

Reactor

Python Web 開發



Map



我係



Kinfey Lo – (盧建暉)

Microsoft Cloud Advocate

前微軟MVP、Xamarin MVP和微軟RD，擁有超過10年的雲原生、人工智能和流動應用程式開發經驗，為教育、金融和醫療提供應用。在微軟，為技術人員和不同行業宣講技術和相關應用場景。



Love Coding(Python , C# , TypeScript , Swift , Rust , Go)

專注於人工智能，雲原生，流動平台移動開發

Github : <https://github.com/kinfey>

Email : kinfeylo@microsoft.com **Blog :** <https://dev.to/kinfey>

Twitter : @Ljh8304

Flask 簡介



Flask 係？

<https://github.com/pallets/flask>

Flask 是一個輕量級web 框架，提供工具，庫和技術來構建web 應用。

Flask 屬於micro-framework， micro-framework通常是很小的不依賴於外部庫的框架。優點是框架很輕量，更新時依賴少，並且專注安全方面的 bug，缺點是，不得不做更多的工作，或通過添加插件增加自己的依賴列表。

pip3 install flask

使用flask的企業



Flask
web development,
one drop at a time

Flask 優缺點

優點

Flask 可以完全控制 Web 開發，從而完全控制應用程序和 Web 開發。

Flask 有完整的單元測試功能和集成開發服務器，可以通過對擴展進行調整來過渡到 Web 框架。
簡單易用

缺點

Flask 的模塊由第三方開發，容易產生安全漏洞。

Flask 具有一個單一的來源，表示它將依次處理每個請求，因此，無論有多少個請求，它仍然會輪流處理它們，這會耗費更多時間。

開發人員水平不高，更容易使用低質量的代碼創建一個不良的 Web 應用程序。

Python Web Framework 的其他選擇



Django 是一個 Python Web 框架，適合具有開發時間限制要求的完美主義者。Django 提供功能齊全的MVC框架。

⚡ FastAPI

用於構建 Web API 的現代、開源、快速、高性能的 Web 框架，它基於Python 3.6+ 標準類型提示，支持異步，FastAPI 就是為構建快速的 API 而生。

Flask vs Django vs Fast API

包豐富程度：

Django 具有使代碼可重用的大多數軟件包，是一個完整的 Web 開發框架，而 Flask 和 FastAPI 是用於構建網站的簡約框架，很多功能比如用戶系統，後台管理要自己實現。

社區活躍程度：

Django 社區是最活躍的社區，這是因為它使用廣泛，很多大廠使用，另一方面，Flask 的社區也很好，僅次於 Django。FastAPI 的社區目前還比較小，因為它相對較新。

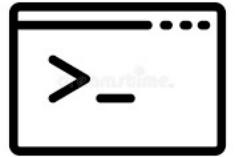
性能：

在性能方面：FastAPI 是領跑者，因為它是面向速度的，其次是 Flask，最後是 Django。

靈活性：

靈活性是開發人員非常重視的東西，並且 Flask 比 Django 更靈活。另一方面，FastAPI 在代碼方面是靈活的，並且不限制代碼佈局。因此，我們可以說 Flask 在這三者中是最靈活的。

第一個Flask程式



flask run



<http://127.0.0.1:5000/>

The screenshot shows a Microsoft Visual Studio Code (VS Code) window with the following details:

- Explorer View:** Shows a folder named "FLASKDEMOAPP" containing "__pycache__" and "app.py".
- Code Editor:** Displays the "app.py" file content:

```
1 # save this as app.py
2 from flask import Flask
3
4 app = Flask(__name__)
5
6 @app.route("/")
7 def hello():
8     return "Hello, World!"
```
- Terminal:** Shows the command line output for running the Flask application:

```
(base) lokinfey@lokinfoeys-MacBook-Pro flaskdemoapp % conda activate tf24
(tf24) lokinfey@lokinfoeys-MacBook-Pro flaskdemoapp % flask run
* Environment: production
  WARNING: This is a development server. Do not use it in a produ
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```
- Status Bar:** Shows the URL "127.0.0.1:5000" and the text "Hello, World!".

Flask 基礎知識



Flask 基礎知識 – Route

現代 web 應用都使用有意義的 URL，這樣有助於用戶記憶，網頁會得到用戶的青睞，提高回頭率

如

主頁：<https://adventure-works.com/>

小組件詳細信息：<https://adventure-works.com/products/widget>

完成購買：<https://adventure-works.com/cart/buy>

Flask - Route

