Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Matija Jeras

Interaktiven spletni sistem za vizualizacijo glasbe

DIPLOMSKO DELO

UNIVERZITETNI ŠTUDIJSKI PROGRAM
PRVE STOPNJE
RAČUNALNIŠTVO IN INFORMATIKA

Mentor: doc. dr. Matija Marolt

Ljubljana, 2016

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Interaktiven spletni sistem za vizualizacijo glasbe:

V ovkiru diplomske naloge razvijte spletni sistem, ki bo omogočal interaktivno upodabljanje na platno ob poslušanju glasbe. Sistem naj bo intuitiven za uporabo, omogočati pa mora hkratno sodelovanje več uporabnikov ali samostojno upodabljanje, pomnjenje narisanih gest in možnost izdelave videa celotne upodobitve.

IZJAVA O AVTORSTVU
diplomskega dela

Spodaj podpisani Matija Jeras, sem avtor diplomskega dela z naslovom:

Interaktiven spletni sistem za vizualizacijo glasbe

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom doc. dr. Matije Marolta,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela,
- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki »Dela FRI«.

V Ljubljani, dne 30. avgusta 2016 Podpis avtorja:

Zahvaljujem se svojemu mentorju doc. dr. Matiji Maroltu za vse nasvete in strokovno pomoč.

Za prijaznost in dobljeno znanje, ki mi je zelo pomagalo pri izdelavi diplomske naloge, se zahvaljujem g. Božidarju Svetku.

Zahvaljujem se tudi svoji punci Mojci, sestrama Ivi in Toniji ter očetu Iztoku, ki so me vseskozi podpirali.

Predvsem pa se zahvaljujem svoji mami Snežani za nasvete in podporo pri pisanju diplomske naloge.

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
HTML	Hyper Text Markup Language	Jezik za označevanje besedila
CSS	Cascading Style Sheets	Jezik za slog spletnih strani
ID	Identity	Identiteta
HD	High Definition	Visoka definicija
3D	Three Dimensional	Tridimenzionalno
RGB	Red Green Blue	Barvni model iz rdeče, zelene in modre
HSL	Hue Saturation Luminance	Barvni model iz odtenka, nasičenosti in svetlosti
MP4	MPEG Layer-4 Audio	Digitalni multimedijski format

Povzetek

Naslov: Interaktiven spletni sistem za vizualizacijo glasbe

Avtor: Matija Jeras

Povzetek: Diplomska naloga se osredotoči na razvoj interaktivnega spletnega sistema, ki uporabniku omogoča in karseda olajša interaktivno upodabljanje glasbe na platno HTML (*HTML canvas*). Gre za sistem, preko katerega uporabnik manipulira s prikazano sliko na platnu, ter tako s pomočjo zvoka, ki se predvaja v ozadju, upodablja glasbo. Sam uporabniški vmesnik tega sistema mora biti enostaven, razumljiv ter predvsem prijazen do uporabnika, saj mora pri upodabljanju glasbe uporabnik hitro reagirati in nima časa za iskanje želenih funkcij vmesnika.
Sistem je predstavljen v obliki spletne strani. Za implementacijo se torej koristijo glavne spletne tehnologije, kot na primer HTML, JavaScript ter CSS. V tem okolju sta na voljo dva glavna načina interaktivne vizualizacije glasbe: v živo ali *live*, kar pomeni, da pri upodabljanju glasbe sodeluje več uporabnikov in *solo* ali samostojen način, kjer uporabnik sam upodablja glasbo in mu je tako na voljo več funkcij. Na voljo je tudi ogled ali prenos posnetka seje, v kateri se je upodabljala glasba na platno.

Ključne besede: vizualizacija, glasba, upodobitev, splet, platno, HTML, v živo, solo.

Abstract

Title: Interactive system for music visualisation on the web

Author: Matija Jeras

This thesis focuses on the development and implementation of interactive web system, that is meant for sound vizualization. User can use this system to manipulate sound and displayed image, thus visualizing the sound on an HTML canvas. The user interface must be simple, understandable and above all user friendly, because while visualizing, user must react quickly and does not have the time to search for the desired functionality of the interface.

The system is presented in the form of a website, therefore, web technologies such as HTML, JavaScript and CSS are used for implementation. There are two main ways in which the user can visualize sound: *live mode*, in which there are number of users involved in the visualization and *solo mode*, where the user visualizes the sound by himself, therefore havinig access to other functions, which are not available in live mode. All vizualization sessions are of course recorded and can be later watched or downloaded.

Keywords: visualization, sound, web, canvas, HTML, live, solo.

Poglavje 1

Uvod

Vsebina diplomskega dela izhaja iz multimedije, in sicer gre za področje med glasbo in grafičnim upodabljanjem. Ljudje doživljamo glasbo na vrsto različnih načinov in zato se ta naloga, skozi upodabljanje glasbe, osredotoči prav na raziskovanje teh načinov.

Kot je povedal g. Božidar Svetek, ki je na to temo napisal knjigo *Ustvarjanje svetlobe*, ki jo uporabljam tudi kot literaturo za to diplomsko nalogo, na tem področju ni opravljenih veliko raziskav [4]. Program za upodabljanje glasbe bi potreboval tudi on sam, saj se s tem področjem profesionalno ukvarja. Pred skoraj dvajsetimi leti, natančneje leta 1997, je za te potrebe gospoda Božidarja Svetka in v mentorstvu doc. dr. Nikolaja Zimica, program za upodabljanje glasbe namreč razvil takratni študent Robert Turnišek. Sprogramiran je bil na *IRIXu*, ki je operacijski sistem podjetja *SGI* in je že precej zastarel, kar je bil tudi glavni povod za izdajo predloge tega diplomskega dela.

1.1 Pregled področja

Z malo brskanja po spletu lahko kar hitro odkrijemo vrsto programov, ki spadajo pod temo *vizualizacija glasbe* ali, po angleško, *music visualization*. Skoraj vsi, predvsem pa najbolj znani predvajalniki glasbe, imajo vgrajeno

funkcijo vizualizacije glasbe, vendar pa je to samodejna vizualizacija, ki jo generira sam program z vrsto različnih algoritmov. Ti algoritmi uporabljajo podatke, kot sta valovna dolžina ter frekvenca digitalnega signala glasbe, za vizualizacijo glasbe na zaslon. Uporabnik pri teh programih (iTunes, Windows Media Player, Winamp, itd.) torej nima nadzora nad vizualizacijo glasbe, razen tega, da lahko izbira, kakšni grafični učinki se bodo po ritmu glasbe prikazovali na zaslonu.

Poleg predvajalnikov glasbe obstajajo tudi programi, ki so narejeni eksplicitno za manipulacijo nad vizualizacijo glasbe. Pri VSXu na primer, ki sem ga tudi sam preizkusil, uporabnik grafično dodaja različne module, ki jih povezuje med sabo in tako skupaj predstavljajo različne elemente na zaslonu. Na ta način lahko uporabnik neposredno spreminja obliko, barvo in odziv elementov na različne karakteristike glasbe ter še veliko več.

V okviru vizualizacije glasbe obstaja tudi taka, ki jo ponuja ta sistem, torej primer, ko uporabnik z določenimi slikarskimi orodji upodablja glasbo na platno. Na tem področju je g. Božidar Svetek naredil že veliko glasbenih vizualizacij, ki jih je objavil na spletnem portalu *YouTube*; največ ogledov ima na primer vizualizacija skladbe „Puccini - O mio babbino caro“. Vse vizualizacije so narejene v programu Roberta Turniška, ki pa ima veliko pomanjkljivosti. To je ena od motivacij za razvoj tega sistema.

1.2 Motivacija

Glavna motivacija za razvoj tega sistema je zastarelost že obstoječega programa za upodabljanje glasbe, ki deluje na starem računalniku podjetja *SGI* in je tako omejen na njegove zmogljivosti. G. Božidar Svetek je namreč s tem programom upodabljal glasbo od leta 1997 do danes, torej skoraj 20 let. Tehnologija se hitro izboljšuje in nam danes omogoča, da zelo poenostavimo in izboljšamo tako uporabniško izkušnjo upodabljanja kot kakovost končnega izdelka. Program Roberta Turniška namreč nima možnosti posneti samega upodabljanja glasbe, zato se za to uporablja zunanji program, ki posname

del zaslona, na katerem uporabnik upodablja. Ta posnetek se nato skupaj s skladbo, ki je predvaja v ozadju, zapeče na CD, ki ga vstavimo v CD pogon. Vse skupaj je torej zelo analogno, kot je tudi običajno za čas, ko je bil program narejen, za danes pa precej zastarelo.

Velika pomanjkljivost tega programa izhaja tudi iz dejstva, da se mora celoten postopek začeti znova, če se uporabnik na katerikoli točki upodabljanja glasbe zmoti. To je motivacija za implementacijo funkcije, ki bi omogočala izbris samo tiste akcije, ki je pokvarila celotno upodobitev glasbe.

