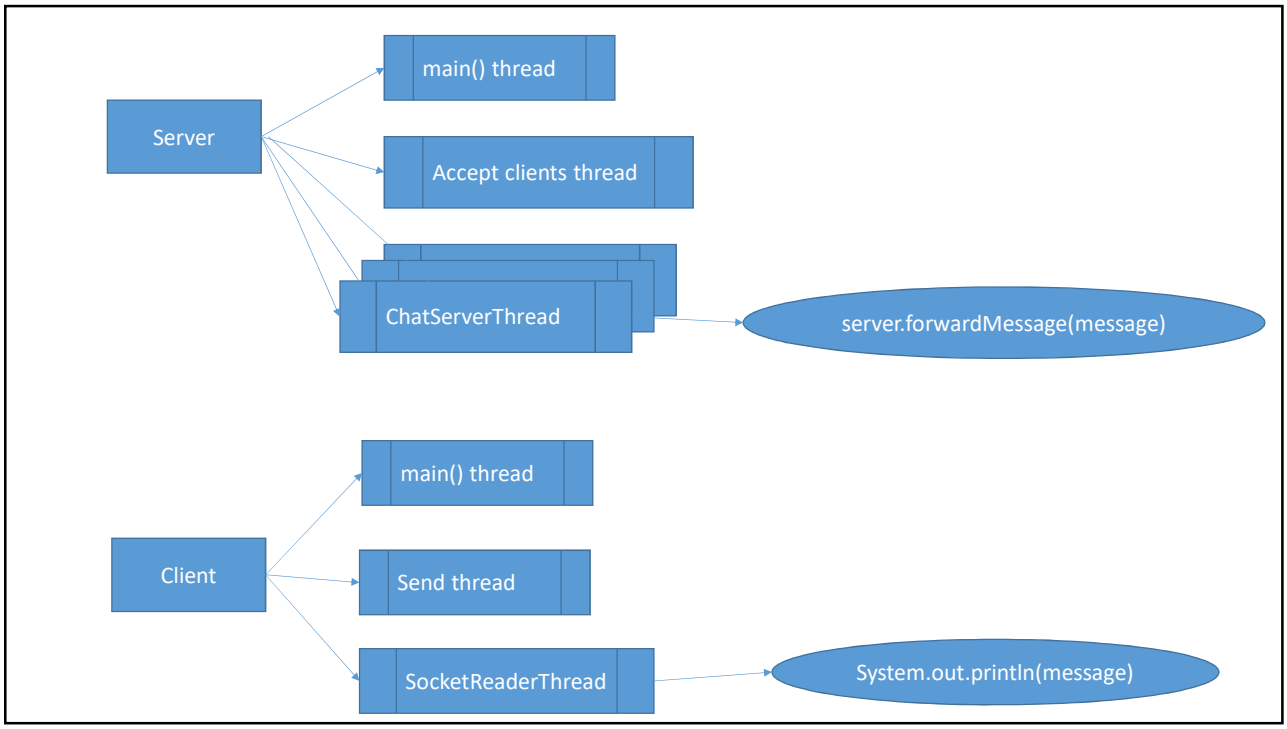
Chat Application: improvements

CSCI 2300

Latest design

- Server
  - accepts multiple clients (implemented in Server.java)
  - new thread per client (ChatServerThread)
  - each thread ‘forwards’ message to the server, then server sends ‘forwarded’ messages to all connected clients
- Client
  - one thread for sending messages (implemented in Client.java)
  - one thread for receiving messages (SocketReaderThread.java)



