

# LabElf 用户手册

---

实验室设备自动化控制系统 — 终端用户使用指南

适用版本: **LabElf 4.0** WPF 客户端 (LabElf.exe)

文档版本: 2026-04

注册、客服: [contacts@lab-elf.com](mailto:contacts@lab-elf.com)

---

# 目录

1. 软件简介 .....	6
1.1 术语速览 (详见 附录 C) .....	6
2. 系统要求 .....	8
2.1 硬件 / 操作系统 .....	8
2.2 外设 .....	8
2.3 网络端口 .....	8
3. 安装 .....	10
3.1 安装包 .....	10
3.2 Python (可选但推荐) .....	10
3.3 NI VISA (仅 GPIB 用户) .....	10
3.4 卸载 .....	10
4. 首次使用流程 .....	11
4.1 配置云端 GroupKey .....	11
4.2 注册 Web 端用户 (可选) .....	11
4.3 切换语言 .....	11
4.4 快速验证安装 .....	11
5. 主界面一览 .....	13
5.1 总体布局 .....	13
5.2 顶部导航与工具栏 .....	13
5.4 端口面板 (端口) .....	14
5.5 图表区 .....	14
5.6 日志面板 .....	15

6. 设置 (配置端口 / 设备)	16
6.1 三栏布局	16
6.2 快捷键	17
6.3 支持的端口类型 (完整表)	17
6.4 添加设备的每一步	19
6.5 IP 字段的扫描 / 下拉 (YouRenPort / TcpPort)	20
6.6 GatewayPort 下拉的动态范围	20
6.7 Serial Mode 下拉	20
6.8 驱动路径 DriverPath	21
7. YouRenPort (有人网关) 完整流程	22
7.1 硬件准备	22
7.2 常见接线示例	22
7.3 工作流程 A: 在 "设置" Tab 手动配置	22
7.4 工作流程 B: AI 助手全自动 (推荐)	23
7.5 AI Apply 后的 4 个步骤 (内部实现)	23
7.6 多设备共用一条 485 总线	24
7.7 修改网关 IP	24
8. ◆ AI 助手 — 特色功能	25
8.1 模型三档	25
8.2 协议三态	25
8.4 附件 📎	26
8.5 多轮对话与信息补齐	26
8.6 提案预览 + 应用	27
8.7 健壮性	27
8.8 典型工作流示例 (完整对话)	28

8.9 "AI 正在思考" 指示.....	29
9. 编写配方 & 运行实验 ("配方" Tab) .....	30
9.1 新建配方.....	30
9.2 行操作.....	30
9.3 保存 / 载入.....	31
9.4 运行.....	31
9.5 配方优化面板 (结构化优化) .....	31
10. 数据记录.....	33
10.1 保存数据 (本地 TXT) .....	33
10.2 OneNote 归档 (自动) .....	33
10.3 笔记标题 / 笔记 (双层标签) .....	33
10.4 云端历史 .....	33
10.5 Archive 两层设计 (进阶) .....	34
11. 远程访问.....	35
11.1 Web Panel (浏览器登录) .....	35
11.2 "Web 界面" 按钮 (本机直达) .....	35
11.3 LabElfPort (客户端互联) .....	35
12. Python 驱动完整开发指南.....	37
12.1 骨架 .....	37
12.2 可用 API.....	37
12.4 Modbus RTU 完整 MFC 示例.....	38
12.5 SCPI 简单字符串协议示例.....	39
12.6 Modbus TCP 示例 (带网络设备) .....	40
12.7 依赖自动安装.....	40
12.8 多 Python 版本切换 .....	40

12.9 脚本调试小贴士 .....	41
12.10 约束 .....	41
13. 内置驱动编辑器 .....	42
14. 界面个性化 .....	43
14.1 记忆设置 .....	43
14.2 主题 .....	43
14.3 语言切换 .....	43
14.4 锁定 .....	43
14.5 日志面板 .....	43
15. 常见问题与故障排查 .....	45
15.1 打开端口失败 .....	45
15.2 YouRenPort 相关 .....	45
15.3 云端连不上 .....	46
15.4 AI 助手 .....	46
15.5 Python 驱动 .....	46
15.6 图表卡顿 / 数据点太多 .....	47
15.7 "Web 界面" 按钮打开后 "GroupKey validation failed" .....	47
15.8 笔记 / 笔记标题保存报错 .....	47
15.9 Archive 数据丢失 / 覆盖 .....	47
15.10 打不开 .lab 文件 .....	48
16. 键盘快捷键 .....	48
16.1 全局 .....	48
16.2 ◆ AI 助手 .....	48
16.3 设置 (配置编辑器) .....	49
16.4 配方 .....	49

16.5 内置驱动编辑器 .....	49
17. 典型场景完整演练.....	51
17.1 场景 A: 接一个 RS-485 Alicat MFC (全 AI 驱动) .....	51
17.2 场景 B: 接一个 Modbus TCP PLC.....	51
17.3 场景 C: GPIB 万用表读电阻 .....	52
17.4 场景 D: 一条 Recipe 烧 CVD .....	52
18. 附录 A: XML 配置文件参考 .....	54
18.1 顶层结构 .....	54
18.2 Info 子节点.....	55
18.4 Control 节点格式.....	55
19. 附录 B: Recipe 表达式语法 .....	56
19.1 常量 .....	56
19.2 时间变量 t.....	56
19.3 支持的函数 .....	56
19.4 等待语法 .....	56
19.5 字符串控件 .....	56
20. 附录 C: 术语表.....	58
21. 获取帮助.....	59
21.1 诊断信息打包 .....	59

---

# 1. 软件简介

---

LabElf 是一款面向实验室的设备自动化管理软件，主要面向材料生长 (CVD / PVD / ALD)、表征、测试类仪器的统一控制。它把“端口 - 设备 - 控件 - 配方 - 数据”这一整套实验流程封装到一个桌面端 + 云端 + 浏览器的协同系统里。核心能力：

- **多协议端口统一接入**：串口 / TCP / UDP / GPIB / WebSocket / HTTP / LabElf-Link / VirtualPort, 外加有人 USR-N5xx 串口服务器
- **AI 助手**：自然语言添加或修改设备，读取用户上传的设备说明书 (PDF / 图片) 自动提取协议；对话式生成配置 XML + Python 驱动脚本 + 配方；并 **自动推送串口参数到有人网关硬件**
- **配方引擎**：顺序化、参数化、表达式化执行；支持等待条件 [wait] / 时间变量 t / 三角 / 指数等
- **实时数据采集与绘图**：ZedGraph 曲线，支持多组、多坐标轴
- **云端同步**：workspace 维度的配置、历史、评分、AI 优化状态都会双向同步到云
- **OneNote 自动归档**：按时间间隔把实验数据 + 图表快照写入 Microsoft OneNote 页面
- **浏览器远程监控**：Web Panel 支持另一台 PC 或手机实时查看图表、触发 Run
- **Python 脚本驱动 (PyDriver)**：接入任意第三方设备，支持热加载
- **双语 UI**：中文 / English 一键切换
- **LabElf-Link**：同 GroupKey 下两台 LabElf 客户端之间互联，典型场景是 A 机传感器 → B 机控制

## 1.1 术语速览 (详见 附录 C)

术语	含义
Workspace	一个 .lab / .xml 工程；一台仪器柜 / 一套装置通常对应一个 Workspace

术语	含义
<b>GroupKey</b>	课题组 / 组织的唯一 Key, 用于云端隔离
<b>Port</b>	物理接口 (串口 / TCP / ...) 抽象
<b>Device</b>	挂在某个 Port 上的仪器 (MFC / 温控仪 / 电机 / 继电器 / ...)
<b>Control</b>	设备上可读/可写的单个变量 (如 "flow"、"temp")
<b>Recipe</b>	配方, "配方" Tab 里一张按时间步排列的参数表
<b>Driver</b>	设备通信的协议实现, 可以是 C# 或 Python 脚本

## 2. 系统要求

---

### 2.1 硬件 / 操作系统

项	最低	推荐
操作系统	Windows 10 x64 1909	Windows 11 x64
.NET 运行时	.NET Framework 4.7.2	同
Python (PyDriver 用)	不装; 或 Python 3.7+ 64 位	Python 3.11 64 位
屏幕分辨率	1366×768	1920×1080 或以上
内存	4 GB	8 GB
CPU	双核 2 GHz	四核 3 GHz+
网络	有 Internet (Cloud / AI 功能) ; 纯局域网也可运行 (仅本地)	同

### 2.2 外设

- **USB 转串口:** FTDI / Silicon Labs / CH340 芯片均可, 装官方驱动
- **GPIB:** NI GPIB-USB-HS (需装 NI VISA)
- **有人网关:** USR-N510 / N520 / N540 / N580 (RS-485/232)
- **各类实验设备:** 见 §6.3 内置驱动列表