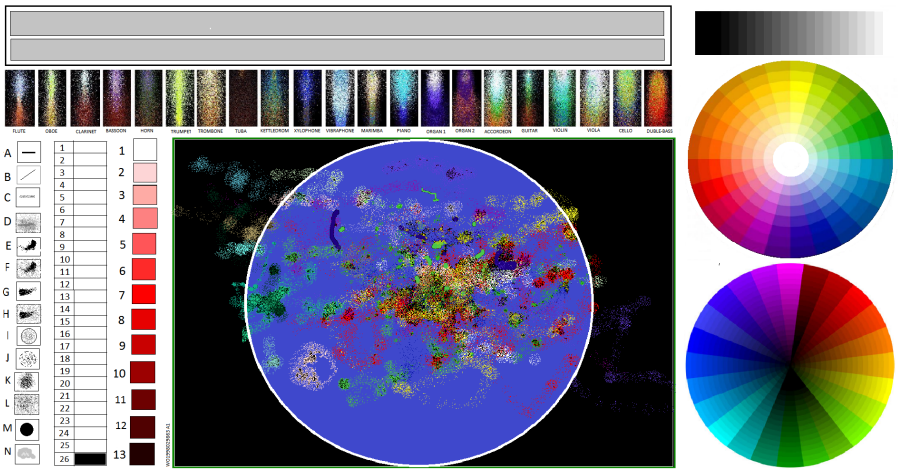
Seveda je tu tudi ločljivost posnetka končnega izdelka, ki je v Robertovemu programu konstantna in enaka 640x480, torej v razmerju 4:3, kar je zelo nizko za današnje razmere. Ta sistem bi z izbiro med različnimi razmerji ločljivosti in predvsem možnostjo HD ločljivosti ponudil lepše posnetke, ki so bolj prijazni očem.

Vse to, poleg dejstva, da podobnega programa za upodabljanje glasbe ni na voljo, je motivacija za razvoj tega sistema.

1.3 Cilj diplomskega dela

Namen ter cilj tega diplomskega dela je torej razviti sistem, s katerim lahko grafično upodobimo glasbo. Narejen bo v obliki strežnika, na katerega se lahko uporabniki povežejo, nato pa skupaj upodabljajo glasbo. Ta postopek bo potekal tako, da se bo v ozadju predvajala glasba, medtem pa bo lahko uporabnik, ali več uporabnikov skupaj, na zaslonu z različnimi grafičnimi orodji upodobili svojo „vizijo“ glasbe.

Gospod Božidar Svetek je predlagal vmesnik, ki je prikazan na sliki 1.1. Vse skupaj bi delovalo na spletni strani, napisano torej v danes najbolj uporabljenih spletnih tehnologijah, kot so HTML, JavaSript ter CSS. Za osnovo bo sistem vzel dognanja in prejšnji program Božidarja Svetka, ki ga je napisal Robert Turnišek. Zaradi večjega števila uporabnikov in različnih dimenzij zaslonov, ki pridejo s tem, bo treba implementirati tudi možnost izbiranja različnih velikosti platna, da bo le-ta ustrezal uporabnikovemu zaslonu.



Slika 1.1: Predviden izgled vmesnika: narejeno s strani g. Božidarja Svetka.

Poglavje 2

Uporabljene tehnologije

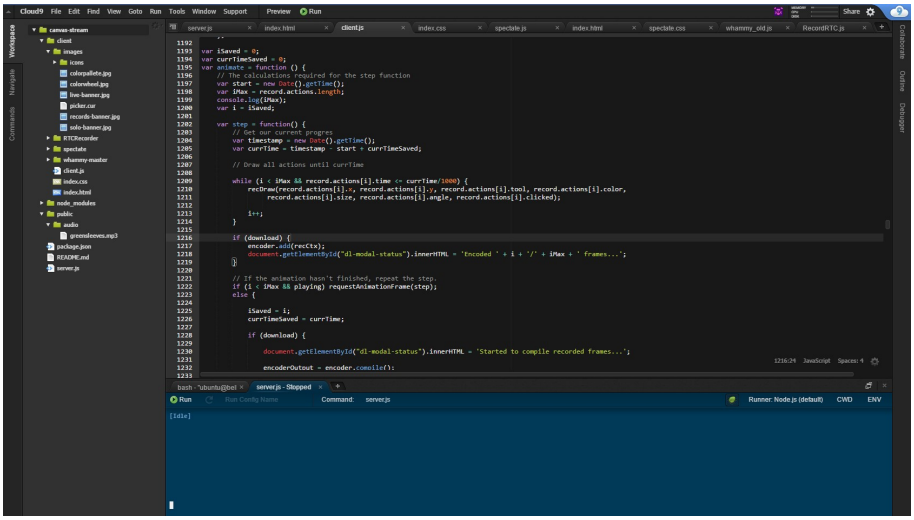
Sistem mora podpirati interakcijo med strežnikom ter več ljudmi hkrati, torej možnost, da se več uporabnikov poveže. Ideja je, da vsak izbere instrument, ki ga želi upodobiti, na koncu pa se vse skupaj združi v en posnetek interaktivne vizualizacije glasbe.

To se da na najboljši način narediti s spletnimi tehnologijami, ki omogočajo princip strežnik - odjemalec, ali angleško, server - client. Strežnik naj bi imel v svojem direktoriju na voljo datoteke, ki jih odjemalec zahteva za pravilno delovanje spletnega vmesnika. Tudi med samo interakcijo z vmesnikom, se med strežnikom in odjemalcem prenaša veliko podatkov. Kar pa naredi ta princip zelo uporaben je to, da lahko te datoteke in podatke zahteva več odjemalcev naenkrat, torej v enem časovnem trenutku je lahko več uporabnikov povezanih na strežnik.

Jasno je torej, da se na strani odjemalca (client-side) uporablja HTML za prikaz same strani, CSS za oblikovanje te strani in postavitev elementov ter JavaScript za delovanje funkcij. Za zaganjanje skript na strani strežnika (server-side), ki omogočajo interakcijo z odjemalcem, pa je na voljo kar nekaj jezikov, dva najbolj znana ter vredna omembe pa sta PHP in Server-side JavaScript. Za ta sistem se uporablja JavaScript in s tem Node.js, ki je strežnik, napisan v tem jeziku in narejen prav za potrebe te diplomske naloge.

Eno od najbolj znanih spletnih razvojnih okolij za take sisteme je Cloud9,

ki je brezplačno in ima na voljo vrsto že vnaprej konfiguriranih delovnih prostorov - *workspaces*, za lažji začetek dela uporabnikov. Služi tudi za gostovanje strežnika, saj lahko kodo, ki je napisana za strežnik, programer kar zažene in tako dobi delujoči strežnik, na katerega se lahko poveže kot odjemalec. Tako lahko preveri, ali vse dela, kot je bilo zamišljeno. Dobra stran Cloud9 je tudi ta, da so vse datoteke s katerimi dela programer, shranjene na njihovih strežnikih. Tako lahko najprej programira na eni napravi ter nadaljuje na drugi, le vpisati se je potrebno v svoj Cloud9 račun.



Slika 2.1: Cloud9 razvojno okolje.

2.1 HTML

HTML ali HyperText Markup Language je jezik za ustvarjanje spletnih strani. Deluje tako, da ima strežnik v svojem direktoriju shranjeno datoteko napisano v jeziku HTML, ki jo pošlje na stran odjemalca, ko se le-ta poveže s strežnikom. Brskalnik (na primer *Google Chrome* ali *Mozilla Firefox*) nato na zaslon prikaže elemente, kot so opisani v datoteki.

Datoteka HTML je zgrajena po elementih, ki se začnejo z oznako:

<element>

in končajo z oznako:

</element>.

