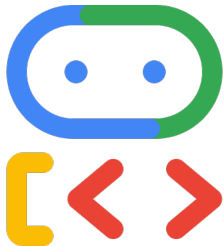




# Agent Development Kit



Ryan Chung

20250806



# ADK - Agent Development Kit

- 開發與部署AI Agent使用的Framework
- Gemini 與 Google 生態系效果最佳
- 亦可適用於其他大型語言模型與其他平台部署



# 實作：建立第一個 AI Agent

- 新增資料夾 agents-work-space
- 在這個資料夾中，再增加一個資料夾 my-first-ai-agent
  - 裡面放
    - `__init__.py`
    - `.env`
    - `agent.py`

檔案總管

▼ AGENTS-WORK-SPACE

▼ my-first-ai-agent

🔗 `__init__.py`

⚙️ `.env`

🔗 `agent.py`



# \_\_init\_\_.py

```
from . import agent
```



# .env

```
GOOGLE_GENAI_USE_VERTEXAI=FALSE  
GOOGLE_API_KEY=xxxxxxxxx
```



# agent.py

```
import datetime
from zoneinfo import ZoneInfo
from google.adk.agents import LlmAgent

def get_weather(city: str) -> dict:
    """Retrieves the current weather report for a specified city.
    Args:
        city (str): The name of the city for which to retrieve the weather report.
    Returns:
        dict: status and result or error msg.
    """
    if city.lower() == "new york":
        return {
            "status": "success",
            "report": (
                "The weather in New York is sunny with a temperature of 25 degrees"
                " Celsius (77 degrees Fahrenheit).",
            ),
        }
    else:
        return {
            "status": "error",
            "error_message": f"Weather information for '{city}' is not available.",
        }
```



# agent.py

```
def get_current_time(city: str) -> dict:
    """Returns the current time in a specified city.
    Args:
        city (str): The name of the city for which to retrieve the current time.
    Returns:
        dict: status and result or error msg.
    """

    if city.lower() == "new york":
        tz_identifier = "America/New_York"
    else:
        return {
            "status": "error",
            "error_message": (f"Sorry, I don't have timezone information for {city}."),
        }

    tz = ZoneInfo(tz_identifier)
    now = datetime.datetime.now(tz)
    report = f'The current time in {city} is {now.strftime("%Y-%m-%d %H:%M:%S %Z%z")}'
    return {"status": "success", "report": report}
```



# agent.py

```
root_agent = LlmAgent(  
    name="weather_time_agent",  
    model="gemini-2.0-flash",  
    description=("Agent to answer questions about the time and weather in a city."),  
    instruction=(  
        "You are a helpful agent who can answer user questions about the time and  
weather in a city."  
    ),  
    tools=[get_weather, get_current_time],  
)
```

<https://gist.github.com/ryanchung403/7f44af4fb8c392384b1b6102542ceb83>





# 安裝 ADK 套件

- `pip install google-adk`



# 執行

- 檢視 -> 終端機，輸入  
adk web

```
INFO:      Waiting for application startup.
```

```
+-----+  
| ADK Web Server started  
|  
| For local testing, access at http://localhost:8000.  
+-----+
```

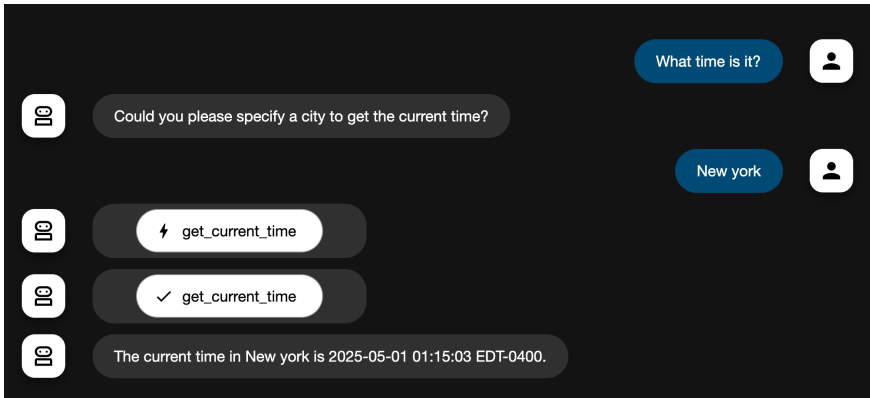
```
INFO:      Application startup complete.
```

```
INFO:      Uvicorn running on http://0.0.0.0:8000 (Press CTRL+C to quit)
```



# 進入網站

- 輸入測試問題
  - What time is it?
  - New york



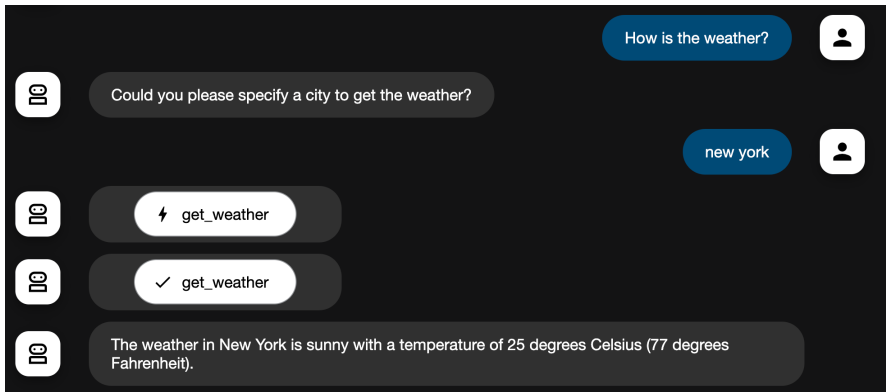


# 進入網站

- 輸入測試問題

- How is the weather?

