TKO_8971 Soveltava projekti

Työraportti: UnicaBot

Ryhmä:
Valtteri Ingman (524114)
Jani Uotinen (524131)
Sanna Volanen (506327)

Työraportti
2021-04-12
12 s.

Turun yliopisto
Tulevaisuuden teknologioiden laitos
Tietojenkäsittelytiede
Soveltava projekti

Sisältö

1 Työn kuvaus

1.1 Työn taustamotivaatio

Pääasiallisena taustamotivaationa oli opiskelijoiden tarve nopealle rajapinnalle ravintoloiden lounasruokalistojen saamiseen. Discord on kurssien lomassa ollut noususuhdanteessa opiskelijoiden käytössä, koska kyseistä alustaa on käytetty kurssien informaatioväylänä. Myös monilla ainejärjestöillä on serverit Discordissa ja rajapinta on hyvin ajankohtainen. Tästä syystä sama alusta tuntui luontevalta valinnalta päivittäisiä ruokailuvaihtoehtoja koskevan informaation jakamiseen ja hakemiseen.

Ajatuksena on ollut tehdä botti, joka yksinkertaisilla komennoilla tulostaa Discord-kanavalle halutun ravintolan ruokalistan opiskelijan tarpeen mukaan. Botin toiminnallisuutta pystyy jatkokehittämään hyvin helposti melkein mihin vain eikä se ole rajattu pelkästään Unica-ravintoloiden ruokalistojen tulostamiseen. On kuitenkin syytä pitää mielessä, että Unica-ravintoloiden JSON-rajapinta toimii vain aktiivinen arkiviikko kerrallaan. Eli tulevien viikkojen ruokalistat eivät ole saatavilla muulloin kuin alkavan viikon maanantaina.

1.2 Käytetyt teknologiat

Discord-botin toimintaa varten projektiin oli lisättävä useampi rajapinta kaiken halutun toiminnallisuuden lisäämiseksi. Näillä valmiilla rajapinnoilla botin luonti oli melko suoraviivaista.

1.2.1 Java Discord API

Projekti suunniteltiin tehtäväksi käyttäen Java-kieltä, joten Discord-botin luontiin käytettiin yleisintä Java-kielelle tarkoitettua Discord REST -rajapintapakettia tai käärettä, Java Discord APIa (myöhemmin JDA). Bottia käytetään reagoimaan Discord-serverin kanavilla syntyviin tapahtumiin (Event). Tällaisia tapahtumia ovat mm. viestit serverin kanavilla. Näitä tapahtumia hyödyntäen voidaan bottia käskeä lähettämään viestejä vastaukseksi. Bottioliota luodessa sille pitää antaa sen käynnistäjän Discordin kehitystyökaluista oma uniikki avain tai token, jonka avulla botti autentikoidaan ja yhdistetään Discordiin. Lisäksi botille annetaan räätälöityjä tapahtumien kuunteluluokkia (EventListener), joissa erilaisiin tapahtumiin reagoidaan. JDA:lla on melko kattavat GitHub-sivut[1], joista löytyy dokumentaatiota ja esimerkkikoodia sekä ohjeet sen lisäämisestä Maven- tai Gradle-projekteihin.

1.2.2 Jackson API

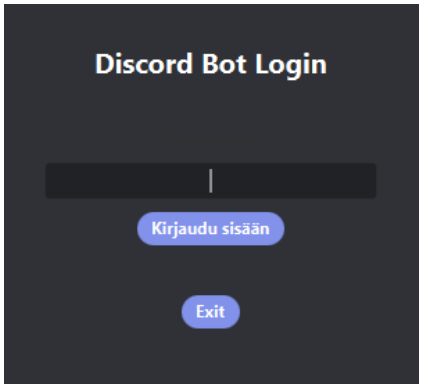
Jackson on Java-ohjelmointikielelle tarkoitettu kevyt ja helppokäyttöinen JSON-notaation parsintakirjasto. Sen avulla luodaan kohdesivun JSON-rakenteen perusteella luokkia, joiden ilmentymiä voi käyttää ohjelmassa. Kirjasto tukee myös JSON-tiedostomuodon lukemista ja kirjoittamista. Projektin kannalta tämä oli siis sopiva kirjasto Unica-ravintoloiden viikoittaisten ruokalistojen siirtämiseen botti-ohjelman käsiteltäväksi sekä botin oman locations.json -tiedoston ylläpitämiseksi.