```
@app.route()
```

```
@app.route('/info/<name>')
def get_name(name):
    return "你好 " + str(name)
```

```
# save this as app.py
from flask import Flask
```

```
app = Flask(__name__)
```

```
@app.route("/")
def hello():
    return "Hello, World!"
```

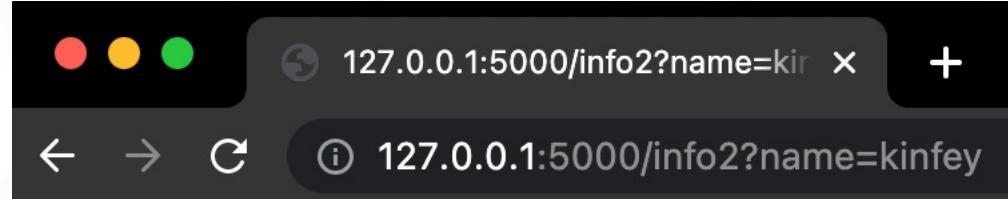
```
@app.route('/about')
def about():
    return '关于Reactor'
```

Flask - Route

<http://127.0.0.1:5000/info2?name=kinfey>

```
from flask import Flask, request
```

```
@app.route('/info2')
def get_name2():
    name = request.args.get('name')
    return "你好 " + str(name)
```



你好 kinfey

Flask - Route

支援多Route

```
@app.route('/blog/posts')
@app.route('/blog/posts/<post_id>') def
get_blog_post(post_id=None):
    # get the post or list of posts
```

Resource參數自定義

```
@app.route('/blog/post/<int:post_id>')
# 支持 string,int ,float , path , any , uuid
```

Flask – HTTP Method

路由也可以配置為接受HTTP方法。路由裝飾器接受methods關鍵字參數，該參數是表示此路由可接受的HTTP方法的字符串列表。正如您可能假設的那樣，默認值是GET only。

HTTP GET

使用GET請求僅檢索資源表示/信息 - 而不是以任何方式修改它。由於GET請求不會更改資源的狀態，因此這些是安全的方法。此外，GET API 應該是幂等的，這意味著每次生成多個相同的請求必須產生相同的結果，直到另一個API（POST或PUT）更改了服務器上的資源狀態。

HTTP POST

使用POST API 創建新的下級資源，例如，文件從屬於包含它的目錄，或者行從屬於數據庫表。嚴格按照REST進行討論，POST方法用於在資源集合中創建新資源。

HTTP PUT

主要使用PUT API 來更新現有資源（如果資源不存在，則API可能決定是否創建新資源）。

HTTP DELETE

正如名稱一樣，DELETE API用於刪除資源（由Request-URI標識）。

HTTP PATCH

HTTP PATCH請求將對資源進行部分更新。如果您看到PUT請求也修改了資源實體以便更清楚 - PATCH方法是部分更新現有資源的正確選擇，只有在您完全替換資源時才應使用PUT

Flask RESTful API HTTP狀態

HTTP方法	CRUD	整个集合（例如/用户）	特定项目（例如/ users / 123）
POST	创建	201（已创建），'位置'标题，其中包含指向/users / {id}的链接，其中包含新ID。	避免在单一资源上使用POST
GET	读	200（OK），用户列表。使用分页，排序和过滤来导航大列表。	200（OK），单用户。404（未找到），如果找不到ID或无效。
PUT	更新/替换	404（未找到），除非您要更新整个资源集合中的每个资源。	200（OK）或204（无内容）。如果未找到ID或无效，请使用404（未找到）。
PATCH	部分更新/修改	404（未找到），除非您想要修改集合本身。	200（OK）或204（无内容）。如果未找到ID或无效，请使用404（未找到）。
DELETE	删除	404（未找到），除非您要删除整个集合 - 请谨慎使用。	200（好的）。404（未找到），如果找不到ID或无效。

<http://restful.p2hp.com/resources/http-methods>

Flask - Route Method

