POI 和 JXL 使用手册

内部文件：[2017000001]

颁布时间：[2017-02-18]

目　　录

文件版本说明

表 1 版本说明

版本	发布时间	修订章节	作者
V1	2017-02-11	初稿	高志远

声明

1. POI 和 JXL 基础知识

1.1 技术简介

Apache POI 是 Apache 软件基金会的开放源码函式库，POI 提供 API 给 Java 程序对 Microsoft Office 格式档案读和写的功能。

JXL 是一个开源的 Java Excel API 项目，通过 JXL，Java 可以很方便的操作微软的 Excel 文档。

1.2 技术特点

	特点	总结
POI	1. 效率高 2. 支持公式，宏，一些企业应用上会非常实用 3. 能够修饰单元格属性 4. 支持字体、数字、日期操作 5. API 丰富，支持多种模式的读写 6. 支持大数量大文件的读写操作	对于简单的单表 Excel 导入导出的需求，建议使用 JXL。数据量稍微小点，占用内存少，速度快。 对于报表类的，涉及月份数据量，多表数据聚合在一起建议使用 POI。
JXL	1. Jxl 对中文支持非常好，操作简单，方法看名知意。 2. Jxl 是纯 javaAPI，在跨平台上表现的非常完美，小文件读取效率比较高。 3. 支持字体、数字、日期操作 4. 能够修饰单元格属性 5. 支持图像和图表，但是这套 API 对图形和图表的支持很有限，而且仅仅识别 PNG 格式。	

1.3 POI 和 JXL 对 Excel 抽象出来的对象对比

	POI	JXL
Excel 文档	HSSFWorkbook	Workbook
Excel 的工作表	HSSFSheet	Sheet

Excel 的行	HSSFRow	无
Excel 中的单元格	HSSFCell	Cell
Excel 字体	HSSFFont	
Excel 单元格样式	HSSFCellStyle	
Excel 颜色	HSSFColor	
合并单元格	CellRangeAddress	