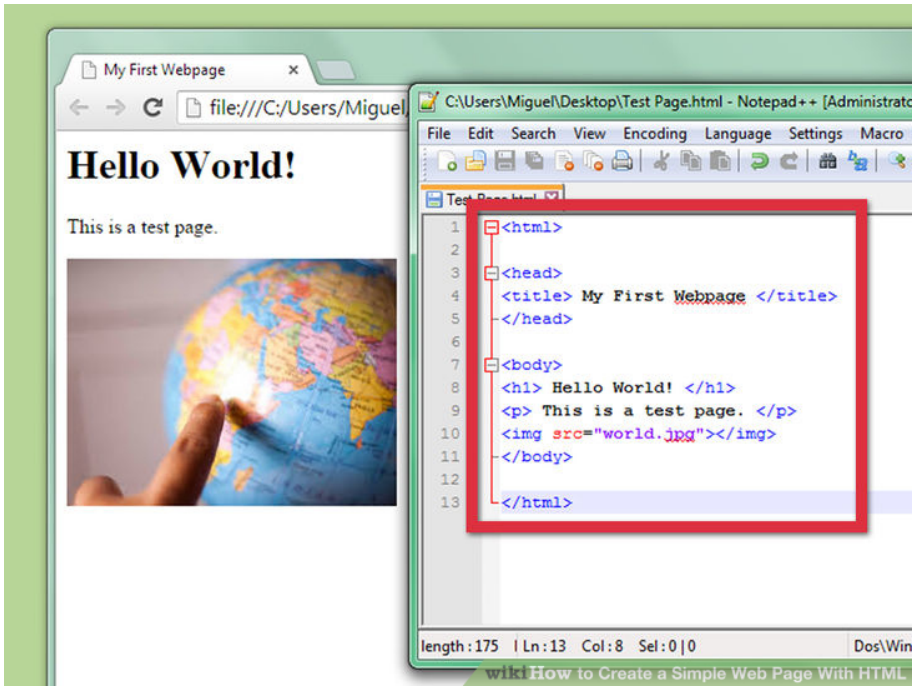
Tako bi bil na primer gumb na določeni strani opisan z:

<button type="button">Klikni me!</button>.

Seveda pa mora imeti datoteka HTML določeno strukturo. Ker v računalništvu pogosto uporabljamo analogije iz resničnega sveta, HTML vsebuje elemente glavo - *head* in telo - *body*.

Glava dokumenta HTML vsebuje neobvezne elemente, kot so *title* v katerem je naslov strani, ki se v brskalniku izpiše na zavihku, *style*, ki brskalniku pove slog prikaza strani (barva besedila, velikost črk, itd.) ter *script*, ki vsebuje funkcije v jeziku JavaScript, ki omogočajo delovanje strani. [3]

V telesu dokumenta pa je dejanska vsebina, torej elementi, ki brskalniku povejo zgradbo strani. [5]



Slika 2.2: Delovanje HTML: v rdečem okvirju je koda HTML z glavo in telesom, ki se odjemalcu v brskalniku pokaže, kot se vidi v ozadju.

2.2 CSS

CSS ali Cascading Style Sheets je jezik, s katerim opišemo predstavitev dokumenta HTML, torej kako bodo elementi HTML izgledali na odjemalčevem zaslonu. Opisuje postavitev in barvo strani, elementov, črk, ozadja in podobno. Po navadi je napisan v svoji datoteki, s končnico *.css*, lahko pa ga vključimo tudi v sam dokument HTML, pod element *style* na tak način:

```
<h1 style="color:red;">Prvo poglavje</h1>.
```

Ta koda bo na strani izpisala „Prvo poglavje“ z rdečo barvo, saj ima ta element (*h1*) v jeziku CSS specificiran slog *color:red*, kar pomeni, da naj bo barva besedila tega elementa rdeča.

Prednost posebne datoteke za CSS je to, da imamo od oblike in predstavitve dokumenta ločeno vsebino, ki je opisana v datoteki HTML. To nam omogoča dosti lažji razvoj in vzdrževanje celotnega sistema. Za več dokumentov HTML lahko tudi uporabimo enako datoteko CSS.

2.3 JavaScript

Poleg HTMLja in CSSa je programski jezik JavaScript eden od treh ključnih tehnologij za implementacijo in delovanje spletnih sistemov. Primarno se torej uporablja na spletu, lahko pa se uporablja tudi za druge stvari, na primer programiranje „desktop widgetov“. V tej diplomski nalogi bo uporabljen na strani odjemalca in strežnika za sporazumevanje v obe smeri. Ravno temu pa je namenjena JavaScript knjižnica *socket.io*, ki uporablja *WebSocket* protokol, ki omogoča prenos podatkov med obema stranema. Ko inicializiramo strežnik, naredimo isto tudi s *socket.io* in sicer ta blok kode izgleda tako:

```
var server = require('http').createServer();
var io = require('socket.io')(server);
io.on('connection', function(socket){
  socket.emit('Hello client!');
  socket.on('disconnect', function(){
```

```
    console.log('Client ' + socket.id + 'disconnected.');
  });
});
server.listen(3000);
```

Tu s knjižnico *http* inicializiramo strežnik, nato pa na tem strežniku inicializiramo *socket.io*. Dogodek (event) *io.on('connection')* se zgodi v trenutku, ko se odjemalec poveže na strežnik. V to funkcijo lahko potem napišemo, kaj se zgodi ob povezavi. V tem primeru, se odjemalcu skozi skripto *emit*, ki jo uporabljamo za prenos podatkov, pošlje sporočilo *Hello client!*.

Po istem principu se ob prekinitvi povezave s strežnikom sproži dogodek *socket.on('disconnect')*. V tem primeru v konzolo strežnika izpišemo ID odjemalca, ki je prekinil povezavo, ta podatek pa dobimo z ukazom *socket.id*. Le-ta vrne identifikacijsko številko odjemalca, ki je za vsakega odjemalca unikatna in se mu dodeli ob povezavi na strežnik.

2.3.1 Node.js

Node.js je razvojno okolje, napisano v JavaScriptu in namenjeno razvoju internetnih aplikacij na strani strežnika. Programerji imajo na voljo vrsto različnih modulov, ki jim zelo olajšajo programiranje spletnih aplikacij in strežnikov. Vsak lahko tudi napiše svoj modul, če želi.

Eden takih modulov, ki sem ga uporabil tudi pri diplomski nalogi, je *express.js*. Omogoča zelo hitro postavitev in inicializacijo strežnika:

```
var express = require('express');
var app = express();
app.listen(3000, function () {
  console.log('Listening on port 3000!');
});
```

S temi štirimi vrsticami imamo postavljen strežnik, ki že posluša na portu 3000 za morebitne zahteve.

2.3.2 jQuery

jQuery je najbolj popularna JavaScript knjižnica na svetu, ki se uporablja za krajšo ter bolj poenostavljeno kodo na odjemalčevi strani. V tej diplomski nalogi, sem *jQuery* uporabljal za preprosto dostopanje do elementov HTML. Brez *jQueryja* je koda za dostop do takega elementa naslednja:

```
var element = document.getElementById("mojElement");
```

V spremenljivki *element* je tako shranjen element HTML z IDjem „mojElement“. Tako lahko nato spreminjamo elementovo postavitev, videz, velikost, itd. Z *jQueryjem* lahko to naredimo na bolj preprost način:

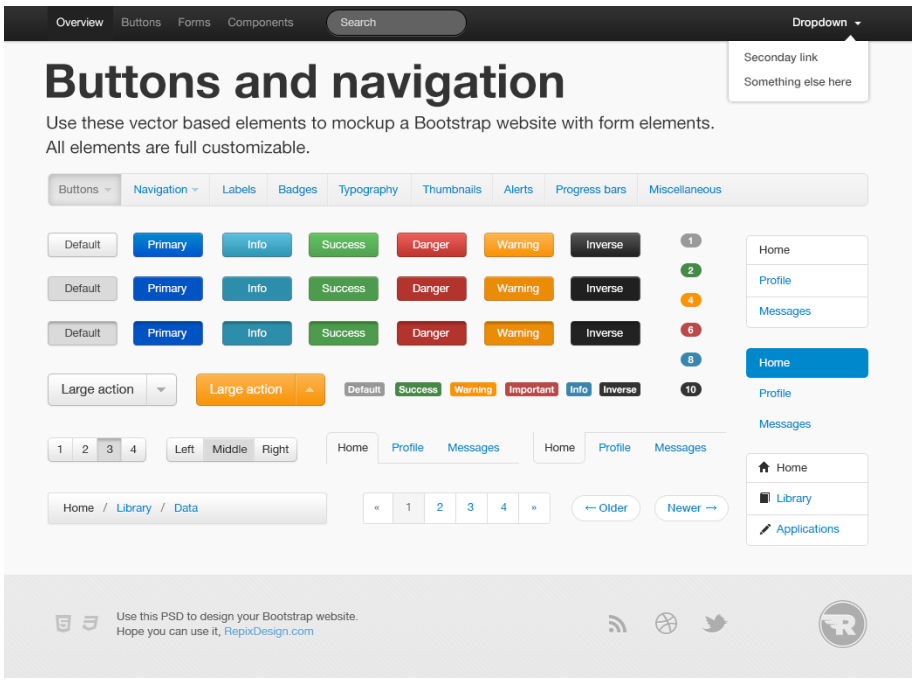
```
$('#mojElement').style.backgroundColor = '#FFFFFF';
```

Tako se z enovrstičnico lahko spremeni barvo ozadja elementa „mojElement“.

