DEMOVIDEO: <https://www.youtube.com/watch?v=RYW1bOm__Oo> (nauhoitus suoraan koneelta ei oikein pyörinyt sulavasti tällä läppärillä..)

Bugi kohdassa 3:04 korjattu viimeiseen versioon

## Tehtävänannon kuvaus ja työn rajaukset.

Luodaan toiminnaltaan Itellaa muistuttava ohjelmisto valmista SmartPost verkostoa käyttäen, ohelman tulee käyttää tietokantaa (tässä tapauksessa sqlite3).

Ohjelmassa luodaan SmartPost-olioita, jotka koostuvat automaatin tiedoista. Automaateista pidetään tieto varastossa erillisessä tietokannasta, jonka avulla hallinnoidaan myös paketteja sekä esineitä.

Paketeille tulee olla 3 eri luokkaa, joilla on erilaisia ominaisuuksia. Paketit sisältävät vapaavalintaisia esineitä, joiden on sovittava pakettiin(ohjelma ei hyväksy virheellisilä sijoituksia. Esineitä tulee olla ainakin 4 erilaista, ja niillä on omat esinekohtaiset ominaisuutensa kuten koko ja rikkoutuvuus.

1.lk paketti:

kantama: 150km

Nopein pakettiluokka, mutta kaikki rikkoutuvat esineet rikkoutuvat

Paketin tilavuus 100l (n. 45 cm sivu)

2.lk paketti:

kantama: rajoittamaton

ei rikkoudu

pieni paketti: 30l (reilu 30cm sivu)

Nopeus on 1 ja 3 luokan väliltä

3.lk paketti:

kantama: rajoittamaton

voi rikkoutua (heitellään seinien kautta)

suuri paino eduksi (tarpeeksi iso --> ei rikkoudu)

Paketin tilavuus 250l (n. 65cm sivu)

Hidas

SmartPost-verkosto sekä pakettivarasto liitetään yhteen pääluokalla, jonka avulla tietoja yhdistetään ja käytetään. Pääluokalle tehdään myös graafinen käyttöliittymä jolla tietoa esitetään. Esim. Automaatit ja pakettien lähettäminen esitetään visuaalisesti kartalla.

## Tietokannan käsitemalli ja eheyssäännöt.

Käsitemalli (ja relaatiomalli) löytyvät erillisinä Dia-tiedostoina.

Tietokantaan on sisällytetty ainoastaan NOT NULL eheyssäännöt, yksi default aikaleima sekä pääavainten(joissan tapauksissa myös vieras) arvon kasvattaminen automaattiesti. Eheyssäännöistä huolehditaan siis pääsääntöisesti ohjelman kautta (rakennetun käyttöliittymän kautta käytettäessä tietokanta pysyy eheänä.)

Koska Smartpost tiedot on hajautettu useampaan tauluun, pidetään ohjelmassa huoli siitä että muutokset tehdään samanaikaisesti kaikkiin tauluihin. (Esim ei voida lisätä pelkästään smartpost taulun tietoja jättäen sijaintitaulut tyhjäksi.)

Mikäli tietokannasta poistettaisiin smartposteja, on myös automaattikohtaisten sijaintitietojen poistosta huolehdittu ON DELETE CASCADE komennoilla.

Jotta ei voi syntyä tilannetta jossa käyttäjä poistaa tavaran joka on jo paketoitu ja lähetetty (historia taulussa), näytetään käyttäjälle ohjelmassa ainoastaan ne tavarat joille ei olla vielä tehty mitään.

Kiitettäen hyväksyttyyn harjoitustyöhön lisävaatimuksena toteutettiin tietokantaan yli 10 taulua. Myös useita JOIN lauseilla rakennettuja kyselyjä löytyy. Tietokannan luontikomennot ovat luo\_db.txt tiedostossa, siellä myös kaksi näkymää. Kiittettävän suorituksen kriteerit täyttynee?

Käytetyt SELECT lauseet löytyvät erillisestä tiedostosta “select.txt”.

## Ohjelman kuvaus

Ohjelman toimivuus testattu Debianilla. Tietokannan oletuspolku on “jdbc:sqlite:/root/TIMOTEI/Documents/db.sqlite3”. Tämän voi koodista muuttaa FXMLDocumentController osaluokan Timo rakentajasta muuttamalla pathToDB Stringiä.