2. POI 和 JXL 基本操作

2.1 POI 导入导出 Excel

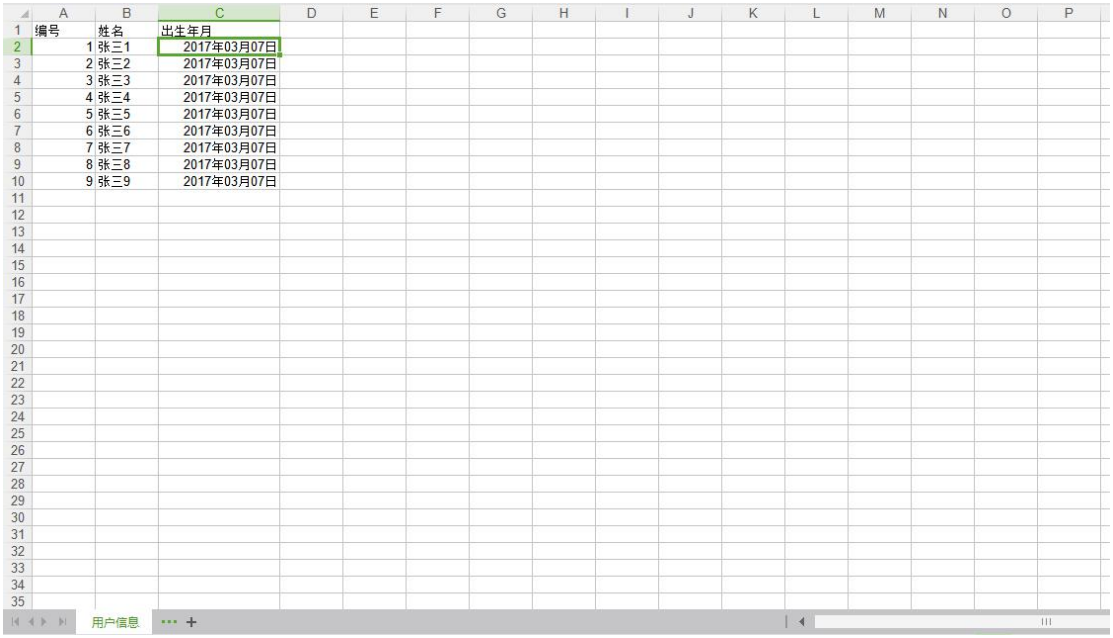
1) POI 导出

导出代码：

```
/**
     *
     * @author gaozhy
     * @2017 年 2 月 15 日下午 9:02:48
     * @version 1.0
     * @description poi 导出 Excel
     */
    @Test
    public void exportUser(){
        //创建 Excel 工作薄对象
        HSSFWorkbook workbook = new HSSFWorkbook();
        //创建工作表
        HSSFSheet sheet = workbook.createSheet("用户信息");
        //创建标题行
        HSSFRow row = sheet.createRow(0);
        String[] title = {"编号","姓名","出生年月"};

        //创建单元格对象
        HSSFCell cell = null;
        for (int i = 0; i < title.length; i++) {
             //i 标示列索引
            cell = row.createCell(i);
            cell.setCellValue(title[i]);
        }
```

```
        //处理日期格式
        HSSFCellStyle cellStyle = workbook.createCellStyle(); //样式对象

        HSSFDataFormat dataFormat = workbook.createDataFormat(); //日期格式

        cellStyle.setDataFormat(dataFormat.getFormat("yyyy 年 MM 月 dd 日")); //设置日期格式

        //处理数据行
        for (int i = 1; i < 10; i++) {
            row = sheet.createRow(i);
            row.createCell(0).setCellValue(i);
            row.createCell(1).setCellValue("张三"+i);

            //设置出生年月格式
            cell = row.createCell(2);
            cell.setCellValue(new Date());
            cell.setCellStyle(cellStyle);
        }
        try {
            workbook.write(new File("e:\\用户.xls"));
            workbook.close();
        } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
```

导出结果：



编号	姓名	出生年月
1	张三1	2017年03月07日
2	张三2	2017年03月07日
3	张三3	2017年03月07日
4	张三4	2017年03月07日
5	张三5	2017年03月07日
6	张三6	2017年03月07日
7	张三7	2017年03月07日
8	张三8	2017年03月07日
9	张三9	2017年03月07日

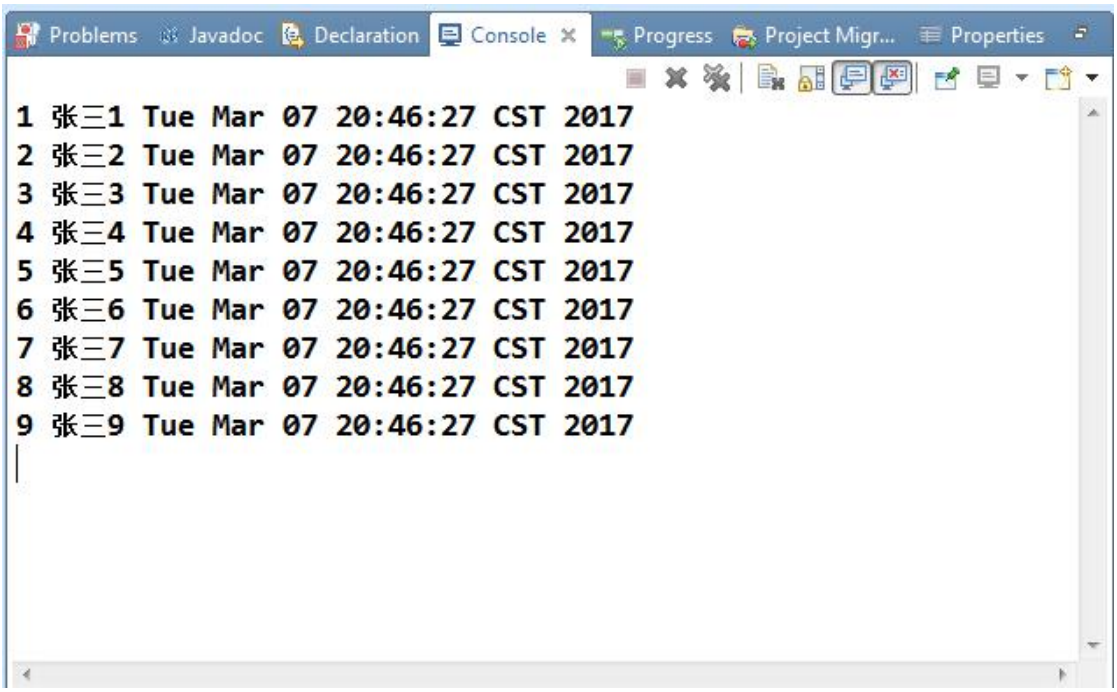
2) POI 导入

导入代码：

```
/**
     * @author gaozhy
     * @2017 年 3 月 8 日上午 9:37:46
     * @version 1.0
     * @description poi 导入
     */
    @Test
    public void importExcel() {
        try {
            // 获取本地 Excel 文件输入流，并创建工作薄对象
            HSSFWorkbook workbook = new HSSFWorkbook(new
FileInputStream("e:\\用户.xls"));

            // 获取工作表
            HSSFSheet sheet = workbook.getSheet("用户信息");

            // 声明行对象
            HSSFRow row = null;

            //注意：获取数据 需排除标题行 从数据行开始读取
            for (int i = 1; i <= sheet.getLastRowNum(); i++) {

                // 获取当前工作表中的数据行信息 数据行索引从 1 开始
                row = sheet.getRow(i);
                // 打印结果
```

```
            System.out.println((int)row.getCell(0).getNumericCellValue()
                    + " " + row.getCell(1).getStringCellValue() + " "
                    + row.getCell(2).getDateCellValue());
            }
        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
```

