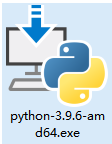
刷用例的脚本的编写流程

## 准备Python环境

1、从Python.org下载python3的版本，如3.9

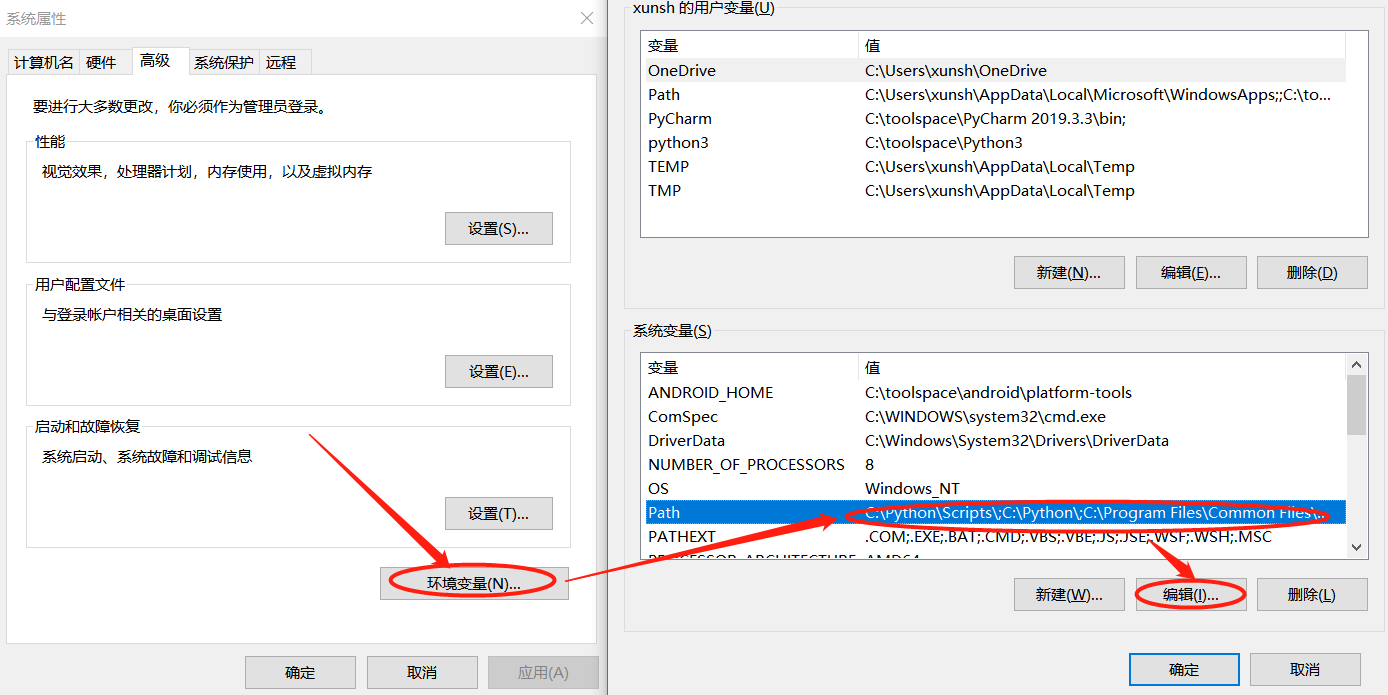
下载链接：https://www.python.org/downloads/

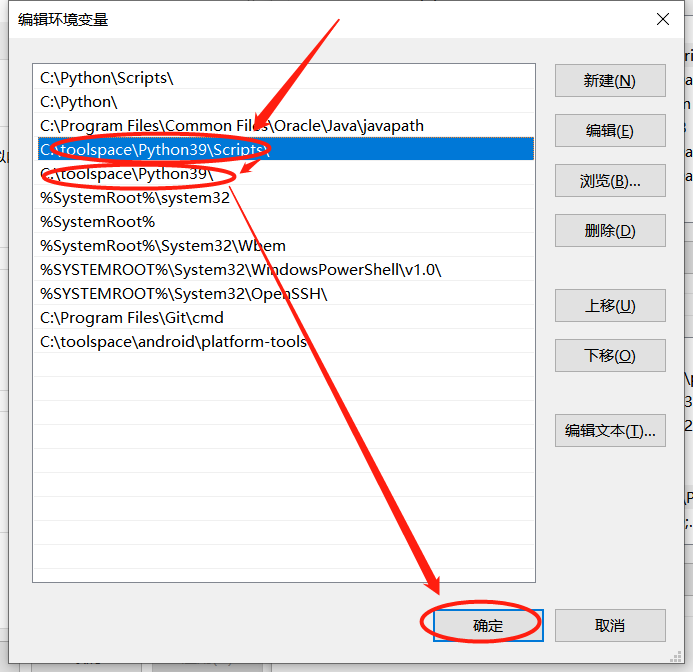


下载后双击安装选择Customize installation> 直接Next> 勾选install for all user> 设置目录为C:\Python39进行安装

