Lab 5: Implementing and Evaluating a Database-Driven Twitter Client in Java

Assigned: Thursday, February 28, 2013

Due: Thursday, March 14, 2013

Purpose

To explore the steps that are needed to implement a complete database application in the Java programming language. First, to use an existing program to connect to, modify, and query a database managed by the HSQLDB database management system. Then, starting with an existing Twitter client that uses a file to store the desired tweets, students will implement a full-featured Twitter client that uses a relational database as the main storage mechanism.

Deliverables

In order to satisfactorily complete this laboratory assignment, you must submit the following deliverables. Each of the laboratory notebook entries must be placed inside of the course Web site area that you created. Each of the printed code segments must include a signed pledge on the printed document. The laboratory notebook should be placed inside the course Web site page associated with this laboratory. Make sure that the name of your laboratory notebook is linked to from the Web site page for this laboratory. If you have questions about this issue, please see the Instructor. When you submit this laboratory assignment, please print out the laboratory notebook and use it as the cover sheet for your entire assignment.

Laboratory Notebook

1. An overview of the HSQLDB database system. In order to complete this deliverable, you should review the online documentation about HSQLDB and then summarize the important details in your own words. Your response to this question should also explain how HSQLDB and SQLite are similar to and different from each other.

2. A brief one to two paragraph description of the steps that you would take to test the `TwitterClient` program. Your discussion should include a description of the types of inputs that you would provide to the program during the testing phase and how you would check to see if the Twitter server was correctly updated. Your response should specifically focus on how you would test the `TwitterClient`'s interaction with the relational database.

3. A review of each pitfall or problem that you encountered during the laboratory and a discussion of the techniques that you used to overcome this difficulty. If no difficulties were encountered, please include a record of important commands, notes, etc. that you think will be useful during later laboratories. You should use the notebook to describe important GNU/Linux commands, operating system tools, or Java source code segments that you found interesting, useful, and/or confusing.

Source Code and Output

1. The output from running the two phases associated with the `FindFile` example. For the second phase of the program's use, you should show that it correctly works for at least three different input patterns.

2. A discussion of the contents and meaning of the three database files that are produced by the `FindFile` application that uses the HSQLDB database. You should provide excerpts from each of these files and then explain the meaning of these excerpts.

3. The source code from the `TwitterClient` and `Tweet` classes that meet the specification outlined in this laboratory assignment sheet. The starting point for `TwitterClient` will be the source code provided in the assignment sheet. However, you are responsible for extending the program so that it stores the input tweets inside of a database. Your `TwitterClient` must also implement one additional feature that relies on the storage of data in a relational database.

4. A commented version of the source code for `TwitterCreateProperties` that explains what the most important lines of code are actually doing.

5. The output from running the `TwitterCreateProperties` program at the command prompt of a terminal window. This

output should clearly demonstrate that you were able to connect to the Twitter Web site and ultimately input the required PIN to `TwitterCreateProperties`. Furthermore, you should take a screen shot of the Twitter Web site showing both that you pasted in the right URL and that you displayed the PIN.

6. Before and after running the `TwitterCreateProperties` program, you should print out the `twitter4j.properties` file that contains various configuration parameters for the `TwitterClient`.

7. The output from executing the final version of `TwitterClient` client at the command prompt of a terminal window. This output should clearly demonstrate that your program meets all of the specifications that are outlined below. For instance, you should show how all of the command-line parameters work.

8. A screen shot that shows what your Twitter home timeline looks like after you have posted one or more tweets using the `TwitterClient`.

9. A discussion of the contents and meaning of the three database files that are produced by the database-driven version of the `TwitterClient` application that uses the HSQLDB database. You should provide excerpts from each of these files and then explain the meaning of these excerpts. In particular, you should explain the relational schema that is created by your `TwitterClient`.

   Your Java classes must fully meet the specifications that are outlined later in the laboratory assignment. Furthermore, they must adhere to the following stylistic conventions.

1. There must be comments at the beginning with your name and the word ``PLEDGE'' (you must sign the program here before you hand in the printout). Following this, there should be comments briefly describing what the program does. For each class and method that you declare, you must use the standard JavaDoc commenting tags to describe their purpose (e.g., `@author` inside of a `/** ... */` comment block).

