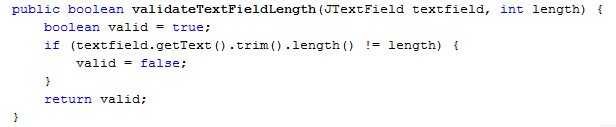
## Exceptionhandling

Vi har valgt at fokusere meget på exceptionhandling og udskrive beskrivende fejlbeskeder som er overskuelige og står steder i GUI’en der giver mening. Samtidig har vi lagt meget vægt på brugervenlighed, og derfor har vi også lavet validering på alt input der bliver indtastet når vi opretter og redigerer varer, fakturaer og ordre. Disse valideringer fungerer også som exceptionhandling da vi undgår en masse fejl der kunne være opstået hvis brugeren af systemet havde prøvet at oprette og redigere diverse ting med invalid input. Hvis inputtet ikke er validt vil udskriver vi fejlbeskeder ved de enkelte TextFields der har forkert input. De næste eksempler på validering er fra vores OrdreSalg panel, da det er der der er mest validering af input og det er derfor også der det er delt mest op. I nogle af vores andre GUI’er er valideringen ikke så kompleks så der har vi ikke har brug for den samme mængde af metoder til valideringen.

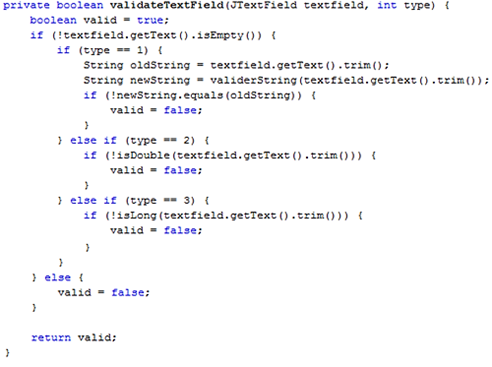
#### isDouble(String input) og isLong(String input)

Metoder isDouble() og isLong() er de meste basiske metoder vi har til validering. De bruges til at tjekke om en String fra f.eks. et TextField er hendholdsvis et heltal i isLong() og et decimaltal i isDouble(). Måden metoderne gør dette på er at vi prøver at parse de strings som metoderne er kaldt med til det input som vi skal bruge, og hvis det kan lade sig gøre returnerer de true og hvis kaster parsen en exception som så bliver fanget og returner false i stedet. Grunden til vi bruger long i stedet for integer til at validere heltal, er at vi har numre som er længere end en integer kan være, f.eks. kontonummer som er på 10 cifre.

#### public boolean validateTextFieldLength(JTextField textfield, int length)

Metoden bliver kaldt med et TextField og en integer som er længden som inputtet skal være. Metoden tager teksten fra TextField’et og bruger trim()metoden som fjerner alle mellemrum der er i starten eller slutningen af den pågældene String. Derefter tjekker den om længden er den samme som den metoden blev kaldt med. Hvis den ikke er returnerer den false og hvis den er returnerer den true. 

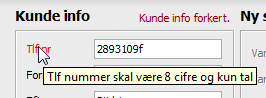
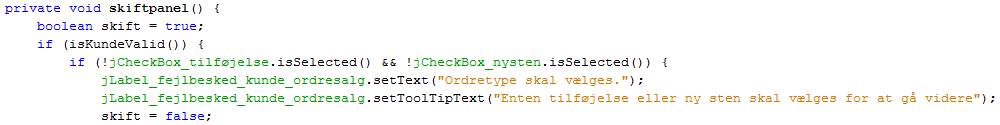
#### private String validerString(String string)

ValiderString metoden tjekker om der er backslash i det input som metoden bliver kaldt med, da et backslash kan ødelægge vores SQL-strenge når vi sætter ind i databasen. Den tager den gamle String og erstatter alle backslashes med ingenting og returnerer den derefter.

