Microsoft Azure Functions

维基百科，自由的百科全书

Microsoft Azure Functions 是 Microsoft Azure 平台上，提供無伺服器運算架構 (Serverless Computing) 的服務，允許開發人員在不用接觸與管理伺服器的情況下，編寫小型的處理程式以處理雲端上的訊息或事件。

Azure Functions 於微軟 Build 2016 大會上宣布，於同年11月15日正式 GA (General Availability)。

Azure Functions 已廣泛於用如 Azure Bot Services、Azure IoT Suite、Azure Logic App 等服務串聯。

目录

簡介

Azure Functions 是以 Azure Web App 開發時所建立的 Microsoft Azure Azure Web Jobs 為基礎所開發的一個服務，Web Jobs 在開發時就已經支援多語言 (包含 Bash, 批次檔, C#, node.js, PowerShell, F#, Python 與 PHP 等) 的執行環境，因此在發展 Azure Functions 時，也將 Web Jobs 的多語言能力移植到 Azure Functions，同時基於無伺服器的架構，Azure Functions 只要求開發人員在 Azure 的管理介面上撰寫程式碼，即可立即測試與執行，不必擔心背後的資源分配與伺服器管理的問題。

Azure Functions 在設計上以函數 (function) 會有的特性來規劃，一個函數本身會有輸入 (input)、處理 (process) 與輸出 (output) 三個部份，微軟將輸入和輸出進行抽象化的處理，以繫結 (binding) 來替代[1]。

- 輸入繫結 (input binding) 表示函數會於繫結指定的條件發生時觸發，並將條件所需的參數傳到函數作為輸入的參數 (argument)。
- 輸出繫結 (output binding) 表示函數會輸出成指定的格式，通常作為回傳值 (return value)。

Azure Functions 如同其他 Azure應用服務 一般，執行於 App Service Plan (應用服務計畫)[2] 之上，不過為了要達到無伺服器架構的目標，Azure Functions 還多了一個使用模式，稱為消費計畫 (Consumption Plan)，以執行次數 (Executions) 和執行時的資源耗用量 (Resource Consumption) 為計費基礎 [3]，但使用者可選擇要用原始的 App Service Plan 還是要使用計量級的 Consumption Plan 作為計費單位。

功能

Azure Functions 基於無伺服器管理架構，因此其功能大多數都落在程式碼的開發、繫結的設定與安全金鑰的管理，當然也可以依需求進一步的設定其所處的 App Service Plan (即使是 Consumption Plan 也有) 的進階設定，例如加上 SSL 或增修應用程式組態檔的設定等等。

程式碼管理

Azure Functions 允許開發人員直接編寫函數的程式碼，也就是處理 (Process) 這個部份，基於與函數的繫結的整合，微軟提供了數個樣板 (template) 給開發人員選用，每個支援的繫結都會有一個樣板，開發人員不需費心於思考如何在函數中使用繫結，只要在函數的本體中編寫處理的程式碼即可。

例如使用 HTTP 觸發 (即送 HTTP 要求給該函數) 的 C# 程式碼如下：

```
public static Task<HttpResponseMessage> Run(HttpRequestMessage request, string category, int? id,
                                                    TraceWriter log)
    {
        if (id == null)
           return  req.CreateResponse(HttpStatusCode.OK, $"All {category} items were requested.");
        else
           return  req.CreateResponse(HttpStatusCode.OK, $"{category} item with id = {id} has been
requested.");
    }
```

而同樣功能的 node.js 程式碼如下：

```
module.exports = function (context, req) {

        var category = context.bindingData.category;
        var id = context.bindingData.id;

        if (!id) {
            context.res = {
                // status: 200, /* Defaults to 200 */
                body: "All " + category + " items were requested."
            };
        }
        else {
            context.res = {
                // status: 200, /* Defaults to 200 */
                body: category + " item with id = " + id + " was requested."
            };
        }

        context.done();
    }
```

