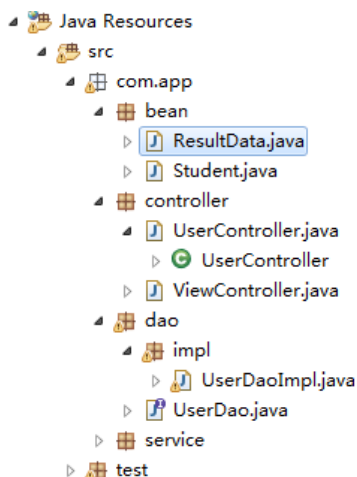


【Java框架型项目从入门到装逼】第十三节 用户新增功能完结篇

这一节，我们把用户新增的功能继续做一个完善。首先，新增成功后，需要给前台返回一个信息，就是告诉浏览器，这次用户新增的操作到底是成功了，还是失败了呢？为此，我们需要专门引入一个结果类，里面只有两个属性，分别为错误码和错误信息，这个类在之前的章节中有提到过。



```
package com.app.bean;

public class ResultData {

    private int errCode = 0;
    private String errMsg;

    public int getErrCode() {
        return errCode;
    }
    public void setErrCode(int errCode) {
        this.errCode = errCode;
    }
    public String getErrMsg() {
        return errMsg;
    }
    public void setErrMsg(String errMsg) {
        this.errMsg = errMsg;
    }
}
```

然后，改写UserController类：

```
package com.app.controller;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.ResponseBody;

import com.app.bean.ResultData;
import com.app.bean.Student;
import com.app.service.UserService;
import com.app.service.impl.UserServiceImpl;

@Controller
public class UserController {

    //用户业务类的引用
    private UserService userService = new UserServiceImpl();

    @RequestMapping("/addUser")
    @ResponseBody
    public ResultData addUser(HttpServletRequest request , HttpServletResponse response, Student student) {
        ResultData data = new ResultData();

        try{
            userService.addUser(student);
        }catch(Exception e){
            data.setErrCode(-1);
            data.setErrMsg(e.getMessage());
        }

        return data;
    }
}
```

```
}  
}
```

如果保存操作出现异常，我们就捕获一下异常，并且记录下异常信息，返回给浏览器。注意，这边我们还给addUser方法加了一个@ResponseBody注解。这样一来，当我们return数据的时候，就会自动转换成json对象，然后用IO流的方式写出到浏览器。

后台控制器解决了之后，我们再来修改前台的ajax方法：

```
//使用ajax传递到后台  
$.post("addUser.do", json, function(data) {  
    //这里是处理返回数据的回调函数  
  
    if(data.errCode < 0){  
        alert('操作发生错误，原因是：' + data.errMsg);  
    }else{  
        alert('保存成功');  
    }  
  
}, "json");
```

效果：




新增用户

用户名:

密码:

姓名:

性别:



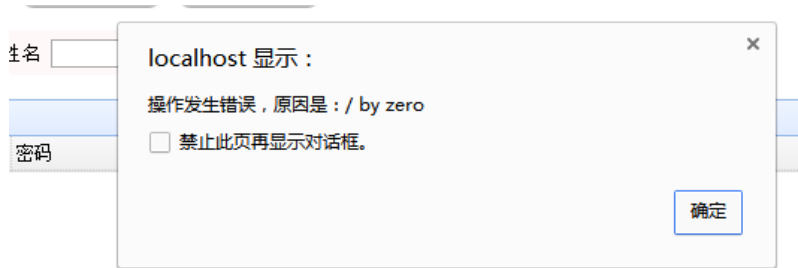
localhost 显示：

保存成功

现在，我们故意制造一个异常：

```
@Controller  
public class UserController {  
  
    //用户业务类的引用  
    private UserService userService = new UserServiceImpl();  
  
    @RequestMapping("/addUser")  
    @ResponseBody  
    public ResultData addUser(HttpServletRequest request , HttpServletResponse response, Student student) {  
        ResultData data = new ResultData();  
  
        try{  
            userService.addUser(student);  
            int i = 1 / 0; //故意写一句错误代码  
        }catch(Exception e){  
            data.setErrCode(-1);  
            data.setErrMsg(e.getMessage());  
        }  
  
        return data;  
    }  
}
```

结果



新增用户

用户名:

a

密码:

1

姓名:

2

性别:

男

保存

这样一来，浏览器就能清楚地知道后台报了什么错误了。

但是这样有一个问题，虽然后台报错了，但是数据依然进了数据库。这里我们就需要规定，所有的业务操作不应该放在controller类中，都应该放到service类中。而且，这边还涉及到一个事务回滚的问题。这些知识点会在以后讲到。

