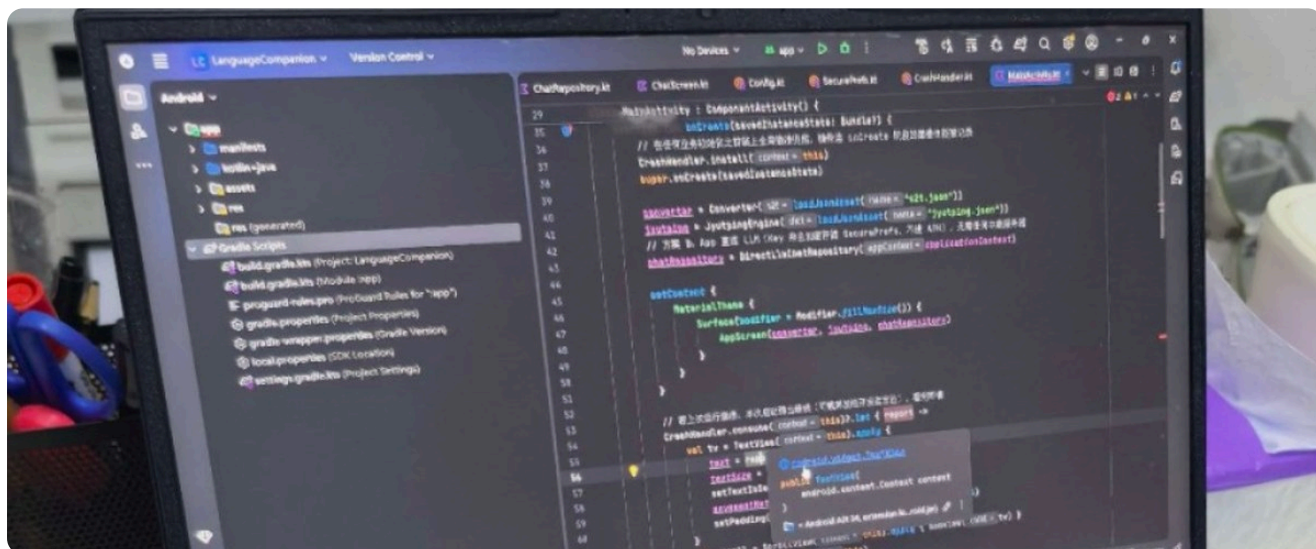


兩 / 文 / 三 / 語

語言融合夥伴

一個為「內地轉校生」而做的 Android App

把「簡繁轉換、粵拼標注、三語場景對話」三件轉校生最需要的事，放進同一個手機應用——完全離線的語言工具，加上一位隨時在線的 AI 語言陪練。



作者

柳宇澤 (Liu Yuze)

技術

Kotlin + Jetpack Compose | Python + FastAPI

年份

2026

本作品集內容一覽

01 封面 項目定位與目錄

02 概覽 這是甚麼項目

03 緣起 為甚麼要做這個 App

04 步驟 01 系統設計與技術選型

05 步驟 02 模塊 ① 離線語言工具

06 步驟 03 模塊 ② AI 三語場景對話

07 步驟 04 動手時攻克的六個難題

08 技術亮點 ① 香港標準，不是台灣標準

09 技術亮點 ② API Key 從不進 App

10 步驟 05 讓數據說話，收穫

這是甚麼項目？

我即將從內地轉學到香港的中學讀書，馬上要面對看繁體字課本、聽講粵語、用英文上課三個挑戰——也就是香港所說的「兩文三語」。市場上找不到一款專門幫「內地轉校生」適應香港語言環境的工具，於是在家人的指導下，我自己設計並開發了這個 Android App。

37,905

簡繁詞庫 · 詞組數
(另 3,848 個單字)

27,438

粵拼詞庫 · 收字數
(源自 rime-cantonese)

100%

真實語境覆蓋率
(5 組句子 / 105 漢字)