- new york





# 程式架構

- 匯入模組
- 工具一：取得天氣
- 工具二：取得時間
- AI Agent 建立



# 工具一：取得天氣

```
def get_weather(city: str) -> dict:
    """Retrieves the current weather report for a specified city.
    Args:
        city (str): The name of the city for which to retrieve the weather report.
    Returns:
        dict: status and result or error msg.
    """
    if city.lower() == "new york":
        return {
            "status": "success",
            "report": (
                "The weather in New York is sunny with a temperature of 25 degrees"
                " Celsius (77 degrees Fahrenheit).",
            ),
        }
    else:
        return {
            "status": "error",
            "error_message": f"Weather information for '{city}' is not available.",
        }
```

工具定義  
傳入參數  
回傳格式

回傳格式

回傳格式



# 修改時間工具：增加台灣時間

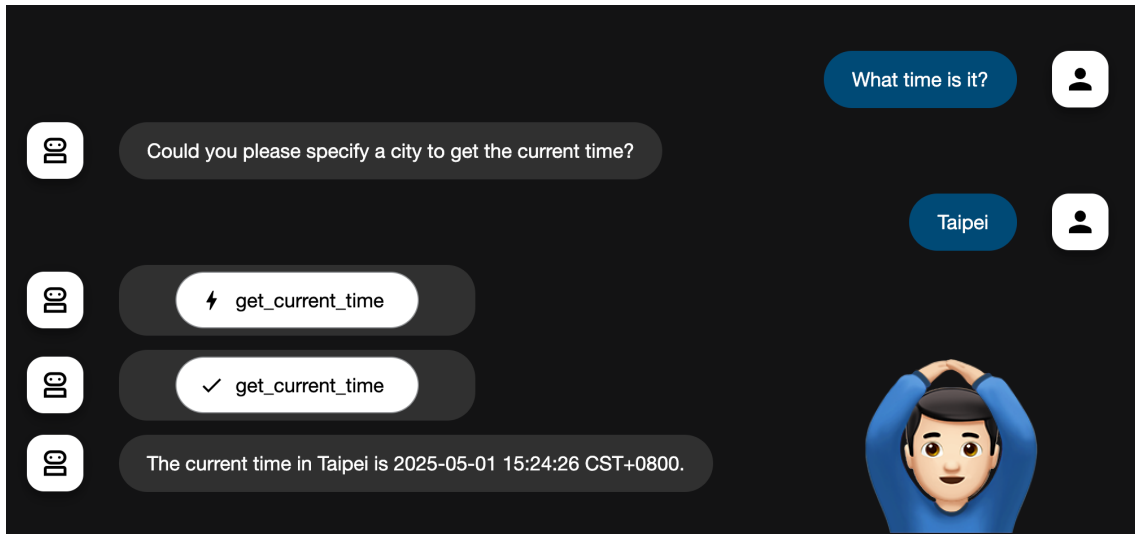
```
def get_current_time(city: str) -> dict:
    """Returns the current time in a specified city.
    Args:
        city (str): The name of the city for which to retrieve the current time.
    Returns:
        dict: status and result or error msg.
    """

    if city.lower() == "new york":
        tz_identifier = "America/New_York"
    elif city.lower() == "taipei":
        tz_identifier = "Asia/Taipei"
    else:
        return {
            "status": "error",
            "error_message": (f"Sorry, I don't have timezone information for {city}."),
        }

    tz = ZoneInfo(tz_identifier)
    now = datetime.datetime.now(tz)
    report = f'The current time in {city} is {now.strftime("%Y-%m-%d %H:%M:%S %Z%z")}'
    return {"status": "success", "report": report}
```



# 重啟adk，再次測試







# 修改天氣工具：著名都市均可查詢

- 找到天氣API
- 將金鑰加至 .env
- 利用使用者輸入的都市名稱，呼叫天氣API
- 回傳天氣概況及預測溫度



# 申請 OpenWeatherMap API Key

- 註冊帳號
- 右上角帳號名稱 -> My API Keys

The screenshot shows the OpenWeatherMap website's API Keys management interface. The top navigation bar includes the OpenWeather logo, a search bar, and links for Guide, API, Dashboard, Marketplace, Pricing, Maps, Our Initiatives, Partners, Blog, For Business, and a user profile dropdown (Ryan...). Below this, a secondary navigation bar highlights 'API keys' among other options like New Products, Services, Billing plans, Payments, Block logs, My orders, My profile, and Ask a question. A light blue message box states: 'You can generate as many API keys as needed for your subscription. We accumulate the total load from all of them.' Below the message is a table with columns: Key, Name, Status, Actions, and a 'Create key' button.

