Selected files

8 printable files

```
src\main\java\org\example\ChatHistory.java
src\main\java\org\example\ChatServer.java
src\main\java\org\example\IterableByUser.java
src\main\java\org\example\Main.java
src\main\java\org\example\Message.java
src\main\java\org\example\MessageMemento.java
src\main\java\org\example\searchMessagesByUser.java
src\main\java\org\example\User.java
```

src\main\java\org\example\ChatHistory.java

```java
package org.example;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.Iterator;
import java.util.Map;

//A class that stores the chat history for a user. It should have methods to add a new
//message to the history and get the last message sent
public class ChatHistory implements IterableByUser{
//    private ArrayList<String> savedMessages;
    private HashMap<String, String> savedMessages;
    MessageMemento messageMemento = new MessageMemento(savedMessages);
    public  ChatHistory(){
        this.savedMessages = new HashMap<>();
    }

    public HashMap<String, String> getSavedMessages() {return this.savedMessages;}
    public void saveMessage(String username, String message)
    {
        String newMessage;

        //get previous chat history of user and append new message to save
        if (savedMessages.get(username) == null)
        {
            newMessage = message;
        }
        else {

            newMessage = message  + "\n"+ savedMessages.get(username);
        }
        savedMessages.put(username, newMessage );


    }

    public void saveMessageMemento()
    {
        //save new memento state
        messageMemento.setState(savedMessages);
        System.out.println("********Save message memento*********");

    }
    public void undoMessageMemento()
    {
        savedMessages = messageMemento.getState();
    }

    public MessageMemento getMessageMemento()
    {
        return messageMemento;
    }

    public String printChatHistoryFromUser( String username ) {
        if (savedMessages.get(username) == null)
            return "No messages from " + username;
        else {
            return savedMessages.get(username);
        }
    }
        @Override
    public String toString()
    {
        if( savedMessages.isEmpty() )
            return "Chat History is empty";

        //getEntireChatHistory()
        String messageHistory = "";
        for( Map.Entry<String, String> m : savedMessages.entrySet() )
            messageHistory += m.getValue() + "\n";
        return messageHistory;

    }



    public Iterator iterator(User userToSearchWith) {
        return new searchMessagesByUser(userToSearchWith);
    }
}

```

src\main\java\org\example\ChatServer.java

```
package org.example;

import java.util.ArrayList;

//The mediator class that manages communication between users. It should have methods to
//register and unregister users, send messages from one user to one or more other users, and
block
//messages from specific users
public class ChatServer {
    private ArrayList<User> activeUsers, blockedUsers;
    private Message message;
```

```java
    public ChatServer()
    {
        this.activeUsers = new ArrayList<>();
        this.blockedUsers = new ArrayList<>();
    }

    public void setMessage(Message message)
    {
        this.message = message;
    }



    public void sendMessage(){

        System.out.println("Sending messages to users...");
        User sender = message.getSender();
        sender.receiveMessage(sender.getUserName(), message.getFormattedMessage());

        if(activeUsers.contains( sender ) ) {
            ArrayList<User> usersToReceiveMessage = message.getRecipients();

                for (User user : usersToReceiveMessage){
                    if(  blockedUsers.contains( user ) ) {
                        System.out.println("User is blocked from receiving messages." );
                    } else{
                        user.receiveMessage(sender.getUserName(), 
message.getFormattedMessage());
                    }
                }
        } else if ( blockedUsers.contains( sender ) ) {
            System.out.println("User is blocked from sending messages." );

        } else {
            System.out.println("Message not sent to recipients. " + message.getSender()
.getUserName() + " is not a registered user.");

        }
    }

    //can send messages
    public void registerUser(User user){
        activeUsers.add(user);
        System.out.println("Registered user " + user.getUserName());
    }

    public void unregisterUser(User user){
        activeUsers.remove(user);
        System.out.println("Unregistered user " + user.getUserName());
    }

    //can not send messages
    public void blockUser(User user){
        blockedUsers.add(user);
        System.out.println("Blocked user " + user.getUserName());
    }

    public void unblockUser(User user){
        blockedUsers.remove(user);
        System.out.println("Unblocked user " + user.getUserName());
    }
}

```

src\main\java\org\example\IterableByUser.java

```java
package org.example;

import java.util.Iterator;

public interface IterableByUser {
    Iterator iterator(User userToSearchWith);
}