2.4 Bootstrap

Bootstrap je brezplačen, odprtokoden *web framework*[1], ki deluje na odjemalčevi strani. Vsebuje predloge HTML ter CSS, ki zelo polepšajo prikazane elemente HTML ter omogočajo izbiro med veliko različnimi postavitvami elementov na spletni strani, zato je delo na programerjevi strani vidno zmanjšano. Boostrapov JavaScript pa ponuja različne animacije, prehode - *transitions* med stranmi, pojavna okna - (modals) in podobno. Bootstrap torej združi moči HTMLja, CSSa ter JavaScripta ter zelo obogati videz in delovanje spletne strani.

[1] Web framework je zbirka knjižnic ter predlog, ki programerju zelo olajšajo delo.



Slika 2.3: Bootstrap elementi HTML: elementi vsebujejo veliko barv ter 3D učinkov, zato so tudi bolj privlačni za oči. [1]

Poglavje 3

Sistem za upodabljanje glasbe

Sistem podpira dva načina interaktivne vizualizacije glasbe, in sicer *live* način, kjer več uporabnikov sodeluje pri upodabljanju iste skladbe, in pa *solo* način, kjer uporabnik upodablja sam in ima na voljo več funkcij, ki jih v *live* načinu ni.

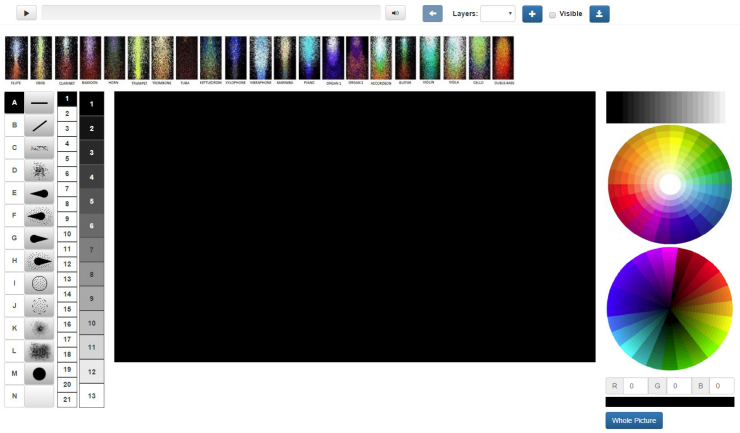
Za potrebe *live* načina mora sistem podpirati prijavo več uporabnikov naenkrat na strežnik in jim omogočati skupno upodabljanje glasbe. Potrebujemo način, ki združi tiste uporabnike, ki želijo upodabljati skupaj. To storimo s sobami, ki jih lahko uporabniki po želji kreirajo, ali pa se pridružijo že obstoječi. Iz sobe lahko nato začnejo sejo upodabljanja glasbe, v kateri z orodji za upodabljanje izrisujejo na HTML platno.

Med upodabljanjem glasbe v *live* seji uporabniki na platnu vidijo samo tisto, kar upodabljajo oni sami ter tisto, kar na platno izrisuje dirigent. Razlog tega je, da je vsakemu uporabniku omogočeno, da nemoteno upodablja svoj del in da ga upodobitve drugih uporabnikov ne motijo. Tako kot končana upodobitev, mora tudi vsaka posamezna upodobitev imeti prave slikovne elemente ter mora, tako rečeno, „stati“. Tu torej pridemo do problema, kako naj uporabnik vidi upodabljanje ostalih uporabnikov. To rešimo s tem, da ima uporabnik možnost odpreti novo okno, ki ga lahko prestavi na želeno mesto na zaslonu in v katerem se izrisuje celotno upodabljanje glasbe, torej skupna slika upodobitev vseh uporabnikov. To okno lahko uporabnik odpre

z gumbom *Whole Picture*, ki ne nahaja na vmesniku za upodabljanje glasbe.

Na voljo mora biti tudi način, ki uporabnikom omogoča ogled posnetkov zaključenih *live* in *solo* sej ter prenos teh posnetkov na lokalni disk uporabnika.

Implementacija spletnega sistema je sestavljena iz več modulov. En od modulov je uporabniški vmesnik, ki deluje na odjemalčevi strani in predstavlja to, kar uporabnik vidi in s čimer je v stalni interakciji. Drugi modul je *client-side JavaScript*, torej koda, ki se izvaja v ozadju na strani odjemalca in samemu uporabniku ni vidna. Tretji modul pa bi bil *server-side JavaScript*, torej delovanje funkcij na strežniku *Node.js*. V nadaljevanju bom opisal module, s pomočjo katerih uporabnik upodablja glasbo.



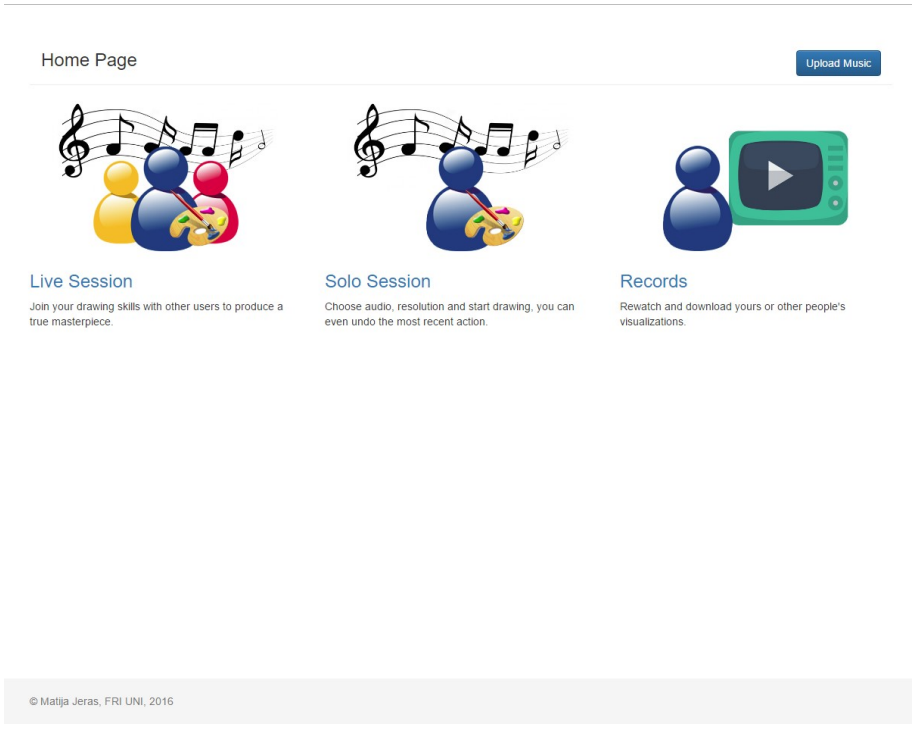
Slika 3.1: Uporabniški vmesnik za upodabljanje glasbe: vmesnik je zelo podoben tistemu, ki ga je predlagal g. Božidar Svetek. Na vrhu je časovnica skladbe z gumbom *Play* ter desno od nje *solo* plošča, ki je na voljo samo v *solo* načinu. Pod časovnico je slika z vzorci orodij, na levi strani je leva plošča (left panel), ki vsebuje tri izbirnike - izbirnik orodja, izbirnik velikosti orodja in izbirnik svetlosti barve. Na sredini je samo platno, ki je v tem primeru veliko 940x540 pikslov, na desni pa desna plošča z izbirnikom barve in prikazovalnikom izbrane barve pod njim. Na prikazovalniku barve vidimo, da je trenutno izbrana barva črna. Spodaj je še gumb *Whole Picture*, ki je na voljo le v *live* načinu.

3.1 Domača stran

Domača stran mora biti prva vidna stvar na vsaki spletni strani. Tu sem se odločil za preprosto, čisto stran s čim manj nepotrebnimi funkcijami. Poleg tega bi bil uporabnik le dva klika stran od začetka upodabljanja glasbe.

V telesu domače strani so na voljo tri povezave, in sicer povezavi do obeh načinov upodabljanja glasbe *(Live Session* in *Solo Session*) in pa povezava *Records*, kjer si uporabnik lahko ogleda vse posnetke sej na strežniku. V glavi strani pa je na voljo gumb *Upload Music*, prek katerega lahko naložimo svoje skladbe na strežnik.

V nadaljevanju bom podrobno opisal oba načina upodabljanja glasbe, in sicer od klika na izbran način na domači strani do samega upodabljanja na platno. Sledil bo ogled posnetkov sej, na koncu pa nalaganje skladb na strežnik.

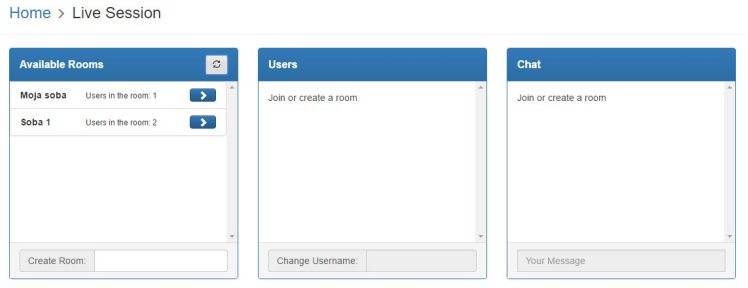


Slika 3.2: Vstopna ali domača stran.

