# Programming Assignment 2: Proxy

## Important Deadlines

- Due: 14/10/2013, 23:59
- Late Penality: 10% per day
- **Submission Format**: Create a folder "P2" under your home directory on cs2105-z.comp.nus.edu.sg, and upload your source code to this folder. Within the folder "P2", the compile command for your source code should be "javac *.java", and the command to run your source code should be "java proxy <portNum> fileSub fileRedirect". (All without quotation marks.)

  **Warning: You should strictly follow the above submission format, otherwise you will get certain penalty!**

## 1 Overview

In this assignment, you will implement a simple web proxy that passes requests and data between a web client and a web server. This will give you a chance to get to know one of the most popular application protocols on the Internet, the Hypertext Transfer Protocol (HTTP).

## 2 Review: The Hypertext Transfer Protocol

The Hypertext Transfer Protocol or (HTTP) is the protocol used for communication on this web. That is, it is the protocol which defines how your web browser requests resources from a web server and how the server responds. For simplicity, in this assignment we will be dealing only with version 1.0 of the HTTP protocol, defined in detail in RFC 1945. You may refer to this document when you need to determine the appropriate behavior of your proxy.

HTTP communications happen in the form of transactions, a transaction consists of a client sending a request to a server and then reading the response. Request and response messages share a common basic format:

- An initial line (a request or response line, as defined below)
- Zero or more header lines
- A blank line (CRLF)
- An optional message body.

For most common HTTP transactions, the protocol boils down to a relatively simple series of steps:

1. A client creates a connection to the server.

2. The client issues a request by sending a line of text to the server. This request line consists of a HTTP method (most often GET, but POST, PUT, and others are possible), a request URI (like a URL), and the protocol version that the client wants to use (HTTP/1.0). The message body of the initial request is typically empty.

3. The server sends a response message, with its initial line consisting of a status line, indicating if the request was successful. The status line consists of the HTTP version (HTTP/1.0), a response status code (a numerical value that indicates whether or not the request was completed successfully), and a reason phrase, an English-language message providing description of the status code. Just as with the the request message, there can be as many or as few header fields in the response as the server wants to return. Following the CRLF field separator, the message body contains the data requested by the client in the event of a successful request.

4. Once the server has returned the response to the client, it closes the connection.

## 3 HTTP Proxies

Ordinarily, HTTP is a client-server protocol. The client (usually your web browser) communicates directly with the server (the web server software). However, in some circumstances it may be useful to introduce an intermediate entity called a proxy. Conceptually, the proxy sits between the client and the server. In the simplest case, instead of sending requests directly to the server the client sends all its requests to the proxy. The proxy then opens a connection to the server, and passes on the client's request. The proxy receives the reply from the server, and then sends that reply back to the client. Notice that the proxy is essentially acting like both a HTTP client (to the remote server) and a HTTP server (to the initial client).

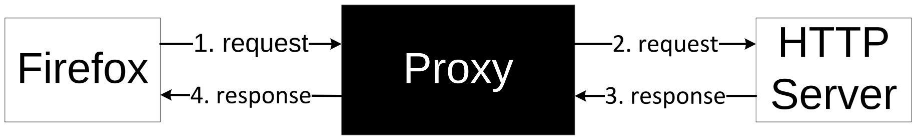
Why use a proxy? There are a few possible reasons:

- **Performance**: By saving a copy of the pages that it fetches, a proxy can reduce the need to create connections to remote servers. This can reduce the overall delay involved in retrieving a page, particularly if a server is remote or under heavy load. Moreover, the system performance could be significantly enhanced via using caching proxies. Caching proxies keep local copies of frequently requested resources, allowing large organizations to significantly reduce their upstream bandwidth usage and costs, while significantly increasing performance.

- **Content Filtering and Transformation**: While in the simplest case the proxy merely fetches a resource without inspecting it, there is nothing that says that a proxy is limited to blindly fetching and serving files. The proxy can inspect the requested URL and selectively block access to certain domains, reformat web pages (for instances, by stripping out images to make a page easier to display on a handheld or other limited-resource client), or perform other transformations and filtering.

