

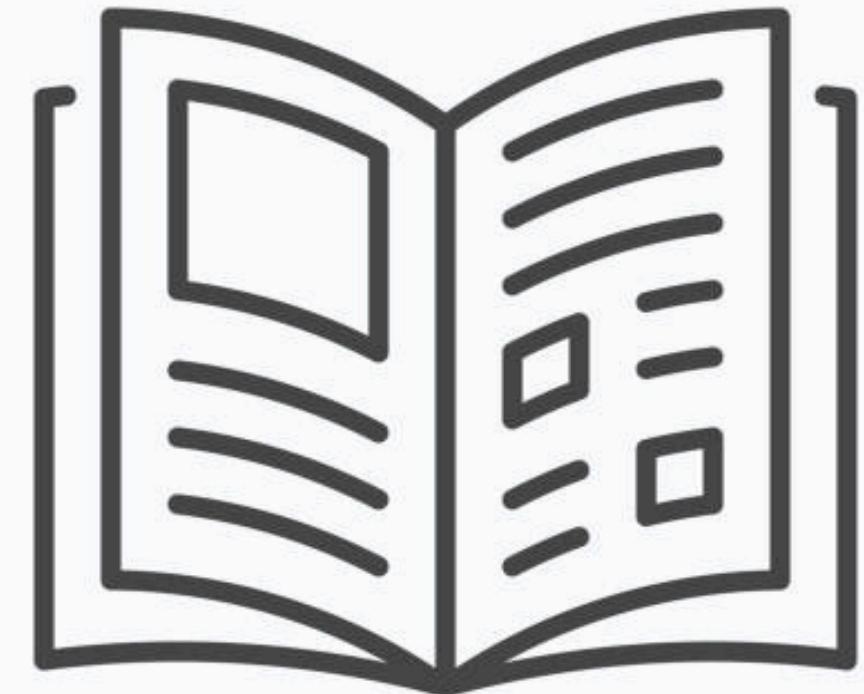
漢語拼音學習遊戲介紹

遊戲名稱：聲韻大師

設計目的：提供遊戲化學習方式

張孔鐸 林依臻 莊凱晴 黃譯霆

目錄



CATALOG

1. 設計動機與開發原因
2. 遊戲整體介紹
3. 遊戲頁面與操作說明
4. 程式介紹
5. 未來改進與發展方向

設計動機



透過遊戲化學習，增加使用者的學習意願。



- 傳統教學以背誦、重複練習為主，學習動機不足

初學者的挑戰

- 拼音枯燥、抽象
- 聲調、聲母韻母難以記憶

漢語拼音是學習中文的重要基礎

開發原因

市面拼音學習遊戲多以"練習題"為主，缺乏：

- 內容無法客製化
- 針對特定拼音難點的強化不足



設計目標

- ❖ 互動性高
- ❖ 學習與娛樂並重
- ❖ 可依程度調整難度
- ❖ 提升拼音學習的趣味性與持續性

遊戲整體介紹

學習內容

- 聲母
- 韻母
- 聲調
- 拼讀練習

學習方式

- 自由練習
- 即時判斷對錯
- 聲音 + 文字輔助學習

適用對象

- 拼音初學者
- 國小學生
- 外國學習中文者

遊戲頁面介紹

畫面內容說明：

- 每日任務
- 智慧推薦學習重點
- 自由練習

設計重點：

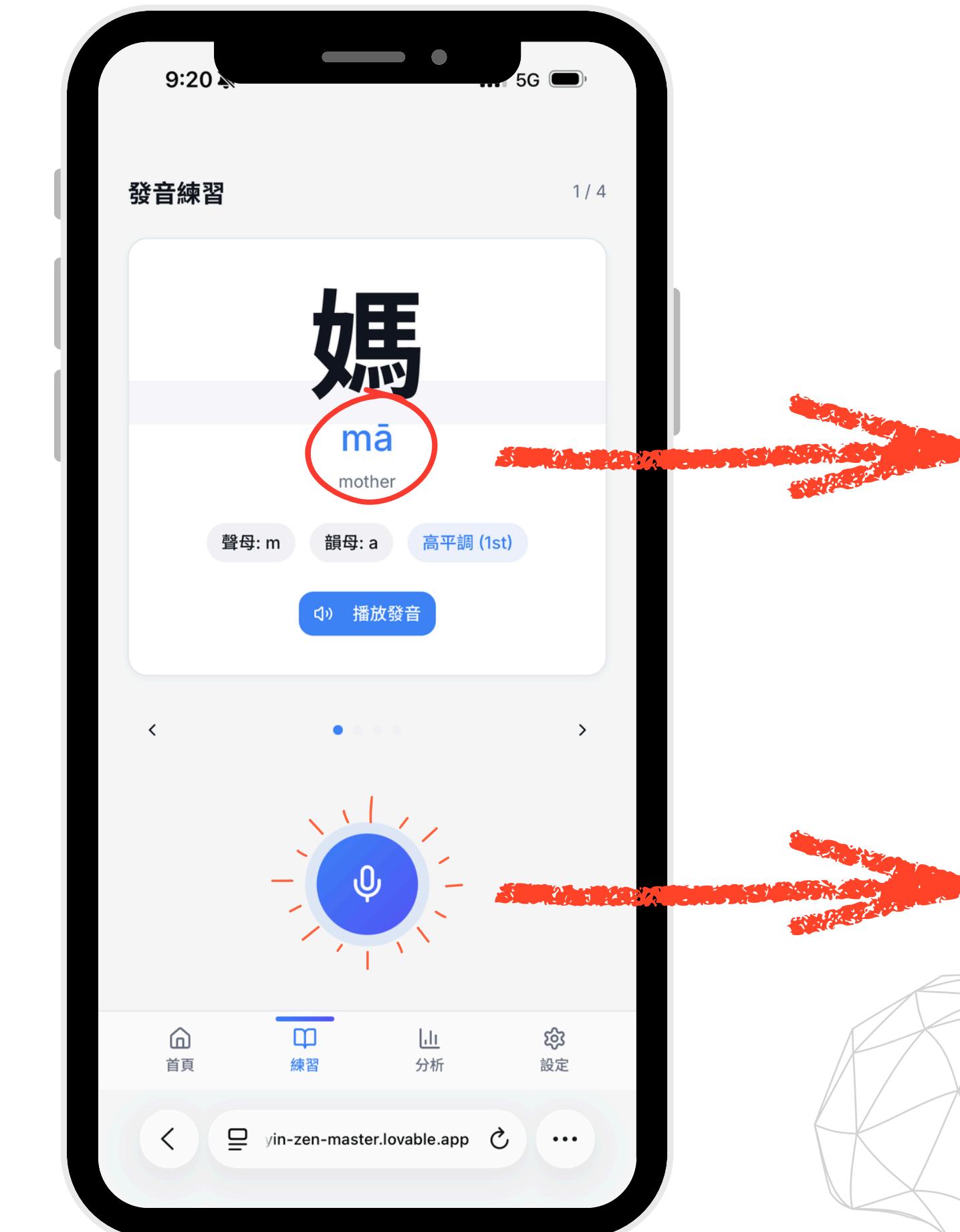
- 介面簡單、直覺
- 適合兒童與初學者操作



遊戲頁面介紹

目標：

- 強化拼音與發音的連結
- 透過重複練習加深記憶



題目顯示拼音

語音示範

遊戲頁面介紹



- 即時顯示正確／錯誤
- 提供學習建議
- 錄音功能自我檢視發音正確性

遊戲頁面介紹



透過學習報表分析常見錯誤
加強相關練習

程式運作介紹

自動準備環境：

程式一執行，會先檢查是否有裝 pygame、sounddevice 等套件，沒有的話會自動安裝。

```
# -----
def install_package(package):
    try:
        subprocess.check_call([sys.executable, "-m", "pip", "install", package])
    except:
        pass

    try:
        import sounddevice as sd
        import scipy.io.wavfile as wav
    except ImportError:
        install_package("sounddevice")
        install_package("scipy")
        install_package("numpy")
        import sounddevice as sd
        import scipy.io.wavfile as wav

    try:
        import speech_recognition as sr
    except ImportError:
        install_package("SpeechRecognition")
        import speech_recognition as sr

    try:
        import edge_tts
        import asyncio
```

