

控制敘述 - 重複

- for 重複敘述
- while 重複敘述
- do while 重複敘述
- break 分支敘述
- continue 分支敘述
- 認證實例演練

前一章介紹 if、if ... else、switch 三種選擇敘述，可以因應程式的需求而改變程式執行的流向。本章將介紹可以將程式區段重複執行多次的「重複結構」，共有 for、while、do ... while 三種重複敘述，以及 break 和 continue 分支敘述。

▶ 4.1 重複敘述 - for 迴圈

重複敘述是撰寫程式時常用的敘述之一，它的主要功用就是能重複執行某項工作，直到符合終止的條件才會停止，Java 提供的重複敘述有 for、while 和 do ... while 三種。

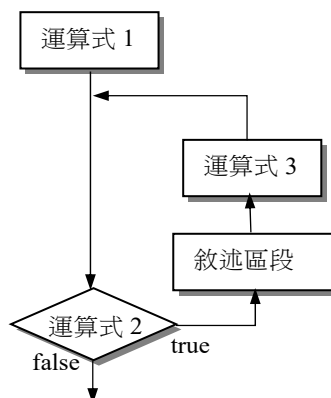
4.1.1 for 迴圈的使用

當迴圈的執行次數是可數時，for 迴圈是一個十分實用的重複執行敘述，其使用語法如下：

```
for(運算式 1; 運算式 2; 運算式 3) {
```

```
    敘述區段
```

```
}
```



說明

1. **運算式 1**：設定迴圈控制變數的資料型別 (可以使用 var 宣告) 和初值，若有多個變數時其資料型別必須相同。當 for 迴圈執行時會先執行 <運算式 1>，<運算式 1> 只會執行一次之後就不會再執行。
2. **運算式 2**：判斷是否要離開迴圈的條件。如 <運算式 2> 的結果為 true 時，就執行迴圈內的敘述區段一次，否則 (結果為 false) 就會跳離 for 迴圈。<運算式 2> 除第一次執行是在 <運算式 1> 之後執行外，其餘都是在執行完 <運算式 3> 後才會執行。
3. **運算式 3**：用來變更迴圈控制變數的值，可以控制迴圈執行的次數。<運算式 3> 是接在迴圈內的敘述區段後執行。
4. 如果迴圈內的敘述區段是多行時，前後一定要加大括號 { }；若只有一行敘述，則允許不使用左右大括號 { } 來框住。
5. 運算式 1~3 間使用分號「;」做分隔。<運算式 1> 和 <運算式 3> 的運算式內若含有多個條件時，中間使用逗號「,」隔開。
6. 如果省略運算式 1~3 表示為無窮迴圈，其中必須有跳離 for 迴圈的敘述。

```
for(;;) { ... }
```



簡例

1. 用 var 宣告 i 並設初值為 1，i 每次加 1，直到 i 值為 101 時離開迴圈：

```

      運算式 1    運算式 2    運算式 3
      ↓          ↓          ↓
for (var i = 1; i <= 100; i++) { ... }
      ↑          ↑
    分號分隔    分號分隔
  
```

2. j 為浮點數初值為 0，每次加 0.25，直到值為 10 時離開迴圈：

```
for(float j = 0; j < 10; j += 0.25) { ... }
```

3. 宣告 x 、 y 為整數並分別設初值為 1、0， x 每次加 2， y 每次加 1，直到 x 值為 11 時離開迴圈：

兩個條件運算式
兩個條件運算式

```
for(int x = 1, y = 0; x < 10; x += 2, y++) { ... }
```

逗號分隔
分號分隔
逗號分隔



簡例 餐廳給星評分程式



結果

請問你給幾顆星？5
***** 最後 i 值為: 6



程式碼

檔名：\ex04\src\ex04\For.java

```
01 package ex04;
02 import java.util.Scanner;
03 public class For {
04     public static void main(String[] args) {
05         Scanner scn = new Scanner(System.in);
06         System.out.print("請問你給幾顆星？");
07         int stars = scn.nextInt();           // 讀取星星數
08         int i;
09         for (i = 1; i <= stars; i++) {
10             System.out.printf("* ");
11         }
12         System.out.printf("最後 i 值為:%d", i);
13         scn.close();
14     }
15 }
```



