

# CHATPROGRAMM

Dokumentation

Erstellt von: Jacqueline Kaefer & Luca Katzenberger  
Matrikelnummer: 9847334 & 2577508  
Betreut von: Prof. Dr. Michael Bauer  
Datum: 20.03.2019

## Inhaltsverzeichnis

<b>I. Lastenheft.....</b>	<b>2</b>
1. Einleitung .....	2
2. Konzept und Rahmenbedingungen .....	2
3. Anforderungen .....	3
4. Erweiterungen.....	3
<b>II. Pflichtenheft .....</b>	<b>4</b>
1. Realisierung .....	4
2. Ausblick .....	5
<b>III. Theoretischer Hintergrund.....</b>	<b>6</b>
1. IP-Adressen und Ports.....	6
2. UDP-Protokoll .....	6
3. Threads.....	7
4. Delegat .....	7
<b>IV. Technische Dokumentation .....</b>	<b>8</b>
1. Versenden einer Nachricht .....	8
2. Codierung einer Nachricht .....	9
3. Speicherung.....	10
<b>V. Anwenderdokumentation.....</b>	<b>11</b>
1. Starten des Chatprogramms.....	11
2. Kontakte hinzufügen und löschen.....	11
3. Nachrichten Senden .....	11
4. Nachrichten empfangen.....	11
5. Speicherung.....	12
6. Fehlerbehebung.....	12
<b>VI. Literaturverzeichnis.....</b>	<b>13</b>
<b>VII. Anhang</b>	

# I. Lastenheft

## 1. Einleitung

Dieses Lastenheft beschreibt die Anforderungen für ein zu entwickelndes Chatprogramm. Dieses Programm wird im Rahmen des Moduls Softwareentwicklung als Informatikprojekt umgesetzt.

### 1.1. Abkürzungen

- UDP-Protokoll: User Datagram Protocoll
- IP-Adresse: Internet Protokoll Adresse

### 1.2. Projektteam

Rolle / Rollen	Name	E-Mail
Auftraggeber	Herr Prof. Dr. Michael Bauer	bauer@dhbw-karlsruhe.de
Projektteammitglied	Jacqueline Kaefer	jkaefer@bluewin.ch
Projektteammitglied	Luca Katzenberger	luca.katzenberger@hotmail.de

## 2. Konzept und Rahmenbedingungen

### 2.1. Nutzen des Anwenders

Der Anwender kann Textnachrichten mit anderen Kontakten austauschen. Dabei hat er die Möglichkeit, mit mehreren Teilnehmern parallel zu kommunizieren und zwischen den Chats zu wechseln.

Der Vorteil gegenüber herkömmlicher Kommunikation per Email ist, dass aus dem Chatten ein wesentlich schnelleres Antworten resultiert. Dies liegt daran, dass Chatnachrichten einen eher formlosen Charakter haben. Dadurch steigt die Kommunikationsgeschwindigkeit.

### 2.2. Benutzer / Zielgruppe

Die Anwender dieses Programms sind Mitarbeiter eines Betriebs, da dieses Programm als firmeninternes Chatprogramm zur schnellen Alternative für Emails verwendet werden soll.

Darüber hinaus kann es auch im privaten Rahmen zur Kommunikation genutzt werden, sofern sich alle Nutzer im gleichen Netzwerk befinden.

### 2.3. Systemvoraussetzungen

Für die Verwendung des Programms wird ein Windowsrechner vorausgesetzt, auf dem .NET Anwendungen verwendet werden können. Außerdem wird ein gemeinsames Netzwerk benötigt, in dem ein Broadcast über eine Broadcast IP-Adresse nicht gesperrt ist.

### 3. Anforderungen

Das Programm soll eine ansprechende und für den Benutzer selbsterklärende Benutzeroberfläche haben.

Es soll die essentielle Funktion eines Chatprogramms erfüllen, nämlich das Versenden von Nachrichten an bestimmte Personen über ein Netzwerk. Dabei sollen mehrere Chats parallel geführt werden können, sodass ein Wechsel zwischen mehreren Chatpartnern möglich ist. Die verschiedenen Chatpartner sollen in einer Kontaktliste organisiert sein.