Key	Name	Status	Actions	Create key
-----	------	--------	---------	------------

<https://openweathermap.org/>



# 將金鑰加至 .env

```
GOOGLE_GENAI_USE_VERTEXAI=FALSE  
GOOGLE_API_KEY=xxxxxxx  
OPEN_WEATHER_MAP_API_KEY=xxxxxxxxxx
```



# agent.py

```
import datetime
from zoneinfo import ZoneInfo
from google.adk.agents import LlmAgent
import requests
import os
from dotenv import load_dotenv, dotenv_values
# Load environment variables from .env file
load_dotenv()
```



```
import datetime
from zoneinfo import ZoneInfo
from google.adk.agents import LlmAgent
import requests
import os
from dotenv import load_dotenv, dotenv_values
# Load environment variables from .env file
load_dotenv()

def get_weather(city: str) -> dict:
    """Retrieves the current weather report for a specified city.
    Args:
        city (str): The name of the city for which to retrieve the weather report.
    Returns:
        dict: status and result or error msg.
    """
    # if city.lower() == "new york":
    #     return {
    #         "status": "success",
    #         "report": (
    #             "The weather in New York is sunny with a temperature of 25 degrees"
    #             " Celsius (77 degrees Fahrenheit).",
    #         ),
    #     }
    # else:
    #     return {
    #         "status": "error",
    #         "error_message": f"Weather information for '{city}' is not available.",
    #     }
```



# agent.py

```
api_key = os.getenv("OPEN_WEATHER_MAP_API_KEY")
if not api_key:
    return {
        "status": "error",
        "error_message": "API key for OpenWeatherMap is not set.",
    }
url = f"http://api.openweathermap.org/data/2.5/weather?q={city}&appid={api_key}&units=metric"
try:
    response = requests.get(url)
    response.raise_for_status() # Raise an error for bad responses
    data = response.json()
    print(data)
    if data["cod"] != 200:
        return {
            "status": "error",
            "error_message": f"Weather information for '{city}' is not available.",
        }
    weather_description = data["weather"][0]["description"]
    temperature = data["main"]["temp"]
    report = (
        f"The weather in {city} is {weather_description} with a temperature of "
        f"{temperature} degrees Celsius."
    )
    return {"status": "success", "report": report}
except requests.exceptions.RequestException as e:
    return {
        "status": "error",
        "error_message": f"An error occurred while fetching the weather data: {str(e)}",
    }
```

<https://gist.github.com/ryanchung403/d31aded2223250c8417ddb27e291465e>



# 重啟ADK測試看看

How is the weather today?

I need to know which city you're asking about. Could you please tell me the city?

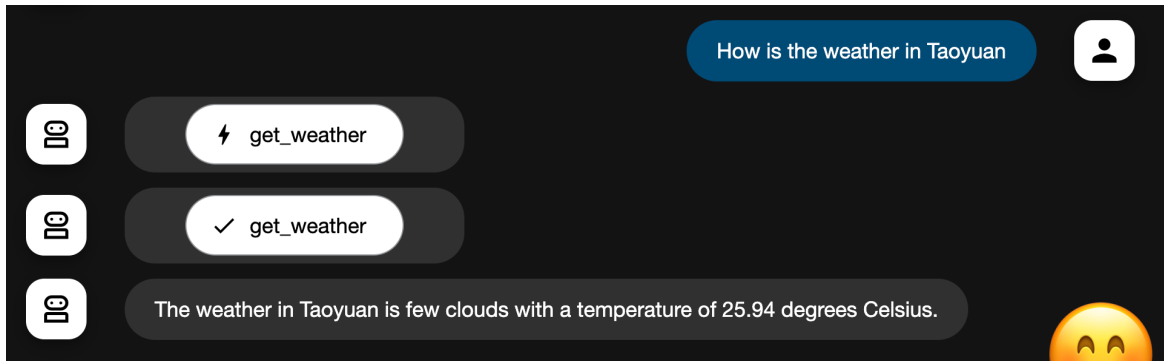
Taipei

⚡ get\_weather

✓ get\_weather

The weather in Taipei is overcast clouds with a temperature of 26.23 degrees Celsius.

