Ylläpitodokumentti

Mozart

Helsinki 7.5.2006

Ohjelmistotuotantoprojekti

HELSINGIN YLIOPISTO
Tietojenkäsittelytieteen laitos

Kurssi

581260 Ohjelmistotuotantoprojekti (6 ov)

Projektiryhmä

Mikko Honkanen
Matias Kirvelä
Iikka Salmi
Reena Setälä
Mika Wahlroos

Ryhmän ohjaaja

Jaakko Saaristo

Asiakas

Kjell Lemström

Johtoryhmä

Juha Taina

Kotisivu

`http://www.cs.helsinki.fi/group/mozart/`

Versiohistoria

Versio	Päiväys	Tehdyt muutokset
0.1	15.4.2006	Kopioitu pohjaksi
0.9	7.5.2006	Katselmoitava versio
1.0	7.5.2006	Hyväksytty versio

Sisältö

1 Johdanto

Ylläpitodokumentissa käydään läpi tuotteen käyttämisen, ylläpitämisen ja ymmärtämisen kannalta tarkeimmät yksityiskohdat. Dokumentissa eritellään ryhmän toteuttamat luokat ja toiminnallisuudet valmiista koodista. Lisäksi käydään läpi projektin aikana ryhmälle tulleita kehitysideoita, sekä otetaan kantaa toteutuneisiin ja toteutumattomiin projektissa määriteltyihin vaatimuksiin. Dokumentissa esitellään myös lopullisessa tuotteessa olevat toiminnallisuuden rajoitukset sekä kehitysehdotukset toiminnallisuuden lisäämiseksi.

2 Sanasto

Firefox Mozilla-projektin kehittämä ilmainen web-selain.
Lisätietoja: `http://www.mozilla.com/firefox/`

IE Internet Explorer, Microsoftin web-selain.
Lisätietoja: `http://www.microsoft.com/ie/`

JMIR C-BRAHMS-tutkimusryhmän kehittämä musiikinhakutietorakenne.

JRE Java Runtime Environment, Java-ohjelmien suorittamiseen tarvittava ympäristö.
Lisätietoja: `http://java.sun.com/`

Konqueror Web-selain.
Lisätietoja: `http://www.konqueror.org/`

LaTeX Dokumenttien ladontajärjestelmä.
Lisätietoja: `http://www.latex-project.org/`

MIDI Musical Instrument Digital Interface on tiedonsiirtojärjestelmä, joka on suunniteltu välittämään viestejä sähköisten musiikkilaitteiden välillä. Tässä tapauksessa MIDI:llä tarkoitetaan Standard MIDI File-määritelmän mukaista tiedostoa. SMF-määritelmää ylläpitää MIDI Manufacturers Association.
Lisätietoja: `http://www.midi.org/`

Opera Web-selain.
Lisätietoja: `http://www.opera.com/`

PCM Pulse Code Modulation eli Pulssikoodimodulaatio on yksi sähköisistä menetelmistä koodata ääni-informaatiota.

PDF Portable Document Format on Adoben kehittämä PostScript-kieleen pohjautuva käyttöjärjestelmäriippumaton, siirrettävä tiedostomuoto.

pianoroll Musiikin esitysmuoto, jossa nuotit esitetään taulukossa peräkkäisinä palkkeina, joissa palkin sijainti pystysuunnassa taulukossa kuvaa nuotin korkeutta ja palkin pituus nuotin pituutta.

PostScript PostScript (PS) on sivunkuvauskieli, jota käytetään etenkin tulostettavien dokumenttien ulkoasun kuvaamisen.

Safari Applen kehittämä web-selain.
Lisätietoja: `http://www.apple.com/safari/`

XML-RPC XML-RPC on etäkutsuprotokolla, jonka avulla eri alustoilla pyörivät ohjelmistot voivat toimia yhdessä.
Lisätietoja: `http://www.xmlrpc.com/`

3 Asennusohje

Tässä luvussa kuvataan sovelluksen asentamiseksi tarvittavat toimenpiteet. Projektin lopputuotokset toimitetaan CD-ROM-levyllä, joka sisältää ohjelmiston lähdekoodin lisäksi valmiiksi käännetyn sovelluspaketin `dist/jmir-webdemo.war`.

