Stanislav Peshterliev, 61096, SE

IM Chat Server/Client Documentation

Project Development Setup

Netbeans IDE 6.9.1 is used for project development. Open Netbeans and import projects server, client, common. Add jars from lib folder of common project and common project itself to server and client classpath. Sqlite3 database should be installed on your machine.

Server can be started from server/test/com.stanthinking.im.server.StartTestServer. When server is started make sure that all unit tests from server/test are passing.

Client can be started from client/test/com.stanthinking.im.client.TestApplication. Start several instances of the client in order to test chat functionality.

IM Server Architecture

IM Server is started as a process and listens on a specific port for incoming connections from clients in the main thread. Additional dispatcher thread is started which is responsible for dispatching message to clients. For each incoming client two threads are started - one for handling incoming messages and one for sending messages. Message queues are maintained in dispatcher and client's sending message thread in order to avoid the case when there are more messages than server machine is able to send. There are predefined actions to which server is designed to respond: Login, Logout, Register, Send. In order for a user to send message to another user they both must have registration and be authenticated with user name and password on the server.

Server is built with standard Java threading and networking APIs. I am using Google Guava library for its convenient API for working with collections and strings. Object persistence is implemented with SQLite database through. Connection to SQLite is made with JDBC driver sqlitejdbc-v056.

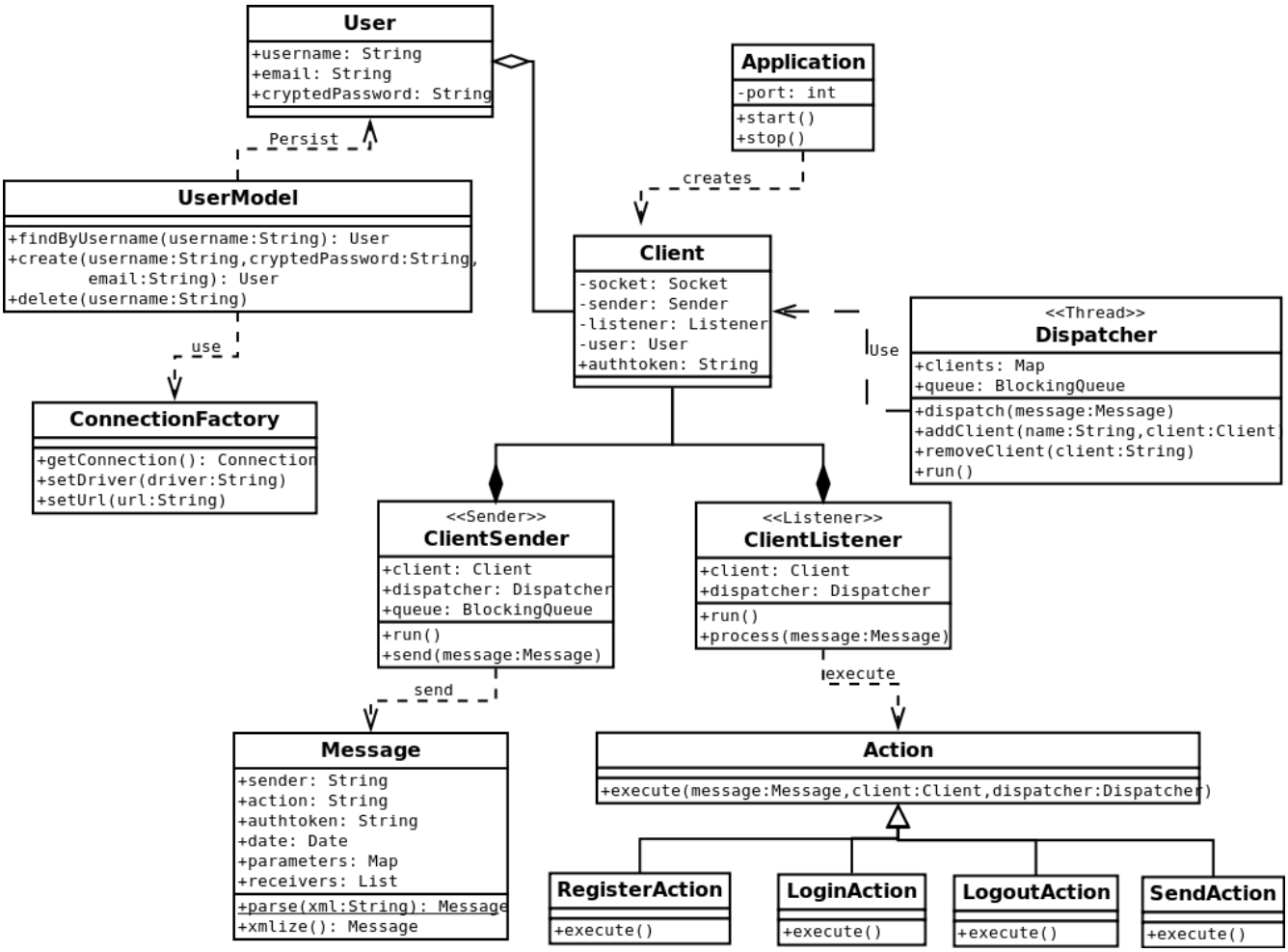
There are unit tests for core functionality and utility classes. I am using popular unit testing framework JUnit 4.5.