說明

1. 第 7,8 行：宣告整數變數 `stars`，讀取使用者輸入的星星數。宣告整數變數 `i`，用來當作是迴圈控制變數。

2. 第 9~11 行：為 for 迴圈，先將 i 的初始值設為 1，再判斷條件式 $i \leq \text{stars}$ ，符合就執行第 10 行；不符合就離開迴圈執行第 12 行。
3. 第 10 行：印出一顆星星後，回到第 9 行，先執行 $i++$ 再檢查 $i \leq \text{stars}$ ？
 - ① 若為 true，執行第 10 行，再回到第 9 行，先執行 $i++$ 再檢查 $i \leq \text{stars}$ ？
 - ② 當 $i = \text{stars} + 1$ 時則 $i \leq \text{stars}$ 的結果為 false，就離開 for 迴圈然後執行第 12 行。
4. 第 12 行：印出 i 值來了解 for 迴圈的運作，結果為 $\text{stars} + 1$ 。若不宣告 i 為 main() 方法的區域變數(程式如下)，則 i 只是 for 迴圈的區塊變數，一離開迴圈就無效，所以第 12 行敘述會產生錯誤。

```

08    //int i;
09    for (int i = 1; i <= stars; i++) {
10        System.out.printf("* ");
11    }

```

4.1.2 巢狀 for 迴圈

if 有巢狀的 if 選擇敘述，for 迴圈也有巢狀結構。例如：顯示如各科成績表、年度銷售統計表、九九乘法表等表格，或是需要多行顯示資料時，都可以使用巢狀迴圈來達成。



簡例 使用巢狀 for 迴圈完成如下圖的九九乘法表輸出畫面：

```

2x1= 2 3x1= 3 4x1= 4 5x1= 5 6x1= 6 7x1= 7 8x1= 8 9x1= 9
2x2= 4 3x2= 6 4x2= 8 5x2=10 6x2=12 7x2=14 8x2=16 9x2=18
2x3= 6 3x3= 9 4x3=12 5x3=15 6x3=18 7x3=21 8x3=24 9x3=27
2x4= 8 3x4=12 4x4=16 5x4=20 6x4=24 7x4=28 8x4=32 9x4=36
2x5=10 3x5=15 4x5=20 5x5=25 6x5=30 7x5=35 8x5=40 9x5=45
2x6=12 3x6=18 4x6=24 5x6=30 6x6=36 7x6=42 8x6=48 9x6=54
2x7=14 3x7=21 4x7=28 5x7=35 6x7=42 7x7=49 8x7=56 9x7=63
2x8=16 3x8=24 4x8=32 5x8=40 6x8=48 7x8=56 8x8=64 9x8=72
2x9=18 3x9=27 4x9=36 5x9=45 6x9=54 7x9=63 8x9=72 9x9=81

```



程式碼

檔名：\ex04\src\ex04\NestFor.java

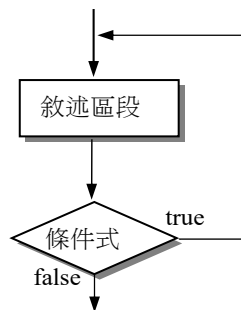
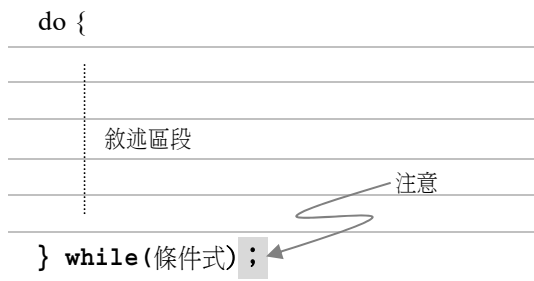
```

01 package ex04;
02 public class NestFor {
03     public static void main(String[] args) {

```

4.2.3 do...while 迴圈的使用

do...while 迴圈和 while 迴圈用法非常相似，差別在於 while 迴圈會先判斷條件式的結果才決定是否執行敘述區段，while 迴圈可稱為前測式迴圈；而 do...while 迴圈會先做完敘述區段才檢查條件式的結果，稱為後測式迴圈。所以，do...while 迴圈中的敘述區段至少會被執行一次；而 while 迴圈的敘述區段可能不被執行。do...while 的格式和 while 類似，只是多了一個 do 而且把 while(條件式); 放到右大括號的後面，要注意的是一定要加上分號！



