. import com.itheima.entity.Student;
5. import org.springframework.jdbc.core.BeanPropertyRowMapper;
6. import org.springframework.jdbc.core.JdbcTemplate;
7. import org.springframework.jdbc.core.RowMapper;
8. import java.util.List;
9. public class StudentDaoImpl implements StudentDao {
10. // 声明JdbcTemplate属性及其setter方法
11. private JdbcTemplate jdbcTemplate;
12. public void setJdbcTemplate(JdbcTemplate jdbcTemplate) {
13. this.jdbcTemplate = jdbcTemplate;
14. }
15. // 查询所有用户信息
16. public List<Student> findAllStudent() {
17. // 定义SQL语句
18. String sql = "select \* from student";
19. // 创建一个新的BeanPropertyRowMapper对象
20. RowMapper<Student> rowMapper =
21. new BeanPropertyRowMapper<Student>(Student.class);
22. // 执行静态的SQL查询，并通过RowMapper返回结果
23. return this.jdbcTemplate.query(sql, rowMapper);
24. }
25. }

在文件9-4代码中，第10~13行代码声明JdbcTemplate属性及其setter方法，第15~23行代码实现了findAllStudent()方法，可以查询所有用户信息。

**6.编写Controller层**

在com.itheima包中创建controller包，在controller包中创建StudentController类，用于实现用户登录操作。StudentController类具体代码如文件9-5所示。

1. StudentController.java
2. package com.itheima.controller;
3. import com.itheima.dao.StudentDao;
4. import com.itheima.entity.Student;
5. import org.springframework.context.ApplicationContext;
6. import
7. org.springframework.context.support.ClassPathXmlApplicationContext;
8. import java.util.List;
9. import java.util.Scanner;
10. public class StudentController {
11. public static void main(String[] args) {
12. System.out.println("欢迎来到学生管理系统");
13. System.out.println("请输入用户名：");
14. Scanner sca=new Scanner(System.in);
15. String name=sca.nextLine();
16. // 加载配置文件
17. ApplicationContext applicationContext =new
18. ClassPathXmlApplicationContext("applicationContext-student.xml");
19. // 获取AccountDao实例
20. StudentDao studentDao =
21. (StudentDao) applicationContext.getBean("studentDao");
22. // 执行findAllAccount()方法,获取Account对象的集合
23. List<Student> student = studentDao.findAllStudent();
24. // 循环输出集合中的对象
25. for (Student stu : student) {
26. if(name.equals(stu.getUsername())){
27. System.out.println("请输入"+stu.getUsername()+"的密码：");
28. String mima= sca.nextLine();
29. if(mima.equals(stu.getPassword())){
30. System.out.println("用户登录成功！");
31. System.out.println(stu.getUsername()+"是
32. "+stu.getCourse()+"班的");
33. return;
34. }
35. }else {
36. System.out.println("账号密码错误！");
37. return;
38. }
39. }
40. }
41. }

至此用户登录的功能就已全部实现，在IDEA中启动StudentController类，在控制台按照提示输入账号密码进行登录。控制台效果如图9-2所示。



1. 控制台效果