模塊 ① 離線語言工具

簡繁轉換 簡體 ↔ 香港標準繁體，雙向轉換
粵拼標注 對漢字逐字標注粵語拼音 Jyutping

完全離線：詞庫全部打包進 App 內部，打開即用，不發送任何數據。

模塊 ② AI 三語場景對話

內置 5 個香港校園 / 生活場景，可選普通話 / 粵語 / English 三種語言回覆。

AI 除了對話，還給出**改進建議**，粵語回覆自動附上**逐字粵拼**並可一鍵朗讀。

這是一個「離線為主、AI 為輔」的語言學習工具：學習者打開 App 就能查字、標音、練對話，不依賴網絡、不洩露密鑰。

項目檔案

項目名稱	兩文三語 · 語言融合夥伴 (Language Companion)
項目類型	Android 手機應用 + 自建後端服務 (個人獨立開發)
手機端	Kotlin + Jetpack Compose (Android 原生) · MVVM 架構，完全離線詞庫引擎
服務器端	Python + FastAPI 中繼服務器：代理調用大語言模型、服務端三語 TTS 朗讀
數據來源	OpenCC 詞典 (簡繁) + rime-cantonese (粵拼) 等開源權威數據
時間 · 作者	2026 年 柳宇澤 (Liu Yuze)

用到的工具與技術

Android Studio

Kotlin

Jetpack Compose

Python

FastAPI

JSON 詞庫

OpenCC 詞典

rime-cantonese

自動化驗證腳本

Git

為甚麼要做這個 App

項目不是「作業」，而是我轉學前真實要面對的三道難關。

繁

看繁體字課本

網上工具多為**台灣標準**，「軟件」會被轉成「軟體」，在香港是錯的。

粵

聽講粵語

課堂、茶餐廳、日常問候，粵語聽不懂就融入不了新環境。

EN

用英文上課

香港中學不少科目以英語授課，需要提前把「敢開口」練起來。

三個挑戰合起來，就是香港所說的「兩文三語」。

既然市面沒有現成工具，我便自己動手——把「查字、標音、練對話」三件事收進同一個 App，讓和我一樣的內地轉校生，一打開手機就能開始適應。

項目定位

內地轉校生

兩文三語

香港標準繁體

粵拼 Jyutping

離線可用

AI 陪練

從想法到成品，過程五步（本作品集的主線）

STEP 01

系統設計

拆成離線與 AI 兩條線

STEP 02

離線工具

簡繁 + 粵拼標注

STEP 03

AI 對話

5 場景 × 3 語言

STEP 04

攻克難題

6 個問題 逐個解決

STEP 05

驗證收穫

100% + 4/4 通過

主要使用者

和我一樣的**內地轉校生**：剛到香港，要看繁體課本、要說粵語、要用英文上課；需要一個**隨時可用、不用花錢、不用註冊**的入門工具。

兩條設計原則

離線優先：最常用的查字、標音必須無網也能用，學習內容不上傳。

安全優先：任何密鑰都不進 App，把風險留在服務器一端。

系統設計：兩條線，一個 App

把功能拆成「離線」與「AI」兩條線，各自用最合適的方式實現。



技術選型

部分	技術
手機 App	Kotlin + Jetpack Compose (Android 官方新一代 UI 框架) · MVVM 架構
詞庫生成	Python 腳本從 OpenCC、rime-cantonese 等開源權威數據源生成詞庫 JSON
AI 對話	自建中繼服務器 (Python + FastAPI) 代理調用大語言模型；服務端 TTS 三語朗讀
質量保障	自動化驗證腳本 + 契約測試 · 鎖定詞庫格式與接口契約

詞庫是怎麼造出來的 (離線模塊的數據流水線)



模塊 ① 離線語言工具

簡繁轉換 + 粵拼標注，詞庫打包進 App，全程不聯網、不發送數據。

簡繁轉換

簡體 ↔ 香港標準繁體，雙向轉換。採用 OpenCC 全量權威詞庫（37,905 條詞組 + 3,848 個單字 + 4,390 條回簡映射），並串接港式異體標準。

粵拼標注

對輸入漢字逐字標注粵語拼音 Jyutping，詞庫共 27,438 字（源自 rime-cantonese）；一字多音用「/」分隔列出。

真實運行輸出（來自離線驗證腳本）

— 示例 1 —

輸入：这个软件的信息很好

繁體：這個軟件的信息很好 ← 香港標準，不是台灣的「軟體」

回簡：这个软件的信息很好

粵拼：這(ze2) 個(go3) 軟(jyun5) 件(gin6) 的(dik1) 信(seon3) 息(sik1) 很(han2) 好(hou2)

— 示例 2 —

輸入：你好，我想问老师一个问题 ← 簡體輸入也能標到粵拼

繁體：你好，我想問老師一個問題

粵拼：你(nei5) 好(hou2) 我(ngo5) 想(soeng2) 問(man6) 老(lou5) 師(si1) 一(jat1) 個(go3) 問(man6) 題(tai4)

