

Lairx-004-总表模块

Lx4-TDR 项目手册

Total Directory Report

Flashy

2024-3-1

修改记录

版本	更新日期	描述
1.0	2023/8/20	建立初始版本
1.1	2024/3/1	添加总表模块相关文档

目 录

目 录	0
第 1 章 总表介绍和使用说明	1
1.1 总表简介	1
1.2 总表使用说明	1
1.3 总表导入器使用说明	1
第 2 章 相关库说明	1
2.1 xlwings 库	1
2.2 openpyxl 库	1
第 3 章 接口与功能实现	1
3.1 SumTable 模块	1
3.1.1 SumTable 模块使用	1
3.1.2 SumTable 模块工作流程	2
3.1.3 stable_add_data()	2
3.1.4 stable_data_pretreatment()	2
3.1.5 stable_check_excel_exist()	3
3.1.6 stable_check_excel_open()	4
3.1.7 stable_add_data_in_open()	4
3.1.8 stable_add_data_in_close()	5
3.1.9 stable_get_hyperlink_path()	6
3.2 SumTableImport 模块	7
3.2.1 STI 模块使用	7
3.2.2 STI 模块工作流程	8
3.2.3 app_sti_import_start()	8
3.2.4 sti_tdr_resolver()	10
3.2.5 sti_import_excel()	11

第1章 总表介绍和使用说明

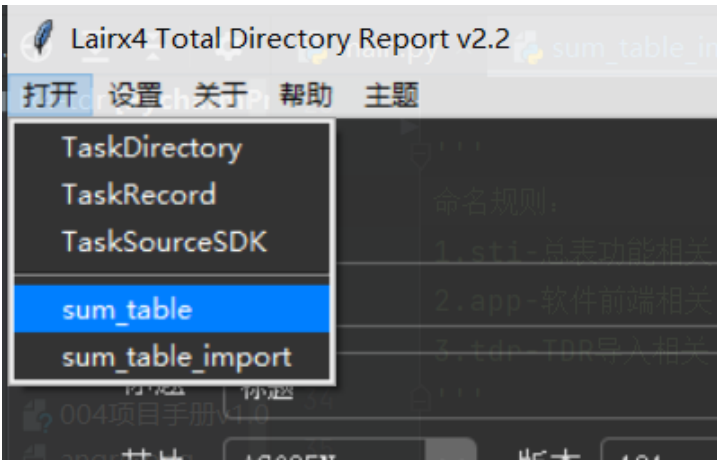
1.1 总表简介

总表（SumTable）记录任务记录单项数据信息，TDR 创建任务单项时可自动生成 excel 数据表格，进而可间接使用 excel 自带查找和筛选功能，同时设有超链接可快速跳转；

总表导入器（SumTableImport，简称 STI），可根据 TDR 文件结构和命名规范，解析出单项任务信息并导入总表

1.2 总表使用说明

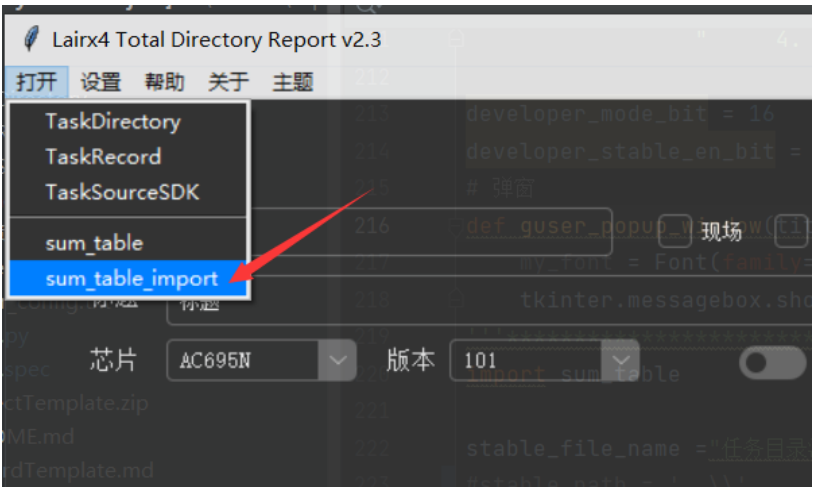
TDR 主界面任务栏，打开 sum_table:



