



---

# SPEZIFIKATION EINES CHAT PROTOKOLLS

---

Amanpreet Kaur & Till Poland  
MAY 22, 2017

RECHNERNETZE PRAKTIKUM – AUFGABEN 2 & 3

# Inhaltsverzeichnis

<b>1.</b>	<b>Systemüberblick .....</b>	<b>1</b>
1.1	ChatClient .....	1
1.2	ChatServer .....	1
1.3	ClientThread.....	1
1.4	ChatRoom .....	1
1.5	ChatMessage.....	1
<b>2.</b>	<b>Technische Aspekte.....</b>	<b>2</b>
2.1	System-Überblick.....	2
2.2	Server .....	2
2.3	ChatMessage.....	Error! Bookmark not defined.
2.2	Port.....	2
2.3	Protokoll .....	2
2.3.1	Anmeldung von Chatclient .....	2
2.3.2	Anmeldung an einem ChatRoom .....	3
2.3.3	Timeout.....	3
2.3.4	Fehlermeldungen.....	3
2.3.5	Textkodierung.....	3
<b>3</b>	<b>Technische Details .....</b>	<b>4</b>
3.2	Chatrooms.....	4
3.3	Befehle / ChatMessage Typen .....	4

## 1. Systemüberblick

Die nachfolgende Spezifikation beschreibt eine Client-Server Architektur, die zum Austausch von Nachrichten zwischen verschiedenen Benutzern benutzt werden soll. Sie setzt sich aus verschiedenen Klassen zusammen:

- ChatClient
- ChatServer
- ClientThread
- ChatRoom
- ChatMessage

### 1.1 ChatClient

Die Komponente ChatClient kann sich als Client mit dem Server unterhalten. Von ihm aus ist es dem Benutzer möglich, sich am Server und darauf bei Chatrooms anzumelden und mit anderen Clients zu chatten.

### 1.2 ChatServer

Der ChatServer lauscht auf einkommende Verbindungen von Clients und verarbeitet Nachrichten und Befehle von verbundenen Clients.

Außerdem verwaltet er die Chatrooms und das Versenden von Nachrichten an andere Benutzer.

### 1.3 ClientThread

Der ClientThread wird erstellt, sobald sich ein ChatClient beim Server anmeldet und stellt den Client auf Serverseite dar.

### 1.4 ChatRoom

Die ChatRooms sind Räume auf dem Server, an denen sich Benutzer anmelden können um mit anderen darin exklusiv zu chatten.

### 1.5 ChatMessage

Die Nachrichten, die der Server intern speichert, werden als ChatMessages gespeichert.

## 2. Technische Aspekte

Unter diesem Punkt werden die technischen Aspekte des Servers und Clients aufgelistet.

### 2.1 System-Überblick

Bei diesem System handelt es sich um eine Client-Server Architektur, die über Chaträume die verschiedenen Clients miteinander kommunizieren lässt und die geschriebenen Nachrichten als eigenes internes Format speichert.

### 2.2 Server

Der Server ist ein auf Java 1.8 basierender Server, der die Clients als einzelne Threads verwaltet.

Er hört kontinuierlich auf Socketanfragen auf seine IP und seinen Port und kommuniziert über das folgend erklärte Protokoll.

#### 2.2 Port

Der von uns benutzte Port ist der Port 50 000.

Dieser ist nach der Liste der IANA noch nicht belegt.

Quelle: [https://en.wikipedia.org/wiki/List\\_of\\_TCP\\_and\\_UDP\\_port\\_numbers](https://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers)

### 2.3 Protokoll

Vom Client aus werden folgend erklärte Nachrichten Socketstream über eine TCP Verbindung gesendet, welche in den technischen Details spezifiziert sind.

#### 2.3.1 Aufbau der Nachrichten und Antworten

Die Nachrichten, die der Server versteht sind wie folgt aufgebaut:

- Integer: Typ der Nachricht (s. Technische Details)
- Integer: Länge des zu versendenden Strings
- String in bytes: Nachricht

Diese Nachrichten beantwortet der Server wiederum folgendermaßen:

- Integer: Länge des Beantwortungsstrings
- String in bytes: Antwort

#### 2.3.2 Anmeldung von Chatclient

Zur Anmeldung am Server baut der Client eine Verbindung mit dem Server über ein Socket auf. Die Socketverbindung verläuft über die IP des gestarteten Servers (zu Testzwecken z.B. localhost) und über den oben spezifizierten Port.

- a. Darauf folgend, muss eine Initialisierungsnachricht gesendet werden, die folgendermaßen aufgebaut ist:

*Typ: Initialize, s. unten.*

*Text: Der gewählte Benutzername*

*Info: Zwei unterschiedliche Benutzer können den gleichen Benutzernamen haben!*

Danach befindet sich der Client im generellen Chatroom „General“. Von diesem aus werden Nachrichten nicht an andere Benutzer gesendet. Alle Befehle sind allerdings möglich.

b. Weitere Kommunikation

Nach der Initialisierung nimmt der Server alle der unter “Technische Details” spezifizierten Befehle entgegen und beantwortet diese sinngemäß. Dieser Austausch findet über dieselbe Socketverbindung statt, mit der auch die Initialisierung stattgefunden hat.

