CA1 Dokumentation

Kort beskrivelse af designvalg(Søren)

Vi har valgt et design hvor at hver del af projektet er delt op i tre pakker. En Chatclient, en Chatserver og en Webserver. Fordelene ved at dele det op, er at hver del fungerer uafhængigt af de andre dele i den forstand at en del kan udskiftes uden at de to andre skal modificeres. Desuden kan projektet skaleres op langt nemmere fordi at webserveren og chatserveren ikke behøves hostes på samme server. Var programmet kodet således at webserveren fx startede chatserveren ville presset på den ene maskine hvor "begge servers" står blive presset mere end hvis de var delt ud på to.
Chatclienten er forsøgt designet efter gang of four's observer pattern. Således at GUI implementere et observerinterface og registrere sig selv som lytter. Og hver gang at Clienten modtager en beksed bliver lyttere notificeret og kan håndtere data som de har lyst.

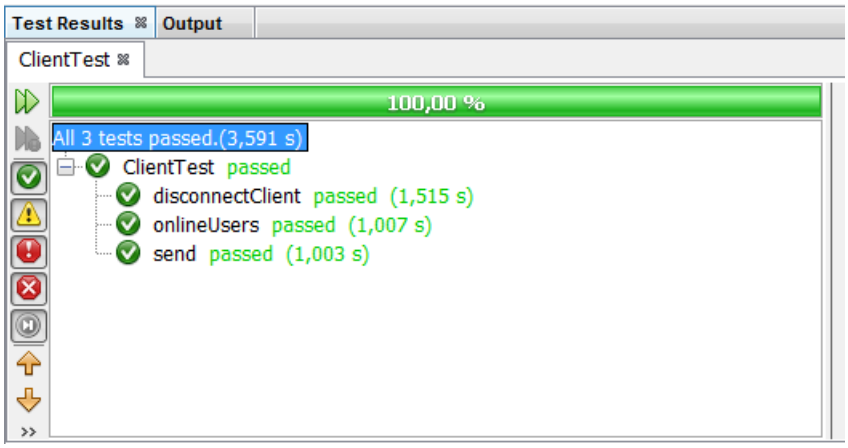
Hvem har lavet hvad(Søren)

Som beskrevet før har vi delt projektet op i tre dele. Og der var også sådan vi startede med at kode. Vi startede op med at kode hver vores del og benyttede hen ad vejen git uden problemer, da vi arbejde i hver vores pakke. Efter grundarbejdet var kodet og projektet virkede i store træk, begyndte vi at arbejde sammen om de sidste problemer.
Vi har delt projektet således:

Chatserver er lavet af Morten
Webserver er lavet af Stefan
Chatclient er lavet af Søren

Derefter har vi udarbejdet vores test-class i fællesskab(Pair programming?), og lavet en metode hver.

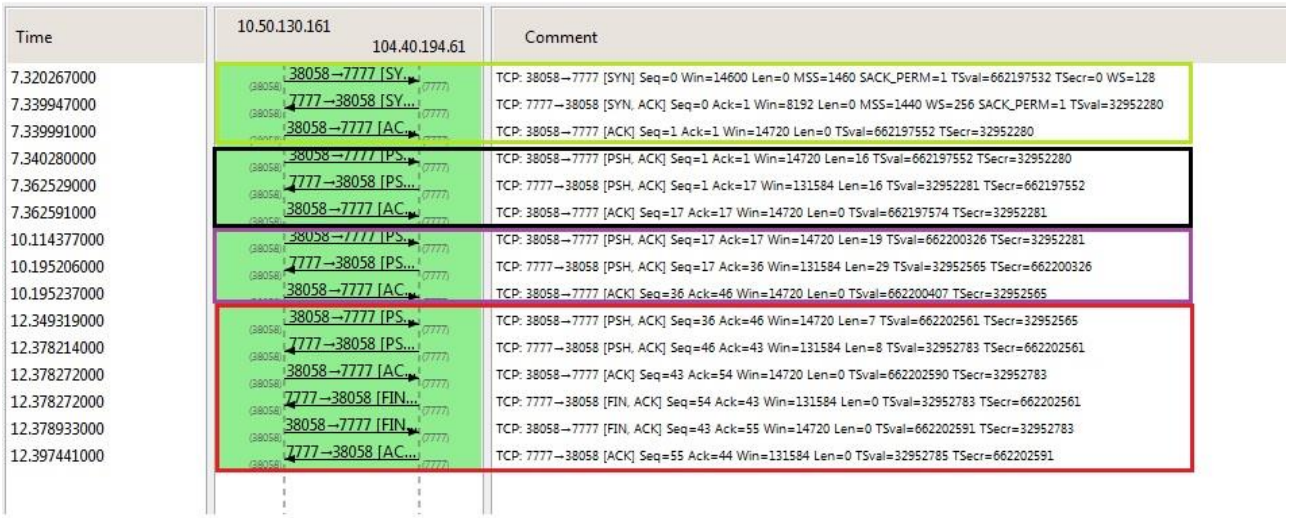
JUnit test



State behaviour(Stefan)

Vores chatserver implementation benytter sig af 1 tråd som står og lytter på den givede port, når den så modtager en forbindelse på den port. Den laver så en ny handler som tager username og socket. Derefter gemmes denne nye handler i et hashmap også startes den nye handler i en ny tråd. Derefter vil server tråden fortsætte med at lytte efter nye forbindelser.

Wireshark sample(Morten)



Vi har brugt Wireshark til at sniffe de TCP pakker, det bliver sendt mellem vores Chatserver og vores klient. Det første der sker er, at der bliver lavet et ”Three way handshake” som kan ses i det grønne felt. Det består af at clienten sender et [SYN] og forventer at få et svar fra serveren med [SYN, ACK]. Til sidst sender client et [ACK] hermed at forbindelsen oprettet.

I det sorte felt, sender vi en besked med Connect#Username [PSH, ACK] som server forventer som det første at få. Serveren sender et Online#users[PSH, ACK] tilbage. Tilsidst sender client en [ACK] på at beskenden er modtaget.

Det røde felt:

Der sendes et CLOSE# command, fra clienten til serveren. Herefter sender Serveren et CLOSE# tilbage. Herefter sender [FIN, ACK] fra serveren til client, hvis clienten er færdig med at sende og klar til at lukke sender den en [FIN, ACK] tilbage. Server sender en [ACK] når den har modtaget [FIN, ACK] fra clienten.