TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Martin Ojamets 158479IAPB

MULTIMEEDIA RAKENDUS "KLAVER" ANNETAJATE AUSTAMISEKS JA KOLLEKTIIVSEKS HELILOOMINGUKS

Bakalaureusetöö

Juhendaja: Jaak Henno
PhD
Dotsent

Tallinn 2018

Autorideklaratsioon

Kinnitan, et olen koostanud antud lõputöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on töös viidatud.

Autor: Martin Ojamets

21.05.2018

Annotatsioon

Käesoleva töö eesmärk on arendada virtuaalne klaver, kus saavad mängida nii harrastajad kui ka algajad. Töös realiseeritakse kaks põhi komponenti. Esiteks luuakse veebileht, kus oleks võimalik klaverit mängida ükskõik mis nuti seadmes, olgu selleks kas telefon, tahvelarvuti või lauaarvuti. Teise komponendiks on veebilehele juurde teha rahvahanke meetod laulude loomiseks.

Peale selle sai veel töös võrreldud erinevaid viise, kuidas oleks üldse võimalik sellist rakendust arendada. Lisaks on veel välja toodud nende erinevused ja eripärad.

Antud töö teeb võimalikuks uue HTML versiooni väljatulek, mis teeb mobiilsetele seadmetele arendamise palju kergemaks ning lisab ka muid uuendusi, mis enne oli puudu. Üheks suurimaks muudatuseks on just audio mängimine otse läbi HTML-i. Rakendus luuakse *canvas* elemendiga ning kasutatakse puhast JavaScripti, et luua valmis klaver. Serveris on kasutusel MySQL andmebaas ja PHP ühendus vastava andmebaasiga. Andmebaasis hoitakse noote, laule, kasutajaid, kui ka laulu noote.

Töö on ülesse laetud renditud serverisse mille IP aadressiks on: http://209.250.228.36/

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 21 leheküljel, 6 peatükki ja 7 joonist.

Abstract

Multimedia Application "Piano" for honoring donators and collective composition

The purpose of this thesis is to showcase a virtual piano, which both professionals and amatures can experience. The work is divided up into two main components. The first is a website which allows the user to play the piano in every smart device that they may own, be it a phone, a tablet or a computer. The second component is to add a crowdsourcing method for collective composition.

Furthermore, there are several different ways under comparison, all of which could have been used to create this application. Additionally, their differences and specialties have been brought out.

The study is made possible due to the release of a new version of HTML several years ago, which makes mobile design a lot easier and adding quite a few new features, which were missing previously. One of the major ones being able to play audio straight through HTML. The application is written using the *canvas* element and pure JavaScript is used alongside it to render and give the piano it's functionality. On the server there is a MySQL database alongside with PHP to make connection with said database. There is information about the notes, songs, users and notes in songs.

The work is publicly viewable at: http://209.250.228.36/

The thesis is in Estonian and contains 21 pages of text, 6 chapters and 7 figures.

Lühendite ja mõistete sõnastik

Ajax	*Asynchronous JavaScript And Xml*
CSS	*Cascading Style Sheets,* Astmelised stiililehed
DOM	*Document Object Model,* Dokumendi objektimudel
HTML	*Hyper-Text Markup Language*, Hüperteksti märgistuskeel
Hz	Hertz
JS	JavaScript
JSON	*JavaScript Object Notation*
MIDI	*Musical Instrument Digital Interface*, MIDI
WWW	*World Wide Web,* Veeb
XSS	*Cross-Site Scripting,* Murdskriptimine

Sisukord

1 Sissejuhatus 8
2 Millest see töö räägib 10
2.1 Praktiline tähendus 10
2.2 Probleem 10
2.3 Lahendus 11
3 Multimeedia rakenduse võimalused 12
3.1 HTML5 12
3.1.1 Canvas 13
3.1.2 Divid ja CSS 15
3.2 Flashi kasutamine sarnastes rakendustes 17
3.3 Java kasutamine sarnastes rakendustes 18
4 HTML-i heli võimalused 19
4.1 SoundFont 19
5 Kasutaja sisendi käsitlemine 21
6 Rahvahange 24
6.1 Rahvahanke populaarsus 24
6.2 Rahvahanke näidised 25
6.3 Rahvahanke teostus töö lahenduses 25
7 Kokkuvõte 28
Kasutatud kirjandus 29

Jooniste loetelu

Joonis 1. HTML5, CSS ja JS logo 13
Joonis 2. Brauserite toetus *canvas* elemndile [3] 13
Joonis 3. Klahvi objektide korduvalt loomine 14
Joonis 4. Valmis joonistatud klaver 14
Joonis 5. Divituse näide [4] 16
Joonis 6. Flashi hoiatus kasutaja nõusoleku küsimisel 18
Joonis 7. JavaScripti *eventi* elutsükkel [10] 22
Joonis 8. Laulu valik mida muuta 26
Joonis 9. Klaver koos kahe juurde mängitud noodiga 26
Joonis 10. Lahenduse avaleht koos annetajatega 27

1 Sissejuhatus

Tehnoloogia kiire areng on toonud kaasa tohutus koguses uusi asju, mida kasutatakse igapäevaselt ning lisaks on see avanud uksi, mis eelnevalt tundusid kinni. Kuuekümnendatel hakati looma tänapäeva Internetti, kuid massideni jõudis see alles HTML-i tulekuga üheksakümnendatel. Selle tulekuga hakkasid arendajad usinalt tööle, algselt küll tehes staatilisi veebilehti, kuid lõpetades keerukate veebirakendustega.

HTML-is sellel ajal puudusid heli mängimise võimalused, mis tõi kaasa kolmanda programmi pistikud, olgu selleks kas Flash või Java Applet. Need pakkusid palju rohkem võimalusi veebirakenduste loomiseks kui ainult HTML, mis oli staatiliste asjade näitamiseks. See tõttu hakatigi arendama Flashi ja Java rakendusi, kuid kahjuks hakkasid need suhteliselt kiiresti vigadest märku anda. Antud programmid vajasid pidevalt turvariskide uuendusi ning kasutasid suuremas koguses ressursse.

Enamik virtuaalseid klavereid mis sellel ajal eksisteerisid olidki just tehtud Flashiga. Tehnoloogia kiire arengu tõttu hakkasid lisandid kaotama enda usaldust ning neid hakati järjest vähem ja vähem toetama brauserites. See põhjustabki olemasolevat probleemi, puudus virtuaalsest klaverist, mida igaüks saaks kasutada vabalt oma veebilehel.

Töö eesmärgiks on valmis saada uue standardi põhiselt, HTML5, klaver mis töötaks kõikidel seadmetel, mis võimaldavad Internetis minna WWW-lehtedele. Lisaks sellele tuleb kindlasti silmas pidada, et telefonil ja tahvelarvutil oleks võimalik kasutada rakendust kuna tänapäeval on pea igal inimesel olemas oma telefon.