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	TDR版本	序号	开始时	芯片系	SDK版本	客户	问题描述	现场	生产	共性	状态	结束时	备注
2	v2.2	4	20231109	AC695N	101	关闭测试	标题	N	N				
3	v2.2	5	20231109	AC695N	101	关闭测试	标题	N	N				

1.3 总表导入器使用说明

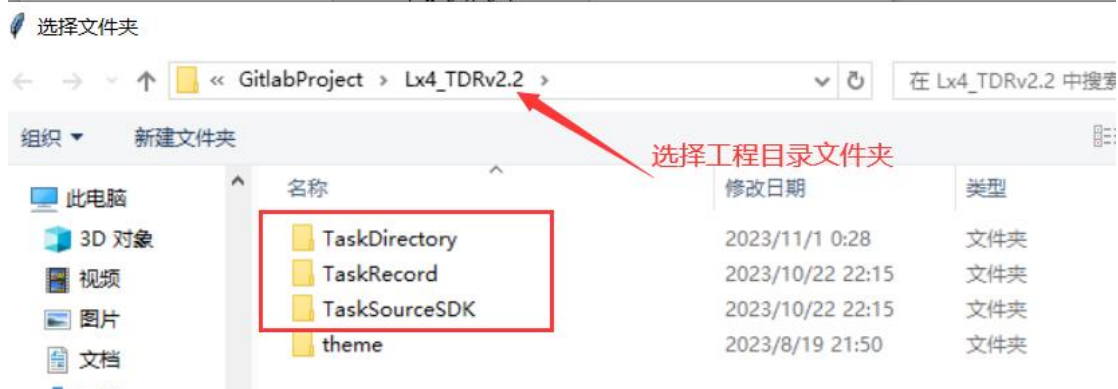
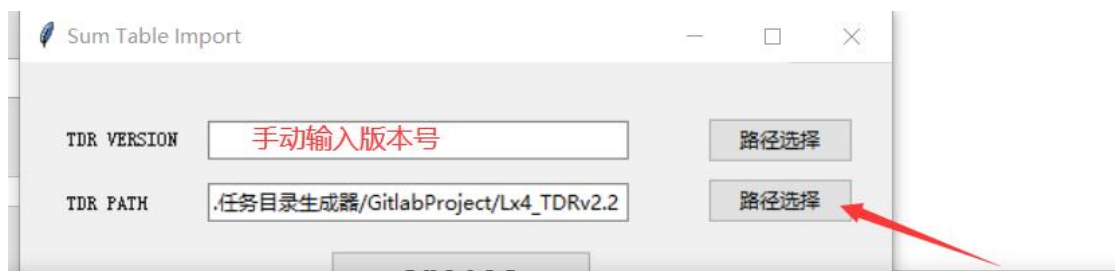
TDR 主界面任务栏，打开 sum_table_import;



