Kolegium Nauk Przyrodniczych
Uniwersytet Rzeszowski

Przedmiot:

Sieci Komputerowe

Projekt prostego serwera www

Wykonał:

Konrad Szyszlak 131524

Prowadzący: Mgr inż. Jarosław Szkoła

Rzeszów 2025

Spis Treści

Dokumentacja projektu serwera HTTP

Opis

Projekt jest prostym serwerem HTTP, który nasłuchuje na określonych portach i obsługuje żądania HTTP. Serwer pozwala na dynamiczne ładowanie konfiguracji serwera z pliku config.json, obsługę różnych typów plików (HTML, CSS, JS, obrazy) oraz zwracanie odpowiednich stron błędów (403, 404, 405, 500) w przypadku problemów z żądaniem.

Struktura plików

```
├── config.json
├── error_pages
│   ├── 400.html
│   ├── 403.html
│   ├── 404.html
│   ├── 405.html
│   └── 500.html
├── main.py
├── README.md
└── static
```

- *config.json - Plik konfiguracyjny definiujący listę serwerów, ich porty, katalogi bazowe oraz dozwolone rozszerzenia plików.*
- *error_pages/ - Katalog zawierający pliki HTML dla stron błędów (np. 404.html, 403.html).*
- *static/ - Domyślny katalog bazowy do serwowania plików.*
- *error_pages/ w tym katalogu są strony z obsługiwanymi błędami*

Projekt korzysta ze standardowych bibliotek dlatego nie ma pliku requirments.txt

Funkcje

Główne funkcje

load_config()

Ładuje konfigurację z pliku config.json. Jeśli plik nie istnieje, zwraca pusty słownik.

handle_client(client_socket, base_dir, allowed_extensions)

Obsługuje połączenie z klientem:

- Parsuje żądanie HTTP.
- Waliduje metodę żądania.
- Sprawdza, czy plik istnieje i czy jego rozszerzenie jest dozwolone.
- Wysyła odpowiedź HTTP z treścią pliku lub odpowiednią stroną błędu.

send_response(client_socket, status_code, status_message, body=None, content_type='text/plain')

Wysyła odpowiedź HTTP do klienta.

- Parametry:
    - client_socket: gniazdo klienta.
    - status_code: kod statusu HTTP.
    - status_message: wiadomość statusowa HTTP.
    - body: treść odpowiedzi (domyślnie brak).
    - content_type: typ treści (domyślnie text/plain).

send_error_page(client_socket, base_dir, status_code)

Wysyła stronę błędu do klienta na podstawie kodu statusu HTTP.

- Parametry:
  - client_socket: gniazdo klienta.
  - base_dir: katalog bazowy.
  - status_code: kod statusu HTTP (np. 404, 403).

get_status_message(status_code)

Zwraca wiadomość statusową HTTP na podstawie kodu statusu.

- Obsługiwane kody: 403, 404, 405, 500.

is_allowed_extension(file_path, allowed_extensions)

Sprawdza, czy plik ma dozwolone rozszerzenie.

- Parametry:
  - file_path: ścieżka do pliku.
  - allowed_extensions: lista dozwolonych rozszerzeń.
- Zwraca True lub False.

get_content_type(file_path)

Zwraca typ treści na podstawie rozszerzenia pliku (np. text/html, image/png).

start_server(port, base_dir, allowed_extensions)

Uruchamia serwer nasłuchujący na podanym porcie.

- Parametry:
  - port: numer portu.
  - base_dir: katalog bazowy.
  - allowed_extensions: lista dozwolonych rozszerzeń.

Główna funkcja

W sekcji if __name__ == '__main__': serwer jest uruchamiany na podstawie konfiguracji z pliku config.json. Każdy serwer działa w osobnym wątku.

Plik konfiguracyjny

Przykład config.json:

```
{
  "servers": [
    {
      "port": 8080,
      "base_dir": "./static/strona1",
      "allowed_extensions": [".html", ".css", ".js", ".png", ".jpg"]
    }
  ]
}
```

- servers: lista serwerów.
- port: numer portu, na którym serwer nasłuchuje.
- base_dir: katalog bazowy dla plików statycznych.
- allowed_extensions: lista dozwolonych rozszerzeń plików.

Obsługiwane kody statusu

Kod	Wiadomość	Opis
403	Forbidden	Brak dostępu do zasobu.
404	Not Found	Zasób nie został znaleziony.
405	Method Not Allowed	Żądana metoda nie jest obsługiwana.
500	Internal Server Error	Błąd wewnętrzny serwera.

Uruchamianie serwera

1. Upewnij się, że plik config.json istnieje i zawiera poprawne dane.
2. Przejdź do katalogu simpleWWW: cd ./simpleWWW
3. Uruchom skrypt: *python main.py*
4. Serwer rozpocznie nasłuchiwanie na określonych portach i będzie serwować pliki z podanych katalogów bazowych.

Przykładowe żądania HTTP

GET

Żądanie:

GET /index.html HTTP/1.1
Host: localhost

Odpowiedź (przykład):

HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 1234

<zawartość pliku index.html>

HEAD

Żądanie:

HEAD /style.css HTTP/1.1
Host: localhost

Odpowiedź (przykład):

HTTP/1.1 200 OK
Content-Type: text/css
Content-Length: 567

Logowanie

Serwer loguje informacje o każdym połączeniu oraz błędach w konsoli.

- Informacje o ścieżce żądanego pliku.
- Informacje o błędach (np. brak pliku, nieobsługiwana metoda).

Wątki

Każde połączenie z klientem jest obsługiwane w osobnym wątku, co pozwala na równoczesną obsługę wielu klientów.

Zatrzymywanie serwera

Serwer można zatrzymać za pomocą kombinacji klawiszy Ctrl+C. Skrypt zamknie wszystkie aktywne wątki i zakończy nasłuchiwanie.