程式運作介紹

硬體掃描 (亮點)：

自動測試電腦上所有的麥克風，尋找哪一個可以使用（測試 48000Hz 和 44100Hz 頻率），解決常見的「麥克風報錯」問題。

```
# 核心：自動尋找可用的麥克風
# =====
def find_working_mic():
    print("\n🔍 正在自動掃描可用的麥克風配置...")

    # 常見的取樣率，Realtek 通常強制要求 48000
    test_rates = [48000, 44100]
    devices = sd.query_devices()

    working_config = None

    # 優先嘗試 MME (兼容性最好)，其次是 WASAPI
    # 我們遍歷所有輸入裝置
    for i, device in enumerate(devices):
        if device['max_input_channels'] > 0:
            device_name = device['name']
            # 過濾掉一些無用的裝置
            if "Stereo Mix" in device_name or "立體聲混音" in device_name:
                continue

            print(f"  正在測試裝置 [{i}] {device_name}...", end="")

            for rate in test_rates:
                try:
                    # 嘗試開啟一個極短的錄音流來測試是否會報錯
                    sd.check_input_settings(device=i, channels=1, dtype='int16', samplerate=rate)

                    # 如果沒報錯，嘗試實際錄音 0.1 秒
                    test_rec = sd.rec(int(0.1 * rate), samplerate=rate, channels=1, device=i, dtype='int16')
                    sd.wait()
```

程式運作介紹

出題與示範：

從內建字典中隨機挑選一個字，並使用微軟的 Edge TTS 唴一遍。

```
class TTSPPlayer:
    def __init__(self):
        pygame.mixer.init()
        self.temp_dir = tempfile.gettempdir()

    def generate_and_play(self, character, tone):
        voice = "zh-TW-HsiaoChenNeural"
        rate = "-50%" if tone == '3' else "-20%"
        unique_filename = f"demo_{uuid.uuid4().hex}.mp3"
        audio_path = Path(self.temp_dir) / unique_filename

        try:
            async def _generate():
                communicate = edge_tts.Communicate(character, voice, rate=rate)
                await communicate.save(str(audio_path))

            try:
                loop = asyncio.get_running_loop()
                nest_asyncio.apply()
                loop.run_until_complete(_generate())
            except RuntimeError:
                loop = asyncio.new_event_loop()
                asyncio.set_event_loop(loop)
                loop.run_until_complete(_generate())
                loop.close()

            if audio_path.exists():
                pygame.mixer.music.load(str(audio_path))
                pygame.mixer.music.play()
                while pygame.mixer.music.get_busy():

        
```

程式運作介紹

錄音與優化：

程式會倒數 3 秒，錄下聲音。如果聲音太小，程式會透過數學運算把音量放大，避免辨識失敗。錄完馬上回放，確認有沒有錄清楚。

```
max_volume = np.max(np.abs(myrecording))
print(f"录音原始音量: {max_volume}")

if max_volume < 100:
    print("⚠️ 警告：音量過小，可能是選到了沒聲音的麥克風！")
    return None

# 自動增益
target_peak = 20000
if max_volume < target_peak:
    print(f"🔊 自動放大音量 ({target_peak/max_volume:.1f}倍)...")
    amplified = myrecording.astype(float) * (target_peak / max_volume)
    amplified = np.clip(amplified, -32768, 32767)
    myrecording = amplified.astype(np.int16)

# --- 錄音回放 ---
print("▶ 正在回放您的錄音（請確認是否清楚）...")
sd.play(myrecording, self.fs)
sd.wait()
print("播放結束")

output_file = self.temp_dir / "user_recording.wav"
wav.write(output_file, self.fs, myrecording)
return str(output_file)

except Exception as e:
    print(f"录音錯誤: {e}")
    return None

def assess_pronunciation(self, audio_file, target_char):
    if not self.recognizer:
        return {"error": "辨識模組未載入"}

    print("🎙 正在辨識中...")
```