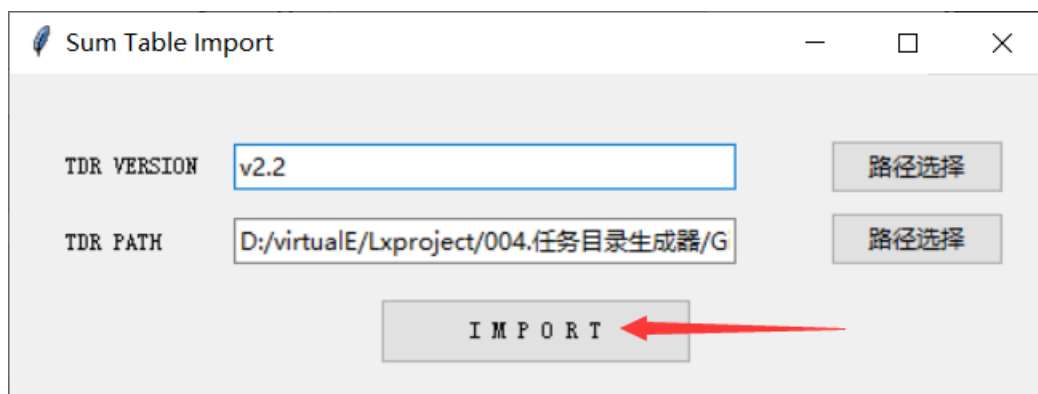
打开后界面如下：



手动输入版本号，点路径选择按钮选择 TDR 目录；



点 IMPORT 按钮开始导入；



第2章 相关库说明

2.1 xlwings 库

xlwings 是一个可在 excel 文件打开时操作的库;
xlwings 安装:

```
pip install xlwings
```

2.2 openpyxl 库

openpyxl 是一个可在 excel 文件关闭时操作的库;
openpyxl 安装:

```
pip install openpyxl
```

第3章 接口与功能实现

3.1 SumTable 模块

3.1.1 SumTable 模块使用

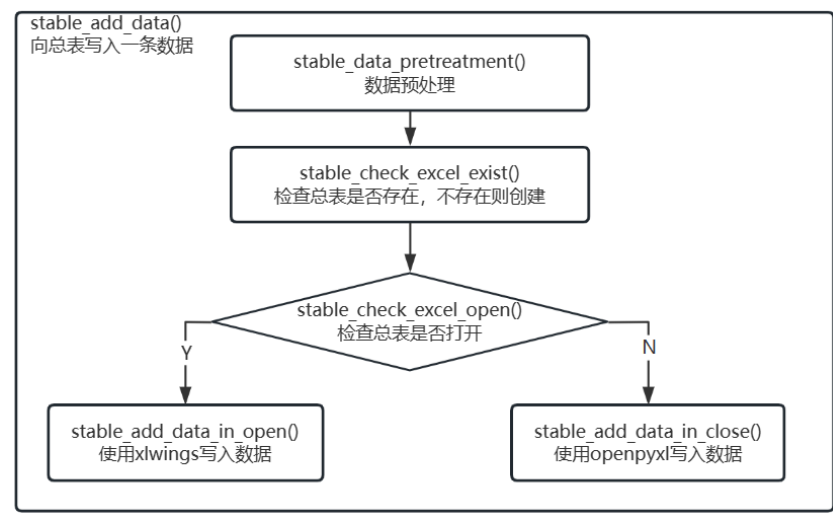
导入 sumtable 模块：import sum_table
调用写入接口：sum_table.stable_add_data()

函数原型	stable_add_data(datalist, file, sheet_name, hyperlink_str, hyperlink_path)
输入参数	datalist: 写入数据列表 file: 目标文件名 sheet_name: 目标工作表 hyperlink_str: 超链接文件名 hyperlink_path: 超链接路径设置, 若传"TDR_mode"或"dev_mode"则使用相对路径
返回参数	void
函数说明	向总表添加一行数据

```
'''***** 总表功能模块 *****'''
import sum_table

stable_file_name = "任务目录汇总表.xlsx"
#stable_path = '..\\' # 总表路径（相对路径，上一级目录）注意，sum_table.py也有该路径定义，需保持统一，建议使用sum_table.stable_path
stable_sheet_made0_name = "客户问题" # 客户问题日常支持模式
stable_sheet_made1_name = "内部测试" # 内部协助测试模式（ESO模式）
stable_add_list = ["TDR版本", "序号", "开始时间", "芯片系列", "SDK版本", "客户", "问题描述", "现场", "生产"]
# 用例，向stable_file_name文件的stable_sheet_made1_name表单最后一行写入stable_add_list列表的数据。
# 其中hyperlink_str为文件名字符串；hyperlink_input_path为路径字符串，若传入"TDR_mode"或"dev_mode"则使用相对路径
# sum_table.stable_add_data(stable_add_list, stable_file_name, stable_sheet_made1_name, hyperlink_str, hyperlink_input_path)
```


3.1.2 SumTable 模块工作流程



3.1.3 stable_add_data()

函数原型	stable_add_data(datalist, file, sheet_name, hyperlink_str, hyperlink_path)
输入参数	datalist: 写入数据列表 file: 目标文件名 sheet_name: 目标工作表 hyperlink_str: 超链接文件名 hyperlink_path: 超链接路径设置, 若传"TDR_mode"或"dev_mode"则使用相对路径
返回参数	void
函数说明	向总表添加一行数据

```
# excel添加一行新数据
def stable_add_data(datalist, file, sheet_name, hyperlink_str, hyperlink_path):
    print("[stable]:stable add data start\n")
    datalist = stable_data_pretreatment(datalist) # 数据预处理
    file_path = stable_path + file

    stable_check_excel_exist(file_path) # 检查总表是否存在, 不存在则创建
    if(stable_check_excel_open(file)): # 检查总表是否处于打开状态
        stable_add_data_in_open(datalist, file_path, sheet_name, hyperlink_str, hyperlink_path)
    else:
        # stable_add_data_in_open(datalist, file_path, sheet_name, hyperlink_str, hyperlink_path) # 兼容模式下, 强行打开excel输入
        stable_add_data_in_close(datalist, file_path, sheet_name, hyperlink_str, hyperlink_path)
    print("[stable]:stable add data succeed \n")
```