[1] https://github.com/DV8FromTheWorld/JDA

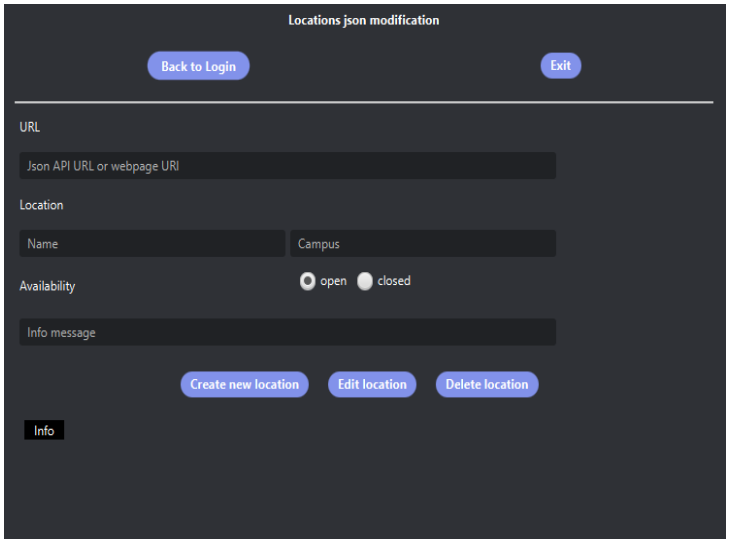
1.2.3 JavaFX

Botin lokaalia käyttöä helpottamiseksi sille kehitettiin graafinen käyttöliittymä käyttäen JavaFX-kirjastoa. Käyttöliittymän kautta voidaan hoitaa botin yhdistäminen Discordiin syöttämällä botin token-avain. Käyttöliittymässä pystyy myös lisäämään, poistamaan ja editoimaan paikkatiedostoa locations.json, joka sisältää ravintolat, joista botti voi hakea ruokalistoja.

JavaFX-rakenne projektissa on seuraavanlainen. Jokaisella scenellä on oma nimi.fxml tiedosto, joka sisältää kaikki scenen UI-elementit. Tähän tiedostoon myös kirjoitetaan viitteet metodeihin, joita elementit käyttävät sekä tunnisteet, joiden avulla elementteihin voi viitata koodissa. Jokaisella scenellä on myös oma muotoilusta vastaava nimi.css tiedosto, johon CSS-syntaksia kirjoittamalla voidaan muotoilla UI-elementtejä. Lisäksi sceneille luotiin omat yksilöidyt ohjainluokat (Controller), jotka sisältävät kaikki kunkin scenen käyttämät metodit ja toiminnallisuudet. Ohjelmassa on tällä hetkellä kaksi (2) sceneä, Login ja Scene2 (kts. kuvat 1.2.1 alla ja 1.2.2 seuraavalla sivulla).