- **Privacy**: Normally, web servers log all incoming requests for resources. This information typically includes at least the IP address of the client, the browser or other client program that they are using (called the User-Agent), the date and time, and the requested file. If a client does not wish to have this personally identifiable information recorded, routing HTTP requests through a proxy is one solution. All requests coming from clients using the same proxy appear to come from the IP address and User-Agent of the proxy itself, rather than the individual clients. If a number of clients use the same proxy (say, an entire business or university), it becomes much harder to link a particular HTTP transaction to a single computer or individual.

## 4 Assignment Details



### 4.1 The Basics

Your first task is to build a basic web proxy capable of accepting HTTP requests, making requests from remote servers, and returning data to a client.

You need to write your program in **JAVA** such that it can be compiled and executed on cs2105-z.comp.nus.edu.sg, producing a binary called proxy that takes three arguments. The first argument is the port to listen from. (Do not use a hard-coded port number). The second and third arguments are file paths. The details about these two files will be described in section 4.6.

### 4.2 Listening

When your proxy starts, the first thing that it will need to do is establish a socket connection that it can use to listen for incoming connections. Your proxy should listen on the port specified from the command line, and wait for incoming client connections.

Once a client has connected, the proxy should read data from the client and then check for a properly-formatted HTTP request. An invalid request from the client should be answered with an appropriate error code.

### 4.3 Parsing the URL

Once the proxy sees a valid HTTP request, it will need to parse the requested URL. The proxy needs at most two pieces of information: the requested host and port (by default is 80).

### 4.4 Getting Data from the Remote Server

Once the proxy has parsed the URL, it can make a connection to the requested host (using the default remote server port 80) and send a HTTP request for the appropriate file. The proxy then sends the HTTP request that it received from the client to the remote server.

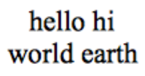
### 4.5 Returning Data to the Client

After the response from the remote server is received, the proxy should send the response message to the client via the appropriate socket. Once the transaction is complete, the proxy should close the connection.

### 4.6 Command Line Arguments

The first command line argument that needs to be passed into the main program is the port number to which the proxy should listen.

The second argument is the path to a file (filename is "fileSub") which will be used by the proxy for text substitution. *The file given will consist of string pairs (an extract of an example file is given in Fig. 1), separated by a space. (Each string consists of only alphabet characters and is case sensitive).* Given this file as input the proxy will need to find whether the first string of a any given string pair is existing in a web page that is being forwarded to the client. If it is the case every occurrence of the first string needs to replaced by the second string before passing on to the client.

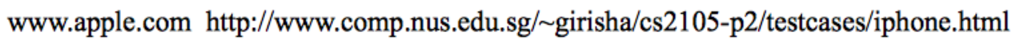


```
hello hi
world earth
```

Figure 1: Extract of a file to be used by the Proxy for text substitution

For an example if the following file (Fig. 1)) was given as input and a request to a web page made by a client contains the string 'hello', then all the occurrence of 'hello' needs to be replaced by 'hi' before being passed to the client.

The third and final argument is the path to a file (filename is "fileRedirect") which will be used by the proxy for re-direction. The file given will consist of (HOST URL) pairs (an extract of an example file is given in Fig. 2), separated by a space. Given this file as input the proxy needs to check whether any client makes a request to a HOST that exists in the input file. If it is the case the client should be re-directed to the corresponding URL.



```
www.apple.com  http://www.comp.nus.edu.sg/~girisha/cs2105-p2/testcases/iphone.html
```

Figure 2: Extract of a file to be used by the Proxy for re-direction

For an example if a client makes a request to a URL `http://www.apple.com/` which corresponds to the HOST www.apple.com then the client should be re-directed to the page `http://www.comp.nus.edu.sg/~girisha/cs2105-p2/testcases/iphone.html`.