```
@app.route('/blog/new', methods=['GET', 'POST'])
def new_post():
    if request.method == 'GET':
        # return the form
    elif request.method == 'POST':
        # get the data from the form values
```

```
@app.route('/info3', methods=['POST'])
def get_name3():
    name = request.form.get('name')
    return "你好 " + str(name)
```

Flask – Route methods

The screenshot shows the Thunder Client extension interface in Visual Studio Code. On the left, the code editor shows a file named `app.py`. In the center, a request is being made to `http://127.0.0.1:5000/info3` via a POST method. The request body contains a single field named `name` with the value `lo`. The response on the right is a `200 OK` status with a size of 9 bytes and a duration of 6.73 seconds. The response body displays the text `你好 lo`.

app.py

tc New Request X Extension: Thunder Client

POST http://127.0.0.1:5000/info3 Send

Query Auth Headers 2 Body 1 Tests

Json Xml Text Form Form-encoded Graphql

Form Fields Files

name lo

field name value

Status: 200 OK Size: 9 Bytes Time: 6.73 s

Response Headers 4 Cookies Test Results { }

1 你好 lo

Preview ↴ ↵

Flask – Web Template

超文本標記語言 (HTML) 是用於構造瀏覽器上顯示的信息的語言，而級聯樣式表 (CSS) 則用於管理樣式和佈局。在創建應用程序時，大多數 HTML 都是靜態的，這意味著該語言不會改變。然而，為使頁面具有動態性，我們需要能夠以編程方式將信息放入 HTML 頁面。

網頁模板需要導入render_template

from flask import Flask, render_template

```
@app.route('/index')
def index():
    return render_template('index.html')
```

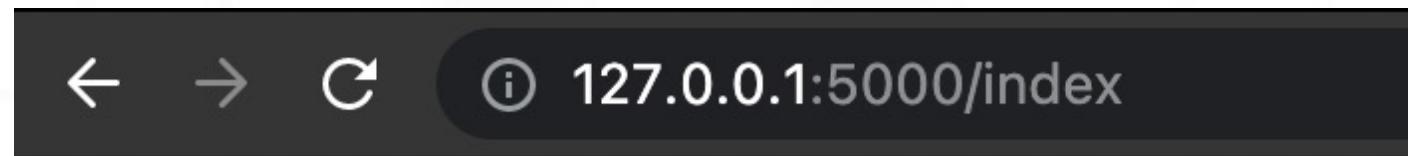
Flask – Web Template

超文本標記語言 (HTML) 是用於構造瀏覽器上顯示的信息的語言，而級聯樣式表 (CSS) 則用於管理樣式和佈局。在創建應用程序時，大多數 HTML 都是靜態的，這意味著該語言不會改變。然而，為使頁面具有動態性，我們需要能夠以編程方式將信息放入 HTML 頁面。

Web Page import render_template **from flask import Flask, render_template**

```
@app.route('/index')
def index():
    return render_template('index.html')
```

/templates , 添加index.html



123

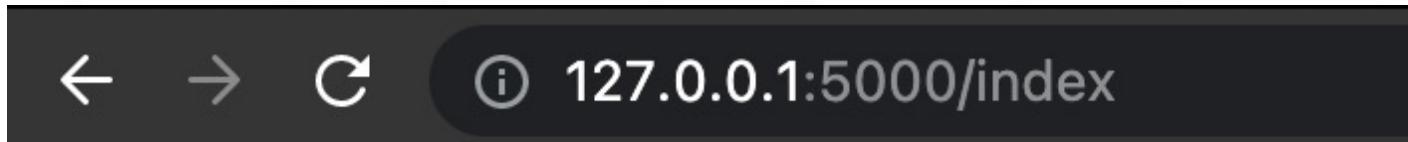
Flask - Web Template

网页模板需要导入render_template **from flask import Flask, render_template**

