

- 1 涂鸦独立授权库 (IndependentAuth) 集成指南
  - 1.1 免责声明与责任边界
  - 1.2 1. 概述
    - 1.2.1 1.1 产品定位
    - 1.2.2 1.2 与统一授权库 (AuthCollection) 的关系
  - 1.3 2. 术语与缩略语
  - 1.4 3. 交付物与工程引用
    - 1.4.1 3.1 交付物清单 (最小集)
    - 1.4.2 3.2 引用路径建议
  - 1.5 4. 环境与前置条件
    - 1.5.1 4.1 运行环境
    - 1.5.2 4.2 业务前置
  - 1.6 5. 核心 API 与事件
    - 1.6.1 5.1 主类型与进度事件
    - 1.6.2 5.2 事件订阅与释放
  - 1.7 6. 串口授权主流程 (逻辑视图)
    - 1.7.1 6.1 工单路径 API
    - 1.7.2 6.2 Token 路径 API
    - 1.7.3 6.3 成功与失败后的读取
  - 1.8 7. 可选测试项 (产线增强)
  - 1.9 8. 静态方法 `GetAndSetMuiltMAC`
  - 1.10 9. 线程、并发与生命周期 (生产建议)
  - 1.11 10. 安全与合规 (集成侧要求)
  - 1.12 11. 常见问题 (FAQ)
  - 1.13 12. 参考实现与类对照 (随附示例, 若有)
  - 1.14 13. 文档维护与支持

# 1 涂鸦独立授权库（IndependentAuth）集成指南

文档副标题：企业对外版

项	内容
文档编号	DOC-INDEPENDENT-AUTH-INT-001
文档类型	对外集成与接口说明
适用程序集	Independentauth.dll（命名空间 Tuya.IndependentAuthorize）
目标读者	产测工具、烧录宿主、MES/产线软件等第三方集成开发人员及技术负责人
版本说明	正文所述 API 以随交付包中的 程序集版本 及《发行说明》为准；若随附参考实现一并交付，版本号应与程序集及《发行说明》对齐后对外发放。

## 1.1 免责声明与责任边界

- 本文档仅描述 **独立授权库** 在典型产线场景下的集成方式与调用约定，**不构成**对具体硬件、固件版本或云端策略的承诺。
- 工单号、Token、账号凭据、设备标识** 等敏感信息须由集成方自行管理；文档中的示例标识均为占位符，**禁止**在生产环境中直接复用。
- 与 TuyaCloudIfLib、Tuya.Core 等组件的版本兼容关系以 **交付包清单** 与 **联调报告** 为准；升级前请在受控环境完成回归验证。

## 1.2 1. 概述

### 1.2.1 1.1 产品定位

**独立授权库**（Independentauth.dll）面向需在 **PC 串口** 侧完成模组授权初始化、正式授权/复检、进度反馈与结果读取的宿主程序。库通过 IndependentAuth 类型封装串口侧状态机，并与云端 TuyaCloudIfLib 协同（登录、Token 信息拉取等）。

1.2.2 1.2 与统一授权库（AuthCollection）的关系

维度	Tuya.AuthCollection.AuthCollection	Tuya.IndependentAuthorize.IndependentAuth
典型场景	多形态通信（如蓝牙、Wi-Fi、Linux TCP 等）下的统一授权入口	串口 侧独立授权：工单或 Token 的初始化、授权/复检、进度与结果读取
事件模型	多类事件（进度、日志、消息、授权数据等）	以 progressEvent 为主（ProgressEventArgs：进度、状态码、提示文案）
Token 辅助能力	由宿主与云端组合实现	提供静态 GetAndSetMulitMAC，用于 Token 场景下多 MAC 相关处理（方法名为历史导出，调用须与程序集一致）
依赖	TuyaCloudIfLib、业务模型等	同样依赖云端登录与 Token/工单上下文；串口参数由宿主传入

**选型建议：**若宿主已统一采用 AuthCollection 且协议与硬件形态均已覆盖，可继续以该库为主。若产线工具已基于 IndependentAuth 深度定制（如授权前复位、GPIO/RSSI 等可选测试项），应保持与本指南推荐的调用顺序与单串口独占原则一致，避免两套串口协议栈争用同一 COM 口。

1.3 2. 术语与缩略语

术语	说明
宿主	集成授权能力的上位机应用程序
工单	云端下发的生产订单/授权凭证上下文
Token	云端生产 Token 及其关联额度、产品信息
COM	Windows 串口设备名，如 COM3
工艺节点	产线工站节点编码，用于云端校验与路由；参考实现中常见默认值为 firstTest（以实际项目配置为准）

1.4 3. 交付物与工程引用

1.4.1 3.1 交付物清单（最小集）

序号	交付项	说明
1	Independentauth.dll	主程序集
2	传递依赖	以实际绑定为准；常见包括 TuyaCloudIfLib、 Newtonsoft.Json、 Tuya.Core 等（以交付包 <b>deps 清单</b> 为准）
3	《发行说明》	版本变更、已知问题、兼容性矩阵

