Http client

[Ugeopgave 1] *

Mirza Hasanbasic

ABSTRACT

In this assignemt we will be looking at how to implement a simple HTTP client, that will have a subset of the entire HTTP protocol. The HTTP client will still be able to communicate with servers

1. INTRODUCTION

I have written the HTTP client in python and you are able to run in from a terminal. In order to run it you need to write two commands i.e. `www.randomwebsite.com filename`. The first part is the website you wish the client to send a `GET` request to and the second is the name the savefile should have.

The libraries I use are socket and sys in the HTTP client and in my parser I use itertools and unittest.

The function that I use to split my url is `stringToUrl(myurl)`, where the variable name `myurl` the string I wish to parse. I start with splitting after `'//'` and then I check wether the user have given me `HTTPS` as input. If this is true, then I raise an error and tell the user that `HTTPS` is not implemented and that they should use `HTTP` instead. After this I check the length of my list. I the length is greater than 1 then the user have written `http://www.example.com` but if the length is 1 then the user have written `www.example.com` and if this is true, then I add the `HTTP` for the user and insert both the scheme and netloc into a list.[1]

This HTTP client can do redirect if you should get an error code 30x. It will use recursion to find the new location.

If you should be able to get a status code 200, then the client will write to the filename the `GET` request.

APPENDIX

A. HEADINGS IN APPENDICES

The rules about hierarchical headings discussed above for the body of the article are different in the appendices. In the **appendix** environment, the command

*

[1]The scheme is 'http' where as the netloc is 'www.example.com'

© 2016 ACM. ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235