Introduction to Web Science

Assignment 4

Prof. Dr. Steffen Staab
staab@uni-koblenz.de

René Pickhardt
rpickhardt@uni-koblenz.de

Korok Sengupta
koroksengupta@uni-koblenz.de

Institute of Web Science and Technologies
Department of Computer Science
University of Koblenz-Landau

Submission until: November 23, 2016, 10:00 a.m.
Tutorial on: November 25, 2016, 12:00 p.m.

In this assignment we cover two topics: 1) **HTTP** & 2) **Web Content**

For all the assignment questions that require you to write code, make sure to include the code in the answer sheet, along with a separate python file. Where screen shots are required, please add them in the answers directly and not as separate files.

Team Name: Bravo

1 Implementing a simplified HTTP GET Request (15 Points)

The goal of this exercise is to review the hyptertext transfer protocol and gain a better understanding of how it works.

Your task is to use the python programming language to create an HTTP client (http-client.py) that takes a URL as a command line argument and is able to download an arbitrary file from the World Wide Web and store it on your hard drive (in the same directory as your python code is running). The program should also print out the complete HTTP header of the response and store the header in a seperated file.

Your programm should only use the socket library so that you can open a TCP socket and and sys library to do command line parsing. You can either use urlparse lib or your code from assignment 3 in order to process the url which should be retrieved.

Your programm should be able to sucessfully download at least the following files:

1. http://west.uni-koblenz.de/en/studying/courses/ws1617/introduction-to-web-science
2. http://west.uni-koblenz.de/sites/default/files/styles/personen_bild/public/_IMG0076-Bearbeitet_03.jpg

Use of libraries like `httplib`, `urllib`, etc are not allowed in this task.

1.1 Hints:

There will be quite some challenges in order to finnish the task

- Your program only has to be able to process HTTP-responses with status 200 OK.
- Make sure you receive the full response from your TCP socket. (create a function handling this task)
- Sperated the HTTP header from the body (again create a function to do this)
- If a binary file is requested make sure it is not stored in a corrupted way

1.2 Example

```
1: python httpclient.py http://west.uni-koblenz.de/index.php
2:
3: HTTP/1.1 200 OK
4: Date: Wed, 16 Nov 2016 13:19:19 GMT
5: Server: Apache/2.4.7 (Ubuntu)
6: X-Powered-By: PHP/5.5.9-1ubuntu4.20
7: X-Drupal-Cache: HIT
8: Etag: "1479302344-0"
9: Content-Language: de
```

```
10: X-Frame-Options: SAMEORIGIN
11: X-UA-Compatible: IE=edge,chrome=1
12: X-Generator: Drupal 7 (http://drupal.org)
13: Link: <http://west.uni-koblenz.de/de>; rel="canonical",<http://west.uni-koblenz.de
14: Cache-Control: public, max-age=0
15: Last-Modified: Wed, 16 Nov 2016 13:19:04 GMT
16: Expires: Sun, 19 Nov 1978 05:00:00 GMT
17: Vary: Cookie,Accept-Encoding
18: Connection: close
19: Content-Type: text/html; charset=utf-8
```

2 Python

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Sun Nov 20 11:05:14 2016
@author: kandy
"""

from assignment4_bravo_functions import *

try :
    socClient = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

    urlInput = input('Input the URL : ')
    #urlInput = 'http://west.uni-koblenz.de/en/mws/dates/index.html'
    #urlInput = 'https://www.uni-koblenz-landau.de/de/koblenz/index.html'
    #urlInput = 'http://west.uni-koblenz.de/en/studying/courses/ws1617/introductio
    #urlInput = 'http://blog.ifimbschool.com/wp-content/uploads/2014/12/Hindi-Quo
    #urlInput = 'http://www.w3schools.com/tags/tag_link.asp'
    #urlInput = 'http://west.uni-koblenz.de/sites/default/files/styles/front-slid
    #urlInput = 'http://west.uni-koblenz.de/'

    try:
        socClient.connect((urlDomain, 80))
        downloadResource(socClient, urlInput)
        socClient.close()
    except socket.error as e:
       print('Connection Invalid. Input proper URL !!!',e)
    except:
        print('Connection Invalid. Input proper URL !!!')

finally:
    socClient = None


#print(urlInput)
```

The header will be printed and stored in index.php.header. The retrieved html document will be stored in index.php

3 Functions

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Mon Nov 21 18:29:33 2016
@author: kandy
"""

import socket
import sys
import urllib

def getResource(socClient, urlInput):

    try:
        urlObj = urllib.parse.urlparse(urlInput)
        print(urlObj)
    except:
        print('Invalid URL')
        return None

    urlScheme = urlObj[0] #http
    urlDomain = urlObj[1] #full domain name
    urlPath = urlObj[2]
    resourceName = (urlPath[urlPath.rfind("/")+1:])



    # Form the http GET request. Two \r\n at the end is very important
    httpRequest = 'GET ' + urlPath + ' ' + urlScheme+'/1.0\r\n'
    httpRequest += 'Host: '+urlDomain+ '\r\n\r\n'

    # Send and Receive the data. Keep the data as bytes
    # Header needs to be extracted with UTF-8 encoding and if the resource conten
    # then the body can be encoded to UTF and saved in file
    # Otherwise it should be retained the same and saved
    socClient.send(httpRequest.encode('utf-8'))
    temp = socClient.recv(4096)
    data = bytearray()
    while (temp != b''):
        #print(temp)
        # Tried a lot to append in bulk
        # RIght now appending char by char
        for char in temp :
            data.append(char)
        temp = socClient.recv(4096)
    return [resourceName, data]