导入结果：



2.2 JXL 导入导出 Excel

1) JXL 导出

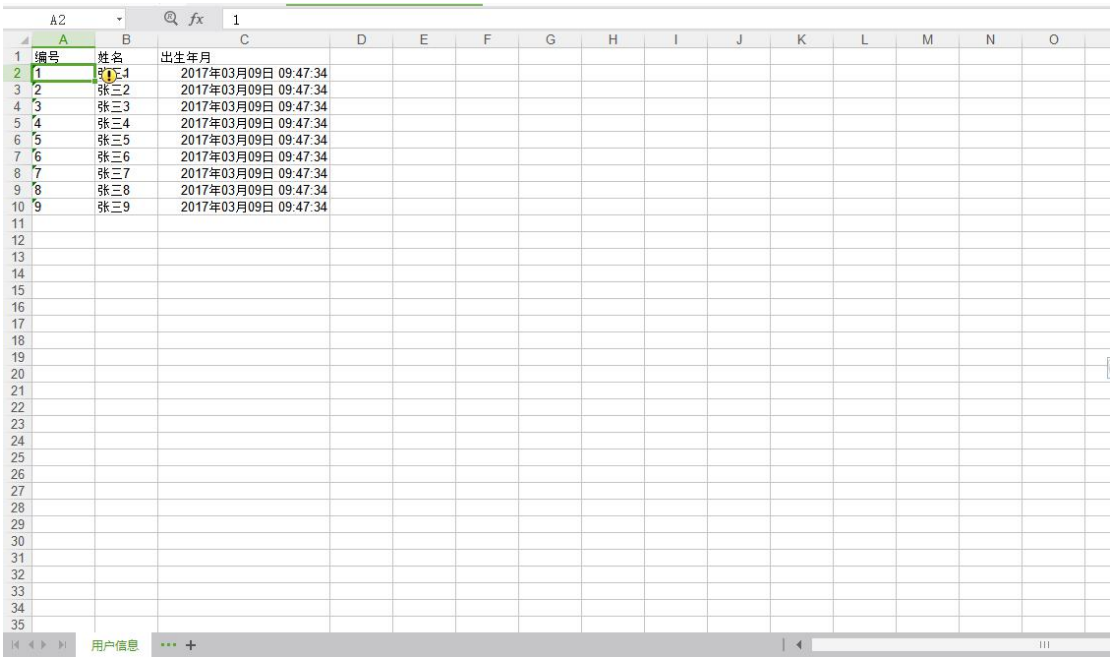
导出代码：

```
/**
     *
     * @author gaozhy
     * @2017 年 2 月 15 日下午 8:30:13
     * @version 1.0
     * @description jxl 导出 excel
     */
    @Test
```

```
    public void jxlExport(){
        try {
            // 创建 Excel 工作薄对象
            WritableWorkbook workbook = Workbook.createWorkbook(new
FileOutputStream("e:\\用户 1.xls"));

            //创建工作表
            //第一个参数：工作表 sheet 名称     第二个参数：工作表 sheet 索引
值   0 表示第一个 sheet 页索引
            WritableSheet sheet = workbook.createSheet("用户信息", 0);

            //设置
            sheet.setColumnView(2, 15);

            String[] title = {"编号","姓名","出生年月"};

            //创建单元格对象
            Label label = null;

            //处理标题栏
            for (int i = 0; i < title.length; i++) {
                //第一个参数  列索引  第二个参数 行索引 第三个参数 单元格内
容
                label = new Label(i,0,title[i]);

                //sheet 页添加单元格
                sheet.addCell(label);
            }

            for (int i = 1; i < 10; i++) {
                label = new Label(0,i,""+i);
                sheet.addCell(label);
                label = new Label(1,i,"张三"+i);
                sheet.addCell(label);

                //日期格式特殊处理
                DateFormat format = new DateFormat("yyyy 年 MM 月 dd 日
HH:mm:ss");

                WritableCellFormat cellFormat = new
WritableCellFormat(format);

                //第一个参数：列索引   第二个参数：行索引   第三个参数：日期
内容   第四个参数：日期格式
```

```
                sheet.addCell(new DateTime(2, i, new Date(),
cellFormat));
            }
            workbook.write();
            workbook.close();

        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
```

