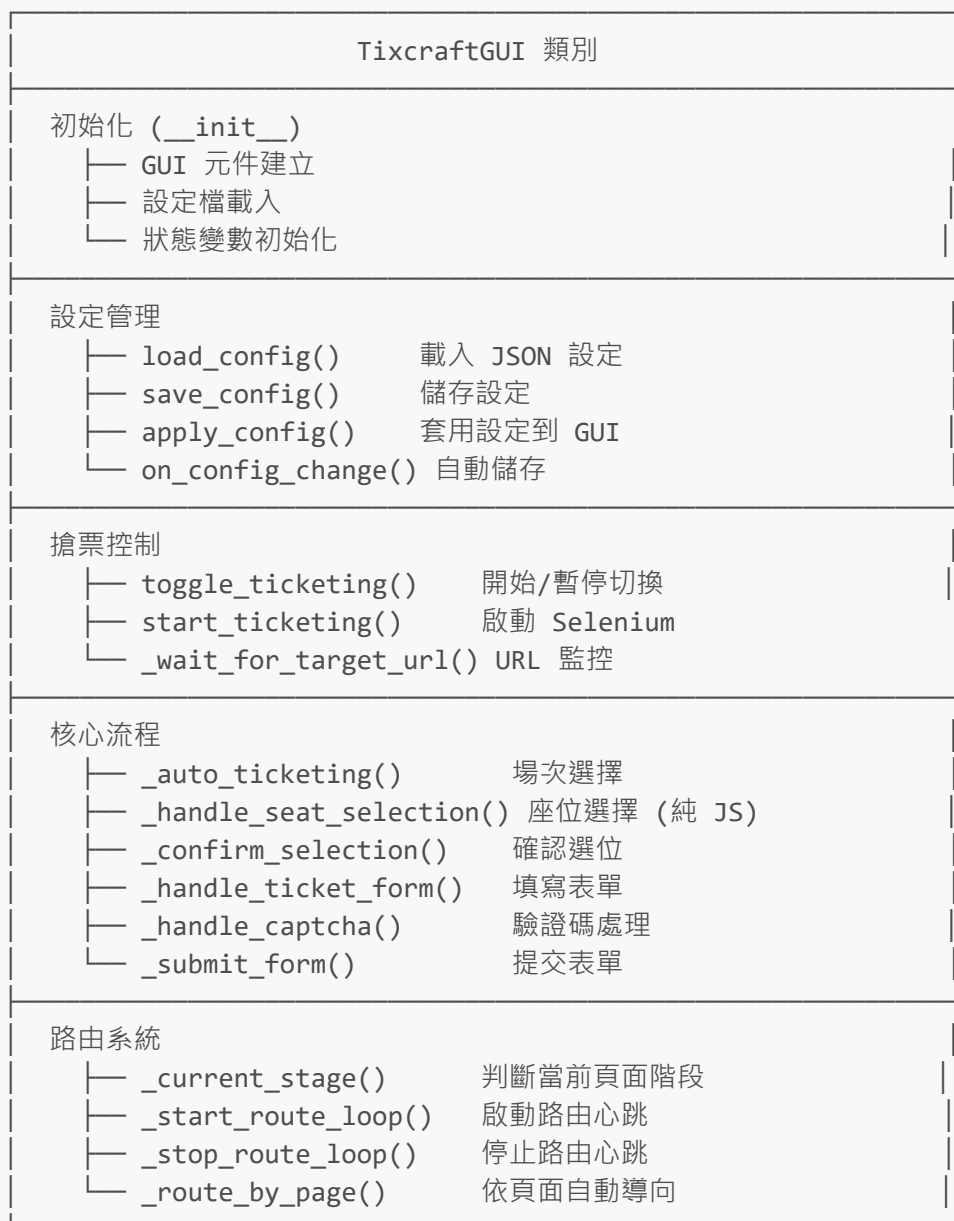


Tixcraft 搶票機器人 - 程式架構說明

📁 檔案資訊

- 檔案: `tixcraft.py`
- 總行數: 980 行
- 功能: 拓元售票網站 (tixcraft.com) 自動搶票機器人
- 技術棧: Python + Tkinter (GUI) + Selenium (瀏覽器自動化) + ddddocr (驗證碼辨識)

📑 程式架構總覽



⚙ 全域常數設定 (第 15-47 行)