```

src\main\java\org\example\Main.java

```java
package org.example;

import java.util.ArrayList;
import java.util.Iterator;

public class Main {
    public static void main(String[] args) {
        ChatServer chatServer = new ChatServer();

        User person1 = new User("Amanda");
        User person2 = new User("Alice");
        User person3 = new User("Bob");
        User person4 = new User("Eve");
        User person5 = new User("John");

        chatServer.registerUser(person1);
        chatServer.registerUser(person2);
        chatServer.registerUser(person3);
        chatServer.registerUser(person4);
        chatServer.registerUser(person5);

        ArrayList<User> person1SendTo = new ArrayList<>();
        person1SendTo.add(person2);
        person1SendTo.add(person3);
        person1SendTo.add(person4);
        person1SendTo.add(person5);
        ArrayList<User> person2SendTo = new ArrayList<>();
        person2SendTo.add(person1);
        person2SendTo.add(person3);
        person2SendTo.add(person4);
        person2SendTo.add(person5);
        ArrayList<User> person4SendTo = new ArrayList<>();
        person4SendTo.add(person1);
        person4SendTo.add(person3);
        person4SendTo.add(person2);
        person4SendTo.add(person5);

        System.out.println("Users can send messages to one or more other users through the chat server.\n=============================================================");
        Message person1Message1 = new Message(person1, person1SendTo, "15:47:14", "Hey guys, anyone want to hangout this weekend?");
        Message person1Message2 = new Message(person1, person1SendTo, "15:47:25", "Let's do something fun!");

//        person1Message1.printMessage();
        chatServer.setMessage(person1Message1);
        chatServer.sendMessage();

//        person1Message2.printMessage();
        chatServer.setMessage(person1Message2);
        chatServer.sendMessage();

        Message person2Message1 = new Message(person2, person2SendTo, "15:49:52", "I'm not available this weekend. Remove me please, thank you.");
//        person2Message1.printMessage();
        chatServer.setMessage(person2Message1);
        chatServer.sendMessage();

        System.out.println("\nUsers can block messages from specific users using the Mediator design pattern.\n=============================================================");
        chatServer.blockUser(person2);
        Message person4Message1 = new Message(person4, person4SendTo, "15:55:12", "I blocked them from the chat.");
        Message person4Message2 = new Message(person4, person4SendTo, "15:55:21", "I'm free both days.");
        Message person4Message3 = new Message(person4, person4SendTo, "15:55:25", "We should go bowling.");

//        person4Message1.printMessage();
        chatServer.setMessage(person4Message1);
        chatServer.sendMessage();

//        person4Message2.printMessage();
        chatServer.setMessage(person4Message2);
        chatServer.sendMessage();

        System.out.println("\nUsers can undo the last message they sent using the Memento design pattern.\n=============================================================");
        person4.backupMessages();
        System.out.println( person4.getEntireChatHistory().getMessageMemento());
        System.out.println( person4.getEntireChatHistory());

//        person4Message3.printMessage();
        chatServer.setMessage(person4Message3);
        chatServer.sendMessage();

        person4.undoLastMessage();
        System.out.println( person4.getEntireChatHistory().getMessageMemento());
        System.out.println( person4.getEntireChatHistory() );


        Message person4Message4 = new Message(person4, person4SendTo, "15:55:25", "We should go roller skating.");

//        person4Message4.printMessage();
        chatServer.setMessage(person4Message4);
        chatServer.sendMessage();

        System.out.println("\nUsers can receive messages from other users and view the chat history for a specific user.\n=============================================================");

        System.out.println("\nGetting entire chat history of user person5...");
        System.out.println( person5.getEntireChatHistory() );

        System.out.println("\nGetting chat history of person5 from person1...");
        System.out.println( person4.printChatHistoryFromUser( person1.getUserName() ) );

        System.out.println("\nUsers can iterate through previous sent messages from a specified user.\n============================================================");
        Iterator searchP1P3  = person2.iterator(person1);
        while( searchP1P3.hasNext() ){
            System.out.println( searchP1P3.next() );
        }
    }
}
```