Peale selle on töös veel plaanitud arendada rahvahanke meetod, mis võimaldab kasutajatel mängida paar nooti korraga laulu luues. Antud lisa peaks rohkem kasutajaid ligi tõmbama, sest siis tekib kasutajal tunne, et ta on osa millestki suuremast, kui tema ise.

Antud töös on veel võrreldud kahte erinevat viisi, kuidas oleks võimeline teha virtuaalset klaverit ning tuuakse välja mõlema eelised ja miinused. Lisaks on veel võrdluseks toodud välja Flashi ja Javaga tehtud näiteid.

2 Millest see töö räägib

2.1 Praktiline tähendus

Töö autor on huvitatud veebirakenduste loomisest pikemat aega, mitte ainult koolitööde jaoks, vaid ka reaalsetele klientidele, kes ootavad tervet töötavat asja, mitte tundides antud ülesandeid.

Probleem, mida siin töös käsitletakse on reaalne ning põhineb tehnoloogia kiirele arengul, mis muudab vanad rakendused kasutuskõlbmatuks.

Peale selle, et probleem on praktiline, peaks Valga Muusikakool selle lahenduse võtma kasutusele enda veebilehel, et koguda annetusi kontserdiklaveri ostmiseks.

2.2 Probleem

Selle töö suurimaks probleemiks on vähene virtuaalsete muusikariistade valik, mida saaks kasutada igal nutiseadmel, olgu selleks kas telefon, tahvelarvuti või lauaarvuti. Selles töös on vaadatud just virtuaalset klaverit.

Enamik muusikariistad, mis hetkel eksisteerivad, on tehtud Flashiga, mida paljud telefoni, tahvelarvuti ja lauaarvuti brauserid enam ei toeta. Peale selle on ka Java Appleti toetus hakanud muutuma problemaatilisemaks, plaanidega see teha vananenuks Java 9-s ning tulevikus ära kaotada. [1]

Olenemata sellest, et üks hästi tehtud virtuaalne klaver on olemas, milleks on virtualpiano.net, siis leiduvad ka sellel mõned asjad, mis võiksid olla teisiti. Sellel on ainult viis oktaavi, alustades C_2 kuni B_6. Tänapäeva klaveritel on seitse oktaavi ja paar lisaklahvi. [2]

Kõige lõpuks on veel Valga Muusikakoolil soov koguda annetusi, et osta endale kontserdiklaver. Virtuaalne klaver oleks nende veebilehel üleval, et inimesed saaksid sellega mängida ning loodetavasti ka annetusi teha.

2.3 Lahendus

Töö lahenduseks on teha HTML5-l põhinev virtuaalne klaver WWW-lehel, mis töötaks igal seadmel, olgu selleks kas tavaline lauaarvuti, tahvelarvuti või telefon. See annaks võimaluse nii harrastajatel kui ka algajatel mängida klaverit enda käeulatuses.

Lahenduse juures on just jälgitud seda, et klaver oleks mängitav ka telefonidel, mitte ainult lauaarvutil. Lisaks sellele on veel võimalus rahvahanke meetodil teha laule, mängides ise vaid paar nooti korraga. Hetkel on antud töö ülesse laetud renditud serveril aadressil: http://209.250.228.36/. Töös on veel peale HTML5 võimaluste kasutatud ka andmebaasi, kuhu salvestada vastavad lood ning nende noodid ja kasutajad.

3 Multimeedia rakenduse võimalused

Ennem kui me klaveri loomisel kasutatud lahenduste detailidesse läheme, tuleks natukene rääkida võimalustest, kuidas oleks võimalik probleemi lahendada. Lahenduse juures peab kindlasti silmas pidama, et rakendust saaks kasutada ka telefonides ja tahvelarvutites, mitte ainult lauaarvutites nagu paljud olemasolevad lahendused on tehtud. Seetõttu ei vaata nii Flashi kui ka Java lahendusi detailsemalt, kuna nende tugi on tänapäeva brauseritest hakanud vähenema ning paljud telefonid ei toeta neid üldse. Peale selle proovime aru saada, milline lahendus on parem ja kuidas seda hästi kasutada.

3.1 HTML5

Internet, mida hakati kasutama juba viiekümnendatel, on tänaseks muutunud tohutult ning kui neid võrrelda, siis oleks raske uskuda, et ühest sai teine. Sellel ajal oli Interneti kasutamiseks omad standardid, kuid selle levimine massidesse põhjustas vajadust luua kindlat standardit, mida oleks kergem kasutada, et veebis ringi seigelda.

Tuhande üheksasaja üheksakümnendate alguses sai Tim Berners-Lee valmis HTML-i, mis sai aluseks tänapäeva veebile. See lubas HTML dokumente veebist alla laadida ja neid näidata kasutajale brauseris. Tänapäevaks on HTML saanud mitu uuendust, millest esimesed neli suurt uuendust said millenniumi vahetuseks tehtud. HTML4, edaspidi lihtsalt HTML, on olnud suureks standardiks tuhande üheksasaja üheksakümne üheksandast aastast. Uuenduste puudumine lubas erinevatel firmadel arendada oma tehnoloogiaid veebirakendusteks, nagu näiteks Flash, Java Web Plugin ja mõni veel. Viimasel ajal on need muutunud ebaturvalisemaks ning neid on hakatud eemaldama veebist. Sellele on kindlasti kaasa aidanud ka HTML5, mis tuli kasutusele kahetuhande neljateistkümnendal aastal.

HTML5 tõi endaga kaasa suuremaid muudatusi, mis võimaldab luua veebi rakendusi turvalisemalt ning koos mobiilse toega, mida Flashi ja Java Web Pluginid ei võimalda. Käesoleva töö lahendus kasutab ühte nendest võimalustest, milleks on *canvas* element,

mis võimaldab JavaScriptiga sellele peale joonistada. Teine võimalus, mida töös vaadatakse, on *div* elementide kasutus koos CSS-iga.

