Lab 2: Add PUT method Functionality

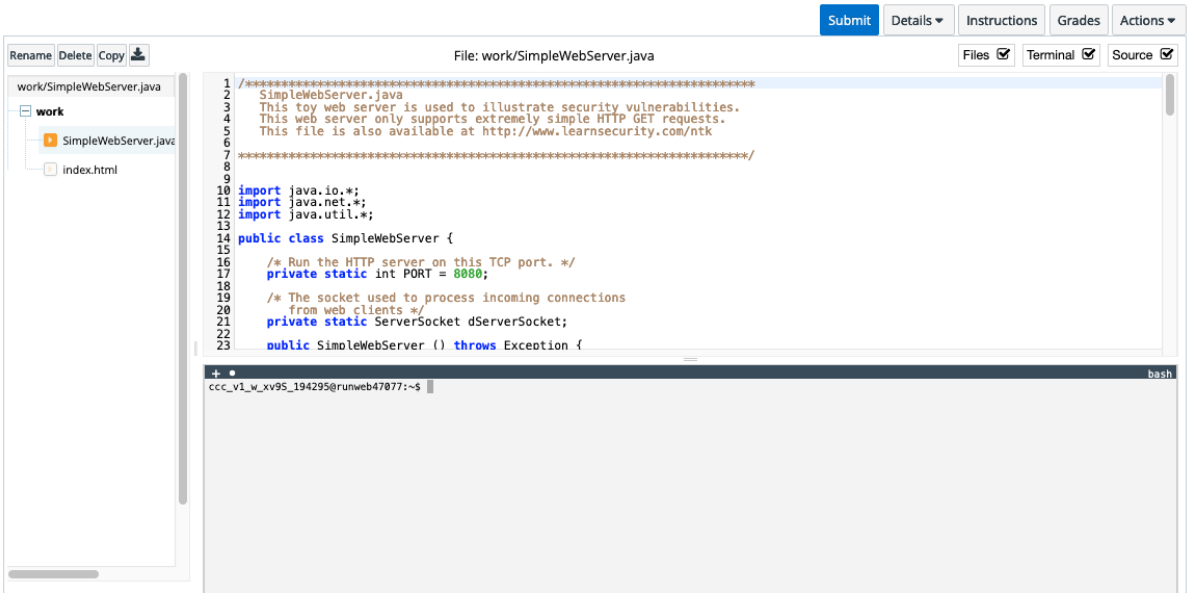
Objective and Set Up:

The goal of this assignment is to make SimpleWebServer support the upload service. With the new version of the web server, a user should be able to upload a file to the server. You will modify a java file with additional code, then you will run the web server (as you did in Lab 1), and send HTTP PUT requests to upload a file.

NOTES:

- *In this exercise, you need to change SimpleWebServer.java by adding new code. The exercise is completed by sending HTTP PUT to the web server.*
- *This assignment is optional and is not graded, but it is highly recommended you complete it.*
- *This guide references code from the book "Foundations of Security" by Daswani, Kern, and Kesavan. The excerpt of the book is provided in the instructions – you don't need to purchase the book.*
- *You can find a few hints below, but we encourage you to try the exercise yourself.*
- ***TIP: Once you are done, save your file. It will be helpful for Lab 3.***

There should be two files in your work area on the left panel to start with. The third file will be compiled during the lab exercise. You will see a preview of the files on the top right, and the terminal window on the bottom right.



This is a description of each file in your work area:

SimpleWebServer.java: The web server file that needs to support the upload service. You will update this file.

SimpleWebServer.class: This is the class file compiled from SimpleWebServer.java (it will appear after the .java file is compiled).

index.html: A simple html file that displays a Hello World message.

Add PUT Method Functionality Lab

Instructions:

1. Add a new method to process uploading

Copy the code of the method "storeFile" (page 78 of the book "*Foundations of Security*", the book excerpt needed is copied below) to SimpleWebServer.java, add it below the "serveFile" method. This won't be done in the terminal, but directly in the code.

```
public void storeFile(BufferedReader br,
                      OutputStreamWriter osw,
                      String pathname) throws Exception {
        FileWriter fw = null;
        try {
                fw = new FileWriter (pathname);
                String s = br.readLine();
                while (s != null) {
                        fw.write (s);
                        s = br.readLine();
                }
                fw.close();
                osw.write ("HTTP/1.0 201 Created");
        }
        catch (Exception e) {
                osw.write ("HTTP/1.0 500 Internal Server Error");
        }
}

public void logEntry(String filename,String record) {
        FileWriter fw = new FileWriter (filename, true);
        fw.write (getTimestamp() + " " + record);
        fw.close();
}

public String getTimestamp() {
        return (new Date()).toString();
}
```

2. Add code to handle PUT method

Modify the method "processRequest" to make it handle the PUT method and invoke the method "storeFile". This will again be done directly in the code, not in the terminal.