现在，我们来看一下数据库：

id	username	password	name	sex
2018012101	zsf	123	张三丰	男
2018012103	gj	123456	郭靖	男
2018012104	aaa	123	啊啊啊	男
2018012107	pikaqiu	123	皮卡丘	男
2018012108	a	1	2	男
2018012109	a	1	2	男

发现一个问题，两条数据的用户名是重复的，在实际情况下，用户名和ID一样，是唯一的。所以，我们需要判断一下用户名是否重复？

```
package com.app.controller;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.ResponseBody;

import com.app.bean.ResultData;
import com.app.bean.Student;
import com.app.service.UserService;
import com.app.service.impl.UserServiceImpl;

@Controller
public class UserController {

    //用户业务类的引用
    private UserService userService = new UserServiceImpl();

    @RequestMapping("/addUser")
    @ResponseBody
    public ResultData addUser(HttpServletRequest request , HttpServletResponse response,Student student){
        ResultData data = new ResultData();

        try{

            //先判断用户名是否重复

            int count = userService.getByUserName(student);

            if(count > 0){
```

```

        data.setErrCode(-1);
        data.setErrMsg("用户名已经存在啦，换一个吧。。。");
    }

    userService.addUser(student);
} catch (Exception e) {
    data.setErrCode(-1);
    data.setErrMsg(e.getMessage());
}

return data;

}

}

```

UserServiceImpl

```

package com.app.service.impl;

import com.app.bean.Student;
import com.app.dao.UserDao;
import com.app.dao.impl.UserDaoImpl;
import com.app.service.UserService;

public class UserServiceImpl implements UserService{

    private UserDao userDao = new UserDaoImpl();

    @Override
    public void addUser(Student student) {

        userDao.addUser(student);

    }

    @Override
    public int getByUserName(Student student) {

        return userDao.getByUserName(student);

    }

}

```

UserDaoImpl

```

package com.app.dao.impl;

import java.util.HashMap;
import java.util.Map;

import com.app.bean.Student;
import com.app.dao.UserDao;
import com.simple.dao.SimpleDao;

public class UserDaoImpl implements UserDao {

    @Override
    public void addUser(Student student) {

        SimpleDao dao = new SimpleDao();

        Map map = new HashMap();

        map.put("id", null);
        map.put("username", student.getUsername());
        map.put("password", student.getPassword());
        map.put("name", student.getName());
        map.put("sex", student.getSex());

        dao.save("db_student", "t_student", map);

    }

    @Override
    public int getByUserName(Student student) {

        String sql = "select count(1) from t_student where username = ?";

        SimpleDao dao = new SimpleDao();
        return dao.queryForInt(sql, student.getUsername());

    }

}

```

效果:

新增用户

用户名: a

密码: 1

姓名: 2

性别: 男

保存

localhost 显示:

操作发生错误, 原因是: 用户名已经存在啦, 换一个吧...

☐ 禁止此页再显示对话框。

确定

```
Object {username: "a", password: "1", name: "2",
```

在保存操作之前, 我们先判断一下用户名在表里是否存在, 只要存在了, 就给他返回一个错误。OK, 让我们看一下数据库:

id	username	password	name	sex
2018012101	zsf	123	张三丰	男
2018012103	gj	123456	郭靖	男
2018012104	aaa	123	啊啊啊	男
2018012107	pikaqiu	123	皮卡丘	男
2018012108	a	1	2	男
2018012109	a	1	2	男
2018012110	a	1	2	男
2018012111	a	1	2	男



我去, 怎么还是保存进去啦, 喵喵喵?

让我们再回过去看一下controller的方法:

```
@Controller
public class UserController {

    //用户业务类的引用
    private UserService userService = new UserServiceImpl();

    @RequestMapping("/addUser")
    @ResponseBody
    public ResultData addUser(HttpServletRequest request, HttpServletResponse response, Student student) {
        ResultData data = new ResultData();

        try{

            //先判断用户名是否重复

            int count = userService.getByUserName(student);

            if(count > 0){
                data.setErrCode(-1);
                data.setErrMsg("用户名已经存在啦, 换一个吧...");
            }

            userService.addUser(student);
        }catch(Exception e){
            data.setErrCode(-1);
            data.setErrMsg(e.getMessage());
        }

        return data;
    }
}
```

发现问题了, 我们只是给data对象设置了错误码和错误信息, 但是保存方法依然执行了, 所以, 我们需要及时return:

```
if(count > 0){  
    data.setErrCode(-1);  
    data.setErrMsg("用户名已经存在啦，换一个吧。。。");  
    return data;  
}
```

再来一次，就好了。

[我要下载源码](#)