Day11 - 請支援 AI - Function Calling (上)

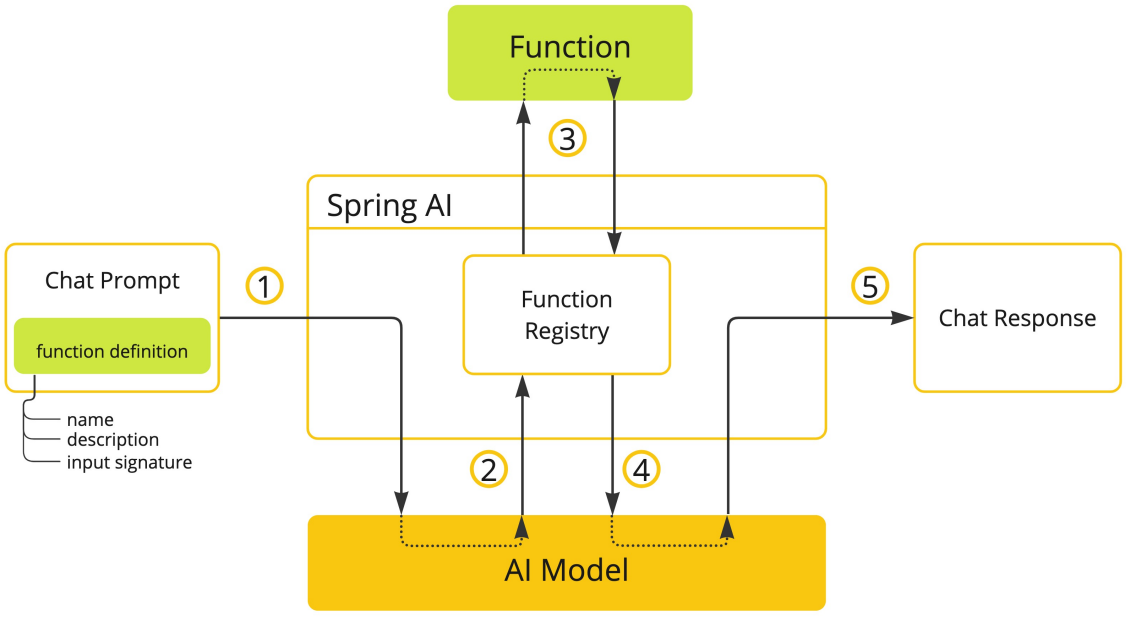
AI 的弱點

雖然 AI 很神奇，不過它不是萬能的，遇到以下幾種問題 AI 就沒轍

- **記憶**: LLM 是一種無狀態推論，雖然跟 ChatGPT 對話似乎能記住對話內容，不過實際上它是根據你送出的資料來推論結果
- **即時資料**: LLM 預訓練資料會在一個時間點後關閉，之後所發生的事情 AI 就無法回答你，例如問 AI 2024 巴黎奧運中華隊得了幾面金牌，AI 不是不知道就是隨便編一個答案給你，因為最近的 LLM 預訓練資料都大概都在去年底關閉
- **數學運算**: LLM 顧名思義就是一種語言模型，就跟文科生數理普遍不好一樣，有些簡單的加減法甚至會算錯，雖然最近的模型似乎有加強計算的能力，不過就連 ChatGPT 也不敢跟你保證算出的答案一定是對的
- **企業內部資料**: 很顯然預訓練只能取得公開得資料來訓練，企業不可能公開自己的內部資訊，常見的作法是透過 fine tuning 將內部資料融入模型內，不過耗時費力

第一點我們後續會有專門章節來說明，其他幾點則可靠 Function Call 來擴充 AI 的能力

簡單的說 Function Call 就像是 AI 的外掛，除了能透過 Function Call 取得即時資料外，一些複雜的運算或是需要分析的部分，也可以透過 Function Call 來呼叫外部函式取得結果，下圖就是 Function Call 的運作流程



每個流程動作如下

1. 發送提示詞時需要告訴 AI 有哪些 Function 可以調用，Spring AI 將這個部份包在 Options 中

```
return chatModel.call(
        new Prompt(
             prompt,
         OpenAiChatOptions.builder()
           .withFunction("CurrectDateTime")
           .build())
        ).getResult().getOutput().getContent();
```

2. AI 若查不到資料會查看是否有與提示詞相關的 Function 描述

```
@Bean
public FunctionCallback currectDateTime() {
    return FunctionCallbackWrapper.builder(new CurrectDateTim
            .withName("CurrectDateTime")
            .withDescription("Get the Date Time")
            .withResponseConverter((response) -> response.cur
            .build();
}
```