Joonis 1. HTML5, CSS ja JS logo

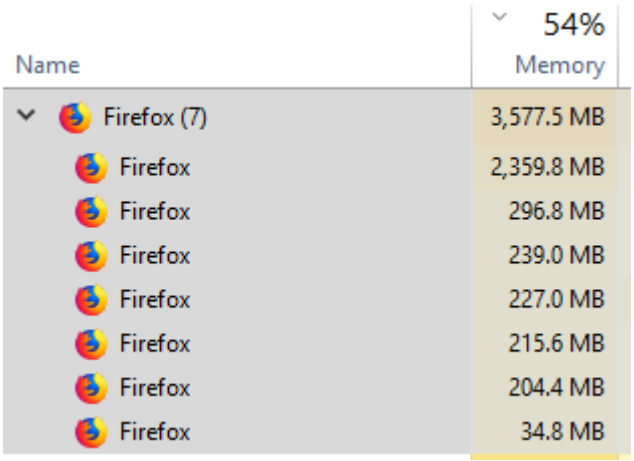
3.1.1 Canvas

Nagu juba eelnevalt mainitud, siis käesoleva töö lahenduses on kasutusel *canvas* element, mis võimaldab arendajal ise selle peale joonistada pea ükskõik mida, alustades paarist lihtsast joonest kuni keeruliste valemiteni välja. See on just üks suuremaid eeliseid eelmise HTML-i versiooni üle, kus oli vaja tuhandeid väikeseid elemente, et valemi joonistamine näeks korralik välja. Kahjuks on *canvas* elementi võimalik kasutada ainult vastavates brauserites, mis on välja toodud Joonis 2.

Element					
<canvas>	4.0	9.0	2.0	3.1	9.0

Joonis 2. Brauserite toetus *canvas* elemndile [3]

Samas võib *canvas*-e kasutamine olla ohtlik, kuna selle peale peab joonistama JavaScriptiga, kus võib mäluleke tekkida suhteliselt kergesti, mis viib olukordadesse kus brauser kasutab gigabaitides mälu nagu on näha Joonis 3. See omakorda viib kas brauseri maha aeglustumiseni või lausa kokku jooksmiseni.

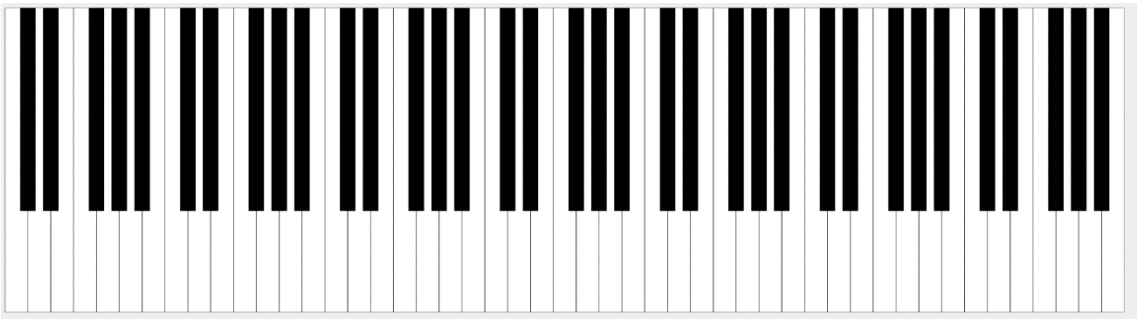


Joonis 3. Klahvi objektide korduvalt loomine

JavaScripti kasuks räägib suhteliselt hea prügi-korjaja, (inglise keeles *garbage-collector)* mis vabastab mitte kasutusel oleva mälu.

Käesolevas töös on kasutusel *canvas*-ele joonistamiseks Path2D objekt, mis lubab iga klahvi eraldi joonistada ja anda neile ka muid väärtusi, näiteks joonistamise alguskoordinaadid, mängitava klahvi noot ja klahvi värv. Nagu eelnevalt juba mainitud, tuleb *canvas* elemendile peale joonistada. Selleks on vaja teada iga klahvi alguskoordinaate ja nende mõõtmeid. Vajalik informatsioon on alguses välja arvutatud, võttes kasutusele brauseri akna mõõtmed ning uuendades seda, kui selle suurust muutadetakse.

Joonistamise järjekord on tähtis, kuna *canvasele* joonistatakse alati peale, nagu maalideski. Kõigepealt saab valmis joonistatud valged klahvid ja alles siis mustad klahvid, pärast mida ongi klaver joonistatud nagu on näha Joonis 4Joonis 4. Klahvide koordinaadid tuleks iga klahvi jaoks eraldi arvutada, sest ilma selleta asetseksid klahvid kõik üksteise peal ning näha oleks ainult viimane klahv.



Joonis 4. Valmis joonistatud klaver

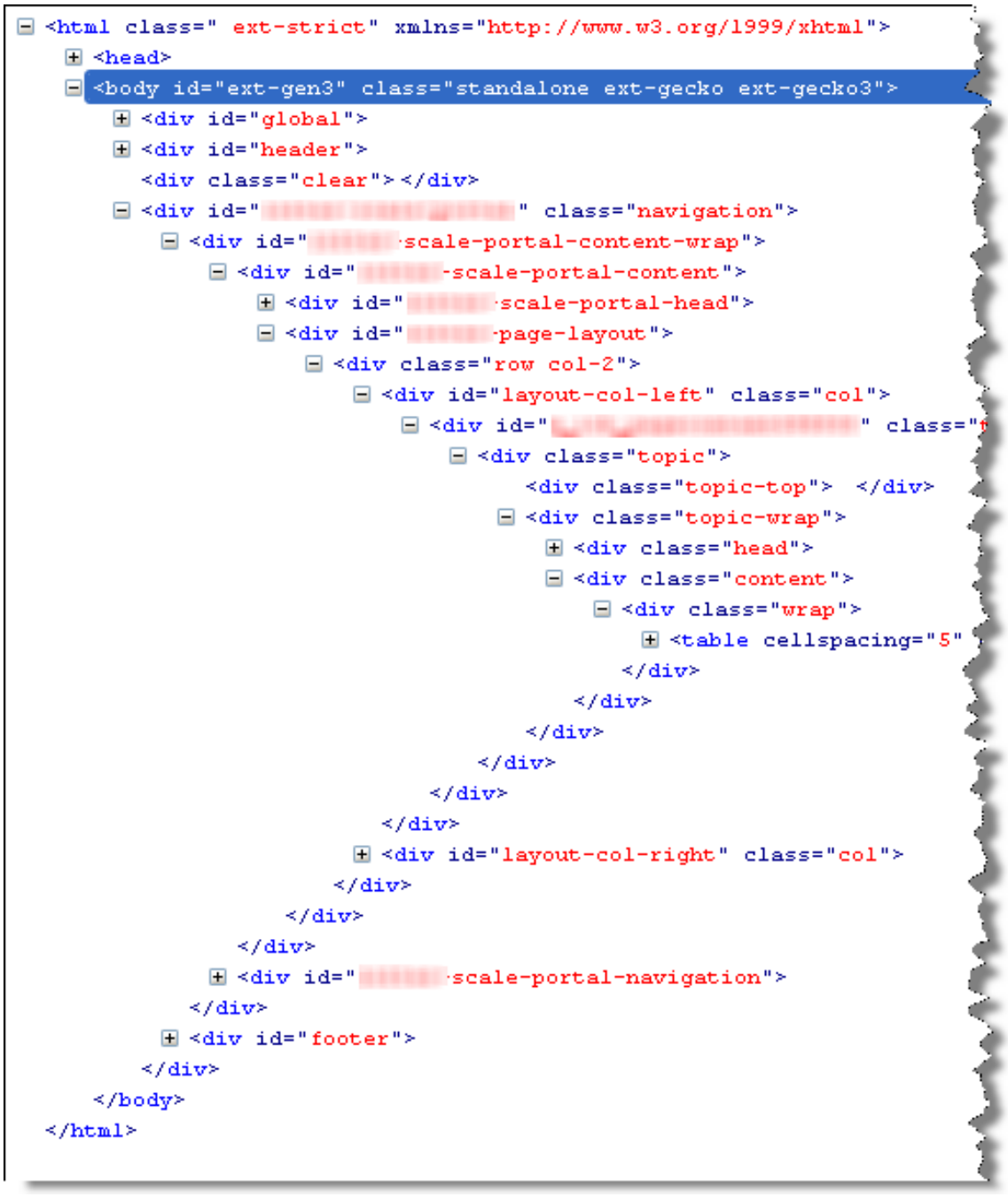
Peale selle on veel ära märgitud igal klahvile ka oma värv, mis muutub vastavalt olukorrale. Olgu selleks kas klahvi allavajutus või kursori liigutamine nende kohal. Selleks, et klahvi värv määrata, on kasutatud *createLinearGradient* funktsiooni, mis võimaldab teha sujuvat üleminekut ühelt värvilt teisele. Värvi muutmisest vastavalt kasutaja sisendile räägitakse lähemalt töö viiendas peatükis.