實作

設計只能輸入 1~15 的正整數，超出範圍就重新輸入直到正確為止。數值正確後，就顯示該數的階乘值。例如 5 階乘值(5!)等於 $5*4*3*2*1$ 值為 120。



結果

請輸入 1~15 正整數來計算階乘值: 7

7! = 5040



程式碼

檔名: \ex04\src\ex04\DoWhile.java

```

01 package ex04;
02 import java.util.Scanner;
03 public class DoWhile {
04     public static void main(String[] args) {
05         Scanner scn = new Scanner(System.in);
06         int num, n, sum;
07         do {
08             System.out.print("請輸入 1~15 正整數來計算階乘值: ");
  
```

使用 OpenAI API 開發聊天機器人

- OpenAI API 服務申請
- 製作主控台應用程式聊天機器人
- Java 呼叫 OpenAI API 開發聊天機器人
- 製件視窗應用程式聊天機器人

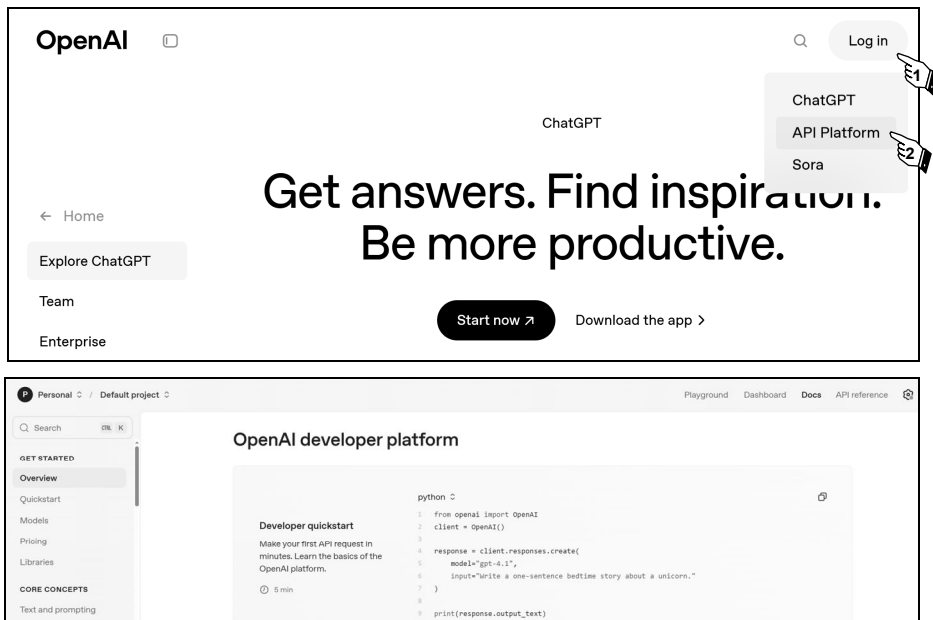
17.1 OpenAI API 服務申請

由上一章體驗可知 ChatGPT 可使用人類自然的對話方式來進行互動，可以完成複雜的工作，例如：文本生成新聞稿與產品描述說明、文本修訂與生成摘要、信件回覆、履歷撰寫、創意企畫發想、語言翻譯與程式開發等應用。

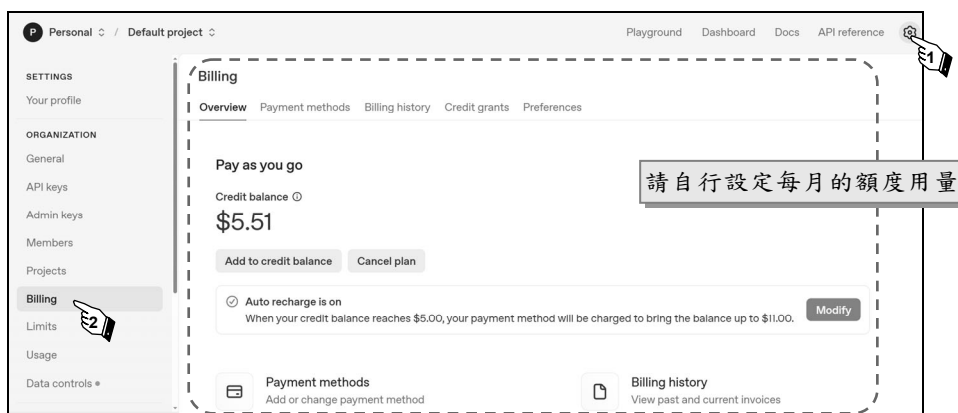
為了讓開發人員與企業有更多應用範圍，OpenAI 釋出了 OpenAI API，讓開發人員與企業可將大型語言模型 (LLM) 技術導入專案、網站、App 或產品中。OpenAI API 讓開發人員可以使用大型語言模型依提示產生文本，就好像使用 ChatGPT 一樣，模型可以產生任何類型的文字回應。