1. Package Structure

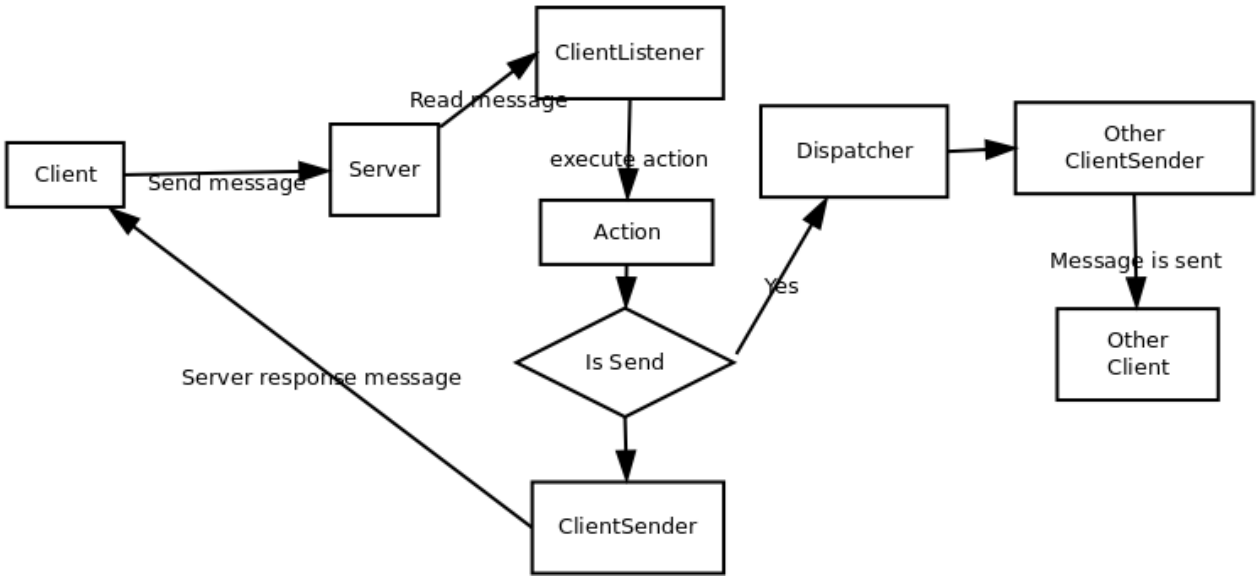
- com.stanthinking.im.server – main server classes for handling low level server details like networking communication and threading
- com.stanthinking.im.server.actions – action classes are responsible for handling user request; when message from user is received it is processed by one of the classess in this package
- com.stanthinking.im.server.actions.helpers – helper classes used by actions; helpers are used to bring a higher level of abstraction in implementation of action classes.
- com.stanthinking.im.server.model – classes connecting to database and manipulating persistent

object

- com.stanthinking.im.common – contains class shared between client and server application
- com.stanthinking.im.util – utility classes which can be used in any application



2. Process diagram



IM Client Architecture

IM Client is an application with graphical user interface which is installed on client machine. It is used to register, login and send/receive messages. On application startup two threads are created one that listens for incoming message and one that sends messages to the server. The thread that sends messages to server maintains a queue in case that the user sends more messages than client machine can handle.

Messages to the server are send in event handler of JFrame object located in com.stanthinking.im.client.ui package. These object are created using utility class Frames which graduate that there is only one object with given name supplied.