3.1.2 Divid ja CSS

Teine võimalus klaveri loomiseks, mida küll siin töös ei rakendatud, oleks kasutada *div* elemente ning kujundamine CSS-iga. Sellel lahendusel eelised, mida töö autor on katsetanud vähesel viisil, kuid otsustas ikkagi *canvas*-e kasuks.

Div-ide kasutamine oleks sarnane sellele stiilile, mida on kasutatud HTML-i loomisest. Iga klahv tuleb teha individuaalselt *div* elemendina ning see ära kujundada, andes sellele klahvi pikkuse ja laiuse protsendilistes väärtustes terve brauseri akna suhtes. Seda just sellepärast, et rakendus töötaks ka telefonis, mille resolutsioonid on erinevad laua- või tahvelarvuti omast. Samas pole vaja välja arvutada algus koordinaate, sest elemendid on juba olemuselt üksteise järel.

Samas võib selline lahendus viia olukorrani kus on üks *div* element teise sees ja nii sügavale minnes võib tekkida olukord, kus arendaja ei saa korralikult koodi vaadata, ilma kõrvale kerides. Sellist olukorda, kus on liigsete div-ide kasutamine, kutsutakse „Divitis“-eks ning selle arendamine või sellest arusaamine, on kordades raskem kui selle puudumisel. Sellist olukorda võib näha, kui vaadata Google Drive-i HTML lähtekoodi, mis on sarnane Joonis 5, kuid halvem. Kasutaja jaoks ei muuda see küll midagi, aga see teeb elu raskemaks just arendajale.



Joonis 5. Divituse näide [4]

Olenemata sellest, et see võib olla halb disaini harjumus, on sellisel lahendusel omad eelised *canvas*-e üle. Üks suurimaid on just kasutaja sisendi kontrollimine. Sellest räägitakse töös lähemalt andud töö viiendas peatükis. Peale selle on HTML5 erinevused suhteliselt väikesed, võrreldes eelmise versiooniga, mis võimaldab aega kokku hoida uuema tehnoloogia sügavuti õppimisest, kiirendades arendus protsessi.

Teine eeliseid *canvas*-e kasutamise üle on just see, et rakendus laeb kiiremini. See on sellepärast, et HTML faili brauseris näitamiseks tuleb see eelnevalt alla laadima, mis võtab rohkem aega olenevalt elementide arvust.

Samas võrreldes *canvas*-ega peab ka sellisel lahendusel olema JavaScripti tugi nootide mängimiseks ja kasutaja sisendi kontrollimiseks, kus võivad ka tulla olukordi mälulekete jaoks, kuid teistel põhjustel. See näitab, et HTML5 tulekuga on vaja kasutada JavaScripti suhteliselt palju rakenduste loomiseks.

3.2 Flashi kasutamine sarnastes rakendustes

Töö raames vaatame ka teisi lahendusi, mida kas võiks kasutada või on eelnevate lahenduste juhul kasutatud. Üks populaarsemaid tehnoloogiaid millega on tehtud nii mänge kui ka rakendusi veebis oli just Flash. See sai alguse aastal tuhat üheksasada üheksakümmend kuus ajaks, kui HTML oli juba rohkem levinud ning tekkis vajadus rohkem dünaamilistele lehtedele, kui staatilistele. See tõi kaasa video, animatsiooni ja kasutatavuse, et teha rakendusi, mis omal ajal töötaksid igal seadmel.

Flash leidis kasutust paljudes populaarsetes kohtades, üks suurimaid oli just YouTube, mis kasutas Flashi, et näidata videoid oma veebilehel. Kahjuks ei jäänud see kauaks kestma. Esimene märk sellest, et Flash hakkab vaikselt välja surema oli juba aastal kaks tuhat seitse, kui Steve Jobs ei toetanud enam Flashi enda uusimal telefonil. Sellel ajal juba jutt levis, et HTML5 saab olema tulevik ning see muutub uueks standardiks. Lisaks aastal kaks tuhat kümme mainis Steve Jobs veel, et Flash on ebaturvaline ja ressursimahukas, mida on telefonis piiratud kogus. [5]

HTML5 välja tulekuga otsustas YouTube, et migreerub Flash mängijalt HTML5 video peale ning Adobe, firma kes omab Flashi, otsustas ise ka, et lõpetab Flashi mängija arendamist ning julgustab arendajaid arendama uute standardite järgi. Google on ka tänaseks võtnud otsuse vastu, et Google Chrome versioon viiskümmend neli ei toeta Flash mängijat enam. [5]

Töös prooviti leida mõned virtuaalsed klaverid, mis oleks tehtud Flashis ja võrrelda neid töös pakutud lahendusega. Esimene asi, mida Flashis tehtud klaver vajab, on kasutaja nõusolekut, et saada klaverit mängida nagu on näha Joonis 6. Peale selle ei olnud klaver eriti teistsugune, HTML5-l põhinevatest klaveritest. Peale selle oli otsides palju lahendusi

mis olid tehtud just uue standardi järgi, millest saab järeldada, et Flashi tooteid on hakatud kas eemaldama või on need ümber arendatud uue standardile. [6]

Joonis 6. Flashi hoiatus kasutaja nõusoleku küsimisel

3.3 Java kasutamine sarnastes rakendustes

Esimesed Java Appletid, mis vajasid brauseri *plug-in*'i tulid kasutusele umbes samal ajal millal Flash, kuid Java ei jäänud püsima nii kauaks. Juba suhteliselt vara leiti, et sellega on probleeme ning paljud selle arendajad läksid Flashi peale üle.