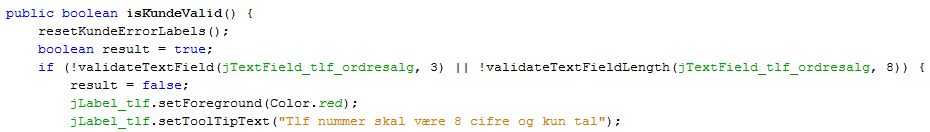
#### public boolean validateTextField(JTextField, int type)

ValidateTextField bruger flere af de ovennævnte metoder til at validere et TextField. Først tjekker den om TextField’et er tomt, som vil sætte TextField’et til invalidt. Hvis der er noget i TextField’et vil koden fortsætte til næste del hvor metoden tjekker hvilken type metoden er kaldt med. Hvis type er 1 tjekker metoden om der er backslashes i String’en, hvis type er 2 tjekker den om String’en er en double ved at bruge isDouble() metoden, og hvis type er 3 bruger den isLong() metoden til at tjekke om det er en long. Hvis valideringen er succesfuld returnerer metoden true og hvis ikke returnerer den false.

#### Eksempel på validering

Et eksempel på den validering der er beskrevet foroven er hvis vi prøver at gå videre til næste del af OrdreSalg ved at trykke på videre knappen i nedre højre hjørne. Hvis f.eks. telefonnummeret ikke er validt vil der Tlf\_nr labelen blive rød og der vil komme et tooltip frem hvis man holder over det hvor der står hvad fejlen er. Derudover kommer der også en general fejlbesked ved kundeinfo panelet så kunden kan lokalisere hvor fejlen er sket henne.

Videre knappen kalder metoden skiftpanel() som starter med at prøve at validere kunden med isKundeValid() metoden. Hvis kunden ikke er valid vil koden ikke gå videre til næste side.

isKundeValid() metoden der kaldes fra skiftpanel() tjekker hvert enkelt TextField i kundeinfo panelet, ved hjælp af validateTextField() og validateTextFieldLength() og hvis de ikke er valide sætter den de fejl i GUI’en som skal vises.

#### Plinger

  
Plinger kan ligesom backslashes ødelægge en SQL-streng da databasen vil opfatte en normal plinger som slutningen af et enkelt input, og derfor laver vi en replace() på alle Strings før de sættes ind i objekterne. Denne replace erstatter en plinger med en plinger med en backslash foran som fungerer som en escape character, så databasen forstår at den plinger der står ikke er en slutning på et input, men derimod et tegn der skal indgå i inputtet.

#### SQLExceptions

Vi har også steder i koden hvor vi fanger SQLExceptions, men da vi allerede har valideret alt data der skal sættes ind i databasen kan fejlene ikke være et problem med vores input. Den eneste fejl der derfor kan være sket når koden fanger SQLExceptions, er en kritisk fejl med databasen, f.eks. hvis der mangler en tabel eller der helt mangler forbindelse til databasen. Derfor har vi valgt at udskrive disse fejl så der står at det er et kritisk problem så brugeren forstår at det er et specielt problem som kræver en løsning eksternt.

#### ControlExceptions

Vi har selv oprettet en klasse der hedder ControlException som bare er en klasse der extender en Exception hvor man selv kan sætte en besked. Vi bruger den i DatabaseObjectHandler klassen til at kaste nye fejl når der er ugyldige data i forbindelse med indsættelse og redigering af databasen.



Her fanger vi exceptionen og skriver at det er ugyldigt objekt data der er problemet og udskriver den detaljerede besked på tooltippet.



Et eksempel på en ControlException. Her prøver vi at oprette en ordre med et givent ordrenummer, men hvis ordrenummeret allerede findes vil den kaste denne exception.

#### DocumentExceptions

Denne exception kan opstå i sammenhæng med vores oprettelse af PDF’er, men da vi har valideret alt vores data, og vores klasse fungerer fint, er det kun uforventede exceptions der kan opstå og vi har derfor valgt at udskrive fejlen der kan opstå på samme måde SQLExceptions.

#### IOExceptions

Denne exception kan også opstå når vi bruger PDF’er. Den kommer når vi prøver at redigere i en PDF der allerede er åben. Dette burde være den eneste fejl der kan opstå de steder vi bruger den, da vi kun bruger den i forbindelse med oprettelse af PDF’er.