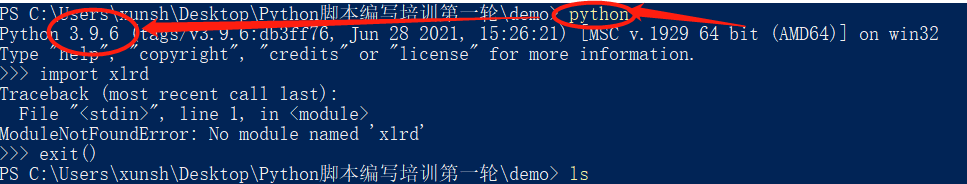
2、系统变量设置

Win+S 搜索环境变量> 设置环境变量

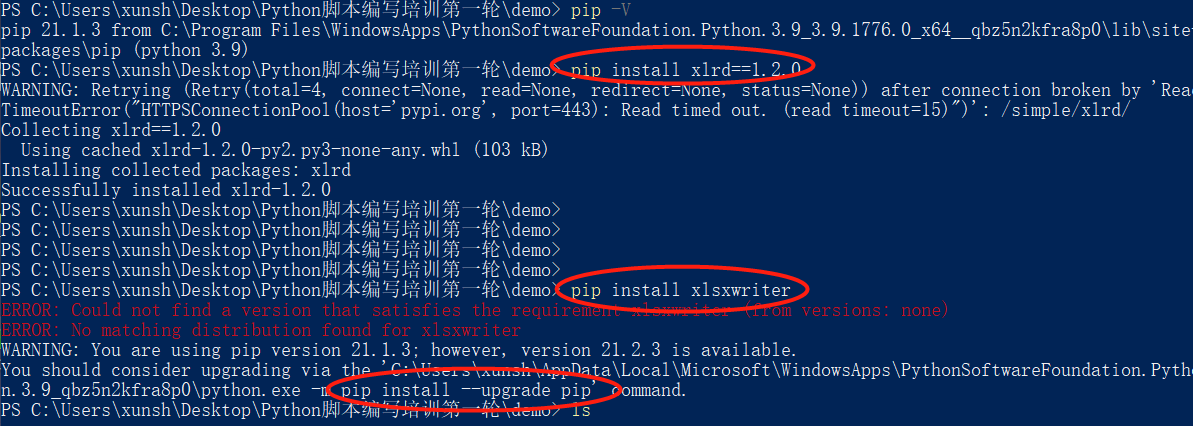


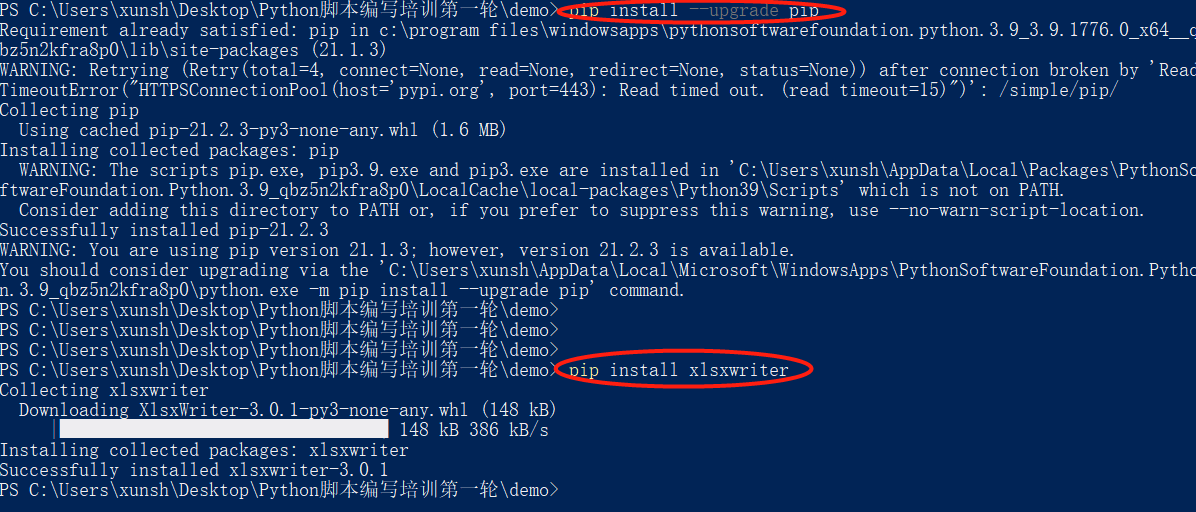


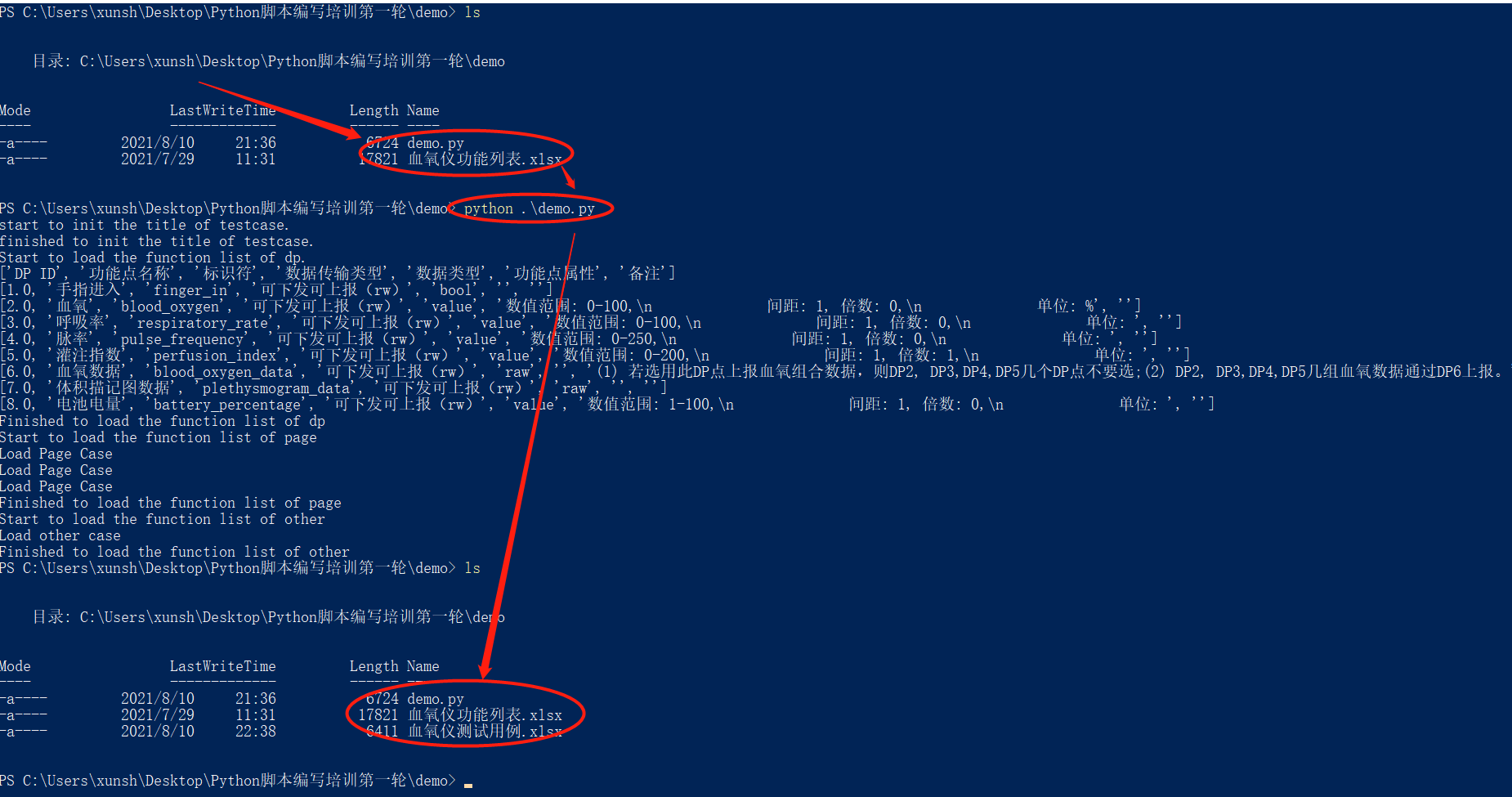
3、确认环境变量是否设置正确



4、pip安装模块



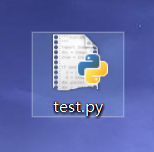




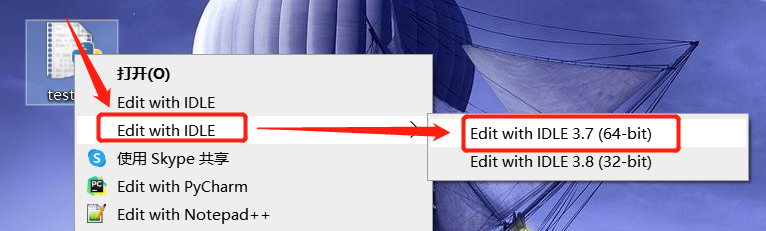
5、将IDLE创建快捷方式



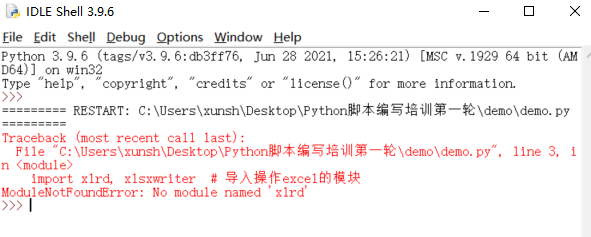
（2）修改文件的名称和后缀



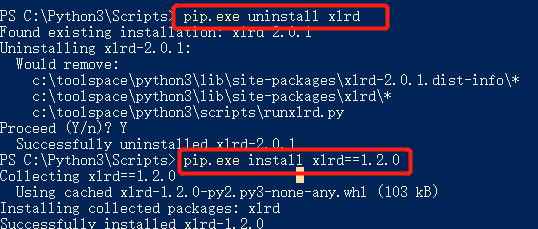
（3）用编译器打开文件



运行脚本文件，提示



（4）pip安装模块





（4）最基本的文件读写功能

|  |
| --- |
| # -\*- coding:utf-8 -\*- # 设置文档的编码格式  import xlrd, xlsxwriter # 导入操作excel的模块  # 创建测试类  class TestProject:  def \_\_init\_\_(self):  self.title = [u"标题\*", u"目录层级", u"前置条件", u"步骤描述", u"期望结果", u"标签", u"优先级", u"关联需求编号", u"备注"]  self.column = ["A", "B", "C", "D", "E", "F", "G", "H", "I"]  self.proLine = u"小家电"  self.product = u"血氧仪"  self.funcList = [u"DP校验", u"主页界面"]  self.pages = [u"首页", u"数据", u"设置"]  self.lineW = 1  self.xlsxInit() # 调用Excel初始化函数  self.caseTitle\_init() # 调用表格的初始化  # Excel输入文件和输出文件的目录初始化  def xlsxInit(self):  self.workbookR = xlrd.open\_workbook(self.product + u"功能列表.xlsx")  self.workbookW = xlsxwriter.Workbook(self.product + u"测试用例.xlsx")  self.worksheetW1 = self.workbookW.add\_worksheet(u"面板测试")  def xlsxClose(self):  self.workbookW.close()  # 输出表格的初始化  def caseTitle\_init(self):  print("start to init the title of testcase.")  