**(Note: There can be up to 10 lines in "fileSub" or "fileRedirect".)**

## 5 Testing Your Proxy

Run your client with the following command:

```
java proxy <portNum>  fileSub fileRedirect
```

**(Note: your "proxy" should be able to handle multiple HTTP requests. In particular, the "proxy" should persistently listen on the port specified without terminating.)**

### 5.1 Configuring Firefox to Use the Proxy

***Download the Firefox 16.0.1 from:*** *`http://www.comp.nus.edu.sg/~wanghui/faq/`*

#### 5.1.1 Configuring Firefox Proxy

1. Select 'Options' from the menu.
2. Click on the 'Advanced' icon in the Options dialog.
3. Select the 'Network' tab, and click on 'Settings' in the 'Connections' area.
4. Select 'Manual Proxy Configuration' from the options available. In the boxes, enter the hostname and port where proxy program is running.

#### 5.1.2 Configuring Firefox to use HTTP/1.0

Because Firefox defaults to using HTTP/1.1 and your proxy speaks HTTP/1.0, there are a couple of minor changes that need to be made to Firefox's configuration. Fortunately, Firefox is smart enough to know when it is connecting through a proxy, and has a few special configuration keys that can be used to tweak the browser's behavior.

1. Type 'about:config' in the title bar.
2. In the search/filter bar, type 'network.http.proxy'
3. You should see three keys: network.http.proxy.keepalive, network.http.proxy.pipelining, and network.http.proxy.version.
4. Set keepalive to false. Set version to 1.0. Make sure that pipelining is set to false.

## 6 Grading

The assignment will be graded out of 140 points:

### 6.1 Basic Proxy (100 pts)

Two files "fileSub" and "fileRedirection" are empty.

1. **(20pts)** When compiling your program, it should compile without errors or warnings on cs2105-z.comp.nus.edu.sg and produce "proxy.class". The first command line argument should be the port number that the proxy will listen from.
2. **(50pts)** Your proxy will be tested with a list of public web pages using the firefox browser. The list of URLs that will be used in the grading are
   - `http://www.comp.nus.edu.sg/~girisha/cs2105-p2/testcases/simple_webpage.html` (10 pts)
   - `http://www.comp.nus.edu.sg/~girisha/cs2105-p2/testcases/my_file.html` (10 pts)
   - `http://www.comp.nus.edu.sg/~girisha/cs2105-p2/testcases/notfound.html` (10 pts)
   - `http://www.comp.nus.edu.sg/maps/images/COM2_L2.jpg` (10 pts)
   - `http://www.comp.nus.edu.sg/~girisha` (10 pts)
3. **(30pts)** Your proxy will be tested with other hidden test cases. (e.g., large files, http video streaming).

### 6.2 Extensions (40 pts)

1. Text Substitution: File "fileSub" contains only one line "Hello Hi" and file "fileRedirect" is empty, the test url is: `http://www.comp.nus.edu.sg/~girisha/cs2105-p2/testcases/simple_webpage.html`. Your proxy needs to identify this url and replace all occurrences of "Hello" with "Hi" and return to the browser (The page displayed will be: `http://www.comp.nus.edu.sg/~girisha/cs2105-p2/testcases/simple_webpage_sub.html`) ). (10 pts)
2. Redirect: File "fileRedirect" contains only one line "www.apple.com `http://www.comp.nus.edu.sg/~girisha/cs2105-p2/testcases/iphone.html`" and file "fileSub" is empty. The test url is: `http://www.apple.com`. (10 pts)
3. **(20 pts)** Reserved for hidden test cases (e.g., both "fileSub" and "fileRedirect" are non-empty; more than one lines in "fileSub" and "fileRedirect").

## 7 Assistance Regarding the Programming Assignment

We have set up a FAQ for the assignment and it can be accessed at `http://www.comp.nus.edu.sg/~wanghui/faq/`. Please note that we will be updating the FAQ depending on the queried made by the students. Therefore, it is advisable to read the FAQ first if you encounter a problem. But if you need to clarify anything regarding the assignment you could contact

- Wang Hui (wanghui@comp.nus.edu.sg)
- Girisha De Silva (girisha@comp.nus.edu.sg)

You are also encouraged to post questions regarding the programming assignment in the IVLE forum.