```java
package ch19.sec07;

import java.io.IOException;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.Collection;
import java.util.Collections;
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;

import org.json.JSONObject;

public class ChatServer {
    //필드
    ServerSocket serverSocket;
    ExecutorService threadPool = Executors.newFixedThreadPool(100);
    Map<String, SocketClient> chatRoom = Collections.synchronizedMap(new HashMap<>());

    //메소드: 서버 시작
    public void start() throws IOException {
        serverSocket = new ServerSocket(50001);
        System.out.println( "[서버] 시작됨");

        Thread thread = new Thread(() -> {
            try {
                while(true) {
                    Socket socket = serverSocket.accept();
                    SocketClient sc = new SocketClient(this, socket);
                }
            } catch(IOException e) {
            }
        });
        thread.start();
    }

    //메소드: 클라이언트 연결시 SocketClient 생성 및 추가
    public void addSocketClient(SocketClient socketClient) {
        String key = socketClient.chatName + "@" + socketClient.clientIp;
        chatRoom.put(key, socketClient);
        System.out.println("입장: " + key);
        System.out.println("현재 채팅자 수: " + chatRoom.size() + "\n");
    }

    //메소드: 클라이언트 연결 종료시 SocketClient 제거
    public void removeSocketClient(SocketClient socketClient) {
        String key = socketClient.chatName + "@" + socketClient.clientIp;
        chatRoom.remove(key);
        System.out.println("나감: " + key);
        System.out.println("현재 채팅자 수: " + chatRoom.size() + "\n");
    }

    //메소드: 모든 클라이언트에게 메시지 보냄
    public void sendToAll(SocketClient sender, String message) {
        JSONObject root = new JSONObject();
        root.put("clientIp", sender.clientIp);
        root.put("chatName", sender.chatName);
        root.put("message", message);
        String json = root.toString();

        Collection<SocketClient> socketClients = chatRoom.values();
        for(SocketClient sc : socketClients) {
            if(sc == sender) continue;
            sc.send(json);
        }
    }

    //메소드: 서버 종료
    public void stop() {
        try {
            serverSocket.close();
            threadPool.shutdownNow();
            chatRoom.values().stream().forEach(sc -> sc.close());
            System.out.println( "[서버] 종료됨 ");
        } catch (IOException e1) {}
    }

    //메소드: 메인
    public static void main(String[] args) {
        try {
            ChatServer chatServer = new ChatServer();
            chatServer.start();

            System.out.println("----------------------------------------------------");
            System.out.println("서버를 종료하려면 q 를 입력하고 Enter.");
            System.out.println("----------------------------------------------------");

            Scanner scanner = new Scanner(System.in);
            while(true) {
                String key = scanner.nextLine();
                if(key.equals("q"))     break;
            }
            scanner.close();
            chatServer.stop();
        } catch(IOException e) {
            System.out.println("[서버] " + e.getMessage());
        }
    }
}
```