redN = self.workbookW.add\_format({'border': 1, 'align': 'center', 'color': '#ff0000', 'bg\_color': '#9BC2E6', 'font\_name':'微软雅黑', 'font\_size': 11, 'bold': False})  self.comFormat\_left = self.workbookW.add\_format({'border': 1, 'align': 'left', 'valign': 'vcenter', 'bg\_color': '#EBEBEB', 'font\_name': '微软雅黑', 'font\_size': 11, 'text\_wrap': 1})  self.comFormat\_Mid = self.workbookW.add\_format({'border': 1, 'align': 'center', 'valign': 'vcenter', 'bg\_color': '#EBEBEB', 'font\_name': '微软雅黑', 'font\_size': 11, 'text\_wrap': 1})  # 设置列行的宽高  self.worksheetW1.set\_column("A:A", 30)  self.worksheetW1.set\_column("B:B", 30)  self.worksheetW1.set\_column("C:C", 18)  self.worksheetW1.set\_column("D:D", 35)  self.worksheetW1.set\_column("E:E", 35)  self.worksheetW1.set\_column("F:F", 18)  self.worksheetW1.set\_column("G:G", 10)  self.worksheetW1.set\_column("H:H", 10)  self.worksheetW1.set\_column("I:I", 10)  for (index, item) in zip(self.column, self.title):  self.worksheetW1.write(index + "1", item, redN)  print("finished to init the title of testcase.")  if \_\_name\_\_ == "\_\_main\_\_":  project\_fan = TestProject() # 实例化测试类  project\_fan.xlsxClose() # 调用用例加载函数 |

（2）测试类中补充函数

|  |
| --- |