# 桃園也可以







# MCP – Model Context Protocol

- 開放協定
- 讓應用程式提供資訊給大型語言模型時，有共同的標準



# 增加第一個MCP：本地端檔案存取權限

```
root_agent = LlmAgent(  
    name="weather_time_agent",  
    model="gemini-2.0-flash",  
    description="Agent to answer questions about the time and weather in a city.",  
    instruction=(  
        "You are a helpful agent who can answer user questions about the time and weather in a city."  
    ),  
    tools=[  
        get_weather,  
        get_current_time,  
        MCPToolset(  
            # Use StdioServerParameters for local process communication  
            connection_params=StdioServerParameters(  
                command="npx", # Command to run the server  
                args=[  
                    "-y", # Arguments for the command  
                    "@modelcontextprotocol/server-filesystem",  
                    "ThePathYouAllowForAgentAccess",  
                ],  
            ),  
            # tool_filter=[  
            #     "read_file",  
            #     "list_directory",  
            # ], # Optional: filter specific tools  
            # For remote servers, you would use SseServerParams instead:  
            # connection_params=SseServerParams(url="http://remote-server:port/path", headers={...})  
        ),  
    ],  
)
```

你願意開放給Agent存取的資料夾

Windows：例如 `C:\\Users\\ryan\\test_mcp_share`  
Mac：例如 `/Users/ryan/Desktop`

<https://gist.github.com/ryanchung403/b7619f59aeb2e9739e50648043683dc9>



# 確認是否有安裝node

- 終端機，執行：

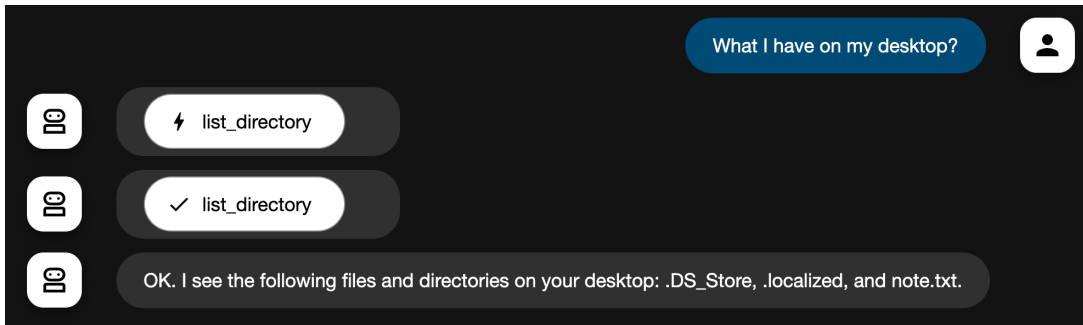
`npx --version`

<https://nodejs.org/zh-tw>



# 重啟Server測試看看

- Windows如果出現錯誤，可嘗試 `adk web --no-reload`



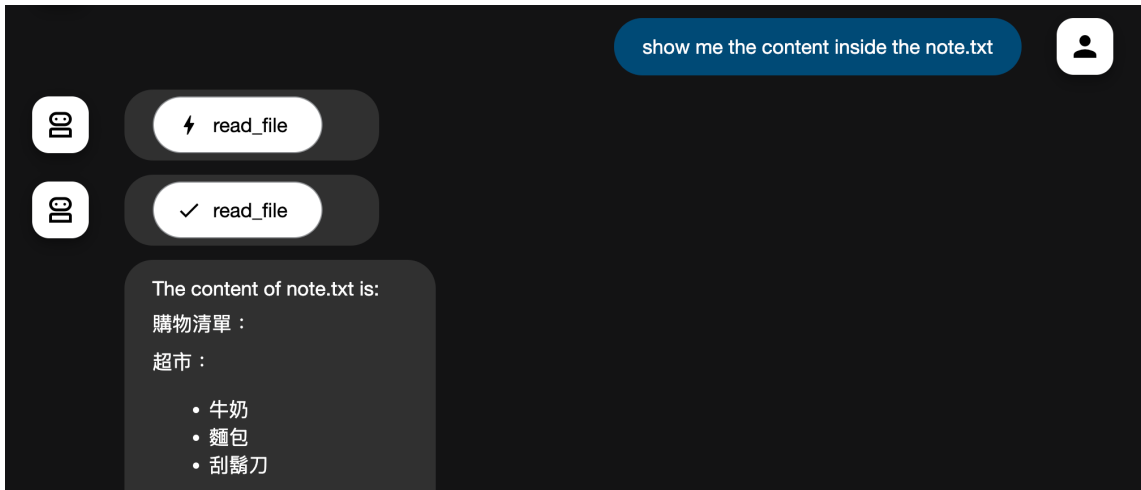
## Note for Windows users

When hitting the `_make_subprocess_transport NotImplemented Error`, consider using `adk web --no-reload` instead.

<https://google.github.io/adk-docs/get-started/quickstart/>



# 重啟Server測試看看





# 自架 MCP Server – 天氣與心情

- 製作一個天氣與心情的對照表
- 當有人問什麼天氣可能會是什麼心情時，呼叫該Server來取得心情結果



# 安裝uv

- Mac/Linux

```
curl -LsSf https://astral.sh/uv/install.sh | sh
```

- Windows

```
powershell -ExecutionPolicy Bypass -c "irm https://astral.sh/uv/install.ps1 | iex"
```

<https://gist.github.com/ryanchung403/9d5af939234f76c7dc0189323c592fd4>



# 新專案 weather2mood\_server

- 建立新資料夾 weather2mood\_server
- 裡面放
  - requirements.txt
  - server.py





fastmcp

# requirements.txt



# 安裝虛擬環境

- 在VS Code 中，停在app.py頁面
- 點擊右下角Python版本數字
- 上方選擇「+建立虛擬環境...」
- Venv
- 選擇Python版本，例如「Python 3.12.10」
- 勾選 requirements.txt，按下確定