Um die Chats auch nach einem neuen Öffnen weiterführen zu können, sollen die Kontakte und die dazugehörigen Nachrichten gespeichert werden.

Die Programmierung soll nachvollziehbar und gut strukturiert in Klassen aufgeteilt werden.

### 4. Erweiterungen

Als mögliche Erweiterung des Programms können Gruppenchats eingeführt werden. Die in diesen Gruppenchat gesendeten Nachrichten werden jedem Gruppenmitglied angezeigt und jeder hat die Möglichkeit, Nachrichten in diesen Chat zu senden.

Eine weitere Erweiterung ist das Verschicken von Anhängen, wie Bildern oder Dokumenten. Hierfür könnte eine Bytefolge statt dem Nachrichtentext übertragen werden. Da die ursprüngliche Datei nicht mehr zusammengesetzt werden kann, wenn ein Teil der Bytes fehlt, muss dabei sichergestellt werden, dass keine Daten verloren gehen.

Um dieses Problem zu lösen, kann als zusätzliche Erweiterung ein Handshake der Netzwerkteilnehmer eingefügt werden. Das bedeutet, dass ein Empfänger zurückmeldet, dass er eine Nachricht erhalten hat. Sendet er diese Rückmeldung nicht, muss die Nachricht erneut versendet werden.

## II. Pflichtenheft

### 1. Realisierung

- a) Das Programm soll eine ansprechende und für den Benutzer selbsterklärende Benutzeroberfläche haben.

Die Oberfläche des Programms wird designtechnisch an „Whatsapp“ angelehnt sein. Dies wird durch grüne Farben und die Anordnung der Elemente erreicht und wirkt dadurch ansprechend auf den Nutzer. Die Elemente der Oberfläche sind beschriftet und dadurch selbsterklärend, gerade für Personen, die bereits Erfahrung mit Chatprogrammen haben.

- b) Es soll die essentielle Funktion eines Chatprogramms erfüllen, nämlich das Versenden von Nachrichten an bestimmte Personen über ein Netzwerk.

Textnachrichten können über ein Textfeld eingegeben werden und werden beim Abschicken über das UDP-Protokoll an das Netzwerk geschickt. Die Identifikation, an wen die Nachricht gesendet werden soll, erfolgt dabei über Usernamen.

- c) Dabei sollen mehrere Chats parallel geführt werden können, sodass ein Wechsel zwischen mehreren Chatpartnern möglich ist.

Auf der linken Seite der Oberfläche befindet sich eine ListBox, über die die laufenden Chats ausgewählt werden können. Es wird immer der aktuell ausgewählte Chat angezeigt und Nachrichten werden ebenfalls an den aktuell ausgewählten Chatpartner geschickt.

- d) Die verschiedenen Chatpartner sollen in einer Kontaktliste organisiert sein.

Im Programm wurde eine Liste als globale Variable angelegt. In dieser können alle Kontakte als User-Objekte gesammelt werden. Diese Liste wird bei Änderungen immer mit der Listbox synchronisiert, sodass der Benutzer seine Kontakte verwalten und zwischen den Chats wechseln kann.

- e) Um die Chats auch nach einem neuen Öffnen weiterführen zu können, sollen die Kontakte und die dazugehörigen Nachrichten gespeichert werden.

Für die Speicherung wurde das XML-Dateiformat gewählt. Dieses ist in einem Hauptknoten und mehreren Unterknoten organisiert. Alle Knoten können Attribute und einen Inhalt enthalten. Als Hauptknoten wird der Mainuser gespeichert. Dieser enthält als Attribut den Usernamen des Anwenders. Darunter wird für jeden Kontakt ein eigener Unterknoten erstellt. Unter diesem wiederum wird für jede eingehende oder gesendete Nachricht ein Message-Knoten angelegt, der alle Attribute, sowie den Text einer Nachricht enthält. Durch diese Organisation wird das Auslesen von Kontakten und Nachrichten erleichtert.

- f) Die Programmierung soll nachvollziehbar und gut strukturiert in Klassen aufgeteilt werden.

