Alens Evra, Sam Blamo

Dr. Li

CSC-360-02

October 20, 2024

Project 1 - Building a Multi-Threaded Web Server

The purpose of this project was to support a multithreaded webserver. Utilizing socket and intercepting http request and response , We were able to intercept requests on our port . We intercept a request and relocate on a different web page based on that request. In our case this program relocates to google, using port for our HttpMovedRequest 5111 and our original site is on local host was 8888 but since it was overused it is now our HttpRquest at localhost 8111.

This project was a team collaboration between Sam Blamo and Alens Evra. In this collaboration. Alens took on Part A where we had to implement the Http request and the listening Socket. Sam took on Part B where we have to extract the filename and imeplementing the listening socket. Part C was a joint collaboration ,we researched the given sites(“https://www.javatpoint.com/java-nio-selector”) to establish the ServerSocket Signal. Overall to use this server we would run Java and call our local html path with our local host for example (“http://host.someschool.edu:8888/index.html”). Overall this project allows users to redirect a webserver using a different port.

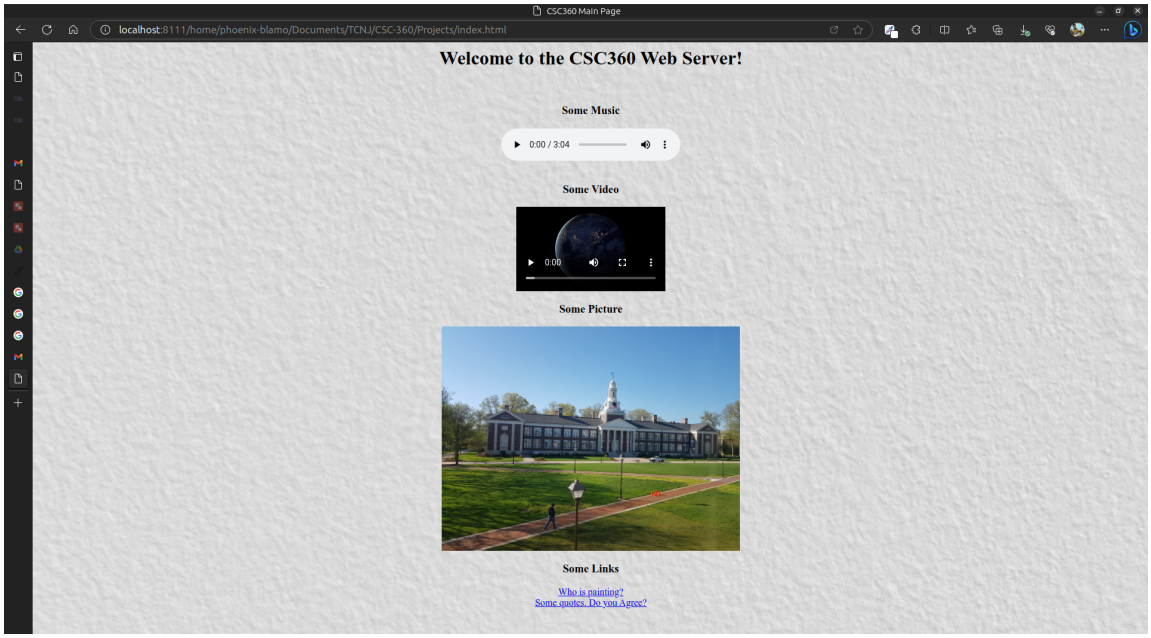
CITE WORKS

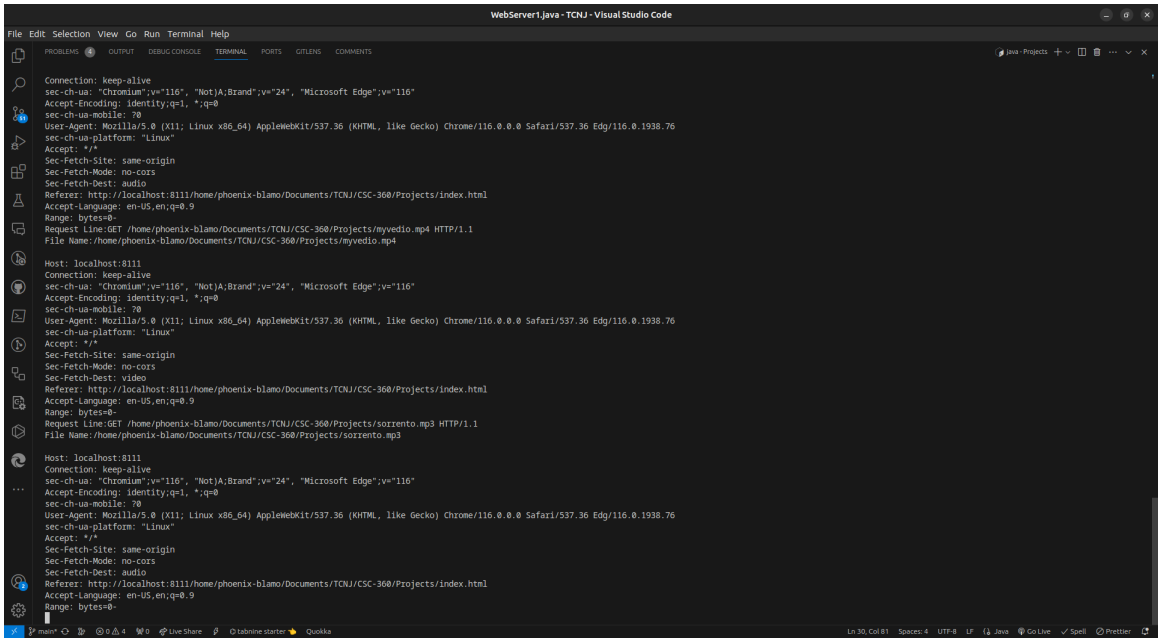
https://www.javatpoint.com/java-nio-serversocketchannel

