

Assignment 2: Web Server

1DV701 Computer Networks
- Introduction-



Ruth Dirnfeld & Alexandra Bjäremo
rd222dv@student.lnu.se
ab223zm@student.lnu.se

Faculty of Technology
Department of Computer Science

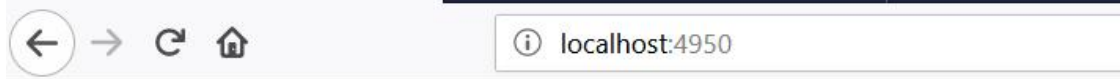
Problem 1

For this part of the assignment we have implemented a web server that contains the following classes: Server (which is used to run the web server), Client, HTTPRequest, HTTPResponse, HTTPException and StatusCode.

The server serves .html pages and .png images from a local directory for corresponding GET requests. The server response contains the contents of the requested file as well as corresponding headers, etc.

To test our server we have prepared a few small .html files / .png images / .jpg images / .gif Gifs. Then organized these files in a hierarchy of directories with some files: "inner", "inner/images/patrick.png" and "inner/images/hello.gif" or "inner/whoWins.gif".

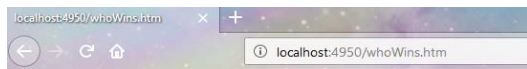
Resulting Screenshots for: "localhost:4950" || "http://localhost:4950/index.html":



It works



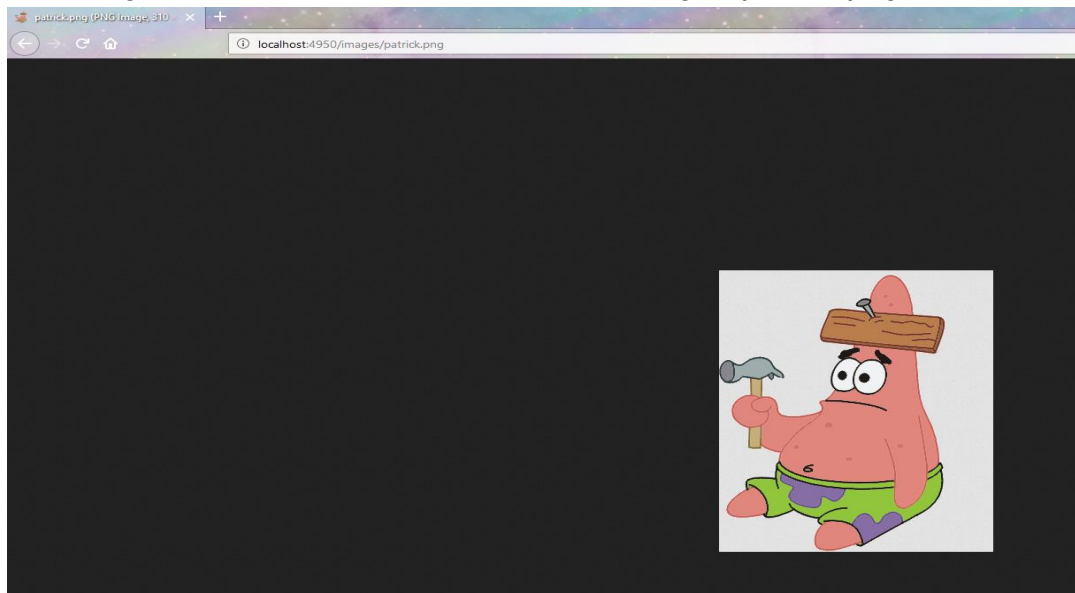
Resulting Screenshots for: "localhost:4950/whoWins.htm":



Who wins?



