Assignment 1

Computer Science 441

Due: 23:55, Friday October 6, 2017

Instructor: Majid Ghaderi

1 Objective

The objective of this assignment is to practice network programming and learn about application layer protocols. Specifically, you will implement an HTTP client program to download web objects from various web servers on the Internet.

2 Overview

In this assignment, you will implement a simple HTTP client from scratch. You have to write code to establish a TCP connection to a given server and send and receive HTTP messages. Your program should be able to download both binary and text objects. All downloaded objects are stored locally.

Your program should maintain a list of the objects (called *catalog*) that are stored in the local storage. The catalog will be used to check if an object needs to be downloaded from the Internet or can be served from the local storage. In case a copy of the object already exists locally, your program should request a new copy from the origin web server only if the object has been recently updated on the server. This is achieved by implementing the **conditional GET** as described in class. In case the requested object has been modified, your program should replace the local copy with the new copy received from the web server. **You should not blindly download every object from the origin web server again**.

To implement conditional GET, the `Last-Modified` information for the object being requested is required. Thus, in addition to object URLs, the catalog should store the `Last-Modified` value for each object as well. While there are a number of ways to implement a catalog, a simple Java `HashMap` might be a good choice (you are free to choose other structures including a simple text file). Make sure that every time an object is downloaded, the catalog is updated with the object's information (*i.e.*, its URL and `Last-Modified`). The catalog should be saved in a file so that its information is preserved between different runs of the program. For simplicity, every time the catalog changes, save it in a file in the local directory where your program is running.

Your implementation should include appropriate exception handling code to deal with various exceptions that could be thrown in the program. Skeleton code for a Java class called `UrlCache` is provided to you. You are asked to write code to complete the implementation of this class. There are three methods in the class that you need to implement. A description of these methods follows:

- `UrlCache() throws IOException`
  This is the no-argument constructor that may include code for initialization. You should

also check to find out if a catalog file already exists. If so, initialize your catalog with the information read from the saved catalog file.

- `void getObject(String url) throws IOException`
  This is the main method for downloading objects from the Internet. The parameter `url` is a properly formatted URL that specifies the object to be downloaded. It has the following format:

  ```
  hostname[:port]/pathname
  ```

  where, `[:port]` is an optional part which specifies the server port. If no server port is specified, use the default port 80.

  To download an object, you should establish a TCP connection to the server on the specified port number. Then create an HTTP request using the object pathname and send the request to the server. Once the request is submitted, start reading the HTTP response from the socket and save the received object to a file. The file name should match the `hostname/pathname` of the URL with the same directory structure being created locally inside the directory where your program is running. You may find class `File` useful for this purpose. Once a URL is downloaded, you need to record its `Last-Modified` value in the catalog, which can be obtained from the header part of the HTTP response.

  As mentioned earlier, your program should download a new copy of an object only if the local copy is out of date by implementing conditional GET.

- `long getLastModified(String url)`
  This method simply returns the `Last-Modified` time associated with the object specified by the parameter `url`. If the object is not in the catalog (*i.e.*, does not exist locally) then throw an exception of type RuntimeException. Otherwise, the time is returned in millisecond as in the method `getTime()` in Java class `Date`.

Restrictions

- You are not allowed to change the signature of the methods provided to you. You can however implement additional methods and classes if you wish.
- You are not allowed to use the class `URL` or `URLConnection` for this assignment. Ask the instructor if you are in doubt about any specific Java classes that you want to use in your program.