Sarnaselt Adobele, kes lõpetasid enda toote arendamise ning julgustavad arendajaid uut standardit kasutama, on ka Oracle ära lõpetanud Java Appletide toetuse. Neil on plaanis liikuda brauseri *plug-in*'ilt maailma, kus pole vaja mingeid lisandeid, et veebil ringi liigelda, kasutades Java Web Start-i. [7]

Kahjuks ei leidnud töö autor ühtegi klaverit, mis oleks tehtud Javal ning seetõttu ei saanud võrrelda ka enda lahendust Java Appletiga tehtud lahendusega.

4 HTML-i heli võimalused

HTML-i algus aegadel, oli tegemist ainult staatiliste asjade näitamisega, alustades lihtsatest kodulehtedest kuni keeruliste tabelitega täidetud lehekülgedeni. Kahjuks puudus sellel ajal heli mängimine läbi HTML-i enda, mis arvatavasti aitas kaasa Flashi ja Java Appletite levikule, mis võimaldasid seda valikut.

HTML5 väljatulek tõi endaga kaasa palju uusi võimalusi, audio mängimine on üks nendest. Seda elementi kasutades on vaja ette anda audiofaili asukoht ning seejärel on kasutajal valik seda kas alustada, pausida, valida alguskoht ja heli muuta. Peale selle on veel võimalus alustada nende mängimist läbi JavaScripti, mis lubas teha klaveri selliselt, et kui nupule vajutada, mängib vastav heli.

Eelmainitud variant kahjuks aga vajab juba olemasolevat helifaili, olgu selleks näiteks kolmesekundiline noot. Olenemata sellest on võimalik JavaScriptis ise teha MIDI faile, mis mängivad heli vastavas sageduses, näiteks C_4 noot on 261,63 MHz sagedusega. Samas kui klaver teha ainult MIDI-ga siis see ei kõla üldse realistlikult. Seda probleemi saab lahendada kui kasutada SoundFonte.

4.1 SoundFont

SoundFont on helisünteesi formaat, mis võtab oma heli põhjaks just mingi olemasoleva heli, mitte siinuse, mida teised sünteesid kasutavad. See lubab saavutada reaalsema heli, mis omakorda jätab mulje nagu tegu oleks päris klaveriga, mitte arvuti poolt genereeritud helide.

Antud lahenduses sai kasutatud juba eelnevalt tehtud SoundFonti mängijat, mis on arendatud Danigb-i poolt ning on saadaval GitHubis, mis võimaldab mängida klaveri noote üllatavalt realistlikul kujul. Selles lahenduses saab klaveri noote mängida üllatavalt kergesti, mida on järgnevalt natukene seletatud.

Alguses on vaja ära deklareerida instrument, määrates ka selle helitugevus ning seejärel saab mängida vastavat nooti, andes argumendiks just noodi nime, näiteks C_4, mitte

sageduse, mida klahv mängiks. Antud lahenduses on peale klaveri veel paljusi teisi instrumente mida valida, kuid töö raames on valitud just klaver. [8]

5 Kasutaja sisendi käsitlemine

Antud töös sai eelnevalt vaadeldud kahte populaarsemat viisi virtuaalse klaveri tegemiseks, kasutades *canvas*-t või *div* elemente. Mõlema puhul on vaja lahendust, kuidas kasutaja sisendit käsitleda. JavaScriptis on selleks kolm erinevat võimalust, millest kahte küll eriti ei soovitata. Siiski on need suhteliselt populaarsed.

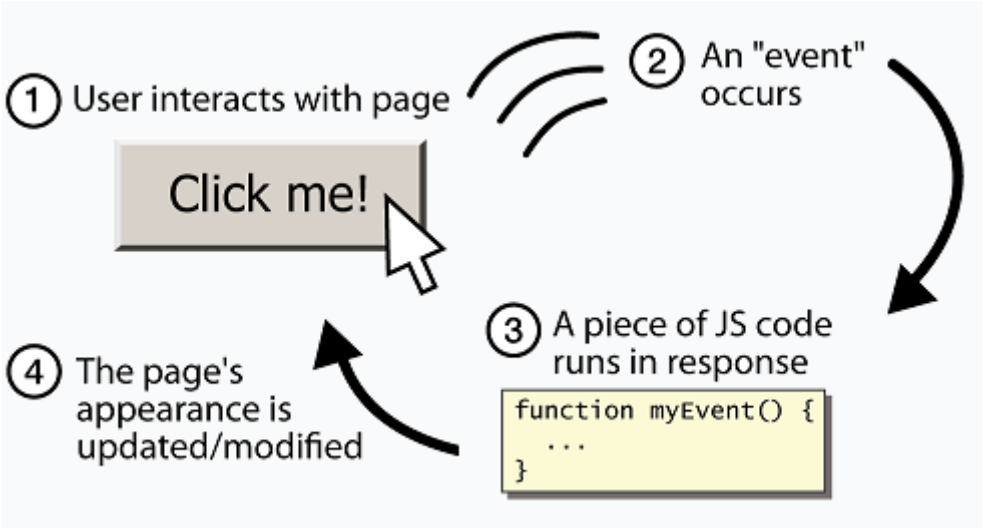
```
1. <div onclick=’myFunction()’>
2. Element.onclick = function(event) {myFunction1()};
3. Element.addEventListener(’click’,function(event) {myFunction2});
```

Esimesed kaks on just need, mida JavaScriptis eriti ei soovitata kasutada, kuid samas on nad kompaktsemad ning vajadusel, saab neid kiiremini eemaldada. Samuti on just need meetodid, mis määratakse DOM-ile ning nende kasutamisel on omad halvad küljed.

Nendel meetoditel saab olla elemendil ainult üks funktsioon juurde määratud. Juhul kui on kasutatud seda meetodit, ei saa teist funktsiooni deklareerida ilma, et see kirjutaks üle esimese funktsiooni. see võib tekitada olukordi kus on vaja just, et lisatakse teine funktsioon. Olenemata sellest võib koodis olla ka olukordi, kus just on vaja seda ühte funktsiooni kasutada. Samuti ei näita need kaks meetodit enda veateateid, kui need on valesti deklareeritud, mida kolmas meetod välistaks. Lisaks on veel esimesel meetodil oma vead. Esiteks ei erista see dokumendi struktuuri ja loogikat, mis võib tekitada segadusi arendajal, kui ta peab mingit viga hakkama ülesse otsima. Teiseks võib see põhjustada aeglasemat lehe kuvamist kuna brauser peab rohkem alla tõmbama, ennem kui ta saab seda ette näidata.

