Kommunikasjon: Tjenester og Nett
TTM4100

Chat - Klient Server

Authors:
Andreas Thyholt Henriksen

March 10, 2017

NTNU – Trondheim
Norwegian University of
Science and Technology

Table of content

1 Introduksjon

Dette prosjektet har som mål å lage en chatte-tjeneste basert på en felles protokoll for alle prosjekter, og server-klient. Til syvende og sist vil suksess avgjøres ved en SAT-test på lab. Brukergrensesnittet vil være svært enkel og minimal via cmd-line, og støtte tjenester som <login>, <logout>, <help>, <msg> osv. Alle typer brukere med litt insikt i sosiale medier skal være i stand til å benytte seg av tjenesten. Alle brukere er å anse som klienter, mens chatte-tjenesten leveres av serveren.

All kommunikasjon vil foregå med treveis "handshake", vha TCP. Protokoll for kommunikasjon mellom server-klient er spesifisert i oppgaveteksten, og koding av meldinger vil gjøres vha JSON. Serveren må være i stand til å håndtere et større antall klienter, lagre alle meldinger i sortert rekkefølge, tilby som nevnt en login/logout tjeneste, og gi klientene er liste over alle klienter for å nevne noen. Minimum til tjenester fra server står spesifisert i oppgaveteksten. Sett bort i fra håndtering av brukerfeil og feilmelding hos klientene dersom server krasher vil det ikke bli implementert noe form for redundans eller den slags feil-håndtering. Dersom serveren krasher går all informasjon tapt.

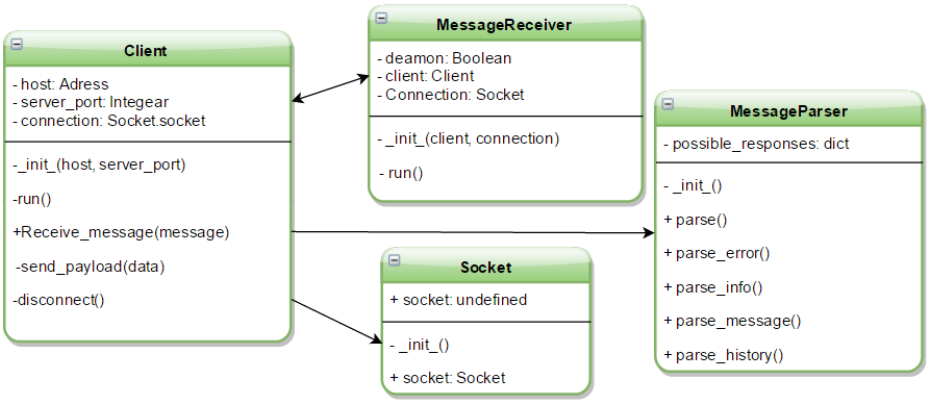
2 Klient

2.1 Formål og oppbygning

Alle brukere som ønsker å benytte seg av chatten, vil tjenes av klienten. Sådan er klienten er tjener for brukeren. I systemet derimot vil klienten vil tjent av serveren. Dermed er klienten nødt til å være i stand til å initiere og opprettholde kontakten med serveren. I tillegg må klienten være i stand til å motta meldinger sendt fra server.

Dette skaper naturlig et behov for to hovedtråder på basis av to ulike objektklasser. Den en objektklassen er <MessageReciver>, hvilket konstant lytter til beskjeder sendt fra server, mens den andre er <Client>, hvilket konstant lytter til input fra brukeren. I tillegg er det behov for en siste klasse, hvilket formaterer meldinger mottatt fra server. <MessageParser> fyller dette behovet, og vil bli benyttet av <Client> etter at en melding er mottatt fra <MessageReciver>. En mer utfyllende beskrivelse av hver klasses funksjonalitet gies i 2.3 etter systemets klassediagram 2.2 nedenfor.