Ohjeistus on laadittu Helsingin yliopiston tietojenkäsittelytieteen laitoksen (TKTL) tietotekniseen ympäristöön, missä laitoksen tunnuksella on mahdollista asentaa ja suorittaa omia Java-pohjaisia web-sovelluksia palvelimella `alkokrunni.cs.helsinki.fi`. Suoritusympäristön saattaminen käyttökuntoon on kuvattu osoitteessa `http://www.cs.helsinki.fi/compfac/servlets.html`.

3.1 Kääntäminen ja paketointi

Sovelluksen levityspaketti sisältää valmiiksi käännetyn sovelluspaketin `dist/jmir-webdemo.war`. Jos lähdekoodia tai muita sovelluspaketin sisältämiä tiedostoja muutetaan, sovellus täytyy kääntää ja paketoida uudelleen.

Sovelluksen juurihakemisto sisältää Ant-sovellukselle tarkoitetun ohjaustiedoston `build.xml`, jonka oletuskohde tuottaa onnistuessaan sovelluspaketin `dist/jmir-webdemo.war`.

Esimerkiksi TKTL:n servlet-ympäristössä käännös ja paketointi suoritetaan asettamalla Tomcat:n lisäykset luokkapolkuun ja suorittamalla Ant oletuskohteelle:

```
setup tomcat
/opt/ant/bin/ant
```

3.2 Asennus ja konfigurointi

Sovellus asennetaan kopioimalla sovelluspaketti web-palvelimelle ja luomalla konteksti, jolle asetetaan parametrina JMIR-palvelimen osoite. Tarvittavien toimenpiteiden yksityiskohdat vaihtelevat eri sovelluspalvelimilla.

Esimerkiksi TKTL:n servlet-ympäristössä sovelluspaketin kopiointi tapahtuu seuraavasti:

```
cp dist/jmir-webdemo.war ~/tomcat/webapps/
```

Sovelluksen konteksti konfiguroidaan XML-tiedostolla, jossa annetaan kopioidun sovelluspaketin polku ja JMIR-palvelimen osoite. Seuraavassa esimerkissä `USER` on korvattava ympäristön todellisella käyttäjätunnuksella ja parametrin `jmirServerURL` arvo viittaa kehityksen aikana käytettyyn palvelimeen.

```
<Context path="/tomcat/USER/jmir-webdemo"
         docBase="/home/USER/tomcat/webapps/jmir-webdemo.war">
  <Parameter name="jmirServerURL"
             value="http://db.cs.helsinki.fi:43434/" />
</Context>
```

Kontekstin konfiguraation muuttamisen jälkeen Tomcat on käynnistettävä uudelleen:

```
stop-tomcat
start-tomcat
```

4 Käynnistysohje

Sovelluksen käyttöliittymä saadaan asennuksen ja konfiguroinnin jälkeen esiin antamalla selaimelle seuraava osoite (rivinvaihto ei kuulu osoitteeseen):

```
http://db.cs.helsinki.fi/tomcat/USER/servlet/
jmir.webdemo.servlet.UserInterface
```

missä USER on servlet-ympäristön suoritukseen käytettävä käyttäjätunnus.

5 Toteutetut ja ulkopuoliset komponentit

Tämän luvun tarkoituksena on tehdä mahdollisimman selvä jako projektin toteuttamien ja projektin ulkopuolella toteutettujen, sovelluksessa käytettyjen osien välillä. Pääosin jako on Javan pakettirakenteen mukaisesti selvä, mutta joidenkin luokkien kohdalla projekti on muokannut ulkopuolella toteutettua koodia projektin tarpeisiin soveltuvaksi.