3.1.4 stable_data_pretreatment()

数据列表在写入 excel 前需要处理修改, 可在此函数进行预处理;

如 UI 传过来的现场和生产勾选框开关状态用“0”和“1”表示，excel 总表希望用“Y”和“N”表示，故需要通过预处理转换；

函数原型	stable_data_pretreatment(datalist)
输入参数	datalist: 预处理数据
返回参数	解析结果列表
函数说明	解析规范命名的字符串，获得对应列表

```
# 数据预处理
def stable_data_pretreatment(datalist):
    output_list = datalist
    # 打印TDR发过来的数据包类型
    if 0:
        print("stable_data_pretreatment:datalist type:\n")
        for temp_data in datalist:
            print(type(temp_data))

    # 判断是否现场
    if output_list[7] == '0' or output_list[7] == 'N':
        output_list[7] = 'N'
    elif output_list[7] == '1' or output_list[7] == 'Y':
        output_list[7] = 'Y'
    else:
        output_list[7] = '*'
    # 判断是否生产
    if output_list[8] == '0' or output_list[8] == 'N':
        output_list[8] = 'N'
    elif output_list[8] == '1' or output_list[8] == 'Y':
        output_list[8] = 'Y'
    else:
        output_list[8] = '*'
    return output_list
```

3.1.5 stable_check_excel_exist()

检查总表文件是否存在

函数原型	stable_check_excel_exist(file_path)
输入参数	file_path: 总表文件路径
返回参数	void
函数说明	检查总表 excel 文件是否存在，若不存在则创建

```
# 检查总表excel文件是否存在，若不存在则创建
def stable_check_excel_exist(file_name):
    if os.path.exists(file_name):
        print("[stable]:excel <%(file_name)s> is exist" % file_name)
    else:
        wb = openpyxl.Workbook() # 若excel不存在，此时必然不存在打开，使用<openpyxl>
        sheet = wb.active # 选择当前活动的工作表
        # 这里会有一个默认的sheet表，建议在这里添加使用说明。有空再写
        sheet.cell(row=1, column=1).value = "使用说明"

        mode0_sheet = wb.create_sheet(stable_sheet_made0_name) # 创建工作表
        mode0_sheet.append(mode0_list) # 设置标题栏
        mode1_sheet = wb.create_sheet(stable_sheet_made1_name) # 创建工作表
        mode1_sheet.append(mode1_list) # 设置标题栏
        wb.save(file_name)
        print('[warning] -> [stable]:no excel <%(file_name)s>,created excel successfully' % file_name)
```

3.1.6 stable_check_excel_open()

函数原型	stable_check_excel_open(file)
输入参数	file: 总表文件名
返回参数	1: 打开 0: 关闭
函数说明	检查总表 excel 文件是否打开(WPS 打开时也有临时文件，注意文件夹不要隐藏)

```
## 检查总表excel文件是否打开(WPS打开时也有临时文件，注意文件夹不要隐藏)
def stable_check_excel_open(file_name):
    temp_file = stable_temp_path + file_name # 打开excel的时候会出现前缀是~$的临时文件
    if os.path.exists(temp_file):
        print('[stable]:excel is open, use <xlwings>')
        return 1
    else:
        print('[stable]:excel is close, use <openpyxl>')
        return 0
```

3.1.7 stable_add_data_in_open()

