Networking and HTTP

Martin Pola

1 Find your IP address

To be able to access a web server, or really any network service, you will need to know which IP address you have. Open a terminal and type `ipconfig` if you run Windows or `ifconfig` if you run a Unix-like system. Decipher output (might be long!) and see if you can find the IP address of your computer. Compare with a neighbor! Do your two addresses look similar? Which parts differs?

2 Run the web server

Download and run the two following two classes that together form the basis of a simple HTTP server. Try to understand what's happening on a conceptual level (not necessarily what every single line does) and run the web server.

- HttpServer.java
- ClientConnection.java

Can you access your web server via your IP address? Note that the port is 8080, which is not default for HTTP, so you'd have to add that in your URL.
`http://192.168....:8080/`

When your neighbor runs their web server, can you also access their web server via their IP address?

3 Serve files from your file system

Try to open a few different files in the web browser by visiting URLs like the following (with your own IP address).

- `http://192.168....:8080/travel_diary.htm`
- `http://192.168....:8080/family_friends.htm`
- `http://192.168....:8080/index.htm`

You will hopefully see that the web server currently always gives you the same, static response. Let's fix that, by allowing visitors to retrieve HTML files directly from our file system.

1. Look in the Java console window and see how the URLs above translate to `request from client` lines in the output (line 35 in `ClientConnection.java`).

2. Add code in the `parseRequest` method to get only the file name e.g. `travel_diary.htm` from a request. Save the result in the `requestedFile` instance variable.
3. In the `fetchResponseData` method, change the value of `responseBody` to contain the contents of the file specified by the `requestedFile` variable.
4. Create a few sample HTML files, such as `family_friends.htm`, and store them in your project.

Now, if you restart your web server, can you access the `family_friends.htm` web page in your browser, through your web server?

Tip! If you get exceptions about files not being found, even when they exist, make sure they're placed in the right directory. Use your knowledge from the file management presentation to let the web server print out its working directory in the console, and make sure your HTML files are placed there.

4 Page not found

If a client (web browser) requests a file that cannot be found on the file system, the web server should respond with `404 Not Found` rather than the default `200 OK`.

Modify the `respondToClient` method to respond with the appropriate status code (`404 Not Found`) when a requested file could not be found. You would likely also have to modify `fetchResponseData`, perhaps by surrounding the code for reading a file by a `try` block.

Tip! Let the value of `responseBody` be either the contents of the requested file or `null`, depending on whether the file exists or not, respectively. Then, you can have an `if` statement in the `respondToClient` method to determine which kind of response your server should send.

5 Index file

If a user visits your website for the first time, they would probably just enter something like
`http://192.168....:8080/`
i.e. they would not specify which file to retrieve.

This is a special case. Add code so that all requests to `/` (with no file name specified) will resolve to a file named `index.htm`. When you are done, attempts to visit the following two URLs should result in you seeing the exact same page.

- `http://192.168....:8080/`
- `http://192.168....:8080/index.htm`

6 Serve other file types (stretch task)

Nowadays, the web does not only contain (HTML) text.

Modify your web server to be able to serve images as well. You would have to change the `Content-Type` header in the response, perhaps based on the extensions of the requested file (e.g. `.htm` or `.jpg`). For JPEG images, the value of `Content-Type` is `image/jpeg`.

Tip! You can find the content type of most files by searching on the web for a file's *MIME type*.