### 2.3 网络端口

方向	端口 / 协议	用途
出站	443 / HTTPS	云端 API、Microsoft Graph (OneNote)
出站	80 / HTTP	有人网关 HTTP API (读 Socket 端口号)
入站	48899 / UDP	有人网关 AT 命令回复
入站	用户指定	TCP / UDP Server 模式 (本机作 Server 时)

## 3. 安装

---

### 3.1 安装包

运行安装包 `LabElfSetup_v4.0.120_x64.exe` (或 x86) 。按向导步骤即可。默认安装路径 `C:\Program Files\LabElf\`。

首次启动一次性引导:

1. 选择 Workspace 保存位置 → 默认 `%UserProfile%\Documents\LabElf\`
2. 选择 `.lab` (推荐, 配置 + Python 驱动一包打) 或 `.xml` (仅配置)
3. 输入 Workspace 名 (合法字符: 英文 / 数字 / 下划线)

### 3.2 Python (可选但推荐)

如果会用 Python 脚本驱动:

1. 安装 [python.org](https://python.org) 64 位 Python 3.7+ (推荐 3.11)
2. **必须勾选 "Add Python to PATH"**
3. LabElf 首次加载 PyDriver 时会弹窗提示, 按提示装即可
4. 一份程序支持 3.7 ~ 3.13 任意版本; 装多个时默认用注册表里版本号最高的 64 位 Python

### 3.3 NI VISA (仅 GPIB 用户)

[ni.com/visa](https://ni.com/visa) 下载 NI VISA Runtime, 或 LabElf 自带 VISA 运行库。

### 3.4 卸载

开始 → 设置 → 应用 → "LabElf" → 卸载。会同时清理桌面快捷方式与注册表项; **不会删除 Workspace / Recipe 文件。**

---

## 4. 首次使用流程

---

### 4.1 配置云端 GroupKey

工具栏 **Cloud** 打开云端设置窗口：

字段	说明
Cloud location	云端区域 (Azure East-US)
GroupKey	团队 Key, 由课题组管理员发放
GroupCode	对应 GroupKey 的访问码

填好后点 **Set GroupKey**。连接成功后日志显示 Added / Updated。

没有 GroupKey? 联系课题组管理员或邮件 [contacts@lab-elf.com](mailto:contacts@lab-elf.com) 注册。

### 4.2 注册 Web 端用户 (可选)

"云端" 窗口下半部分 **Users** 区域：

1. 输入 User name、Password、Confirm (两次密码一致)
2. 点 **Sign Up**
3. 注册成功后可用这组账号在 Web 端 (§11.1) 登录

### 4.3 切换语言

标题栏右侧 **Help** → **Language** → **English / 中文**。切换会重启程序，配置不丢。

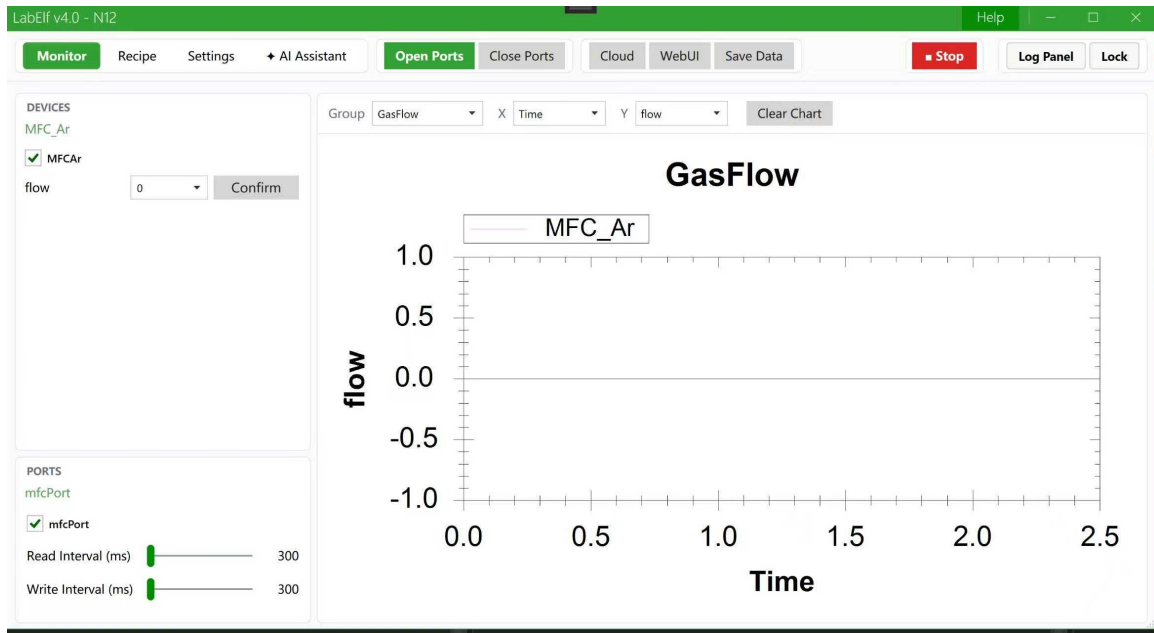
### 4.4 快速验证安装

1. 顶部 **设置** Tab → 左下端口列 **新建**
2. 新建 SerialPort, PortName = COM1 (或一个实际的 COM 口) , **启用** 勾上
3. 保存、切回 **监控** Tab

4. 工具栏 **打开端口**
5. 日志面板出现 COM1 opened 字样即安装无误 (COM 不存在会报 IOException, 也正常 — 说明流程通了)

## 5. 主界面一览

### 5.1 总体布局



### 5.2 顶部导航与工具栏

主窗口上半是四个导航 Tab：

Tab	说明
监控	主监控面板：设备 / 端口 / 图表 / 日志
配方	配方表格编辑与运行
设置	端口 / 设备 / 驱动 / 系统配置编辑器
◆ AI 助手	对话式 AI 配置助手

工具栏按钮（分组显示）：

分组	按钮	功能
连接	打开端口 / 关闭端口	启动 / 停止所有启用端口的轮询
模块	云端	云端 / 用户 / GroupKey 设置
	Web 界面	在默认浏览器打开本机 Web 面板
	保存数据	导出图表所有曲线为 TXT
视图	日志面板	切换底部日志条显示 (可折叠)
	锁定	锁定左侧配置面板, 防止误操作
紧急	■ 停止	立即中止当前实验运行

### ### 5.3 设备面板 (设备)

"监控" Tab 里的左上卡片 "设备"。选一个设备 Tab 展开, 显示它所有控件:

- **R 控件**: 只读, 显示实时值
- **W 控件**: 只写, 输入框 + Enter 确认
- **W/R 控件**: 双栏, 左显实时值、右可写

数值输入支持 §19 的完整配方表达式语法 (含 t、cos、exp 等)。

## 5.4 端口面板 (端口)

"监控" Tab 里的左下卡片 "端口"。每个端口一个 Tab:

- **启用 (复选框)**: 启用/禁用 (运行期动态切换, 不用重连)
- 协议字段 (波特率 / IP / ...) 只读, 要改去 "设置" Tab

## 5.5 图表区

右侧是 ZedGraph 图表。顶部下拉:

- **分组**: 按 "设置" 里 Device.Group 字段分组 (同组画到一张图)
- **X**: Time (默认) 或任何 R/W/R 控件

- **Y**: 任何 R/W/R 控件
- **清空图表** 按钮: 手动清空当前曲线

右键图表可缩放、保存 PNG。

## 5.6 日志面板

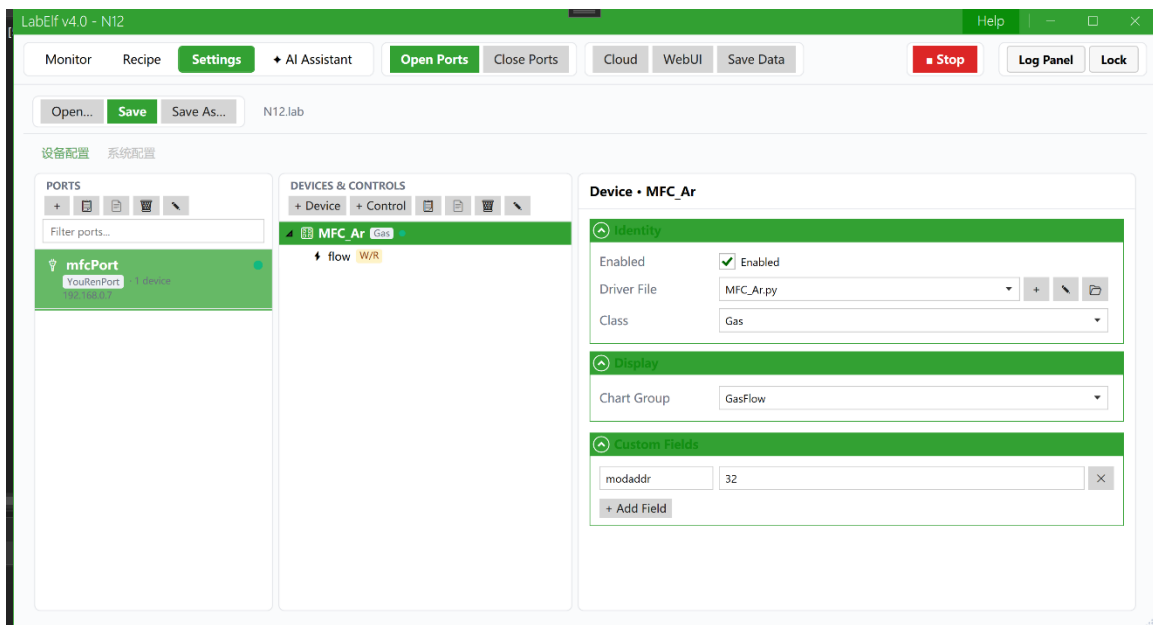
底部可折叠。右侧有 **清空** 按钮。显示:

- 端口打开/关闭事件
- AI 推送反馈 ("已推送 1 个网关...")
- Python 驱动 `print()` 输出
- C# 驱动 `ShowMsg` 事件
- 错误堆栈

## 6. 设置 (配置端口 / 设备)

顶部导航 **设置** Tab 打开 WPF 配置编辑器。三栏布局突出 **端口** → **设备** → **控件** 的一对多关系。

### 6.1 三栏布局



- **左列"端口"**: 端口卡片。每张显示 Type 徽章 + 启用点 + 挂载的设备数。顶部"过滤端口..."输入框可按名过滤。底部按钮: 新建 (Ctrl+N) / 复制 (Ctrl+C) / 粘贴 (Ctrl+V) / 删除 (Del) / 重命名 (F2)
- **中列"设备与控件"**: 设备树, 按端口自动聚合; 子节点是控件。△ **孤立项** 分组列出所有 Port 字段指向已不存在的孤儿设备。底部按钮: + **设备** / + **控件**
- **右列"属性"**: 属性面板。按端口 Type 动态切换字段集合; 字段按组折叠 (Connection / Gateway Config / Serial / Timing / Identity / ...)。底部有 + **添加字段** (加自定义字段)
- **Tab 页**:
- **设备配置** — 三栏主界面
- **系统配置** — 全局设置: Workspace、DriverPath、NoteBook、ChartPoints、ResultPoints、PythonHome

## 6.2 快捷键

按键	场景	动作
<b>F2</b>	端口 / 设备节点	重命名 (改端口名会级联更新所有引用它的 Device.Port 字段)
<b>Ctrl+N</b>	端口列	新建端口
<b>Ctrl+N</b>	设备列 (选中某个端口时)	新增设备 (挂在当前端口下)
<b>Ctrl+Alt+N</b>	选中某个设备时	新增控件
<b>Del</b>	任意节点	删除 (带确认)
<b>Ctrl+C / Ctrl+V</b>	节点	复制粘贴 (可跨端口粘贴设备)
<b>拖拽设备</b>	拖到另一个端口	改 Device.Port (迁移而非复制)
<b>拖拽控件</b>	拖到另一个设备	改归属

## 6.3 支持的端口类型 (完整表)

Type	适用场景	关键字段	备注
<b>SerialPort</b>	本地 RS-232 / RS-485 / USB 转串口	PortName, BaudRate, DataBits, Parity, StopBits	需物理串口或 USB-RS485 转接器

Type	适用场景	关键字段	备注
<b>YouRenPort</b> ★	有人 USR-N5xx 网关接入 RS-485/232	IP, GatewayPort, Username, Password, SerialMode, BaudRate/DataBits/Parity/StopBits	一个网关可 1/2/4/8 路串口
<b>TcpPort</b>	TCP 客户端	IP, Port	用于以太网接口设备
<b>SocketPort</b>	原始 TCP socket	IP, Port	低层应用
<b>UdpPort</b>	UDP 数据流	IP, Port	仅收包
<b>GPIB-USB</b>	NI GPIB-USB-HS 适配器	GPIBAddress	需 NI VISA Runtime
<b>WebSocketPort</b>	WebSocket 连接	IP, Port	某些新型设备
<b>WebrequestPort</b>	HTTP GET/POST 轮询	IP, Port, RequestMethod, ReceiveData, ReceiveAddress	接 REST API 设备
<b>LabElfPort</b>	客户端互联	ClientName (下拉选同组其它 workspace)	§11.3
<b>VirtualPort</b>	虚拟端口 (不真收发)	—	调试 / 纯软件模块

**共同字段** (所有端口) : Enable / Interval\_read\_min / Interval\_write\_min / Interval\_read / Interval\_write (读写间隔, 单位 ms, 默认 50/50/200/200) 。

## 6.4 添加设备的每一步

1. Ports 列选中目标 Port (如 yrPort1)
2. Devices 列 → **Ctrl+N** 或右键 → Add Device
3. 输入设备名 (英文数字下划线, 不能以数字开头), 如 MFC\_Ar
4. 右侧属性面板填:

字段	说明
Enable	是否启用
Port	挂在哪个 Port (此处应已经是 yrPort1; 拖拽可跨 Port 迁移)
Type	IOGeneral / MFC / Furnace / HVsource / Relay / SourceMeter / Multimeter / Fan / Rail_AB
Driver	驱动类名 (PyDriver37x64、MFC_alicat、relay_KMtronic 等)
Script	PyDriver 脚本路径 (仅 Python 驱动)。点  从文件系统选, 或点  从驱动库拉, 或点 + 新建
Class	自由分类标签 (DynamicCombo, 自动汇总已有值)
Group	图表分组 (同 Group 画到一张图)
业务字段	视 Driver 而定: addr (Modbus 地址)、to (邮件收件人)、max_flow、...

5. 设备节点下 **Ctrl+Alt+N** 新增控件:

字段	说明
Method	W (只写) / R (只读) / W/R (读写)
Expression	初始值或表达式 (见 §19)

## 6.5 IP 字段的扫描 / 下拉 (YouRenPort / TcpPort)

IP 字段右侧两个按钮:

- **[扫描]**: UDP 48899 广播发 `WWW.USR.CN`, 收集在线网关 IP + Model + MAC。下拉自动列出。
- **[修改 IP]**: 选已知 IP, 弹框让你改网关静态 WAN 口 (`AT+WANN`), 改完网关自动重启。

扫描结果由 "◆ AI 助手" 和 "设置" 共享: AI 里扫完, "设置" 下拉就能直接选; 反之亦然。扫描缓存是进程级 (关客户端就清)。

## 6.6 GatewayPort 下拉的动态范围

选好 IP 后 (AI 或 "设置" 任一路径), 软件按 Model 推断 GatewayPort 下拉范围:

Model	串口数
USR-N510	1
USR-N520	2
USR-N540	4
USR-N580	8

选中 GatewayPort 时立即触发 `AT+UART{N}` 查询, 回填 BaudRate/DataBits/Parity/StopBits。

## 6.7 Serial Mode 下拉

Model	可选
N580	485 (硬件只支持)
其它	485 / 232 / 422

**注意：** AT 协议无法远程切换 Serial Mode，必须：

- 网关硬件拨码开关（部分型号），或
- 网关 Web 设置页面（浏览器输入网关 IP，如 `http://192.168.0.7`，默认账号 `admin/admin`）

"设置" 保存时会弹提醒。

## 6.8 驱动路径 DriverPath

**系统配置** Tab → `DriverPath` 设一个本地文件夹（如 `C:\LabElf\Drivers`）。Python 脚本放进去，启动时自动加载。使用 `.lab` 文件时不用设，驱动直接打进 `.lab` 包。

---

## 7. YouRenPort (有人网关) 完整流程

---

### 7.1 硬件准备

有人 USR-N5xx 系列 (N510 / N520 / N540 / N580) :

1. **电源:** 9-24 V DC
2. **网口:** 网关 LAN 接电脑 / 交换机 / 路由器
3. **串口:**
  - N510: 1 路 RS-485/232/422 (15 针 DB15)
  - N520: 2 路
  - N540: 4 路 (RS-485 为主)
  - N580: 8 路 RS-485
1. **DIP 拨码:** 硬件上切换 485 / 232 / 422 模式 (**AT 协议改不了, 必须物理拨**)
2. **出厂 IP:** 192.168.0.7 / Mask 255.255.255.0 / Gateway 192.168.0.1
3. **出厂账号:** admin / admin

### 7.2 常见接线示例

设备	RS-485 接线	对应网关端脚 (N540 DB15)
Alicat MFC 9 针	DB9-5 = A+, DB9-6 = B-	A = 网关 Pin 5, B = Pin 4
Spellman HV	485 A / B / GND	同上 + GND
MTI 炉温控	485 A (+) / B (-) / GND	同上

