


```
3 data.add(Arrays.asList("111","222","333"));
4 data.add(Arrays.asList("111","222","333"));
5 data.add(Arrays.asList("111","222","333"));
6 List<String> head = Arrays.asList("表头1", "表头2", "表头3");
7 ExcelUtil.writeBySimple(filePath,data,head);
```

结果

	表头1	表头2	表头3
1	111	222	333
2	111	222	333
3	111	222	333
4			
5			

模型映射导出

1、定义好模型对象

```
1 package com.springboot.utils.excel.test;
2
3 import com.alibaba.excel.annotation.ExcelProperty;
4 import com.alibaba.excel.metadata.BaseRowModel;
5 import lombok.Data;
6 import lombok.EqualsAndHashCode;
7
8 /**
9  * @description:
10  * @author: chenmingjian
11  * @date: 19-4-3 14:44
12  */
13 @EqualsAndHashCode(callSuper = true)
14 @Data
15 public class TableHeaderExcelProperty extends BaseRowModel {
16
17     /**
18      * value: 表头名称
19      * index: 列的号, 0表示第一列
20      */
21     @ExcelProperty(value = "姓名", index = 0)
22     private String name;
23
24     @ExcelProperty(value = "年龄",index = 1)
25     private int age;
26
27     @ExcelProperty(value = "学校",index = 2)
28     private String school;
29 }
30
```

2、调用方法

```
1 String filePath = "/home/chenmingjian/Downloads/测试.xlsx";
2 ArrayList<TableHeaderExcelProperty> data = new ArrayList<>();
3 for(int i = 0; i < 4; i++){
4     TableHeaderExcelProperty tableHeaderExcelProperty = new TableHeaderExcelProperty();
5     tableHeaderExcelProperty.setName("cmj" + i);
6     tableHeaderExcelProperty.setAge(22 + i);
7     tableHeaderExcelProperty.setSchool("清华大学" + i);
8     data.add(tableHeaderExcelProperty);
9 }
10
11 ExcelUtil.writeWithTemplate(filePath,data);
```

多个Sheet导出

1、定义好模型对象

```
1 package com.springboot.utils.excel.test;
2
3 import com.alibaba.excel.annotation.ExcelProperty;
4 import com.alibaba.excel.metadata.BaseRowModel;
5 import lombok.Data;
6 import lombok.EqualsAndHashCode;
7
8 /**
9  * @description:
10  * @author: chenmingjian
11  * @date: 19-4-3 14:44
12  */
13 @EqualsAndHashCode(callSuper = true)
14 @Data
15 public class TableHeaderExcelProperty extends BaseRowModel {
16
17     /**
18      * value: 表头名称
19      * index: 列的号, 0表示第一列
20      */
21     @ExcelProperty(value = "姓名", index = 0)
22     private String name;
23
24     @ExcelProperty(value = "年龄",index = 1)
25     private int age;
26
27     @ExcelProperty(value = "学校",index = 2)
28     private String school;
29 }
30
```

2、调用方法

```
1 ArrayList<ExcelUtil.MultipleSheelPropety> list1 = new ArrayList<>();
2 for(int j = 1; j < 4; j++){
3     ArrayList<TableHeaderExcelProperty> list = new ArrayList<>();
4     for(int i = 0; i < 4; i++){
5         TableHeaderExcelProperty tableHeaderExcelProperty = new TableHeaderExcelProperty();
6         tableHeaderExcelProperty.setName("cmj" + i);
7         tableHeaderExcelProperty.setAge(22 + i);
8         tableHeaderExcelProperty.setSchool("清华大学" + i);
9         list.add(tableHeaderExcelProperty);
10    }
11
12    Sheet sheet = new Sheet(j, 0);
13    sheet.setSheetName("sheet" + j);
14
15    ExcelUtil.MultipleSheelPropety multipleSheelPropety = new ExcelUtil.MultipleSheelPropety();
16    multipleSheelPropety.setData(list);
17    multipleSheelPropety.setSheet(sheet);
18
19    list1.add(multipleSheelPropety);
20
21 }
22
23 ExcelUtil.writeWithMultipleSheel("/home/chenmingjian/Downloads/aaa.xlsx",list1);
```

