Web Server Semplice in Python

Introduzione

Il progetto consiste nell'implementare un web server semplice che gestisca richieste di base per file statici. Il server dev'essere in grado di gestire più richieste simultanee restituendo risposte appropriate ad ogni client.

Obiettivi del Progetto

- Implementazione funzionante del server che può gestire richieste GET per file statici.
- Capacità del server di gestire più richieste simultaneamente in modo concorrente o tramite thread.
- Corretta gestione degli header HTTP e codici di stato nelle risposte del server.

Descrizione del Server

1) Importazioni:
    - Modulo 'sys' serve per accettare informazioni da riga di comando.
    - Modulo 'signal' consente di definire handlers per la gestione di un segnale.
    - Modulo 'http.server' crea e ascolta sul socket http.
    - Modulo 'socketserver' permette di utilizzare i socket.
2) Inizializzazioni:
    - Il server http si mette in ascolto sulla porta fornita da riga di comando all'esecuzione, altrimenti si usa la porta 8080 di default.
    - Si crea 'server' per poter gestire più richieste contemporanee del server tramite thread.
    - Impostando i due flag 'server.daemon_threads' e 'server.allow_reuse_address' a 'True' si assicura che in caso di chiusura del server vengano chiusi in modo corretto tutti i threads e che il socket possa essere riutilizzato anche se non rilasciato.
3) Funzioni:
    - La funzione 'signal_handler()', chiamata in caso del segnale 'SIGINT' (CTRL+C), chiude il server.
    - La funzione 'signal.signal()' permette di definire handler personalizzati. Se da tastiera arriva il segnale 'SIGINT' chiama l'handler 'signal_handler()'.
4) Loop di esecuzione:
    - È un loop infinito in cui il server rimane in ascolto finché non riceve una richiesta, la quale nel caso viene gestita.

Esecuzione

Per eseguire si può lanciare il server dall'editor di codice,o da riga di comando con il comando 'python web_server.py' a cui si può aggiungere il numero della porta su cui si vuole mettere in ascolto il server.

- Il server si mette in ascolto per richieste http.
- In caso di arrivo di una richiesta, questa viene gestita.
- In caso di digitazione della sequenza 'CTRL+C' viene interrotta l'esecuzione del server, compresi tutti i thread creati per gestire le ipotetiche richieste multiple.

Conclusioni

Il sistema implementato è un esempio molto banale ma funzionante di web server http. È in grado di gestire richieste per pagine statiche anche in multithreading.