函数原型	stable_add_data_in_open (datalist,file,sheet_name,hyperlink_str,hyperlink_path)
输入参数	datalist: 输入数据列表 file: 总表文件名 sheet_name: 总表工作簿名 hyperlink_str: 超链接文件名 hyperlink_path: 超链接路径设置，若传"TDR_mode"或"dev_mode"则使用相对路径
返回参数	void
函数说明	excel 文件打开条件下添加一行数据，使用 <xlwings>

```
132 # excel文件打开条件下添加一行数据，使用 <xlwings>
133 def stable_add_data_in_open(datalist,file,sheet_name,hyperlink_str,hyperlink_path):
134     wb = xw.Book(file) # 连接到已打开的 Excel 文件
135     # 获取工作表列表
136     num = len(wb.sheets) # 获取工作表个数
137     sheet_list = list(range(0, num))
138     for i in range(0, num): # 通过for循环把工作表名字一个个读出来写入列表
139         temp_sheet = wb.sheets[i]
140         sheet_list[i] = temp_sheet.name
141     print(sheet_list)
142     if sheet_name in sheet_list:
143         print("[stable]:sheet <{}> is exist\n" % sheet_name)
144     else:
145         wb.sheets.add(sheet_name) # 工作表不存在，新建工作表
146         print("[warning] -> [stable]:no sheet <{}>,xlwings created sheet successfully \n" % sheet_name)
147     sheet = wb.sheets[sheet_name] # 选择工作表
148     nrows = sheet.used_range.last_cell.row # 获取最大行数
149     str_s = 'A' + str(nrows+1) # 设置起始单元格，最后一行+1表示新增数据行
150     print("[stable]:write data list:{}".format(datalist))
151     sheet.range(str_s).value = datalist # 向单元格写入数据
152     new_row = nrows + 1

153
154 # 设置超链接
155 if sheet_name == stable_sheet_made0_name: # 日常支持模式
156     task_directory_hyperlink_path = stable_get_hyperlink_path(0, datalist, hyperlink_str,hyperlink_path) # 获取目录超链接地址
157     task_record_hyperlink_path = stable_get_hyperlink_path(1, datalist, hyperlink_str,hyperlink_path) # 获取记录超链接地址
158     # serial_text = str(datalist[1]) # 要转成字符串，format里不能传0 # 已确认TDR传过来的数据都是str
159     sheet.range(new_row, 7).value = '={HYPERLINK("{}","{}")}{}'.format(task_directory_hyperlink_path, datalist[6])
160     sheet.range(new_row, 2).value = '={HYPERLINK("{}","{}")}{}'.format(task_record_hyperlink_path, datalist[1])
161 elif sheet_name == stable_sheet_made1_name: # 内部协助模式，没有md记录文件，只需要设置一个超链接
162     task_directory_hyperlink_path = stable_get_hyperlink_path(0, datalist, hyperlink_str,hyperlink_path) # 合成超链接地址
163     sheet.range(new_row, 7).value = '={HYPERLINK("{}","{}")}{}'.format(task_directory_hyperlink_path, datalist[6])
164 else:
165     print("[error] -> [stable]:mode error,set hyperlink fail\n")
166     # print(sheet.range("A1").value) # 读取单元格数据
167     sheet.range(str_s).expand('right').api.HorizontalAlignment = -4131 # 设置在对齐，统一格式
168     wb.save() # 保存修改后的Excel文件
169
```

3.1.8 stable_add_data_in_close()

