

## Limbus Company 自动化 v0.1

### Codex 开发文档 + Prompt

版本：v0.1（只做到：启动 → 登录界面 → 主菜单）

分辨率：1600×900；窗口模式；语言：日语

约束：不读内存、不改文件，只做 UI 操作

参考：用户文档《自动化契约 4》（2025-12-25）与《自动化契约 3》（含截图标志）

## 1. 目标与验收标准

本版本只实现最小闭环：

- 启动游戏进程（LimbusCompany.exe）
- 从启动闪屏等待到登录界面（检测“标志 A”）
- 在登录界面点击红框以外区域触发进入（等待 connecting → 主界面）
- 30 秒内进入主菜单则判定成功，弹窗提示“成功进入主菜单”
- 任意步骤超时/失败：截图留证 → 关闭游戏 → 返回失败码

验收标准（必须满足）：

- 连续运行 5 次，至少 4 次成功进入主菜单（同一台机器、同一分辨率）
- 失败时必须能在 logs/ 里找到对应截图与日志行（便于复现）

## 2. 技术栈（推荐）

语言：Python 3.11+（本地脚本最快落地）

依赖：

- mss：窗口区域截屏（稳定、快）
- opencv-python：模板匹配（检测标志 A / connecting / 主菜单标志 B）
- pyautogui：鼠标点击（必要时可用 pydirectinput 作为替代）
- pywin32：窗口置顶/聚焦、获取窗口位置（减少点错屏幕）
- psutil：检测/关闭进程（对应契约中的进程名判定）

说明：v0.1 不引入 OCR；通过模板匹配识别界面标志即可。

### 3. 目录结构（建议）

- limbus\_automation/
  - main.py           # 入口：跑一次 v0.1
  - config.py         # 超时、阈值、窗口标题等配置
  - vision/
    - match.py         # 截屏 + 模板匹配封装
  - input/
    - click.py         # 点击封装（按窗口坐标）
  - runtime/
    - process.py       # 进程检测/关闭
    - window.py        # 置顶/聚焦/窗口 bbox
  - assets/
    - login\_marker\_A.png   # 标志 A（红框里的固定图案）
    - connecting\_marker.png   # CONNECTING 黄框区域
    - mainmenu\_marker\_B.png   # 主菜单标志 B（你标注的黄框）
  - logs/            # 运行日志与失败截图输出

### 4. v0.1 状态机（写代码就照这个走）

状态与转移：

状态	规则（成功条件 / 超时处理）
S0_START	启动游戏；30 秒内检测到进程 LimbusCompany.exe，否则失败退出
S1_WAIT_LOGIN	循环截屏并匹配 标志 A；找到则进入 S2； 超时失败退出
S2_CLICK_LOGIN	在登录界面点击红框以外区域；若出现 connecting 标志则进入 S3

S3_WAIT_MAINMENU	connecting 出现后最多等待 30 秒；检测到主菜单标志 B 视为成功，否则失败退出
S_OK	弹窗/打印：成功进入主菜单；退出
S_FAIL	截图留证 + 关闭游戏 + 退出码非 0

## 5. 关键实现细节（避免翻车）

- 只截取游戏窗口 bbox（不要全屏截）：更快、更稳定、也避免点错坐标。
- 模板匹配建议用多尺度/或固定尺度 + 合理阈值（例如 0.80~0.90），并对匹配结果做去抖（连续命中 3 帧再判定）。
- 每个状态都要有硬超时；超时路径必须一致：记录日志 → 截图 → 关闭游戏。
- 把所有等待/阈值放在 config.py，避免“散落在代码里改不动”。

## 6. Codex Prompt（直接复制到 Codex 的 System / Instruction）

把下面整段作为 Codex 的“系统指令/最高优先级指令”。要求：温度=0（或尽量降低随机性）。

你是一个严格的工程实现助手。请只实现文档中的 v0.1：启动 → 登录界面 → 主菜单。

必须遵守：

- 1) 只做 UI 自动化：不读内存、不改文件、不注入、不修改游戏数据。
- 2) 使用技术栈：Python 3.11+, mss, opencv-python, pyautogui, pywin32, psutil。
- 3) 代码必须可运行、可重复。所有超时/阈值集中在 config.py。
- 4) 任何失败都必须：写日志（含当前状态名）+ 保存截图到 logs/ + 关闭游戏进程 + 以非 0 退出。
- 5) 不要添加 v0.1 之外的功能（例如疲劳转换、OCR 读数、复杂重试策略）。

实现要求：

- 目录结构按文档第 3 节生成。
- main.py 作为入口：运行一次流程，并在成功时打印并弹窗“成功进入主菜单”。

- 状态机按文档第 4 节实现：S0\_START → S1\_WAIT\_LOGIN → S2\_CLICK\_LOGIN → S3\_WAIT\_MAINMENU → S\_OK / S\_FAIL。
- 模板图片从 assets/ 读取：login\_marker\_A.png / connecting\_marker.png / mainmenu\_marker\_B.png。
- 模板匹配函数：输入窗口截图，返回是否命中 + 命中位置（用于可选的调试日志）。

输出格式：

- 直接输出完整代码（多文件），每个文件用清晰分隔标识文件路径。
- 不要输出解释性长文；最多用简短注释说明关键点。

## 7. 你需要准备的素材清单

把你在契约里标注的红框/黄框区域裁剪成小图，放到 assets/：

- login\_marker\_A.png：登录界面左上角固定标志（红框内图案）
- connecting\_marker.png：右上角 CONNECTING 的黄色方框区域
- mainmenu\_marker\_B.png：主菜单界面你标注的黄色方框区域

裁剪原则：只保留“稳定不变”的部分，尽量小（但要有足够纹理/文字特征）。

## 8. 迭代路线（v0.2 以后再做）

- v0.2：加入“点击 xxx/yyy → 点击 >> → 点击确认”的流程（仍优先模板匹配，不上 OCR）。
- v0.3：只在必要时对小区域做 OCR 读取 xxx/yyy 并做阈值判断 ( $xxx < 20$ )。
- v1.0：加入每日定时/托盘程序/失败通知等工程化能力。