5.1 Projektin toteuttamat Java-luokat

Kaikki paketin `jmir.webdemo` alla sijaitsevat Java-luokat ovat projektin toteuttamia. Paketti sisältää seuraavat luokat:

jmir.webdemo.applet Keyboard, MidiPlayer, MidiPlayerListener, MozartApplet, NoteDurationSelector, ResultTable, Sender

jmir.webdemo.common JMIRClient

jmir.webdemo.servlet JMIRProxyServlet, UserInterface

5.2 Projektin muokkaamat Java-luokat

Osa ohjelmistossa käytettävistä Java-luokista on projektin ulkopuolella toteutettuja, mutta projekti on tehnyt niihin muutoksia. Seuraavassa luetellaan muutetut luokat ja projektin niihin tekemät muutokset.

jmir.gui.ScoreEditor Lisätty metodit: addNote(), addRest(), emptyScore(), getCursorPosition(), getLastEndingNote(), getLastNote(), getHighest(), getLowest(), getScore(), moveNotes(), removeEndingAt(), setCursorAtEnd(), setCursorPosition().

Lisäksi muokattiin luokan konstruktoria asettamaan editorille hiirikuuntelija myös silloin, kun varsinainen muokkaus ei ole sallittu. Tynkämetodiin getScoreLength() lisättiin toteutus ja muutettiin se palauttamaan double-tyyppinen paluuarvo. Myös hiirikuuntelijalle, joka aiemmin koostui pelkistä tynkämetodeista, lisättiin osittainen toteutus.

Metodissa setScore() näkymän koon päivitys metodilla updateViewSize() otettiin pois käytöstä, koska sen käyttö yhdessä uusien lisättyjen metodien kanssa aiheutti ongelmia.

5.3 Projektin ulkopuolella toteutetut osat

Ohjelmiston toteutuksessa hyödynnetään useita projektin ulkopuolella toteutettuja osia.

Seuraavat luokat on toteutettu C-BRAHMS-tutkimusryhmän työnä ja projekti on käyttänyt niitä sovelmallisen käyttöliittymän toteutuksessa ilman muutoksia lähdekoodiin:

jmir InvalidValueException, Log

jmir.dataformat Instrument, IntervalComparator, IntervalSortedMap, IntervalTreeMap, Measure, MonoNoteList, Note, NoteArray, NoteComparator, NoteData, NoteList, NoteListener, NoteSet, Part, QueryResult, QueryResultMatch, Score, SimpleIntervalComparator, TextNotation

jmir.gui MediaPlayerListener

jmir.sound AudioRecorder

JMIR-palvelimen kanssa kommunikointiin käytetään Apachen XML-RPC-toteutusta (`http://ws.apache.org/xmlrpc/xmlrpc2/`).

HTML-lomakkeella lähetettyjen tiedostojen käsittelyyn käytetään Jakarta Commons -projektin FileUpload -komponenttia (`http://jakarta.apache.org/commons/fileupload/`.

6 Tunnetut viat

Tässä luvussa luetellaan ohjelmiston tunnetut viat.

6.1 Pianorollin vaakavierityspalkki

Pianorollin näyttöalueen leveys ei alustamisen jälkeen mukaudu melodiaan tehtyihin muutoksiin.

Pianoroll on toteutettu ScoreEditor-luokan ilmentymänä. Luokan metodin updateViewSize() käyttäminen pianorollin koon päivittämiseen esimerkiksi nuotin lisäämisen jälkeen ei ole ilmeisen ongelmatonta. Muuta yksinkertaista tapaa koon päivittämiseen ei liene. Selvää syytä ongelmiin ei ole löydetty, kuten ei myöskään toimivaa ratkaisua. Ongelma saattaa liittyä luokan sisäisen kirjanpidon päivittämiseen, kun nuotteja ja taukoja lisätään.

Vaakavierityspalkki tulee käyttöön, jos pianorollille asetetaan melodia sovelman ulkopuolelta esimerkiksi näytetiedostolla, jolloin koko melodia asetetaan kerralla ScoreEditor-luokan setScore()-metodilla. Tällöinkään pianoroll ei kuitenkaan kasvata kokoaan, jos melodian asettamisen jälkeen nuotteja lisätään koskettimistolta.