| # 文本显示据中、据左的设置函数  def comWrite(self, row, col, contentDate):  if col in [6,]:  self.worksheetW1.write(row, col, contentDate, self.comFormat\_Mid)  else:  self.worksheetW1.write(row, col, contentDate, self.comFormat\_left)  # 连接符函数，默认"-"连接，也可以输入"|"来连接  def connector(self, \*key, constr="-"):  return constr.join(key)  # 加载测试用例  def loadCaselist(self):  # 写入功能校验用例  # 根据sheet索引或者名称获取sheet内容  sheet = self.workbookR.sheet\_by\_index(0)  print("Start to load the function list of dp.")  for i in range(sheet.nrows):  self.writeFuncCase(sheet.row\_values(i))  print("Finished to load the function list of dp")  # 写入页面校验用例  print("Start to load the function list of page")  for page in self.pages:  self.writePageCase(page)  print("Finished to load the function list of page")  # 写入其他测试用例  print("Start to load the function list of other")  self.writeOtherCase()  print("Finished to load the function list of other")  self.workbookW.close()  def writeFuncCase(self, dpInfo):  print(dpInfo)  dpid = str(dpInfo[0]).split(".")[0]  if dpInfo[4] == 'bool' and '可下发可上报' in dpInfo[3]:  self.comWrite(self.lineW, 0, self.connector(self.product, self.funcList[0], dpInfo[1]+"下发-开"))  self.comWrite(self.lineW, 1, self.connector(self.proLine, self.product, self.funcList[0], dpInfo[1], constr="|"))  self.comWrite(self.lineW, 2, "1、打开APP")  self.comWrite(self.lineW, 3, "1、打开" + dpInfo[1] + "按钮\n2、查看下发的dp信息")  self.comWrite(self.lineW, 4, "1、dpid为" + dpid + ",value为on")  self.comWrite(self.lineW, 5, self.product + "公版面板用例")  self.comWrite(self.lineW, 6, "P1")  self.lineW += 1  def writePageCase(self, pageInfo):  print("Load Page Case")  if pageInfo == u"首页":  self.comWrite(self.lineW, 0, self.connector(self.product, pageInfo + "界面", "文本显示校验"))  self.comWrite(self.lineW, 1, self.connector(self.proLine, self.product, pageInfo + "界面", "显示校验", constr="|"))  self.comWrite(self.lineW, 2, "1、打开APP\n2、进入" + pageInfo + "界面")  self.comWrite(self.lineW, 3, "1、查看设备名称显示是否正确\n2、查看内容文本是否为：XXXX\n3、底部菜单文本是否为：首页、数据、设置")  self.comWrite(self.lineW, 4, "1、界面文本显示正确")  self.comWrite(self.lineW, 5, self.product + "公版面板用例")  self.comWrite(self.lineW, 6, "P1")  self.lineW += 1  def writeOtherCase(self):  print("Load other case")  # 网页跳转  self.comWrite(self.lineW, 0, self.connector(self.product, "设置界面", "网页跳转入口校验"))  self.comWrite(self.lineW, 1, self.connector(self.proLine, self.product, "网页跳转", "入口校验", constr="|"))  self.comWrite(self.lineW, 2, "1、打开APP\n2、进入设置界面")  self.comWrite(self.lineW, 3, "1、在IoT平台上开启高级云功能中的跳转网页\n2、查看面板的设置界面中是否增加了网页跳转的入口\n3、在IoT平台上关闭高级云功能中的跳转网页\n4、查看面板的设备界面中的网页跳转的入口是否消失")  self.comWrite(self.lineW, 4, "1、开启跳转网页后，设置界面有网页跳转的入口显示\n2、关闭跳转网页后，设备界面没有网页跳转的入口显示")  self.comWrite(self.lineW, 5, self.product + "公版面板用例")  self.comWrite(self.lineW, 6, "P2")  self.lineW += 1 |

