Progetto Programmazione di Reti

Elena Fucci

June 2025

Indice

1 Obiettivo del progetto

Realizzare un semplice server HTTP utilizzando Python e il modulo `socket`. Il server serve un sito web statico con pagine **HTML** e **CSS** che riprendono il design e i contenuti di un negozio di alta moda chiamato *Atelier 101*, rispondendo alle richieste HTTP GET con le pagine richieste o con una pagina di errore 404 se il file non è presente.

2 Struttura e tecnologie utilizzate

- **Linguaggio:** Python 3.x
- **Modulo:** `socket` per gestire le connessioni TCP/IP
- `server.py`: server HTTP minimale in Python
- **Cartella `www/`:** contiene i file statici (HTML, CSS, immagini)
- **Porta di ascolto:** localhost sulla porta 8080

3 Funzionalità implementate

- Ascolto delle **richieste HTTP** in locale sulla porta 8080
- Gestione delle richieste HTTP **GET**
- Risposta con codice **200** e contenuto del file richiesto, se presente
- Risposta con codice **404** e pagina di errore se il file non esiste
- Supporto ai **MIME types** più comuni (`.html`, `.css`, `.png`)
- **Logging** semplice delle richieste (timestamp, metodo, risorsa, codice risposta)

4 Contenuti del sito statico

La cartella `www/` include almeno 3 pagine HTML, ad esempio:

- `index.html` (pagina principale)
- `maglie.html`
- `pantaloni.html`

È presente anche un file CSS (`style.css`) per la formattazione.

5 Funzionamento del server

1. Riceve una richiesta GET e ne estrae il percorso della risorsa
2. Cerca il file nella cartella `www/`
3. Se il file esiste, invia risposta HTTP 200 con il contenuto e header MIME corretti
4. Se il file non esiste, invia risposta HTTP 404 con pagina di errore
5. Registra la richiesta con data, metodo, risorsa e codice risposta

6 Esempio di log generati

```
[2025-06-25 09:52:27] GET /index.html -> 200
[2025-06-25 09:52:27] GET /css/style.css -> 200
[2025-06-25 09:52:27] GET /img/maglie.png -> 200
[2025-06-25 09:52:29] GET /img/maglie/maglia1.png -> 200
[2025-06-25 09:52:32] GET /maglie/maglia1.html -> 404
```

7 Risultati

Il server risponde correttamente alle richieste GET e serve le pagine con gli stili CSS applicati. Le pagine inesistenti restituiscono l'errore 404 personalizzato.

Per eseguire il server

```
python server.py
```

8 Conclusioni e sviluppi futuri

Il progetto ha raggiunto pienamente gli obiettivi prefissati, realizzando un web server minimale in Python in grado di gestire correttamente le richieste HTTP GET sulla porta `localhost:8080`. Il server serve contenuti statici presenti nella cartella `www/`, tra cui almeno tre pagine HTML, fogli di stile CSS e immagini.

Il buon funzionamento del server è garantito dall'utilizzo combinato delle funzionalità minime richieste e di alcune estensioni opzionali:

- La gestione delle richieste GET è robusta e consente di fornire una risposta `200 OK` per i file esistenti, oppure una risposta `404 Not Found` per quelli non presenti.
- È stato implementato il supporto ai **MIME types**, permettendo al client di ricevere correttamente contenuti HTML, CSS e immagini.
- Il server include un sistema di **logging** semplice ma efficace, che registra data, metodo HTTP, risorsa richiesta e codice di risposta.

- Sono state adottate misure di sicurezza base, come la prevenzione del *path traversal*, con una risposta `403 Forbidden` per richieste potenzialmente pericolose.
- A livello di interfaccia, il sito statico include elementi di stile CSS e layout responsive, ispirati a un sito di alta moda (*Atelier 101*), migliorando l'esperienza utente.

Grazie a queste funzionalità, il server dimostra una buona affidabilità nella gestione di contenuti statici e può rappresentare una solida base per progetti più avanzati. Eventuali sviluppi futuri potrebbero includere il supporto per ulteriori metodi HTTP, la gestione dei cookie o l'integrazione con contenuti dinamici.