Viimasel meetodil on aga palju eeliseid eelmise kahe üle. Üks suurimaid on just see, et elemendiga seotud funktsioone, saab olla lõpmatult palju ning neid saab sama kergesti eemaldada ka teise funktsiooniga. Peale selle eristab see dokumendi struktuuri, milleks on HTML ja dokumendi loogika, mis teeb arendaja elu kergemaks juhul kui on vajadus viga üles leida või ta peab lihtsalt edasi arendama. Kõige viimaseks on see meetod toetatud kõikidel Interneti brauseritel, mis on vanemad kui Internet Explorer 7. [9]

Arvutil ja telefonil pole JavaScriptis suuri erinevusi. Ainukene erinevus on just *eventide* nimed. Arvutil on *mouseup, mousemove, mousedown* ja *mousecancel*. Telefonil sellega võrreldes on *touchend, touchmove, touchstart* ja *touchcancel*. Mingi eelnimetatud *eventi* peale käivitatakse vastav funktsioon. Kasutaja sisendi käsitlemise elulugu on ära toodud Joonis 7.



Joonis 7. JavaScripti *eventi* elutsükkel [10]

Üks suurimaid erinevusi mis on võrreldav antud töö lahenduses ning paljudel teistel virtuaalsetel klaveritel on oktaavide erinevus. Otseselt autor ei suutnud sellele põhjust leida, kuid arvatavasti on põhjuseks see, et väiksem oktaavide ulatus annab võimaluse mängida klaverit ka arvuti variandil, mida antud töö lahendus ei ole rakendanud. Antud töö eesmärgiks oli saada nii arvutil kui ka telefonis töötav virtuaalne klaver, mitte arendada klaverit ainult arvutile.

Antud töö lahenduses on kasutatud *canvas* elementi. Kuna tegemist on ainult ühe HTML elemendiga, kuhu peale on joonistatud, siis on sellel raskem kasutaja sisendit tuvastada. Seetõttu on autori lahenduses rakendatud *canvas* elemendile eelnimetatud *event*-id. Ning sealt omakorda on kasutatud kasutaja sisendi koordinaate, et kindlaks määrata kuhu vajutati või liigutati hiir ja seejärel vaadata kas mingi klahv jäi vastava koha alla. Pärast kindlaks tegemist, kuhu kasutaja on vajutanud, joonistatakse klaver uuesti ning antud klahv on enda värvi muutnud. Põhjuseks, miks terve klaver uuesti on vaja, tuleneb sellest, kuidas *canvas* töötab. Eelnevalt sai mainitud, et sellele joonistades pannakse joonistatud asi kõige peale, see aga varjaks ära osa mustadest klahvidest. Selle parandamiseks saabki terve klaver uuesti joonistatud.

Alternatiivne võimalusega, virtuaalse klaveri tegemiseks *div* elementidega, oleks kasutaja sisendi käsitlemine kergem. Seal saab igale HTML elemendile, milleks oleks CSS-iga kujundatud klaveri klahv, määrata kindel noot, mida mängida. Sellist lahendust kasutab just üks populaarseim virtuaalne klaver, virtualpiano.net. Seal on kasutusel eelmainitud esimene meetod, HTML elemendile külge määratud funktsioon, mis nooti mängib. [2]

6 Rahvahange

Tänapäeva maailmas võib tunduda, et kõik uued tehnoloogiad on saanud alguse just viimasel paarikümnel aastal, kuid esimesed dokumenteeritud rahvahanke üritused toimusid just tuhande seitsmesajandatel aastatel. Sellel ajal oli tegemist küll Briti valitsuse poolt välja pakutud probleemiga, kuidas usaldusväärselt leida laeva geograafiline pikkus. Antud probleemile oli pakutud veel kahekümne tuhande naela väärtuses autasu. [11]

Nüüdseks on rahvahange muutunud dramaatiliselt. Tänapäeval peetakse rahvahankeks just mingi kindla probleemi lahendamist, kasutades vabatahtlikke enda eesmärgi saavutamiseks. Tavaliselt ei saa probleemi lahendamise eest rahalist kompensatsiooni, nende tasuks on eelkõige intellektuaalne rahuldus. Probleemi lahenduse leidmisel ei tunnustata ühte kindlat isikut, vaid kõiki, kes rahvahankest osa võtsid. Peale rahvahanke on veel paar teist mõistet mis on tihedalt seotud: ühiskatsetus ja ühisrahastus.

6.1 Rahvahanke populaarsus

Olenemata sellest, et rahvahanget kasutati juba paar sajandit tagasi, on see suuremas mastaabis kasutust leidnud just viimastel aastatel. Üheksakümnendatel hakati kasutama rohkem just sellisel meetodil probleemi lahendamise võimalust. Üheks suureks faktoriks mängis rolli kindlasti just tehnoloogia areng. Eelnevalt polnud tehnoloogia piisavalt kaugele arenenud, mis võimaldaks antud meetodit korralikult rakendada.

Leiti, et sellega saab paremat tagasisidet, millest rahvas on huvitatud. See võimaldab rahvahanke korraldajal näha lahendust, milleni muidu poleks pruugitud jõuda. Ainuüksi inimeste koguse tõttu, kes võtavad osa rahvahankes, annab tohutu eelise. Lisaks on inimesed kes osa võtavad motiveeritud just probleemi lahendamisest, ning neil ei ela suur korporatsioon seljas, kes kohustab neid seda tegema. Ilma firma mõjuta on inimestel palju ausamad vastused.

Peale selle, et osavõtjate arv on suurem, on veel rahvahanke populaarsuseks selle hind. Probleemi lahendamiseks üldjuhul ei maksta lahendajatele mitte midagi. Selle asemel

tunnustatakse kõiki inimesi, kes võtsid osa selle lahendamisel. See omakorda võimaldab teha projekte palju massiivsemal kujul, kui muidu võimalik oleks.