1.4.2 3.2 引用路径建议

- 推荐将 Independentauth.dll 置于企业内部的 **标准制品库目录**（例如与云端接口库同级的 IndependentAuth 目录），并在工程中设置 **复制到输出目录**（Private=true），保证运行时解析一致。
- 参考示例工程使用 **多路径回落** 引用：优先 ..\releaselibrary\IndependentAuth\Independentauth.dll，否则使用 packages\Independentauth.dll。**对外项目**应以贵司制品管理规范为准，不强制目录名。

1.5 4. 环境与前置条件

1.5.1 4.1 运行环境

- **操作系统**：与宿主程序一致，须满足 Independentauth.dll 目标框架（常见为 .NET Framework，具体以交付物为准）。
- **权限**：串口访问需避免被其他进程占用；必要时以管理员或设备策略放行。
- **网络**：云端登录、Token 查询等依赖 **HTTPS** 可达的涂鸦生产/测试环境（环境 URL 以合同与交付配置为准）。

1.5.2 4.2 业务前置

1. **云端登录成功**：例如通过 TuyaCloudIf.UserLogin 完成后再进行 Token 拉取或授权。
2. **入参有效**：工单号、TokenId、串口号、波特率、工艺节点编码须与云端及产线配置一致。
3. **可选 JSON 配置**：部分产线工具在启动时读取 %AppData%\tuya\AuthConfig.json，用于调整进入测试模式等参数（如 testModeCmdSendTimes、testModeCmdSendInterval）。若宿主无该文件，应使用库默认或由 **MES/配置中心** 下发等价配置；字段语义以随库说明为准。
4. **硬件就绪**：串口独占；无设备或参数错误时，AuthInit\_SerialPort / TokenAuthInit\_SerialPort 将失败，应通过 GetErrorCode 获取原因并上报产线 UI。

## 1.6 5. 核心 API 与事件

### 1.6.1 5.1 主类型与进度事件

- **类型：** `Tuya.IndependentAuthorize.IndependentAuth`
- **进度事件：** `progressEvent`，参数类型 `Tuya.IndependentAuthorize.Event.ProgressEventArgs`（典型成员：`Progress`、`Code`、`Message`）。
- **界面语言：** `SetLanguage(MessageCodes.LANGUAGE.CHINESE | ENGLISH)`，建议与宿主 UI 语言一致，并在每次发起授权前同步。

### 1.6.2 5.2 事件订阅与释放

宿主须在 **实例释放或重建前** 取消 `progressEvent` 订阅，避免重复订阅与悬空引用。推荐模式：**取消订阅** → **释放引用** → **新建实例** → **再次订阅** → **再次应用可选测试项配置**。

## 1.7 6. 串口授权主流程（逻辑视图）

下列步骤与成熟产线工具中「串口线程内执行授权」的时序一致，分为 **工单** 与 **Token** 两条路径。

图 1 串口授权主流程（逻辑视图）

阶段	宿主动作	库与设备
T1	<code>new IndependentAuth()</code> ，并 <code>SetLanguage</code>	实例就绪
T2	按需调用 <code>EnableResetBeforeStart</code> 、 <code>EnableGpioTest / SetGpioParameters</code> 、 <code>EnableRssiTest / SetRssiParameters</code>	可选测试项配置
T3	订阅 <code>progressEvent</code>	后续进度回调
T4a（工单）	<code>AuthInit_SerialPort(orderCode, portName, baudRate, nodeCode)</code>	与模组串口交互完成初始化
T4b（Token）	<code>TokenAuthInit_SerialPort(tokenId, portName, baudRate)</code>	与模组串口交互完成初始化
T5a（工单）	<code>Auth(sn)</code> 或 <code>AuthCheck(sn)</code>	正式授权或复检
T5b（Token）	<code>TokenAuth(sn, needMac)</code> 或 <code>TokenAuthCheck(sn)</code>	正式授权或复检
T6	若任一步返回 <code>false</code> ，调用 <code>GetErrorCode(ref code, ref msg)</code>	失败诊断
T7	若成功，调用 <code>GetTokenAmount</code> 、 <code>GetAuthMac</code> 、 <code>GetAuthUuid</code>	结果汇总、上报 MES

1.7.1 6.1 工单路径 API

步骤	方法	说明
1	<code>bool AuthInit_SerialPort(string orderCode, string portName, int baudRate, string nodeCode)</code>	绑定工单、串口与工艺节点；失败时配合 <code>GetErrorCode</code>
2	<code>bool Auth(string sn)</code>	正式授权
2'	<code>bool AuthCheck(string sn)</code>	二次检/抽检

1.7.2 6.2 Token 路径 API

步骤	方法	说明
1	<code>bool TokenAuthInit_SerialPort(string tokenId, string portName, int baudRate)</code>	绑定 Token 与串口
2	<code>bool TokenAuth(string sn, bool needMac)</code>	第二参数表示是否需要 MAC 相关处理（与产线 UI 勾选一致）
2'	<code>bool TokenAuthCheck(string sn)</code>	Token 复检