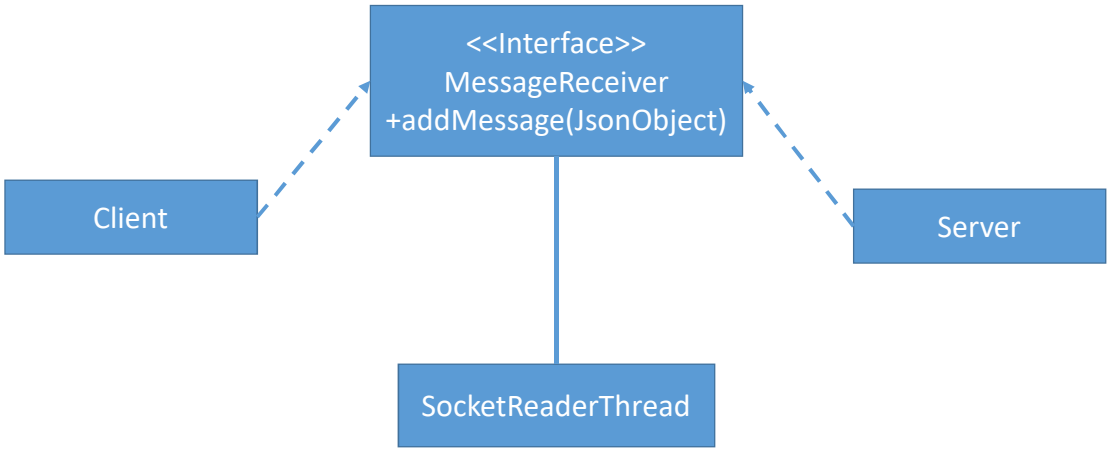
SocketReaderThread and ChatServerThread

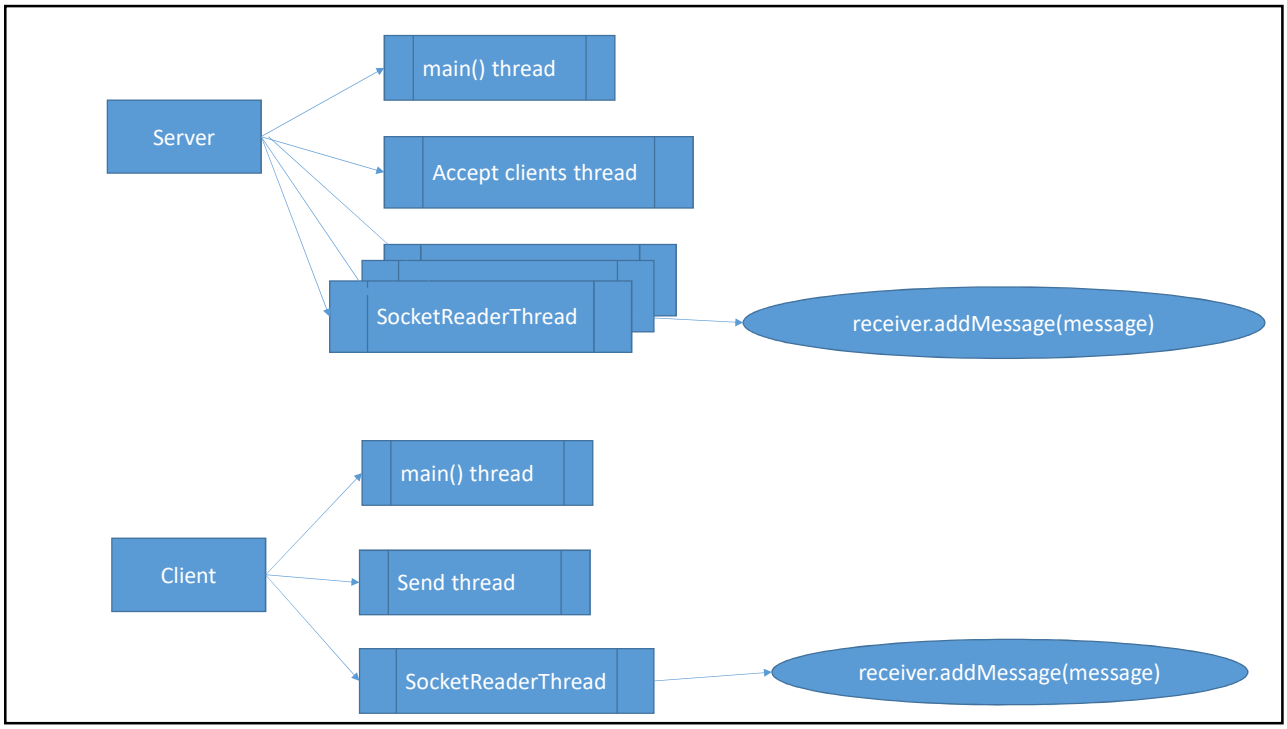
- Responsibility: Read messages from a connected socket
- SocketReadereThread – prints received messages
- ChatServerThread – prints received messages and calls `server.forwardMessage()`
- Identical responsibility, two different implementations
- Can we combine them into one class?