覆蓋率：100%



模塊 ② AI 三語場景對話

5 個香港場景 × 3 種語言 · AI 回覆 + 改進建議 + 本地粵拼 + 一鍵朗讀。

內置 5 個香港校園 / 生活場景

可選回覆語言

課堂提問

茶餐廳點餐

問路

便利店購物

普通話

粵語

English

日常問候

真實運行輸出 (來自中繼服務器契約測試)

場景 課堂提問 / 語言 粵語

reply : 唔錯啲！香港課堂通常會先講「唔該」再加老師稱呼，例如：「唔該老師，我想問吓呢條題。」

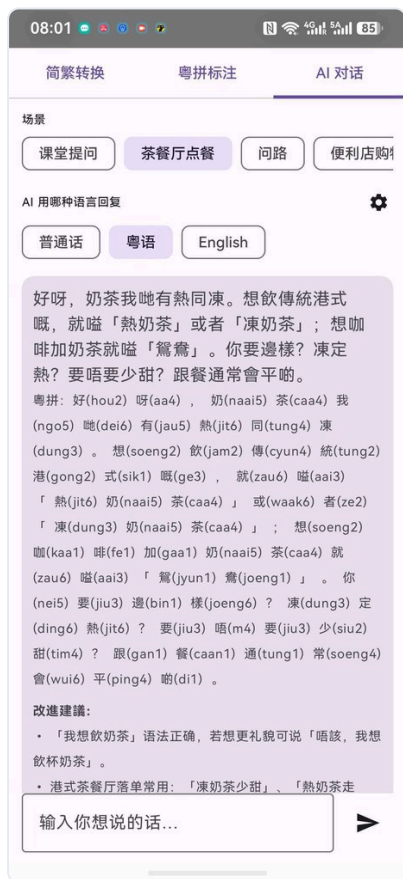
notes : 「唔該」係香港課堂常用禮貌用詞，比「謝謝」更貼近口語。

「問吓」嘅「吓」係粵語口語助詞，等同普通話嘅「一下」。

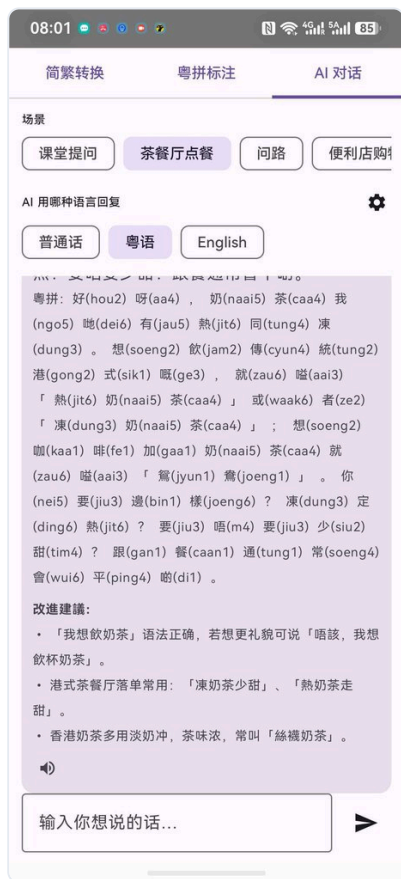
場景 茶餐廳點餐 / 語言 English

reply : Try this natural phrasing: 'I'll have a scrambled egg sandwich and a hot milk tea, thank you.'

notes : Hot milk tea = 'silk stocking milk tea' (絲襪奶茶) – a HK specialty.