```

```
48: def extractHeaderAndResource(data):
49:     #The header and resource will be separated  by two \r\n's
50:     try:
51:         splitData = data.split(b'\r\n\r\n')
52:         #print(splitData)
53:     except:
54:         splitData = [None, None]
55:     return splitData
56:
57: def checkForRequest200(header):
58:     splittedHeader = header.split(b' ')
59:     print('Inside check for request 200')
60:     #print(splittedHeader)
61:     if(splittedHeader):
62:         return splittedHeader[1] == b'200'
63:     return False
64:
65: def saveResource(data,iname):
66:     fopen = open(iname,'wb')
67:     fopen.write(data)
68:     fopen.flush()
69:     fopen.close()
70:
71: def downloadResource(socClient, urlInput):
72:     [name, data] = getResource(socClient, urlInput)
73:     if(name == ''):
74:         name = 'index.html'
75:     if (data):
76:         try:
77:             header,resource = extractHeaderAndResource(data)
78:             if (header):
79:                 if(checkForRequest200(header)) :
80:                     saveResource(header,name+'.header')
81:                     saveResource(resource,name)
82:                     print('Resource is downloaded successfully !!!')
83:                 else:
84:                     print('Invalid Http response !!!!')
85:             else:
86:                 print('Header and Resource extraction is invalid')
87:         except:
88: print('Error in downloading the resource !!!')
```

4 Download Everything (15 Points)

If you have successfully managed to solve the previous exercise you are able to download a web page from any url. Unfortionately in order to successfully render that very webpage the browser might need to download all the included images

In this exercise you should create a python file (downloadEverything.py) which takes two arguments. The first argument should be a name of a locally stored html file. The second argument is the url from which this file was downloaded.

Your program should

1. be able to find a list of urls the images that need to be downloaded for successful rendering the html file.
2. print the list of URLs to the console.
3. call the program from task 1 (or if you couldn't complete task 1 you can call wget or use any python lib to fulfill the http request) to download all the necessary images and store them on your hard drive.

To finnish the task you are allowed to use the 're' library for regualar expressions and everything that you have been allowed to use in task 1.

4.1 Hints

1. If you couldn't finnish the last task you can simulate the relevant behavior by using the program wget which is available in almost any UNIX shell.
2. Some files mentioned in the html file might use relative or absolut paths and not fully qualified urls. Those should be fixed to the correct full urls.
3. In case you run problems with constructing urls from relative or absult file paths you can always check with your web browser how the url is dereferenced.

5 DownloadEverything

```
# -*- coding: utf-8 -*-
"""
Created on Sun Nov 20 22:05:14 2016
@author: Shriharsh Ambhore
@author: Kandhasamy Rajasekaran
@author: Daniel Akbari
"""

#from assignment4_bravo_Q1_http import downloadResource
import urllib

import os
import socket
import re
import time

import sys
import urllib
from assignment4_bravo_functions import *


def extractImageUrl(fileLocation):
    urllist=[]
    pattern=re.compile(r'<img [^>]*src="([^"]+)') # regex for extracting the valu
    hrefPattern=re.compile(r'<link [^>]*href="([^"]+)')
    with  open(fileLocation) as f:
        con=f.read()
        urllist= pattern.findall(con)
        urlhref=(hrefPattern.findall(con))

    return urllist+urlhref

fileLoc=input("Enter the html file location of the downloaded file:")
#fileLoc = 'index.html'
#fileLoc="C:\\Users\\ShreeH\\resource"
inputurl=input("Enter the url from which the file was downloaded:")
#inputurl='http://west.uni-koblenz.de'

urlObj = urllib.parse.urlparse(inputurl)
urlScheme = urlObj[0] #http
urlDomain = urlObj[1] #full domain name
urlPath = urlObj[2]


listOfURL=extractImageUrl(fileLoc)

#filters out list .css and non image files
```

```
imageList = list(filter(lambda x: x.find('.ico')!=-1 or x.find('.png')!=-1 or x.f

print("List of URLs",imageList)



# iterate over the url to download the image
for url in imageList:

    try:
        socClient = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        socClient.connect((urlDomain, 80))
        path=urllib.parse.urlparse(url).path
        imageName=(path[path.rfind("/")+1:])
        tempUrl = url
        if url.find(inputurl)== -1:
            #append the input url to construct the absolute path for the image
            tempUrl=inputurl+url

        downloadResource(socClient,tempUrl)
        socClient.close()
    except socket.error as e:
       print('Connection Invalid. Input proper URL !!!',e)
    except:
        print('Connection Invalid. Input proper URL !!!')

url=None
data=None
```

Important Notes

Submission

- Solutions have to be checked into the github repository. Use the directory name `groupname/assignment4/` in your group's repository.
- The name of the group and the names of all participating students must be listed on each submission.
- Solution format: all solutions as *one* PDF document. Programming code has to be submitted as Python code to the github repository. Upload *all* `.py` files of your program! Use `UTF-8` as the file encoding. *Other encodings will not be taken into account!*
- Check that your code compiles without errors.
- Make sure your code is formatted to be easy to read.
  - Make sure you code has consistent indentation.
  - Make sure you comment and document your code adequately in English.
  - Choose consistent and intuitive names for your identifiers.
- Do *not* use any accents, spaces or special characters in your filenames.

Acknowledgment

This latex template was created by Lukas Schmelzeisen for the tutorials of "Web Information Retrieval".

LaTeX

Currently the code can only be build using LuaLaTeX, so make sure you have that installed. If on Overleaf, there's an error, go to settings and change the LaTeXengine to `LuaLaTeX`.