MessageReceiver interface

- `public void addMessage(JsonObject message);`
- `Client` implements `MessageReceiver`
  - Print the message to the screen
- `Server` implements `MessageReceiver`
  - Forward the message to all connected clients
- `SocketReaderThread`
  - has a reference to a `MessageReceiver`
  - Calls `addMessage(message)` on MessageReceiver, after reading a message from a socket

Class Diagram

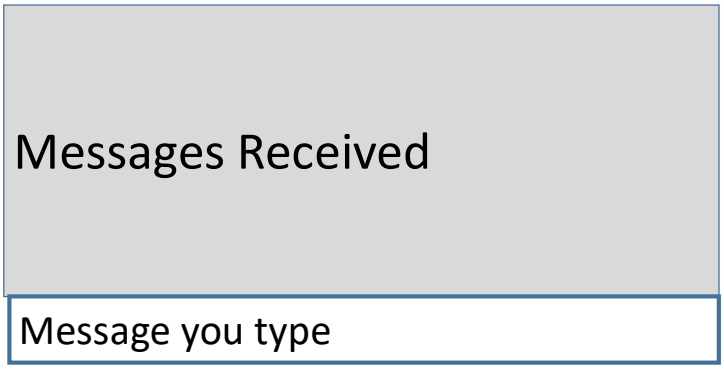




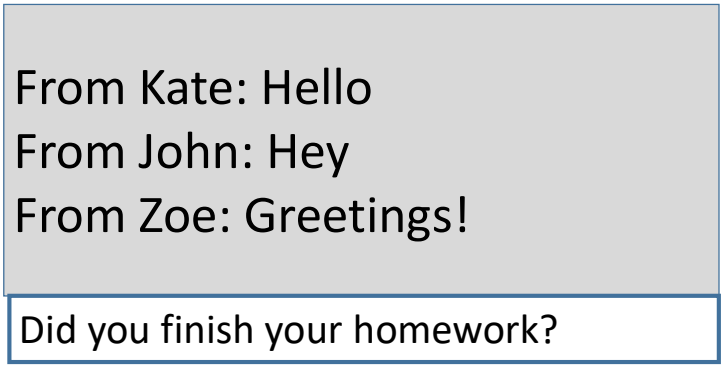
The basic_chat_improved directory

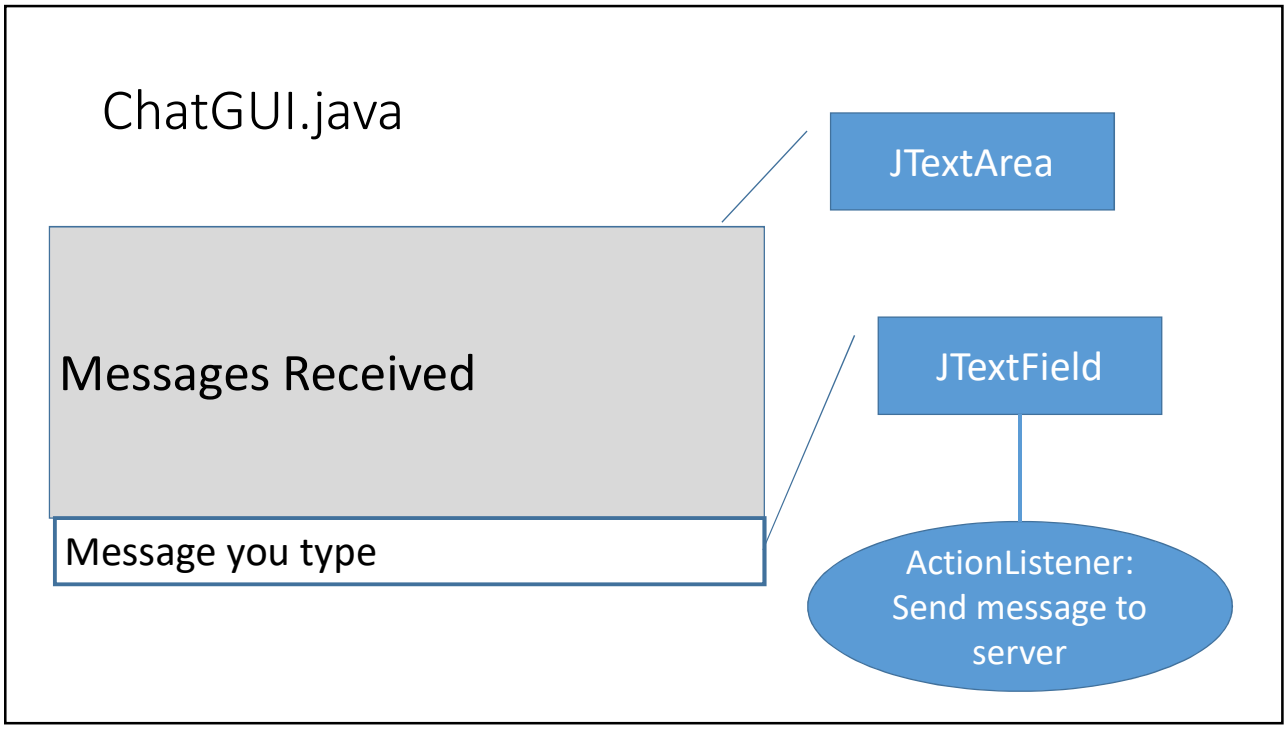