Lisäksi vierityspalkki on yhteinen koskettimistolle ja pianorollille, joten vieritettäessä pianorollia oikealle koskettimisto katoaa vastaavasti näkyvistä. Toivottavaa olisi, että pianoroll olisi omassa vieritettävässä komponentissaan. Tätä yritettiin toteuttaa, mutta ilmeisesti Javan Swing-käyttöliittymäkehyksen ongelmien takia se ei onnistunut.

Intuitiivinen lähestymistapa asiaan olisi sijoittaa ScoreEditor-luokan ilmentymä vain vaakasuunnassa vieritettävän JScrollPane-komponentin sisään, minkä jälkeen koskettimistoa edustava Keyboard-luokan ilmentymä ja pianorollin vieritysalusta sijoitettaisiin yhteiseen säiliökomponenttiin (Container). Tämä puolestaan sijoitettaisiin ainoastaan pystysuunnassa vieritettävään JScrollPane-komponenttiin, jotta koskettimistoa ja pianorollia voitaisiin vierittää toisiinsa kiinnitettyinä pystysuunnassa.

Nykyinen ratkaisu on toteutettu käyttäen vain jälkimmäistä vieritysalustaa, koska kahden sisäkkäisen säiliökomponentin sijoittaminen JScrollPane-komponentin sisään ei toimi.

6.2 Haun säikeistäminen

Sovelmalla nauhoitetun ääninäytteen lähettäminen palvelimelle muunnettavaksi symboliseen nuotinnosmuotoon (transkriptio) tehdään käyttöliittymästä erillisessä omassa säikeessään, jotta käyttöliittymä ei jää jumiin siksi aikaa, kun näytettä siirretään verkon yli.

Hakua nuotinnosmuodossa olevan melodian perusteella ei kuitenkaan tehdä omassa säikeessään, joten haun aikana käyttöliittymä on jumissa. Käytännössä tämä ei yleensä ole ongelma, koska hakupyynnössä ja -tuloksissa on vähän dataa eikä siirrossa yleensä kestä kauan, joten useimmiten asia on käyttäjän kannalta lähes huomaamaton. On kuitenkin mahdollista, joskin harvinaista, että jostain syystä hakutulosten saaminen kestää pitkäänkin, ja palvelin voi mahdollisesti jopa kaatua sovelman lataamisen ja melodian haun välisenä aikana, jolloin käyttöliittymä jäisi jumiin siihen asti, kunnes verkkojärjestelmä lakkaisi yrittämästä saada palvelimeen yhteyttä.

Tämän vuoksi olisi hyvä siirtää myös varsinainen hakuoperaatio sovelmassa erilliseen säikeeseen. Tämän toteuttamisessa voisi käyttää anonyymiä luokkaa MozartApplet-luokan sisällä tai vastaavaa tapaa kuin on käytetty transkription toteuttamiseksi omassa säikees-

sään. Tällöin olisi asianmukaista käyttää transkription keskeyttämiseen käytettävää painiketta myös haun keskeyttämiseen mahdollisessa ongelmatilanteessa.

6.3 Käyttäjälle esitettävät virheilmoitukset

Osa loppukäyttäjälle esitettävistä virheilmoituksista sisältää käyttäjän kannalta merkityksettömiä Java-luokkien nimiä. Esimerkiksi tunnistamattomasta tai virheellisestä ääninäytetiedoston muodosta ilmoitetaan seuraavan esimerkin mukaisesti:

```
Error: org.apache.xmlrpc.XmlRpcException:
Unsupported audio format or corrupted data
(java.io.EOFException)
```

Luokkien nimet asetetaan virheilmoitukseen ennen kuin ilmoitus saadaan projektissa toteutetun koodin käsiteltäväksi. Esimerkissä jälkimmäinen, suluissa ilmoitettu EOFException, on peräisin JMIR-palvelimelta, luokan JMIRServer metodilta transcribe().