6.2 Rahvahanke näidised

Tänapäeval on paljud firmad kasutanud mingisugust rahvahanke meetodit, olgu see kas logo loomiseks või lausa terve rakenduse selgrooks. Üheks suureks näiteks on just Waze, mille kogu süsteem töötabki just kasutajate tõttu. Kasutajad märgivad reaalajas erinevaid juhtumeid kaardile, et teised autojuhid näevad, mis nende ümber toimub, alustades ummikutest ja lõpetades politseinikute asukohtadest. Terve nende firma töötab just rahvahanke meetodil. Neil on vaja kasutajaid, kes märgiks kaardile olukordi, mis julgustab veel rohkem kasutajaid seda rakendust usaldama. [12]

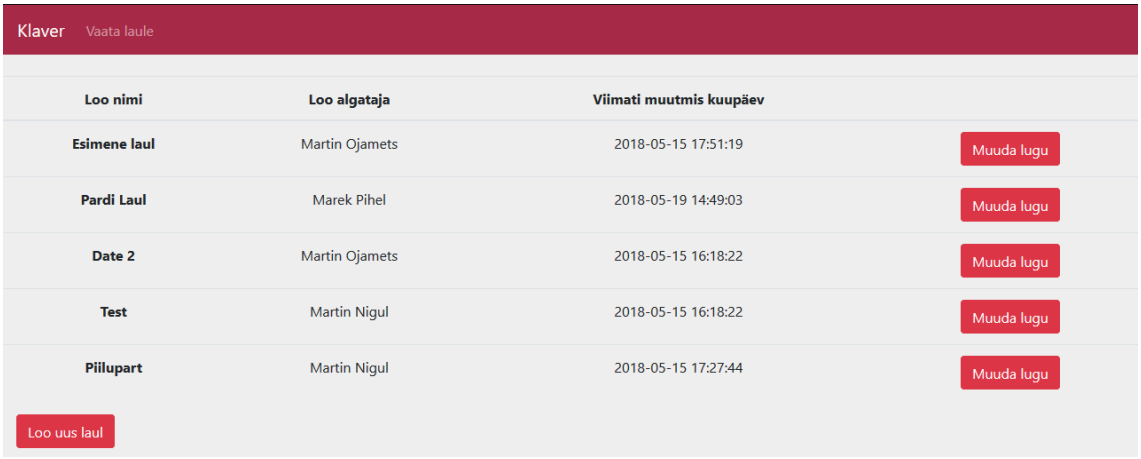
Samas pole see ainukene näide. Üks populaarsemaid Interneti entsüklopeediaid, Wikipedia, on ülesse arendatud just sellel samal meetodil. Wikipedia eesmärgiks on koguda kokku informatsioon entsüklopeedia formaadis, ning teha see kogu maailmale kättesaadavaks. Sellisel skaalal projekt on võimalik ainult vabatahtlike abiga. Samas võimalus muuta kõiki sissekandeid Wikipedias ei ole teinud sellest usaldusväärset allikat just selle sama põhimõtte pärast, et igaüks saab seda muuta. Osad sissekanded võivad olla küll valed või omakasuks moonutatud, kuid paljud on siiski usaldusväärsed, kuigi nendel võib leiduda aegunud informatsiooni.

Üheks kodumaise lahenduse näiteks on veebileht ajapaik.ee. See lehekülg on tehtud arhiiviks, näidates kuidas Eesti kohad nägid välja aastate eest. Olenemata sellest, et tegemist on küll rahvahanke projektiga on nad sattunud rahalistesse raskustesse, kus rakendavad ka eelnimetatud ühisrahastust eesmärgiga hoida teenus tasuta kättesaadavana, kasutades selleks saadavaid annetusi. [13]

6.3 Rahvahanke teostus töö lahenduses

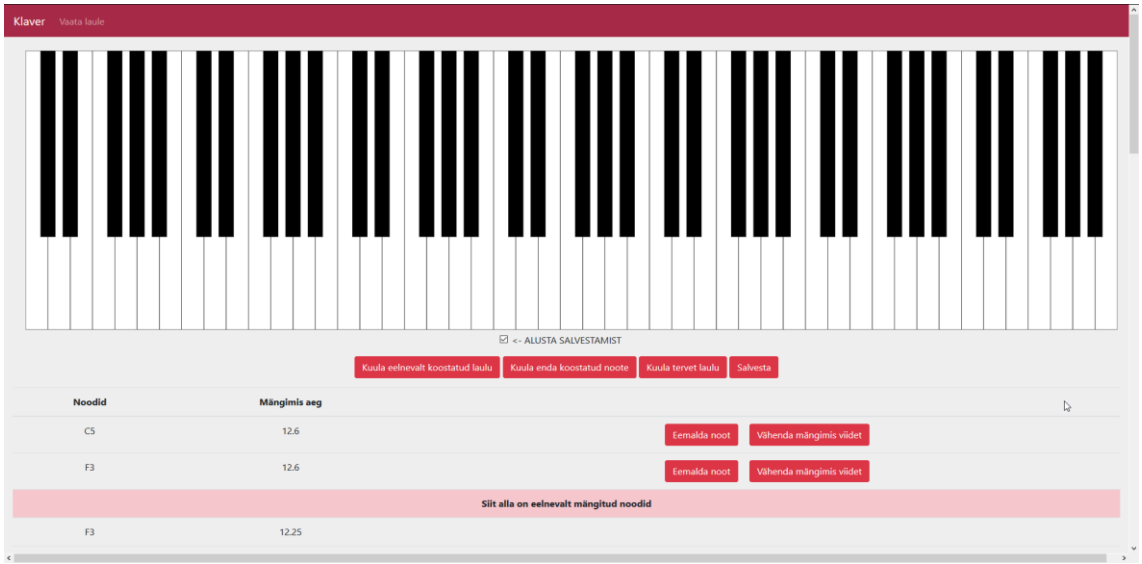
Antud töö lahenduses on samuti rakendatud rahvahanke meetodit, kuid natukene erinevalt võrreldes eeltoodud näidetega. Rahvahanke teostamiseks on küll erinevaid viise, aga

programmis on see lahendatud just niimoodi, et kasutajal on võimalik luua uus laul või valida olemas olev laul, nagu on näha Joonis 8.