函数原型	stable_add_data_in_close (datalist,file,sheet_name,hyperlink_str,hyperlink_path)
输入参数	<p>datalist: 输入数据列表</p> <p>file: 总表文件名</p> <p>sheet_name: 总表工作簿名</p> <p>hyperlink_str: 超链接文件名</p> <p>hyperlink_path: 超链接路径设置，若传"TDR_mode"或"dev_mode"则使用相对路径</p>
返回参数	void
函数说明	excel 文件关闭条件下添加一行数据，使用 <openpyxl>

```

178 # excel文件关闭条件下添加一行数据, 使用 <openpyxl>
179 def stable_add_data_in_close(datalist, file, sheet_name, hyperlink_str, hyperlink_path):
180 |
181 |     wb = openpyxl.load_workbook(file) # 打开已存在的Excel文件
182 |     sheet_list = wb.sheetnames # 获取工作表列表
183 |     if sheet_name in sheet_list: # 检查工作表是否存在
184 |         print("[stable]:sheet <%(s)> is exist\n" % sheet_name)
185 |     else:
186 |         wb.create_sheet(sheet_name) # 工作表不存在, 新建工作表
187 |         print("[warning] -> [stable]:no sheet <%(s)>,openpyxl created sheet successfully \n" % sheet_name)
188 |         sheet = wb[sheet_name] # 选择工作表
189 |         last_row = sheet.max_row # 获取当前工作表的最后一行
190 |         new_row = last_row + 1
191 |         sheet.insert_rows(new_row) # 在指定位置插入新行, 最后一行+1表示新增数据行
192 |         print("[stable]:write data list:%(s)" % datalist)
193 |         sheet.append(datalist) # 设置新行的数据
194 |         # 设置超链接
195 |         if sheet_name == stable_sheet_made0_name: # 日常支持模式
196 |             task_directory_hyperlink_path = stable_get_hyperlink_path(0, datalist, hyperlink_str, hyperlink_path) # 获取目录超链接地址
197 |             task_record_hyperlink_path = stable_get_hyperlink_path(1, datalist, hyperlink_str, hyperlink_path) # 获取记录超链接地址
198 |             sheet.cell(row=new_row, column=2).value = 'HYPERLINK("{}","{}").format(task_directory_hyperlink_path, datalist[6]) # 以超链接方式写入表格单元
199 |             sheet.cell(row=new_row, column=7).value = 'HYPERLINK("{}","{}").format(task_record_hyperlink_path, datalist[1]) # 以超链接方式写入表格单元
200 |         elif sheet_name == stable_sheet_made1_name: # 内部协助模式, 没有md记录文件, 只需要设置一个超链接
201 |             task_directory_hyperlink_path = stable_get_hyperlink_path(0, datalist, hyperlink_str, hyperlink_path) # 合成超链接地址
202 |             sheet.cell(row=new_row, column=7).value = 'HYPERLINK("{}","{}").format(task_directory_hyperlink_path, datalist[6]) # 以超链接方式写入表格单元
203 |         else:
204 |             print("[error] -> [stable]:mode error,set hyperlink fail\n")
205 |
206 |     # 设置超链接字体和下划线
207 |     font = Font(
208 |         name=None, # 字体
209 |         size=11, # 字体大小
210 |         color="0563C1", # 字体颜色, 用16进制rgb表示
211 |         bold=False, # 是否加粗, True/False
212 |         italic=False, # 是否斜体, True/False
213 |         strike=None, # 是否使用删除线, True/False
214 |         underline='single', # 下划线, 可选'singleAccounting', 'double', 'single', 'doubleAccounting'
215 |     )
216 |     if sheet_name == stable_sheet_made0_name:
217 |         temp_cell = 'B' + str(new_row)
218 |         sheet[temp_cell].font = font
219 |         temp_cell = 'G' + str(new_row)
220 |         sheet[temp_cell].font = font
221 |
222 |     wb.save(file) # 保存修改后的Excel文件

```

3.1.9 stable_get_hyperlink_path()

为保证总表数据移动后仍可以使用访问 TDR, sum_table 驱动对 TDR 新创建的任务目录使用相对路径, 故需要确保工程目录文件夹命名规范为【Lx4_TDR】+版本;

对 STI (sum table import 总表导入器) 导入的旧版本数据, 由于相对路径可能较复杂, 故使用绝对路径, 若旧版本数据移动导致总表超链接访问失效, 重新使用 STI 导入即可
v2.2 版本后 TDR 目录结构和命名如下图:

```

总工程目录工具文件夹
-Lx4_TDRv1.0 (工程目录文件夹)
-Lx4_TDRv1.1
-Lx4_TDRv2.0
-Lx4_TDRv2.1
任务目录汇总表.xlsx

```

函数原型	stable_get_hyperlink_path(mode,datalist,hyperlink_str,hyperlink_input_path)
输入参数	mode:0 获取任务目录超链接, 1 获取任务记录超链接 datalist:写入数据列表 hyperlink_str: 超链接文件名 hyperlink_path: 超链接路径设置, 若传"TDR_mode"或"dev_mode"则使用相对路径
返回参数	返回超链接字符串
函数说明	获取超链接地址

```
# 获取超链接地址
def stable_get_hyperlink_path(mode,datalist,hyperlink_str,hyperlink_input_path):
    if hyperlink_input_path != "TDR_mode" and hyperlink_input_path != "dev_mode":
        hyperlink_path = "error:01"
        print("[error] -> [stable]:mode error:01,return hyperlink path fail\n")
    else:
        version_str = tdr_version_unify_name + datalist[0]
        if hyperlink_input_path == "dev_mode":
            version_str = "lx4-tdr" # 修改目录名, 测试用
        if mode == 0:
            hyperlink_path = version_str + "\\ " + task_directory_path + "\\ " + hyperlink_str
        elif mode == 1:
            hyperlink_path = version_str + "\\ " + task_record_path + "\\ " + hyperlink_str + ".md"
        else:
            hyperlink_path = "error:02"
            print("[error] -> [stable]:mode error:02,return hyperlink path fail\n")
    return hyperlink_path
```

3.2 SumTtableImport 模块

3.2.1 STI 模块使用

调用 sti_app_start()启动 STI, 注意, 启动 STI 后会关闭其他 tkinter 窗口

```
def sti_app_start():
    sti_top = Tk()
    Application2(sti_top).mainloop()
    try: sti_top.destroy()
    except: pass
```