```
@app.route('/index')
def index():
    return render_template('index.html')
```

创建templates 子文件夹，添加index.html

```
templates > <> index.html > ⚙ b
1   <b>123</b>
```



Flask – Web Template

```
@app.route('/index2')
def index2():
    message = {
        'enName': 'kinfey',
        'chName': '建晖'
    }
    return render_template('index2.html', info = message)
```

```
templates > ◊ index2.html > □ div
1   <div>
2       我中文名字叫: {{ info.chName }}<br/>
3       我英文名字叫: {{ info.enName }}<br/>
4   </div>
```



我中文名字叫: 建晖
我英文名字叫: kinfey

Flask 進階學習



Flask – Blueprint

Flask 開發的Web程式可以通過Blueprint進行模塊化的處理

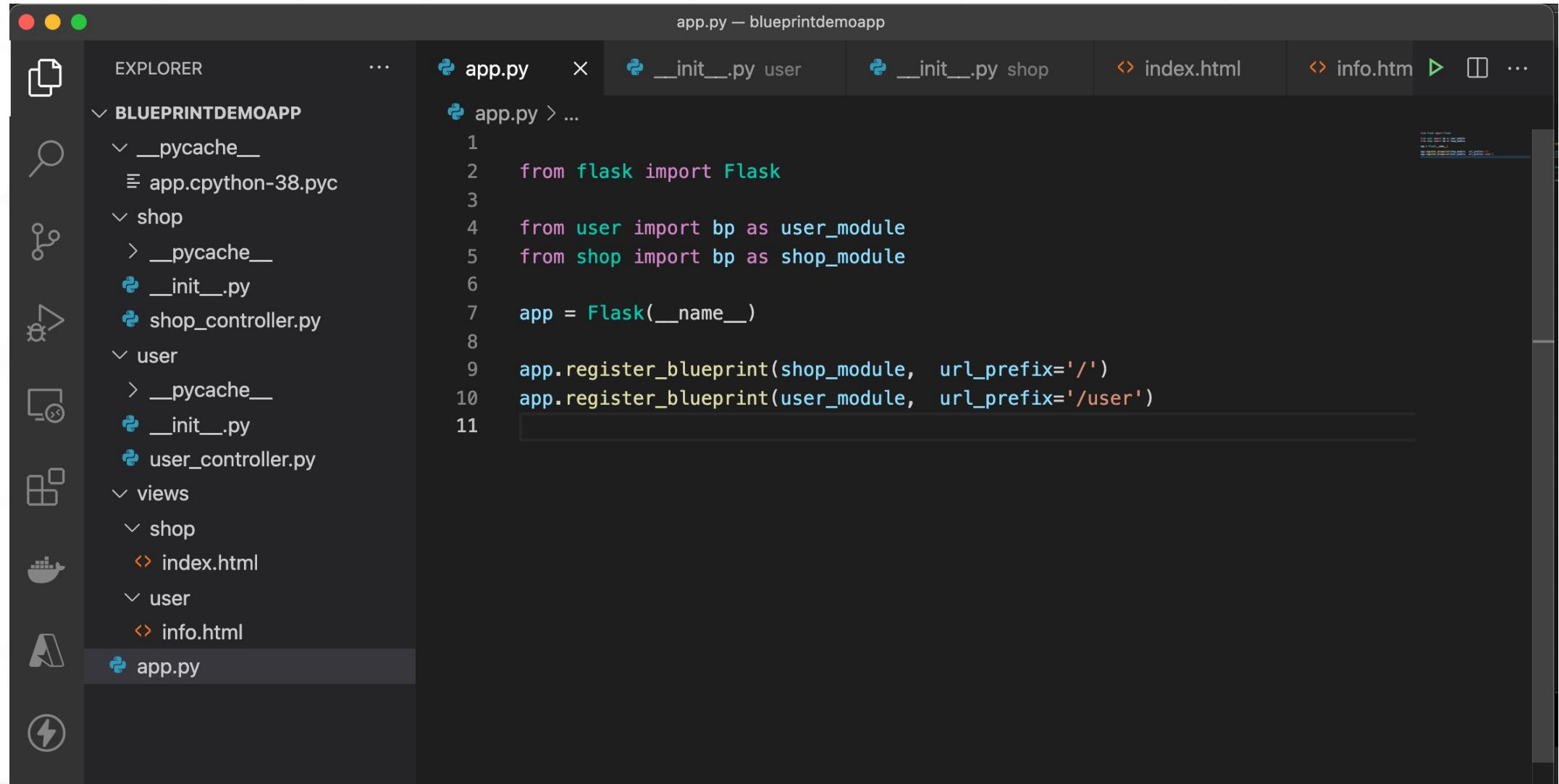
Flask 可以通過Blueprint來組織URL以及處理請求。

一個項目可以具有多個Blueprint

可以將一個Blueprint註冊到任何一個未使用的URL下, 比如 “/”、“/sample”或者子域名

在一個應用中, 一個模塊可以註冊多次 Blueprint可以單獨具有自己的模板、靜態文件或者其它的通用操作方法, 它並不是必須要實現應用的視圖和函數的 在一個應用初始化時, 就應該要註冊需要使用的Blueprint

Flask – Blueprint



The screenshot shows a dark-themed code editor interface, likely Visual Studio Code, displaying a Flask application structure and its configuration.

File Explorer: Shows the project structure under "BLUEPRINTDEMOAPP".

- __pycache__
- shop
 - __pycache__
 - __init__.py
 - shop_controller.py
- user
 - __pycache__
 - __init__.py
 - user_controller.py
- views
 - shop
 - index.html
 - user
 - info.html
- app.py

Code Editor: The active file is `app.py`.