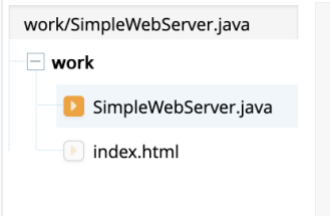
You may choose to modify the file in the Vocareum lab area, or you can download it to your computer. After you finish, save (or upload) the modified SimpleWebServer.java to Vocareum. To upload a file, you will need to select the 'work' folder in the File pane on the left, then click the 'Upload' button above the File pane. A dialog appears that allows you to navigate to the location of your SimpleWebServer.java. Select the file and click OK.

3. Compile SimpleWebServer.java

Compile the modified SimpleWebServer.java in a terminal by typing

javac SimpleWebServer.java

Fix any bugs and compile the file again. If your SimpleWebServer is compiled successfully you will see a new file "SimpleWebServer.class" and the cursor in the terminal will be blinking. If you don't see "SimpleWebServer.class" but your code executed without an error, try hitting the orange button besides "SimpleWebServer.java", which will open your "SimpleWebServer.class" file.



4. Start SimpleWebServer

Start SimpleWebServer as you did in Lab 1.

5. Upload files

Upload a file to SimpleWebServer to test it. Here are two ways to send a PUT request to your SimpleWebServer**:**

a) Curl command line

You can use the Curl command to send a PUT request in a terminal (open a new terminal by clicking the '+' symbol on the top left corner). Remember that your main terminal is running the web server.

Example:

curl -X PUT -H “Content-Type: application/x-www-form-urlencoded” -d “your input here” http://localhost:$port/test.html

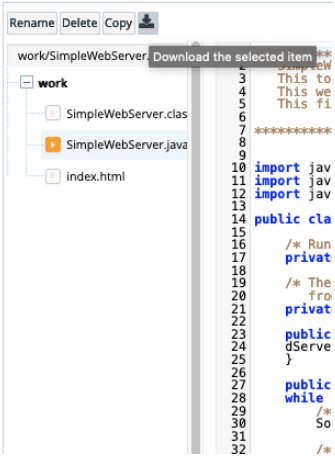
https://reqbin.com/req/c-d4os3720/curl-put-example

b) Postman tool

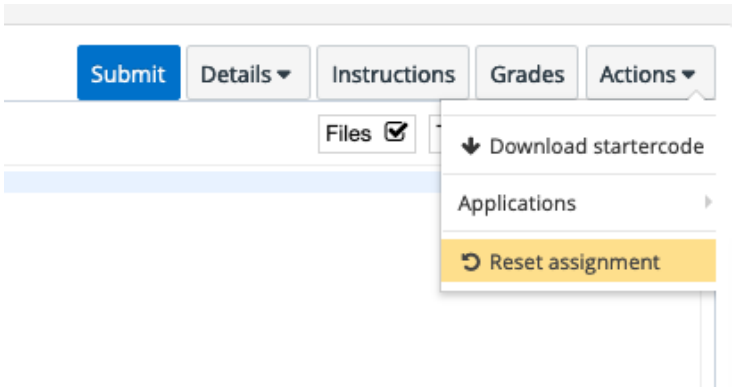
Alternatively, you can find Postman tool and use it to send a PUT request to upload the file to your SimpleWebServer. There are plenty of tutorials on the web you can look up.

6. Save your work

Once you are done, save your “SimpleWebServer.java” file. It will be helpful for Lab 3. Select the file and pick “Download the selected item”.



At any point, if you would like to reset the assignment and start over, find "Actions" and "Reset assignment" on the upper navigation pane



Hints:

1. Copy the method storeFile from page 78 of the book (the excerpt is included above as an image) and paste the method storeFile below the serveFile method in the SimpleWebServer.java.
2. In the method **processRequest** above the else statement at line 69, add a new "else if" statement to handle the PUT command. Inside the new "else if" block add a method call to the storeFile method:

   } else if (command.equals("???")) {

   storeFile(br, osw, pathname); }

3. Compile, run the web server, and curl PUT in a new terminal.

   You can start the web server the same way as you did in Lab 1:

   port=`voc_get_proxied_server_port`

   java SimpleWebServer $port