App 截圖·場景對話 (一)
粵語回覆 + 逐字粵拼 + 改進建議



App 截圖·場景對話 (二)
茶餐廳點餐，一鍵朗讀

動手時，我攻克的六個難題

遇到的問題 → 怎麼查、怎麼解 —— 這是在項目裏成長最快的地方。

難題 1 回簡總是「回錯字」

把「綫」轉成繁體再轉回簡體，卻變成「線」，字都不認識了。查資料才明白：**繁簡不是一一對應**，一個簡體字可能對應多個繁體字，直接把映射表倒過來用就會「撞車」。

→ **解決**：改用 OpenCC 官方「繁→簡 (TS)」字典**單獨生成回簡映射**，兩個方向各用各的權威數據，不再靠倒推。

難題 2 轉出來是「台灣繁體」

現成詞庫把「软件」轉成「軟體」、「里」轉成「裡」—— 這些在台灣是對的，**在香港卻是錯的**，香港課本寫的是「軟件」、「裏」。

→ **解決**：研究 OpenCC 的詞典體系，把「簡體→傳統繁體→香港異體 (HKVariants)」**兩層串接**，用字才與香港中學課本一致。

難題 3 詞組被「拆散」逐字轉換

替換時先命中了某個單字，把整個詞組「破壞」掉，後面的詞組就再也匹配不上了，結果輸出的是半繁半簡的「夾生」文字。

→ **解決**：把詞庫按詞組長度**由長到短排序**：先做整詞替換，最後才輪到單字處理。

難題 4 AI 的 Key 放哪都不安全

AI 對話需要大模型 API Key，可是一旦寫進 App，**APK 被反編譯就等於公開密鑰**，別人能盜用我的額度。

→ **解決**：自建「中繼服務器」，Key 只存在服務器環境變量；再加一道口令驗證（詳見第 9 頁）。

難題 5 詞庫裏混進「假粵拼」

抽查粵拼詞庫時，發現個別條目的內容**根本不是粵拼格式**，卻安靜地躺在數據裏，等着被 App 標注到學生眼前。

→ **解決**：在驗證腳本加「自我檢查」：用正則規定合法粵拼格式（聲母韻母 + 1-6 聲調），逐條掃描 **27,438 個字**，不合規立即報錯中斷。

難題 6 電腦上的 python 是「假的」

在終端輸入 `python` 毫無反應 —— 原來它指向的是 Microsoft Store 的**佔位程序**，一個看上去像、實際不能用的「假 python」。

→ **解決**：查到真正的 Python 安裝路徑，做一個把完整路徑寫死的啟動腳本。小事，但讓我記住：**命令報錯，先確認它指向哪個程序**。

六個難題，同一套解題方法

STEP A

先查權威來源

卡住時不靠猜、不亂改。先去讀開源詞庫的官方文檔與字典分層，弄清「標準」是甚麼。

STEP B

用數據驗證

改完不靠感覺。寫腳本掃全部數據、用真實語境句子測覆蓋率，讓數字告訴我改對了沒有。

STEP C

分清「誰的錯」

先分清是「數據錯」還是「程序錯」，再決定從哪裏動手 —— 這一條幫我省下了很多瞎改的時間。

回頭看，六個難題其實只分三類：**標準**（1、2、5 —— 該用哪一套字、哪一套格式）、**邊界**（4 —— 密鑰該守在哪一端）、**驗證**（3、6 —— 先分清誰的錯再動手）。三類的共同答案是同一句話：**先查清楚，再改下去**。

同一個「繁體」，標準並不一樣

這是我在項目中下的第一份功夫，也是最容易被忽略的細節。

問題：我起初做簡轉繁時發現，網上現成方案多為台灣標準——把「软件」轉成「軟體」。這在台灣是對的，在香港卻是錯的。香港中學課本寫的是「軟件」。

台灣標準

軟體

✗ 在香港是錯的

網上現成詞庫多數輸出這個

香港標準

軟件

✓ 與香港課本一致

我的 App 輸出這個

我的做法：兩層詞典串接

我研究 OpenCC 的詞典體系，發現它分了好幾層。我不直接用「簡體→繁體」這一層，而是把它串成兩級：

① 簡體 → 傳統繁體 → ② 傳統繁體 → 香港異體 (HKVariants)，再用 Python 腳本把兩層合成詞庫。轉出來的用字，就和香港中學課本一致了。

港式用字對照 (部分)

簡體	香港標準 (我採用)	台灣寫法
软件	軟件	軟體
里	裏	裡
峰	峯	峰
着	着	著

API Key 從不進 App

一條「中繼伺服器 + 口令驗證」的安全防線，並用測試把它鎖住。

問題：AI 功能需要大模型 API Key。若直接寫進 App，APK 一被反編譯，密鑰就等於公開，別人可以盜用我的額度。



兩道鎖

- ① 密鑰不落地 App：只存在伺服器環境變量，App 端拿不到。
- ② 口令驗證：App 請求必須帶上 `X-App-Secret` 請求頭，否則伺服器直接拒絕，防止別人盜用伺服器資源。

用測試證明它真的有效

我寫了自動化測試驗證「無口令 → 403 / 帶口令 → 200」，並把 App 與服務器的 JSON 格式當作「契約」鎖住。

契約測試輸出

```
[OK] /health -> {'ok': True, 'mock': True, 'tts': True}
[OK] /chat (cantonese) 粵語回覆 + notes
[OK] /chat (english) 英語回覆 + notes
[OK] APP_SECRET 防濫用
    無口令 403 / 有口令 200

ALL PASS ✓
契約已驗證
```