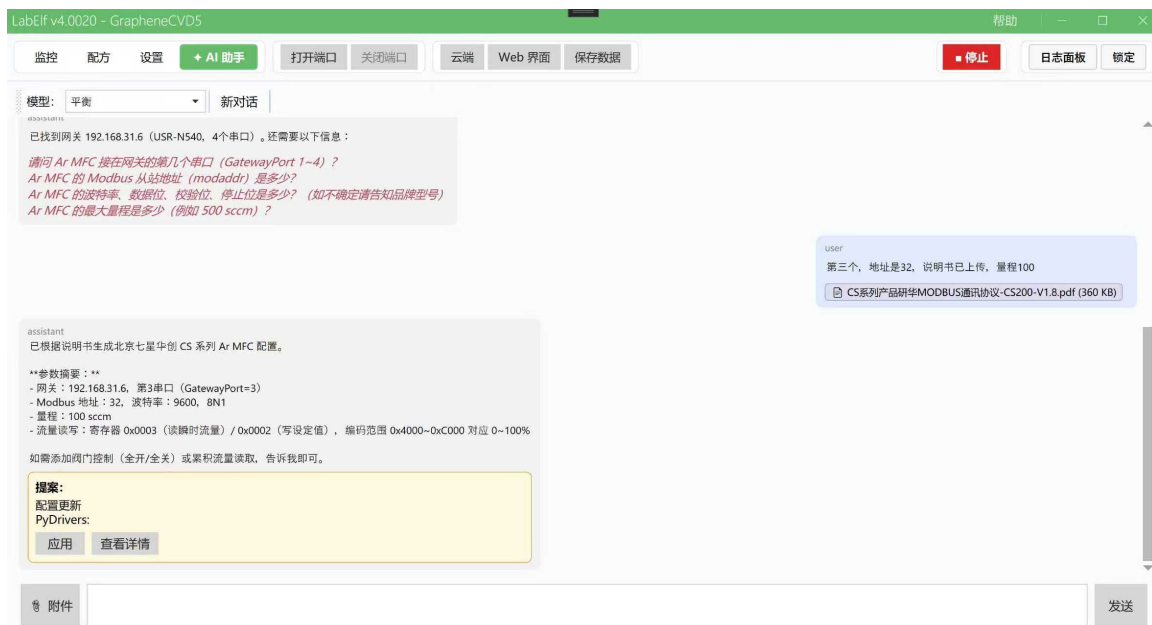
**绝大多数问题是 A/B 接反。**如果网关指示灯有 RX 闪烁但软件读不到数据, 先交换 A/B 两根线再试。

### 7.3 工作流程 A: 在 "设置" Tab 手动配置

1. 设置 Tab → 端口列 **新建** → 选 YouRenPort → 命名 yrGas
2. Gateway IP 字段 → 点 **扫描**
3. 下拉选到目标 IP (或手填) , SerialPort 下拉自动限制为 1-N (按 Model)
4. 选 GatewayPort (如 3) → 立即触发 AT+UART3 查询, BaudRate/DataBits/Parity/StopBits 回填实际值
5. 按需改 Baud / Parity 等 (若要让网关配合设备手册改波特率)
6. **Serial Mode** 下拉选 485 (或 232 / 422) , 保存弹提醒记得物理拨码
7. 保存 → 客户端 AT+UART3= + AT+Z 推到网关硬件生效

## 7.4 工作流程 B: AI 助手全自动 (推荐)

见 §8.8 的详细对话示例。总览:



## 7.5 AI Apply 后的 4 个步骤 (内部实现)

1. **快照**: 记录 Apply 前每个 YouRenPort 的 UART 字段和完整 XML 子树
2. **加载 AI XML**: `Info.InfoXml.LoadXml(proposal.Configuration)`
3. **Stub 保护**: 若 AI 漏写已存在的 YouRenPort, 从快照恢复 (防止丢失)
4. **同步填默认**: 每个 YouRenPort 缺的 Timing 和 SerialMode 填默认值; **落盘到文件**

5. **异步补全**: 对每个有 IP+GatewayPort 但 UART 空的 YouRenPort, 查 AT+UART{N} 回填; **再次落盘**
6. **按需推送**: UART 跟网关实际不一致才 AT+UART + AT+Z
7. **提醒**: 在日志面板弹出 ⚠ 网关 串口 →

这一套流程保证:

- 永远不会用随便的默认值 (9600,8,N,1) 覆盖用户的真实网关配置
- 用户打开"设置"看到的字段跟网关完全一致
- SerialMode 硬件拨码该提醒的都提醒

## 7.6 多设备共用一条 485 总线

多个 Modbus 设备接在同一条 RS-485 总线:

- 在 XML 里共享一个 YouRenPort (同一个 GatewayPort)
- 每个设备的 modaddr 不同 (1-247)
- Baud/Data/Parity/Stop 必须全总线统一 (485 硬件限制)

AI 理解这个规则, 会自动把同一总线上的多个设备聚到一个 YouRenPort 下; 不会为每个设备单独建 YouRenPort。

## 7.7 修改网关 IP

物理到手的新网关默认 IP 192.168.0.7, 实际部署要改到办公网段:

1. 电脑临时加一个 192.168.0.x 的静态 IP 别名 (或换网段)
2. 在"设置"或"AI 助手"里扫到 192.168.0.7
3. 点 **修改 IP** → 填新 IP (如 192.168.31.6)
4. 客户端发 AT+WANN=STATIC,,255.255.255.0, + AT+Z
5. 网关重启, 用新 IP 重新发现

---

## 8. ✦ AI 助手 — 特色功能

---

顶部导航 ✦ AI 助手 Tab 打开 AI 对话界面。

### 8.1 模型三档

AI 助手顶部工具栏 "模型：" 下拉，三档可选：

档位显示名	定价	速度	适用
快速	★	⚡ ⚡ ⚡	简单问答、基本配置、纯文本对话
平衡 (默认)	★★	⚡ ⚡	绝大多数场景
高质量	★★★	⚡	复杂说明书解读、多设备配方生成

切换不影响已发送消息，只作用于下一次发送。**新对话** 按钮清空会话重新开始。

### 8.2 协议三态

AI 按 `Status = chat / need_info / ready` 三态工作：

Status	含义	气泡样式
chat	纯聊天	普通 assistant 气泡
need_info	信息不够	下方列出问题列表，有交互动作时渲染按钮
ready	给出"提案 (Proposal) "，等你应用	提案卡片 (含 摘要 / 配置更新 / PyDrivers 数 / 配方更新)

### 8.3 交互动作：扫描网关

AI 判断需要有人网关 IP 时，会在气泡里渲染一组交互控件：

- **扫描网关**: UDP 48899 广播发现本网段所有有人网关
- 下拉框: 扫到的 IP + 型号 + 串口数
- **使用这个网关**: 客户端在配置里插入 YouRenPort stub (含 IP / Username / Password / SerialMode=485 / Timing 默认) → 自动给 AI 发一条系统消息告知 stub 已就位, AI 接着往下问设备细节

## 8.4 附件 📎

AI 助手输入栏左侧的 📎 **附件** 按钮 (也可直接把文件拖到聊天区):

- 支持: PDF (设备手册)、PNG / JPG / GIF / WebP (仪器照片)
- 单文件建议 < 10 MB; 硬上限每个 PDF 32 MB / 100 页, 每张图 5 MB
- AI 把附件当 **权威来源**: 寄存器地址、波特率、校验位优先从附件抽
- 附件在输入栏上方以小卡片显示文件名 + 大小, 右侧 × 按钮 (提示文字 "移除") 可撤销
- 历史消息里只保留附件 **标签** (文件名 + 大小), 不重复上传 base64 — 省带宽、避免对话包过大

## 8.5 多轮对话与信息补齐

The screenshot shows the LabElf v4.0020 - GrapheneCVD5 interface. At the top, there are tabs for '监控', '配方', '设置', '+ AI 助手', '打开端口', '关闭端口', '云端', 'Web 界面', and '保存数据'. Below the tabs, there's a '模型: 平衡' dropdown and a '新对话' button. The main chat area shows a conversation:

**assistant**  
好的, 我来帮您添加一个 Ar MFC 设备。在生成配置之前, 我需要确认几个参数。首先, 让我扫描网络上的人网关来确定 IP 地址。  
请点击下方按钮扫描网络中的人网关, 或手动输入网关 IP 地址。

**AI 需要你确认有人网关 IP。点击下面的按钮扫描本网段, 再从下拉里选一个, 或直接手动回复 IP。**

🔍 扫描网关 发现 13 个网关。  
192.168.31.6 (USR-N540, 4 ports) 使用这个网关

**USER**  
Selected gateway: 192.168.31.6 (USR-N540, 4 serial port(s))

**assistant**  
已找到网关 192.168.31.6 (USR-N540, 4 个串口)。还需要以下信息:  
请问 Ar MFC 接在网关的第几个串口 (GatewayPort 1~4) ?  
Ar MFC 的 Modbus 站地址 (modaddr) 是多少?  
Ar MFC 的波特率、数据位、校验位、停止位是多少? (如不确定请告知品牌型号)  
Ar MFC 的最大量程是多少 (例如 500 sccm) ?

**USER**  
第三个, 地址是 32, 说明书已上传, 量程 100  
CS系列产品研发MODBUS通讯协议-CS200-V1.8.pdf (360 KB)

At the bottom, there's an '附件' button and a '发送' button.