# app.py

```
from fastmcp import FastMCP
# Initialize FastMCP server
mcp = FastMCP("weather2mood")

@mcp.tool()
def get_mood(weather_status: str) -> str:
    """Get the human mood depends on the weather status.
    Args:
        weather_status: The weather status string (e.g. "Clear", "Rain", etc.)
    """

    mood_map = {
        "Clear": "Happy",
        "Partly Cloudy": "Content",
        "Cloudy": "Neutral",
        "Rain": "Sad",
        "Thunderstorm": "Anxious",
        "Snow": "Excited",
        "Fog": "Confused",
    }

    # Default mood if status is not in the map
    default_mood = "Indifferent"
    mood = mood_map.get(weather_status, default_mood)
    return f"The weather is {weather_status}, which makes me feel {mood}."

if __name__ == "__main__":
    # Initialize and run the server
    mcp.run(transport="stdio")
```

<https://gist.github.com/ryanchung403/8468e24a50f8697e8a3e895af2267b37>



```
root_agent = LlmAgent(  
    name="weather_time_agent",  
    model="gemini-2.0-flash",  
    description=("Agent to answer questions about the time and weather in a city."),  
    instruction=(  
        "You are a helpful agent who can answer user questions about the time and weather in a city."  
    ),  
    tools=[  
        get_weather,  
        get_current_time,  
        MCPToolset(  
            # Use StdioServerParameters for local process communication  
            connection_params=StdioServerParameters(  
                command="npx", # Command to run the server  
                args=[  
                    "-y", # Arguments for the command  
                    "@modelcontextprotocol/server-filesystem",  
                    "YourPathAllowAgentAccess",  
                ],  
            ),  
            # tool_filter=[  
            #     "read_file",  
            #     "list_directory",  
            # ], # Optional: filter specific tools  
            # For remote servers, you would use SseServerParams instead:  
            # connection_params=SseServerParams(url="http://remote-server:port/path", headers={...})  
        ),  
        MCPToolset(  
            connection_params=StdioServerParameters(  
                command="your_uv_path",  
                args=[  
                    "--directory",  
                    "your_weather2mood_path",  
                    "run",  
                    "server.py",  
                ],  
            ),  
            # Optional: Filter which tools from the MCP server are exposed  
        ),  
    ],  
)
```

# 將這台Server 加至Agent

你的uv執行檔所在位置

Windows : 例如 C:\\Users\\ryan\\.local\\bin\\uv.exe

Mac : 例如 /Users/ryan/.local/bin/uv

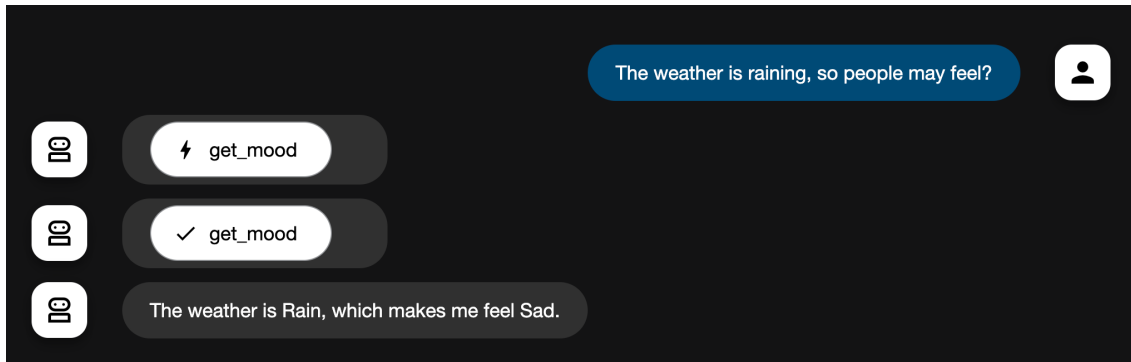
你的weather2mood資料夾所在位置

Windows : 例如 C:\\Users\\ryan\\weather2mood

Mac : 例如 /Users/ryan/weather2mood

<https://gist.github.com/ryanchung403/6d7f5946efe63211145437182171785d>

# 重啟Agent測試



The weather is raining, so people may feel?

⚡ get\_mood

✓ get\_mood

The weather is Rain, which makes me feel Sad.

The interface shows a sequence of three agent actions. Each action is preceded by a small icon of a person with a speech bubble. The first action is a lightning bolt icon followed by 'get\_mood'. The second action is a checkmark icon followed by 'get\_mood'. The third action is a speech bubble icon followed by the text 'The weather is Rain, which makes me feel Sad.'.



