**package** com.ssm.utils;  
  
**import** org.apache.poi.ss.usermodel.\*;  
**import** org.apache.poi.xssf.usermodel.\*;  
  
**import** javax.servlet.http.HttpServletResponse;  
**import** java.io.\*;  
**import** java.lang.reflect.Field;  
**import** java.lang.reflect.Method;  
**import** java.lang.reflect.Type;  
**import** java.util.Collection;  
**import** java.util.Date;  
**import** java.util.Iterator;  
  
**public class** ExcelUtil<T> {  
 */\*\*  
 \* 创建一个工作簿，把数据写入到工作簿  
 \** ***@param headers*** *table的 th 标题  
 \** ***@param dataset*** *封装的数据集合  
 \** ***@param fileName*** *手动输入的文件名  
 \** ***@param response*** *\*/  
// public void exportExcel(String[] headers, Collection<T> dataset, String fileName, HttpServletResponse response){* **public void** exportExcel(String[] headers, Collection<T> dataset, String fileName){  
 Workbook workbook=**new** XSSFWorkbook();*//声明一个工作簿(Excel工作簿)* Sheet sheet=workbook.createSheet(**"User"**);*//使用工作簿创建一个sheel就是excel新建一个表格* sheet.setDefaultColumnWidth((**short**)20);*//设置列宽15个字节* sheet.setDefaultRowHeight((**short**)600);  
 Row row = sheet.createRow(0);*//标题行 就是第一行的标题* CreationHelper creationHelper = workbook.getCreationHelper();  
 CellStyle cellStyle = workbook.createCellStyle();  
 cellStyle.setDataFormat(creationHelper.createDataFormat().getFormat(**"yyyy-mm-dd"**));*//m/dd/yy h:mm* cellStyle.setAlignment(HorizontalAlignment.***CENTER***);*//水平居中* cellStyle.setVerticalAlignment(VerticalAlignment.***CENTER***);*//垂直居中* cellStyle.setBorderBottom(BorderStyle.***THIN***);*//下表框* cellStyle.setBottomBorderColor(IndexedColors.***RED***.getIndex());*//边框颜色* **for** (**short** i=0;i<headers.**length**;i++){  
 Cell cell = row.createCell(i);*//创建单元格  
// XSSFRichTextString text=new XSSFRichTextString(headers[i]);//创建单元格数据对象* cell.setCellValue(headers[i]);*//把内容写入到表单元* }  
 **try** {  
 Iterator<T> iterator = dataset.iterator();*//遍历集合数据，产生数据行* **int** index = 0;*//初始化行* **while** (iterator.hasNext()) {  
 index++;*//行进行+1操作* row = sheet.createRow(index);*//创建该行 标题下的行* T t = iterator.next();*//得到对象* Class clazz = t.getClass();  
 Field[] fields = clazz.getDeclaredFields();*//利用反射，根据javabean属性的先后顺序，动态调用getXxx()得到属性值* **for** (**short** i = 0; i < headers.**length**; i++) {*//为什么用short不用int* Cell cell = row.createCell(i);*//创建单元格* Field field = fields[i];*//得到字段对象* Type type = field.getGenericType();*//得到字段的类型* String fieldName = field.getName();*//获取字段的名字* String mothedName = **"get"** + fieldName.substring(0, 1).toUpperCase() + fieldName.substring(1);*//getXxx()* Method method = clazz.getMethod(mothedName, **new** Class[]{});*//得到方法的全部* Object invoke = method.invoke(t, **new** Object[]{});*//得到方法的返回值类型* **if** (invoke!=**null**&&invoke!=**""**){  
*// XSSFRichTextString textString=new XSSFRichTextString(value);* cell.setCellStyle(cellStyle);*//使用单元格样式* cell.setCellType(CellType.***STRING***);*//设置单元格类型* **if** (type.toString().equals(**"class java.util.Date"**)) cell.setCellValue((Date) invoke);  
 **else** cell.setCellValue(invoke.toString());  
 }  
 }  
 }  
 saveExcel(workbook,fileName);*//调用方法保存工作簿* }**catch** (Exception e){  
 e.printStackTrace();  
 }  
 }  
  
