Dokumentation af CA-1 Chat server / Client + Project Web Server.

Af: Andreas Heindorff Larsen, Christian Lind & Jonas Simonsen

På baggrund af det givne oplæg har vi formået at lave at lave vores CA, som er end ud med et, syntes vi selv, ganske glimrende funktionelt resultat.

Beskrivelse af vores design:

Vores program tager udgangspunkt i designet fra Echo projektet udleveret af Lars. Vi har dog implementeret en UserHandler klasse, der står for at håndhæve at de forskellige beskeder fra ChatClient til ChatServer overholder de givende protokoller, samt definere hvad de enkelte input skal returnere, når de bliver sendt til vores ChatServer.
Programmet består af 2 sider. Vores client side, der indeholder ChatClient og View, og vores server side, der består af ChatServer og UserHandler. Idéen med vores projekt er at vores client skulle kunne oprette samt skrive til andre gruppers servere, og at andre gruppers clienter skulle kunne oprette og skrive til vores server. Vi opnår dette ved at overholde og definere de udgivende protokoller i vores ProtocolStrings klasse, og sørge for at Vores ChatClient og ChatServer er uafhængige af hinanden.
I vores GUI kan vi ved hjælp af at extende Observable implementere en update metode, der konstant opdatere vores UserList og jTextArea1 uden en event eller lignende.
Neden for et eksempel på en defenitionen af vores protokol ses:

```
public class ProtocolStrings {

    public static String STOP = "STOP#";

    public static String connectUser(String name) {
        return "USER#" + name;
    }
```

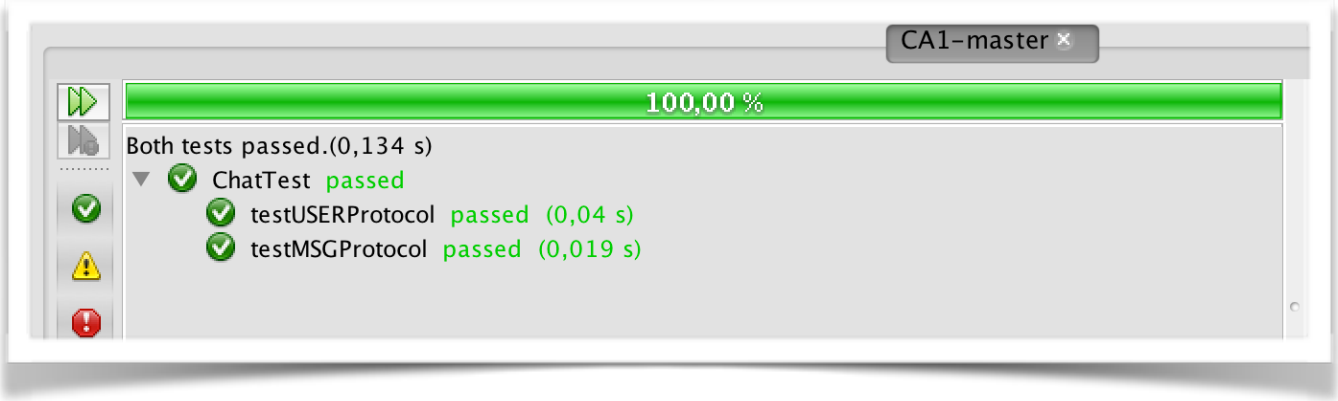
Hvem har lavet hvad:

I gruppen har vi alle været med til at sørge for at vores klient og server virker efter hensigten, dog har Andreas stået for væsentligt flere betydelige commits til GitHub idet rigtig meget af klient og server er laver fra hans computer. Vi valgte at lave store dele af klienten og serveren fra en computer, for at undgå merge konflikter og andre sjove ting på

GitHub. Det er hovedsagligt Andreas og Christian som har stået for GUI'en og dens funktionalitet. Jonas har stået for vores hjemmeside med hvad der hører til af sikkerhed. Upload til Azure har vi alle arbejdet med, da vi syntes det foregik på en lidt mærkelig måde. Især sikkerhedsdelen på websiden havde vi vores udfordringer med. Vi havde og vores udfordringer med hvordan vi fik startet serveren efter den var lagt op på azure, men vi fandt løsninger på begge problemstillinger. Dokumentationen er udarbejdet af hele gruppen så alle er indforstået med hvad denne indholder.

Resultater af JUnit test:

Vi har haft store problemer med vores JUnit test, men det er lykkedes at lave 2 test som viser funktionaliteten af serveren uafhængig af klienten. For at komme uden om klienten benytter vi os af sockets til at oprette forbindelse til chat serveren og af printWriter metoden til at skrive til serveren, efterfølgende benytter vi os af en scanner som scanner den linje serveren svare tilbage på, slutteligt bruger vi så metoden assertEquals til at se hvorledes det resultat er identisk med det forventede resultat. Neden for kan testresultaterne af vores test ses:



Implementering af protokollen:

Protokollen har selvfølgelig haft stor indflydelse på hvordan det endelige program er kommet til at se ud. Vi har overholdt protokollen således at det kun er de definerede kommandoer som kan benyttes. Der sker dog ikke noget ved at man indtaster noget andet, brugeren får ikke en fejlmeddelse tilbage. Såvel som at vi heller ikke tjekker om brugernavn allerede eksisterer. For at logge ind skal kommandoen USER# benyttes, når brugeren er logges ind, tilføjes denne til et array som indeholder alle allerede eksisterende online users, efter login sendes der en opdateret userList til alle online brugeren. Efter login har brugeren mulighed for at benytte sig af to kommandoer MSG# og STOP#.

Benytter man sig af MSG# kommandoen har brugeren mulighed for at chatte med andre online brugere. Hvem den specifikke besked skal sendes til håndteres af serverens send metode. Når beskeden er sendt vil modtageren efterfølgende kunne se beskeden i den aftalte syntaks hvis telnet benyttes som klient. Hvis syntaksen ikke er overholdt fra afsender vil beskeden ikke gå igennem til modtageren.
Benytter man kommandoen STOP#, bliver afsenderens socket lukket og vi kalder metoden removeHandler. Metoden removeHandler fjerner vores user og sender derefter en opdateret userList ud til alle online brugere.