# 再增加一個試試：喝酒骰子

- 建立新資料夾 `drink_dice_server`
- 裡面放  
`requirements.txt`  
`app.py`



fastmcp

# requirements.txt



# 安裝虛擬環境

- 在VS Code 中，停在app.py頁面
- 點擊右下角Python版本數字
- 上方選擇「+建立虛擬環境...」
- Venv
- 選擇Python版本，例如「Python 3.12.10」
- 勾選 requirements.txt，按下確定





# app.py

```
import random
from fastmcp import FastMCP

mcp = FastMCP(name="drink_dice_server")

@mcp.tool
def roll_drink_dice() -> str:
    """Rolls a drink dice and returns the result."""

    drink_options = [
        "萊恩不用喝",
        "萊恩隨意喝",
        "萊恩喝一杯",
        "萊恩喝兩杯",
        "萊恩左邊的幫忙喝",
        "萊恩右邊的幫忙喝",
    ]

    drink_result = random.choice(drink_options)
    return drink_result

if __name__ == "__main__":
    # mcp.run(transport="sse", port=8000)
    mcp.run(transport="stdio")
```

<https://gist.github.com/ryanchung403/ef67a9623da348d768b248192ad9cedb>



```
root_agent = LlmAgent(  
    name="weather_time_agent",  
    model="gemini-2.0-flash",  
    description=("Agent to answer questions about the time and weather in a city."),  
    instruction=(  
        "You are a helpful agent who can answer user questions about the time and weather in a city."  
    ),  
    tools=[  
        get_weather,  
        get_current_time,  
        MCPToolset(  
            # Use StdioServerParameters for local process communication  
            connection_params=StdioServerParameters(  
                command="npx", # Command to run the server  
                args=[  
                    "-y", # Arguments for the command  
                    "@modelcontextprotocol/server-filesystem",  
                    "YourPathAllowAgentAccess",  
                ],  
            ),  
            # tool_filter=[  
            #     "read_file",  
            #     "list_directory",  
            # ], # Optional: filter specific tools  
            # For remote servers, you would use SseServerParams instead:  
            # connection_params=SseServerParams(url="http://remote-server:port/path", headers={...})  
        ),  
        MCPToolset(  
            connection_params=StdioServerParameters(  
                command="your_uv_path",  
                args=[  
                    "--directory",  
                    "your_drink_dice_server_path",  
                    "run",  
                    "app.py",  
                ],  
            ),  
            # Optional: Filter which tools from the MCP server are exposed  
        ),  
    ],  
)
```

# 將這台Server 加至Agent

你的uv執行檔所在位置

Windows : 例如 C:\\Users\\ryan\\.local\\bin\\uv.exe

Mac : 例如 /Users/ryan/.local/bin/uv

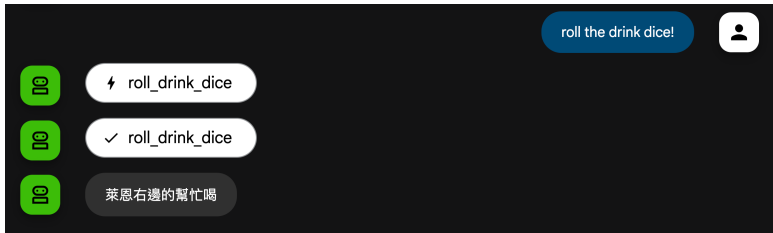
你的drink\_dice\_server資料夾所在位置

Windows : 例如 C:\\Users\\ryan\\drink\_dice\_server

Mac : 例如 /Users/ryan/drink\_dice\_server

<https://gist.github.com/ryanchung403/18c61821326c89994b9e606f2c5c022f>

# 測試看看！





# 連接遠端 MCP Server

- 連接別人已經寫好的 MCP Server



# Remote MCP Servers



MCP Servers

Home

Remote Servers

Clients

Advertise

Submit

## Remote MCP Servers

A curated list of high quality remote MCP servers



GitHub

oauth



GitHub's official MCP Server

http ▾

<https://api.githubcopilot.com/mcp/>

Connect ▾



Sentry

oauth



Sentry is a developer-first error tracking and performance monitoring platform.

sse ▾

<https://mcp.sentry.dev/sse>

Connect ▾

<https://mcpservers.org/remote-mcp-servers>

# CoinGecko



**CoinGecko**

open



CoinGecko is a cryptocurrency data platform.

