S04-chat

Schober, Wortha 4AHIT

22.11.2014

Inhalt

Inhalt 1

Angabe 2

Zeit 3

Zeitabschätzung 3

Zeitaufstellung 3

UML-Diagramm/Designüberlegung 4

Things I've done 4

Testing 6

# Angabe

Erstellt ein einfaches Chat-Programm für "Schwerhörige", mit dem Texte zwischen zwei Computern geschickt werden können.  
  
Dabei soll jeder gesendete Text "geschrien" ankommen (d.h. ausschließlich in Großbuchstaben, lächelnd wird zu \*lol\*, Buchstaben werden verdoppelt, … - ihr dürft da kreativ sein)  
  
Zusätzlich sollen "böse" Wörter ausgefiltert und durch "$%&\*" ersetzt werden. Diese Funktionalität soll aber im Interface jederzeit aktiviert und deaktiviert werden können.  
  
Verwende dafür ausgiebig das Decorator-Pattern.

# Zeit

## Zeitabschätzung

Wir schätzen mit einem Arbeitsaufwand von circa 2,5 Stunden pro Person. Also insgesamt 5 Stunden.

## Zeitaufstellung

|  |  |  |
| --- | --- | --- |
| Person | Arbeit | Zeit |
| Wortha | Verbindung zwischen den Hosts implementiert | 30 min |
| Wortha | GUI implementiert | 30 min |
| Wortha, Schober | Arbeit in der Schule, UML upgedated, Decorater-Pattern bedacht, Begin Implementation | 80 min |
| Wortha | Implementation von Badwordilter und UpperCase | 15 min |
| Wortha | Kommentiert, andere kleine Aenderungen | 30 min |
| Wortha | Testcases zu UpperCase und Badwordfilter | 15 min |
| Schober | Testcases zu ChatClient und ChatServer | 15 min |

Gesamter Arbeitsaufwand:

Wortha: 3 Stunden 20 Minuten

Schober: 1 Stunde 20 Minuten

# UML-Diagramm/Designüberlegung

# Things I've done

Wie oben im UML-Diagramm dargestellt wurde das Decorator-Pattern angewandt.

Wobei ChatMessage die eigentliche Nachricht enthält und UpperCase erweitert die getString Methode um ein toUpperCase und der Baswordfilter kann optional hinzu geschaltet werden, wobei dieser nur die drei Wörter „ARSCH“, „SCHEISS“ und „DUMBASS“ durch „\*\*\*\*“ ersetzt.

WriteAble message = **new** ChatMessage(neu);

message = **new** UpperCase(message);

**if** (filter.isSelected())

message = **new** Badwordfilter(message);

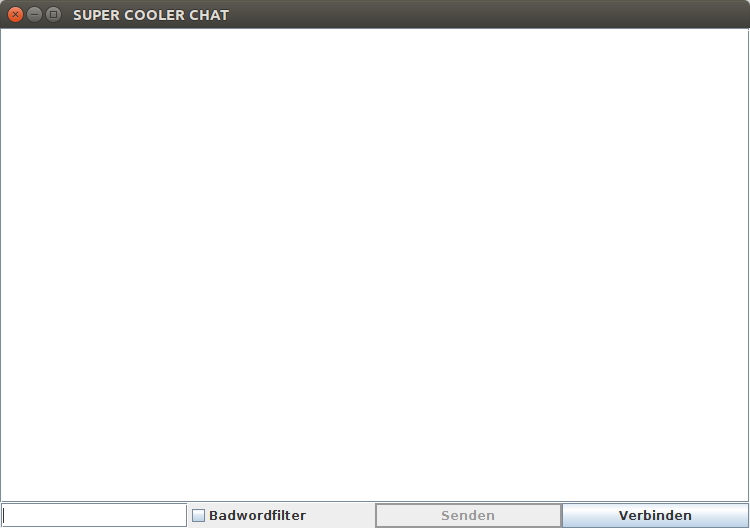
**this**.chat.append(message.getString() + "\n");

Beim Starten des Programms müssen folgende Parameter übergeben werden:

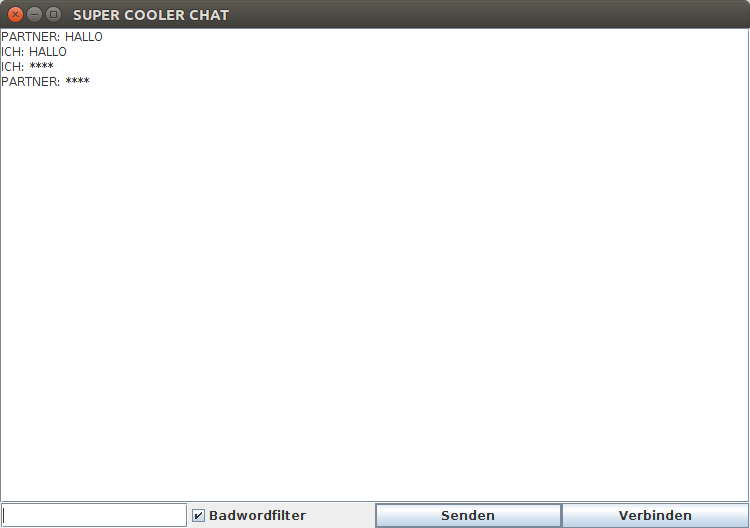
<IP-Adresse> <Portnummer> also z.B.: 127.0.0.1 6666

Als Verbindung wurde Interprozesskommunikation verwendet (siehe KnockKnock Beispiel in SYT)

Die GUI sollte selbst erklärend sein:



Der Senden-Button ist erst enabled, wenn der Verbinden Button getätigt wurde. Denn erst dann Verbindet sich der Client (bzw erst dann wird er gestartet) mit dem Server des Partners. So wurde das Problem gelöst, dass beide Anwender das Programm gleichzeitig starten müssen.Ist die Verbindung aufgebaut kann geschattet werden:



# Testing

Beim Testen gab es keine Probleme.

Für weitere Informationen siehe Javadoc.