時間設定

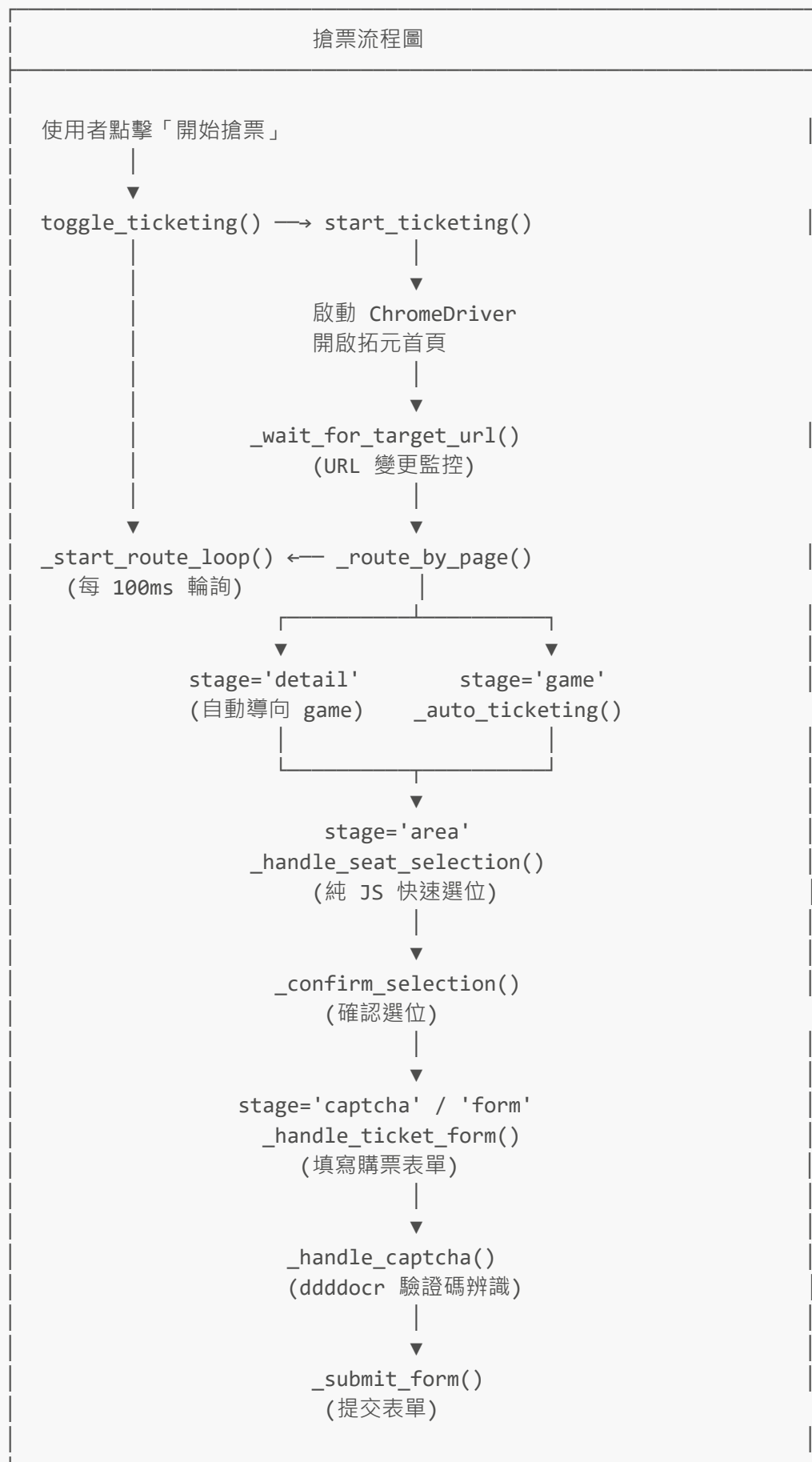
常數	值	用途
POLL_INTERVAL_MS	100ms	GUI 輪詢間隔
SEAT_POLL_INTERVAL_MS	100ms	選區頁輪詢間隔
SLEEP_SHORT_SEC	0.1s	最小延遲時間
AREA_REFRESH_INTERVAL_MS	2000ms	選區頁重整間隔