导出结果：



	A	B	C
1	编号	姓名	出生年月
2	1	张三1	2017年03月09日 09:47:34
3	2	张三2	2017年03月09日 09:47:34
4	3	张三3	2017年03月09日 09:47:34
5	4	张三4	2017年03月09日 09:47:34
6	5	张三5	2017年03月09日 09:47:34
7	6	张三6	2017年03月09日 09:47:34
8	7	张三7	2017年03月09日 09:47:34
9	8	张三8	2017年03月09日 09:47:34
10	9	张三9	2017年03月09日 09:47:34

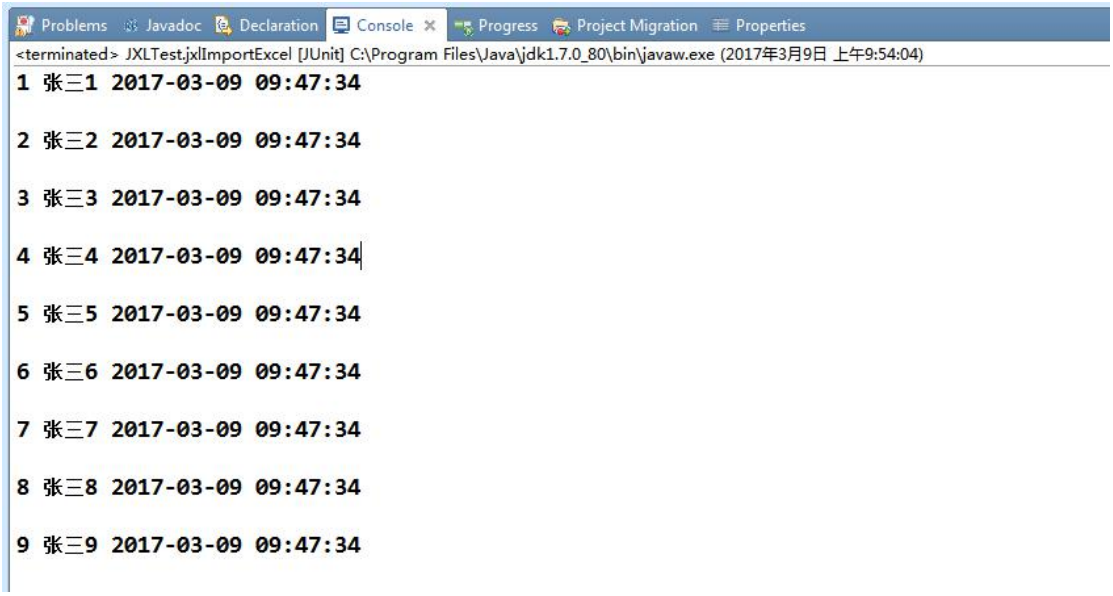
2) JXL 导入

导入代码：

```
/**
     *
     * @author gaozhy
     * @2017 年 2 月 15 日下午 8:20:31
     * @version 1.0
     * @description jxl 文件导入
     */
    @Test
    public void jxlImportExcel(){

        try {
            FileInputStream inputStream = new FileInputStream("e:\\用户
```

```
1.xls");
            //创建工作薄
            Workbook workbook = Workbook.getWorkbook(inputStream);
            //获取 sheet 页
            Sheet[] sheets = workbook.getSheets();

            //获取第一个 sheet
            Sheet sheet = sheets[0];

            //获取总行数
            int rows = sheet.getRows();

            for (int i = 1; i < rows; i++) {
                //获取指定索引 数据行 返回单元格数组
                Cell[] cells = sheet.getRow(i);

                for (int j = 0; j < sheet.getColumns(); j++) {

                    if(cells[j].getType() == CellType.DATE){

                        DateCell dateCell = (DateCell)cells[j];
                        // 注意：JXL 读取 Excel 日期时间多出了 8 个小时
                        // 获取格林时区
                        TimeZone zone = TimeZone.getTimeZone("GMT");
                        // 定义日期格式
                        SimpleDateFormat format = new
SimpleDateFormat("yyyy-MM-dd HH:mm:ss");

                        format.setTimeZone(zone);


    System.out.println(format.format(dateCell.getDate()));
                    }else{
                        System.out.print(cells[j].getContents()+" ");
                    }
                }
                System.out.println();
            }
        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
```

