/\*--------------------------------------------------------

1. Name / Date: Jessica Bender. 3/7/2020

2. Java version used, if not the official version for the class:

My Java:

java version "9.0.4"

Java(TM) SE Runtime Environment (build 9.0.4+11)

Java HotSpot(TM) 64-Bit Server VM (build 9.0.4+11, mixed mode)

3. Precise command-line compilation examples / instructions:

e.g.:

> javac MyWebServer.java

4. Precise examples / instructions to run this program:

> java MyWebServer

All acceptable commands are displayed on the various consoles.

This runs across machines, in which case you have to pass the IP address of

the server to the clients. For exmaple, if the server is running at

140.192.1.22 then you would type:

> java JokeClient 140.192.1.22

> java JokeClientAdmin 140.192.1.22

5. List of files needed for running the program.

e.g.:

a. serverlog.txt

b. MyWebServer.java

c. http-streams.txt

d. checklist-mywebserver.html

5. Notes:

alot of code taken from mini webserver assignment. Code also taken from MyListener.java MyTelnetClient.java

has a bug. does not list all the files for some reason.

----------------------------------------------------------\*/

//code starts here

import java.io.\*; //imports all directories located in java.io

import java.net.\*; //imports all directories located in java.net

import java.util.StringTokenizer;

class ListenWorker extends Thread { //start of ListenWorker class

Socket sock; //new Socket called sock

ListenWorker (Socket s) {//start of ListenWorker takes in a Socket

sock = s;// assigns the Socket we took in s to Socket sock

} //end of ListenWorker

public void run(){ //start of run

PrintStream out = null; // initialize a PrintStream called out and sets its value to null

BufferedReader in = null; //initialize a BufferedReader called in and sets its value to null

try {//Start try statement. If try doesnt work moved down to catch statement

out = new PrintStream(sock.getOutputStream()); //makes a new PrintStream and assigns it to out and gets the OutputStream from the Socket sock

in = new BufferedReader //makes a new BufferedReader and assigns it to in

(new InputStreamReader(sock.getInputStream())); //gets the InputStream from the Socket sock and makes a new InputStreamReader and puts in inside the BufferedReader in

String inLine = in.readLine();//readslines of in and assigns it to inLine

String file = "";

String MIME = "";

StringTokenizer token = new StringTokenizer(inLine, " ");

if (file.indexOf("..") >= 0) { // security concern

System.out.println("Error! try again later.");//prints error

}

if(token.nextToken().equals("GET") && token.hasMoreTokens()){

file=token.nextToken();

}

else{

System.out.println("Error! try again later.");//prints error

}

if(inLine != null){//checks to see if inline is empty

if(file.endsWith(".txt")){//checks to see if file is text type

MIME = "text/plain";//MIME is plain

System.out.println("The file in use is: "+file+". This files MIME-Type is:" + MIME +".");//prints filename and MIME type

File(file, out, MIME);//sends file, out, MIME to File

}//end if

else if(file.endsWith(".html") || file.endsWith("/") || file.endsWith("cgi")){//checks to see if file is any type of html

MIME = "text/html";//MIME is HTML

System.out.println("The file in use is: "+file+". This files MIME-Type is:" + MIME +".");//prints filename and MIME type

if(file.endsWith(".html")){

File(file, out, MIME);//sends file, out, MIME to File

}

else if(file.endsWith("/")){

directories(file, out, MIME);//sends file, out, MIME to File

}

else if(file.endsWith("cgi")){

add(file, out, MIME);//sends file, out, MIME to File

}

}//ends else if

else{//if none of the above:

MIME = "text/plain";//MIME is plain

System.out.println("The file in use is: "+file+". This files MIME-Type is:" + MIME +".");//prints filename and MIME type

File(file, out, MIME);//sends file, out, MIME to File

}//ends else

}// end if

else{//if inline is empty

System.out.println("Error. Please try again!");//prints out error

}//end else

sock.close();

}//end try

catch (IOException x) { //start catch. Catches an IOException if the try fails.

System.out.println("Error: Connetion reset. Listening again...");//prints message if try fails

}//end catch

}//end run

public void File(String fileName, PrintStream out, String MIME){

try{//

int indexof = fileName.indexOf("/");

if (indexof == 0) {

fileName = fileName.substring(1);

}

InputStream input;// new InputStream called input

File name;

long len;

input = new FileInputStream(fileName);

name = new File(fileName);

len = name.length();

System.out.println("The file in use is: "+fileName+". This files MIME-Type is:" + MIME +". the length is: "+len+".");

out.println("HTTP/1.1 200 OK");//sends to out

out.println("Connection: close"); //sends to out

out.println("Content-Length: 400"); //sends to out

out.println("Content-Type: text/html \r\n\r\n");//sends to out

byte[] bites = new byte[10000]; //byte array of 10k

int num = input.read(bites);

out.write(bites, 0, num); //sends files to browser

System.out.write(bites, 0, num);//prints files

out.flush();//flushes out.

input.close();//clsoes input

}

catch (IOException x) { //start catch. Catches an IOException if the try fails.