3.2 Live način

V *live* načinu upodabljanja glasbe se več uporabnikov zbere v isti sobi in nato skupaj začnejo sejo upodabljanja izbrane skladbe na platno. Tu je zelo pomembna sinhronizacija zvoka, saj morajo vsi uporabniki v seji slišati isti del skladbe, da jo lahko skupaj uspešno upodobijo. Gre torej za problem prenosa skladbe s strežnika vsem uporabnikom hkrati, saj imajo različni uporabniki različne hitrosti medmrežja. To lahko rešimo tako, da najprej vsakemu uporabniku posebej prenesemo skladbo in jo nato pri vseh naenkrat začnemo predvajati. Medtem ko se skladba prenaša, vsi uporabniki čakajo, nakar jo lahko po končanem prenosu vsi skupaj začnejo upodabljati na platno.

Že ko se uporabnik poveže na stran, strežnik to zabeleži v seznam, ki vsebuje ID številko vsakega uporabnika. Po tej številki strežnik ve, kdo od uporabnikov je zahteval določeno akcijo. Ko na domači strani kliknemo na povezavo *Live Session*, se nam odpre nova stran, na kateri lahko vidimo vse sobe, ki so trenutno odprte in v katerih je vsaj en uporabnik. Ta seznam sob dobi odjemalčev JavaSript od strežnika preko *socket.io* in ga nato prikaže v oknu *Available Rooms*. Na vrhu tega okna je tudi gumb *Refresh*, ki osveži seznam ustvarjetnih sob. Od tu lahko nato tudi sami naredimo svojo sobo ali pa se pridružimo obstoječi.

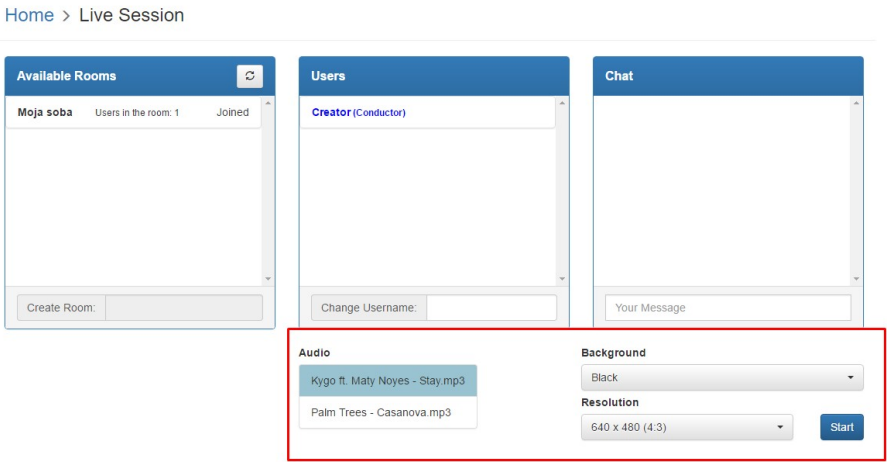


Slika 3.3: Izbira sob za Live način: na levi strani slike je okno z vsemi sobami, ki so na voljo. Trenutno sta na voljo dve sobi in sicer *Moja soba* in *Soba 1*. Spodaj pa je možnost *Create Room*, za ustvarjanje nove sobe.

3.2.1 Ustvarjanje nove sobe

Če želi uporabnik ustvariti svojo sobo, ima na dnu okna *Available Rooms*, vnosno polje (textbox) *Create Room*, kjer napiše želeno ime sobe. Ime, ki ga bo tu napisal, bodo videli vsi uporabniki, ki bodo po tem vstopili na to stran ter iskali sobe. Ko potdrdimo ime sobe, jo JavaScript pošlje strežniku, ki nato to ime pripne v seznam sob. Uporabnik, ki je ustvaril sobo, je v tej sobi klasificiran kot *dirigent* ali *conductor*. To pomeni, da on izbira vse možnosti seje, kot sta velikost in ozadje (belo ali črno) platna ter navsezadnje skladbo, ki se jo bo upodabljalo. To stori s pomočjo menija, ki ga vidi samo on, ostali uporabniki v sobi ga ne vidijo.

Velikost platna lahko izbere s padajočim menijem *Resolution*, kjer lahko izbere med petimi različnimi ločljivostmi platna. Najmanjša možna ločljivost je 640x480 pikslov, nato je na voljo 1024x768, ki je ravno tako v razmerju 4:3. Zatem pridejo na vrsto tudi *widescreen* ločljivosti, začenši s 640x360, nato 960x540 in pa najvišja 1280x720, ki je 720p HD ločljivost platna. Ločljivost se tako prilagodi tistemu uporabniku v sobi, ki ima najmanjši zaslon. Poleg platna mora na zaslonu namreč biti tudi prostor za orodja ter selekcijo barv, vse skupaj pa zasede okoli 500 pikslov širine zaslona. Skupaj s HD ločljivostjo je to 1720 pikslov v širino, kar pomeni, da bi to možnost lahko izbrali samo tisti z zaslonsko ločljivostjo 1920x1080 ali več. To je sicer danes najbolj pogosta ločljivost zaslonov, vendar pa imajo uporabniki lahko tudi zaslone starejših datumov.



Slika 3.4: Kreacija sobe: v oknu *Users* so navedeni vsi uporabniki trenutno v sobi *Moja soba*, z rdečo pa je označen menu, ki ga vidi le kreator sobe. Na strežniku sta naloženi dve zvočni datoteki - „Kygo ft. Maty Noyes - Stay.mp3“ ter „Palm Trees - Casanova“, med katerima lahko izbiramo tako, da kliknemo na želeno ime datoteke pod zavihkom *Audio*. V padajočem meniju *Background* sta na voljo dve barvi ozadja platna, bela in črna, v meniju *Resolution* pa vse ločljivosti platna.

3.2.2 Pridružitev sobi

Če se želi uporabnik pridružiti že obstoječi sobi, ima za to na voljo gumb tik pri imenu sobe. Tako odjemalec pošlje strežniku zahtevo za priključitev sobi skupaj z IDjem izbrane sobe. Strežnik nato preprosto doda tega uporabnika v seznam sob, in sicer v tisto sobo, ki se ji želi pridružiti. JavaScript na odjemalčevi strani, kot odgovor dobi seznam sob in uporabnikov v njih, iz seznama izbere sobo, v kateri je ta uporabnik, in v okno *Users* pripne vse uporabnike v tej sobi. Tako uporabnik vidi, kdo je z njim v sobi, ki se ji je pridružil. Uporabnikovo ime je obarvano z modro, imena ostalih uporabnikov pa so črne barve.

3.2.3 Sprememba uporabniškega imena

Ko uporabnik naredi svojo sobo, se mu samodejno dodeli uporabniško ime „Creator“, če pa se uporabnik pridruži sobi, je njegovo uporabniško ime „User i“, kjer je i zaporedna številka uporabnika v sobi. Če so v sobi že trije uporabniki, bo naslednji uporabnik, ki vstopi v sobo, imel uporabniško ime „User 4“, saj je četrti uporabnik po vrsti v sobi. V sobi bodo tako:

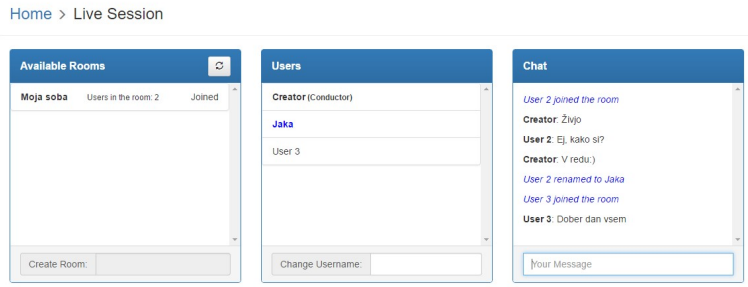
- „Creator“
- „User 2“
- „User 3“
- „User 4“

Uporabniki imajo seveda tudi možnost spremembe uporabniškega imena, kar lahko naredijo z možnostjo *Change Username*, ki se nahaja na dnu okna *Users*. Ko uporabnik vnese želeno ime, funkcija na strani odjemalca preveri, ali ime ni predolgo, in ga nato kot zahtevo za spremembo uporabniškega imena pošlje strežniku. Strežnik nato poišče sobo, kjer je uporabnik, in temu uporabniku spremeni ime. Nato vsem uporabnikom v sobi pošlje seznam, v katerem so njihova lastna imena. Vsi nato posodobijo imena uporabnikov v

sobi, prav tako pa se vsem v oknu *Chat* izpiše sporočilo: „*staro ime* renamed to v *novo ime*“, kjer je *staro ime* prejšnje uporabniško ime, *novo ime* pa spremenjeno uporabniško ime uporabnika.

