Задание к практической работе 1

Измените работу простого веб-сервера, чтобы он корректно возвращал json, отображаемый браузером и txt, которой скачивается браузером.

Вводные

Итак, вам дан базовый код на Java, который представляет собой простой пример минимального HTTP-сервера и используется только для демонстрации базовой работы заголовков и ответов.

Класс ConnectionHandler:

```
import java.io.*;
import java.net.Socket;
import java.nio.charset.StandardCharsets;

public class ConnectionHandler {

    private static final String HTTP_HEADERS = "HTTP/1.1 200
OK\n" +
            "Date: Mon, 18 Sep 2023 14:08:55 +0200\n" +
            "HttpServer: Simple Webserver\n" +
            "Content-Length: 180\n" +
            "Content-Type: text/html\n";

    private static final String HTTP_BODY = "<!DOCTYPE html>\n"
+
            "<html lang=\"en\">\n" +
            "<head>\n" +
            "<meta charset=\"UTF-8\">\n" +
            "<title>Simple Http Server</title>\n" +
            "</head>\n" +
            "<body>\n" +
            "<h1>Hi!</h1>\n" +
            "<p>This is a simple line in html.</p>\n" +
            "</body>\n" +
            "</html>\n" +
            "\n";
    private Socket socket;

    public ConnectionHandler(Socket socket) {
        this.socket = socket;
        handle();
    }

    public void handle() {
        try {
            var inputStreamReader = new BufferedReader(
                    new
InputStreamReader(socket.getInputStream(),
```

```
StandardCharsets.US_ASCII));
            var outputStreamWriter = new BufferedWriter(
                    new
OutputStreamWriter(socket.getOutputStream(),
StandardCharsets.US_ASCII));

            parseRequest(inputStreamReader);
            writeResponse(outputStreamWriter);

        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    }

    private void parseRequest(BufferedReader inputStreamReader)
throws IOException {
        var request = inputStreamReader.readLine();

        while (request != null && !request.isEmpty()) {
            System.out.println(request);
            request = inputStreamReader.readLine();
        }
    }

    private void writeResponse(BufferedWriter
outputStreamWriter) {
        try {
            outputStreamWriter.write(HTTP_HEADERS);
            outputStreamWriter.newLine();
            outputStreamWriter.write(HTTP_BODY);
            outputStreamWriter.newLine();
            outputStreamWriter.flush();
        } catch (IOException e) {
            throw new RuntimeException(e);
        }

    }
}
```

Класс HttpServer:

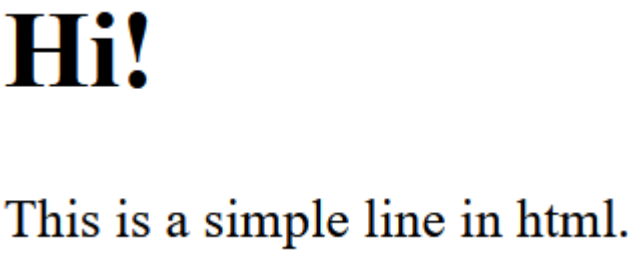
```
import java.io.IOException;
import java.net.ServerSocket;

public class HttpServer {

    private int tcpPort;
    public HttpServer(int tcpPort) {
        this.tcpPort = tcpPort;
    }
```

```
    public void startServer() {
        try (
                var serverSocket = new
ServerSocket(this.tcpPort);
        ) {
            System.out.println("Server accepting requests on
port " + tcpPort);

            while (true) {
                var acceptedSocket = serverSocket.accept();
                var connectionHandler = new
ConnectionHandler(acceptedSocket);
                connectionHandler.handle();
            }

        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    }

}
```

Класс Main:

```
public class Main {
    public static void main(String[] args) {
        new HttpServer(8585).startServer();
    }
}
```

Запустите код, откройте браузер и перейдите по ссылке - http://localhost:8585/ и вы увидите ответ от нашего сервера.



Класс ConnectionHandler обрабатывает каждое входящее соединение от клиента (браузера). У него есть два приватных статических поля HTTP_HEADERS и HTTP_BODY, которые содержат заголовки HTTP-ответа и тело HTML-страницы.

У класса есть приватное поле socket, представляющее собой сокет, через который происходит обмен данными с клиентом (браузером).

Метод handle() обрабатывает входящее соединение:

- Создает потоки для чтения и записи данных из/в сокет.
- Вызывает метод parseRequest(), который читает и выводит HTTP-запрос от клиента.
- Вызывает метод writeResponse(), который отправляет HTTP-ответ клиенту.

Класс HttpServer представляет собой сам HTTP-сервер. В классе есть приватное поле tcpPort, которое хранит значение порта, на котором сервер будет слушать входящие соединения. Конструктор public HttpServer(int tcpPort) инициализирует объект HttpServer с указанным портом. Метод startServer() запускает сервер, принимая входящие соединения от клиентов и создавая для каждого из них объект ConnectionHandler, который обрабатывает соединение.

В классе Main создается объект HttpServer с портом 8585 и вызывается метод startServer() для запуска сервера.

Поэтому, когда приложение запускается, оно создает HTTP-сервер, который слушает порт 8585. Когда клиент (браузер) подключается к этому порту, сервер создает экземпляр ConnectionHandler для обработки соединения с клиентом, который затем читает HTTP-запрос от клиента и отправляет ему HTTP-ответ.

Задание

Измените работу простого веб-сервера, чтобы он корректно возвращал json, отображаемый браузером и txt, которой скачивается браузером со следующими URL:

/json — возвращает JSON-ответ.

/download — возвращает текстовый файл для скачивания.