範本會先為開發人員設定好參數，開發人員只要填入處理的程式即可。

另外，Azure Functions 也支援由 git 部署[4]，然而若使用 git 部署時，Azure Portal 上的程式碼編輯器會強制停用。

繫結

繫結決定了函數的輸入、觸發時間與輸出，Azure Functions 支援了下列的觸發程序、輸入與輸出。由於觸發程序和輸入參數可以不同，因此下表中無輸入但有觸發支援時，觸發的參數會作為輸入的參數。

類型	觸發	輸入	輸出
排程	支援 (設定的時間或週期到時)		
HTTP	支援 (接到由 REST 或 WebHook 發送的 HTTP 要求時)		支援
Blob	支援 (Azure Storage Blob 監控的容器有資料新增或更新時)	支援	支援
事件	支援 (EventHub 有事件進入時)		支援
佇列	支援 (Azure Storage Queue 佇列內有訊息時)		支援
佇列和主題	支援 (Azure Service Bus Queue 佇列或 Topic 主題內有訊息時)		支援
儲存體資料表		支援	支援
SQL 資料表		支援	支援
NoSQL 資料庫		支援	支援
通知訊息			支援
Twilio SMS 服務			支援
SendGrid			支援

每個輸入、輸出繫結與觸發程序也會有必要的參數設定 (例如排程需要設定執行週期、Blob 需要設定監控的儲存體帳戶與容器等等)。

繫結的設定除了可在使用者介面上設定外，也可使用每個函數都有的 function.json 檔案設定，例如下列設定表示了監控 blob 的輸入，如果有輸入就觸發函數，並且將函數的輸出指定為另一個 blob 的繫結。

```
{
  "bindings": [
    {
      "name": "image",
      "type": "blobTrigger",
      "path": "sample-images/{filename}",
      "direction": "in",
      "connection": "MyStorageConnection"
    },
    {
      "name": "imageSmall",
      "type": "blob",
      "path": "sample-images-sm/{filename}",
      "direction": "out",
      "connection": "MyStorageConnection"
    }
  ],
}
```

存取控制

使用 HTTP 繫結作為輸入的函數 (通常用於 REST 與 WebHook)，Azure Functions 會另外提供金鑰 (Key) 給開發人員使用，以作為存取控制的方法[5]。

金鑰預設有兩種：

- 主機金鑰 (master key)：所有建於此應用程式內的所有函數都可使用。
- 函數金鑰 (function key): 只有金鑰所屬的函數可使用。

除了上面兩個預設的金鑰外，開發人員也可建立自己的金鑰，並決定存取權限是限制於函數，還是要支援所有的函數。

開發人員支援

Azure Functions 除了提供 Azure Portal 上的程式碼編修，以及採用 git 的持續整合的程式碼管理方式外，也為開發人員在地端的作業提供了 Azure Function Runtime，以加速開發人員在地端的開發與測試。

Visual Studio Code 也有套件支援 Azure Functions 的開發。

參考

1. Azure Functions 觸發程序和繫結概念 (https://docs.microsoft.com/zh-tw/azure/azure-functions/functions-triggers-bindings)
2. Azure Functions 取用和 App Service 方案 (https://docs.microsoft.com/zh-tw/azure/azure-functions/functions-scale)
3. Azure Functions Pricing (https://azure.microsoft.com/en-us/pricing/details/functions/)
4. Deploying Azure Functions Automatically. [2017-05-28]. （原始内容存档于2017-09-10）.
5. Azure Functions HTTP 和 Webhook 繫結 (https://docs.microsoft.com/zh-tw/azure/azure-functions/functions-bindings-http-webhook#a-nameworking-with-keysa使用金鑰)

外部連結

- Introducing Azure Functions (https://docs.microsoft.com/zh-tw/azure/azure-functions/functions-overview)
- Azure Functions Overview (https://azure.microsoft.com/zh-tw/services/functions/)