Assignment 4:
Dynamic Web Pages, Part 1: CGI Scripts & Java Applets

- This assignment will be introduced in a class meeting on November 15th, 2016 at 11:30 in seminar room 0.124.
- The groups will present the solution of this assignment on November 22nd, 2016 at 11:30. We will meet in seminar room 0.124.
- You may work on the assignments in groups of up to three people (if possible, keep the same groups as in the previous assignments).
- All members of a group have to show up together for the grading of the assignment. For the grading, each group will need to have the source code ready for the discussion as well as the running implementation to be presented by the group. Moreover, each group member might be asked questions about the solution.
- If you have questions, send an email to adnan.tariq@ipvs.uni-stuttgart.de.

Task 1 – CGI Scripts

In this task, a web-based phone book shall be implemented using CGI scripts. The user interface consists of an HTML page containing an HTML form with two text fields (first name and last name) to enter the name of the person whose phone number is to be retrieved. The phone book entries are stored on the web server as plain text file (see below for details about the file format and the location of a sample file). When the user submits a query to the web server by entering the name into the HTML form and clicking the "submit" button, a CGI script on the web server is invoked that searches the text file for the wanted name. If the name is found, the telephone number associated with the name is returned in form of an HTML page. If the name is not found, an HTML page with an error message is displayed.

Task 1.1

Write an HTML page that contains an HTML form with two text fields for first name and last name, and a submit button. Make sure to write valid HTML code including all necessary HTML elements (not only the form elements). Upload your page to the web server on *netappsvm* (see below).

Task 1.2

Write a CGI script implementing the mentioned search functionality. On invocation by the web server, this script should receive the query parameters (first and last name) from the web server and return an HTML page with the query result or an HTML page with an error message to the web server as mentioned above. The script of this task should support the GET method for passing parameters between the web client and web server, which also influences the way parameters are passed between web server and CGI script.

You can find a text file called `phonebook.txt` with names and phone numbers in the directory `/home/tariqan/assignment4/` of the server *netappsvm* (log in via ssh to *netappsvm*, and use the copy command cp to copy the file). Each line contains one entry as comma separated values

(CSV) list (lastname,firstname,phone number). You can read this file from your CGI script by using common file I/O operations (i.e., you do not need the HTTP protocol or network communication to load the phone book from the CGI script).

For the implementation, you should use Java, C, or C++. Note that if you use Java, you have to use a shell script to invoke your Java application (see below). Upload your CGI script to the web server *netappsvm* (see below) and test it with the HTML form of Task 1.1.

How does the HTTP GET request look like? In particular, describe the place and format of the search parameters in the HTTP request. (Hint: You may use your implementation of the simple web server from Assignment 3 to inspect the request received from the web browser.)

How are special characters encoded? Your script should be able to handle parameters including special characters. For instance, test your script with the name “Frank Dürr”.

Task 1.3

Implement a CGI script with the same search functionality as the script of Task 1.2, however, this time using the POST method for passing search parameters.

How does the HTTP POST request look like? In particular, describe the place and format of the search parameters in the HTTP request.

Task 1.4

Assume you have to upload a JPEG image from a browser to a web server, for instance, to upload an image to Flickr or Facebook. Which HTTP method would you use (POST or GET)? Explain your answer.

Continued on next page →

Task 2 – Java Applets

In this task, you will implement a phone book viewer (Phone Book Applet) using Java Applets. The Phone Book Applet shall display the entries from the phone book file of Task 1 in a table. As starting point, a skeleton of the Phone Book Applet is provided on the server *netappsvm*:

```
/home/tariqan/assignment4/PhoneBookApplet.java
```

This class already implements the basic applet and the table GUI element. However, it only displays one dummy entry in the phone book. You will have to extend this applet later in Task 2.2.

Since applets are executed on the client (browser) rather than the server, the Applet first has to download the phone book file from the server using HTTP. Then it parses the entries and inserts them into the table object. Basically, you need to fill the two-dimensional array `data` in the method `loadPhoneBookData()` (cf. source code in the file `PhoneBookApplet.java`).

Task 2.1

Write an HTML page that loads the Phone Book Applet class from the web server and executes it in a browser. Upload your HTML page to the web server *netappsvm*.

Compile the above skeleton class and upload the class file to the web server so it can be loaded by your HTML page.

Test your page by calling it with a browser.

Note: In order to execute Java Applets, you need to install the Java plugin for your browser, which is included in the Java runtime environment (JRE):

http://www.oracle.com/technetwork/java/javase/downloads/index.html

Task 2.2

Extend the skeleton implementation such that it loads the phone book entries from the web server from the URL http://netappsvm.informatik.uni-stuttgart.de/~username/phonebook.txt by using an HTTP request. You may use a helper class such as `URLConnection` to request the file (hint: have a look at the method `openConnection()` from the URL class).

Upload your extended applet to the web server *netappsvm* and test it using the HTML page from Task 2.1.

Task 2.3

Change the phone book URL from

http://netappsvm.informatik.uni-stuttgart.de/~username/phonebook.txt

to

http://duerr-vm1.informatik.uni-stuttgart.de/~duerrfk/phonebook.txt

and test your applet again. Does it still work? Explain the behavior.

System Information

- You can use the web server `netappsvm.informatik.uni-stuttgart.de` to upload and test your CGI scripts and Java Applets implemented in this assignment.
- You can login to *netappsvm* with your personal login name and using SSH. The login names and passwords will be provided on request by email. The server *netappsvm* is only accessible from within the University network. From outside you have to use VPN.
- The web directory of the users is located at `/home/username/public_html/`. Please replace "username" by your user name. The URL of an HTML file, say foo.html, in this directory is

  http://netappsvm.informatik.uni-stuttgart.de/~username/foo.html.
- CGI scripts (or the shell script invoking a Java application) must be stored in the directory `/home/username/public_html/cgi-bin/` to be executed by the web server.
- To make a CGI script executable, use the command `chmod a+x filename`. This is necessary for scripts implemented in a scripting language (not considered here), or shell scripts invoking a Java application (see below) as well as binary programs implemented, for instance, in C or C++.
- If you use Java for implementing the CGI scripts, you have to call them from a shell script. Upload your (single) class file to the CGI directory

  (`/home/username/public_html/cgi-bin/`). Then create a shell script with the following content and call this shell script from the action argument of the form:

  ```
  #!/usr/bin/bash
  java ClassName
  ```
- Environment variables can be read using the `getenv()` method in Java or C.
- To let the web server access resources (HTML pages, CGI scripts, etc.), they must be readable by the server process. You can make files readable by everyone (including the web server) using the following command: `chmod a+r filename`

Recommended reading

- Tutorial on Java Applets:
  http://download.oracle.com/javase/tutorial/deployment/applet/