最简化的用例编写脚本

|  |
| --- |
| # -\*- coding:utf-8 -\*- # 设置文档的编码格式  import xlrd, xlsxwriter # 导入操作excel的模块  # 创建测试类  class TestProject:  def \_\_init\_\_(self):  self.title = [u"标题\*", u"目录层级", u"前置条件", u"步骤描述", u"期望结果", u"标签", u"优先级", u"关联需求编号", u"备注"]  self.column = ["A", "B", "C", "D", "E", "F", "G", "H", "I"]  self.proLine = u"小家电"  self.product = u"血氧仪"  self.funcList = [u"DP校验", u"主页界面"]  self.pages = [u"首页", u"数据", u"设置"]  self.lineW = 1  self.xlsxInit() # 调用Excel初始化函数  self.caseTitle\_init() # 调用表格的初始化  # Excel输入文件和输出文件的目录初始化  def xlsxInit(self):  self.workbookR = xlrd.open\_workbook(self.product + u"功能列表.xlsx")  self.workbookW = xlsxwriter.Workbook(self.product + u"测试用例.xlsx")  self.worksheetW1 = self.workbookW.add\_worksheet(u"面板测试")  def xlsxClose(self):  self.workbookW.close()  # 文本显示据中、据左的设置函数  def comWrite(self, row, col, contentDate):  if col in [6, ]:  self.worksheetW1.write(row, col, contentDate, self.comFormat\_Mid)  else:  self.worksheetW1.write(row, col, contentDate, self.comFormat\_left)  # 连接符函数，默认"-"连接，也可以输入"|"来连接  def connector(self, \*key, constr="-"):  return constr.join(key)  # 加载测试用例  def loadCaselist(self):  # 写入功能校验用例  # 根据sheet索引或者名称获取sheet内容  sheet = self.workbookR.sheet\_by\_index(0)  print("Start to load the function list of dp.")  for i in range(sheet.nrows):  self.writeFuncCase(sheet.row\_values(i))  print("Finished to load the function list of dp")  # 写入页面校验用例  print("Start to load the function list of page")  for page in self.pages:  self.writePageCase(page)  print("Finished to load the function list of page")  # 写入其他测试用例  print("Start to load the function list of other")  self.writeOtherCase()  print("Finished to load the function list of other")  self.workbookW.close()  def writeFuncCase(self, dpInfo):  print(dpInfo)  dpid = str(dpInfo[0]).split(".")[0]  if dpInfo[4] == 'bool' and '可下发可上报' in dpInfo[3]:  self.comWrite(self.lineW, 0, self.connector(self.product, self.funcList[0], dpInfo[1] + "下发-开"))  self.comWrite(self.lineW, 1,  self.connector(self.proLine, self.product, self.funcList[0], dpInfo[1], constr="|"))  self.comWrite(self.lineW, 2, "1、打开APP")  self.comWrite(self.lineW, 3, "1、打开" + dpInfo[1] + "按钮\n2、查看下发的dp信息")  self.comWrite(self.lineW, 4, "1、dpid为" + dpid + ",value为on")  self.comWrite(self.lineW, 5, self.product + "公版面板用例")  self.comWrite(self.lineW, 6, "P1")  self.lineW += 1  def writePageCase(self, pageInfo):  print("Load Page