3.2.4 Komunikacija med uporabniki

Uporabniki, ki so v isti sobi, lahko med sabo komunicirajo v oknu *Chat*, preko možnosti *Your message*. Tu uporabniki napišejo sporočilo, ki ga želijo poslati souporabnikom v sobi. Ko je s strani uporabnika sporočilo napisano, odjemalčeva funkcija pošlje to sporočilo strežniku, ki ga nato posreduje vsem uporabnikom v tej sobi, vključujoč uporabnika, ki je sporočilo poslal. To sporočilo se nato pripne na okno *Chat* uporabnikov in tako lahko vsi v sobi komunicirajo med sabo.



Slika 3.5: Komunikacija med uporabniki: kot se vidi iz slike, je v sobo, ki jo je naredil „Creator“, vstopil „User 2“ in začel komunicirati s „Creatorjem“. Nato si je „User 2“ zamenjal uporabniško ime v „Jaka“, nakar je v sobo vstopil še tretji uporabnik „User 3“.

3.2.5 Začetek live seje

Ko je vse pripravljeno in dirigent izbere skladbo, ozadje ter ločljivost platna, lahko začne sejo zase in ostale uporabnike v sobi. Ko to naredi, se sproži JavaScript funkcija na dirigentovi strani, ki pošlje strežniku podatke seje, ki jih je dirigent izbral. Strežnik nato te podatke (tudi samo skladbo) posre-

duje vsem udeležencem v seji, ki prilagodijo platno na določeno ločljivost in ga obarvajo z določeno barvo, kot to narekujeta podatka velikost ter barva ozadja. Tu sledi čas, ko morajo uporabniki počakati, da se vsem prenese skladba.

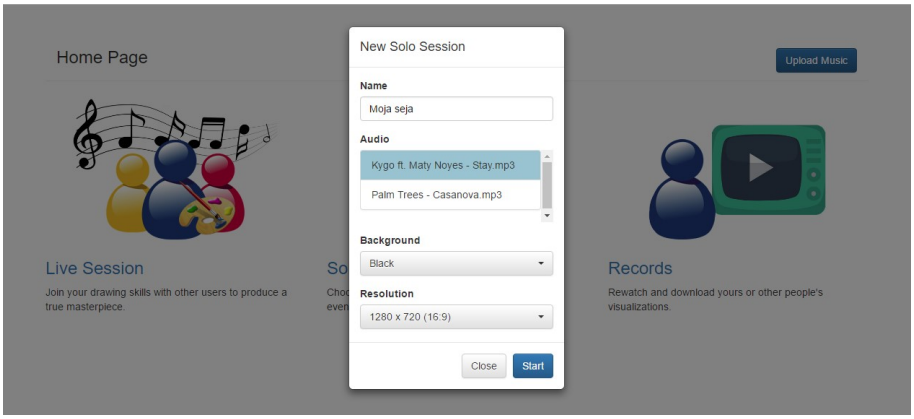
3.3 Solo način

V primeru, da uporabnik želi sam upodabljati glasbo, lahko za to uporabi samostojen ali *solo* način. Za razliko od *live* načina, je pri *solo* načinu en sam uporabnik, zato ni treba skrbeti za prenašanje podatkov do strežnika in nazaj med samim upodabljanjem. Celotno dogajanje je tako pri enem uporabniku, torej se odpre možnost za implementacijo še več različnih akcij, kot na primer razveljavi (undo), ki omogoča razveljavitev najnovejše akcije ter možnost različnih plasti (layers) za različne instrumente. Več o teh akcijah v poglavju *Dodatne možnosti solo načina*.

3.3.1 Začetek solo seje

S pritiskom na povezavo *Solo Session* se nam odpre Bootstrapovo pojavno okno, v katerem lahko izberemo ime in skladbo seje ter ozadje in ločljivost platna. Tu moramo izbrati enake možnosti kot pri *live* seji, torej ime in skladbo seje ter ozadje in ločljivost platna, na katerega bomo upodabljali. Ko je ime napisano in vse drugo izbrano, lahko sejo začnemo s klikom na gumb *Start*.

Z začetkom seje se odpre nova stran, kjer moramo najprej počakati, da se skladba, ki smo jo izbrali, naloži iz strežnika na odjemalčevo stran za predvajanje. Zatem je vse pripravljeno na upodabljanje na platno.



Slika 3.6: Začetek nove solo seje: ko uporabnik klikne gumb *Start*, bo začel novo sejo pod imenom „Moja seja“, s skladbo „Kygo ft. Maty Noyes - Stay“ ter platnom velikim 1280x720, torej HD ločljivost in črnim ozadjem.

3.4 Upodabljanje na platno

V nadaljevanju bom razložil, kako deluje samo upodabljanje na platno ter kaj se v live in solo načinu v ozadju dogaja pri odjemalcu in na strežniku.

Odjemalčev JavaScript ima tudi vrsto globalnih spremenljivk, na primer za izbrano orodje, izbrano barvo ter izbrano velikost. Ko uporabnik izbere želeno orodje za upodabljanje, se ta globalna spremenljivka spremeni, enako za barvo in velikost orodja.

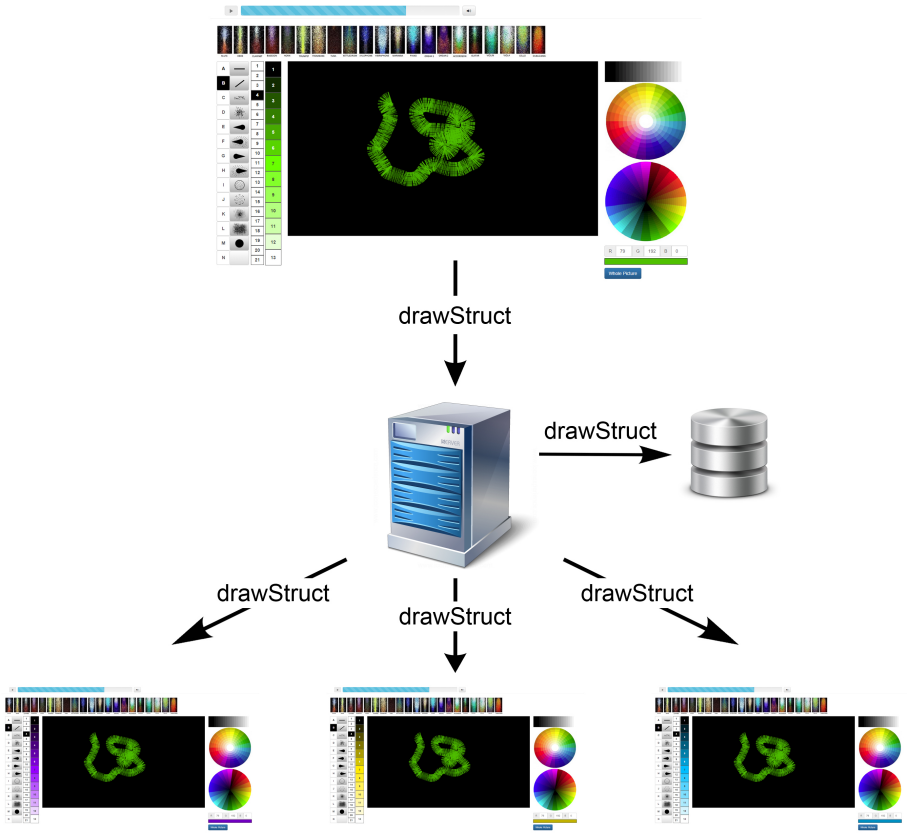
Funkcija *findxy* se kliče, ko nad platnom izvajamo akcije z miško in nato preveri, na katerih koordinatah platna smo kliknili, izračuna kot (angle), ostale argumente pa ima že shranjene v globalnih spremenljivkah. Nato s temi podatki kliče funkcijo *draw*, ki na mesto, določeno s koordinatama nariše podano orodje s podanimi parametri. Ta funkcija ima sedem argumentov:

- koordinata x
- koordinata y
- orodje (tool)
- barva (color)
- velikost (size)
- kot (angle)
- klik (clicked)

Koordinati x in y povesta, kje točno na platnu naj se izriše izbrano orodje. Barva pove, s katero barvo naj se orodje izriše, velikost pa s kakšno velikostjo. Kot ali „angle“ služi temu, da funkcija ve, pod katerim kotom naj izriše črto. Ta argument sicer uporablja samo eno orodje, ki ga bom opisal pozneje. Zadnji argument, klik ali „clicked“, je *true* če je uporabnik kliknil na platno, *false* pa če uporabnik drži miškin gumb in vleče čez platno.