System.out.println("Error: Connetion reset. Listening again...");//prints message if try fails

}//end catch

}//end file

public void directories(String directory, PrintStream out, String MIME) throws IOException{

BufferedWriter write;

File name;

String Indexhtml = "Index.html";

File newfile;

File[] list;

write = new BufferedWriter(new FileWriter("Index.html"));

name = new File("./" + directory + "/");

list = name.listFiles();//lists files in name

String Directory = "<html> <h1> Hello Directory!</h1> <p><p> <hr> <p>";//send this to the webpage

write.write(Directory);//pritns the tirecty list to termial

write.write("Directory: " + directory + "<p><p>");

write.write("click on here to reload to main page (points to http://localhost:2540/) :<a href=\""+ "http://localhost:2540"+ "/\">"+"<u>HERE</u>"+ "</a> <p><p>"); //will reload http://localhost:2540/

for(File file : list ){

String Name = file.getName();

String DirectoryLink = "<a href=\""+ Name + "/\">"+Name+ "</a> <p><p>";//send this to the webpage

String FileLink = "<a href=\"" + Name + "\" >" + Name + "</a> <p><p>";//send this to the webpage

if(Name.startsWith(Indexhtml)){//checks to see if Name starts with "Index.html"

continue;//continues

}

if(file.isDirectory()){//checks to see if there is a Directory

write.write(DirectoryLink);//writes the DirectoryLink

}

if(file.isFile()){//checks to see if there is a Directory

write.write(FileLink);//writes the FileLink

}

write.flush();//flushes write

}//end for

write.write("</html>");//cloases html tag

newfile = new File(Indexhtml);

out.println("HTTP/1.1 200 OK");//sends to out

out.println("Connection: close"); //sends to out

out.println("Content-Length: 400"); //sends to out

out.println("Content-Type: text/html \r\n\r\n");//sends to out

InputStream input;

input = new FileInputStream(Indexhtml);

System.out.println("The Directory in use is: "+directory+". This directory's MIME-Type is:" + MIME +".");

byte[] bites = new byte[10000]; //byte array of 10k

int num = input.read(bites);

out.write(bites, 0, num); //sends files to browser

System.out.write(bites, 0, num);//prints files

write.close();//close write

out.flush();//flushes out

newfile.delete();//deletes files

}

public void add(String url, PrintStream out, String MIME){

String equ = "="; // new string called equ and is set to =

String and = "&"; // new string called and and is set to &

url = url.replaceAll(and, equ); //replaces all & with = in url

String HTTP = " HTTP"; //new string called HTTP and is set to HTTP

url = url.replaceAll(HTTP, equ); //replaces all HTTP with = in url

String split[] = url.split("="); //splits url by the =

String name = split[1]; // gets the name from the split at index 1

String num1 = split[3]; // gets the first from the split at index 3

String num2 = split[5];// gets the second number from the split at index 5

int sum = Integer.parseInt(num1) + Integer.parseInt(num2);// add num1 and num2

String hello = "<h2> Hello " + name + "! The sum of "+ num1 +" and " + num2 + " is " + sum +". </h2> <p><p> <hr> <p>"; //prints the user

out.println(hello);//sends hello to out.

System.out.println(hello);//prints hello.

}

// System.out.println("Sending the HTML Reponse now: " + Integer.toString(MyWebServer.i) + "\n" );//prints this to terminal

// //example: Sending the HTML Reponse now: 5

// String HTMLResponse = "<html> <h1> Hello Browser World! This is Browser Number:" + Integer.toString(MyWebServer.i++) + "</h1> <p><p> <hr> <p>";//send this to the webpage

// //Example: Hello Browser World! This is Browser Number:5

// String url = in.readLine(); //reads in and sets it to string variable url

// // local definition of socketData of type String

// String equ = "="; // new string called equ and is set to =

// String and = "&"; // new string called and and is set to &

// url = url.replaceAll(and, equ); //replaces all & with = in url

// String HTTP = " HTTP"; //new string called HTTP and is set to HTTP

// url = url.replaceAll(HTTP, equ); //replaces all HTTP with = in url

// String split[] = url.split("="); //splits url by the =

// String name = split[1]; // gets the name from the split at index 1

// String num1 = split[3]; // gets the first from the split at index 3

// String num2 = split[5];// gets the second number from the split at index 5

// int sum = Integer.parseInt(num1) + Integer.parseInt(num2);// add num1 and num2

// //int x =Integer.parseInt(num1);

// //int y =Integer.parseInt(num2);

// //int summ = x + y;

// //out.println(summ);

// String hello = "<html> <h2> Hello " + name + "! The sum of "+ num1 +" and " + num2 + " is " + sum +". </h2> <p><p> <hr> <p>"; //prints the user

// //Example: Hello jessica! The sum of 4 and 5 is 9.

// out.println(hello);

// //out.println(hello);

// for(int j=0; j<6; j++){

// out.println(in.readLine() + "<br>\n");

// } //closes for statement

// out.println("</html>"); //prints message to PrintStream out

// sock.close(); // close the connection to the Socket sock, but does not colose the connecton to the server;

// } //closes try statement

// catch (IOException x) { //start catch. Catches an IOException if the try fails.