3.2.2 STI 模块工作流程

```
# 总表导入启动
# 1.有效性检测，把TDR文件夹path获取下来
# 2.获取TDR列表
# 3.解析器：解析TDR单项
# 4.导入器：写入excel总表
# 5.更新UI
# 6.TDR列表导入未结束，回到3循环
```

3.2.3 app_sti_import_start()

前端按键启动该函数后，从 UI 中获取版本信息和地址，开始启动 STI 导入 TDR 数据

函数原型	app_sti_import_start(self)
输入参数	void
返回参数	void
函数说明	开始 STI 导入

```
196 def app_sti_import_start(self):
197     global sti_input_version
198     global sti_input_path
199     global tdr_directory_exist
200     global tdr_record_exist
201     global import_data_cnt
202     global import_is_run
203     if import_is_run == 1: # 若STI正在导入中，按import无效
204         print("[STI]:sti is busy, import key close\n")
205         return
206     else:
207         import_is_run = 1
208     # 检查用户输入信息
209     sti_input_version = self.entry1Var.get() # 获取客户输入的TDR版本
210     sti_input_path = self.entry2Var.get() # 获取客户输入的TDR路径
211     if sti_input_version == "": # 检查用户输入TDR版本号是否为空
212         print("[error] -> [STI]:TDR version is null\n")
213         sti_popup_window(app_version_errpop_title, app_version_errpop_show)
214         return
215     elif sti_input_path == "": # 检查用户输入TDR路径是否为空
216         print("[error] -> [STI]:TDR path is null\n")
217         sti_popup_window(app_path_errpop_title, app_path_errpop_show)
218         return
219
```

```

220     # 检查TDR路径有效性, directory和record至少存在一个
221     tdr_directory_path = sti_input_path + '\\ ' + tdr_directory_name
222     tdr_record_path = sti_input_path + '\\ ' + tdr_record_name
223     if os.path.exists(tdr_directory_path) == 1:          # 检查TDR目录文件夹是否已存在
224         tdr_directory_exist = 1
225         print("[STI]:TDR directory path exist\n")
226     if os.path.exists(tdr_record_path) == 1:          # 检查TDR记录文件夹是否已存在
227         tdr_record_exist = 1
228         print("[STI]:TDR record path exist\n")
229     if tdr_directory_exist or tdr_record_exist:        # 目录一个都不存在, 路径无效
230         print("[STI]:sti start success\n")
231     else:
232         print("[error] -> [STI]:TDR path is invalid\n")
233         sti_popup_window(app_path_errpop_title, app_path_errpop_show)
234     return
235

```

```

236     # 获取列表
237     import_data_cnt = 0
238     if tdr_directory_exist == 1: # directory目录存在
239         directory_list = os.listdir(tdr_directory_path)
240         for i in range(len(directory_list)):
241             import_data_list = sti_tdr_resolver(directory_list[i])
242             if import_data_list == 0: # 解析文件失败, 跳过本次循环
243                 continue
244         # 写入excel
245         import_data_cnt = import_data_cnt + 1
246         self.app_import_update_entry(directory_list[i])
247         if "0x" in import_data_list[1]:
248             sti_import_excel(import_data_list, '1', directory_list[i])
249         else:
250             sti_import_excel(import_data_list, '0', directory_list[i])
251     elif tdr_record_exist == 1: # directory目录不存在, record目录存在
252         print("[STI]:not [directory] file, use [record] file\n")
253         record_list = os.listdir(tdr_record_path)
254         for i in range(len(record_list)):
255             if '.md' not in record_list[i]: # 使用record目录导入时, 过滤非md文件
256                 continue
257             import_data_list = sti_tdr_resolver(record_list[i])
258             if import_data_list == 0: # 解析文件失败, 跳过本次循环
259                 continue

```

```

260         # 写入excel
261         record_list[i] = record_list[i][:-3] # 去掉.md字符
262         import_data_cnt = import_data_cnt + 1
263         self.app_import_update_entry(record_list[i])
264         if "0x" in import_data_list[1]:
265             sti_import_excel(import_data_list, '1', record_list[i])
266         else:
267             sti_import_excel(import_data_list, '0', record_list[i])
268     import_is_run = 0
269     self.app_update_success_entry(import_data_cnt) # 更新导入完成ui
270
271     return
272