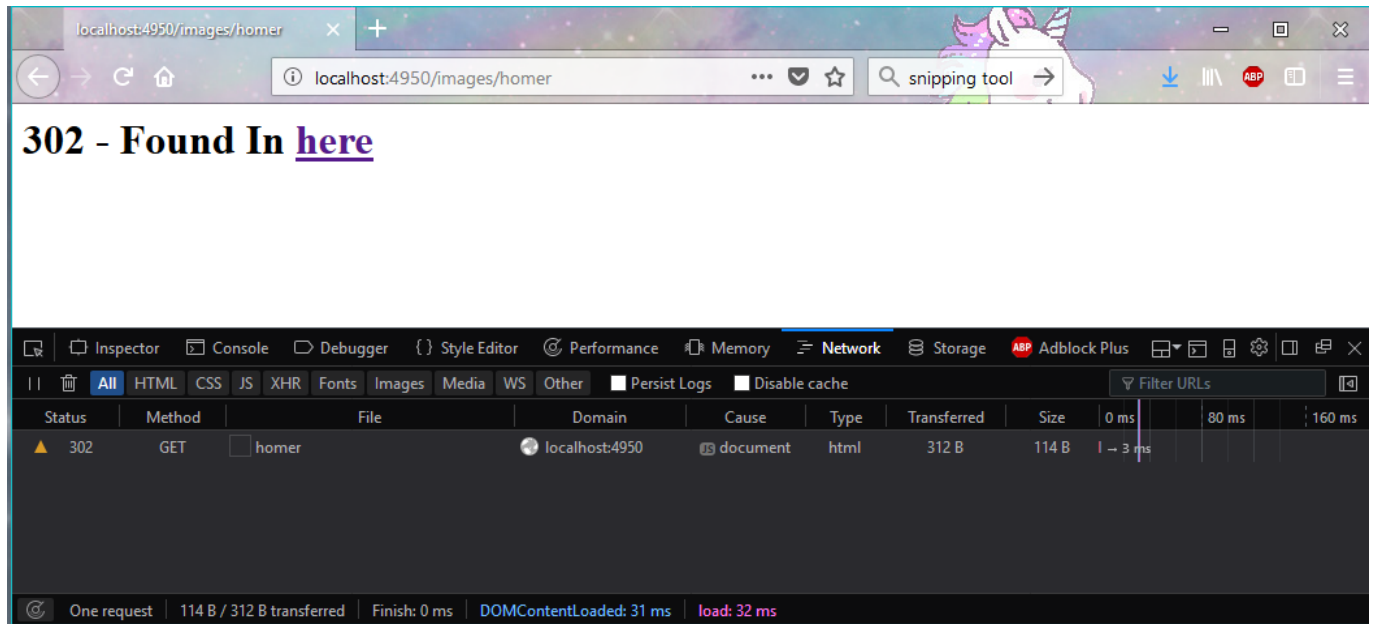
Resulting Screenshots for: "localhost:4950/images/patrick.png":