工具类

```
1 package com.springboot.utils.excel;
2
3 import com.alibaba.excel.EasyExcelFactory;
4 import com.alibaba.excel.ExcelWriter;
5 import com.alibaba.excel.context.AnalysisContext;
6 import com.alibaba.excel.event.AnalysisEventListener;
7 import com.alibaba.excel.metadata.BaseRowModel;
8 import com.alibaba.excel.metadata.Sheet;
9 import lombok.Data;
```

6

45

分享海报说明

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45

45


45


45


45


45



```
10 import lombok.Getter;
11 import lombok.Setter;
12 import lombok.extern.slf4j.Slf4j;
13 import org.springframework.util.CollectionUtils;
14 import org.springframework.util.StringUtils;
15
16 import java.io.*;
17 import java.util.ArrayList;
18 import java.util.Collections;
19 import java.util.List;
20
21 /**
22  * @description:
23  * @author: chenmingjian
24  * @date: 19-3-18 16:16
25  */
26 @Slf4j
27 public class ExcelUtil {
28
29     private static Sheet initSheet;
30
31     static {
32         initSheet = new Sheet(1, 0);
33         initSheet.setSheetName("sheet");
34         // 设置自适应宽度
35         initSheet.setAutoWidth(Boolean.TRUE);
36     }
37
38     /**
39      * 读取少于1000行数据
40      * @param filePath 文件绝对路径
41      * @return
42      */
43     public static List<Object> readLessThan1000Row(String filePath){
44         return readLessThan1000RowBySheet(filePath,null);
45     }
46
47     /**
48      * 读小于1000行数据，带样式
49      * filePath 文件绝对路径
50      * initSheet :
51      *     sheetNo: sheet页码，默认为1
52      *     headLineMun: 从第几行开始读取数据，默认为0，表示从第一行开始读取
53      *     clazz: 返回数据List<Object> 中Object的类名
54      */
55     public static List<Object> readLessThan1000RowBySheet(String filePath, Sheet sheet){
56         if(!StringUtils.hasText(filePath)){
57             return null;
58         }
59
60         sheet = sheet != null ? sheet : initSheet;
61
62         InputStream fileStream = null;
63         try {
64             fileStream = new FileInputStream(filePath);
65             return EasyExcelFactory.read(fileStream, sheet);
66         } catch (FileNotFoundException e) {
67             log.info("找不到文件或文件路径错误，文件: {}", filePath);
68         }finally {
69             try {
70                 if(fileStream != null){
71                     fileStream.close();
72                 }
73             } catch (IOException e) {
74                 log.info("excel文件读取失败，失败原因: {}", e);
75             }
76         }
77         return null;
78     }
79
80     /**
81      * 读大于1000行数据
82      * @param filePath 文件觉得路径
83      * @return
84      */
85     public static List<Object> readMoreThan1000Row(String filePath){
86         return readMoreThan1000RowBySheet(filePath,null);
87     }
88
89     /**
90      * 读大于1000行数据，带样式
91      * @param filePath 文件觉得路径
92      * @return
93      */
94     public static List<Object> readMoreThan1000RowBySheet(String filePath, Sheet sheet){
95         if(!StringUtils.hasText(filePath)){
96             return null;
97         }
98
99         sheet = sheet != null ? sheet : initSheet;
100
101         InputStream fileStream = null;
102         try {
103             fileStream = new FileInputStream(filePath);
104             Excellistener excellistener = new Excellistener();
105             EasyExcelFactory.readBySax(fileStream, sheet, excellistener);
106             return excellistener.getDatas();
107         } catch (FileNotFoundException e) {
108             log.error("找不到文件或文件路径错误，文件: {}", filePath);
109         }finally {
110             try {
111                 if(fileStream != null){
112                     fileStream.close();
113                 }
114             } catch (IOException e) {
115                 log.error("excel文件读取失败，失败原因: {}", e);
116             }
117         }
118         return null;
119     }
120
121     /**
122      * 生成excle
123      * @param filePath 绝对路径，如: /home/chenmingjian/Downloads/aaa.xlsx
124      * @param data 数据源
125      * @param head 表头
126      */
127     public static void writeBySimple(String filePath, List<List<Object>> data, List<String> head){
128         writeSimpleBySheet(filePath,data,head,null);
129     }
130
131     /**
132      * 生成excle
133      * @param filePath 绝对路径，如: /home/chenmingjian/Downloads/aaa.xlsx
134      * @param data 数据源
135      * @param sheet excle页面样式
136      * @param head 表头
137      */
138     public static void writeSimpleBySheet(String filePath, List<List<Object>> data, List<String> head, Shee
139         sheet = (sheet != null) ? sheet : initSheet;
140
141         if(head != null){
142             List<List<String>> list = new ArrayList<>();
143             head.forEach(h -> list.add(Collections.singletonList(h)));
144             sheet.setHead(list);
145         }
146
147         OutputStream outputStream = null;
148         ExcelWriter writer = null;
149         try {
150             outputStream = new FileOutputStream(filePath);
```


