CS421 - Computer Networks
Programming Assignment 1
FileDownloader

Due: 11 November 2016

In this programming assignment, you are asked to implement a program in Java. This program is supposed to download an index file to obtain a list of text file URLs and download some of these files depending on their sizes. For this programming assignment, you are not allowed to use any third party HTTP client libraries, or the HTTP specific core or non-core APIs supplied with the JDK including but not limited to *java.net.HttpURLConnection*. The goal of this assignment is to make you familiar with the internals of the HTTP protocol and using any (third party, core or non-core JDK supplied) API providing any level of abstraction specific to the HTTP protocol is prohibited. In short, you have to implement your program using barely the Socket API of the JDK. If you have any doubt about what to use or not to use, please contact us.

Your program must be a console application (no graphical user interface (GUI) is required) and should be named as *FileDownloader* (i.e., the name of the class that includes the main method should be *FileDownloader*). Your program should run with the

```
java FileDownloader <index_file> [<lower_endpoint>-<upper_endpoint>]
```

command, where `<index_file>` and `<lower_endpoint>-<upper_endpoint>` are the command-line arguments. The details of the command-line arguments are as follows:

- `<index_file>`: [Required] The URL of the index that includes a list of text file URLs.
- `<lower_endpoint>-<upper_endpoint>`: [Optional] If this argument is not given, a file in the index is downloaded if it is found in the index. Otherwise, the bytes between `<lower_endpoint>` and `<upper_endpoint>` inclusively are to be downloaded.

When a user enters the command above, your program will send an HTTP GET request to the server in order to download the index file with URL `<index_file>`. If the index file is not found, the response is a message other than 200 OK. In this case, your program will print an error message to the command-line and exits. If the index file is found, the response is a 200 OK message. When this is the case, your program will print the number of file URLs in the index file and send an HTTP HEAD request for each file URL in the index file.

Requested file is not found If the requested file is not found in the server, the response is a message other than 200 OK. In this case, your program will print a message to the command-line indicating that the file is not found. Then, an HTTP HEAD request is sent for the next file.

Requested file is found If the requested file is found in the server, the response is a 200 OK message which includes the size of the file in bytes. When this is the case, there are three possibilities:

1. If the user does not give a range as a command-line argument, your program should send an HTTP GET message to obtain the content of the file from the 200 OK response.
2. If a range is given as a command-line argument and the size of the file is smaller than `<lower_endpoint>`, the file will not be downloaded and your program will print a message to the command-line indicating that the file is not requested. Then, an HTTP HEAD request is sent for the next file.
3. If a range is given as a command-line argument and the size of the file is not smaller than `<lower_endpoint>`, the range is satisfiable. Then, your program should send an HTTP GET message with the range `<lower_endpoint>-<upper_endpoint>` and obtain a part of the file content from the HTTP 206 Partial Content response.

If your program successfully obtains the file or a part of the file, it saves the content under the directory in which your program runs. The name of the saved file should be the same as the downloaded file and a message indicating that the file is successfully downloaded is printed to the command-line. Then an HTTP HEAD request is sent for the next file.

Assumptions and hints

- Please refer to W3Cs RFC 2616 for details of the HTTP messages.
- You will assume that `<lower_endpoint>` and `<upper_endpoint>` are both non-negative integers and `<lower_endpoint>` is not smaller than `<upper_endpoint>`. Note that there should be a hyphen '-' character between the endpoints.
- You will assume that each line of the index file includes one file URL.
- You will assume that the name of each file in the index is unique.
- Your program will not save the index file to the local folder.
- Your program should print a message to the command-line to inform the user about the status of the files.
- The downloaded file should be saved under the directory containing the source file *FileDownloader.java* and the name of the file should be the same as the name of the downloaded file.
- You may use the following URLs to test your program:
  `www.cs.bilkent.edu.tr/~morhan/cs421/index_1.txt`
  `www.cs.bilkent.edu.tr/~morhan/cs421/index_2.txt`
- Please contact your assistant if you have any doubt about the assignment.

Example

Let `www.foo.com/abc/index.txt` be the URL of the file to be downloaded whose content is given as

```
www.cs.bilkent.edu.tr/file.txt
www.cs.bilkent.edu.tr/folder2/temp.txt
wordpress.org/plugins/about/readme.txt
humanstxt.org/humans.txt
www.cs.bilkent.edu.tr/morhan/cs421/deneme.txt
```

where the first file does not exist in the server and the sizes of the other files are 6000, 4567, 1587, and 9000 bytes, respectively.