3. 找到匹配的 Function 後執行並調用其資訊

```
public Response apply(Request request) {
    return new Response(new Date());
}
```

4. 將 Function 回傳的資料以 FunctionMessage 的格式一起傳送給 AI
5. AI 根據提示詞以及 FunctionMessage 的綜合結果包成 Chat Response 回傳

1-3 項主要是由我們來撰寫，4 、5 則是 Spring AI 幫我們處理了，還記得 Day5 提到的 ToolResponseMessage 嗎？這就是 Function Calling 所回傳的解答

接下來就教大家從外部函式的撰寫到提示詞如何調用一步一步帶著大家做
我們先從簡單的開始

問 AI 目前幾點，程式會返回目前時間

1. 撰寫外掛程式: 實作 java.util.function

```
public class CurrectDateTimeFunction implements Function<Curr
    @Override
    public Response apply(Request request) { //這個函式就是 Fun
        return new Response(new Date());
    }
    public record Request(String State){ //其實可以不用請求參數，
    }
    public record Response(Date currDateTime) { //回傳的型態
    }
}
```

2. 註冊外掛程式: 寫一隻 Config 並使用 FunctionCallbackWrapper 將外掛程式包進 Builder 內，這裡須給予幾個設定
    a. Function Name
    b. Description
    c. Response回傳內容

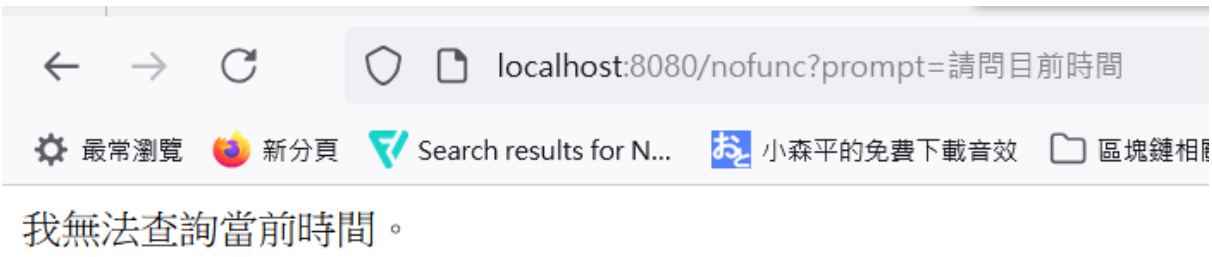
```
@Configuration
public class AiConfig {
    @Bean
  public FunctionCallback currectDateTime() {
      return FunctionCallbackWrapper.builder(new CurrectDateT
              .withName("CurrectDateTime") //Function Name
              .withDescription("Get the Date Time")  //Descri
              .withResponseConverter((response) -> response.c
              //回傳結果要如何轉換，一個參數通常都直接使用 .toString
              .build();
  }
}
```

3. 在 Option 中定義可被調用的 Function Name，這裡設定的名稱需與上一步一樣

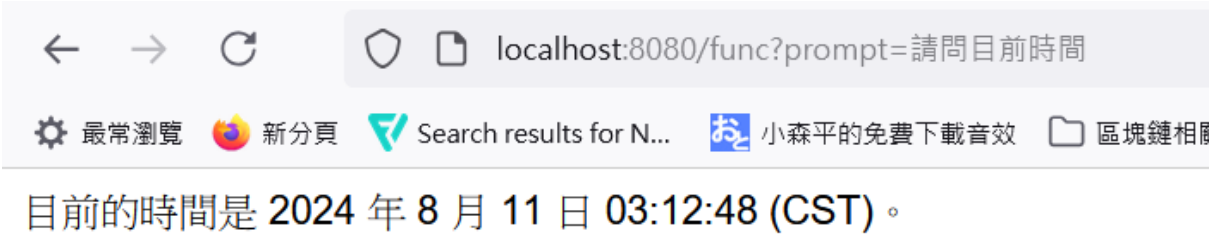
```
    @GetMapping("/func")
    public String func(String prompt) {
        return chatModel.call(
            new Prompt(prompt,
               OpenAiChatOptions.builder()
               .withFunction("CurrectDateTime")
               .build())
                ).getResult().getOutput().getContent();
    }
```

測試結果

未使用Function Call



使用Function Call



回顧今天學到的內容:

1. AI 的限制
2. Function Call 的調用流程
3. 程式撰寫
    a. 實作 java.util.function
    b. 使用 FunctionCallbackWrapper 註冊 Function，需提供名稱及描述
    c. 在 Options 中加入可調用的 Function