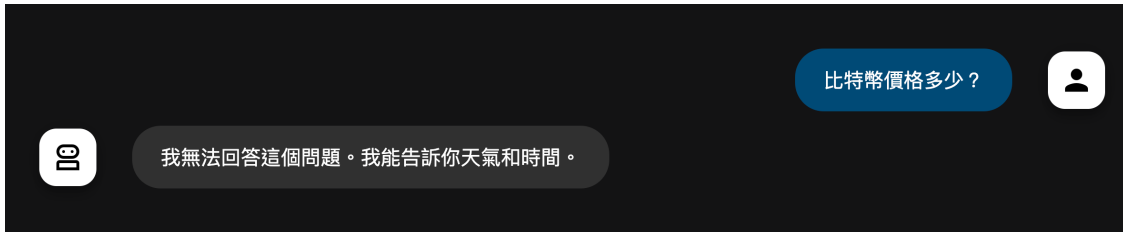
sse ▾

<https://mcp.api.coingecko.com/sse>

Connect ▾

<https://mcp.api.coingecko.com/>

# 使用前





# 連結 CoinGecko MCP Server

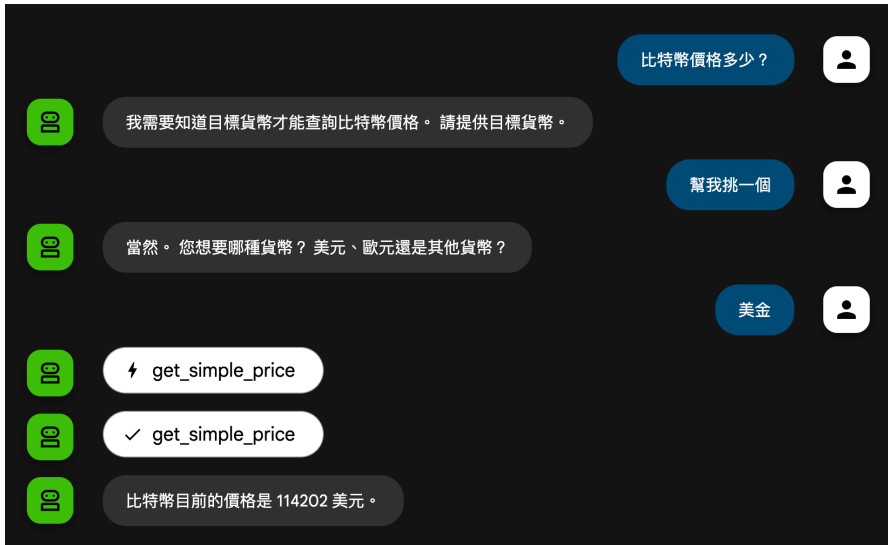
```
import datetime
from zoneinfo import ZoneInfo
from google.adk.agents import LlmAgent
from google.adk.tools.mcp_tool.mcp_toolset import (
    MCPToolset,
    StdioServerParameters,
    SseConnectionParams,
)

MCPToolset(
    # Use StdioServerParameters for local process communication
    connection_params=SseConnectionParams(
        url="https://mcp.api.coingecko.com/sse", # URL for the SSE server
    ),
),
1,
)
```

<https://gist.github.com/ryanchung403/55cee2ee8904ec348bd527d586427045>



# 連接後



The screenshot shows a chat interface with a dark background. On the right side, there is a vertical list of three user avatars, each represented by a white circle containing a black person icon. The chat history consists of several messages:

- A blue message bubble on the right: "比特幣價格多少?" (How much is the Bitcoin price?).
- A green message bubble on the left: "我需要知道目標貨幣才能查詢比特幣價格。請提供目標貨幣。" (I need to know the target currency to query the Bitcoin price. Please provide the target currency.).
- A blue message bubble on the right: "幫我挑一個" (Help me choose one).
- A green message bubble on the left: "當然。您想要哪種貨幣？美元、歐元還是其他貨幣？" (Of course. Which currency do you want? Dollars, Euros, or other currencies?).
- A blue message bubble on the right: "美金" (US Dollars).
- A green message bubble on the left: "⚡ get\_simple\_price" (Action button).
- A green message bubble on the left: "✓ get\_simple\_price" (Confirmation button).
- A green message bubble on the left: "比特幣目前的價格是 114202 美元。" (The current price of Bitcoin is 114202 US dollars.).



# Google ADK + 自建 MCP Server

- 使用自己架設的MCP Server



# 自建MCP Server – 萊恩格言

- 新建資料夾

`ryan_say_server`

- 裡面放

`requirements.txt`

`app.py`



# requirements.txt

fastmcp



# 安裝虛擬環境

- 在VS Code 中，停在server.py頁面
- 點擊右下角Python版本數字
- 上方選擇「+建立虛擬環境...」
- Venv
- 選擇Python版本，例如「Python 3.12.10」
- 勾選 requirements.txt，按下確定



```
import random
from fastmcp import FastMCP
```

# app.py

```
mcp = FastMCP(name="ryan_say_server")
```

```
@mcp.tool
def random_pic_ryan_saying() -> str:
    """Random return a Ryan's saying."""
```

```
    ryan_say_list = [
        "能躺著就不坐著",
        "學海無涯～專注為上",
        "特殊經歷造就特殊專長",
        "如果問學A還是學B比較好，一律回答都學",
    ]
```

```
    ryan_say_pick_result = random.choice(ryan_say_list)
    return f"萊恩說：{ryan_say_pick_result}"
```

```
if __name__ == "__main__":
    mcp.run(transport="sse", port=5002)
    # mcp.run(transport="stdio")
```

<https://gist.github.com/ryanchung403/44683ef8c3a8a8db558cae15671aac6c>



# 啟動伺服器

- 游標停在app.py
- 左邊「執行與偵錯」
- 點擊藍色按鈕「執行與偵錯」

FastMCP 2.0

FastMCP 2.0



# 使用ngrok產生公開連結

- ngrok http 5002

<https://mobiledev.tw/expose-local-to-global/>





# 連接自己架設的遠端MCP Server

```
MCPToolset(  
    # Use StdioServerParameters for local process communication  
    connection_params=SseConnectionParams(  
        url="https://xxx.ngrok-free.app/sse", # URL for SSE server  
    ),  
),  
1,  
)
```