2. There should be comments as needed throughout the program to explain variables and summarize the important points in your program. For example, when you declare a variable, you should include a comment that describes the purpose of this variable. When you assign a new value to a variable, you should include a comment to describe the intended purpose of the assignment.

3. The indenting scheme of your Java program must be consistent-all ``{...}'' pairs should be aligned, all statements inside ``{...}'' should be indented the same amount. Remember, `emacs` or another text editor or integrated development environment can help with this task!

4. Use blank lines in your program to improve readability and to separate things into logical groupings of statements. For instance, it makes sense to group your variable declarations together and separate them from other statements.

5. Use Courier or another fixed-width font for the program and the output.

Using Relational Databases in Java programs

After learning more about the HSQLDB database management system, download the database and configure it correctly for your system. Next, download the `FindFile.java` class and study it carefully. In particular, please note that `FindFile` must run in two separate phrases. Please pick a suitable directory and then run the two phases of `FindFile`. What are the inputs and outputs of this program? How does the program work? What are the steps that `FindFile` takes to interact with an HSQLDB database? Finally, after you have run both phases of the `FindFile` program, you should investigate the contents of the files that start with the name `FindFileDatabase`. How many files are there that start with this name? What is the purpose of each file? What types of data are stored in these files?

Implementing Database-Driven Applications

Once you have learned how to use an HSQLDB database in a Java program, you are ready to extend an existing program to use an HSQLDB database of its own. Today you will download and use a Twitter client implemented in the Java programming language. However, after examining the source code of this Twitter client you will notice that it stores all of the tweets inside of a plain text file. This approach creates a number of problems for the Twitter client. For instance, instead of allowing a database to handle these concerns, the current implementation must check that each of the tweets adheres to the length requirements. In response to these limitations, today we will implement our own database-driven Twitter client. After creating your own account in Twitter, you will need to download the following files and save them in the directory for this assignment.

1. `twitter4j-core-3.0.3.jar`
2. `twitter4j.properties`
3. `TwitterCreateProperties.java`
4. `Tweet.java`

5. TwitterClient.java
6. Tweets.txt

Now, you need to set your CLASSPATH environment variable to contain the twitter4j-core-3.0.3.jar file. After correctly setting this environment variable, you should be able to compile the program called TwitterCreateProperties.java. However, before you run this program you should examine the contents of twitter4j.properties and learn about the Twitter4J Java library that we will use to complete this laboratory assignment. What did you see in the properties file? Finally, go ahead and run TwitterCreateProperties, following the instructions to create a new twitter4j.properties and to authorize your own twitter client to access your Twitter account. What output did this program produce? What Web sites did you load in the Twitter system? After running TwitterCreateProperties, what are the contents of the twitter4j.properties file?

At this point, you are ready to run the TwitterClient. Currently, this program provides functionality to display your home timeline and to post the tweets that are provided in a text file. Please edit the Tweets.txt text file to add some tweets that you can send to Twitter. In order to ensure that other students in the class can easily follow what you are tweeting, please use the hash tag #CS3804Life.

Once you have started to correctly use this program, please study the source code to learn how it works. For instance, if you want to learn more about the updateStatus method, you can check the Twitter4J JavaDocs. Right now, our TwitterClient has a very limited user interface. As one way to overcome this limitation, you should implement a command-line interface that will allow the user to only read the home timeline, only update the status through the "Tweets.txt" file, or to do both. In order to implement this feature, you will need to add conditional logic to your program. Finally, you should implement at least one additional feature of your choice. For instance, you could implement a random tweet generator that will create a random tweet and then post it to Twitter. Or, you could implement functionality to support direct messaging through Twitter.

Once you are finished with the enhancements to the base TwitterClient, you should start to think about how you can make a database-driven version of the same program. Using the source code from the FindFile program as a source of inspiration, please implement a TwitterClient that works in two phases. In the first phase, the program should read tweets from the user and store them in an HSQLDB database. You should design the schema of the database so that it verifies that all of the tweets are of the correct length. The second phase of the program should be able to connect to the database, read the tweets, and send each of the tweets to the Twitter servers. You should also implement at least one additional feature using the database. For example, you could implement an "offline mode" for TwitterClient that could connect to the Twitter servers, download your home timeline, and store it in a database so that it could be accessed even when you are not able to connect to the network.