6


45











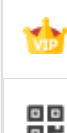
分享海报说明














```
292         // context.getCurrentRowNum()
293         if (object != null) {
294             datas.add(object);
295         }
296     }
297
298     /**
299     * 解析完所有数据后会调用该方法
300     */
301     @Override
302     public void doAfterAllAnalysed(AnalysisContext context) {
303         // 解析结束销毁不用的资源
304     }
305
306
307 }
308
309 /*****匿名内部类结束，可以提取出去*****/
310 }
311
312
```


测试类


```
1 package com.springboot.utils.excel;
2
3 import com.alibaba.excel.annotation.ExcelProperty;
4 import com.alibaba.excel.metadata.BaseRowModel;
5 import com.alibaba.excel.metadata.Sheet;
6 import lombok.Data;
7 import lombok.EqualsAndHashCode;
8 import org.junit.runner.RunWith;
9 import org.springframework.boot.test.context.SpringBootTest;
10 import org.springframework.test.context.junit4.SpringRunner;
11
12 import java.util.ArrayList;
13 import java.util.Arrays;
14 import java.util.List;
15
16 /**
17  * @description: 测试类
18  * @author: chenmingjian
19  * @date: 19-4-4 15:24
20  */
21 @SpringBootTest
22 @RunWith(SpringRunner.class)
23 public class Test {
24
25     /**
26      * 读取少于1000行的excle
27      */
28     @org.junit.Test
29     public void readLessThan1000Row(){
30         String filePath = "/home/chenmingjian/Downloads/测试.xlsx";
31         List<Object> objects = ExcelUtil.readLessThan1000Row(filePath);
32         objects.forEach(System.out::println);
33     }
34
35     /**
36      * 读取少于1000行的excle，可以指定sheet 和从几行读起
37      */
38     @org.junit.Test
39     public void readLessThan1000RowBySheet(){
40         String filePath = "/home/chenmingjian/Downloads/测试.xlsx";
41         Sheet sheet = new Sheet(1, 1);
42         List<Object> objects = ExcelUtil.readLessThan1000RowBySheet(filePath,sheet);
43         objects.forEach(System.out::println);
44     }
45
46     /**
47      * 读取大于1000行的excle
48      * 带sheet 参数的方法可参照测试方法readLessThan1000RowBySheet()
49      */
50     @org.junit.Test
51     public void readMoreThan1000Row(){
52         String filePath = "/home/chenmingjian/Downloads/测试.xlsx";
53         List<Object> objects = ExcelUtil.readMoreThan1000Row(filePath);
54         objects.forEach(System.out::println);
55     }
56
57
58     /**
59      * 生成excle
60      * 带sheet 参数的方法可参照测试方法readLessThan1000RowBySheet()
61      */
62     @org.junit.Test
63     public void writeBySimple(){
64         String filePath = "/home/chenmingjian/Downloads/测试.xlsx";
65         List<List<Object>> data = new ArrayList<>();
66         data.add(Arrays.asList("111","222","333"));
67         data.add(Arrays.asList("111","222","333"));
68         data.add(Arrays.asList("111","222","333"));
69         List<String> head = Arrays.asList("表头1", "表头2", "表头3");
70         ExcelUtil.writeBySimple(filePath,data,head);
71     }
72
73
74     /**
75      * 生成excle，带用模型
76      * 带sheet 参数的方法可参照测试方法readLessThan1000RowBySheet()
77      */
78     @org.junit.Test
79     public void writeWithTemplate(){
80         String filePath = "/home/chenmingjian/Downloads/测试.xlsx";
81         ArrayList<TableHeaderExcelProperty> data = new ArrayList<>();
82         for(int i = 0; i < 4; i++){
83             TableHeaderExcelProperty tableHeaderExcelProperty = new TableHeaderExcelProperty();
84             tableHeaderExcelProperty.setName("cmj" + i);
85             tableHeaderExcelProperty.setAge(22 + i);
86             tableHeaderExcelProperty.setSchool("清华大学" + i);
87             data.add(tableHeaderExcelProperty);
88         }
89         ExcelUtil.writeWithTemplate(filePath,data);
90     }
91
92
93     /**
94      * 生成excle，带用模型，带多个sheet
95      */
96     @org.junit.Test
97     public void writeWithMultipleSheel(){
98         ArrayList<ExcelUtil.MultipleSheelPropety> list1 = new ArrayList<>();
99         for(int j = 1; j < 4; j++){
100             ArrayList<TableHeaderExcelProperty> list = new ArrayList<>();
101             for(int i = 0; i < 4; i++){
102                 TableHeaderExcelProperty tableHeaderExcelProperty = new TableHeaderExcelProperty();
103                 tableHeaderExcelProperty.setName("cmj" + i);
104                 tableHeaderExcelProperty.setAge(22 + i);
105                 tableHeaderExcelProperty.setSchool("清华大学" + i);
106                 list.add(tableHeaderExcelProperty);
107             }
108
109             Sheet sheet = new Sheet(j, 0);
110             sheet.setSheetName("sheet" + j);
111
112             ExcelUtil.MultipleSheelPropety multipleSheelPropety = new ExcelUtil.MultipleSheelPropety();
113             multipleSheelPropety.setData(list);
114             multipleSheelPropety.setSheet(sheet);
115
```