若要申請使用 OpenAI API，可以按照以下步驟進行：

- Step 1 註冊 OpenAI 帳戶，此步驟可參考 16.1.3 節步驟。
- Step 2 連接到「<https://openai.com/chatgpt/overview/>」網頁，並依下圖操作進入 OpenAI API 網站平台。



Step 3 如下圖操作，先點選「」設定鈕，點選「Billing」進入 Billing 網頁 (可能會更換網址)，請自行設定 OpenAI API 可使用額度，此步驟需要使用信用卡設定付費額度。



Step 4 如下圖操作，先點選「Dashboard」，再點選「Usage」進入 Usage 網頁 (可能會更換網址)，查詢目前 OpenAI API 可使用額度。其費用計算方式每 1000 token 收費 0.002 美元，1 個 token 約為 4 個 characters (之後付費額度方案可能會修正)。如下圖 Usage 網頁會呈現指定月份每天的用量，以及使用期限。

完成 API 金鑰的申請，此時開發人員就可以使用 Java、C# 或 Python 語言，透過申請的 API 金鑰來呼叫 OpenAI API。

17.2 Java 呼叫 OpenAI API 開發聊天機器人

在 Java 程式中，可以透過 HTTP 或 HTTPS 向網路服務 (Web API) 發送請求 (Request)，並接收回應 (Response)。目前多數網路 API 所傳遞的資料，普遍採用 JSON (JavaScript Object Notation) 格式。

為了方便處理這些 JSON 資料，Java 可使用 org.json 套件 (又稱 JSON-Java 套件) 進行處理。這個套件提供以下主要功能：

- 將 JSON 字串轉換成 Java 物件
- 將 Java 物件轉換成 JSON 格式
- 在 JSON 與其他格式 (如 XML、HTTP Header、Cookie 等) 之間轉換

本節將說明如何將 org.json 套件加入 Java 專案中，並示範如何使用該套件處理下面事項：

1. 傳送請求至 OpenAI API (需事先擁有 API 金鑰)
2. 接收並取得 JSON 格式的回應結果
3. 使用 org.json 套件解析回傳的 JSON 資料，並擷取其中的 API 回應內容



實作 建立主控台應用程式的 ChatGPT 聊天機器人。



結果

請輸入提示 (Prompt)：請說明 Java 的 System.IO 的功能 **Enter**
GPT-4o 回應：
System.IO 是 Java 中提供的一個 I/O (輸入/輸出) 類別庫，提供了許多方法和類別來執行各種 I/O 操作。它提供了一個抽象化的檔案系統，可以讀取和寫入檔案，並且提供了許多其他的 I/O 操作，如管道、輸入/輸出流、檔案複製、檔案移動、檔案刪除等。System.IO 還提供了一個簡單的 API 來操作序列化的物件，以及一個基於文字的文件格式，例如 XML 和 JSON。