AI 会回避盲猜: 缺关键参数一定 need\_info, 不会瞎编。

## 8.6 提案预览 + 应用

提案卡片显示:

- **摘要 (Summary)** : 一句话总结本次要做的修改
- **配置更新**: 有 / 无 ("✓ 配置更新")
- **PyDrivers**: N 个 ("✓ PyDrivers: 2")
- **配方更新**: 有 / 无 ("✓ 配方更新")

两个按钮:

- **查看详情** — 纯文本对话框显示完整 XML / Python 源 / 配方 JSON, 只读、可复制
- **应用** — 确认后客户端按顺序执行:
  1. 保留本地 Workspace 名 (防和同组其它 workspace 撞名)
  2. 应用 AI 的 XML (`Info.InfoXml.LoadXml`)
  3. 恢复 AI 漏掉的 YouRenPort (stub 保护)
  4. 填 Timing / SerialMode 默认值
  5. 触发 "保存" 按钮 — 写 `.lab / .xml` 到盘
  6. PyDriver 写到 `%TEMP%\CVDclient_lab\drivers\` (打进 `.lab` 包)
  7. 异步查 AT+UART 回填空字段
  8. 按需 AT+UART + AT+Z 推到网关
  9. 配方上传到云端 Lab 任务队列

应用后按钮变成 ✓ **已应用**, 同一条提案不能重复应用。

"应用" 成功后弹 "是否要连线指导?":

- **是**: 自动发一条提示词给 AI, AI 返回针对你刚配置的设备的**具体引脚 / 线材 / 接地指南**
- **否**: 直接收工

## 8.7 健壮性

故障场景	行为
附件超大 / 格式错误	客户端 UI 拦截, 弹 "附件过大可能被拒"、"不支持的格式"

故障场景	行为
AI 回 JSON 含非法字符	客户端救援 Text 字段显示; 救援不到则显示 "AI 回复格式异常, 无法解析"
会话历史里有空 text (Anthropic 拒绝)	新客户端自动用 Str_Ai_AttachmentOnlyPrompt 占位, 避免后续消息连带 400
服务端超时 / 500	显示具体错误
AI 用量超额	显示剩余等待秒数

遇到死锁 (某条消息始终 400) 点 **新对话** 清历史, 通常能恢复。

## 8.8 典型工作流示例 (完整对话)

The screenshot shows the LabElf v4.0020 - GrapheneCVD5 interface. At the top, there are tabs for '监控', '配方', '设置', '+ AI 助手', '打开端口', '关闭端口', '云端', 'Web 界面', and '保存数据'. A '停止' button is visible on the right. Below the tabs, there's a '模型: 平衡' dropdown and a '新对话' button. The chat area shows an assistant message: '你好! 请描述要添加的设备 (例如"添加一个 Ar MFC"或"添加一个温控器")。如果是 RS-485 设备, 默认通过有人网关接入 — 告诉我使用哪个网关端口即可。如有设备手册, 附上即可 — 我会提取协议细节, 只询问缺失的信息。' A user bubble on the right says '添加一个 AI/MFC'. Below that, another assistant message says: '好的, 我来帮您添加一个 Ar MFC 设备。在生成配置之前, 我需要确认几个参数。首先, 让我扫描网络上的人网关来确认 IP 地址。' A blue box contains the text: 'AI 需要你确认有人网关 IP。点击下面的按钮扫描本网段, 再从下拉里选一个, 或直接手动回复 IP。' Below this is a '扫描网关' button that says '发现 13 个网关.' and a dropdown menu showing '192.168.31.6 (USR-N540, 4 ports)' with a '使用这个网关' button. At the bottom, there's an '附件' field and a '发送' button.



## 8.9 "AI 正在思考" 指示

点 **发送** 之后，对话流末尾出现一个灰色气泡 **••• AI 正在思考...**，三个点错峰跳动。AI 回包一到气泡自动消失。

---

## 9. 编写配方 & 运行实验 ("配方" Tab)

---

### 9.1 新建配方

1. 顶部导航 **配方** Tab 切过去
2. 底部按钮区点 **↓ 加载** 或 **📁 本地加载** 打开已有配方；新建直接在网格里编辑即可
3. 每一行代表一个时间步：

Minute	Second	MFC_Ar_flow	TC_furnace_temp	relay1_value	...
0	30	0	25	off	...
5	0	100	500	on	...
10	0	[wait]500	500	on	...
0	30	0	25	off	...

- Minute / Second: 该步持续时间。Minute = -1 表示无限长 (需手动停止)





- 列名格式: `_` (精确匹配配置里的命名)
- 每个单元格填常数、表达式、或 `[wait]...` 等待语法 (点 **⚡ 命令** 可插入模板)

### 9.2 行操作

按钮	动作
+ 添加行	在当前行下方插入空行

按钮	动作
- 删除行	删除当前行（带确认）
↶ 取消	撤销上一步（有限深度）
⚡ 命令	向当前选中单元格插入命令模板： <code>[wait]</code> / <code>[wait(tol)]</code> / <code>[wait(表达式)]</code> / <code>[if(...)]</code> 等

### 9.3 保存 / 载入

-  **全部保存** — 保存到云端（含 recipe / note / archive）
-  **本地保存** — 保存为本地 `.xml` 备份
-  **加载** — 从云端拉列表双击载入
-  **本地加载** — 从本地 `.xml` 导入

### 9.4 运行

1. 顶部工具栏 **打开端口** 先连硬件
2. 回到 **配方** Tab 选中要跑的配方 → 点 **运行**（按钮位置在配方执行器卡片上）
3. 运行中：
  - 图表实时更新
  - 当前行高亮
  - 倒计时显示该步剩余时间
1. 中途控制：
  - **Hold** — 暂停在当前行
  - **Modify** — 改当前行参数，实时生效
  - **Stop** — 中止
  - **Skip** — 跳过当前行

### 9.5 配方优化面板（结构化优化）

"配方" Tab 左下还有一个 **结构化优化** 面板（独立于 "◆ AI 助手" Tab，专门为 recipe 调优）。底部有一个蓝色的 **结构化优化** 按钮打开设置对话框：

字段	作用
AI Prompt	自然语言目标描述 ("让气流稳定后最高温达到 800°C")
AI 模型	快速 / 平衡 / 高质量
优化算法	Bayesian Opt / TPE / CMA-ES / Nelder-Mead / Random
复核策略	确认一次长期沿用 / 每次都要复核
评分来源	手打评分 / 自动评分 (表达式)
切换阈值	历史条数 < 阈值 时走 LLM 直生; ≥ 阈值 时走数学优化器

两个按钮：

- **生成配方** — 基于历史评分 + 算法生成更好的配方
- **查看详情** — 展开优化 Schema

**配方评分** 是下一次 "生成配方" 的依据。运行结束后可手打 0-100 分，或配自动评分表达式。

---

## 10. 数据记录

---

### 10.1 保存数据 (本地 TXT)

工具栏 **保存数据** 按钮导出图表所有曲线为制表符分隔 TXT (每列一个 设备名\_控件名)。文件保存到桌面或用户选的路径。

### 10.2 OneNote 归档 (自动)

**前置:** 云端 窗口首次登录 Microsoft 账号 (Graph API OAuth)。

**配置:**

1. **设置** → **系统配置** Tab → **NoteBook** — 从下拉选一个 OneNote 笔记本
2. **NoteTime** 设置周期 (单位分钟, 如 5 = 每 5 分钟写一页)

**运行:** 运行期间按周期自动写:

- 一页新 OneNote
- 嵌入图表 PNG
- 表格形式列出各控件实时值
- 顶部附配方当前步信息

### 10.3 笔记标题 / 笔记 (双层标签)

"配方" Tab 右侧:

- **笔记标题** (下拉) — "大类" 标签 (如 Exp-2026-04) ; 工具按钮 + **添加标题** / - **删除标题**
- **笔记** (下拉) — "细节" 标签 (如 sample\_A3\_run\_12) ; 工具按钮 + **添加笔记** / - **删除笔记**
- **笔记本** (可选) — OneNote section 名

保存 → 云端 workspace 关联。下次 ↓ **加载** 按标题 / 笔记检索所有历史运行。


### 10.4 云端历史

云端 窗口 → 控制台 → 发送 ToWeb 命令把当前运行的全量数据打包上传云端。之后 Web 界面任意位置检索。

## 10.5 Archive 两层设计 (进阶)

LabElf 的 Archive 存档分两层 (开发者关心, 普通用户不用管) :

- **配方级 Archive**: 绑定到具体配方, 由 "配方" Tab 的保存写入
- **workspace 级 Archive**: 跨配方, 含 DataGridFileName、AI 面板状态 (prompt / model / optimization) 等

 **全部保存** 按钮会先写配方级, 再调 SaveArchive() 写 workspace 级, 不会互相覆盖。

---

## 11. 远程访问

---

### 11.1 Web Panel (浏览器登录)

1. 已注册用户 ("云端" → Users) 打开:

`` https://www.lab-elf.com/\_Default?GroupKey= ``

1. 选 Workspace, 输入用户名、密码, **登录**
2. Panel 页面:
  - 左: 设备参数面板 + 配方表 + AI 面板 (Web 版功能子集)
  - 右: 实时图表 (通过 SignalR 推送)
  - 顶部可切 运行 / 暂停 / 停止

### 11.2 "Web 界面" 按钮 (本机直达)

工具栏 **Web 界面** 按钮会在默认浏览器打开本机当前 Workspace 的 Panel 页。

- **免登录**: 客户端已登录云端, 凭 GroupKey + GroupCode 做机器级认证换取一次性 URL
- 仅本机有效; 另一台机器打开同样 URL 进不去
- 适合同一台机器切桌面 / 浏览器快速查看图表

### 11.3 LabElfPort (客户端互联)

同 GroupKey 下两台 LabElf 客户端可以互联:

**场景**: A 机跑传感器采集, B 机跑控制算法, B 机要实时读 A 机的传感器值做反馈。

**步骤**:

1. B 机 "设置" Tab → 新增 LabElfPort 类型端口
2. ClientName 字段下拉 — 云端列出同 GroupKey 下其它 workspace 名 (排除本机)
3. 选中 A 机 workspace 名

4. 创建设备挂到这个 LabElfPort 上, Driver 用 PyDriver, 脚本里 `self.SendData(...)` 就发到 A 机
5. A 机同样建 LabElfPort 指回 B 机, 收 B 机的指令

**身份握手:** 用户端固定 `user = MD5(ClientName)`, 服务端 `CheckIdentification` 放行。  
无需交换密码。信任级别: 知道对方 workspace 名就能接入 (和浏览器登录同级)。

---

## 12. Python 驱动完整开发指南

---

### 12.1 骨架

```
# -*- coding: utf-8 -*-
import PyDriverCom

class pyDriver:
    def __init__(self, *args):
        pass

    def initialize(self):
        # 只在 "打开端口" 时调用一次
        # self.ParamList 含配置文件里本设备 + 控件的所有键值
        self.addr = int(self.ParamList["addr"])

    def end(self):
        # 只在 "关闭端口" 时调用一次, 用来释放资源
        pass

    def getValue(self, control):
        # 读: R / W/R 控件被触发读
        # 用 self.SendData(...) 发二进制或字符串给端口
        self.SendData(f":READ?{control}\n")

    def setValue(self, control, sv):
        # 写: W / W/R 控件被设置
        # sv 是字符串形式的目标值; 自行转换数值
        self.SendData(f":SET {control}={sv}\n")

    def receiver(self, rawdata):
        # 设备返回时回调, rawdata 是 bytes 或 str
        # 用 self.DataReturn(control, value) 把读数关联到 Control
        if rawdata.startswith("OK:"):
            self.DataReturn("flow", float(rawdata[3:]))
```

### 12.2 可用 API

属性 / 方法	作用
<code>self.ParamList[""]</code>	读配置文件里该设备 + 控件的字段 (字符串)

属性 / 方法	作用
<code>self.SendData(bytes or str)</code>	向挂载的端口发送数据
<code>self.DataReturn(control_name, value)</code>	把解析出的数值挂到某个控件（驱动图表 + 配方 Tab 显示）
<code>self.modbus_485 / self.modbus_tcp</code>	Modbus 辅助对象（见下）
<code>self.CrossRef / self.CrossCtrl</code>	跨设备引用 / 跨设备控制（高级，一般不用）
<code>print(...)</code>	输出到主界面日志面板

### ### 12.3 Modbus 辅助

```

from Drivers import ModBus_485, ModBus_TCP

# 读保持寄存器 0x0023, 长度 1
self.SendData(self.modbus_485.DataToModbus(
    ModBus_485.Code.Read, 0x0023, 1))

# 写寄存器 0x0012 = value (2 字节)
self.SendData(self.modbus_485.DataToModbus(
    ModBus_485.Code.Write, 0x0012,
    bytes([value // 256, value % 256])))

# 解析回包 (receiver 里)
data = self.modbus_485.ModbusToData(bytes(rawdata))
if len(data) >= 4:
    # data[2], data[3] = 16-bit 寄存器值
    val = (data[2] * 256 + data[3])
    self.DataReturn("flow", round(val / 10000 * self.max_flow, 2))

```

## 12.4 Modbus RTU 完整 MFC 示例

```

# -*- coding: utf-8 -*-
import PyDriverCom
from Drivers import ModBus_485

class pyDriver:
    def __init__(self, *args):
        pass

    def initialize(self):

```

```

self.addr = int(self.ParamList["modaddr"])
self.max_flow = int(self.ParamList["flow_max"]) # e.g. 500

def end(self):
    pass

def getValue(self, control):
    if control == "flow":
        # 读 0x0023, 1 寄存器
        self.SendData(self.modbus_485.DataToModbus(
            ModBus_485.Code.Read, 99 + self.addr, 1))

def setValue(self, control, sv):
    if control == "flow":
        v = max(0, min(int(sv), self.max_flow))
        # 换算成 0-10000 的 raw 值
        d = int(v / self.max_flow * 10000)
        self.SendData(self.modbus_485.DataToModbus(
            ModBus_485.Code.Write, 109 + self.addr,
            bytes([d // 256, d % 256])))

def receiver(self, rawdata):
    data = self.modbus_485.ModbusToData(bytes(rawdata))
    if len(data) >= 4:
        self.DataReturn("flow",
            round((data[2] * 256 + data[3]) / 10000 * self.max_flow, 2))

```

配合 "设置" Tab:

- Device: Type=MFC, Driver=PyDriver37x64, Script=alicat.py
- ParamList 字段: modaddr=3, flow\_max=500
- Control: flow, Method=W/R

## 12.5 SCPI 简单字符串协议示例

```

# -*- coding: utf-8 -*-
import PyDriverCom

class pyDriver:
    def __init__(self, *args):
        pass

    def initialize(self):
        pass

    def end(self):
        pass

    def getValue(self, control):
        self.SendData(f"READ? {control}\r\n")

```

```

def setValue(self, control, sv):
    self.SendData(f"SET {control} {sv}\r\n")

def receiver(self, rawdata):
    parts = str(rawdata).strip().split(" ")
    if len(parts) >= 2:
        self.DataReturn(parts[0], float(parts[1]))

```

## 12.6 Modbus TCP 示例 (带网络设备)

```

from Drivers import ModBus_TCP

class pyDriver:
    def initialize(self):
        self.addr = int(self.ParamList["modaddr"])

    def getValue(self, control):
        self.SendData(self.modbus_tcp.DataToModbus(
            ModBus_TCP.Code.Read, 99 + self.addr, 1))

    def setValue(self, control, sv):
        v = int(sv)
        self.SendData(self.modbus_tcp.DataToModbus(
            ModBus_TCP.Code.Write, 109 + self.addr,
            bytes([v // 256, v % 256])))

    def receiver(self, rawdata):
        data = self.modbus_tcp.ModbusToData(bytes(rawdata))
        if len(data) >= 4:
            self.DataReturn("val", data[2] * 256 + data[3])

```

## 12.7 依赖自动安装

PyDriver 扫描脚本里的 `import` 语句, 缺失的包自动 `pip install` (优先用所选 Python 的 `Scripts\pip.exe`, 没有则回退 `python -m pip`)。第一次打开端口可能要等几秒装包; 日志面板会显示 "Installing xxx..."。

## 12.8 多 Python 版本切换

默认用注册表 `Software\Python\PythonCore` 下版本号最高的 64 位 Python。

要锁版本: "设置" → **系统配置** Tab → 新建 `PythonHome` 字段 → 填安装目录 (如 `C:\Python311`)。

注意: 填的是 **安装根目录**, 不是 `python.exe` 路径。

## 12.9 脚本调试小贴士

- `print()` 立即输出到日志面板 — 最简单的调试手段
- 修改后保存脚本: PyDriver 自动 `importlib.reload`, 无需重启 LabElf
- 卡死 / 异常: 点 关闭端口 后在日志面板看栈
- **VirtualPort + Python print** 组合: 写业务逻辑不用插真硬件


## 12.10 约束

- 禁止用 `raw socket / raw serial`, 必须通过 `self.SendData` (它知道端口归属)
- 每个 `initialize / end / getValue / setValue / receiver` 里建议套 `try/except`, 防一个错误把整条轮询链卡死
- `DataReturn` 的 `control_name` 必须跟 "设置" 里的控件名精确匹配 (大小写敏感)

---

## 13. 内置驱动编辑器

---

"设置" Tab 里 Driver / Script 字段旁  按钮打开 **LabElf 内置驱动编辑器** (基于 AvalonEdit) :

- **语法高亮**: .py / .cs / .lua / plain text 自动识别
- **等宽字体** Consolas 13 + 行号
- **Ctrl+S** 保存
- Python 默认 Tab 自动转 4 空格
- UTF-8 无 BOM 写盘
- 编辑完保存 → PyDriver 自动热加载, 不用重启