6


45











分享海报说明












```
116         list1.add(multipleSheelProperty);
117     }
118 }
119
120 ExcelUtil.writeWithMultipleSheel("/home/chenmingjian/Downloads/aaa.xlsx",list1);
121
122 }
123
124
125 /***** 匿名内部类，实际开发中该对象要提取出去*****/
126
127
128 /**
129  * @description:
130  * @author: chenmingjian
131  * @date: 19-4-3 14:44
132  */
133 @EqualsAndHashCode(callSuper = true)
134 @Data
135 public static class TableHeaderExcelProperty extends BaseRowModel {
136
137     /**
138      * value: 表头名称
139      * index: 列的号, 0表示第一列
140      */
141     @ExcelProperty(value = "姓名", index = 0)
142     private String name;
143
144     @ExcelProperty(value = "年龄",index = 1)
145     private int age;
146
147     @ExcelProperty(value = "学校",index = 2)
148     private String school;
149 }
150
151 /***** 匿名内部类，实际开发中该对象要提取出去*****/
152
153 }
154
155 }
```


6





分享海报说明


45











微软人工智能教育与学习共建社区

我们将在此提供人工智能应用开发的真实案例，与广大师生、开发者一起学习、一起贡献!

想对作者说点什么

weixin_44248561： 工具类报错啊

```
1
2 for (MultipleSheelPropety multipleSheelProperty : multipleSheelPropetys) {
3     Sheet sheet = multipleSheelProperty.getSheet() != null ? multipleSheelProperty.getSheet() : initSheet;
4     if(!CollectionUtils.isEmpty(multipleSheelProperty.getData())){
5         sheet.setClazz(multipleSheelProperty.getData().get(0).getClass());
6     }
7     writer.write(multipleSheelProperty.getData(), sheet);
8 }
9
```

(2周前 #12楼) 查看回复(12)

渣渣it男： 这里面读取大于1000行是异步进行的，怎么知道什么时候已经读取完了？

(2周前 #11楼) 查看回复(1)

weixin_41175630： Exception in thread "main" java.lang.NoClassDefFoundError: org/apache/poi/ss/usermodel/Table at java.lang.ClassLoader.defineClass1(Native Method) at java.lang.ClassLoader.defineClass(ClassLoader.java:763) at java.security.SecureClassLoader.defineClass(SecureClassLoader.java:142) at java.net.URLClassLoader.defineClass(URLClassLoader.java:468) 据说个错右没有人遇到过?