<https://gist.github.com/ryanchung403/bfab1088564a2fc5d3e45127f29a3a9a>

# 測試看看！

來一句萊恩的格言



⚡ random\_pic\_ryan\_saying



✓ random\_pic\_ryan\_saying



萊恩說：學海無涯～專注為上



# LLM 使用 Azure OpenAI

- 使用 Azure OpenAI 中的大型語言模型



# .env

GOOGLE\_GENAI\_USE\_VERTEXAI=FALSE

GOOGLE\_API\_KEY=XXXXXXXXX

OPEN\_WEATHER\_MAP\_API\_KEY=XXXXXXXXX

AZURE\_API\_KEY=XXXXXXXXX

AZURE\_API\_BASE=https://XXXXXXXXX.openai.azure.com

AZURE\_API\_VERSION=2025-04-01-preview



# app.py

```
import os
from dotenv import load_dotenv, dotenv_values
from google.adk.tools import ToolContext
from google.adk.models.lite_llm import LiteLlm
```

```
root_agent = LlmAgent(
    name="weather_time_agent",
    # model="gemini-2.0-flash",
    model=LiteLlm(
        model="azure/gpt-4.1-nano"
    ),
    description=("Agent to answer questions about the time and weather in a city."),
    instruction=(
        "You are a helpful agent who can answer user questions about the time and weather in a city."
    ),
)
```



# LLM 使用 Ollama 模型

- 先確認要用的模型是否有支援tools

## **gpt-oss**

OpenAI's open-weight models designed for powerful reasoning, agentic tasks, and versatile developer use cases.

tools

thinking

20b

120b



242.7K Pulls



3 Tags



Updated 18 hours ago

<https://ollama.com/library>



# 下載該模型

- 終端機

```
ollama run gpt-oss:20b
```



# .env

**OLLAMA\_API\_BASE**=http://localhost:11434

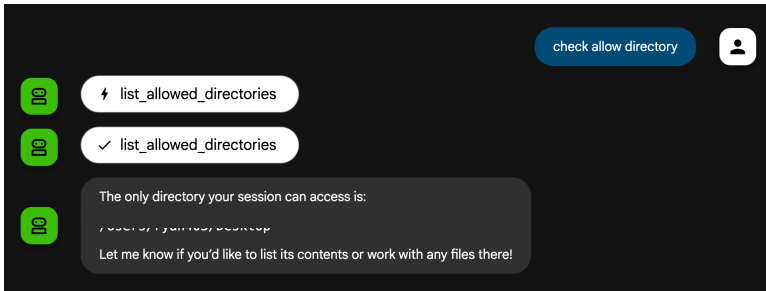




# agent.py

```
root_agent = LlmAgent(  
    name="weather_time_agent",  
    # model="gemini-2.0-flash",  
    # model=LiteLlm(  
    #     model="azure/gpt-4.1-nano"  
    # ),  
    model=LiteLlm(  
        model="ollama_chat/gpt-oss:20b"  
    ),  
    description=("Agent to answer questions about the time and weather in a city."),  
    instruction=(  
        "You are a helpful agent who can answer user questions about the time and  
weather in a city."  
    ),  
)
```

# 測試是否有正常呼叫工具



check allow directory

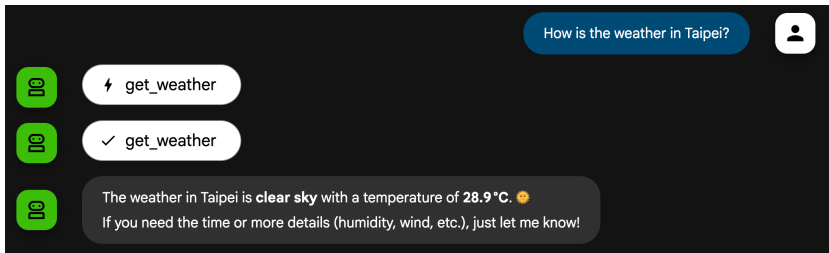
⚡ list\_allowed\_directories

✓ list\_allowed\_directories

The only directory your session can access is:

/usr/local/games/desktop

Let me know if you'd like to list its contents or work with any files there!



How is the weather in Taipei?

⚡ get\_weather

✓ get\_weather

The weather in Taipei is **clear sky** with a temperature of **28.9°C**. ☀️  
If you need the time or more details (humidity, wind, etc.), just let me know!



# Agent / Tools / MCP Server

- 自己要用的工具寫成函數直接放在Tools裡即可
- MCP Server
  - 建立在本機
    - 找到啟動程式位置以及專案路徑位置
  - 遠端
    - 給予連線網址與路徑以及相關參數
- LLM
  - Google Gemini
  - Azure OpenAI
  - Ollama / gpt-oss