src\main\java\org\example\Message.java

```
package org.example;

import java.util.ArrayList;

//A class representing a message sent by a user. It should have properties for the sender,
//recipient(s), timestamp, and message content
public class Message {
    private User sender;
    private ArrayList<User> recipients;
    private String timestamp, message;


    public Message( User sender, ArrayList<User> recipients, String timestamp, String message)
    {
        this.sender = sender;
        this.recipients = recipients;
        this.timestamp = timestamp;
        this.message = message;
    }

    public ArrayList<User> getRecipients() { return recipients;}
    public User getSender() { return sender;}
    public String getMessage() { return this.message;}
    public String getFormattedMessage() { return "<" + this.sender.getUserName()  + ">\t[" + this.timestamp + "]\t" + this.message;}
    public void printMessage() { System.out.println(this);}

    @Override
    public String toString()
    {
        String formattedMessage = "\n";

        formattedMessage += "Timestamp: " + this.timestamp + "\nFrom: <" +
this.sender.getUserName() + ">\nTo: ";
        for(User user: this.getRecipients() )
            formattedMessage += "<" +user.getUserName() + "> ";
        formattedMessage += "\n" + this.message + "\n";

        return formattedMessage;
    }
}

```

src\main\java\org\example\MessageMemento.java

```
package org.example;

import java.util.HashMap;
import java.util.Map;

//A class that represents a snapshot of a message sent by a user. It should have
//properties for the message content and timestamp
public class MessageMemento {
    private HashMap<String, String> state;

    public MessageMemento(HashMap<String, String> state)
    {
        this.state = state;
    }

    public HashMap<String, String> getState()
    {
        return state;
    }

    public void setState(HashMap<String, String> state)
    {
        this.state = state;
    }

    @Override
    public String toString()
    {
        System.out.println( "\n***************\nmessage memento state\n***************" );

        if( state.isEmpty() )
            return "empty";

        //getEntireChatHistory()
        String messageHistory = "";
        for( Map.Entry<String, String> m : state.entrySet() )
            messageHistory += m.getValue() + "\n";
        return messageHistory;

    }

}

```

src\main\java\org\example\searchMessagesByUser.java

```java
package org.example;

import java.util.ArrayList;
import java.util.Arrays;
import java.util.Iterator;
import java.util.List;

public class searchMessagesByUser implements Iterator {
    private String userName;
    private List<String> chatHistoryFromUser;
    private int indexInChatHistory;
    private int chatHistoryFromUserSize;

    public searchMessagesByUser(User userToSearch){
        this.userName = userToSearch.getUserName();

        //history in form "message\nanother message\nlast message\n"
        String allMessages = userToSearch.getEntireChatHistory().getSavedMessages()
.get(this.userName);
        String[] messages = allMessages.split("\\r?\\n|\\r");
        this.chatHistoryFromUser = Arrays.asList(messages);

        this.indexInChatHistory = 0;
        this.chatHistoryFromUserSize = chatHistoryFromUser.size();
    }

    @Override
    public boolean hasNext() {
        String message = null;

        while( indexInChatHistory < chatHistoryFromUserSize ) {
            message = chatHistoryFromUser.get(indexInChatHistory);
            if( message != null) {
                return true;
            }
            else {
                indexInChatHistory++;
            }
        }

        return false;
    }

    @Override
    public String next() {
        if( hasNext()){
            return chatHistoryFromUser.get( indexInChatHistory++ );
        }
        return null;
    }

    @Override
    public void remove() {
        Iterator.super.remove();
    }
}

```

src\main\java\org\example\User.java

```java
package org.example;

import java.util.Iterator;

//A class representing a user of the chat application. It should have methods to send messages
to
//other users, receive messages from other users, and undo the last message sent.
//NOTE: You will NOT communicate with other Users directly you will use Mediator!
public class User implements IterableByUser{
    private String userName;
    private ChatHistory chatHistory;

    public User(String userName){
        this.userName = userName;
        this.chatHistory = new ChatHistory();
    }

    public String getUserName(){ return this.userName;}
    public ChatHistory getEntireChatHistory(){return this.chatHistory;}
    public String printChatHistoryFromUser( String username ){
        return this.chatHistory.printChatHistoryFromUser(username);
    }

    public void receiveMessage(String senderUsername, String message)
    {
        chatHistory.saveMessage(senderUsername, message);
        System.out.println(this.userName + ": Message received!");
    }

    public void undoLastMessage()
    {
        chatHistory.undoMessageMemento();
        System.out.println("Last message undone!");
    }

    public void backupMessages()
    {
        chatHistory.saveMessageMemento();
        System.out.println("Messages backed up successfully!");
    }

    @Override
    public Iterator iterator(User userToSearchWith) {
        return this.chatHistory.iterator(userToSearchWith);
    }
}

```