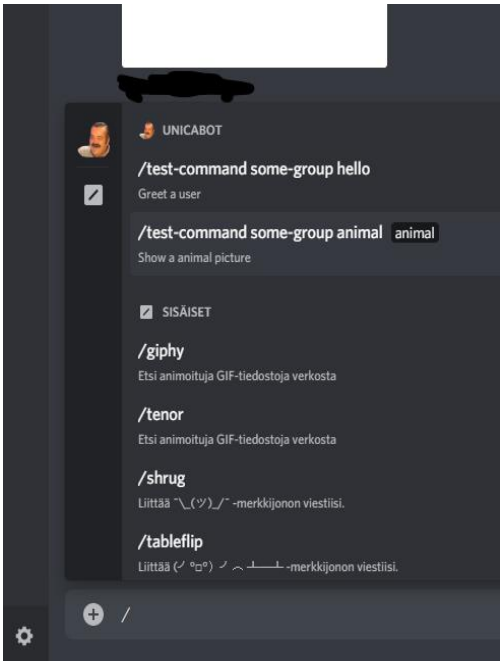
Kuva 1.2.1 Login-ikkuna GUIssa



Kuva 1.2.2 Scene2-ikkuna eli kirjautumisen jälkeinen muokkausnäkymä GUIssa

1.2.4 JDA slash commands

Valtaosa Discordin API:n uusimmista ominaisuuksista julkaistaan sen JavaScript-versiolle. Yksi näistä uusista ominaisuuksista on *slash commands*. Käytännössä se tarkoittaa sitä, että painamalla tekstikenttään /-merkin, ehdottaa Discord kaikkia niitä komentoja, joita voidaan kullakin botilla käyttää (kts. kuva 1.2.3 alla).



Kuva 1.2.3 Esimerkki slash-komennon toiminnasta

Edellä olevassa kuvassa 1.3 näkyy Githubista kopioitu esimerkki kyseisestä kirjastosta käytännössä. Esimerkissä näkyvät komennot eivät kuitenkaan toimi UnicaBotilla, vaan palauttavat virheen, koska kyseisiä komentoja ei ole tähän bottiin sisällytetty.

1.2.5 OkHttp

Edellä mainitut slash-komennot vaativat toimiakseen myös OkHttp-palvelimen, joka mahdollistaa mm. botin serverisoketin käytön käskyjen vastaanottoon. Ryhmä ei perehtynyt tarkemmin tämän tekniikan saloihin, koska työ ei vaatinut muutoksia sen implementointiin UnicaBotin tarvitsemien slash-komentojen osalta. Lisätietoa tästä tekniikasta löytyy täältä.

1.2.6 Maven jar -lisäosa

Ryhmä päätyi tekemään projektista jar-tiedostosta ajettavan version, joten tämän lisäosan implementointi oli tarpeen. Yksinkertaisesti se luo jar-tiedoston projektin root-kansioon.

1.2.7 Maven shade plugin sekä muut tässä nimeämättömät riippuvaisuudet ja repot

Jostain syystä edellä mainittu Maven jar -lisäosa ei ottanut mukaan Javafx-tiedostoja. Tämän puutteen korjattiin Maven shade -lisäosan mukaan tuonnilla. Samaa virkaa toimittavat muutkin tässä mainitsemattomat riippuvaisuudet, jotka löytyvät UnicaBot-projektin pom-tiedostosta. Ne korjaavat jonkun todella pienen ja yksityiskohtaisen toiminnallisuuspuutteen tai bugin, joten niiden lähempi esittely ei ole tälle projektille oleellista.

2 Kommentoitu koodi

2.1 Javadoc

Linkki repositoriossa olevaan Javadociin tässä.

2.2 README.md

Tämä README.md on tehty englanniksi, jotta se palvelisi paremmin Githubissa ei-suomalaisia repoon eksyjiä. Seuraavassa alaluvussa esitellään lyhyesti projektin kahden pää paketin sisältö suomeksi.

2.2.1 JSONParse ja Botti -pakkaukset

Tämän tekstin kirjoitus hetkellä Botti-pakkauksessa on seuraavat luokat:

i. **BotEventListener** sisältää botin asetuksia kuten prefiksin vaihtamisen ynnä muuta mukavaa. Luokka sisältää komentoja, joilla ei ole mitään tekemistä Unican kanssa.

ii. **UnicaMenuEventListener** sisältää nimensä mukaisesti kaikki komennot, jotka liittyvät Unican menujen tulostamiseen.

iii. **Botti** luo uusia UnicaBotti-instansseja yllä mainittujen tapahtumakuuntelijoiden kanssa.

iv. **jarBootMain** määrittää ollaanko koneella vaiko etäpalvelimella, ja päättää sen perusteella, millainen admin-käyttöliittymä tulostetaan.

