```
 response  HttpServletResponse
     * @param list      数据 list，每个元素为一个 BaseRowModel
     * @param fileName  导出的文件名
     * @param sheetName 导入文件的 sheet 名
     * @param object    映射实体类，Excel 模型
     */
    public static void writeExcel(HttpServletResponse response, List<? extends BaseRowModel> list,
                                  String fileName, String sheetName, BaseRowModel object) {
        try {
            ExcelWriter writer = new ExcelWriter(getOutputStream(fileName, response), ExcelTypeEnum.XLSX);
            Sheet sheet = new Sheet(1, 0, object.getClass());
          sheet.setSheetName(sheetName);
          writer.write(list, sheet);
          writer.finish();
        } catch (Exception e) {
          log.error("excel输出异常\n" + JSONObject.toJSON(list));
        }
    }

    /**
     * 导出 Excel ： 多个 sheet，带表头
     *
     * @param response  HttpServletResponse
     * @param list      数据 list，每个元素为一个 BaseRowModel
     * @param fileName  导出的文件名
     * @param sheetName 导入文件的 sheet 名
     * @param object    映射实体类，Excel 模型
     */
    public static ExcelWriterFactroy writeExcelWithSheets(HttpServletResponse response, List<? extends BaseRowModel> list,
                                                          String fileName, String sheetName, BaseRowModel object) {
        ExcelWriterFactroy writer = new ExcelWriterFactroy(getOutputStream(fileName, response), ExcelTypeEnum.XLSX);
        Sheet sheet = new Sheet(1, 0, object.getClass());
        sheet.setSheetName(sheetName);
        writer.write(list, sheet);
        return writer;
    }
```

```
    /**
     * 导出文件时为Writer生成OutputStream
     */
    private static OutputStream getOutputStream(String fileName,
HttpServletResponse response) {
        //创建本地文件
        String filePath = fileName + ".xlsx";
        File dbfFile = new File(filePath);
        try {
            if (!dbfFile.exists() || dbfFile.isDirectory()) {
                dbfFile.createNewFile();
            }
            fileName = new String(filePath.getBytes(), "ISO-8859-1");
            response.setContentType("application/vnd.ms-excel");
            response.setCharacterEncoding("utf-8");
            response.addHeader("Content-Disposition", "filename=" +
fileName);
            response.setContentType("application/vnd.ms-excel;charset=utf-
8");
            response.flushBuffer();
            return response.getOutputStream();
        } catch (IOException e) {
            throw new ExcelException("创建文件失败！");
        }
    }

    /**
     * 返回 ExcelReader
     *
     * @param excel         需要解析的 Excel 文件
     * @param excelListener new ExcelListener()
     */
    private static ExcelReader getReader(MultipartFile excel,
                                         ExcelListener excelListener) {
        String filename = excel.getOriginalFilename();
        if (filename == null || (!filename.toLowerCase().endsWith(".xls") &&
!filename.toLowerCase().endsWith(".xlsx"))) {
            throw new ExcelException("文件格式错误！");
        }
        InputStream inputStream;
        try {
            inputStream = new BufferedInputStream(excel.getInputStream());
            return new ExcelReader(inputStream, null, excelListener, false);
        } catch (IOException e) {
            e.printStackTrace();
        }
        return n am());
            // 返回 ExcelReaderBuilder
```

```
            return EasyExcel.read(inputStream, excelListener);
        } catch (IOException e) {
            throw new ExcelException("读取文件失败！");
        }
    }
}
```

四、新版本工具类使用例子

实体类（使用@ExcelProperty注解）

```
package com.zzjdyf.tools.easyexcel;

import com.alibaba.excel.annotation.ExcelProperty;
import lombok.Data;

@Data
public class User {
    @ExcelProperty("ID")
    private Long id;

    @ExcelProperty("姓名")
    private String name;

    @ExcelProperty("年龄")
    private Integer age;
}
```

导入Excel数据

```
import com.zzjdyf.tools.easyexcel.EasyExcelUtil;
import org.springframework.web.multipart.MultipartFile;

import java.util.List;

public class ExcelImportExample {
```

```
    public void importExcel(MultipartFile file) {
        // 导入Excel文件，并将数据映射到User类
        List<Object> userList = EasyExcelUtil.readExcel(file, User.class);

        // 处理导入的数据
        for (Object obj : userList) {
            User user = (User) obj;
            System.out.println(user);
        }
    }
}
```

导出Excel数据

```
import com.zzjdyf.tools.easyexcel.EasyExcelUtil;
import javax.servlet.http.HttpServletResponse;
import java.util.ArrayList;
import java.util.List;

public class ExcelExportExample {

    public void exportExcel(HttpServletResponse response) {
        // 创建一些示例数据
        List<User> userList = new ArrayList<>();
        userList.add(new User(1L, "Alice", 30));
        userList.add(new User(2L, "Bob", 25));
        userList.add(new User(3L, "Charlie", 35));

        // 导出Excel文件
        EasyExcelUtil.writeExcel(response, userList, "用户信息", "Sheet1",
User.class);
    }
}
```

导入指定Sheet的数据

```
import com.zzjdyf.tools.easyexcel.EasyExcelUtil;
import org.springframework.web.multipart.MultipartFile;
```

```
import java.util.List;

public class ExcelImportExample {

    public void importExcel(MultipartFile file, int sheetNo) {
        // 从指定的Sheet导入数据
        List<Object> userList = EasyExcelUtil.readExcel(file, User.class,
sheetNo);

        // 处理导入的数据
        for (Object obj : userList) {
            User user = (User) obj;
            System.out.println(user);
        }
    }
}
```

导出到多个Sheet

```
import com.zzjdyf.tools.easyexcel.EasyExcelUtil;
import com.alibaba.excel.write.builder.ExcelWriterBuilder;
import javax.servlet.http.HttpServletResponse;
import java.util.ArrayList;
import java.util.List;

public class ExcelExportExample {

    public void exportExcelWithSheets(HttpServletResponse response) {
        // 创建一些示例数据
        List<User> userList1 = new ArrayList<>();
        userList1.add(new User(1L, "Alice", 30));
        userList1.add(new User(2L, "Bob", 25));

        List<User> userList2 = new ArrayList<>();
        userList2.add(new User(3L, "Charlie", 35));
        userList2.add(new User(4L, "David", 28));

        // 导出第一个Sheet
        ExcelWriterBuilder writerBuilder =
EasyExcelUtil.writeExcelWithSheets(response, userList1, "用户信息",
"Sheet1", User.class);
```

```
        // 导出第二个Sheet
        writerBuilder.sheet("Sheet2").doWrite(userList2);
    }
}
```

以上便是EasyExcel从1.x到2.x的升级，以及基础的例子，希望对你也有帮助。

原文链接: https://juejin.cn/post/7377326783581159464