• https://www.javatpoint.com/java-nio-selector

• https://stackoverflow.com/questions/2819274/listening-for-tcp-and-udp-

Requests-on-the-same-por


CSC360 Main Page
localhost:8111/home/phoenix-blamo/Documents/TCNJ/CSC-360/Projects/index.html
Welcome to the CSC360 Web Server!
Some Music
0:00 / 3:04
Some Video
0:00
Some Picture
Some Links
Who is painting?
Some quotes. Do you Agree?


WebServer1.java - TCNJ - Visual Studio Code
File Edit Selection View Go Run Terminal Help
PROBLEMS 4 OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS COMMENTS
Connection: keep-alive
sec-ch-ua: "Chromium";v="116", "NotJA;Brand";v="24", "Microsoft Edge";v="116"
Accept-Encoding: identity;q=1, *;q=0
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36 Edg/116.0.1938.76
sec-ch-ua-platform: "Linux"
Accept: */*
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: no-cors
Sec-Fetch-Dest: audio
Referer: http://localhost:8111/home/phoenix-blamo/Documents/TCNJ/CSC-360/Projects/index.html
Accept-Language: en-US,en;q=0.9
Range: bytes=0-
Request Line:GET /home/phoenix-blamo/Documents/TCNJ/CSC-360/Projects/myvedio.mp4 HTTP/1.1
File Name:/home/phoenix-blamo/Documents/TCNJ/CSC-360/Projects/myvedio.mp4
Host: localhost:8111
Connection: keep-alive
sec-ch-ua: "Chromium";v="116", "NotJA;Brand";v="24", "Microsoft Edge";v="116"
Accept-Encoding: identity;q=1, *;q=0
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36 Edg/116.0.1938.76
sec-ch-ua-platform: "Linux"
Accept: */*
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: no-cors
Sec-Fetch-Dest: video
Referer: http://localhost:8111/home/phoenix-blamo/Documents/TCNJ/CSC-360/Projects/index.html
Accept-Language: en-US,en;q=0.9
Range: bytes=0-
Request Line:GET /home/phoenix-blamo/Documents/TCNJ/CSC-360/Projects/sorrento.mp3 HTTP/1.1
File Name:/home/phoenix-blamo/Documents/TCNJ/CSC-360/Projects/sorrento.mp3
Host: localhost:8111
Connection: keep-alive
sec-ch-ua: "Chromium";v="116", "NotJA;Brand";v="24", "Microsoft Edge";v="116"
Accept-Encoding: identity;q=1, *;q=0
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36 Edg/116.0.1938.76
sec-ch-ua-platform: "Linux"
Accept: */*
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: no-cors
Sec-Fetch-Dest: audio
Referer: http://localhost:8111/home/phoenix-blamo/Documents/TCNJ/CSC-360/Projects/index.html
Accept-Language: en-US,en;q=0.9
Range: bytes=0-

```
        String statusLine = null;
        String contentTypeLine = null;
        String entityBody = null;
        if(fileExists){
            statusLine = "HTTP/1.1 301 Moved Permanently";
            contentTypeLine = "Location: https://www.google.com/"+ CRLF;
        } else {
            statusLine = "HTTP/1.1 404 N Found";
            contentTypeLine = "text/html";
            entityBody = "<HTML><HEAD><TITLE>Not Found</TITLE></HEAD><BODY>Not Found</BODY></HTML>";
        }

        String headerLine = null;
        while((headerLine = br.readLine()).length() != 0){
            System.out.println(headerLine);
        }


        String response = statusLine + CRLF + contentTypeLine +  CRLF;
        if(fileExists){
            os.writeBytes(response);
            sendBytes(fis, os);
        } else {
            os.writeBytes(response + CRLF + entityBody);
        }

        os.close();
        br.close();
        socket.close();
    }
  }
}
```

```
You, 4 seconds ago | 2 authors (You and others)
final static class MovedRequest implements Runnable {
    final static String  CRLF = "\r\n";
    Socket socket;

    public MovedRequest(Socket socket) {
        this.socket = socket;
    }

    public void run(){
        try {
            processRequest();
        } catch (Exception e){
            System.out.println(e);
        }
    }

    private static void sendBytes(FileInputStream fis, OutputStream os) throws Exception {
        byte[] buffer = new byte[1024];
        int bytes = 0;

        while((bytes = fis.read(buffer)) != -1){
            os.write(buffer, 0, bytes);
        }
    }

    private void processRequest() throws Exception {
        InputStream inputStream = socket.getInputStream();
        DataOutputStream os = new DataOutputStream(socket.getOutputStream());

        InputStreamReader reader = new InputStreamReader(inputStream);
        BufferedReader br = new BufferedReader(reader);

        String requestLine = br.readLine();
        System.out.println("Request Line:" + requestLine);

        StringTokenizer tokens = new StringTokenizer(requestLine);
        tokens.nextToken();
        String fileName = tokens.nextToken();
        FileInputStream fis = null;
        boolean fileExists = true;
        try {
            fis = new FileInputStream(fileName);
        } catch (FileNotFoundException e) {

            fileExists = false;
        }
