```
/*******************************************************************************

   SimpleWebServer.java


   This toy web server is used to illustrate security vulnerabilities.
   This web server only supports extremely simple HTTP GET requests.

   This file is also available at http://www.learnsecurity.com/ntk

*******************************************************************************/

package com.learnsecurity;

import java.io.*;
import java.net.*;
import java.util.*;

public class SimpleWebServer {

    /* Run the HTTP server on this TCP port. */
    private static final int PORT = 8080;

    /* The socket used to process incoming connections
       from web clients */
    private static ServerSocket dServerSocket;

    public SimpleWebServer () throws Exception {
        dServerSocket = new ServerSocket (PORT);
    }

    public void run() throws Exception {
        while (true) {
            /* wait for a connection from a client */
            Socket s = dServerSocket.accept();
            /* then process the client's request */
            processRequest(s);
        }
    }

    /* Reads the HTTP request from the client, and
       responds with the file the user requested or
       a HTTP error code. */
    public void processRequest(Socket s) throws Exception {
        /* used to read data from the client */
        BufferedReader br =
            new BufferedReader (
                                new InputStreamReader (s.getInputStream()));

        /* used to write data to the client */
        OutputStreamWriter osw =
            new OutputStreamWriter (s.getOutputStream());

        /* read the HTTP request from the client */
        String request = br.readLine();

        String command = null;
        String pathname = null;

        /* parse the HTTP request */
        StringTokenizer st =
            new StringTokenizer (request, " ");

        command = st.nextToken();
        pathname = st.nextToken();

        if (command.equals("GET")) {
            /* if the request is a GET
               try to respond with the file
               the user is requesting */
            serveFile (osw,pathname);
        }
    else {
        /* if the request is a NOT a GET,
           return an error saying this server
           does not implement the requested command */
        osw.write ("HTTP/1.0 501 Not Implemented\n\n");
    }

    /* close the connection to the client */
    osw.close();
}

public void serveFile (OutputStreamWriter osw,
                       String pathname) throws Exception {
    FileReader fr=null;
    int c=-1;
    StringBuffer sb = new StringBuffer();

    /* remove the initial slash at the beginning
       of the pathname in the request */
    if (pathname.charAt(0)=='/')
        pathname=pathname.substring(1);

    /* if there was no filename specified by the
       client, serve the "index.html" file */
    if (pathname.equals(""))
        pathname="index.html";

    /* try to open file specified by pathname */
    try {
        fr = new FileReader (pathname);
        c = fr.read();
    }
    catch (Exception e) {
        /* if the file is not found,return the
           appropriate HTTP response code  */
        osw.write ("HTTP/1.0 404 Not Found\n\n");
        return;
        }

        /* if the requested file can be successfully opened
           and read, then return an OK response code and
           send the contents of the file */
        osw.write ("HTTP/1.0 200 OK\n\n");
        while (c != -1) {
            sb.append((char)c);
            c = fr.read();
        }
        osw.write (sb.toString());
    }

    /* This method is called when the program is run from
       the command line. */
    public static void main (String argv[]) throws Exception {

        /* Create a SimpleWebServer object, and run it */
        SimpleWebServer sws = new SimpleWebServer();
        sws.run();
    }
}
```