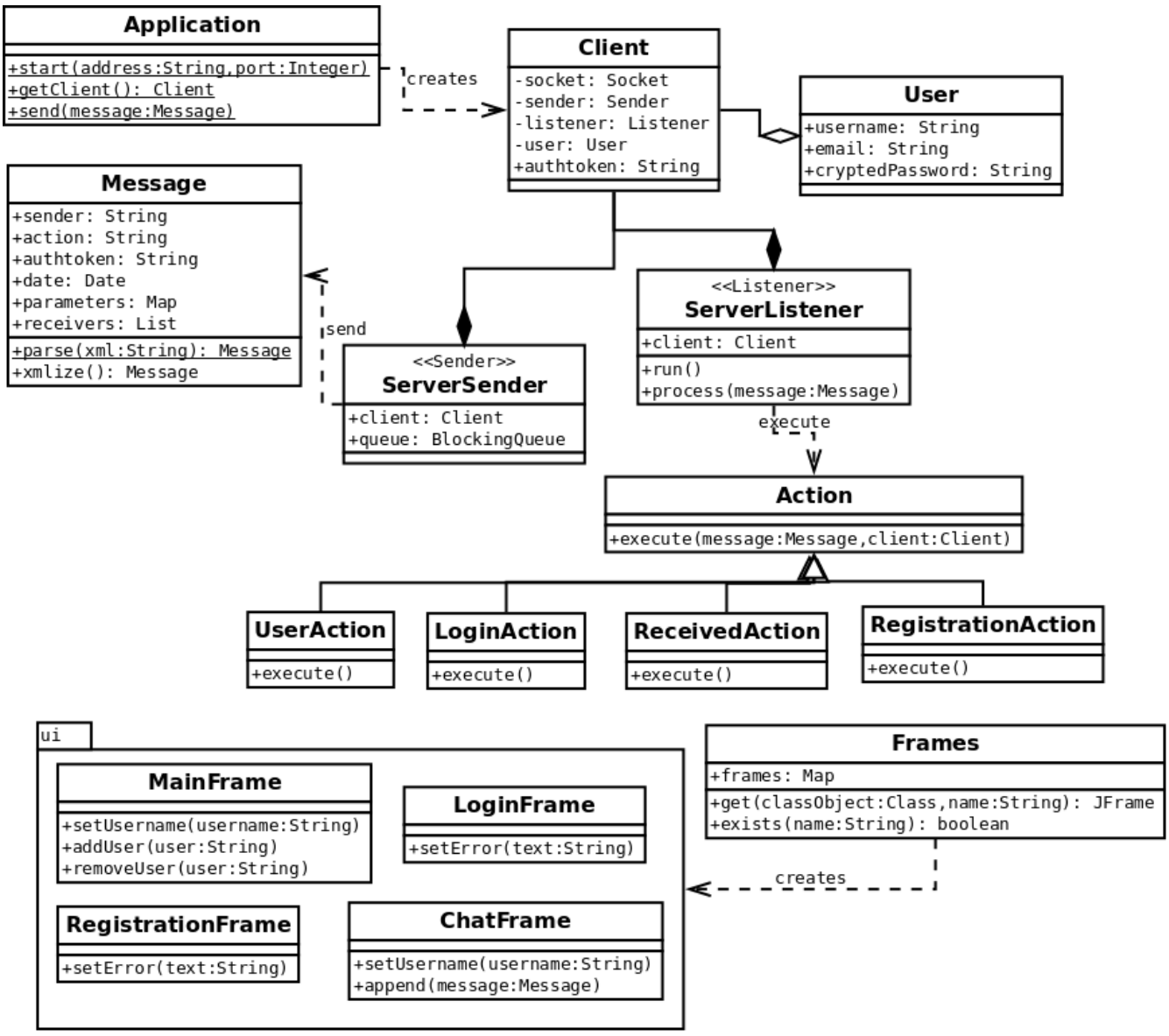
Messages sent from server are handled by action object located in com.stanthinking.im.client.ui.actions package.

Client is also build with standard Java threading and networking APIs. Graphical user interface is implemented with Swing GUI toolkit. I am using Google Guava library for its convenient API for working with collections and strings.

1. Package Structure

- com.stanthinking.im.client - main client classes for handling low level server details like networking communication and threading
- com.stanthinking.im.client.ui – classes related to graphical user interface and user interaction.
- com.stanthinking.im.client.ui.actions - action classes are responsible for handling server response responses and messages from other users; when message from server is received it is processed by one of the classess in this package and user interface is updated
- com.stanthinking.im.common - contains class shared between client and server application
- com.stanthinking.im.util - utility classes which can be used in any application

2. Class diagram



3. Use case diagram