Example run 1 Let your program start with the

```
java FileDownloader www.foo.com/abc/index.txt
```

command. Then all files except the first one in the index file are downloaded. After the connection is terminated, the command-line of the client may be as follows:

```
Command-line:
URL of the index file: www.foo.com/abc/index.txt
No range is given
Index file is downloaded
There are 5 files in the index
1. www.cs.bilkent.edu.tr/file.txt is not found
2. www.cs.bilkent.edu.tr/folder2/temp.txt (size = 6000) is downloaded
3. wordpress.org/plugins/about/readme.txt (size = 4567) is downloaded
4. humanstxt.org/humans.txt (size = 1587) is downloaded
5. www.cs.bilkent.edu.tr/morhan/cs421/deneme.txt (size = 9000) is downloaded
```

Example run 2 Let your program start with the

```
java FileDownloader www.foo.com/abc/index.txt 0-999
```

command. Then the first 1000 bytes of all files except the first one in the index file `www.foo.com/abc/index.txt` are downloaded. After the connection is terminated, the command-line of the client may be as follows:

```
Command-line:
URL of the index file: www.foo.com/abc/index.txt
Lower endpoint = 0
Upper endpoint = 999
Index file is downloaded
There are 5 files in the index
1. www.cs.bilkent.edu.tr/file.txt is not found
2. www.cs.bilkent.edu.tr/folder2/temp.txt (range = 0-999) is downloaded
3. wordpress.org/plugins/about/readme.txt (range = 0-999) is downloaded
4. humanstxt.org/humans.txt (range = 0-999) is downloaded
5. www.cs.bilkent.edu.tr/morhan/cs421/deneme.txt (range = 0-999) is downloaded
```

Example run 3 Let your program start with the

```
java FileDownloader www.foo.com/abc/index.txt 1587-6999
```

command. The last byte of the fourth file is 1586, so it is not requested. Hence, the bytes in the range `1587-6999` are requested for the second, third, and fifth files. Since the second and third files do not include 7000 bytes, the response includes the bytes in the ranges `1587-5999` and `1587-4566`, respectively. After the connection is terminated, the command-line of the client may be as follows:

```
Command-line:
URL of the index file: www.foo.com/abc/index.txt
Lower endpoint = 1587
Upper endpoint = 6999
Index file is downloaded
There are 5 files in the index
1. www.cs.bilkent.edu.tr/file.txt is not found
2. www.cs.bilkent.edu.tr/folder2/temp.txt (range = 1587-5999) is downloaded
3. wordpress.org/plugins/about/readme.txt (range = 1587-4566) is downloaded
4. humanstxt.org/humans.txt (size = 1587) is not downloaded
5. www.cs.bilkent.edu.tr/morhan/cs421/deneme.txt (range = 1587-6999)
is downloaded
```

Submission rules

You need to apply all of the following rules in your submission. **You will lose points if you do not obey the submission rules below or your program does not run as described in the assignment above**.

- The assignment should be submitted as an e-mail attachment sent to **morhan[at]cs.bilkent.edu.tr**. Any other methods (Disk/CD/DVD) of submission will not be accepted.
- The subject of the e-mail should start with *[CS421_PA1]*, and include your name and student ID. For example, the subject line must be

  [CS421_PA1]AliVelioglu20111222

if your name and ID are *Ali Velioglu* and *20111222*. If you are submitting an assignment done by two students, the subject line should include the names and IDs of both group members. The subject of the e-mail should be

[CS421_PA1]AliVelioglu20111222AyseFatmaoglu20255666

if group members are *Ali Velioglu* and *Ayse Fatmaoglu* with IDs *20111222* and *20255666*, respectively.

- All the files must be submitted in a zip file whose name is the same as the subject line **except** the [CS421_PA1] part. The file must be a **.zip** file, not a **.rar** file or any other compressed file.
- All of the files must be in the root of the zip file; directory structures are **not** allowed. Please note that this also disallows organizing your code into Java packages. The archive should **not** contain:
  - Any class files or other executables,
  - Any third party library archives (i.e. jar files),
  - Any text files,
  - Project files used by IDEs (e.g., JCreator, JBuilder, SunOne, Eclipse, Idea or NetBeans etc.). You may, and are encouraged to, use these programs while developing, but the end result must be a clean, IDE-independent program.
- The standard rules for plagiarism and academic honesty apply, if in doubt refer to 'Student Disciplinary Rules and Regulation', items 7.j, 8.l and 8.m.