 */\*\*  
 \* 保存工作簿的方法  
 \** ***@param workbook*** *工作簿 一张excel表格  
 \** ***@param fileName*** *\** ***@param response*** *\*/  
// public void saveExcel(Workbook workbook,String fileName,HttpServletResponse response) throws Exception {* **public void** saveExcel(Workbook workbook,String fileName) **throws** Exception {  
 fileName=fileName+**".xlsx"**;  
*// response.setContentType("application/x-msdownload");  
 /\*response.setHeader("Content-Disposition","attachment;filename="+new String(fileName.getBytes("gb2312"),"utf-8"));  
 BufferedOutputStream out=new BufferedOutputStream(response.getOutputStream());\*/* File file=**new** File(**"f:\\datatoexcel"**);  
 **if** (!file.exists()) file.mkdirs();  
 BufferedOutputStream out=**new** BufferedOutputStream(**new** FileOutputStream(file+File.***separator***+File.***separator***+fileName));  
 workbook.write(out);  
 out.flush();  
 out.close();  
 }  
}

**package** com.ssm.test;  
  
**import** com.opensymphony.xwork2.ActionSupport;  
**import** com.ssm.entry.User;  
**import** com.ssm.service.UserService;  
**import** com.ssm.utils.DataUtils;  
**import** com.ssm.utils.ExcelUtil;  
**import** org.apache.struts2.ServletActionContext;  
**import** org.easymock.EasyMock;  
**import** org.junit.Test;  
**import** org.junit.runner.RunWith;  
**import** org.springframework.beans.factory.annotation.Autowired;  
**import** org.springframework.stereotype.Controller;  
**import** org.springframework.test.context.ContextConfiguration;  
**import** org.springframework.test.context.junit4.SpringJUnit4ClassRunner;  
  
**import** javax.servlet.http.HttpServlet;  
**import** javax.servlet.http.HttpServletResponse;  
**import** java.util.ArrayList;  
**import** java.util.List;  
  
@RunWith(SpringJUnit4ClassRunner.**class**)  
@ContextConfiguration(locations = {**"classpath\*:conf/applicationContext.xml"**})  
**public class** DataToExcel **extends** HttpServlet {  
 @Autowired  
 **private** UserService **userService**;  
 @Test  
 **public void** toExcel(){  
 HttpServletResponse response = EasyMock.*createMock*(HttpServletResponse.**class**);  
 ExcelUtil<User> excel=**new** ExcelUtil<>();  
 List<User> users = **userService**.useAliasResultMap();  
 String[] headers={**"编号"**,**"姓名"**,**"密码"**,**"日期"**,**"年龄"**};  
 String fileName=**"用户信息表"**+ DataUtils.*getUID*();  
 excel.exportExcel(headers,users,fileName);  
 }  
}

**package** com.ssm.service.impl;  
  
**import** com.ssm.dao.UserDao;  
**import** com.ssm.entry.User;  
**import** com.ssm.mapper.UserMapper;  
**import** com.ssm.service.UserService;  
**import** org.springframework.beans.factory.annotation.Autowired;  
**import** org.springframework.stereotype.Service;  
  
**import** java.util.Date;  
**import** java.util.List;  
  
@Service(**"userService"**)  
**public class** UserServiceImpl **implements** UserService {  
  
 @Override  
 **public** List<User> useAliasResultMap() {  
 **return** userMapper.useAliasResultMap();  
 }  
  
}

**package** com.ssm.mapper;  
  
**import** com.ssm.entry.User;  
**import** org.springframework.stereotype.Repository;  
  
**import** java.util.List;  
  
@Repository(**"userMapper"**)  
**public interface** UserMapper {  
  
 List<User> useAliasResultMap();  
  
}

*<?***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.ssm.mapper.UserMapper"**>  
  
 *<!--当查询的字段有别名的时候-->* <**resultMap id="aliasdemo" type="user"**>  
 *<!--如果有id的时候要写id column是别名 property是属性名-->* <**id column="编号" property="id"** />  
 *<!--不是id的时候要使用result-->* <**result column="用户名" property="username"** />  
 <**result column="年龄" property="age"** />  
 </**resultMap**>  
 *<!--配合resultMap使用-->* <**select id="useAliasResultMap" resultMap="aliasdemo"**>  
 select **id** 编号,**username** 用户名,birthday,age 年龄 from user  
 </**select**>  
  
</**mapper**>