Ohjelma sisältää ylempänä kuvatut kolme pakettiluokkaa ja seuraavat esineet:

Nimi: Raskas vaasi

Rikkoutuva: 1

Tilavuus: 225

paino: 100kg

Nimi: Tietokoneen keskusyksikkö

Rikkoutuva: 1

Tilavuus: 50

paino: 5kg

Nimi: Lompakko

Rikkoutuva: 0

Tilavuus: 1

paino: 1kg

Nimi: Pikachu

Rikkoutuva: 0

Tilavuus: 20

paino: 15kg

## Raportti ohjelmallisesta toteutuksesta ja lista toiminnallisuuksista.

Tein ensin karkean käsitemallin jonka pohjalta aloin suunnittelemaan relaatiomallia. Suunnitteluun kului melko paljon aikaa, mutta sen jälkeen tietokanta toimi oikein ilman suurempia rakenteellisia muutoksia.

Itse tietokannan luonti SQLiteen oli pelkkää mekaanista suorittamista, ja onnistui nopeasti. Tietokannalle käyttöliittymän suunnitteluun ei enää ehtinyt käyttää paljoa aikaa, ja karkeana pohjana käytettiin tehtävänannon kaaviota. Pieni virhe lienee se, että tietokanta tehtiin suomeksi ja itse ohjelma pääosin englanniksi. Toiminnallisuuksia käyttöliittymään mitä ehdin toteuttaa:

-Aloitusnäyttö jotka kustomoitu hieman css:llä.

-Käynnistäessä ilmoittaa mikäli uusinta XML-dataa ei saatu ladattua tietokantaan ja kehoittaa käyttäjää tarkistamaan internet-yhteyden.

-XML-dataa ladattaessa tarkistaa löytyykö automaatit jo tietokanasta.

-Kartalle voi lisätä SmartPosteja. Valittaessa Smartpost automaatti pudotusvalikosta esitetään ko. Automaatin tietoja alla olevassa tekstilaatikossa.

-Voidaan luoda 4 erilaista esinettä (luodut esineet tallentuvat tietokantaan)

-3 Eri pakettiluokkaa joiden ominaisuudet kuvattuna ylempänä

-Esineistä ja pakettiluokasta esitetään tietoja valittaessa

-Esineet voidaan paketoida , onnistuneesti luotu paketti tallentuu tietokantaan

-Tietokannassa olevia esineitä sekä paketteja voidaan poistaa (jo paketoituja / lähetettyjä esineitä/paketteja ei voida poistaa)

-Pakettien lähetys, luominen ja poistaminen sekä historiatiedot eri välilehdissä

-Kartalle lisättyjen automaattien välillä voidaan lähettää luotuja paketteja riippuen pakettiluokasta, paketin lähetys kuvataan kartalla visuaalisesti

-Kartalle lisätyt automaatit ja piirretyt reitit voidaan tyhjentää kartalta

-Yksittäisiä historiatietoja voidaan tarkastella. Pudotusvalikosta valitaan vanha lähetys ja sen tiedot sekä luotujen pakettien yhteismäärä esitetään. (tieto haetaan tietokannan näkymästä)

-Kartalle lisättyä automaattia voi klikata jolloin sen aukiolon pitäisi tulla näkyviin, ei testattu sillä javafx:ssä jotain vikaa koneellani ja voin kontrolloida karttaa ainoastaan nuolilla.

Näillä toiminnallisuuksilla ohjelmani pitäisi täyttää **26 pisteen** kriteerit.

## Ohjelman täydellinen luokkakaavio

Erillisessä tiedostossa.

## Yhteenveto. Mitä ongelmia jouduttiin ratkomaan tai mitä erityistä havaittiin ja opittiin tietokannan toteutuksen yhteydessä.

Tietokannan suunnittelu osoittautui hankalammaksi kuin osasin odottaa. Alkuun on vaikeaa hahmottaa miten taulujen väliset yhteydet olisi optimaalista toteuttaa, ja miten tieto tulisi sijiottaa eri tauluihin. Nähdäkseni lopputuloksena on kuitenkin oikeaoppinen tietokanta kolmanteen asteeseen normalisoituna. Käyttöliittymää luodessa meni myös hetki perehtyessä siihen miten tietokantaa käytetään ja kannattaa käyttää Javan kautta.