News Server
C++ Programming

Alexander Skafte	tfy13ask@student.lu.se
Andreas Forslöw	tfy14afo@student.lu.se
Linus Röman	tpi13lro@student.lu.se
Mattias Johansson	tpi13mjo@student.lu.se

April 18, 2016

1 Classes and files

1.1 `Newsgroup`

The `Newsgroup` class is meant to represent one newsgroup containing articles. Every newsgroup stores the articles that belongs to the newsgroup in a list. Every newsgroup object also has an ID number and a name.

1.2 `Article`

The `Article` class is meant to represent the different articles in a newsgroup. Every article object has a title, an ID number, an author and a text.

1.3 `Client`

The `Client` class is meant to handle all the different options one can choose from in the program. This class does the transformation from user input to something that can be sent via a connection.

1.4 `Adapter`

The `Adapter` class converts byte input into strings and integers.

1.5 `MessageHandler`

The `MessageHandler` class is meant to handle the conversion from strings and integers to bytes. These bytes are then sent to server.

1.6 `server_main`

The `server_main` file's purpose is to run the entire program; it is via this file that all the work is distributed throughout the classes.

1.7 `InMemoryDataBase`

The `InMemoryDataBase` class describes the in-memory database. Newsgroup objects are stored in a vector, which makes it easy to print out newsgroups in the order they were added.

1.8 `Protocol`

The `Protocol` class describes all the different constants in the program. It defines classification of the communication; making it easy to classify error messages and successful messages.

1.9 `DataBase`

The `DataBase` class is the interface class for both the in-memory database and the disk memory database.

1.10 DiskMemoryDataBase

The `DiskMemoryDataBase` class describes the disk memory database.

Think of the case when we are supposed to create an article; the first thing is that all the relevant info is going to be scanned and after that everything is sent via a connection to the server. In `server_main` we wait for activity. First we create a `MessageHandler` object. The first thing that is going to be received is `Protocol::CON_1`. Then the server will take over the control of the info from the user. This is done by receiving 4 strings from the connection. After this a `Protocol::COM_END` should be received. If it is the article is created in the database.

2 Conclusion

One thing we have not done is allowed formating of the article text. Once newline is entered, the article text is finished.

Requirements we fulfill is that the in memory database is functioning as it should.