Instant-Messenger Documentation

WHITE BOX TEST

MessageTester:

- SetUp: Prior to each test, this function creates a 'Message' object with predetermined properties. It configures the sender ID, recipient IDs, type, status, and content for future testing.

- TestConstructor: Determines if the constructor appropriately initializes the 'Message' object with the supplied characteristics, including sender ID, receiver IDs, timestamp, type, status, and content.

- testConstructorWithNullRecipients: Checks that the constructor handles null recipients correctly. It generates a message object with null recipients and determines whether the recipient IDs are not null and the size is zero.

- testGetSenderId: Determines if the 'getSenderId()' function gives the right sender ID for the message.

- testGetReceiverId: Determines whether the 'getReceiverIds()' method returns the correct set of receiver IDs associated with the message.

- TestGetTimeStamp: Verifies that the 'getTimestamp()' function gives a non-null timestamp showing when the message was created.

- TestGetType: Determines if the 'getType()' method appropriately returns the message's type, which in this instance is 'TEXT'.

- testGetStatus: Determines if the 'getStatus()' function appropriately returns the message's status, in this instance 'REQUEST'.

- TestGetContent: Determines whether the 'getContent()' function gives the message's right content, which is "Hello".

ClientTester:

- constructorTest: Verifies that when a new Client is created with the IP address provided and port number is provided by the user. It is not null, which means that the constructor successfully instantiates a Client object.

ClientUserTester:

ConstructorTest: Creates a ClientUser object from scratch with the username "tester" and a user ID of 1. The test determines whether the username and user ID correspond to the predicted values.The user ID is guaranteed to be set correctly by the assertion assertEquals(userId, user.getUserId()).In a similar vein, the username's correctness is confirmed using assertEquals(username, user.getUsername()).

- Constructor Test for Invalid parameters: This test examines how the constructor responds to invalid parameters. It specifically produces a ClientUser object with a null username and a negative user ID. Even when the user ID is negative, the assertion assertEquals(userId,user.getUserId()) verifies that the user ID is set appropriately.The username is kept null by using assertNull(user.getUsername()).

- Getter and Setter Test: examining how the user ID and username are handled by the getter and setter methods.Using the setter methods, set the username "Tester" and the user ID to 1.The assertions confirm that the anticipated values are returned by the getter methods.
- LoginTest: checks whether the login() method of a ClientUser object returns true.
- LogoutTest:verifies that the logout() method can be called without any exceptions.
- SetterAndGetterTest: The setter and getter methods for the conversation ID are the main topics of this test.You use the setter method to set a conversation ID of 123.The getter method's accuracy in returning the conversation ID is guaranteed by the assertion assertEquals(conversationId, user.getConversationId()).

ServerUserTester:

- setUp: Sets up a 'ServerUser' object with a username and password. It also generates a message queue to test message delivery and returns the output stream to null.

- testConstructor: Determines if the constructor appropriately configures the username, initial login status, and mailbox count.

- testAuthentication: Runs the 'authenticate' function to ensure that the user's password validation works properly.

- testLogin: Checks if the user can successfully log in and changes the login status accordingly.

- testLogout: Determines if the user can successfully log out and changes the login status accordingly.

- testReceive: Determines if the user can receive a message and updates the inbox count accordingly.

- testSendMessage: Evaluates the user's message sending ability. It mimics receiving a message, creating an output stream, sending the message, and determining if the inbox count reduces after delivery.

- testMessageQueue: Checks that the message queue works properly. It checks if the inbox count remains constant while no messages are in the queue.

- testConnection: Checks if the user's connection can be set and retrieved successfully.

- testConnectionNull: Determines whether the user's connection is set to null properly.

ConversationTester:

- ConstructorTest: this test will verify the correct initialization of a “Conversation” object by checking if the constructor correctly sets the participants and log file.
- addMsgTest: Checks the functionality of adding a message to the conversation.It creates a sample conversation,adds a message to

it,and then verifies if the added message is present in the conversation's list of messages.

- getAllMessageTest: verifies functionality of retrieving all messages from the conversation. It creates a conversation, adds multiple messages to it, and then checks if all the added messages are present in the list of all messages retrieved from the conversation.
- getLastMessageTest : this test makes sure the method "getLastMsg()"works correctly by retrieving the last message added to the conversation. It adds multiple messages to the conversation and verifies if the last added message matches the one retrieved using the "getLastMsg()"method.
- getIDTest: this test will verify if the "getID()" method of the Conversation class works correctly by returning the conversationID that was set during the initialization.
- getLogTest: this test checks if the "getLog()"method returns the correct log file associated with the conversation.By initiating a conversation with a log file and verifies the returned log file and verifies if the returned log file matches the one provided during initialization.

Black Box Test:

Multiple Clients:

- Create multiple instances of GUI clients (Minimum 3).

Login:

- Run the server and the GUI.

- The GUI should prompt you for server address and port Number.
- Test if the connection was successful.
- Upon successful connection, test if the GUI prompts you for Username and Password.
- Test if the login is successful after entering username and password and the Homescreen shows up.
- Repeat the login procedure for all the open client GUI.

Send Message:

- Start a conversation by Clicking New button. It should display the available users on the server.
- Press on one of them or hold shift to select multiple people to create a conversation.
- Test if the conversation was created successfully.
- Type in your message and hit send.
- After this, the message should show up in everyones screen.
- Do this for all participants and client GUI that are open.

Logout:

- Press logout on the homescreen and the server should log a client out and the client Gui should stop working.

Logs:

- See if the log files are being created and they have the record of messages sent in between different clients.
- Each conversation should have their own log file.

ServerUser:

- testCreateUser: Create ServerUser object and verifies that username and password initialized correctly, such as userID, username,login status and inbox count.
- testAddConversation: Also create a ServerUser object and add a conversation to its list of conversations, then verifies that the conversation is successfully added.
- testReceiveMessage: Simulates receiving a message, then verifies that the inbox count increases by 1, indicating that the message was received.
- testLoginAndLogout: Verifies the login and logout of a ServerUser object, ensuring that the user's login status is updated correctly after calling the login() and logout() methods.

ClientUser:

- LoginClientUser: Create ClientUser object, and attempting to log in and print the result
- testSetConversation: Test the setting of conversation ID for a ClientUser object by simulating message reception and printing the message content.
- testLogoutForClient: Logging the user out and printing confirmation message that user has been logged out.
- testLoadConversation : Loading conversation and printing confirmation message.

Message class :

Create a Message object with attributes such as senderID, message type and status, content , conversationID. Display the attributes of the Message class passed as an argument. Print out senderID, receiverID, message Type, content,conversation ID , and the timestamp.