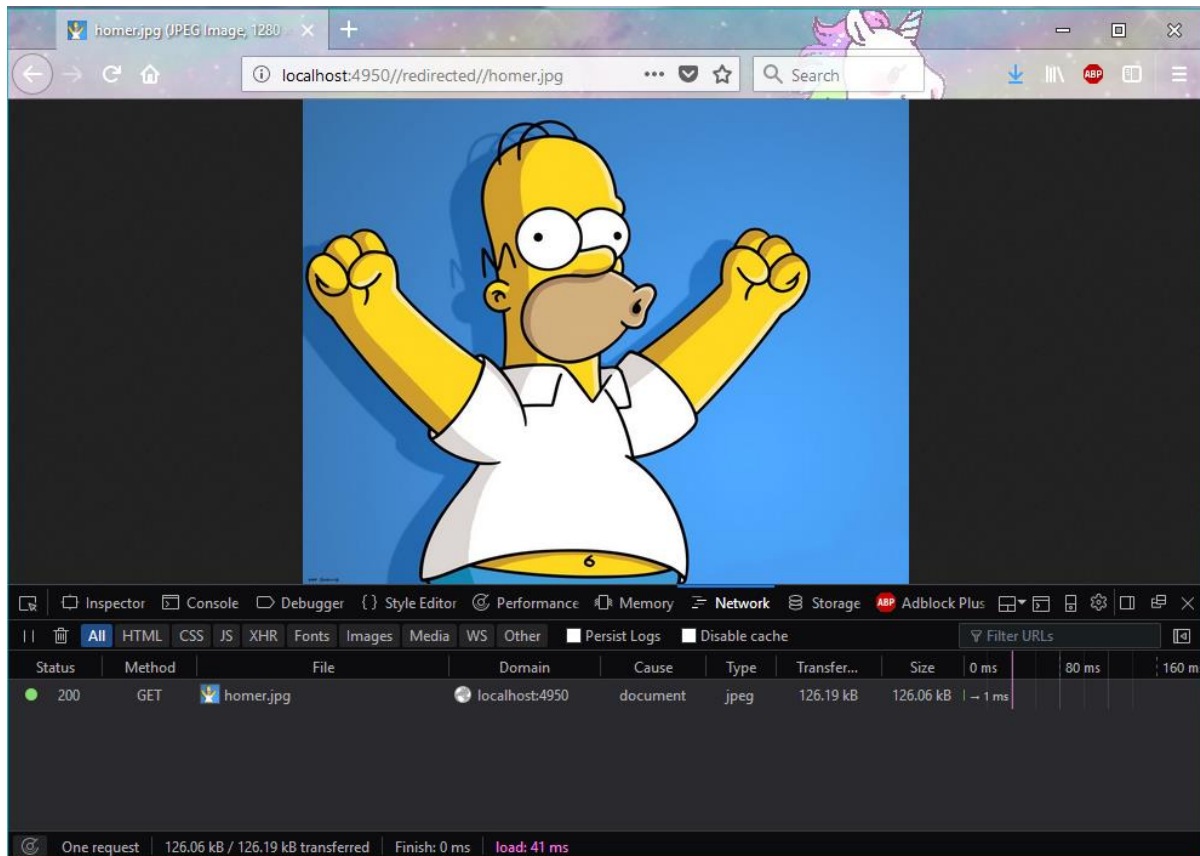
Problem 2

For this part of the assignment our server produces "200 OK". The new task was to add code support for the following 4 server responses: 302, 403, 404, 500. We made up our own access policy for the web server to demonstrate the use of 403. 302 Found, uses the location header and redirects the GET request. Including corresponding web browser screenshots:

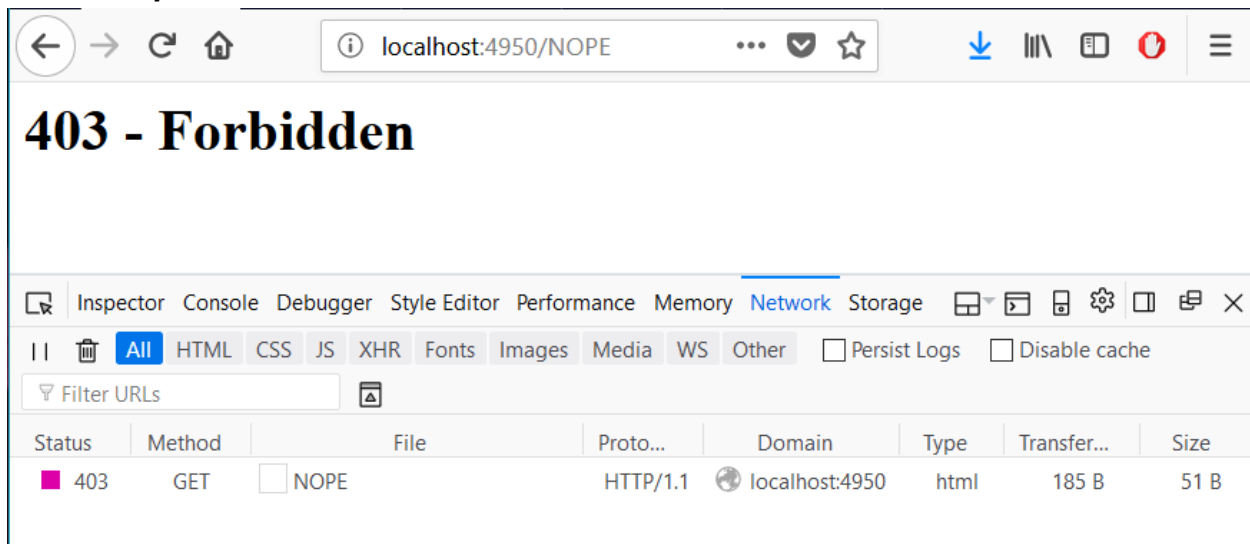
- **Response 302:**



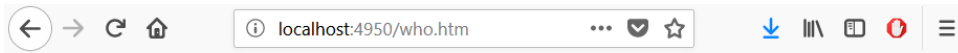
Then clicking the here button, the web server will redirect to the correct address and the following will be displayed:



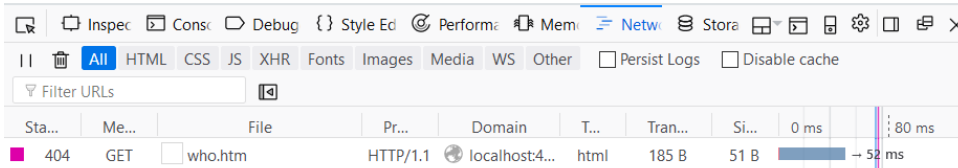
- **Response 403:**



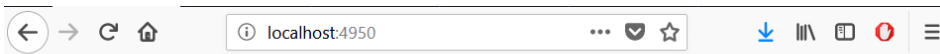
- **Response 404:**



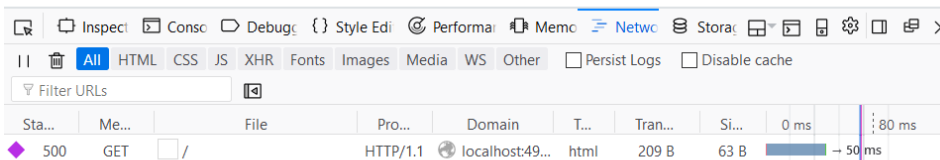
404 - Not Found



- **Response 500:**



500 - Internal Server Error



Regarding the rules we made, a 200 response will bring the client to the start page of the web server, which contains the text *"It works"*, along with a .gif file.

We have assumed that a 302 response will be given when the client tries to access a certain file (*homer.jpg*) from the last known address (*images* directory) and that file has been moved (temporarily) to another location (in our case, the *homer.jpg* is located in the *redirected* directory). We also assumed that such a response will also provide a link to the correct location of the file (*"here"* button).

Our web server responds with 403 when the client tries to access a forbidden address (*"/NOPE"*).

To produce a 404 the client would have to either misspelled something in the url or try to access a directory/file that is not provided by the web server.

Finally, the 500 response is displayed when an internal server error takes place. In order to create this response we have implemented in our code an if statement in the `HttpRequest` class, `readHeader(BufferedReader reader)` method, that upon uncommenting will do the job.

Problem 3

For this part of the assignment we have used PuTTY to repeat our requests.

We used the following parameters:

Host Name: *localhost*

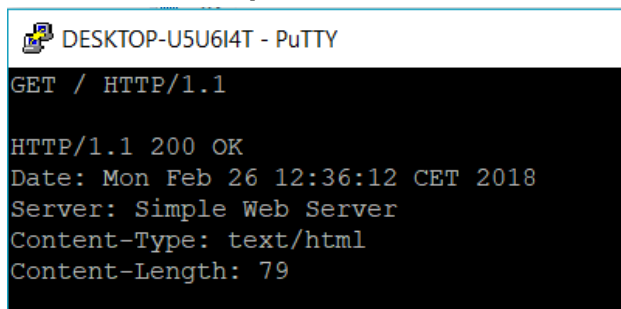
Port: *4950*

Connection Type: *Raw*

Close window on exit: *Never*

The input in PuTTY and the results can be seen in the following screenshots.