- Functionally equivalent to basic_chat directory
- Nicer design:
  - Reusing SocketReaderThread instead of duplicating code
- OO Design Principle: Don't Repeat Yourself (DRY)
- Duplication is wasteful: bug fixes need to be duplicated
- Avoid duplication by applying abstraction

Adding GUI



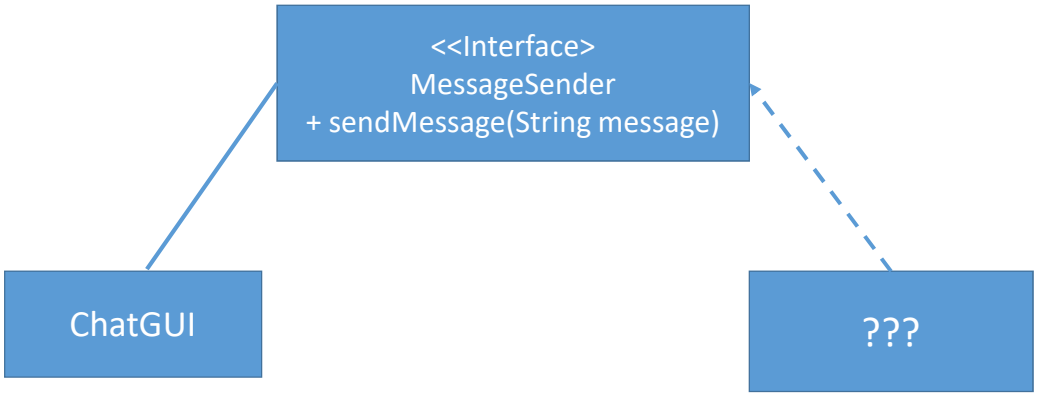
GUI Example





`MessageSender` Interface

- Another abstraction
- Define `sendMessage(String message)` method
- `ChatGUI` takes `MessageSender` in constructor
- `ActionListener` of `JTextField` calls `sendMessage()` of `MessageSender`