导入结果：

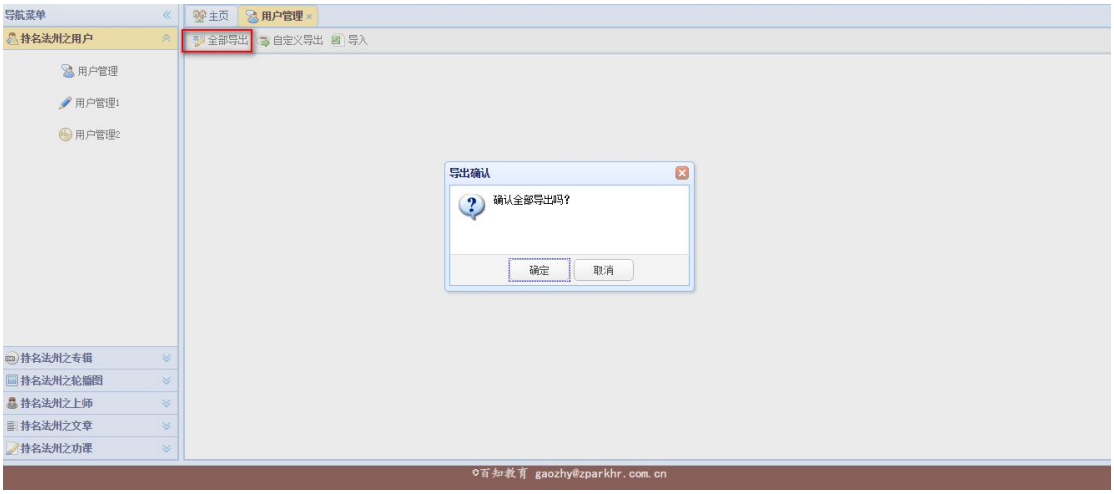


3. POI 和 JXL 实际应用

在上面的演示代码中，不难发现 POI 在导出数据到 Excel 时功能强大，导入处理相对复杂。而 JXL 恰恰相反，导入简单，导出复杂。所以企业在开发过程中常用 POI 做数据导出，JXL 做 Excel 数据导入。

3.1 POI 导出数据到 Excel

1) 导出所有数据



如上图所示，当点击全部导出时，需导出所用用户的全部信息，也就意味着用户表中的所有数据都要导出到 Excel 文件中。

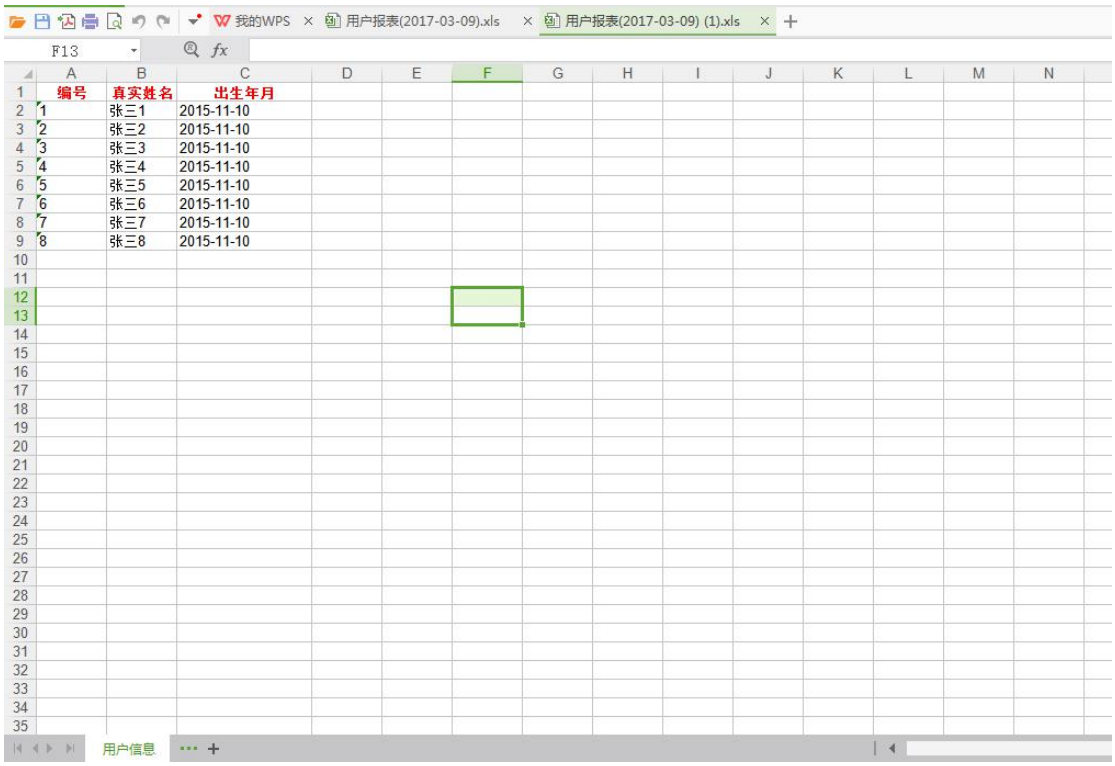
实现代码：

```
/**
     *
     * @author gaozhy
     * @2017 年 2 月 15 日下午 9:58:09
     * @version 1.0
     * @description 全部信息导出
     */
    @RequestMapping("/exportAll")
    public void exportAll(HttpServletResponse resp){
        //模拟从数据库查到的所有用户及用户信息
        List<User> users = new ArrayList<User>();
        User user = new User("1","张三 1","2015-11-10");
        User user1 = new User("2","张三 2","2015-11-10");
        User user2 = new User("3","张三 3","2015-11-10");
        User user3 = new User("4","张三 4","2015-11-10");
        User user4 = new User("5","张三 5","2015-11-10");
        User user5 = new User("6","张三 6","2015-11-10");
        User user6 = new User("7","张三 7","2015-11-10");
        User user7 = new User("8","张三 8","2015-11-10");
        users.add(user);
        users.add(user1);
        users.add(user2);
        users.add(user3);
        users.add(user4);
        users.add(user5);
        users.add(user6);
        users.add(user7);

        //创建工作薄
        HSSFWorkbook workbook = new HSSFWorkbook();
        //创建工作表
        HSSFSheet sheet = workbook.createSheet("用户信息");
        //设置列宽  第一个参数：列索引  第二个参数：列宽
        sheet.setColumnWidth(2, 4500);
        //创建导出样式
        HSSFCellStyle cellStyle = workbook.createCellStyle();
        //创建字体
        HSSFFont font = workbook.createFont();
        //设置字体颜色
        font.setColor(HSSFFont.COLOR_RED);
        //设置加粗
        font.setBold(true);
```

```
        //设置字体
        font.setFontName("宋体");
        //设置居中
        cellStyle.setAlignment(CellStyle.ALIGN_CENTER);
        //管理字体样式
        cellStyle.setFont(font);

        //创建标题栏
        HSSFRow row = sheet.createRow(0);
        HSSFCell cell = null;
        String[] titles = {"编号","真实姓名","出生年月"};
        for (int i = 0; i < titles.length; i++) {
            cell = row.createCell(i);
            cell.setCellValue(titles[i]);
            //标题行使用样式
            cell.setCellStyle(cellStyle);
        }

        for (int i = 1; i <= users.size(); i++) {
            //创建数据行对象
            row = sheet.createRow(i);
            //数据行第一列设值
            cell = row.createCell(0);
            cell.setCellValue(users.get(i-1).getUserId());
            //数据行第二列设值
            cell = row.createCell(1);
            cell.setCellValue(users.get(i-1).getRealname());
            //数据行第三列设值
            cell = row.createCell(2);
            cell.setCellValue(users.get(i-1).getDharmaName());
        }

        String fileName = "用户报表("+new
SimpleDateFormat("yyyy-MM-dd").format(new Date())+").xls";
        //处理中文下载名乱码
        try {
            fileName = new
String(fileName.getBytes("gbk"),"iso-8859-1");
            //设置 response
            resp.setContentType("application/vnd.ms-excel");

    resp.setHeader("content-disposition","attachment;filename="+fileN
ame);
```