2.2 Klassediagram



2.3 Funksjonalitet

2.3.1 <Client>

Klientens hovedklasse! Denne klassen lagrer nødvendig informasjon og håndterer alle handlinger knyttet til server. Alle meldinger fra klient til server sendes direkte herfra. I tillegg tjener denne klassen brukerene av chatte-tjenesten, gjennom cmd-line.

2.3.2 <MessageReciver>

Døra inn til klienten! All kommunikasjon fra server til klient går gjennom denne klassen, som aktiv lytter etter meldinger ved å kjøre som en tråd. Mottatte meldinger sendes videre til <Client>.

2.3.3 <MessageParser>

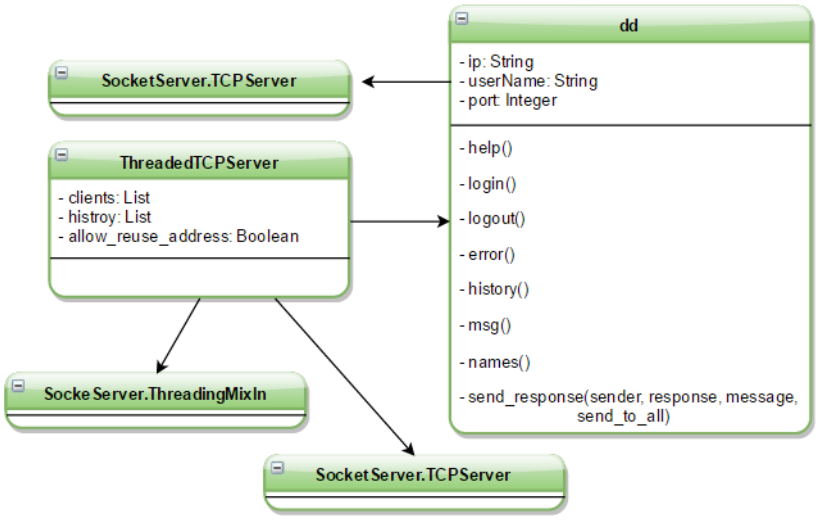
Funksjonaliteten ligger i navnet! <Client> benytter seg av denne klassen til å formatere meldinger mottatt fra server til et brukervennlig format som kan presanteres for brukeren.