Lisäksi joistain sovelman virhetilanteista tulostuu ilmoitus vain Java-konsoliin, jota loppukäyttäjillä ei yleensä ole näkyvillä. Tällaisia virhetilanteita voi syntyä Sender-luokan run()-metodin suorituksen aikana. Enimmäkseen poikkeustilanteet on pyritty käsittelemään keskitetysti sovelman pääluokassa (MozartApplet), jotta niistä voidaan tiedottaa loppukäyttäjälle, mutta poikkeuskäsittelyn siirtäminen pois Sender-luokan run-metodista edellyttäisi muutoksia sovelman luokkarakenteeseen.

Konsoliin tulostetaan tarkoituksella virheenjäljitystä varten lisäksi tarkempia teknisiä yksityiskohtia myös niistä virheistä, jotka ilmaistaan loppukäyttäjälle yksinkertaisemmassa muodossa.

6.4 Pianorollin kohdistaminen melodian asettamisen jälkeen

Oletuksena pianorollin näkyvä osa on kohdistettu siten, että se näyttää noin puolitoista oktaavia koskettimia ja nuotteja lähes asteikon keskeltä.

Kun editorille asetetaan melodia muutoin kuin syöttämällä yksittäisiä nuotteja koskettimistolta, editori asetetaan automaattisesti pystysuunnassa näyttämään kohtaa, jolla nuotit näkyvät.

Tämä toimii, mutta on toistaiseksi toteutettu hieman hankalasti. Editorin näkymän asettamiseksi oikeaan kohtaan on tiedettävä editorin näkyvän alueen korkeus. Koska ominaisuutta tarvitaan jo sovelman käynnistymisen yhteydessä, ja koska tässä vaiheessa käyttöliittymän komponenttien koot eivät vielä ole täysin selvinneet, kokoa ei voi selvittää käyttämällä JViewport-luokan metodia getExtentSize(). Tämän vuoksi koko on toistaiseksi asetettu kiinteästi, ja sitä on muutettava, jos sovelman tai muokkauskomponenttien koko muuttuu olennaisesti.

6.5 Viive koskettimiston äänissä

Koskettimen painalluksen ja sen seurauksena soitettavan äänen välillä on testatuissa ympäristöissä esiintynyt huomattava, jopa noin 400 - 500 millisekunnin viive. Viiveen syytä ei tiedetä, mutta se liittyy ilmeisesti joko JRE:n tai käyttöjärjestelmän MIDI-järjestelmään ja sen käsittelyyn. Viive on havaittu vain käytettäessä sovelmaa Linux-alustalla.

Koskettimistolla tapahtuvat hiiren klikkaukset käsitellään luokan MozartApplet metodissa mouseClicked(). Jos koskettimiston äänet on sallittu, metodi luo yhden nuotin Score-olion, joka välitetään parametrina metodille playMidi(). playMidi() kutsuu Score:n metodia exportMidi() ja välittää tuloksen InputStream-oliona MidiPlayer -olion metodille play(). MidiPlayer-luokan play()-metodi luo parametrista MIDI-sekvenssin ja kutsuu sekvensserin start()-metodia.

Viive on ongelmallinen lähinnä koskettimiston äänten toistamisessa, koska tällöin se koskee jokaista koskettimen napsautusta ja vastaavan äänen soittamista. Ongelman voisi mahdollisesti kiertää tai sen vaikutusta vähentää, jos sekvensseriä ei alustettaisi uudestaan joka kerta, kun soitto aloitetaan.

7 Kehityskohteet

Tässä luvussa käsitellään projektin aikana esitettyjä, mutta toteuttamatta jätettyjä ohjelman toimintoja ja ominaisuuksia.

7.1 Nuotin korkeuden muuttaminen

Pianorolliin voisi toteuttaa toiminnon, jolla nuotin korkeutta voisi muuttaa tarttumalla nuottia vastaamaan palkkiin ja pudottamalla sen halutulle korkeudelle (drag and drop).