Geschrieben werden kann anderen Benutzern nur über neue ChatRooms, nicht über den generellen.

Alles, was der Server an den Client sendet wird als String gesendet. Darunter fallen Antworten auf Befehle, Nachrichten von anderen in einem ChatRoom und Fehlermeldungen.

Wird diese Verbindung unterbrochen, muss sich der Client auf dem Server neu initialisieren.

### 2.3.3 Anmeldung an einem ChatRoom

Der ChatRoom hat einen eindeutigen Namen, über den dieser angesprochen bzw. erreicht werden kann. In einem Chatroom können sich mehrere Clients unterhalten, sprich sie verschicken Nachrichten untereinander.

Außerdem verwaltet jeder Chatroom eine Nachrichtenliste bzw. ein Nachrichtenlog, mit den Nachrichten die in diesem Raum versendet wurden. Sprich der Raum führt ein Protokoll über die gesendeten Nachrichten und gibt dieses auch an neue Benutzer im Raum an.

Die Anmeldung an einem ChatRoom findet nach der Anmeldung am Server über den „Join“ Befehl statt.

#### 2.2.1 Timeout

Der Benutzer wird nach 5 Minuten, in der er/sie nichts geschrieben hat, automatisch abgemeldet und vom Server gelöscht.

#### 2.2.2 Fehlermeldungen

Fehlermeldungen werden wie auch alles andere vom Server als Strings an den Client gesendet, behindern aber nicht den weiteren Verlauf des Chats. Sie sind bei den Befehlen in den Technischen Details aufgelistet.

#### 2.2.3 Textkodierung

Die Nachrichten müssen in UTF-16 kodiert sein, da dies der nativen Stringkodierung von Java entspricht. Andernfalls kann der Server diese gegebenenfalls nicht korrekt darstellen bzw initialisieren.

### 3 Technische Details

In dem folgenden Abschnitt werden die Technischen Daten unseres Systems beschrieben und zusammengefasst.

#### 3.2 Chatrooms

Falls sich der letzte Benutzer aus einem Chatroom ausloggt, wird der Chatroom direkt gelöscht.

#### 3.3 Befehle / Typen

Die Befehle, die auf dem Server ausgeführt werden, werden über einen Integer Wert typisiert. Je nachdem, welcher Integer gesendet wird, wird die gesendete Nachricht anders vom Server behandelt.

Die Antwort vom Server ist immer ein String. Auch wenn es eine Fehlermeldung ist.

Die Liste der Typen sieht folgendermaßen aus:

##### **Logout - Wert: 0**

Folge: Abmeldung und Löschung von dem aufrufenden Benutzer auf dem Server.

Antwort d. Servers: Keine Antwort.

##### **Message - Wert: 1**

Folge: Die Nachricht wird an alle anderen Benutzer im momentanen ChatRoom gesendet, außer der Benutzer ist im generellen ChatRoom.

Antwort d. Servers: Keine Antwort.

##### **List Users - Wert: 2**

Folge: Der Server gibt eine Liste aller momentan angemeldeten Benutzer heraus, aufgelistet mit usernamen.

Antwort d. Servers: Liste aller angemeldeten Benutzer

##### **Join Chatroom - Wert: 3**

Folge: Der Benutzer wird aufgefordert, den Namen des beizutretenden ChatRooms einzugeben.

Antwort d. Servers: Log des ChatRooms

Fehler: ChatRoom nicht vorhanden

##### **List Chatrooms – Wert 4**

Folge: Der Server listet alle ChatRooms auf, die auf dem Server existieren

Antwort d. Servers: Liste aller ChatRooms, die auf dem Server gespeichert sind.

**Chatroom logout- Wert 5**

Folge: Der Benutzer wird aus dem momentanen Chatroom gelöscht und in den generellen ChatRoom verschoben.

Falls der Benutzer schon im generellen Chatroom war, passiert nichts.

Antwort d. Servers: Keine Antwort

**Create Chatroom – Wert 6**

Folge:

1. Von dem Benutzer wird ein Name abgefragt (Antwort d. Servers), der als Name für den Server benutzt werden soll.
2. Nach der Eingabe eines Namens wird der ChatRoom mit dem gewählten Namen erstellt und der Benutzer dem ChatRoom hinzugefügt.

Antwort d. Servers: Keine Antwort

**In Chatroom – Wert 8**

Folge: Der Server gibt den Namen des Chatrooms zurück, in dem sich der Benutzer momentan befindet.

Antwort d. Servers: Momentaner Chatroom

**Users in chatroom – Wert 9**

Folge: Der Server gibt als String eine Liste aller Benutzer zurück, die momentan in dem Chatroom sind, in dem der Benutzer sich befindet.

Antwort. d. Servers: Liste aller Benutzer, die in dem Chatroom sind, in dem der aufrufende Benutzer sich befindet.

**Initialize – Wert 11**

Folge: Wird zur Initialisierung am Server verwendet. Der Benutzer ist mit dem mitgegebenen Benutzernamen initialisiert.

Antwort d. Servers: Keine Antwort