3 Server

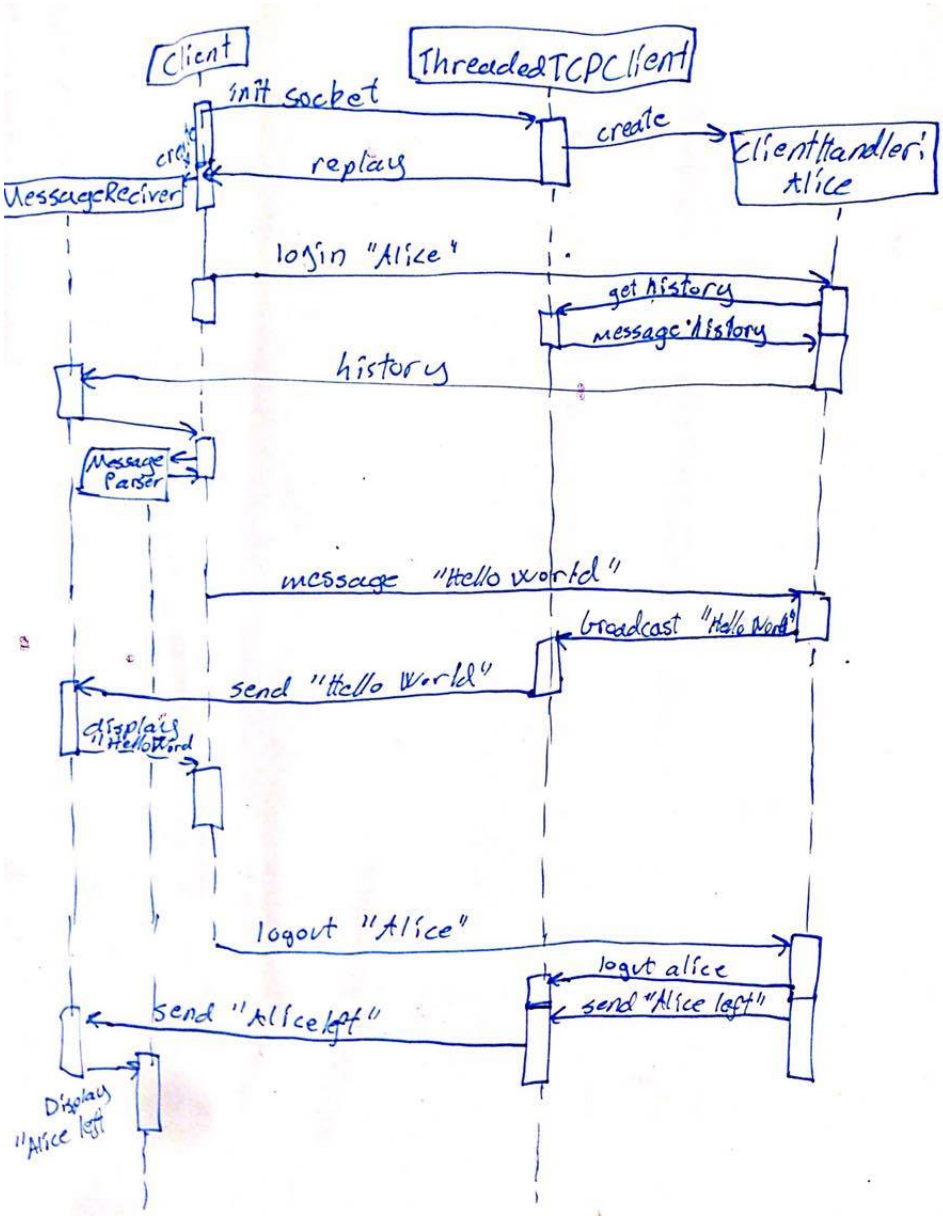
3.1 Formål og oppbygning

Serverens formål vil være å kjøre hele chatte-tjenesten. Den lytter alltid etter nye klienter som ønsker å koble seg til, og tjener deretter enhver ny klient. Klasssen <ThreadedTCPServer> utvider server fra python-biblioteket, og håndterer alle nye oppkoblinger mellom server og klient. Klassen kjører som en tråd, og lytter kostant. Hver gang serveren får en ny forespørsel fra en klient som ønsker å koble seg til, starter serveren en ny klasse av type <ClientHandler> tilegnet denne klienten. <ClientHandler> lagrer nødvendig informasjon, og betjener alle nye pakker send fra klienten. Denne betjeningen skjer ved at listen over handlere for alle typer pakker traverseres, og riktig handler kalles.

3.2 Klassediagram



4 Sekvensdiagram



5 Verktøy

Etter oppfording i oppgaveteksten inkluderer jeg også en kort beskrivelse av ulike verktøy benyttet til dette prosjektet.

5.1 Vmware Player m. Ubuntu

Virituel maskin benyttes for å kunne kjøre Ubuntu ovenpå Windows. Dette gir mange fordeler til vanlig, og gjør det likedan også for dette prosjektet.

5.2 Git og Github

Versjonskontroll-systemet benyttet til hovedsakelig del to, KTN2, av prosjektet. Alle filer med unntak av latex-filene vil bli lastet opp her. Jeg unnlater å lase opp latex-filene for å hindre fosskok.

5.3 Sublime

Tekst-redigeringsverktøy benyttet i dette prosjektet, med dvs utvidelser.

5.4 Sharelatex

Nettbasert tekst-redigeringsvertøy basert på Latex, brukt for del en KTN1 av prosjektet.