Programmet er delt opp i tre moduler: Boligpanel, Personpanel og Kontraktpanel. Hver av disse delene skal utgjøre en egen fane i et vindu med JTabbedPane implementert.

Boligpanelet

Boligtabell

Klassen Boligtabell er bygd opp kun for å håndtere tabellene som skal skrives ut i vinduet. Opprinnelig skulle programmet også håndtere oppdateringer og innlegging av ny data, men dette ansvaret ble overført til klassen DwellingUnit, som kan opprette en rekke type objekter som i teorien kan hente ut diverse filtrerte lister og gjennomføre masseoppdateringer. Denne klassen er beskrevet grundigere nedenfor.

Boligtabell-klassens metoder returnerer primært DefaultTableModel. Etter mye undersøking, fant jeg frem til at dette var det letteste for å få skrevet ut resultater basert på enkle databasesøk for å få skrevet ut alt eller skrevet ut spesifikke adresser.

Metoden visAltIBoligTabellen() returnerer hele tabellen basert på et enkelt søk etter alt i tabellen. Denne metoden mater data fra ResultSet inn i to vektorer, som blir sendt med DefaultTableModel-en som returneres.

Det samme gjelder metoden finnBoligVedAaSkriveInnAdresse(), men denne returnerer i stedet et DefaultTableModel med spesifikke data basert på en enkel søking etter adresse. Søkingen er satt opp slik at når man søker etter adresse, kan man søke med etter adresser med en hvilken som helst bokstav eller en hvilken som helst rekkefølge av bokstaver. Man trenger ikke skrive hele adressenavnet.

Boligvindu

Denne klassen er, tross navnet, et JPanel med indre JPanels med faste plasser. Hver av panelene har faste plasser på vinduet. Øverste panel er et buttonpanel hvor man kan legge til en rekke JButtons for andre funksjoner, type masseoppdatering eller lignende. Panelet på venstre side er der man skal legge inn verdiene i databasen som ikke blir automatisk opprettet ved opprettelse av en ny rad med info.

Til høyre er et panel hvor man kan få en lang tekststreng med info utskrevet ved søk etter id eller ved å trykke hvor som helst på en utskrevet tabellrad (Se metodene visAlleBoliger() og finnBoligVedAdresse() og tilhørende kommentarer).

I bunnen er panelet som skriver ut tabellrader ved søk etter adresse eller ved å trykke på «Vis alle boliger» - knappen. Denne knappen er i toppen av dette panelet, søkeknappen er i bunnen av dette panelet. Ved å trykke på søkeknappen, kan man få valget mellom å søke etter id eller adresse(igjen, se metoden finnBoligVedAdresse og metoden finnBoligVedId).

Konstruktøren tar imot og deklarer et objekt av Boligtabell-klassen og et objekt av DwellingUnit-klassen. Boligtabell-objektet brukes til å opprette hver sin JTable, som sendes inn i tabellpanelet ved kall på de respektive metodene(finnBoligVedAdresse() eller visAlleBoliger()).

DwellingUnit-objektet brukes til å hente ut bolig-informasjon ved kall på finnBoligvedId(), samt legge inn nye rader med boliger og oppdatere enkelte kolonneverdier.

Hvert panel har en egen ActionListener.

Metoden for å oppdatere kolonneverdier heter oppdater(), og ved å trykke på «Oppdater»-knappen, blir man først spurt om å skrive inn ID-en til boligen man vil endre. DwellingUnit-objektet finner frem til boligen som skal endres. Deretter får man opp et vindu hvor man kan velge mellom en rekke verdier å endre. Ut fra hva man velger å endre, får man opp et vindu der man blir spurt om hvilken ny verdi kolonnen skal få. Har kolonnen den verdien man velger, får man melding om dette. Hvis ikke, endres verdien. Foreløpig er dette bare implementert for boolean-verdier, men DwellingUnit har som nevnt metoder for å endre String-verdier og int-verdier også.