Um die Lesbarkeit des Codes zu gewährleisten wurde der Programmcode in fünf Klassen und zwei zusätzliche Dialoge strukturiert. Diese werden teilweise objektorientiert, beispielsweise die Klassen „User“ und „Message“, und teilweise als Methodensammlungen, wie „SaveAndLoad“, verwendet.

## 2. Ausblick

Aktuell funktioniert das Chatprogramm nicht ganz zuverlässig. Durch das UDP-Protokoll werden zufällige Fehler ausgelöst, die nicht behoben werden können. Das UDP-Protokoll funktioniert ohne Handshake und versendet eine Nachricht, ohne zu überprüfen, ob der Empfänger sie auch erhalten hat und ohne zu überprüfen, ob der Inhalt auch korrekt angekommen ist. Diese Verluste können auch dazu führen, dass beispielsweise die Identifikation nicht mehr funktioniert. Dadurch kommen viele Nachrichten nicht beim Empfänger an. Eine mögliche Lösung wäre es, ein anderes Protokoll zu verwenden. Bsp. Das TCP-Protokoll. Dieses Protokoll verwendet einen Handshake, um sicherzustellen, dass die Daten vollständig ankommen. Über das TCP-Protokoll ist jedoch kein Broadcast möglich. Wodurch eine Umstellung leider nicht ohne weiteres möglich ist. Zukünftig wäre das Verwenden eines Handshakes jedoch vorteilhaft und würde das Chatprogramm deutlich zuverlässiger machen.

### III. Theoretischer Hintergrund

#### 1. IP-Adressen und Ports

Eine Internet Protokoll Adresse (IP-Adresse) ist ein eindeutiges Identifizierungsmerkmal und wird in einem Netzwerk an nur ein einziges Gerät vergeben. Die meist verwendete IP-Adresse-Version-4 (IPv4) besteht aus vier Bytes.

Bsp: 192 . 168 . 000 . 002

Man unterscheidet bei der Vergabe von IP-Adressen zwei Verfahren. Bei der statischen IP-Vergabe wird einem Gerät eine feste Adresse zugeordnet, unter der dieses sich immer ins Netzwerk einwählt. Im Gegensatz dazu steht die dynamische IP-Vergabe. Bei dieser Methode wird einem Gerät, das sich ins Netzwerk einwählen möchte, eine gerade freie IP-Adresse zugewiesen. Somit kann sich die IP-Adresse eines Geräts mit jedem Einwählen in ein Netzwerk ändern.

Neben den IP-Adressen für konkrete Geräte, gibt es in einem Netzwerk spezielle IP-Adressen, die für einen Broadcast verwendet werden können, wie zum Beispiel 255.255.255.255. Ein Broadcast verschickt die Daten an alle eingewählten Geräte eines Netzwerks.

Ein Port funktioniert als Schnittstelle zwischen einem Computer und dem Netzwerk, in welchem er sich befindet. Dabei wird über einen Port das jeweilige Datenpaket weitergegeben. Portadressen liegen im Bereich von 0 bis 65535, wovon die Adressen 0 bis 49151 für spezielle Funktionen oder Programme, wie Email-Programme, reserviert sind. Die restlichen Ports von 49152 bis 65535 können frei verwendet werden. Somit kann in einem Programm einer dieser Ports verwendet werden, um Daten auszutauschen. Dadurch werden auch nur diejenigen Datenpakete vom Programm ausgewertet, die auf dem verwendeten Port eingehen.

#### 2. UDP-Protokoll

Das User Datagram Protocol (UDP-Protokoll) ist ein verbindungsloses Protokoll. Das bedeutet, dass vor dem Versenden von Daten keine Verbindung zwischen Sender und Empfänger aufgebaut wird. Dadurch ist es unsicher bei der Übertragung von Datenpaketen, da der Sender eines Datenpaketes nicht weiß, ob dieses beim Empfänger angekommen ist.

Insgesamt ist es ein sehr schlankes Protokoll und enthält keinerlei Kontroll- und Datenmanagementfunktionen. Es unterstützt aber die Kommunikation über Ports. Somit ist ein leichter Verbindungsaufbau möglich.

