DTEK1049 Käyttöliittymät, kevät 2019

Harjoitustehtävä A-2: Viikkoharjoitus 2

Kurssin teorian osalta viikkoharjoituksen 2 tehtävissä sovelletaan luentojen 2–4 asioita. Viikkoharjoituksia voi pohtia ryhmissä (harjoituksista järjestetään keskusteleva tutoriaalisessio joka torstai), mutta jokaiselta osallistujalta odotetaan henkilökohtaista suoritusta ja palautusta. Jokainen tehtävä arvioidaan pistein 0 – 0,5 – 1,0 viim. 3 viikon sisällä, mutta mahd. nopeasti palautuksesta.

Varmista, että saat FXML-tiedostot tallennettua ja myöhemmin avattua Scene Builder -ohjelmalla (ohjelman käyttämä tiedostomuoto on .fxml) ja ajettavan ohjelmakoodin (Java/Scala) käännettyä käyttämällä esim. jotakin ohjeiden projektirungoista. Kurssin valmiiden projektirunkojen hakemistorakenne on seuraava:

1) projektin alla src-kansio sisältää ratkaisun kannalta oleelliset resurssit ja koodit (.java/.fxml-tiedostot)
2) projektin alla project-kansiossa IDEA-projektissa tarpeelliset plugin-säädöt
3) mavenin pom.xml (Eclipse) ja sbt:n build.sbt (IDEA) projektin juuressa projektin konfiguraation (javafx ja muut kirjastoriippuvuudet, esim. JFoenix)
4) projektin alla target-kansioon generoidaan class/jar-tiedostot suorittamista varten (näitä ei tarvita palautuksessa)

Sisällytä näistä 1)–3) tehtävän palautukseen. Tai jos et käytä valmiita projektipohjia, huolehdi että tarvittavat tiedostot tulevat mukaan muuten. Tehtävän palautusmuoto on zip-pakettina nämä em. työkalujen vaatimat syötetiedostot (ei class/jar).

Projektirunkojen git on säädetty niin, että jos repositoryn “forkkaa” käyttöönsä, gitlabiin tallentuvat vain oleelliset tiedostot ja class/jar-tiedostot suodatetaan pois automaattisesti. Näin palautuksen zip-paketin voi luoda helposti esim. gitlabin projektisivulta pilvinapin takaa. Huomaa kuitenkin, että git-repositoryn forkattu klooni näkyy oletusarvoisesti kaikille julkisesti eli työsi hedelmät voidaan “varastaa”, mikäli et säädä repositoryn näkyvyyttä piilotetuksi.

Palauta tehtävien 1 ja tehtävän 2 tiedostot yhteen pakettiin zipattuna (ilman ääkirjaimia tiedostonimissä) 22.2. mennessä kurssin Moodlen viikkoharjoituksen palautuslokeroon.

Tehtävä 1

Tehtävässä jatketaan 1. harjoituskerran vuokravälitysohjelman rakentamista. Lisää ohjelmaan seuraavat toiminnallisuudet

1) Toteuta Java-ohjelmaan alkuvalikko, josta voi avata nappia painamalla joko A) vuokrakohteen lisäämisen tai B) vuokrakohteiden etsinnän (viime kerran lomakkeet). Alkuvalikon tulee piilottaa itsensä, kun ikkuna A) tai

B) aktivoituu. Toteuta kyseinen alkuvalikko ohjelmoimalla sitä vastaavat käyttöliittymän osat Java-koodilla ilman Scene Builderia.

Vinkki: Alkuvalikko kannattaa avata JavaFX:n oletus-Stagella, mutta ikkunat A) ja B) vaativat uuden Stagen ja Scenen luontia kullekin. Napit voi koostaa siististi esim. VBoxiin BorderPanen keskelle sopivalla spacing-arvolla. Tarvitset napeille tapahtumankäsittelijöitä. Avattavat ikkunat puolestaan lataisivat vastaavat FXML-tiedostot kuten 1. harjoituskerran vastauksessa. Ikkunaa ei tarvitse poistaa vaan ainoastaan muuttaa sen näkyvyysominaisuus.

Ikkunoista A) ja B) tulisi päästä joko kohteen lisäämisen / haun jälkeen tai toiminnon peruuttamisen seurauksena päästä takaisin päävalikkoon niin, että ikkuna A) / B) sulkeutuu ja päävalikko avautuu. Tätä varten käyttöliittymissä on yleensä nappeja otsikoilla “Lisää” ja “Hae” (tehtävän bisneslogiikalle) ja “Peruuta” / “Sulje” jos ei halutakaan tehdä mitään.

