

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: mike

Team members: Slobodan Kocevski, Shohel Ahamad, Anish Girijashivaraj

1 Implementing a simplified HTTP GET Request (15 Points)

The goal of this exercise is to review the hypertext 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 (`httpclient.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 separated 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/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
```

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

Answer:

```
1: import socket
2: from urllib.parse import urlparse
3: import sys
4: import time
5:
6: #function timeout for getting the entire data from the file
7: def recv_timeout(the_socket, timeout=2):
8:     # make socket non blocking
9:     the_socket.setblocking(0)
10:    # total data partwise in an string
11:    totalData=""
12:    data = '';
13:    # beginning time
14:    begin = time.time()
15:    while 1:
16:        # if we got some data, then break after timeout
17:        if totalData and time.time() - begin > timeout:
18:            break
19:
20:        # if you got no data at all, wait a little longer, twice the timeout
21:        elif time.time() - begin > timeout * 2:
22:            break
23:
24:        # recv something
25:        try:
26:
27:            data = the_socket.recv(8192)
28:            if data:
29:                totalData = totalData + data.decode()
30:                # change the beginning time for measurement
31:                begin = time.time()
32:            else:
```

```
33:             # sleep for sometime to indicate a gap
34:             time.sleep(0.1)
35:         except:
36:             pass
37:     if(totalData.find("200 OK") != -1):
38:         separete(totalData)
39:     else:
40:         print("The status of the HTTP response was other then 200 OK. \n Could not")
41:         # return 0 for successfully executed function
42:         return 0
43:
44:
45: def separete(list_to_Separate):
46:
47:     splitted = str(list_to_Separate).split('\r\n\r\n')
48:     print(splitted[0])
49:     #writing data from the HTTP header
50:     iphph = open("index.php.header", "w")
51:     iphph.write(splitted[0])
52:     iphph.close()
53:     #writing data from the HTTP body
54:     iphp = open("index.php", "w")
55:     iphp.write(splitted[1])
56:     iphp.close()
57:     return 0
58:
59:
60: url=input("Enter URL:")
61: o = urlparse(url)
62:
63: host=o.netloc
64: path=o.path
65:
66: port = 80 # web
67:
68: print('\n # Creating socket')
69: try:
70:     s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
71: except socket.error:
72:     print('Failed to create socket')
73:     sys.exit()
74:
75: print('# Getting remote IP address')
76: try:
77:     remote_ip = socket.gethostbyname(host)
78: except socket.gaierror:
79:     print('Hostname could not be resolved. Exiting')
80:     sys.exit()
81:
```

```
82: # Connect to remote server
83: print('# Connecting to server, ' + host + ' (' + remote_ip + ')')
84: s.connect((remote_ip, port))
85:
86: # Send data to remote server
87: print('# Sending data to server')
88: request = "GET " + path + " HTTP/1.0\r\n\r\n"
89:
90: try:
91:     s.sendall(request.encode('utf'))
92: except socket.error:
93:     print
94:     'Send failed'
95:     sys.exit()
96:
97: # Receive data
98: print('# Receive data from server \n ')
99: recv_timeout(s)
100:
101: s.close()
102: print ("\ndone")
```



```
Run httpclient
C:\Users\Sloboda\AppData\Local\Programs\Python\Python36\pythonw.exe C:/Users/Sloboda/httpclient.py
Enter URL: http://west.uni-koblenz.de/en/studying/courses/vs1617/introduction-to-web-science

# Creating socket
# Getting Remote IP address
# Connecting to server, west.uni-koblenz.de (141.26.208.71)
# Sending data to server
# Receive data from server

HTTP/1.1 200 OK
Date: Tue, 22 Nov 2016 23:34:48 GMT
Server: Apache/2.4.7 (Ubuntu)
X-Powered-By: PHP/5.5.9-1ubuntu4.20
X-Drupal-Cache: HIT
Etag: "1479855724-0"
Content-Language: en
X-Frame-Options: SAMEORIGIN
X-UA-Compatible: IE=edge,chrome=1
X-Generator: Drupal 7 (http://drupal.org)
Link: <http://west.uni-koblenz.de/en/studying/courses/vs1617/introduction-to-web-science>; rel="canonical", <http://west.uni-koblenz.de/en/node/4959>; rel="shortlink"
Cache-Control: public, max-age=0
Last-Modified: Tue, 22 Nov 2016 23:02:04 GMT
Expires: Sun, 19 Nov 1978 05:00:00 GMT
Vary: Cookie, Accept-Encoding
Connection: close
Content-Type: text/html; charset=utf-8