伺服器運行中的接口文檔

LanguageCompanion Relay 0.1.0 OKS13

openapi.json

default

- GET /health Health
- POST /chat Chat
- POST /tts Tts

Schemas

- Chatin > Expand all object
- HTTPValidationError > Expand all object
- TtsIn > Expand all object
- ValidationError > Expand all object

為甚麼不是「把 Key 藏進 App」？ — 混淆代碼只能拖延反編譯、無法真正保護；只要密鑰隨 App 發佈出去，就總有被翻出來的一天。把密鑰留在自己控制的服務器上，才是最徹底的做法——這也是我學到的一個安全設計原則。

讓「數據」說話

不靠感覺——詞庫好不好、接口穩不穩，都用自動化驗證給出答案。

電腦先驗，手機後搬

離線邏輯用 Python 寫驗證腳本，與 App 共用同一份 JSON 詞庫。先在電腦跑通，再落到手機，出錯時能分清「數據錯」還是「App 錯」。

用測試鎖住約定

App 與服務器之間的 JSON 格式是「契約」，寫四項自動化測試鎖住它，每次改服務器都重跑，防止改壞 App 依賴的字段。

覆蓋率說話

詞庫質量不靠感覺，用 5 組真實語境句子測試，簡繁與粵拼覆蓋率都達 100% 才算數。

離線模塊驗證

```
=== 離線模塊驗證 (Python) ===
簡繁詞庫：詞組 37,905 + 單字 3,848
粵拼詞庫：27,438 字
真實語境覆蓋率 (105 漢字)：100.0%
```

```
輸入：你想去邊度玩呀？今天天氣好靚
繁體：你想去邊度玩呀？今天天氣好靚
粵拼：你(nei5)想(soeng2)去(heoi3)邊(bin1)度(dou6)...
```

覆蓋率：100%

✓ 離線邏輯驗證通過 (簡繁 + 粵拼)

項目成果一覽

100%

離線覆蓋率

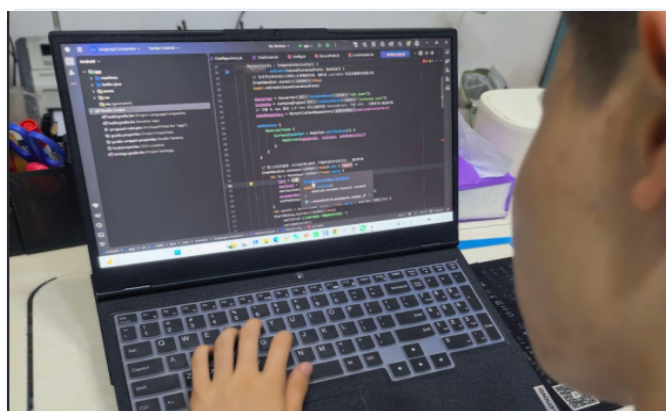
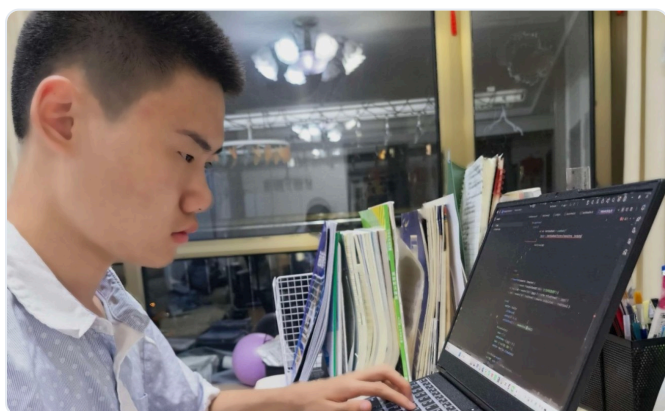
4 / 4

契約測試通過

App 源碼可在 Android Studio 中直接編譯運行；
兩大模塊均通過自動化驗證 (覆蓋率 100% + 契約全通過)。

「做一個能給別人用的工具，先要弄清用戶的『標準』——差一個字，工具對用戶就是錯的。」

這個項目最大的收穫不是學會某個框架，而是明白：工程裏「驗證」比「實現」更重要。我從一個「給自己用的小腳本」開始，一步步把它變成一個有詞庫、有服務器、有測試、有安全設計的完整應用。



項目實作掠影