輸入問題後
按 **Enter** 鍵

機器人回答內容
每次可能不同

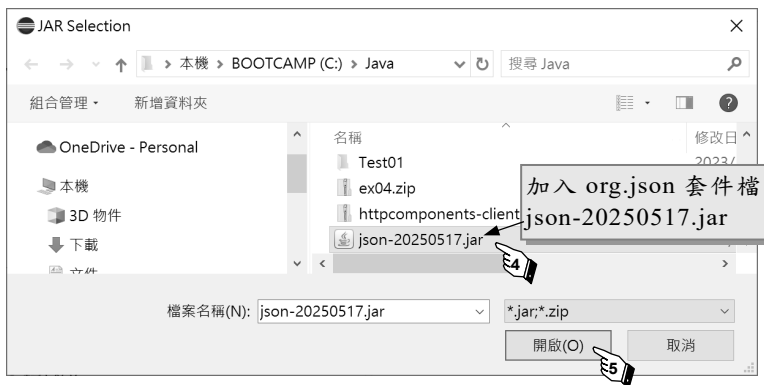
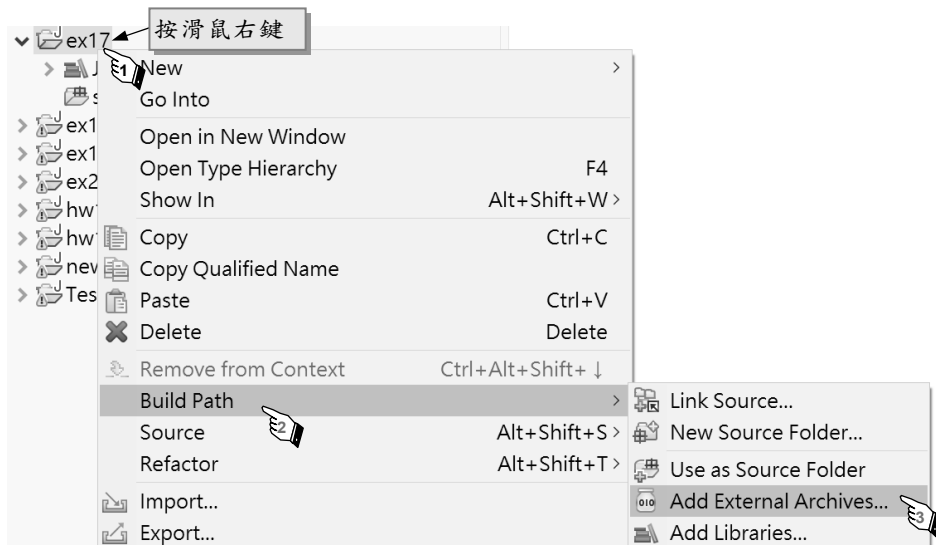
Step 1 下載 org.json 套件

連結「<https://github.com/stleary/JSON-java>」前往下圖的 github 網頁，再點選下圖連結「Click here if you just want the latest release jar file.」，下載 org.json 套件的「 json-20250517.jar」檔案。



Step 2 專案加入 org.json 套件的「 json-20250517.jar」檔案

建立名為 ex17 的 Java 專案，依下圖操作在專案中加入 org.json 套件的「 json-20250517.jar」檔案。



Step 3 撰寫程式碼：

在 ex17 專案的 src 資料夾下新增 sample01 套件資料夾，接著在「src/sample01」內新增 OpenAIGpt4o01.java。本例會使用 OpenAI API 金鑰，申請方式請參閱 17.1 節。完整程式碼如下：



程式碼

檔名：\ex17\src\sample01\OpenAIGpt4o01.java

```
01 package sample01;
02
03 import java.io.*;
04 import java.util.Scanner;
05 import java.net.HttpURLConnection;
06 import java.net.URL;
07 import org.json.JSONObject;
08 import org.json.JSONArray;
09 import java.nio.charset.StandardCharsets;
10
11 public class OpenAIGpt4o01 {
12
13     // 請替換成你的 OpenAI API 金鑰
14     private static final String API_KEY = "你的 OpenAI API 金鑰";
15     private static final String ENDPOINT =
16         "https://api.openai.com/v1/responses";
17     public static void main(String[] args) throws IOException {
18         Scanner scanner = new Scanner(System.in);
19         System.out.print("請輸入提示 (Prompt) : ");
20         String prompt = scanner.nextLine();
21
22         String reply = callOpenAIGpt4o(prompt);
23         System.out.println("GPT-4o 回應：\n" + reply);
24     }
25
26     public static String callOpenAIGpt4o(String prompt)
27         throws IOException {
28         URL url = new URL(ENDPOINT);
29         HttpURLConnection conn=(HttpURLConnection) url.openConnection();
```

API 金鑰申請請參閱 17.1 節

呼叫 OpenAI API 的 Responses
API(回應 API)服務