```
1  from flask import Flask
2
3
4  from user import bp as user_module
5  from shop import bp as shop_module
6
7  app = Flask(__name__)
8
9  app.register_blueprint(shop_module, url_prefix='/')
10 app.register_blueprint(user_module, url_prefix='/user')
```

Flask Flask-SQLAlchemy

flask中一般使用flask-sqlalchemy來操作數據庫，使用起來比較簡單，易於操作。

<http://www.pythondoc.com/flask-sqlalchemy/api.html#flask.ext.sqlalchemy.SQLAlchemy>

pip3 install flask-sqlalchemy

pip3 install mysql-connector-python

```
HOST = '127.0.0.1'  
PORT = '3306'  
DATABASE = 'flask1'  
USERNAME = 'root'  
PASSWORD = '123456'  
  
DB_URI = "mysql+pymysql://{username}:{password}@{host}:{port}/{database}"  
  
SQLALCHEMY_DATABASE_URI = DB_URI  
SQLALCHEMY_TRACK_MODIFICATIONS = False  
SQLALCHEMY_ECHO = True
```

Flask - MySQL Connect

```
from flask import Flask
from flask_sqlalchemy import SQLAlchemy

app = Flask(__name__)

app.config['SECRET_KEY'] = "xxxxxxxx"

db = SQLAlchemy(app)

app.config["SQLALCHEMY_DATABASE_URI"] =
    "mysql://{}:{}@{}/testdb".format(username, password, server)
```

Flask 知識補充



Flask 的一些細節

端口切換 flask run -p 8000

env FLASK_APP=index.py flask run -p 5001

VSCode Debug 是很好的

Docker / Apache / IIS