### 3. Threads

Ein Thread ermöglicht es in einem Programm Funktionen nebenläufig auszuführen. Dies bedeutet, dass das Programm quasiparallel mehrere Aufgaben ausführen kann. Dabei wird von quasiparallel gesprochen, da der Computer schnell zwischen den einzelnen Aufgaben hin und her wechselt, sodass es von außen parallel erscheint. In Wirklichkeit arbeitet der Computer aber nicht parallel.

### 4. Delegat

Delegaten ermöglichen es, einer Methode als Argument eine Methode mitzugeben. Dabei wird dem Delegaten bei der Initialisierung eine Methode mit einer bestimmten Parameterliste und einem Rückgabewert zugewiesen. Dieser Delegat kann dann als Argument einer Methode verwendet werden. Beispielsweise werden Delegaten verwendet, um einem Thread eine Methode zuzuweisen, die vom Thread ausgeführt wird.



## IV. Technische Dokumentation

### 1. Versenden einer Nachricht

Die Identifikation der User in diesem Chatprogramm erfolgt über Usernamen. Die Information an welchen User eine Nachricht geschickt wird und von welchem User sie kommt, wird in einer speziellen Codierung codiert (siehe 0.

Speicherung).

erhalten, die mit diesem Chatprogramm verschickten Nachrichten, werden an alle Computer eines Netzwerkes gesendet. Jedoch werden nur die Computer die Nachrichten aus, die auch auf den entsprechenden Port achten. Normalerweise ignoriert ein Computer alle eingehenden Signale aus einem Netzwerk, die nicht auf einem der Standardports eingehen. Läuft dieses Chatprogramm, wird der zugewiesene Port 54546, auf dem die Nachrichten gesendet werden, überwacht und die ankommenden Nachrichten ausgewertet.

Jeder User des Chatprogramm erhält also im Hintergrund alle Nachrichten. Auch Nachrichten, die er selbst abgeschickt hat, oder die nicht an ihn geschickt wurden. Das Programm prüft nach Erhalt einer Nachricht, ob die Nachricht für den User bestimmt ist. Ist sie für den User bestimmt oder vom User abgeschickt worden, wird die Nachricht weiterverarbeitet, wenn nicht wird sie ignoriert.

Diese Methode hat zwei größere Nachteile:

- Die Nachrichten können relativ einfach auch von dritten mitgelesen werden, für die die Nachrichten nicht bestimmt sind.
- Existiert ein Username doppelt, können die Nachrichten nicht mehr eindeutig zugeordnet werden. Dies kann passieren, da die Usernamen nicht auf Einzigartigkeit geprüft werden.

