# 09. Springboot+Vue实现批量导入导出excel功能

# 1. 批量导出excel

## 1.1. 页面代码

页面

<el-button type="success" style="margin-left: 10px" @click="exp()">导出报表</el-button>

js:

exp() {  
 let user = localStorage.getItem("user");  
 location.href = 'http://localhost:8080/api/type/export?token=' + JSON.parse(user).token  
},

## 1.2. 后台接口

@GetMapping("/export")  
public Result export(HttpServletResponse response) throws IOException {  
 // 思考：  
 // 要一行一行的组装数据，塞到一个list里面  
 // 每一行数据，其实就对应数据库表中的一行数据，也就是对应Java的一个实体类Type  
 // 我们怎么知道它某一列就是对应某个表头呢？？ 需要映射数据，我们需要一个Map<key,value>，把这个map塞到list里  
  
 // 干！  
 // 1. 从数据库中查询出所有数据  
 List<Type> all = typeService.findAll();  
  
 if (CollectionUtil.isEmpty(all)) {  
 throw new CustomException("未找到数据");  
 }  
  
 // 2. 定义一个 List，存储处理之后的数据，用于塞到 list 里  
 List<Map<String, Object>> list = new ArrayList<>(all.size());  
  
 // 3. 定义Map<key,value> 出来，遍历每一条数据，然后封装到 Map<key,value> 里，把这个 map 塞到 list 里  
 for (Type type : all) {  
 Map<String, Object> row = new HashMap<>();  
 row.put("分类名称", type.getName());  
 row.put("分类描述", type.getDescription());  
  
 list.add(row);  
 }  
  
 // 4. 创建一个 ExcelWriter，把 list 数据用这个writer写出来（生成出来）  
 ExcelWriter wr = ExcelUtil.getWriter(true);  
 wr.write(list, true);  
  
 // 5. 把这个 excel 下载下来  
 response.setContentType("application/vnd.openxmlformats-officedocument.spreadsheetml.sheet;charset=utf-8");  
 response.setHeader("Content-Disposition","attachment;filename=type.xlsx");  
 ServletOutputStream out = response.getOutputStream();  
 wr.flush(out, true);  
 wr.close();  
 IoUtil.close(System.out);  
  
  
 return Result.success();  
}

Tips：会有个报错：

cn.hutool.core.exceptions.DependencyException: You need to add dependency of 'poi-ooxml' to your project, and version >= 4.1.2

<dependency>  
 <groupId>org.apache.poi</groupId>  
 <artifactId>poi-ooxml</artifactId>  
 <version>4.1.2</version>  
</dependency>

# 2. 批量导入

## 2.1. 页面代码

<el-upload action="http://localhost:8080/api/type/upload" style="display: inline-block; margin-left: 10px" :show-file-list="false" :on-success="successUpload">  
 <el-button size="small" type="primary">批量导入</el-button>  
</el-upload>

successUpload(res) {  
 if (res.code === '0') {  
 this.$message.success("批量导入成功")  
 this.findBySearch()  
 } else {  
 this.$message.error(res.msg)  
 }  
}

## 2.2. 后台接口

@PostMapping("/upload")  
public Result upload(MultipartFile file) throws IOException {  
 List<Type> infoList = ExcelUtil.getReader(file.getInputStream()).readAll(Type.class);  
 if (!CollectionUtil.isEmpty(infoList)) {  
 for (Type type : infoList) {  
 try {  
 typeService.add(type);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
 }  
 return Result.success();  
}

注意：实体类上面需要加@Alias("分类名称")，否则没法自动映射

批量导入的接口放行

// 加自定义拦截器JwtInterceptor，设置拦截规则  
@Override  
public void addInterceptors(InterceptorRegistry registry) {  
 registry.addInterceptor(jwtInterceptor).addPathPatterns("/api/\*\*")  
 .excludePathPatterns("/api/files/\*\*")  
 .excludePathPatterns("/api/type/upload")  
 .excludePathPatterns("/api/admin/login")  
 .excludePathPatterns("/api/admin/register");  
}