V live načinu se ob vsakem klicanju funkcije *findxy* podatke, ki so pomembni za izris določene akcije, shrani v podatkovno strukturo, ki jo funkcija

draw potrebuje za izris te akcije. To strukturo odjemalec pošlje strežniku, ki jo pripne v seznam *record*. Tako imamo na koncu seje na strežniku na voljo seznam, iz katerega lahko skonstruiramo celotno glasbeno upodobitev. Strežnik nato to strukturo posreduje ostalim uporabnikom v seji, ki lahko s funkcijo *draw*, izrišejo to akcijo na platno.



Slika 3.7: Interaktivna vizualizacija glasbe: v seji na sliki sodelujejo dirigent ter trije uporabniki. Na vrhu slike je uporabniški vmesnik dirigenta, ki je na platno izrisal več akcij določenega orodja (tu je izrisanih približno 4000 akcij). Podatkovna struktura *drawStruct*, se nato kot prikazuje slika, prenese od dirgenta do strežnika in od strežnika do podatkovne baze ter ostalih uporabnikov v seji.

3.4.1 Orodja za upodabljanje

Na levi plošči imamo na izbiro trinajst različnih orodij za upodabljanje glasbe, ki so označena od A do M. Prostor je še za eno, torej štirinajsto orodje, ki ga trenutno še ni, ker je najzahtevnejše za implementirati.

Pri funkciji *draw* moramo še omeniti implementacijo razpršila (spray tool). Razpršilo je orodje, ki po določeni površini izriše določeno število pikslov; v bistvu so to majhni kvadrati dolžine in širine 1, zato so veliki kot piksli. S tem učinkom dobimo na platnu videz razpršila. Ta učinek dosežemo na naslednji način. Ob inicializaciji platna odjemalec dobi od strežnika tudi naključna števila od 0 do 1000 in pa naključne vrednosti kotov od 0 do 2PI. Te vrednosti funkcija *draw* uporabi, ko dobi zahtevo za izris orodij, ki uporabljajo funkcijo razpršila. Razpršilo je implementirano na dva načina, pri čemer eden vrne vizualno drugačen rezultat od drugega.

Prvi način uporablja samo naključna števila od 0 do 1000 in sicer tako, da za vsak piksel razpršila vzame 2 naključni števili. Ti števili bosta predstavljali odmik tega piksla po x in y koordinati od tistega mesta, kjer uporabnik klikne z miško. Najprej te števili pretvori v dejanski odmik v pikslih tako, da sta delež od 0 do 1000 ter delež od 0 do velikosti, ki jo izbere uporabnik, proporcionalna. Nato izriše piksel na tistem mestu, ki ga narekuje odmik.

Drugi način uporablja kot (angle) ter odmik od mesta klika miške za izris vsakega piksla. Uporablja torej nakjlučne vrednosti kotov in naključne vrednosti od 1 do 1000 za odmik iz sredine krožnice, ki obdaja razpršilo. To vizualno izgleda tako, da je na sredini te krožnice večja gostota pikslov. Bolj kot se pomikamo navzven, proti krožnici ali robu razpršila, redkejši postajajo piksli.

V nadaljevanju bom navedel teh trinajst orodij v slovenskem in v angleškem jeziku (v oklepaju), jih opisal in za vsako podal sliko, na kateri se vidi, kako orodje izgleda na platnu.

A: Vodoravna črta (horizontal line tool)

Vodoravna črta je orodje, ki na mestu klika izriše vodoravno črto debeline 1 piksel in dolžine, ki je odvisna od izbrane velikosti.

Slika 3.8: *Vodoravna črta* črne barve: leva stran slike je kliknjeno orodje, desna pa orodje potegnjeno po platnu v diagonali iz smeri levo zgoraj v smeri desno spodaj.

B: Cirkularna črta (circlular line tool)

To orodje je edino, ki potrebuje argument kota (angle) v funkciji *draw*. Izraz cirkularna izhaja iz tega, ker se črta nekako vrti v smeri miške. Črta je sicer neenakomerno črtkana, na sredini so prostori med črno manj pogosti, proti robovoma črte pa vedno bolj pogosti. Debelina črte in dolžina sta enaki kot pri vodoravni črti, torej debelina 1 piksel, dolžina pa odvisna od izbrane velikosti.

Slika 3.9: *Cirkularna črta* črne barve: slika prikazuje poteg orodja po platnu v smeri črke S od zgoraj desno do spodaj levo.

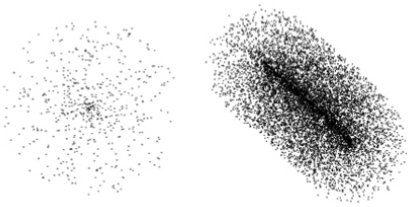
C: Pravokotno razpršilo (rectangle spray tool)

To je orodje, ki uporablja drugi način razpršila in ga omeji na območje pravokotnika. Meja razpršila je torej pravokotnik, katerega stranici sta v razmerju 4:1, kjer sta zgornja in spodnja večji stranici.

Slika 3.10: *Pravokotno razpršilo* črne barve: leva stran slike je kliknjeno orodje, desna pa orodje potegnjeno po platnu v diagonali iz smeri levo zgoraj v smeri desno spodaj.

D: Razpršilo (spray tool)

To je navadno razpršilo, kakršnega smo navajeni pri večini grafičnih urejevalnikov. To orodje uporablja prvi način razpršila, meja razpršila pa je krožnica, katere polmer je odvisen od izbrane velikosti orodja.



Slika 3.11: *Razpršilo* črne barve: leva stran slike je kliknjeno orodje, desna pa orodje potegnjeno po platnu v diagonali iz smeri levo zgoraj v smeri desno spodaj.

E: Naraščajoči čopič (increasing brush tool)

To orodje je zelo zanimivo, saj predstavlja čopič, ki s časom narašča v velikosti. Ko uporabnik klikne z orodjem na platno, je velikost vedno 0, ne glede na izbrano velikost. Medtem ko uporabnik z orodjem vleče po platnu, pa velikost orodja narašča za 2% prejšnje velikosti orodja vsako desetinko sekunde, in sicer vse do izbrane velikosti. To ustvari lep in gladek učinek večanja velikosti orodja v realnem času. Rob čopiča ni oster, saj je celoten čopič obdan z drugim načinom razpršila, ki je le za vzorec večji od čopiča in zato služi kot lep prehod med ozadjem platna in čopičem.

Slika 3.12: *Naraščajoči čopič* črne barve: slika prikazuje poteg orodja po platnu v smeri črke S od zgoraj desno do spodaj levo.

F: Naraščajoči čopič z razpršilom (increasing brush spray tool)

To je enako orodje kot *naraščajoči čopič*, le da vsebuje dodaten drugi način razpršila, ki je izrazito večji od samega čopiča.

Slika 3.13: *Naraščajoči čopič z razpršilom* črne barve: slika prikazuje poteg orodja po platnu v smeri črke S od zgoraj desno do spodaj levo.

G: Padajoči čopič (decreasing brush tool)

Gre za enako orodje kot *naraščajoči čopič*, le da se velikost orodja s časom manjša in ne veča.

Slika 3.14: *Padajoči čopič* črne barve: slika prikazuje poteg orodja po platnu v smeri črke S od zgoraj desno do spodaj levo.

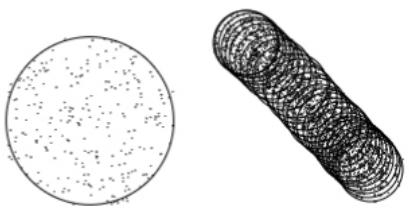
H: Padajoči čopič z razpršilom (decreasing brush spray tool)

To je enako orodje kot *naraščajoči čopič z razpršilom*, le da se velikost orodja s časom manjša in ne veča.

Slika 3.15: *Padajoči čopič z razpršilom* črne barve: slika prikazuje poteg orodja po platnu v smeri črke S od zgoraj desno do spodaj levo.

I: Razpršilo s krožnico (full circle spray tool)

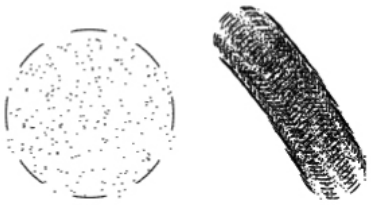
Orodje, ki uporablja prvi način razpršila in poudari krožnico, ki služi kot rob tega razpršila. Krožnica je črta debeline 1 piksel, premer pa je enak izbrani velikosti orodja.



Slika 3.16: *Razpršilo s krožnico* črne barve: leva stran slike je kliknjeno orodje, desna pa orodje potegnjeno po platnu v diagonali iz smeri levo zgoraj v smeri desno spodaj.