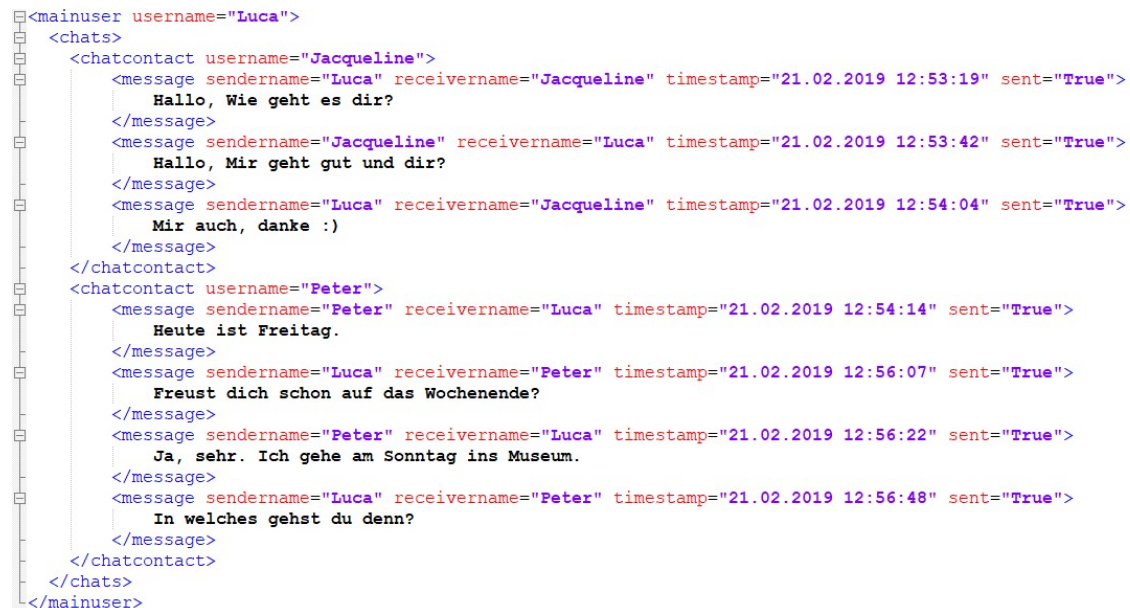
## 2. Codierung einer Nachricht

Die Informationen, die eine Nachricht charakterisieren, werden in einem String zusammengefasst und zwischen den einzelnen Informationen ein spezielles Trennzeichen („\$%&“) eingefügt. Dieser String kann anschließend verschickt werden. Da das Trennzeichen und die Reihenfolge der Attribute bekannt sind, können die Informationen vom Empfänger auch wieder decodiert werden.

### 3. Speicherung

Alle wichtigen Daten des Programms werden in einem XML-File gespeichert. Das XML-File wird beim Anlegen des eigenen Usernamens erzeugt und mit dem Namen „Backupfile.xml“ im gleichen Ordner wie die .exe Datei gespeichert.

Die XML-Datei ist in mehreren Knoten und Attributen (Vgl.: Baumstruktur) organisiert. Der Aufbau dieser Datei ist in **Fehler! Verweisquelle konnte nicht gefunden werden.** zu sehen.



```
<mainuser username="Luca">
  <chats>
    <chatcontact username="Jacqueline">
      <message sendername="Luca" receivername="Jacqueline" timestamp="21.02.2019 12:53:19" sent="True">
        Hallo, Wie geht es dir?
      </message>
      <message sendername="Jacqueline" receivername="Luca" timestamp="21.02.2019 12:53:42" sent="True">
        Hallo, Mir geht gut und dir?
      </message>
      <message sendername="Luca" receivername="Jacqueline" timestamp="21.02.2019 12:54:04" sent="True">
        Mir auch, danke :)
      </message>
    </chatcontact>
    <chatcontact username="Peter">
      <message sendername="Peter" receivername="Luca" timestamp="21.02.2019 12:54:14" sent="True">
        Heute ist Freitag.
      </message>
      <message sendername="Luca" receivername="Peter" timestamp="21.02.2019 12:56:07" sent="True">
        Freust dich schon auf das Wochenende?
      </message>
      <message sendername="Peter" receivername="Luca" timestamp="21.02.2019 12:56:22" sent="True">
        Ja, sehr. Ich gehe am Sonntag ins Museum.
      </message>
      <message sendername="Luca" receivername="Peter" timestamp="21.02.2019 12:56:48" sent="True">
        In welches gehst du denn?
      </message>
    </chatcontact>
  </chats>
</mainuser>
```

Abbildung 1: Sicherungsdatei

Die Datei enthält für jeden Chatkontakt einen Chatcontact-Knoten. An diesen wird für jede Nachricht ein Message-Knoten angehängt. Die Informationen zu den Kontakten und Nachrichten sind dabei in den Attributen gespeichert. Dies macht das Einfügen von neuen Kontakten und Nachrichten einfach, da lediglich ein neuer Knoten mit den jeweiligen Attributen erzeugt und dieser an den entsprechenden Oberknoten angehängt wird.

Durch die Baumstruktur können Informationen gezielt ausgelesen werden.

## V. Anwenderdokumentation

Die Hauptfunktion, die das Programm bietet, ist wie der Name schon sagt, das Chatten mit anderen Kontakten. Im Folgenden ist beschrieben, wie der User das Programm verwendet.

### 1. Starten des Chatprogramms

Wird das Programm zum ersten Mal gestartet, muss ein Username eingegeben werden. Dieser muss zwischen 3 und 20 Zeichen lang sein und kann im Nachhinein nicht mehr geändert werden. Bei allen folgenden Programmstarts werden alle Daten automatisch geladen.