WebDriverWait 超時設定

常數	值	用途
WAIT_LIST_ROWS	10s	場次列表載入
WAIT_AREA_PAGE	15s	選位頁載入
WAIT_AREA_LIST	10s	區域列表顯示
WAIT_CONFIRM_BTN	10s	確認按鈕可點
WAIT_FORM_PAGE	15s	表單頁載入
WAIT_SELECTS	5s	張數下拉載入
WAIT_CHECKBOXES	5s	Checkbox 載入
WAIT_CAPTCHA_IMG	15s	驗證碼圖片出現
WAIT_CAPTCHA_VISIBLE	10s	驗證碼圖片可見
WAIT_CAPTCHA_READY	5s	驗證碼載入就緒
WAIT_CAPTCHA_INPUT	10s	驗證碼輸入框可點
WAIT_POST_SUBMIT	1s	提交後狀態變化

其他設定

常數	值	用途
CAPTCHA_MAX_ATTEMPTS	2	驗證碼最大嘗試次數
CAPTCHA_LEN	4	驗證碼長度
REFRESH_ON_AREA	True	選區頁自動重整開關
ALLOWED_STAGES_AFTER_CAPTCHA	<code>{'detail', 'game', 'area', 'captcha'}</code>	驗證碼送出後白名單頁面



方法詳細說明

1. 初始化與設定 (第 49-156 行)

`__init__(self, root)` (第 49-109 行)

初始化 GUI 與狀態變數：

- 建立 GUI 元件 (網址輸入框、LOG 區、場次輸入框、張數輸入框、按鈕)
- 載入設定檔
- 初始化狀態變數：
 - `is_running`: 搶票執行狀態
 - `_route_timer`: 路由心跳排程
 - `_routing`: 路由重入保護
 - `_after_captcha_submitted`: 驗證碼送出旗標
 - `_ocr`: OCR 物件 (懶載入)

設定管理方法

方法	行數	功能
<code>load_config()</code>	111-127	從 <code>tixcraft_config.json</code> 載入設定
<code>save_config()</code>	129-140	儲存設定至 JSON
<code>apply_config()</code>	142-145	將設定套用到 GUI 輸入框
<code>on_config_change()</code>	147-148	設定變更時自動儲存
<code>on_closing()</code>	150-156	視窗關閉時清理資源

2. 搶票控制 (第 172-271 行)

`toggle_ticketing()` (第 172-185 行)

開始/暫停搶票切換：

- 切換 `is_running` 狀態
- 更新按鈕文字
- 啟動/停止路由心跳

`start_ticketing()` (第 187-220 行)

啟動 Selenium：

- 建立 `ChromeOptions` (停用自動化偵測)
- 使用獨立 `user-data-dir` 避免衝突
- 開啟拓元首頁

- 啟動 URL 監控

`_wait_for_target_url()` (第 222-271 行)

監控 URL 變更：

- 偵測 URL 變化
- `detail` 頁自動導向 `game` 頁
- 符合監控網址時啟動搶票流程

3. 場次選擇 (第 273-318 行)

`_auto_ticketing()` (第 273-318 行)

自動選擇場次：

流程：

1. 等待場次列表載入 (`tr.gridc.fcTxt`)
2. 遍歷所有場次
3. 檢查按鈕是否 `enabled`
4. 依關鍵字匹配或選第一個可購票場次
5. JS 點擊按鈕
6. 呼叫 `_handle_seat_selection()`

4. 座位選擇 (第 320-454 行)

`_handle_seat_selection()` (第 320-398 行)

純 JS 快速選位 (核心優化)：

```
// JS 選位邏輯 ( 一次通訊完成 )
(function(n) {
    const container = document.querySelector('div.zone.area-list');
    const links = container.querySelectorAll('li[class^="select_form_" ] > a[id]');

    for (const a of links) {
        // 可見性檢查
        if (!a.offsetParent) continue;
        if (getComputedStyle(a).pointerEvents === 'none') continue;
        if (a.classList.contains('disabled') || a.classList.contains('sold')) continue;

        // 剩餘數量檢查：剩餘 < 購票數則跳過
        const m = text.match(/剩餘\s*(\d+)/);
        if (m && parseInt(m[1]) < n) continue;
    }
})
```

```

        // 能點就點
        a.click();
        return {found: true, ...};
    }
    return {found: false, ...};
})(n);

```

效能優勢：

項目	舊版 (Selenium)	新版 (純 JS)
通訊次數	N × 5+ 次	1 次
選位速度	~500ms-2s	~10-50ms

`_confirm_selection()` (第 400-454 行)