7.2 Viimeisimmän muokkauksen peruutus

Muokkaustoimintoihin voisi liittää peruutustoiminnon (undo), joka palauttaa tilanteen viimeisintä muokkaustoimenpidettä edeltävään tilanteeseen.

7.3 Hakutulosten kuuntelu

Hakutuloksen yhteydessä välitetään nykyisin MIDI-tiedoston URL, mutta sovelmallisessa käyttöliittymässä ei ole keinoja osoitteesta ladattavan kappaleen kuuntelemiseen. Sovelmaan voidaan haluttaessa liittää musiikkisoitin ja JMIR-palvelimen rajapintaa laajentamalla hakutulosten yhteydessä voidaan välittää musiikin eri esitysmuotoja.

7.4 Tilarivin taustaväri

Sovelman alalaidassa sijaitsevan tilarivin taustaväri on valkoinen, mikä saattaa harhauttaa käyttäjän luulemaan, että kyseessä on muokattava tekstikenttä. Selkeyttä voisi parantaa yksinkertaisesti muuttamalla tilarivin taustaväri harmaaksi.

7.5 Transkriptio asiakaspäässä

Nykyisessä toteutuksessa transkriptio (ääninäytteen muuntaminen symboliseen muotoon) suoritetaan JMIR-palvelimella, joka käyttää Gentin yliopiston MAMI-projektin kehittämää C/C++ -kirjastoa. Hitaammilla tietoliikenneyhteyksillä tyypillisen hyräilynäytteen siirtäminen sovelmalta palvelimelle saattaa kestää häiritsevän pitkään, eikä siirron etenemistä voida seurata käyttöliittymässä.

Mikäli sopiva Java-kirjasto löydetään, transkription suorittaminen asiakaspäässä olisi luultavasti parempi vaihtoehto.

7.6 Tempon muuttaminen

Melodian toistossa ja muunnoksessa tekstimuotoon käytetään tempoa 120 bpm. Käyttöliittymään voisi lisätä tempon valinnan ennalta määriteltyjen vaihtoehtojen mukaan.

7.7 Nauhoituksen ilmaisin

Ääninäytettä nauhoitettaessa käyttäjälle olisi hyvä ilmaista esimerkiksi oskilloskoopilla, että nauhoitin vastaanottaa signaaleja. Tällä hetkellä käyttäjällä ei ole mitään mahdollisuuksia tietää, nauhoittaako sovelma käyttäjän hyräilyjä ja jos nauhoittaa, niin millä äänenvoimakkuudella.

7.8 Näkymän ulkopuolisten nuottien ilmaisin

Jos pianorollin näkymän ulkopuolella on nuotteja, siitä voisi ilmoittaa käyttöliittymässä.

7.9 Toistettavan kohdan ilmaisin

Melodiaa toistettaessa pianorollin kursori voisi liikkua toistettavan kohdan mukana. Tämä voitaisiin toteuttaa esimerkiksi laajentamalla MidiPlayerListener-rajapintaa määrittelemään myös metodin soiton etenemisen kuuntelemiseen ja toteuttamalla tarvittava toiminnallisuus MidiPlayer-soitinluokassa.

7.10 Viivasto pianorollilla

Pianorollin oikeassa laidassa esitettävät nuottipalkit olisi helpompi yhdistää vasemman laidan koskettimiin, jos palkit sijoitettaisiin koskettimia mukailevalle vaakasuoralle viivastolle. Eräs tapa viivoituksen toteuttamiseksi olisi lisätä kevyt tummennus mustia koskettimia vastaaville viivoille.

7.11 Nuotin kestovalintojen lisääminen

Nuotin kestovalintoja halutaan ehkä tuleivaisuudessa esittää useampia kuin tällä hetkellä. Nuottien kestovalinnat määritellään MozartApplet.javassa ja nuotti-ikonit on haettu Wikipediasta osoitteesta `http://http://en.wikipedia.org/wiki/Note_value`. Ne ovat PNG-formaatissa läpinäkyvällä taustalla ja niiden koko on 30x45 pikseliä. Kuvat tulee sijoittaa hakemistoon `data/applet/images`.