Документация на проекта Web Scraper

Описание на проекта

Проектът **Web Scraper** реализира клиент-сървър архитектура, чрез която извлича уеб съдържание. Разработен на Java с помощта на Maven и библиотеката Jsoup. Комуникацията се осъществява чрез сокети. Клиентът изпраща заявки към сървъра, посочвайки брой редове с текст, които желае да извлече от HTML документ. Сървърът обработва заявката, извлича съдържанието на HTML файла и връща отговор на клиента.

Основни функционалности:

- Обработване на текстово съдържание от HTML файл с помощта на библиотеката **Jsoup**.
- Поддръжка на комуникация между клиент и сървър чрез сокети.
- Управление на множество клиентски заявки с многопоточност.

Архитектура

Компоненти:

1. **Сървър**:
   - Използва клас **ServerSocket** за създаване на сървър, който слуша на порт 8080.
   - Обработва множество клиенти чрез отделни нишки.
   - Извлича текстово съдържание от HTML файл и изпраща резултатите към клиента.

2. **Клиент**:
   - Свързва се със сървъра чрез сокет.
   - Изпраща брой редове, които желае да извлече.
   - Получава и отпечатва резултатите от сървъра.

Подробно описание на кода

Сървърна част (WebScraperServer)

1. **Инициализация на сървъра**:
   - Сървърът стартира на порт 8080, използвайки **ServerSocket**.
   - Приема клиентски заявки чрез метод **accept()**.

2. **Обработка на заявки**:
   - Всяка клиентска връзка се обработва в отделна нишка с помощта на вътрешен клас **ClientHandler**.
   - Нишката:
     - Чете входните данни от клиента.
     - Извиква метод **getFirstNRows** за извличане на първите *n* реда текст от HTML файл.
     - Изпраща резултата обратно на клиента.

3. **Извличане на HTML**:
    - Използва **Jsoup** за обработка на HTML файл (**example.html**).
    - Методът **getFirstNRows** извлича текст от първите *n* елемента с текстово съдържание.

Клиентска част (WebScraperClient)

1. **Свързване със сървъра**:
    - Клиентът се свързва със сървъра на адрес **localhost** и порт 8080 чрез **Socket**.

2. **Изпращане на заявки**:
    - Потребителят въвежда брой редове за извличане или "exit" за изход.
    - Клиентът изпраща броя редове на сървъра чрез **PrintWriter**.

3. **Получаване на отговор**:
    - Клиентът чете отговора на сървъра чрез **BufferedReader** и го отпечатва на екрана.