1.7.3 6.3 成功与失败后的读取

方法	说明
<code>void GetTokenAmount(out int totalAmount, out int usedAmount)</code>	额度信息
<code>void GetAuthMac(out string authMac)</code>	授权 MAC
<code>void GetAuthUuid(out string uuid)</code>	UUID；部分产线在成功后调用云端上报接口做数据打标（以业务为准）
<code>void GetErrorCode(ref string errCode, ref string errMsg)</code>	在 <code>AuthInit / Auth / TokenAuth</code> 等返回 <code>false</code> 后调用

1.8 7. 可选测试项（产线增强）

下列 API 用于授权前/过程中的附加测试能力，**是否启用**应以产品与工艺要求为准。

方法	作用
<code>EnableResetBeforeStart(bool)</code>	授权前是否执行模组复位
<code>EnableGpioTest(bool) / SetGpioParameters(string moduleType, int overtime, int num)</code>	GPIO 测试及参数
<code>EnableRssiTest(bool enable, string ssid, int min, int max) / SetRssiParameters(string moduleType, int overtime, int num)</code>	RSSI 测试及参数

**说明：** `moduleType` 建议使用云端 `GetTokenInfo` 返回的 `result.type`（与模组品类一致）。

### 1.9 8. 静态方法 `GetAndSetMulitMAC`

在 Token 上下文就绪后，部分产线工具会调用：

```
IndependentAuth.GetAndSetMulitMAC(tokenInfo.result.id, tokenInfo.result.type.ToLower(), "zh");
```

参数	说明
第一参数	Token 侧标识，与 <code>GetTokenInfo</code> 结果一致
第二参数	模组 <code>type</code> 的小写形式
第三参数	语言简码，如 <code>zh</code> 、 <code>en</code>

**注意：**方法名中的 `Mulit` 为历史拼写，须与程序集导出名称完全一致。

该方法可能涉及 **阻塞或 UI 交互**（取决于程序集实现与运行环境）。**WPF/WinForms** 宿主应注意线程模型与消息泵；控制台或服务进程应在集成测试中验证是否适合直接调用。

### 1.10 9. 线程、并发与生命周期（生产建议）

- 1. **避免阻塞 UI 线程：**产线 GUI 程序宜在后台工作线程中执行授权序列，并通过 `progressEvent` 封送回 UI 线程更新进度。
- 2. **单串口独占：**同一 COM 口上，不得与另一套串口协议栈交叉并发访问。
- 3. **配置变更后重建实例：**GPIO/RSSI/复位等配置变化时，应 **取消事件** → **新建** `IndependentAuth` → **重新订阅与配置**；禁止在授权进行中对同一端口并发构造两个实例。
- 4. **异常隔离：**建议在工作线程外层捕获未处理异常，记录 **脱敏** 后的错误码与工位信息，避免进程崩溃导致整线停摆。

1.11 10. 安全与合规（集成侧要求）

- 1. **凭据**：账号、密码、Token、私钥等 **不得** 硬编码在可发布源码或客户端配置中；应使用安全存储或受控配置下发。
- 2. **日志**：禁止在日志中输出完整 Token、密码、原始 `Authorization` 头或完整会话标识；错误排查可使用 **错误码 + 脱敏工单/序列号片段**。
- 3. **传输**：与云端通信须使用 **HTTPS**，并遵守企业网络与证书策略。
- 4. **数据最小化**：仅采集与授权、追溯相关的必要字段；结果上报 MES 的内容与保留周期遵循贵司数据治理制度。

1.12 11. 常见问题（FAQ）

- Q**： `AuthInit_SerialPort` **立即返回 false**？  
**A**：依次检查：`COM` 是否存在、是否被占用、波特率是否与模组一致、工单与工艺节点是否有效；然后读取 `GetErrorCode` 。
- Q**：与 `AuthCollection` **同进程集成是否可行**？  
**A**：可行前提是 **不同物理链路或分时复用**；同一串口同一时刻只允许一个栈访问。
- Q**：无桌面 UI 的服务端能否调用？  
**A**：须以实际交付的 `Independentauth.dll` 行为为准；若内部依赖对话框，需在集成测试中验证或改为有 UI 的宿主承载。

1.13 12. 参考实现与类对照（随附示例，若有）

以下类名来自随附参考源码，仅供集成商快速对照，**不作为**对外交付的强制工程结构。

类/成员（示例）	说明
<code>IndependentAuthExample</code>	控制台骨架：登录复用、可选 <code>GetAndSetMulitMAC</code> 、工单/Token 两段独立会话、进度订阅与可选测试项封装
<code>Program</code>	在统一授权示例之后调用独立授权示例入口
<code>AuthCollectionExample.TryLogin</code>	共享云端登录态，避免重复登录（示例中的便利封装）

1.14 13. 文档维护与支持

- **文档维护**：与交付程序集及《发行说明》同步修订；对外发放前应核对 **文档编号** 与程序集版本一致。
- **技术支持**：请联系贵司对接的 **涂鸦商务/技术支持渠道**；本文档不列具体联系人，以免过期。