```


3.2.4 sti_tdr_resolver()

解析 TDR 规范命名，判断有效性并获取任务记录信息

函数原型	sti_tdr_resolver(input_str)
输入参数	input_str: TDR 规范命名字符串
返回参数	0: 无效文件 output_list: 有效文件信息列表
函数说明	TDR 解析器

output_list 格式:

["TDR 版本", "序号", "开始时间", "芯片系列", "SDK 版本", "客户", "问题描述", "现场", "生产"]

```
69 # TDR解析器
70 def sti_tdr_resolver(input_str):
71     # 输入str, 拆分各个excel单元格单项, 返回一个列表
72     counter = 0
73     input_str_len = len(input_str)
74     output_serial = "" # 序号
75     output_time = "" # 开始时间
76     output_series = "" # 芯片系列
77     output_version = "" # SDK版本
78     output_client = "" # 客户
79     output_title = "" # 问题
80     output_scene = "N" # 现场
81     output_production = "N" # 生产
82
83     output_list = ["TDR版本", "序号", "开始时间", "芯片系列", "SDK版本", "客户", "问题描述", "现场", "生产"]
84
85     # 获取序号
86     if input_str[:2] == "0x": # 内部模式
87         output_serial = "0x"
88         counter = 2
89     for i in range(counter, input_str_len):
90         if input_str[i] >= '0' and input_str[i] <= '9':
91             output_serial = output_serial + input_str[i]
92         elif input_str[i] == '.':
93             counter = i + 1
94             break
95         else:
96             sti_tdr_resolver_err(input_str)
97             return 0
98
99     # 获取时间, 现场, 生产
100     for i in range(counter, input_str_len):
101         if input_str[i] >= '0' and input_str[i] <= '9':
102             output_time = output_time + input_str[i]
103         elif input_str[i] == 'x':
104             output_scene = 'Y'
105         elif input_str[i] == 's':
106             output_production = 'Y'
107         elif input_str[i] == '_':
108             counter = i + 1
109             break
110         else:
111             sti_tdr_resolver_err(input_str)
112             return 0
```

```
112     # 获取芯片系列、SDK版本
113     version_flag = 0
114     for i in range(counter, input_str_len):
115         if input_str[i] == 'v':
116             version_flag = 1
117         elif input_str[i] != '_':
118             if version_flag == 0:
119                 output_series = output_series + input_str[i]
120             else:
121                 output_version = output_version + input_str[i]
122         else:
123             counter = i + 1
124             break
125     if i == input_str_len:
126         sti_tdr_resolver_err(input_str)
127     return 0

128     # 获取客户
129     for i in range(counter, input_str_len):
130         if input_str[i] != '_':
131             output_client = output_client + input_str[i]
132         else:
133             counter = i + 1
134             break
135     if i == input_str_len:
136         sti_tdr_resolver_err(input_str)
137     return 0

138     # 获取任务标题
139     for i in range(counter, input_str_len):
140         output_title = output_title + input_str[i]
141     if ".md" in output_title:
142         output_title = output_title[:-3]

143
144     output_list[0] = sti_input_version
145     output_list[1] = output_serial
146     output_list[2] = output_time
147     output_list[3] = output_series
148     output_list[4] = output_version
149     output_list[5] = output_client
150     output_list[6] = output_title
151     output_list[7] = output_scene
152     output_list[8] = output_production
153     # print(output_list)
154     return output_list
```

3.2.5 sti_import_excel()

excel 导入器，实际调用 SumTable 模块实现

函数原型	sti_import_excel(import_data, mode, hyperlink)
输入参数	import_data: 写入 excel 的列表 mode: 0 日常支持, 1 内部测试 hyperlink: 超链接文件名
返回参数	void
函数说明	Excel 导入器

```
158 # Excel导入数据
159 def sti_import_excel(import_data, mode, hyperlink):
160     # 向总表添加一行新数据，模式：0日常支持，1内部测试
161     add_list = import_data # 获写入总表的数据
162     file_name = sti_table_name
163     if (mode == '0'):
164         sheet_made_name = sti_sheet_made0_name
165     elif (mode == '1'):
166         sheet_made_name = sti_sheet_made1_name
167     print("import:%s" % hyperlink)
168     hyperlink_path = sti_input_path
169     sum_table.stable_add_data(add_list, file_name, sheet_made_name, hyperlink, hyperlink_path)
170     return
```