Exercise 2 – Praktikum Entwicklung Mediensysteme

Using HTTP-Connections and Record Stores

This exercise combines two important topics in Java ME-development: Establishing and using connections to remote resources via the Internet as well as permanently storing data on mobile devices. The following steps explain the different tasks of this exercise in more detail and give hints for their solution:

- For this exercise create a MIDlet that allows you to send an arbitrary string to a server/website on the Internet, retrieve its answer and store it permanently in the MIDlet.
- The layout of this MIDlet's interface is up to you, but nevertheless it should reflect the different mandatory parts of the exercise's tasks: Inserting an arbitrary string, checking the validity of the input (e.g. no connection to the website if the input is an empty string), sending the input to the server (e.g. a waiting screen), displaying the answer from the server and displaying a history of its last 5 answers. Figure 1 gives some visual examples of how the necessary interfaces could look like.
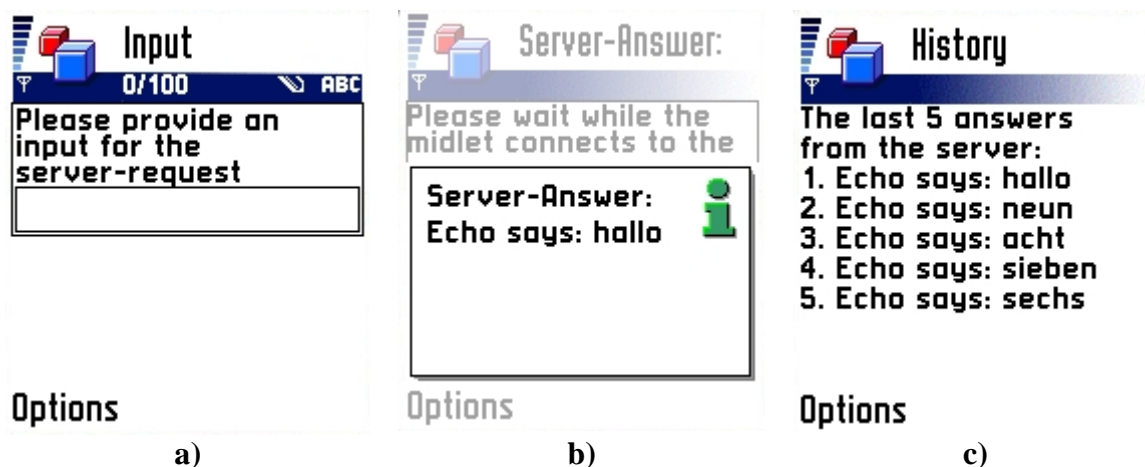

a)   b)   c)

Figure 1: Examples for possible interfaces

Each student must check-in his/her own solution to his personal SVN folder by **Tuesday, May 1th, 12 p.m.** Create a sub-folder called *excercise2* and hand in a zip-file named *exercise2.zip*.

Appendix: Hints and Details

- [1] provides good some resources for getting to know different topics in Java ME-development. For the creation of interfaces, you can start with [2]. Another good source for details on implementation issues is the MIDP 2.0 API [3].
- After the user has provided an arbitrary string, he should be able to send it to the server using an HTTP POST- or GET-request. For that purpose, please make yourself familiar with the basics of MIDP network programming and have a look at [4], [5] and [6].
- The URL of the target website is **http://murx.medien.ifi.lmu.de/~gregor/echo.php.** Use "**?input=tralala**" as parameter for your requests in order to get a response.
- Please implement and run your connection to the server in an extra thread and show a waiting screen (e.g. with a gauge) until your request to the server is answered. Simple threading is sufficient!
- However, if you are interested in a more sophisticated and elegant threading solution, have a look at [7]. Even if this approach is not mandatory for this exercise, it might be useful for future applications ;-)
- After the server has sent its answer, display it on the screen of the mobile phone and store it permanently using Java ME's Record Management System (RMS). [8] provides all necessary information.
- Storing the server's answers is the foundation for the history-feature. It displays the last 5 answers from the server even after the MIDlet has been shut down and restarted. Please use the afore mentioned resources for implementing an appropriate interface and the corresponding application logic.

Resources

[1]   http://developers.sun.com/techtopics/mobility/reference/techart/index.html
[2]   http://developers.sun.com/techtopics/mobility/midp/articles/ui/
[3]   http://java.sun.com/javame/reference/apis/jsr118/
[4]   http://developers.sun.com/techtopics/mobility/midp/articles/servlets/
[5]   http://developers.sun.com/techtopics/mobility/midp/articles/network/
[6]   http://developers.sun.com/techtopics/mobility/midp/ttips/HTTPPost/
[7]   http://developers.sun.com/techtopics/mobility/midp/articles/threading/
[8]   http://developers.sun.com/techtopics/mobility/midp/articles/persist/