done

Process finished with exit code 0
```

2 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. Unfortunately 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 finish the task you are allowed to use the 're' library for regular expressions and everything that you have been allowed to use in task 1.

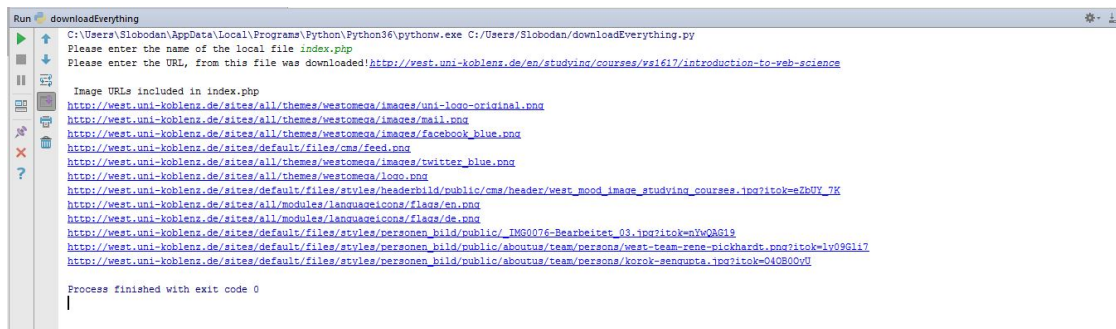
2.1 Hints

1. If you couldn't finish 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 absolute 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 absolute file paths you can always check with your web browser how the url is dereferenced.

Answer:

```
1: import wget
2: import re
3: from urllib.parse import urlparse
4:
5:
6: def download_ALL(file, url):
7:     m = file.read()
8:     o = urlparse(url)
9:     host = o.netloc
10:    protocol = o.scheme
11:
```

```
12:     #defininf regexp for img tags
13:     link = r'<img[^>]*\ssrc="(.*?)" '
14:     imgs = re.findall(link, str(m))
15:
16:     i=0 #printing and download the images
17:     while(i<len(imgs)):
18:         img = imgs[i]
19:         if(img.find(protocol + "://" + host)==-1):
20:             img=protocol + "://" + host + imgs[i]
21:         print(img)
22:         wget.download(img)
23:         i=i+1
24:
25:
26: f = input("Please enter the name of the local file ")
27: u = input("Please enter the URL, from this file was downloaded!")
28: print("\n Image URLs included in " + f )
29: f1=open(f, "r")
30: download_ALL(f1, u)
```



```
Run C:\Users\Slobodan\AppData\Local\Programs\Python\Python36\pythonw.exe C:/Users/Slobodan/downloadEverything.py
Please enter the name of the local file index.php
Please enter the URL, from this file was downloaded:http://west.uni-koblenz.de/en/studying/courses/rsi617/introduction-to-web-science

Image URLs included in index.php
http://west.uni-koblenz.de/sites/all/themes/weatomega/images/uni-logo-original.png
http://west.uni-koblenz.de/sites/all/themes/weatomega/images/mail.png
http://west.uni-koblenz.de/sites/all/themes/weatomega/images/facebook_blue.png
http://west.uni-koblenz.de/sites/default/files/cms/feed.png
http://west.uni-koblenz.de/sites/all/themes/weatomega/images/twitter_blue.png
http://west.uni-koblenz.de/sites/all/themes/weatomega/logo.png
http://west.uni-koblenz.de/sites/default/files/styles/headerbild/public/cms/header/west_mood_image_studying_courses.jpg?itok=7b07Y_7K
http://west.uni-koblenz.de/sites/all/modules/language/flags/en.png
http://west.uni-koblenz.de/sites/all/modules/language/flags/de.png
http://west.uni-koblenz.de/sites/default/files/styles/personen_bild/public/IMG0076-Bearbeiter_03.jpg?itok=wnYwQ8G18
http://west.uni-koblenz.de/sites/default/files/styles/personen_bild/public/aboutus/team/persona/west-team-rene-pickhardt.png?itok=lv08G1i7
http://west.uni-koblenz.de/sites/default/files/styles/personen_bild/public/aboutus/team/persona/korok-sengupta.jpg?itok=040800yU

Process finished with exit code 0
```

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 \LaTeX engine to LuaLaTeX.