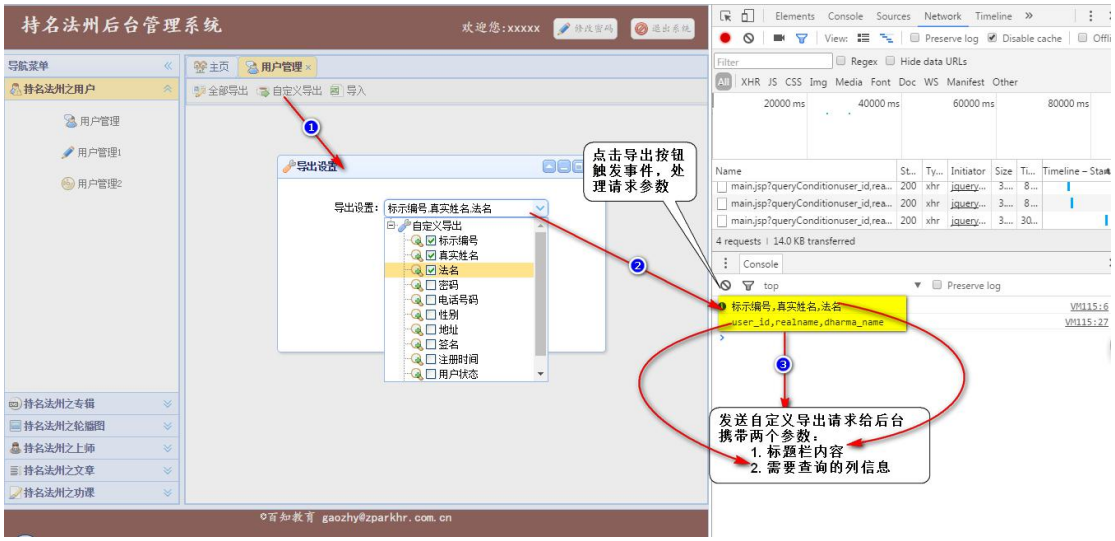
```
            workbook.write(resp.getOutputStream());
            workbook.close();
        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
```