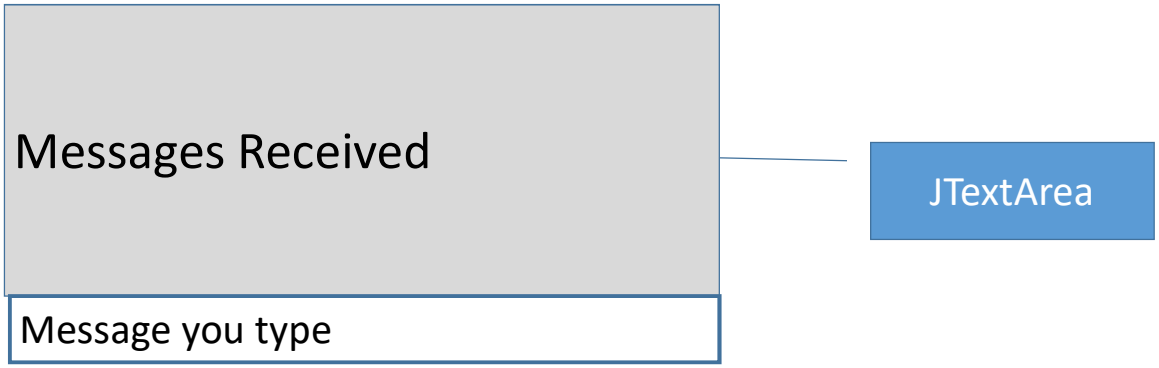
Given the class diagram above, which class should be in the box labelled with question marks?

A. Client
B. Server
C. SocketReaderThread
D. ChatGUI
E. TextMessage

Tying objects together

- `Client client = new Client(name, host_ip, port);`
- `Client` implements `MessageSender`
- `ChatGUI gui = new ChatGUI(client);`

Next: update “Messages Received”

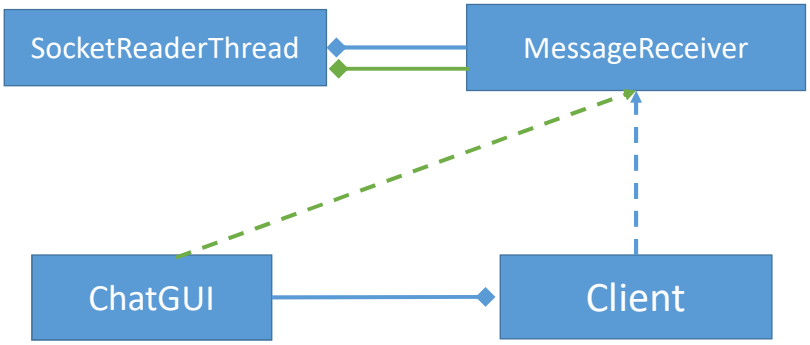


Which Interface should be used to update “Messages Received” text area

A. MessageSender
B. MessageReceiver
C. Need to define a new interface

Option 1 – blue connections

Option 2 – green connections



Which design is implemented in gui_chat directory?

A. Option 1 B. Opton 2

Final changes

- ChatGUI class
  - updateHistory(TextMessage message)
- Client's implementation of addMessage() (of MessageReceiver interface)
  - call gui.updateHistory(messageReceived)
- Client has a new method for adding ChatGUI to it:
  - addGUI(ChatGUI gui)
- ChatGUI main:
  - Client client = new Client(name, host_ip, port);
  - ChatGUI gui = new ChatGUI(client);
  - client.addGUI(gui);