```
1 // Use IntelliSense to learn about possible values
2 // Hover to view descriptions of existing attributes
3 // For more information, visit: https://go.microsoft.com/fwlink/?linkid=830302
4
5 "version": "0.2.0",
6 "configurations": [
7   {
8     "name": "Python: Flask",
9     "type": "python",
10    "request": "launch",
11    "module": "flask",
12    "env": {
13      "FLASK_APP": "app.py",
14      "FLASK_ENV": "development"
15    },
16    "args": [
17      "run",
18      // "-p 8000",
19      "--no-debugger"
20    ],
21    "jinja": true
22  }
23 ]
24 }
```

Azure AI 研發能力

提供一些預定義的模型
能降低開發門檻



Vision



Speech



Language



Search

兼容不同的開發工具
快速完成模型開發，簡化開發流程



PyCharm



Jupyter



Visual Studio Code



Command line

對人工智能開發框架的支持
根據你的需要創建深度學習的解決方案



Pytorch



TensorFlow



Scikit-Learn



Onnx

提供生產服務
為開發團隊提供數據，和訓練環境



Azure Databricks



Azure Machine Learning



Machine Learning VMs

強大的硬件架構支持
加速深度學習環境



CPU



GPU

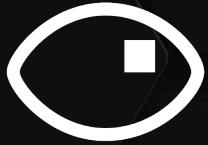


FPGA

實現



Azure Cognitive Service



視像

識別和確定你的圖片、視頻和數字墨跡內容，為它們添加描述文字和編制索引，並審查這些內容。



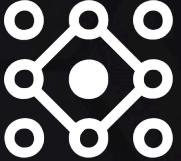
語音

將語音音頻轉換為文本、使用語音進行驗證，或在應用中添加說話人辨識功能。



語言

允許應用使用預建腳本處理自然語言、評估情緒及學習如何識別用戶想要的內容。



決策

構建應用，用於呈現有助於做出明智和高效決策的建議。



搜尋

將必應搜索 API 添加到應用中，並利用使用單一 API 調用梳理數十億網頁、圖像、視頻和新聞的能力。

選擇 Azure Cognitive Service



易用



多平臺



經驗



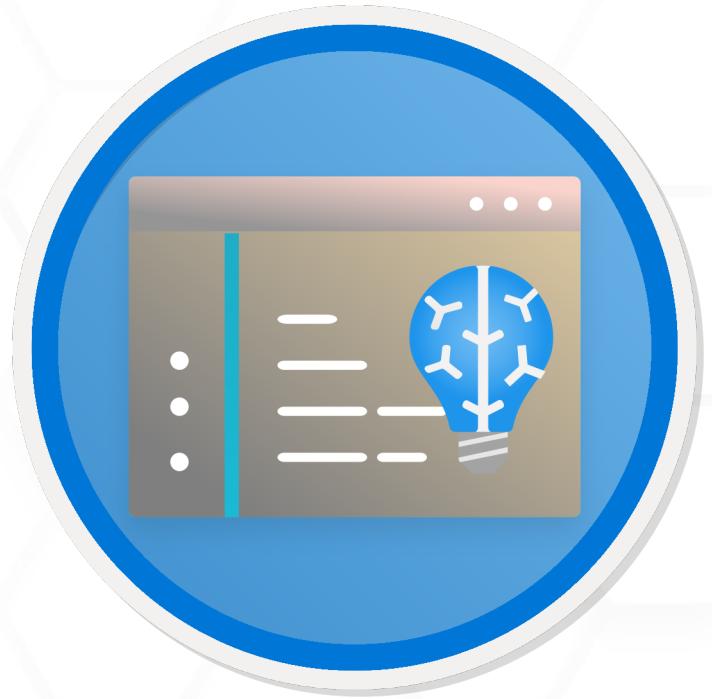
DEMO

翻譯頁面



Q&A

MS Learn學習推薦



<https://aka.ms/HKPythonLearn002>



Reactor

Thank You!

Q&A