效果图：



2) 自定义导出

原理图：



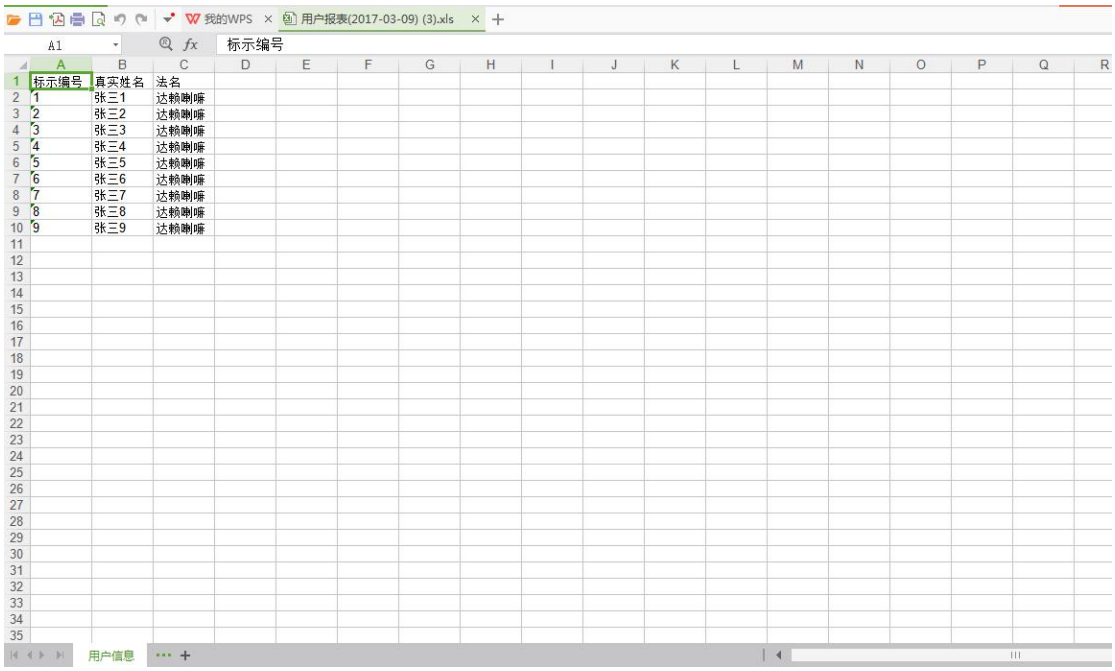
实现代码：

```
/**
     * @author gaozhy
     * @2017 年 2 月 15 日下午 9:57:33
     * @version 1.0
     * @param resp
     * @description 自定义导出
     */
    @RequestMapping("/customExport")
    // 参数一：需要查询的列信息 参数二：标题栏内容
    public void customExport(String queryCondition, String title,
HttpServletResponse resp) {
        try {
            // 根据列信息，查用户信息
            // select ${queryCondition} select cmfz_user
            List<User> list =
userService.queryUserByField(queryCondition);
            HSSFWorkbook workbook = new HSSFWorkbook();

            HSSFSheet sheet = workbook.createSheet("用户信息");

            // 获取所有标题
            String[] titles = title.split(",");

            // 获取导出字段
            String[] fields = queryCondition.split(",");

            // 处理标题栏
            HSSFRow row = sheet.createRow(0);

            for (int i = 0; i < titles.length; i++) {
                row.createCell(i).setCellValue(titles[i]);
            }

            // 处理数据行
            for (int i = 1; i <= list.size(); i++) {
                // 获得数据行对象
                HSSFRow dataRow = sheet.createRow(i);
                // 获取 User 对象
                User user = list.get(i - 1);

                Class<? extends User> c = user.getClass();
                // 拿到字段名
                for (int j = 0; j < fields.length; j++) {
```

```
                    // user_id 拆分字段
                    String[] str = fields[j].split("_");

                    // 处理 get 方法名
                    String name = "get";
                    // 属性没有 _ 拼接
                    if (str.length == 1) {
                        name += str[0].substring(0, 1).toUpperCase()
                                + str[0].substring(1, str[0].length());
                    } else {
                        name += str[0].substring(0, 1).toUpperCase()
                                + str[0].substring(1, str[0].length());
                        // user id
                        for (int k = 1; k < str.length; k++) {
                            name += str[k].substring(0, 1).toUpperCase()
                                    + str[k].substring(1,
str[k].length());
                        }
                    }
                    System.out.println(name);

                    // 调用 get 方法给单元格单元格设值
                    Object obj = c.getDeclaredMethod(name,
null).invoke(user,null);
                    dataRow.createCell(j).setCellValue(obj.toString());

                }
            }
            String fileName = "用户报表("
                    + new SimpleDateFormat("yyyy-MM-dd").format(new
Date())
                    + ").xls";
            // 处理中文下载名乱码

            fileName = new String(fileName.getBytes("gbk"),
"iso-8859-1");
            // 设置 response
            resp.setContentType("application/vnd.ms-excel");
            resp.setHeader("content-disposition",
"attachment;filename="
                    + fileName);

            workbook.write(resp.getOutputStream());
            workbook.close();
```