// System.out.println("Error: Connetion reset. Listening again...");//prints message if try fails

// }//closes catch statement

}//closes ListenWorker class

public class MyWebServer { //start of MyWebServer class. This is a public class.

public static void main(String a[]) throws IOException { //start of main void method

int q\_len = 6; //sets q\_len to 6

int port = 2540; //sets port to 2540

Socket sock; //new Socket called sock

ServerSocket servsock = new ServerSocket(port, q\_len); //new ServerSocket called servsock. also sets it equal to the port and q\_len that we se above

System.out.println("Jess Bender's MyWebServer running at " + port + ".");//Prints statement. Changed code to have my name and changed the statement so that the variable port would be printed and it was not hardcoded in the string statement.

System.out.println("Point Firefox browser to http://localhost:2540/\n");//prints message

while (true) { //starts wile true statement to look for the next client connection

sock = servsock.accept(); //accepts servsock

new ListenWorker (sock).start();// starts new ListenWorker

}//clsoes while true

}//closes main void statement

}//closes MyWebServer class.

C:\Users\jessi\Documents\2020-2021 school\Winter\CSC435winter2021\MyWebServer>java MyWebServer

Jess Bender's MyWebServer running at 2540.

Point Firefox browser to http://localhost:2540/

The file in use is: /. This files MIME-Type is:text/html.

The Directory in use is: /. This directory's MIME-Type is:text/html.

<html> <h1> Hello Directory!</h1> <p><p> <hr> <p>Directory: /<p><p>click on here to reload to main page (points to http://localhost:2540/) :<a href="http://localhost:2540/"><u>HERE</u></a> <p><p><a href="Bender\_Webserver.zip" >Bender\_Webserver.zip</a> <p><p><a href="cat.html" >cat.html</a> <p><p><a href="dog.txt" >dog.txt</a> <p><p><a href="DS 435 -- Elliott Web Server Tips \_\_center\_.html" >DS 435 -- Elliott Web Server Tips \_\_center\_.html</a> <p><p><a href="http-streams.txt" >http-streams.txt</a> <p><p><a href="ListenWorker.class" >ListenWorker.class</a> <p><p><a href="MyListener.java.txt" >MyListener.java.txt</a> <p><p><a href="MyTelnetClient.java.txt" >MyTelnetClient.java.txt</a> <p><p><a href="MyWebServer Checklist.html" >MyWebServer Checklist.html</a> <p><p><a href="MyWebServer.class" >MyWebServer.class</a> <p><p><a href="MyWebServer.java" >MyWebServer.java</a> <p><p><a href="ReadFiles.java.txt" >ReadFiles.java.txt</a> <p><p><a href="serverlog.txt" >serverlog.txt</a> <p><p><a href="test.class" >test.class</a> <p><p><a href="test.java" >test.java</a> <p><p><a href="WebAdd.html" >WebAdd.html</a> <p><p><a href="WebServerTII.docx" >WebServerTII.docx</a> <p><p><a href="WebServerWorker.class" >WebServerWorker.class</a> <p><p><a href="~$bServerTII.docx" >~$bServerTII.docx</a> <p><p>The file in use is: /cat.html. This files MIME-Type is:text/html.

The file in use is: cat.html. This files MIME-Type is:text/html. the length is: 132.

<html> <body>

<center>

<h1> This is Elliott's </h1>

<h1> CAT </h1>

<h1> html file </h1>

</center>

</body> </html>

The file in use is: /dog.txt. This files MIME-Type is:text/plain.

The file in use is: dog.txt. This files MIME-Type is:text/plain. the length is: 49.

This is Elliott's dog file on condor. Good job!

Exception in thread "Thread-3" java.lang.NullPointerException

at java.base/java.util.StringTokenizer.<init>(StringTokenizer.java:199)

at java.base/java.util.StringTokenizer.<init>(StringTokenizer.java:221)

at ListenWorker.run(MyWebServer.java:72)

C:\Users\jessi\Documents\2020-2021 school\Winter\CSC435winter2021\MyWebServer>java MyListener

Clark Elliott's Port listener running at 2540.

GET / HTTP/1.1

Host: localhost:2540

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:86.0) Gecko/20100101 Firefox/86.0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,\*/\*;q=0.8

Accept-Language: en-US,en;q=0.5

Accept-Encoding: gzip, deflate

Connection: keep-alive

Upgrade-Insecure-Requests: 1

GET /dog.txt HTTP/1.1

Host: localhost:2540

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:86.0) Gecko/20100101 Firefox/86.0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,\*/\*;q=0.8

Accept-Language: en-US,en;q=0.5

Accept-Encoding: gzip, deflate

Connection: keep-alive

Referer: http://localhost:2540/

Upgrade-Insecure-Requests: 1

GET /cat.html HTTP/1.1

Host: localhost:2540

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:86.0) Gecko/20100101 Firefox/86.0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,\*/\*;q=0.8

Accept-Language: en-US,en;q=0.5

Accept-Encoding: gzip, deflate

Connection: keep-alive

Referer: http://localhost:2540/

Upgrade-Insecure-Requests: 1