### 2. Kontakte hinzufügen und löschen

Der Benutzer hat zunächst die Möglichkeit über den Button „Kontakt hinzufügen“ einen neuen Kontakt in seiner Kontaktliste zu erstellen. Hierfür muss er lediglich dessen Benutzernamen in das angezeigte Dialogfeld eingeben. Die Anzahl an Kontakten ist nicht beschränkt. Ein hinzugefügter Kontakt kann auch über den Button „Kontakt löschen“ wieder entfernt werden. Hierdurch werden jedoch auch alle bisher mit diesem Kontakt ausgetauschten Nachrichten unwiderruflich gelöscht.

### 3. Nachrichten Senden

Es können nur Textnachrichten versendet werden, die jedoch nicht in der Zeichenanzahl beschränkt sind. Um eine Nachricht zu verschicken, muss der Text zunächst in die große TextBox rechts unten eingegeben und der Kontakt, an den die Nachricht gesendet werden soll ausgewählt werden. Dafür muss auf den entsprechenden Benutzernamen in der linken Liste geklickt werden. Anschließend kann die Nachricht wahlweise über den Button „Senden“ oder das Drücken der ENTER-Taste abgeschickt werden. Anschließend wird sie im Nachrichtenverlauf oberhalb der TextBox dargestellt. Es ist nicht möglich Nachrichten zu verschicken, ohne dass ein Kontakt ausgewählt ist.

### 4. Nachrichten empfangen

Erhält der Benutzer eine Nachricht, wird er durch einen Benachrichtigungston und ein Popup darauf hingewiesen. Dem Text des Popups kann er direkt entnehmen, von welchem seiner Kontakte die Nachricht gekommen ist. Ist dieser Kontakt gerade ausgewählt, wird die empfangene Nachricht auch sofort angezeigt.

Sollte der Sender einer Nachricht nicht in der Kontaktliste des Benutzers sein, erscheint ein Dialogfenster, das ihn auf die neue Nachricht hinweist. Der Benutzer kann sich daraufhin entscheiden, ob er den Sender, als neuen Kontakt, seiner Kontaktliste hinzufügen will, oder ob die Nachricht ignoriert werden soll.

## 5. Speicherung

Empfange oder gesendete Nachrichten werden immer sofort in einem Backupfile gespeichert. Dadurch ist es möglich, dass mehrere Chats parallel geführt werden können und alte Nachrichten immer im Verlauf angezeigt werden. Darüber hinaus sind die Nachrichten auch nach dem Schließen und erneuten Öffnen des Programms noch verfügbar und werden nach dem Auswählen eines Kontakts im Verlauf dargestellt. Alle Daten werden in einer Datei namens Backupfile gespeichert. Dieses sollte deshalb nicht gelöscht werden.

## 6. Fehlerbehebung

Erscheinen während der Benutzung des Programms Fehlermeldungen, sollte der Benutzer den angezeigten Anweisungen folgen. Sollten unvorhergesehene Fehler auftreten, kann es helfen, das Backupfile dennoch zu löschen, wodurch das Programm auf Werkseinstellungen zurückgesetzt wird. Das bedeutet auch, dass alle Nachrichten und Kontakte gelöscht werden.

## VI. Literaturverzeichnis

<https://de.ryte.com/wiki/IP-Adresse>

<https://de.wikipedia.org/wiki/User-Thread>

<https://docs.microsoft.com/de-de/dotnet/csharp/programming-guide/delegates/using-delegates>

[https://support.biamp.com/Audia-Nexia/Control/TCP\\_and\\_UDP\\_discovery\\_methods](https://support.biamp.com/Audia-Nexia/Control/TCP_and_UDP_discovery_methods)

<https://www.dreamincode.net/forums/topic/231058-peer-to-peer-chat-advanced/>

<https://www.ip-insider.de/was-ist-ein-netzwerk-port-a-691212/>

[https://www.uni-weimar.de/kunst-und-gestaltung/wiki/TCP/IP\\_UDP](https://www.uni-weimar.de/kunst-und-gestaltung/wiki/TCP/IP_UDP)

Hartmut Ernst, Jochen Schmidt, Gerd Beneken:      Grundkurs Informatik,  
Grundlagen und Konzepte für die Erfolgreiche IT-Praxis – Eine  
umfassende, praxisorientierte Einführung, Springer, 6. Auflage, Wiesbaden  
2016