Ved trykk på «Opprett ny bolig»-knappen, kalles settInnNyBolig(), som henter ut alle verdier som er skrevet inn i tekstfeltene. Dersom alle verdiene er forskjellige fra null, kalles metoden som oppretter ny tabellrad.

visAlleBoliger() er den enkleste metoden. Ved trykk på «Vis alle boliger»-metoden, opprettes et JTable med DefaultTableModel fra Boligtabell-klassen. Denne tabellen blir opprettet med objekt-klasser, slik at boolean-verdier fremstår som check-boxer, dato-verdier blir skrevet ut med satt dato og måned, og lignende. Tabellen får dessuten en muselytter, som ved klikk på en hvilken som helst kolonne sender en tekststreng med info fra hele den tilhørende raden til utskriftsområdet(se visAlleBoliger()-metoden). Tabellen blir opprettet med en JScrollPane.

Når man søker etter adresse, får man opp et vindu hvor man skriver inn adressenavn, eventuelt en bokstav eller en rekke med bokstaver som adressen(e) man leter etter inneholder. Input-en blir sendt til finnBoligVedAaSkriveInnAdresse(), som oppretter en DefaultTableModel med verdiene i. Deretter lages JTable med denne i og sendes til panelet. Denne tabellen har objekt-klasser og muselytter på samme måte som visAlleBoliger()(se metoden finnBoligVedAdresse()).

finnBoligVedId() er metoden som blir kalt ved å velge å søke etter ID. Ettersom hver bolig blir opprettet med en unik ID, er det ingen mening i å skrive ut en JTable. Derfor hentes boligen med DwellingUnit-metoden findDwellingUnitWithID(). I utskriftsområdet skrives det ut info om boligen med ID-en man har søkt etter.

Milepælene

Uke 13

Selv om vi fikk mye god planlegging i gang, så var det et faktum at vi måtte lese oss selv opp på det vi drev med. Brukte derfor slutten av denne uken på å lese oss opp på databasebruk.

Uke 14

Begynte å kode. Mye prøving og feiling. Sebastian begynte med databasen.

Uke 15

Like før påske, ble vi noenlunde enige om utseende på programstrukturen, og hvem som skulle gjøre hva. Planen var at vi skulle kode så mye som mulig i løpet av påsken, og at Sebastian skulle bli ferdig med databasen.

Uke 16

I påskeuken ble jeg ferdig med Boligtabell-klassen. Sebastian ble ferdig med databasen.

Uke 17

På grunn av prioritering, ble det ikke så mye arbeid denne uken. Vi hadde dessuten noen spørsmål om arbeidet til stud-ass, som vi ikke fikk kontakt med denne uken. Her var vi usikre på hvilken retning prosjektet tok.

Uke 18

Ble denne uken klart at vi fikk to klasser for panelene: En for tabeller og en for vinduer. Dette var først og fremst fordi vi programmerte litt vel individuelt, men separat fant gode løsninger på forskjellige problemer, en også for å kunne lese gjennom koden litt enklere.

På grunn av dårlig tid denne uken, ble vi enige om virkelig å sette inn støtet fra og med neste uke.

Uke 19

Jobbet svært mye og lenge denne uken. Fant ut at det var lettest å vise tabellene med DefaultTableModel. I tabellklassene fant Carl ut at det var litt for vanskelig å få cachedrowset over i JTable. Dermed ble det DefaultTableModel. Han opprettet også et brukervindu og utviklet dette ferdig. Når det ble prøvd med boligdelen av prosjektet, funket det veldig bra, og vi ble dermed enige om at vi skulle bruke Boligvindu-klassen som modell for resten av panelene.

Sebastian ble denne uken helt ferdig med arbeidet rundt klassene, som omfattet spesiell søking, gjennomløpninger, get-metoder, oppdateringsmetoder for kolonner med mer.

Uke 20