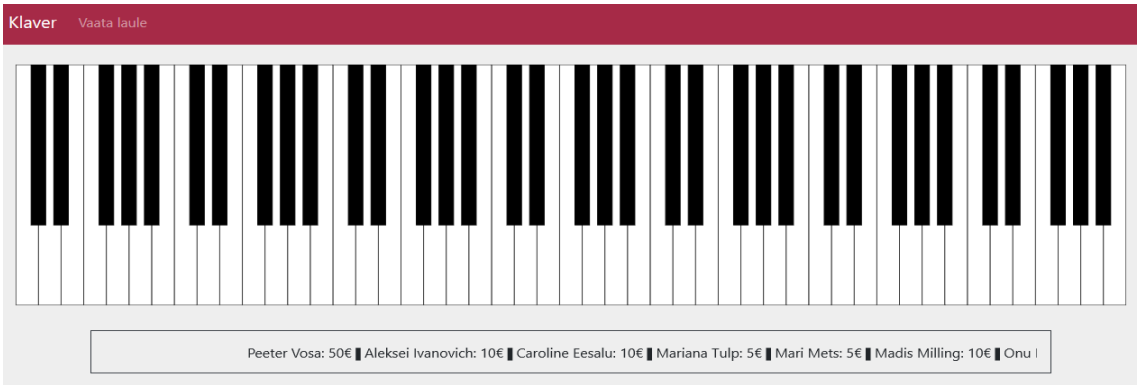
Joonis 8. Laulu valik mida muuta

Laulu valides saab kasutaja salvestada kuusteist nooti ühe korraga. Joonis 9 on välja toodud kuidas näidatakse juurde mängitud noote. Seejärel saavad ka teised kasutajad kuulata juba kirjutatud laulu ning sellele ise juurde mängida oma kuusteist nooti. Põhjus just kuueteist noodi valikuks on autori enda katsetus meetod avastades, et ühe noodi mängimine muutuks liiga tüütuks ja arvestades Eesti populatsiooni võib võtta loo valmis saamisega pikka aega. Samuti kui lasta rohkem mängida, siis poleks eriti mõtet rahvahanke meetodit kasutada, sama hästi oleks võinud lasta ühel inimesel terve laul koostada. Rahvahanke lisamine töösse tekitaks kasutajates rohkemat huvi kui tavaline virtuaalne klaver, sest kasutajad tunnevad, et nad on osa millestki suuremast kui nad ise.



Joonis 9. Klaver koos kahe juurde mängitud noodiga

Lahenduses on kasutatud veel MySQL-i andmebaasi, kuhu on salvestatud laulud, nende noodid ning kasutajad. Annetatud kasutajad on välja toodud esilehel liikuva tekstina nagu on näidatud Joonis 10. Selleks, et laule ja noote kätte saada andmebaasis, on tehtud ajaxi kutse PHP koodile, mis pärib andmebaasist vastava informatsiooni ning tagastab selle JSON formaadis.



Joonis 10. Lahenduse avaleht koos annetajatega.

7 Kokkuvõte

Töös püstitatud põhiprobleem, virtuaalse klaveri arendamine, sai autori poolt tehtud kasutades HTML5 elementi *canvas*. See võimaldab joonistada JavaScriptiga ning läbi *event-handler*-ite on võimalik klaverit mängida nii telefonis, tahvelarvutis kui ka lauaarvutil. Peale selle uuriti veel ka teisi võimalusi, mida oleks saanud kasutada antud lahenduse asemel neid omavahel võrreldes. Võrreldes populaarse virtuaalse klaveriga, virtualpiano.net, on töös puudu arvuti klaviatuuril mängimise võimalus.

Töös rakendatud rahvahanke meetod tõi positiivseid tulemusi, kuid sellele jäid mõningad puudused, mis olid algselt plaanitud. Algselt oli plaanis teha *branching*, et oleks võimalik sama laulu lõpetada erinevalt. Selline lahendus nõuaks kordades rohkem arendamist andmebaasi tasandil, kus salvestatakse noote ja nende mängimis aega. Hetkel on ainult võimalik lisada laulule kuusteist nooti korraga. See kogus on piisav, et kasutaja tunneks enda panust loo tegemises, samas on see piisavalt väike, et üks inimene tervet laulu ära ei kirjutaks. Andmebaasi tasandil on laulul üks väärtus, mis kontrollib, kas see on hetkel muutmises. Kui kasutaja valib laulu mida muutma hakata, lukustatakse see laul viieks minutiks, mis peaks olema piisav aeg, et loole mängida kuusteist nooti juurde.

Soov annetuste kogumiseks jäi kahjuks täitmata antud töö raames. Selleks, et saada SWEDbankilt raha ülekandmise nuppu, peab olema äriklient, mida töö autor ei ole. Teiste pankade puhul seda ei uuritud. Juhul kui Viru Muusikakool tõesti võtab rakenduse kasutusele, tuleb neil see ise ära teostada.

Kasutatud kirjandus

[1] C. Duckett, „ZDNet,“ 28 Jan 2016. [Võrgumaterjal]. Available: https://www.zdnet.com/article/java-browser-plugin-to-be-sent-to-death-row-in-september/. [Kasutatud 04 Mar 2018].

[2] Crystal Magic Studio, „Virtual Piano,“ 2006. [Võrgumaterjal]. Available: https://virtualpiano.net/. [Kasutatud 03 Mar 2018].

[3] W3School, „W3School,“ [Võrgumaterjal]. Available: https://www.w3schools.com/graphics/canvas_intro.asp. [Kasutatud 27 Feb 2018].

[4] Harvard Extension School, „CSCI E-12 Fundamentals of Website Development,“ 24 Feb 2014. [Võrgumaterjal]. Available: https://cscie12.dce.harvard.edu/lecture_notes/2015/20150224/images/divitis.png. [Kasutatud 03 Mar 2018].

[5] K. Collins, „qz,“ 29 Dec 2016. [Võrgumaterjal]. Available: https://qz.com/863467/how-adobe-flash-once-the-face-of-the-web-fell-to-the-brink-of-obscurity-and-why-its-worth-saving/. [Kasutatud 06 Apr 2018].

[6] D. Kazakov, „real-royal,“ 18 Nov 2016. [Võrgumaterjal]. Available: https://real-royal.com. [Kasutatud 06 Apr 2018].

[7] D. Topic, „Oracle,“ 27 Jan 2016. [Võrgumaterjal]. Available: https://blogs.oracle.com/java-platform-group/moving-to-a-plugin-free-web. [Kasutatud 06 Apr 2018].

[8] danigb, „GitHub,“ 29 Aug 2017. [Võrgumaterjal]. Available: https://github.com/danigb/soundfont-player. [Kasutatud 03 Mar 2018].

[9] M. Perlakowski, „GitHub,“ 29 Jan 2016. [Võrgumaterjal]. Available: https://stackoverflow.com/questions/6348494/addeventlistener-vs-onclick. [Kasutatud 20 Mar 2018].

[10] K. Kareem, „Catchpoint,“ 16 Dec 2016. [Võrgumaterjal]. Available: http://blog.catchpoint.com/2016/12/16/web-performance-101-optimizing-javascript/. [Kasutatud 11 Mar 2018].

[11] R. Dawson ja S. Bynghall, Getting Results From Crowds: The definitive guide to using crowdsourcing to grow your business, Advanced Human Technologies Inc, 2011.

[12] Waze, „Waze,“ 2006. [Võrgumaterjal]. Available: https://www.waze.com/et/. [Kasutatud 11 Mar 2018].

[13] „Ajapaik,“ 2011. [Võrgumaterjal]. Available: https://ajapaik.ee. [Kasutatud 11 Mar 2018].