```

```
            StringTokenizer tokens = new StringTokenizer(requestLine);
            tokens.nextToken();
            String fileName = tokens.nextToken();
            System.out.println("File Name:" + fileName+ "\n");
            FileInputStream fis = null;
            boolean fileExists = true;
            try {
                fis = new FileInputStream(fileName);
            } catch (FileNotFoundException e) {

                fileExists = false;
            }

            String statusLine = null;
            String contentTypeLine = null;
            String entityBody = null;
            if(fileExists){
                statusLine = "HTTP/1.1 200 OK";
                contentTypeLine = "Content-type: " +
                contentType( fileName ) + CRLF;
            } else {
                statusLine = "HTTP/1.1 404 N Found";
                contentTypeLine = "text/html";
                entityBody = "<HTML><HEAD><TITLE>Not Found</TITLE></HEAD><BODY>Not Found</BODY></HTML>";
            }

            String headerLine = null;
            while((headerLine = br.readLine()).length() != 0){
                System.out.println(headerLine);
            }


            String response = statusLine + CRLF + contentTypeLine +  CRLF;
            if(fileExists){
                os.writeBytes(response);
                sendBytes(fis, os);
            } else {
                os.writeBytes(response + CRLF + entityBody);
            }

            os.close();
            br.close();
            socket.close();
        }
    }
```

```java
final static class HttpRequest implements Runnable {
    final static String  CRLF = "\r\n";
    Socket socket;

    public HttpRequest(Socket socket) {
        this.socket = socket;
    }

    public void run(){
        try {
            processRequest();
        } catch (Exception e){
            System.out.println(e);
        }
    }

    private static void sendBytes(FileInputStream fis, OutputStream os) throws Exception {
        byte[] buffer = new byte[1024];
        int bytes = 0;

        while((bytes = fis.read(buffer)) != -1){
            os.write(buffer, 0, bytes);
        }
    }

    private static String contentType(String fileName) {
        if(fileName.endsWith(".htm") || fileName.endsWith(".html")) {
            return "text/html";
        }
        if(fileName.endsWith(".png") || fileName.endsWith(".jpeg") || fileName.endsWith(".gif") || fileName.endsWith(".tiff")) {
            int index = fileName.lastIndexOf(".");
            String fileType = fileName.substring(index+1);
            return "image/" + fileType;
        }
        return "application/octet-stream";
    }

    private void processRequest() throws Exception {
        InputStream inputStream = socket.getInputStream();
        DataOutputStream os = new DataOutputStream(socket.getOutputStream());

        InputStreamReader reader = new InputStreamReader(inputStream);
        BufferedReader br = new BufferedReader(reader);

        String requestLine = br.readLine();
        System.out.println("Request Line:" + requestLine);
```

```java
import java.io.* ;
import java.net.* ;
import java.nio.channels.SelectionKey;
import java.nio.channels.Selector;
import java.nio.channels.ServerSocketChannel;
import java.nio.channels.SocketChannel;
import java.util.* ;
You, 2 minutes ago | 2 authors (You and others)
public final class WebServer {
    Run | Debug
    public static void main(String argv[]) throws Exception {
        Selector selector = Selector.open();
        int port = 8888;        You, 40 seconds ago • Uncommitted changes
        int port1 = 5555;

        ServerSocketChannel serverChannel = ServerSocketChannel.open();
        ServerSocketChannel serverChannel1 = ServerSocketChannel.open();

        serverChannel.configureBlocking(false);
        serverChannel.socket().bind(new InetSocketAddress(port));
        serverChannel.register(selector, SelectionKey.OP_ACCEPT);

        serverChannel1.configureBlocking(false);
        serverChannel1.socket().bind(new InetSocketAddress(port1));
        serverChannel1.register(selector, SelectionKey.OP_ACCEPT);

        while (true) {
            SocketChannel listeningSocket = serverChannel.accept();
            SocketChannel listeningSocket1 = serverChannel.accept();
            if (listeningSocket != null) {
                HttpRequest request = new HttpRequest(listeningSocket.socket());
                Thread thread = new Thread(request);
                thread.start();
            }
            if(listeningSocket1 != null){
                MovedRequest request1 = new MovedRequest(listeningSocket1.socket());
                Thread thread1 = new Thread(request1);
                thread1.start();
            }
        }
    }
```