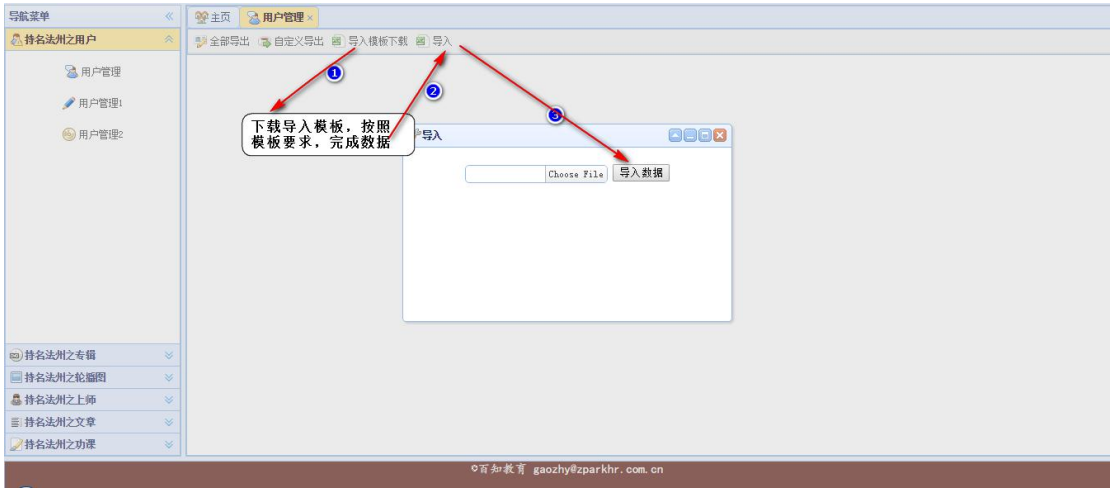
```
        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
```

效果图：



3.2 JXL 导入 Excel 数据保存到数据库

导入数据时，需下载导入模板，按照导入模板的要求填写数据，完成导入。



导入代码：

```
/**
     *
     * @author gaozhy
     * @2017 年 3 月 9 日上午 11:56:31
     * @version 1.0
     * @description jxl 导入 Excel 数据保存到数据库中
     */
    @RequestMapping("/importExcel")
    public void importExcel(MultipartFile multipartFile) {

        System.out.println(multipartFile.getOriginalFilename());

        List<User> list = new ArrayList<User>();

        try {
            InputStream inputStream = multipartFile.getInputStream();

            Workbook workbook = Workbook.getWorkbook(inputStream);

            Sheet sheet = workbook.getSheet("用户信息");

            Cell cell = null;
            User user = null;

            for (int i = 1; i < sheet.getRows(); i++) {

                user = new User();

                for (int j = 0; j < sheet.getColumns(); j++) {
                    //获取单元格  第一个参数：列索引  第二个参数：行索引
                    cell = sheet.getCell(j, i);

                    if (j == 0)
                        user.setUserId(cell.getContents());
                    else if (j == 1)
                        user.setRealname(cell.getContents());
                    else if (j == 2) {
                        if(cell.getType() == CellType.DATE){
                            DateCell dateCell = (DateCell)cell;
                            user.setBirthday(dateCell.getDate());
                        }
                    }
                }
                list.add(user);
```

```
            }
            if (list.size() != 0) {
                userService.importUsers(list);
            }
        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
```

UserMapper.xml 中批量插入用户数据：

```
<insert id="insertUsers">
    insert into cmfz_user(user_id,realname,create_time)
    values
    <foreach collection="list" item="item" index="index" separator=",">
        (#{item.userId},#{item.realname},#{item.birthday})
    </foreach>
</insert>
```

效果图：

user_id	realname	create_time
1	张三1	2010-05-01 08:00:00
2	张三2	2010-05-01 08:00:00
3	张三3	2010-05-01 08:00:00
4	张三4	2010-05-01 08:00:00
5	张三5	2010-05-01 08:00:00
6	张三6	2010-05-01 08:00:00
7	张三7	2010-05-01 08:00:00
8	张三8	2010-05-01 08:00:00
9	张三9	2010-05-01 08:00:00