Case")  if pageInfo == u"首页":  self.comWrite(self.lineW, 0, self.connector(self.product, pageInfo + "界面", "文本显示校验"))  self.comWrite(self.lineW, 1,  self.connector(self.proLine, self.product, pageInfo + "界面", "显示校验", constr="|"))  self.comWrite(self.lineW, 2, "1、打开APP\n2、进入" + pageInfo + "界面")  self.comWrite(self.lineW, 3, "1、查看设备名称显示是否正确\n2、查看内容文本是否为：XXXX\n3、底部菜单文本是否为：首页、数据、设置")  self.comWrite(self.lineW, 4, "1、界面文本显示正确")  self.comWrite(self.lineW, 5, self.product + "公版面板用例")  self.comWrite(self.lineW, 6, "P1")  self.lineW += 1  def writeOtherCase(self):  print("Load other case")  # 网页跳转  self.comWrite(self.lineW, 0, self.connector(self.product, "设置界面", "网页跳转入口校验"))  self.comWrite(self.lineW, 1, self.connector(self.proLine, self.product, "网页跳转", "入口校验", constr="|"))  self.comWrite(self.lineW, 2, "1、打开APP\n2、进入设置界面")  self.comWrite(self.lineW, 3,  "1、在IoT平台上开启高级云功能中的跳转网页\n2、查看面板的设置界面中是否增加了网页跳转的入口\n3、在IoT平台上关闭高级云功能中的跳转网页\n4、查看面板的设备界面中的网页跳转的入口是否消失")  self.comWrite(self.lineW, 4, "1、开启跳转网页后，设置界面有网页跳转的入口显示\n2、关闭跳转网页后，设备界面没有网页跳转的入口显示")  self.comWrite(self.lineW, 5, self.product + "公版面板用例")  self.comWrite(self.lineW, 6, "P2")  self.lineW += 1  # 输出表格的初始化  def caseTitle\_init(self):  print("start to init the title of testcase.")  redN = self.workbookW.add\_format({'border': 1, 'align': 'center', 'color': '#ff0000', 'bg\_color': '#9BC2E6', 'font\_name':'微软雅黑', 'font\_size': 11, 'bold': False})  self.comFormat\_left = self.workbookW.add\_format({'border': 1, 'align': 'left', 'valign': 'vcenter', 'bg\_color': '#EBEBEB', 'font\_name': '微软雅黑', 'font\_size': 11, 'text\_wrap': 1})  self.comFormat\_Mid = self.workbookW.add\_format({'border': 1, 'align': 'center', 'valign': 'vcenter', 'bg\_color': '#EBEBEB', 'font\_name': '微软雅黑', 'font\_size': 11, 'text\_wrap': 1})  # 设置列行的宽高  self.worksheetW1.set\_column("A:A", 30)  self.worksheetW1.set\_column("B:B", 30)  self.worksheetW1.set\_column("C:C", 18)  self.worksheetW1.set\_column("D:D", 35)  self.worksheetW1.set\_column("E:E", 35)  self.worksheetW1.set\_column("F:F", 18)  self.worksheetW1.set\_column("G:G", 10)  self.worksheetW1.set\_column("H:H", 10)  self.worksheetW1.set\_column("I:I", 10)  for (index, item) in zip(self.column, self.title):  self.worksheetW1.write(index + "1", item, redN)  print("finished to init the title of testcase.")  if \_\_name\_\_ == "\_\_main\_\_":  project\_fan = TestProject() # 实例化测试类  project\_fan.loadCaselist() # 调用用例加载函数 |