News

Python e il web con Flask: un micro-framework open source flessibile e leggero

Rizzoli EDUCATION

News

Il linguaggio di rifermento della didattica del coding nelle scuole secondarie di secondo grado è ormai Python, mi sono chiesta dunque se potesse essere possibile integrare Python anche nella didattica dei linguaggi per il Web e mi sono imbattuta in Flask.

Flask è un micro-framework, cioè un insieme di moduli pre-costruiti che aiuta a creare applicazioni web in modo più rapido ed efficiente anche con il linguaggio Python, micro perché ha un nucleo semplice ma è possibile aggiungere le funzionalità di cui non dispone nativamente.

Utilizzando un framework si risparmia tempo perché non si deve scrivere da zero codice per le funzionalità più utilizzate, il programma risulta più leggibile e manutenibile inoltre le funzioni del framework sono testate da più programmatori e dunque il codice risulta più affidabile.

In genere gli studenti imparano a programmare con Python almeno fino al paradigma ad oggetti e poi passano ai linguaggi per il Web, il mio obiettivo è dunque cercare di sostituire javaScript e PHP, che dovrei cominciare a insegnare da zero, con Python, in modo da sfruttare le conoscenze già acquisite precedentemente.

Ho quindi provato a farmi un'idea sulla fattibilità di questa idea.

Per quanto riguarda il confronto con javaScript, javaScript è più reattivo perché non ha bisogno di ricaricare la pagina, le animazioni e gli effetti sono più fluidi, però richiede un backend separato per gestire dati e autenticazione.

Per quanto riguarda PHP: Flask e PHP condividono un approccio comune, basato sulla generazione di risposte HTTP a partire da richieste ricevute, con la distinzione che Flask, in quanto framework Python, offre spesso una maggiore flessibilità e una sintassi più concisa rispetto a PHP.

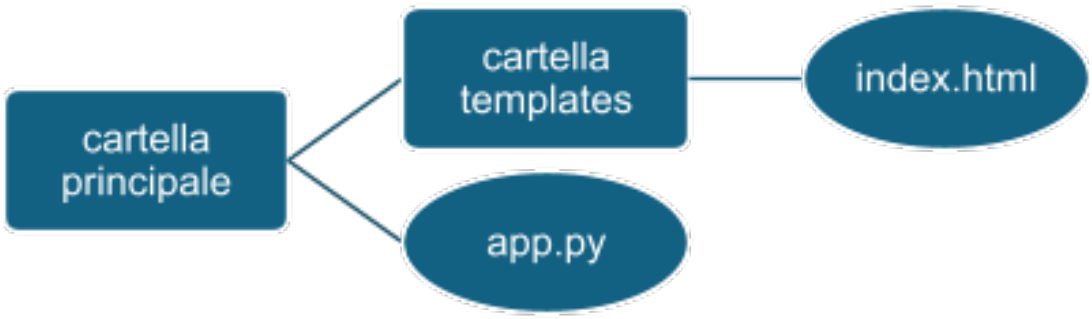
E' vero che Flask integra in modo sinergico diverse tecnologie per la creazione di applicazioni web, in un'applicazione Flask possiamo trovare **Python** (funge da linguaggio di programmazione principale, gestendo la logica lato server e le interazioni con il database), **HTML** (struttura il contenuto delle pagine web), **CSS** (definisce l'aspetto visivo), **JavaScript** (aggiunge

interattività dinamica).

Un vantaggio che possiamo avere con l'utilizzo di Flask è che possiamo anche provare le pagine con backend in locale senza bisogno di installare un software che trasformi il nostro device in un server (come XAMPP o Uniform Server), si può simulare il comportamento di un server e successivamente portare l'applicazione nel web su una piattaforma come PythonAnywhere, Heroku, o Replit.

Per poter utilizzare Flask con Python occorre installare la libreria dei comandi Flask (pip install flask).

Per creare un'applicazione in locale occorre predisporre una cartella, nella quale mettere il programma Python e una sottocartella templates, nella sottocartella va impostata la pagina web, che in questo caso abbiamo chiamato index.html.



Ecco qui di seguito un esempio di codice minimale:

```
from flask import Flask, render_template
app = Flask(__name__)
@app.route('/')
def home():
    return render_template('index.html')
if __name__ == '__main__':
    app.run(debug=True)
```

L'istruzione (app = Flask(__name__)) crea un'istanza dell'applicazione Flask, assegnandola alla variabile app. L'argomento __name__ passa al costruttore il nome del modulo corrente, così Flask può trovare correttamente le risorse statiche e i template.

Il decoratore (@app.route('/')) associa la funzione home() all'URL radice '/' (la home page).

Ogni volta che l'utente visita l'URL radice, viene eseguita quindi la funzione home() definita nella riga successiva, questa

funzione richiama la pagina index.html (che viene cercata nella cartella templates).

L'istruzione (app.run(debug=True)) avvia l'app Flask in modalità debug (che permette di visualizzare messaggi di errore più dettagliati e ricaricare automaticamente l'app ad ogni modifica).

Il template index.html contiene il testo per visualizzare la pagina web:

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>La mia prima app Flask</title>
</head>
<body>
    <h1>Benvenuto nella tua prima applicazione Flask!</h1>
</body>
</html>
```

Per eseguire l'app basta dare un doppio click sul file app.py, si aprirà una finestra del prompt dei comandi:

```
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment.
 Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 103-773-120
```

Per poter vedere il risultato dell'elaborazione bisogna visitare il sito http://127.0.0.1:5000 da un browser (in locale).

In questo primo esempio abbiamo utilizzato Python-Flask per mostrare nel browser una pagina Web statica, possiamo integrare il codice aggiungendo la parte interattiva tramite un form o con l'accesso a un database, in questo caso dovremo gestire la comunicazione HTTP tra server e client.

Il sistema predefinito per il rendering di template in Flask sfrutta un particolare **motore di template**: Jinja2, che permette di inserire variabili, strutture di controllo (if, for), filtri e macro all'interno di file HTML e quindi generare HTML dinamico.

In conclusione, continuo a insegnare JavaScript per presentare una modalità efficace di gestione del frontend delle applicazioni Web, poiché lo reputo ancora la soluzione più appropriata per garantire animazioni fluide e un'interattività avanzata, nonostante l'aumento della curva di apprendimento generato dalla necessità di imparare un nuovo linguaggio.

Per quanto riguarda la parte backend devo ancora esaminare bene le difficoltà intrinseche nel passare a usare SQLAlchemy, che sarebbe una cosa completamente nuova.

L'adozione di Python-Flask per la gestione del backend delle applicazioni Web invece rappresenta una prospettiva interessante, soprattutto per sfruttare le competenze pregresse degli studenti in Python e semplificare l'integrazione con altri strumenti del linguaggio. Tuttavia, rimangono alcune incertezze legate alla curva di apprendimento di SQLAlchemy e alla necessità di approfondire le best practice per garantire efficienza, sicurezza e scalabilità nell'implementazione di soluzioni basate su Flask.

(collegato a lezione 6 del Volume 2 (V2U09) Il framework Flask)