程式運作介紹

辨識與評分：

將錄音檔傳送到 Google 語音辨識，將聽到的內容轉成文字，再跟題目的字比對是否正確。

```
def assess_pronunciation(self, audio_file, target_char):
    if not self.recognizer:
        return {"error": "辨識模組未載入"}

    print("🔊 正在辨識中...")
    try:
        with sr.AudioFile(audio_file) as source:
            audio_data = self.recognizer.record(source)

        recognized_text = self.recognizer.recognize_google(audio_data, language="zh-TW")

        clean_rec = recognized_text.strip().replace(" ", "")
        clean_tar = target_char.strip()
        is_correct = (clean_rec == clean_tar)

        return {
            "recognized_text": clean_rec,
            "is_correct": is_correct,
            "target": clean_tar
        }
    except sr.UnknownValueError:
        return {"error": "無法辨識 (聲音模糊或發音不標準)"}
    except sr.RequestError:
        return {"error": "網路連線問題"}
    except Exception as e:
        return {"error": f"錯誤: {e}"}
```

Demo

The screenshot shows a Python development environment with the following components:

- Code Editor:** A dark-themed code editor window titled "完成品 2.py" containing Python code. The code includes imports for sys, subprocess, time, random, os, uuid, Path, tempfile, re, and numpy. It defines a function `install_package` for automatically installing packages via pip. It also lists required packages in `needed_packages` and attempts to import them.
- Variable Explorer:** A panel titled "No variables to show" indicating no global variables are currently listed.
- Console I/A:** A terminal window showing Chinese text related to audio processing, including "正在自動掃描可用的麥克風配置..." (Scanning available microphones) and "正在測試裝置 [0] Microsoft 音效對應表 - Input... 成功! (48000Hz)" (Testing device [0] Microsoft Sound Driver - Input... Success! (48000Hz)).
- IPython Console:** A terminal window showing the command "漢語拼音發音練習 (錄音回放版)" (Hanyu Pinyin Pronunciation Practice (Recording and Playback Version)) and a list of menu items: "1. 開始出題 2. 離開".

未來改進與發展

1. 引入拼音比對：

現況：比對漢字 (`if "雨" == "與"`)，容易因同音字誤判。

改進：引入 `pypinyin` 套件。將使用者的語音辨識結果轉回拼音（例如 `yu3`），再與題目的拼音進行比對。

2. 離線辨識模型 (Offline AI)：

現況：依賴 Google Web API，速度受網路影響，且有請求限制。

改進：整合 OpenAI Whisper (Small/Tiny model)，能實現完全離線執行。

3. 聲調精細分析：

現況：只知道對錯，不知道「哪裡錯」。

改進：利用 `librosa` 或 `parselmouth` (Praat 的 Python 介面) 分析音頻的 FO (基頻) 曲線。

畫出使用者的聲調軌跡與標準音軌跡進行疊加對比，讓使用者「看見」自己的三聲有沒有壓下去。

參考資料

- 語言科技 標音易2普通話發聲插件使用手冊
- Python如何使用pypinyin实现中文拼音转换？
- 轻松检测麦克风功能：使用Python的sounddevice和soundfile库
- 口播神器,基于Edge,微软TTS(text-to-speech)文字转语音免费开源库edge-tts语音合成实践(Python3.10).
- 語言科技 《標音易》漢語拼音標注系統 使用手冊
- 解决报错：Invalid number of channels [PaErrorCode -9998]
- 全字庫 CNS11643 (2024)
- Python中的中文拼音转换库：pypinyin与xpinyin
- 新华字典在线查字【澳典：用规范汉字，筑文字功底】
- Get windows devices names with python and sounddevice
- [Sounddevice] Error opening InputStream: Invalid number of channels [PaErrorCode -9998]

參考資料

- Mic not working properly- Audio device number changing between sessions
- PsychPortAudio only lets me use sampling frequency 48000
- python-pinyin
- pinyin-data
- python-sounddevice
- hanzipinyin
- Google Gemini
- Google NotebookLM
- Claude
- perplexity
- Lovable
- Eigma



Thank You