JSONParse-pakkaus määrittelee, miltä haluttu tuloste näyttää, kun annetaan komento jostakin ruokalasta. Pakkauksessa on seuraavat luokat:

i. **SetMenu** rakentaa yksittäisen ruokalajin muotoilun järjestyksessä nimi (lounas, jälkiruoka, take away), hinta (opiskelija/työntekijä/muut), itse ruoat.

ii. **MenusForDay** tekee listan näistä eri menuista käyttäen SetMenussa määriteltyä muotoilua. Yhdessä päivässä on useampia ruokalajeja: vegaaninruoka, normaalien ihmisten ruoka, jälkiruoka sekä klo 12 eteenpäin tarjottava ruoka.

iii. **Restaurant** rakentaa ravintolaolion Unican jsonin pohjalta ja lisää siihen mukaan sarjan erilaisia gettereitä esimerkiksi ravintolan URL:ille, nimelle, errorviestille, eri päivien menuille yms. Restaurant-luokasta tehtyä oliota käytetään unicaMenuEventListener-luokan embedviesti-metodissa, jolla saadaan aikaan alla oleva tuloste.

Kuva 2.1Embedviestin ulkonäkö Discordissa

Kaikilla yllä mainituilla luokilla on omat toString()-metodinsa, joita käytettiin alkuvaiheessa, kun ei oltu ihan varmoja siitä, miltä tulostuksen tulisi näyttää. Ne on kuitenkin säästetty testausta varten.

3 Jatkokehitys

Projektin aikana on esiin tullut lukuisia kehitysideoita, joista osa on tässä vaiheessa jo kehityksessä ja osa jää tulevaisuuteen. Botin ajamiseen liittyen suunnitteilla on command line -käytön implementointi sekä botin siirto ylläpitopalvelimeen. Itse toiminnallisuuksiin kuuluvia kehityskohteita ovat slash-komentojen lisäys, embedviestin siistintä sekä GUI:n monipuolistaminen. Suunnitelmissa on myös mahdollisesti lisää testejä.

Yllä mainitut ominaisuudet toteutuvat tai ovat toteutumatta esityspäivään mennessä. Tällä hetkellä kuitenkin projektin master-haara on se uusin ja käyttökelpoisin versio, jonka palautamme lähdekoodina. Valmiin tai ainakin toimintakelpoisen ohjelman näkee sitten esityspäivänä.

4 Projektiryhmän toiminta

Discord,-alusta, jolle bottia kehitettiin, toimi ryhmän pääasiallisena kommunikaatioväylänä sekä samalla testialustana. Tämä mahdollisti sen, että kaikki ryhmän jäsenet näkivät suoraan botin toiminnan eri kehityksen vaiheissa sekä mahdolliset ongelmatulostukset sekä mahdollisti välittömän keskustelun. Tekstikanaville pystyi kirjottamaan niin sanottuihin code blockeihin syntaksikorostettua koodia (syntax highlight), linkkaamaan kuvia, ja niitä pystyi käyttämään nopeasti pienien tiedostojen siirrossa. Määrätyille kanaville pystyi kanavien keskusteluhistorian takia säilömään projektin kannalta hyödyllisiä linkkejä ja materiaalia. Käytössä oli myös äänikanavat, joissa ryhmän jäsenet pystyivät tarvittaessa keskustelemaan. Tämä teki esimerkiksi debuggaamisesta jouhevaa ja reaaliaikaista, kun joku ryhmäläisistä apua tarvitsi.

Projektin versiohallinnassa käytettiin git-repositoriota GitHub-palvelussa. Jokainen kehittäjä jatkokehitti omaa osiotaan omassa haarassaan ja toimiva versio yhdistettiin master-haaraan “feature branch” -mallin mukaisesti. Haarat noudattivat lähtökohtaisesti ryhmän kesken sovittua tehtävänjakoa.

[versio 19.4.21]