替代了以前 "在默认编辑器打开" (以前 .py 双击默认是 python.exe 会直接执行黑窗一闪)。实验用机不配 VSCode / PyCharm 也能放心改驱动。

顶部工具栏:

- **Save / Save As / Reload** (从盘重读)
- **语法** 下拉: Python / C# / Lua / Plain

底部状态栏:

- 当前文件绝对路径
- 脏标 (\*) — 有未保存改动

关闭时若脏 → 弹 Yes/No/Cancel (保存 / 丢弃 / 取消关闭)。

---

## 14. 界面个性化

---

### 14.1 记忆设置

- **窗口大小 / 位置** — 关闭时记住，下次启动原位恢复
- **分割条位置** — 同上
- **最后一次 Workspace 路径** — 下次自动打开
- **语言** — 中 / 英

### 14.2 主题

默认 MahApps.Metro Light + Cobalt accent (工业软件"干净专业"调性)。需要改主题色联系技术支持 (要改 App.xaml 的 Light.Cobalt.xaml 引用)。

### 14.3 语言切换

**帮助** → **语言** → **English / 中文**。切换会重启程序。

覆盖范围：

- 全部按钮 / 菜单 / 标签
- 几乎所有对话框 / 提示消息
- "设置" 里的字段标签 + 分组名
- "◆ AI 助手" 的固定文案 (欢迎语、"AI 正在思考"、错误占位 ...)

协议字段 / 英文 Tag (如 forwhat、Claude Haiku、CVDRun) 不翻译。

### 14.4 锁定

工具栏 **锁定** 按钮锁定左侧设备 / 端口参数面板，防止运行过程中误改参数。锁定状态持久到下次启动。

### 14.5 日志面板

顶部工具栏 **日志面板** 按钮切换底部日志条显隐。日志内容不会截断 — 除非点右上 **清空** 按钮。

---

## 15. 常见问题与故障排查

---

### 15.1 打开端口失败

决策树:

打开端口 → 失败

- ├─ 看 "日志面板" 具体错误
  - ├─ "PortName is invalid / 拒绝访问" → SerialPort 占用或不存在
    - 设备管理器核对 COM 号 → 关掉占用它的软件
  - ├─ "SocketException: 10060 (连接超时)" → TCP 目标不通
    - ping 目标 IP → 防火墙 → telnet <IP> <port> 手测
  - ├─ "GPIB error" → NI VISA 未装或 GPIB 地址错
  - ├─ "Assembly not found" → DriverPath 设置错
  - └─ "No module named 'xxx'" → PyDriver 依赖装失败 → 手工 pip
- ├─ "设置" → PortInfo → <该端口> → 启用 是否勾上
- └─ 该端口下的设备 启用 / Driver 是否正确

### 15.2 YouRenPort 相关

#### 扫描不到网关

1. 电脑和网关在同一网段? (不同网段 UDP 单播回复被 Windows 丢弃)
  - 临时方案: 加一个同网段 IPv4 别名
  - 永久方案: 改网关 WAN IP (§7.7)
1. Windows 防火墙是否放行 **UDP 48899**?
2. 网关 AT+SEARCH 关键字是否被改过? 默认 WWW.USR.CN
3. 交换机 / 路由器禁用广播? 手填 IP 后点 **扫描** (单播兜底)
4. 网关固件 AT+BCAST=ON? 进 Web 管理页面关掉

#### AI 应用后 AT 推送失败

1. 看 "日志面板" 里 [诊断日志] 网关 段
2. → MAGIC "WWW.USR.CN" ... ← login 超时 → UDP 48899 被防火墙拦
3. +ERR1 / +ERR2 → 网关固件拒绝命令, 检查 Username/Password

4. 端口 N: AT+UART 写入失败 → AT 命令格式不对; 反馈给技术支持

### 网关改配置后没有立刻生效

AT+UART 命令需要 **AT+Z (重启)** 才生效。重启后 30 秒左右才恢复响应, 期间端口状态会短暂断开。

## 15.3 云端连不上

1. "云端" 窗口 Cloud location / GroupKey / GroupCode 是否正确
2. ping www.lab-elf.com 网络是否通
3. 防火墙 / 企业代理可能拦 HTTPS 出站 — 代理需配置白名单

## 15.4 AI 助手

### "AI usage is over the limit"

每个 GroupKey 有日用量上限 (每天 0 点 UTC 重置)。经常触顶 → 联系管理员升额度。

### 400 Bad Request / 附件发不出去

1. 附件是否 < 10 MB? (单 PDF 建议 < 5 MB; 大说明书先裁出协议章节)
2. 点 **新对话** 清对话历史重试
3. 仍不行 → 发 "日志面板" 截图给技术支持

### AI 回复是一坨 JSON

服务端 AI 返回的 JSON 含非法字符 (极少见)。最新客户端会自动救援 `Text` 字段; 若仍看到整坨 JSON, 复制内容邮件给技术支持。

### AI 生成的配置应用后部分字段空着

已在 4.0 修复。如果依然空: **帮助** → **关于** 确认客户端版本是否最新。

## 15.5 Python 驱动

## "找不到 PyDriverCom"

- PyDriverCom.py 必须在 LabElf 程序目录 (同 LabElf.exe) 或下
- 切换 Python 版本后需重启 LabElf
- PythonHome 填 Python 安装 **根目录**, 不是 python.exe 路径

ModuleNotFoundError: No module named ''

- pythonnet 3.x 的路径问题 — 客户端自动处理; 若还出现, 确认脚本 .py 文件真实存在在 DriverPath 或 .lab 解包目录
- .lab 解包目录: %TEMP%\CVDclient\_lab\drivers\

## pip 装不上

1. 网络出问题? pip install -U pip 手工试
2. 配置 pip 代理: %UserProfile%\pip\pip.ini 或 %AppData%\pip\pip.ini

## 15.6 图表卡顿 / 数据点太多

"设置" → 系统配置 Tab →

- ChartPoints — 图表显示最大点数 (默认 200)
- ResultPoints — 内存缓存上限 (默认 14000)

调小重启即可; 或用 **清空图表** 手动清理。

## 15.7 "Web 界面" 按钮打开后 "GroupKey validation failed"

客户端当前 GroupKey / GroupCode 和云端不匹配。重新打开 **云端** 窗口确认后 **Set GroupKey** 再试。

## 15.8 笔记 / 笔记标题保存报错

4.0 已修复: 新建标签时 notetitle\_id / note\_id 会正确发 -1 给云端。若仍出错, **帮助** → **关于** 检查 Cloud.dll 是否最新。

## 15.9 Archive 数据丢失 / 覆盖

Archive 分两层 (§10.5)。多个子系统之间通过 `CVD.ComposeArchive / ApplyArchive` 统一 — 写代码时不要直接发 `UpdateWorkspace Archive=` 请求。

## 15.10 打不开 .lab 文件

- 如果是从 3.x 版本带过来的老 .lab — 4.0 首次打开会弹升级对话框，按 Yes 升级（会备份为 .lab.bak）
- 如果弹 "Unable to unpack" — .lab 可能被之前版本的 bug 写成了纯 XML（不是 zip 结构）。解决：用 4.0 打开 .xml 同名文件（自动从纯 XML 模式重建）

---

## 16. 键盘快捷键

---

### 16.1 全局

场景	按键	动作
主界面	F5	刷新当前 Workspace（从盘重读）
主界面	Ctrl+S	保存 Workspace

### 16.2 ◆ AI 助手

按键	动作
Enter	发送
Shift+Enter	换行
Ctrl+V	粘贴（含图片：从剪贴板自动作为附件）

## 16.3 设置 (配置编辑器)

按键	动作
F2	重命名当前节点
Ctrl+N	新建端口 (端口列选中时) / 新增设备 (设备列选中端口时)
Ctrl+Alt+N	新增控件 (选中设备时)
Del	删除 (带确认)
Ctrl+C / Ctrl+V	复制 / 粘贴节点 (可跨端口)

## 16.4 配方

按键	动作
Ctrl+S	 全部保存
F5	运行
Space	暂停 / 恢复

## 16.5 内置驱动编辑器

按键	动作
Ctrl+S	保存并触发 PyDriver 热加载
Ctrl+F	搜索

按键	动作
Ctrl+H	替换
Ctrl+Z / Ctrl+Y	撤销 / 重做

## 17. 典型场景完整演练

### 17.1 场景 A: 接一个 RS-485 Alicat MFC (全 AI 驱动)

**前置:** 电脑能连 Internet; USR-N540 网关已通电接入同一交换机。

**步骤:**

1. 打开 LabElf → 新建 Workspace CVD\_01
2. 工具栏 **云端** → 填 GroupKey / GroupCode → **Set GroupKey**
3. 顶部 **AI 助手** Tab
4. 对话:

“你: 添加一个 Ar MFC, 量程 500 sccm, RS-485 AI: [🔍 扫描网关] [IP ▼] [使用这个网关] 你: [🔍 扫描网关] → [选 192.168.31.6] → [使用这个网关] 你: modaddr=3, port=3; 这是说明书: 📄 alicat.pdf AI: [提案卡片] 你: [应用]”

1. 应用成功 → 提醒确认 485 拨码 (物理拨网关 port 3 对应的 DIP)
2. 顶部 **设置** Tab → yrPort1 所有字段已填 (IP=192.168.31.6, GatewayPort=3, Baud=115200, ...)
3. 顶部 **监控** Tab → 设备面板看到 MFC\_Ar → flow 控件
4. 工具栏 **打开端口** → 日志显示 yrPort1 opened
5. flow 控件里输入 100 → Enter → 实际流量达到 100 sccm

### 17.2 场景 B: 接一个 Modbus TCP PLC

**前置:** PLC 已接以太网, IP=192.168.1.100, Modbus TCP 标准端口 502, modaddr=1。

**步骤:**

1. **设置** Tab → 端口列下方 **新建** → 选 TcpPort → 命名 plc\_tcp
2. 属性面板: IP=192.168.1.100, Port=502, **启用** 勾上
3. 设备列 + **设备** 挂到 plc\_tcp 下 → 命名 PLC1
4. Driver=PyDriver37x64, Script=modbus\_tcp.py
5. 属性面板 + **添加字段** 加 modaddr=1

6. 控件: temperature, Method=R
7. Driver 字段旁 打开驱动编辑器 → 粘贴 §12.6 的 Modbus TCP 代码 → Ctrl+S
8. 保存 → 工具栏 打开端口 → 监控 Tab 看 temperature 实时更新

### 17.3 场景 C: GPIB 万用表读电阻

前置: 装了 NI VISA; HP 34401 万用表 GPIB 地址 22。

步骤:

1. 设置 Tab → 端口列 新建 → GPIB-USB → 命名 gpib0
2. GPIBAddress=22 (是设备地址, 不是 Board 号; Board 号是 NI VISA 里的 GPIB0, 已默认)
3. 设备列 + 设备 → DMM
4. Type=Multimeter, Driver=HP34401 (内置 C# 驱动)
5. 控件: resistance, Method=R
6. 工具栏 打开端口 → 日志: GPIB address 22 opened
7. 监控 Tab 看 resistance 实时值

### 17.4 场景 D: 一条 Recipe 烧 CVD

前置: 已配好 MFC\_Ar / MFC\_H2 / TC\_furnace 三个设备。

Recipe CVD\_growth.xml:

Minute	Second	MFC_Ar_flow	MFC_H2_flow	TC_furnace_temp
0	30	0	0	25
10	0	100	0	400
0	0	100	0	[wait]400
60	0	100	50	800

Minute	Second	MFC_Ar_flow	MFC_H2_flow	TC_furnace_temp
120	0	100	50	800
30	0	100	0	400
0	0	100	0	[wait]300
10	0	0	0	25

解读：

- 30 秒 预热 (低流量)
- 10 分钟 升到 400°C
- 等温度到 400°C 再往下 ([wait]400)
- 1 小时升到 800°C, 过程中同时开 H2
- 2 小时 800°C 生长
- 30 分钟降到 400°C
- 等降到 300°C 以下 ([wait]300)
- 10 分钟降到室温, 关气

运行：打开端口 → 配方 Tab → 选 CVD\_growth → 运行

---

## 18. 附录 A: XML 配置文件参考

---

### 18.1 顶层结构

```
<?xml version="1.0" encoding="utf-8"?>
<SysInfo>
  <Info>
    <Workspace>CVD_01</Workspace>
    <IsOnline>True</IsOnline>
    <DriverPath>C:\LabElf\Drivers</DriverPath>
    <ChartPoints>200</ChartPoints>
    <ResultPoints>14000</ResultPoints>
    <NoteBook>Lab Notebook</NoteBook>
    <NoteTime>5</NoteTime>
    <PythonHome>C:\Python311</PythonHome>
    <_Workspace>abcd-1234-...-guid</_Workspace>
  </Info>
  <PortInfo>
    <yrPort1>
      <Type>YouRenPort</Type>
      <Enable>True</Enable>
      <IP>192.168.31.6</IP>
      <Username>admin</Username>
      <Password>admin</Password>
      <GatewayPort>3</GatewayPort>
      <SerialMode>485</SerialMode>
      <BaudRate>115200</BaudRate>
      <DataBits>8</DataBits>
      <Parity>None</Parity>
      <StopBits>One</StopBits>
      <Interval_read_min>50</Interval_read_min>
      <Interval_write_min>50</Interval_write_min>
      <Interval_read>200</Interval_read>
      <Interval_write>200</Interval_write>
    </yrPort1>
  </PortInfo>
  <DeviceInfo>
    <MFC_Ar>
      <Enable>True</Enable>
      <Port>yrPort1</Port>
      <Type>MFC</Type>
      <Driver>PyDriver37x64</Driver>
      <Script>alicat.py</Script>
      <Class>Gas</Class>
      <Group>Flows</Group>
      <modaddr>3</modaddr>
      <flow_max>500</flow_max>
      <flow type="control">
```

```

    <Method>W/R</Method>
    <Expression>0</Expression>
  </flow>
</MFC_Ar>
</DeviceInfo>
</SysInfo>

```

## 18.2 Info 子节点

节点	必填	含义
Workspace	是	workspace 名 (向导时指定)
IsOnline	否	True = 正常; False = 云端只读模式
DriverPath	否 (.lab 用不到)	本地 Python 驱动目录
ChartPoints	否	图表最大显示点数
ResultPoints	否	内存缓存上限
NoteBook	否	OneNote 笔记本名
NoteTime	否	OneNote 归档周期 (分钟)
PythonHome	否	锁定的 Python 安装目录
_Workspace	<b>自动生成</b>	workspace 唯一 GUID, 勿手改

### 18.3 Port 字段 (按 Type 区分) 见 §6.3、§6.7、§6.8

## 18.4 Control 节点格式

```

<控件名 type="control">
  <Method>W/R</Method>
  <Expression>100</Expression>
</控件名>

```

---

## 19. 附录 B: Recipe 表达式语法

---

### 19.1 常量

直接写数字: 100、3.14、-5、1e-3。

### 19.2 时间变量 t

当前步已耗时, 单位秒。

```
30*cos(t) + 200      # 余弦振荡
200*exp(-t/60)       # 指数衰减
(t < 30) ? 100 : 50   # 条件 - 30 秒前 100, 之后 50
```

### 19.3 支持的函数

```
abs, min, max, pow, sqrt
sin, cos, tan, asin, acos, atan, atan2
exp, log, log10
floor, ceil, round
```

### 19.4 等待语法

[wait]

等某控件读数达到设定值:

```
[wait]500           # 等到数值达到 500 (方向自适应 - 升或降)
```

[wait(条件表达式)]

```
[wait(>500)]        # 等到数值 > 500
[wait(<=100)]       # 等到 ≤ 100
[wait(>=300 && t>60)] # 既要 ≥300 又要耗时 > 60 秒 (逻辑与)
```

### 19.5 字符串控件

写继电器 on/off、邮件收件人这类非数值控件：

```
on / off / true / false  
email@domain.com
```

直接字符串，不是表达式。

---

## 20. 附录 C: 术语表

---

术语	英文	含义
工作区	Workspace	一个 .lab 工程 + 云端对应的记录
配置文件	Config	Workspace 的 XML 文件 (.lab 内嵌 / .xml)
端口	Port	物理 / 虚拟接口抽象
设备	Device	挂在 Port 上的仪器
控件	Control	设备的可读/写变量
驱动	Driver	通信协议实现 (C# 或 Python)
配方	Recipe	按时间步参数化的控制序列
归档	Archive	云端 workspace 或 recipe 的元数据
会话	Conversation	"◆ AI 助手" 的对话历史
提案	Proposal	AI 返回的可应用的修改建议
拨码	DIP Switch	网关硬件上的物理开关
总线	Bus	RS-485 上多设备共享的一对线
从站地址	Slave Address / modaddr	Modbus 协议中从站唯一标识 (1-247)

---

## 21. 获取帮助

---

- **用户手册** (本文)
- 软件内 **Help** → **Guide**
- 邮件: `contacts@lab-elf.com`
- 软件内 **Help** → **Check Update** 主动检查新版
- **反馈 Bug / 功能需求**: 邮件附日志面板截图 + 当前 `.lab` 文件 (可匿名去数据)

### 21.1 诊断信息打包

遇到复杂 Bug 时, 可打包以下信息方便技术支持定位:

1. `Help` → `About` 里的版本号
2. 日志面板里最近一轮错误的上下文
3. 当前 `.lab` 或 `.xml` 文件
4. 如果是 AI 相关: **新对话** 之前的对话截图 + 相关附件
5. 如果是网关 / 网络: `ping` 结果 + 电脑网络配置截图

---

*LabElf 4.0 用户手册 — 2026-04 · 正式版*