確認選位並進入下一步：

- 等待確認按鈕載入
- 檢查按鈕文字含「確認」「下一步」等
- JS 點擊按鈕
- 呼叫 `_handle_ticket_form()`

5. 表單處理 (第 456-524 行)

`_handle_ticket_form()` (第 456-524 行)

處理購票表單：

流程：

1. 等待表單頁載入
2. 設定購票張數 (下拉選單)
3. 勾選所有 checkbox (同意條款)
4. 呼叫 `_handle_captcha()`

`_normalize_captcha()` (第 526-548 行)

標準化 OCR 結果：

- 去除空白，保留英數字
- 長於 4 位取前 4 位
- 短於 4 位以最後字元補齊

6. 驗證碼處理 (第 566-795 行)

`_on_captcha_form()` (第 566-580 行)

判斷是否在驗證碼表單頁：

- URL 含 `/ticket/ticket/`
- 有驗證碼圖片或輸入框

`_handle_captcha()` (第 582-795 行)

驗證碼處理流程：

1. 確認在驗證碼表單頁
2. 等待驗證碼圖片載入
3. Canvas 擷取圖片 Base64
4. ddddocr OCR 辨識
5. 標準化結果 (4位英數字)
6. 填入輸入框
7. 提交表單
8. 檢查是否成功 (最多重試 2 次)

7. 表單提交 (第 797-843 行)

`_submit_form()` (第 797-843 行)

提交購票表單：

- 按鈕搜尋順序：
1. `button.btn.btn-primary.btn-green`
 2. `button[type="submit"].btn-green`
 3. `input[type="submit"]`
 4. `button[type="submit"]`
 5. `input[value*="確認/提交/下一步"]`
 6. `button[onclick*="submit"]`
 7. `button.btn-green`

8. 路由系統 (第 845-980 行)

`_current_stage()` (第 845-870 行)

判斷當前頁面階段：

URL 特徵	回傳值
<code>/activity/detail/</code>	<code>'detail'</code>

URL 特徵	回傳值
/activity/game/	'game'
/ticket/area/	'area'
/ticket/ticket/ + 驗證碼元素	'captcha'
/ticket/ticket/	'form'
其他	'unknown'

_start_route_loop() (第 872-891 行)

啟動路由心跳：

- 每 100ms 執行 `_route_by_page()`
- 持續監控頁面狀態

_stop_route_loop() (第 893-900 行)

停止路由心跳

_route_by_page() (第 902-972 行)

依頁面自動導向：

路由邏輯：

```

if stage == 'detail':
    → 自動導向 game 頁

if stage == 'game':
    → _auto_ticketing()

if stage == 'area':
    → _handle_seat_selection()
    → 選位失敗時自動重整

if stage == 'captcha':
    → _handle_ticket_form()

if stage == 'form':
    → _handle_ticket_form()

if stage == 'unknown':
    → 繼續輪詢

```

防護機制

機制	說明
重入保護	<code>_routing</code> 旗標防止路由重複執行
白名單限制	驗證碼送出後僅在特定頁面動作
自動重試	驗證碼錯誤自動重試 (最多 2 次)
自動重整	選位無票時自動重整頁面
狀態監控	持續監控 URL 變化，自動導向正確流程
錯誤處理	各階段皆有 try-catch，錯誤後回到路由輪詢

依賴套件

```
# 標準庫
import tkinter as tk
from tkinter import ttk, scrolledtext
import json
import os
import time
import base64
import tempfile

# 第三方套件
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from PIL import Image
import ddddocr
```

設定檔格式

檔案: `tixcraft_config.json`

```
{
  "url": "搶票網址",
  "session": "購票場次關鍵字",
  "ticket_count": "1"
}
```

使用流程

1. 執行程式，開啟 GUI

2. 輸入搶票網址、場次關鍵字、購票張數
 3. 點擊「開始搶票」
 4. 程式自動開啟 Chrome，使用者手動登入拓元帳號
 5. 進入目標頁面後，程式自動執行：
 - 選擇場次 → 選擇座位 → 填寫表單 → 辨識驗證碼 → 提交
-

版本特點

- **純 JS 選位**：減少 Selenium 通訊次數，選位速度提升 10-50 倍
- **剩餘數量篩選**：自動跳過剩餘數量不足的區域
- **路由系統**：統一管理頁面導向，程式碼更清晰
- **自動重整**：選位失敗時自動重整頁面
- **懶載入 OCR**：避免每次嘗試都新建 OCR 物件