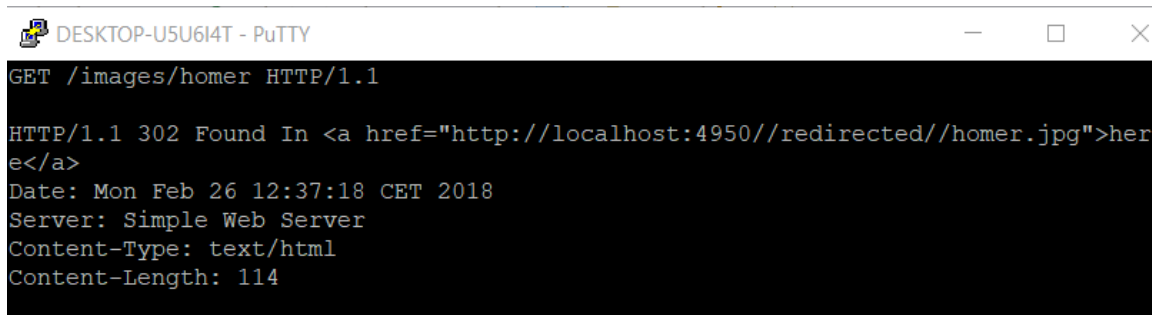
- ***PuTTY response 200:***



```
DESKTOP-U5U6I4T - PuTTY
GET / HTTP/1.1

HTTP/1.1 200 OK
Date: Mon Feb 26 12:36:12 CET 2018
Server: Simple Web Server
Content-Type: text/html
Content-Length: 79
```

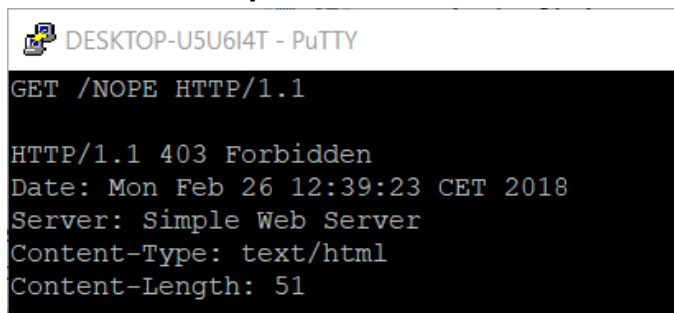
- ***PuTTY response 302:***



```
DESKTOP-U5U6I4T - PuTTY
GET /images/homer HTTP/1.1

HTTP/1.1 302 Found In <a href="http://localhost:4950//redirected//homer.jpg">here</a>
Date: Mon Feb 26 12:37:18 CET 2018
Server: Simple Web Server
Content-Type: text/html
Content-Length: 114
```

- ***PuTTY response 403:***



```
DESKTOP-U5U6I4T - PuTTY
GET /NOPE HTTP/1.1

HTTP/1.1 403 Forbidden
Date: Mon Feb 26 12:39:23 CET 2018
Server: Simple Web Server
Content-Type: text/html
Content-Length: 51
```


- ***PuTTY response 404:***



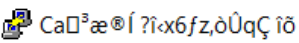
```
GET /indx.html HTTP/1.1
HTTP/1.1 404 Not Found
Date: Mon Feb 26 12:40:14 CET 2018
Server: Simple Web Server
Content-Type: text/html
Content-Length: 51
```

- ***PuTTY response 500:***



```
HTTP/1.1 500 Internal Server Error
Date: Mon Feb 26 12:40:52 CET 2018
Server: Simple Web Server
Content-Type: text/html
Content-Length: 63
```

PutTTY did not work with images, so upon trying to access the patrick.png file with the command: GET /images/patrick.png HTTP/1.1 , the following was produced:

[illegible]

The reason for that is the fact that PuTTY is a terminal emulator that supports html/text based types. Where, an image is not text based but a collection of data displayed on a display grid. Each pixel is defined by numbers expressed in bits with each bit containing two colors. Furthermore, most image files (including the patrick.png) are pallet based, with pallets of 24 bit RGB or 32-bit RGBA colors. Meaning that our telnet client will not be able to display an image.