Database prosjekt del 2

Robert Rognerud, Kenneth Gabrielsen, Simon Tveit, Mats Ove Sannes

16. mars 2018

1 Klassebeskrivelse

Apparat

Klassen brukes til å legge til og hente informasjon om apparater. Med metoden “save” kan vi legge til nye apparater, mens metoden ”refresh”, som igjen bruker ”initialize”, henter vi informasjon om apparatet med oppgitt ID.

ApparatØvelse

Klassen brukes til å legge til og hente informasjon om gjennomført øvelse ved et apparat. Med metoden “save” kan vi legge til nye gjennomførte øvelser med tid og kilo, mens metoden ”refresh”, som igjen bruker ”initialize”, henter vi informasjon om øvelsen med øvelsesID og øktID.

Treningsøkt

Klassen brukes til å legge til og hente informasjon om treningsøkt. Med metoden “save” kan vi legge til ny treningsøkt med dato, tidspunkt, varighet, form og prestasjon. Metoden ”refresh”, som igjen bruker ”initialize”, henter informasjon om treningsøkt med oppgitt øktID.

FriØvelse

Klassen brukes til å legge til og hente informasjon om friøvelser. Med metoden “save” kan vi legge til ny friøvelse med øktID, øvelsesID, og resultat. Metoden ”refresh”, som igjen bruker ”initialize”, henter informasjon om treningsøkt med oppgitt øktID og øvelsesID.

Øvelse

Klassen brukes til å legge til og hente øvelser. Med metoden “save” kan vi lagre øvelser, mens metoden ”refresh”, som igjen bruker ”initiliaze”, henter vi øvelse med oppgitt ID.

ØvelsesGruppe
Klassen brukes til å legge til og hente gruppebeskrivelse. Med metoden “save” kan vi legge til en beskrivelse, mens metoden ”refresh”, som igjen bruker ”initiliaze”, henter vi en gruppebeskrivelse med oppgitt gruppeID.

Main
Hovedklassen som setter opp og starter applikasjonen ved å laste TreningsdagbokFXML.

ActiveDomainObject
Grensesnitt med metodene initialize(conn), refresh(conn) og save(conn). Alle klassene som representerer entitetene i relasjonsdatabasemodellen utvider ActiveDomainObject.

DBConn
En abstrakt klasse som setter opp forbindelse mellom program og database. Brukes i SQLConn.

SQLConn
SQLConn utvider DBConn for å sette opp forbindelse med databasen satt opp av Setup.sql.

SQLQuery Klassen inneholder metoder for å hente ut data fra databasen. StaticMethods
Inneholder en statisk toQuote-metode som setter inn strings i riktig format i SQL-koden.

TreningsDagbokController
Denne klassen kobler logikken fra resten av programmet opp mot FXML-fil.

Setup.sql
Lager databasen med tabeller.

TreningsdagbokFXML
Genererer et brukergrensesnitt til bruker av programmet for å skrive inn spørringer til databasen omgjort til sql-kode i programmet og gir svar i lesbar form på skjerm.

Main
Main-klassen bruker TreningsDagbokFXML til å sette opp et grafisk grensesnitt hvor man kan innføre og hente ut informasjon for å løse use casene. Grensesnittet bruker de andre klassene og deres metoder for å utføre oppdatering og uthenting av data fra databasen.

2 Use Cases

1. **Registrere apparater, øvelser og treningsøkter med tilhørende data**
   Dette løser vi ved hjelp av klassene apparat, apparatøvelse, friøvelse, treningsøkt, øvelse, og øvelsesgruppe. Etter å ha lagt til et apparat kan en øvelse kobles til det apparatet ved hjelp av apparatID. Når en treningsøkt er opprettet kan man legge til øvelser, både apparatøvelser og friøvlser, ved bruk av metoden addØvelseToØkt i TreningsDagbokController.

2. **Få opp informasjon om et antall n sist gjennomførte treningsøkter med notater, der n spesifiseres av brukeren.**
   Brukeren av applikasjonen fører inn hvor mange av de siste treningsøktene hen vil ha informasjon om, og ved bruk av metoden getNSisteØkter i klassen SQLQuery sendes denne spøringen til databasen om n siste treningsøkter. Databasen returnerer denne informasjonen.

3. **For hver enkelt øvelse skal det kunne være mulig å se en resultatlogg i et gitt tidsinterval spesifisert av brukeren.**
   Bruker gir øvelsesID og start- og slutttidspunkt for den øvelsen og det intervallet den vil se resultater fra. Deretter lages SQL-spørringen i getØvelsesResultat-metoden i SQLQuerry-klassen. Deretter blir SQL-spørringen sendt til databasen via SQLQuerry-klassen, og svaret lagres tidspunktet for øvelsen og resultatet i en string. Vi valgte å lagre det i en streng i stedet for et objekt da det ikke var et objekt med attributtene resultat ***og*** tidspunkt. Deretter blir strengen sendt ut til bruker via FXML og controller-klassen.

4. **Lage øvelsesgrupper og finne øvelser som er i samme gruppe.**
   Klassen øvelsesgruppe brukes for å lage en øvelsesgruppe med en id og en beskrivelse. Når en gruppe er lagd kan øvelser legges til gruppe ved metoden addToGroup som sender spørring til tabellen ØvelseTilhørerGruppe. Vi henter ut informasjon om hvilke øvelser som finnes i samme gruppe ved at bruker oppgir gruppeID på den gruppen han vil vite om og det sendes spørring til tabellen om ved metode getOvelserInGroup.

5. **Finne ut hvilke øvelser et apparat kan brukes til**
   Bruker kan få informasjon om hvilke øvelser et apparat kan brukes til ved å oppgi appratets ID. Metoden getOvelserTilApprat tar denne IDen og bruker den til å sende spørring til tabellen Øvelse i databasen. Det returneres øvelsesID, øvelsesnavn, øvelsesbeskrivelse og apparatID for øvelsene som kan gjennomføres ved det spesifikke apparatet.