2) Kytke FXML-kuvaus osaksi Java-koodia säätämällä kontrollerit ja FXML-komponenttien fx:id:t. Luennon ohjeistuksen täydennys tässä:

- Aseta tarvittaville komponenteille **fx:id** (Scene Builderin vasemmassa palkissa **Assigned fx:id Controller**-valikon alla ja oikeassa **fx:id** valikon **Code**:n alla)
- Aseta koko lomakkeelle kontrolleriluokka (Scene Builderin vasemmassa palkissa **Controller class**.
- Saat luotua automaattisesti FXML-tiedoston kanssa yhteensopivan Java-koodin **View**-valikosta toiminnolla **Show sample controller skeleton**.

  Kytke ja toteuta komponentit Java-koodiin niin, että 1. harjoituskerralla mainittu syötteen validointi voidaan suorittaa kullekin kriittiselle tietotyypille. Esim. valuutan kanssa kannattaa kikkailla niin, että voi käyttää Javan primitiivityyppejä ja tulostaa euro esim. Label-leimana erikseen, niin ei tarvitse jäsentää euromerkkiä. Validointiin voi käyttää Javan standardikirjaston Integer / Double -jäsennysrutiineja.

3) Lataa kurssin Moodle-sivulta viikkoharjoituksen 2 materiaalin zip-tiedosto. Paketti sisältää tiedostot Asunto.java ja AsuntoGeneraattori.java sekä svg-tiedostoina mahdollisia asuntojen kuvia, joita voi käyttää tehtävässä. (Kuva)tiedostojen valikointia varten JavaFX:ssä on FileChooser-luokka. Kuvatiedoston polkujen saaminen oikein Maven/SBT-projektissa niin että kuvat näkyvät ImageView-näkymässä voi vaatia pientä perehtymistä ja debuggausta.

4) Luo A) vuokrakohteen lisäämissivulle tapahtumankäsittely niin, että syötetyistä tiedoista luodaan Asunto-olio. Voit joutua soveltamaan koodeja niin, että Asunto-olio on yhteensopiva tekemäsi lomakkeen kanssa. Lisää luodut oliot esim. ArrayList-kokoelmaan. Tulosta kokoelman sisältö

esim. tekstikonsoliin jokaisen lisäämisen jälkeen (Asunto-luokalla on määritelty tähän .toString()). Tarkista, että pystyt lisäämään käyttöliittymää käyttäen lisäämään useita asuntoja ko. listaan.

Tehtävä 2

5) Toteuta vastaavasti asuntojen hakuikkuna niin, että ikkunan hakutuloksien näkymä generoidaan dynaamisesti 4)-kohdan ArrayListin pohjalta siten, että kutakin listan alkiota vastaa yksi näkymä hakulomakkeessa. Ikkunassa pitäisi ilman haun rajausta pystyä listaamaan kaikki kohdassa 4) luodut asunnot. Tehtävän voi mahdollisesti ratkaista jopa niin, että hakunäkymää vastaa kaksi FXML-tiedostoa, yksi generoitavaa hakutulosta varten ja yksi hakuikkunaa. Tai voit käyttää yhtä FXML-tiedostoa ja generoida näkymän koodilla tai jollain muulla tavalla. Pääasia, että sisältö generoidaan dynaamisesti koodilla.

   Luo tehtäväntestaamisen helpottamiseksi ArrayListiin AsuntoGeneraattori.luo() -metodilla valmiita kohteita pari kappaletta jo ohjelman käynnistyksessä.

   Haun rajaamisen osalta riittää toteuttaa koodi yhden hakukriteerin osalta. Näkymän tulisi siis päivittyä, kun hakukriteeri on syötetty ja haku suoritettu uudelleen.

6) Simuloidaan AsuntoGeneraattori.luoAsuntoja(int) -metodilla hidasta käyttöliittymätoimintoa. Metodi luo taustasäiettä käyttäen 2 sekunnin välein listaan uuden Asunto-olion, mikä voisi vastata esimerkiksi asuntotietojen kyselyn erittäin hitaan internet-linkin yli. Lisää hakukäyttöliittymään nappi, joka aktivoi asuntojen generoinnin AsuntoGeneraattori.luoAsuntoja -metodilla esim. 50 asunnon kappalemäärällä.

   Käytä JavaFX:n monisäikeisiä menetelmiä apuna ja toteuta näkymä niin, että A) näkymään päivittyy uusia kohteita hakukriteerillä suodatettuna (tai kaikki, jos kriteeriä ei säädetty) sitä mukaa kun Asuntoja luodaan. Lisäksi luo lomakkeeseen edistymispalkki, joka näyttää Asuntojen luonnin edistymisen, jotta käyttäjä osaa arvioida, koska haku on valmis. Luokassa on staattiset metodit tehtäviäAsuntoja() ja tehtyjäAsuntoja() avuksi. Huomaa että luokassa kaikki public-jäsenet ovat säieturvallisia ja private-jäsenet eivät. Toki luokka luo metodissa luoAsuntoja erillisen säikeen, joka ei ole turvallinen JavaFX:n ohjastamiseen.