(1个月前 #10楼) 查看回复(1)

登录 查看 45 条热评

Java解析excel工具easyexcel 助你快速简单避免OOM	阅读量 4万+
Java解析、生成Excel比较有名的框架有Apachepoi、jxl。但他们都存在一个严重的问题就是非常的耗内存，poi有一... 博文 来自： 成长中的巨人	
如何优雅地实现 Excel 文件导出功能？（阿里出品的 EasyExcel，安利一波）	阅读量 646
EasyExcel前言导出是后台管理系统的常用功能，当数据量特别大的时候会内存溢出和卡顿页面，曾经自己封装过一... 博文 来自： Mr_chen的博客	
easyexcel 导入excel内存溢出	阅读量 449
之前使用的是阿里开源的easyexcel，可以避免大部分的OOM情况，之前测试的时候也确实是，100Mexcel文件sh... 博文 来自： hello world	
easyexcel 读取大excel 十万级 java	阅读量 2290
转发请注明：https://blog.csdn.net/somdip/article/details/85243005<dependency> <groupid>co... 博文 来自： somdip的博客	
EasyExcel阿里开源excel导入导出	阅读量 1058
一、为什么使用easyexcelJava解析、生成Excel比较有名的框架有Apachepoi、jxl。但他们都存在一个严重的问题就... 博文 来自： HOME的博客	
easyExcel	阅读量 185
https://blog.csdn.net/jiangjiandecsd/article/details/81115622https://github.com/alibaba/easyexcel 博文 来自： u011243684的博客	
阿里easyExcel进行导出遇到问题	03-07
我用的是阿里的easyExcel 进行导出，导出的Excel中的数据是以.zip 结尾的一个文件，哪位大神帮忙看一下，这是我的代码...	论坛
基于EasyExcel的excel解析工具类。	02-25
基于EasyExcel框架的excel解析工具类。EasyExcel是阿里的excel解析工具，使用快捷方便，轻量。封装了常用的方法。使用前需要了解easyEx...	下载
谁用过easyexcel，怎么用它实现锁定单元格和下拉表头的功能	12-05
-	问答
史上最全的Excel导入导出之easyexcel - 我的猪夫跑了 - CSDN博客	6-24
史上最全的Excel导入导出之easyexcel - 很多时候犯错都..._CSDN博客	7-20
easyexcel-好用的java Excel工具	11-22
好用的java Excel工具,代替poi easyexcel重写了poi对07版Excel的解析，能够原本一个3M的excel用POI sax依然需要100M左右内存降低到KB...	下载
史上最全的Excel导入导出之easyexcel - weixin_3921448..._CSDN博客	6-20
使用EasyExcel对excel进行数据处理(导入导出) - wmx199..._CSDN博客	7-3
使用阿里开源easyexcel工具遇坑案例	阅读量 1万+
项目地址：https://github.com/alibaba/easyexcel需要实现：上传excel文件，第一行需要按照模板固定一行文字： ... 博文 来自： daisy_apk的博客	
<div><div>姜小哲 6篇文章 排名:千里之外</div><div>陈永佳 132篇文章 排名:千里之外</div><div>laixiaoxing 225篇文章 排名:千里之外</div><div>somdip 72篇文章 排名:千里之外</div></div>	
使用EasyExcel导入导出Excel - 很多时候犯错都是在不知..._CSDN博客	7-30
使用阿里开源easyexcel工具遇坑案例 - daisy_apk的博客 - CSDN博客	8-20
EasyExcel使用的正确姿势，工具类封装	阅读量 3448
项目中很可能用到导出excel文件的需求。easyexcel代码量较小，使用简单，而且性能较佳，是一个非常好的选择。 ... 博文 来自： 明明如月的专栏	
使用easy-Excel实现Excel的读取与导出	阅读量 1256
一.Excel数据的读取1.java代码packagecom.immo.framework.utils;importjava.io.FileInputStream;importjava.io.I... 博文 来自： qq_34542964的博客	
java导出excel表格 使用alibaba easyexcel - 如果有一..._CSDN博客	7-6