J: Razpršilo s črtasto krožnico (striped circle spray tool)

Gre za enako orodje kot *razpršilo s krožnico*, le da je krožnica črtasta, torej sestavljena iz prekinjenih črtic, kjer vsaka črtica predstavlja osmino celotne krožnice.



Slika 3.17: *Razpršilo s črtasto krožnico* črne barve: leva stran slike je kliknjeno orodje, desna pa orodje potegnjeno po platnu v diagonali iz smeri levo zgoraj v smeri desno spodaj.

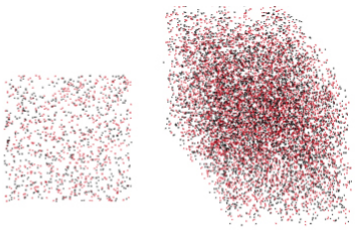
K: Dvobarvno razpršilo (two-colored spray tool)

To orodje vsebuje in izriše oba načina razpršila s krožnico kot robom. Oba načina sta enako velika, razlika je le, da uporabnik izbira barvo za prvi način razpršila, drugi način, torej s kotom in odmikom od središča, pa je vedno črne barve.

Slika 3.18: *Dvobarvno razpršilo* rdeče barve: leva stran slike je kliknjeno orodje, desna pa orodje potegnjeno po platnu v diagonali iz smeri levo zgoraj v smeri desno spodaj.

L: Kvadratno dvobarvno razpršilo (two-colored square spray tool)

To je enako orodje kot *dvobarvno razpršilo*, le da rob razpršila ni krožnica, ampak kvadrat. Zaradi tega tudi ne moremo uporabiti drugega načina razpršila, ker mora v tem primeru razpršilo biti okroglo. Orodje torej uporablja prvi način razpršila dvakrat, barvo enega izbere uporabnik, barva drugega pa je črna.



Slika 3.19: *Kvadratno dvobarvno razpršilo* rdeče barve: leva stran slike je kliknjeno orodje, desna pa orodje potegnjeno po platnu v diagonali iz smeri levo zgoraj v smeri desno spodaj.

M: Čopič (brush tool)

Gre za navaden čopič, kot bi ga pričakovali v večini grafičnih urejevalnikov. Deluje enako kot orodje *naraščajoči čopič*, le da je velikost konstantno enaka izbrani velikosti in ne vsebuje razpršila, rob je torej oster.

Slika 3.20: *Čopič* črne barve: leva stran slike je kliknjeno orodje, desna pa orodje potegnjeno po platnu v diagonali iz smeri levo zgoraj v smeri desno spodaj.

3.4.2 Izbira barve orodja

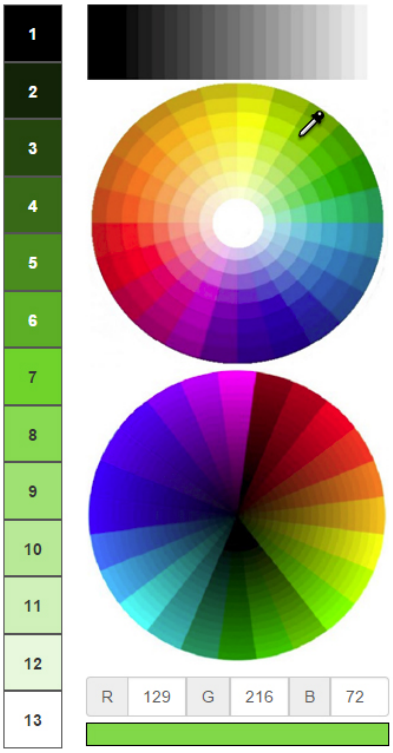
Barva orodja je pomembna za izražanje upodobitve glasbe. Pri tej diplomski nalogi, uporabniški vmesnik vsebuje izbirnik barve, ki se nahaja na desni plošči (right panel), torej desno od platna. Vsebuje dva barvna kroga, velika 256x256 pikslov, in barvni trak velikosti 265x40 pikslov. Oba kroga vsebujeta čiste barve na robu, pri zgornjem krogu se barve proti sredini svetlijo, pri spodnjem pa temnijo, barvni trak pa vsebuje odtenke sive, od čiste črne barve na levi do čiste bele barve na desni.

Celoten izbirnik barve je v bistvu platno ali *canvas*, ki ima kot ozadje sliko obeh krogov in traka. Ko uporabnik z miško klikne na to platno, se na enak način kot pri sredinskem platnu, dobi x in y koordinato klika. JavaScript funkcija nato poišče, kakšne barve je piksel na tej lokaciji slike in iz tega dobi vrednosti *RGB*, torej rdečo, zeleno in modro vrednost barve. Miškin gumb lahko uporabnik tudi drži in vleče po izbirniku, medtem pa se nenehno spreminja izbrana barva.

Vsakič ko se barva spremeni, se v globalno spremenljivko *color* zapiše šestnajstiška vrednost izbrane barve. To barvo se nato uporabi za ozadje prikazovalnika barve, ki se nahaja pod izbirnikom. Tako uporabnik vidi, katera barva je trenutno izbrana. Izbira nove barve vpliva tudi na izbirnik svetlosti barve, ki se nahaja na levi plošči uporabniškega vmesnika in je sestavljen iz trinajstih kvadratov, vsak od njih pa predstavlja eno svetlost izbrane barve. To deluje tako, da se izbrana barva iz *RGB* vrednosti pretvori v *HSL* vrednost. H in S vrednost ostaneta enaki, L ali svetlost pa je v prvem kvadratu 0 (barva je torej črna), v zadnjem 100 (barva je bela), vmes pa linearno narašča in kvadrate obarva s pripadajočo barvo. Tako bo izbirnik za svetlost barve enak v primeru, ko izberemo čisto barvo in ko izberemo enako, vendar bolj svetlo ali temno barvo. Če uporabnik želi upodabljati z barvo v enem izmed kvadratov v izbirniku za svetlost barve, klikne na ta kvadrat in izbrana barva se spremeni na tisto, ki je v ozadju kvadrata.

Med izbirnikom barve in prikazovalnikom barve, se nahajajo tri vnosna polja, v katere lahko uporabnik vpiše vrednosti *RGB* želene barve. Te vre-

dnosti se spremenijo vedno, ko uporabnik izbere barvo s pomočjo izbirnika barve in tako služijo tudi kot informacija, kakšne vrednosti *RGB* smo izbrali.

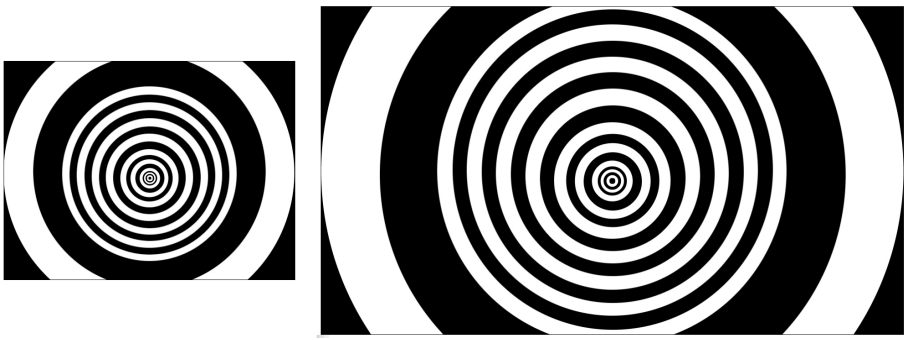


Slika 3.21: Izbirnik barve orodja: vidi se, da se nad izbirnikom miškin kazalec spremeni v kapalko, s katero nato izberemo barvo. V tem primeru je uporabnik kliknil na svetlo zeleno barvo, ki se pokaže v prikazovalniku barve spodaj. Spremenijo se tudi vrednosti *RGB* in barve v izbirniku za svetlost barve na levi strani. Ta svetlo zelena barva ima naslednje *RGB* vrednosti: R - 129, G - 216, B - 72.

3.4.3 Izbira velikosti orodja

Uporabnik ima na voljo 21 različnih velikosti orodja, med katerimi lahko izbira v izbirniku, ki se nahaja na levi plošči (left panel), med izbirnikom orodja in izbirnikom svetlosti barve orodja. Zaradi možnosti izbire med različnimi ločljivostmi platna mora biti velikost orodja implementirana tako, da vse velikosti orodja predstavljajo določen delež platna. Tako na primer velikost 10 pokrije 20% platna in če izberemo na primer *čopič* te velikosti, bo premer pike, ki jo izriše to orodje, enak 20% velikosti platna. Če bi bile velikosti orodja fiksne, bi na platnu z veliko ločljivostjo ista velikost izgledala dosti manjša kot pa na platnu z manjšo ločljivostjo. Vsaka številka velikosti orodja predstavlja svoj delež velikosti platna, številka 1 na primer predstavlja 1% platna, 5 pa predstavlja 5%, pri manjših velikostih torej procenti naraščajo počasneje kot pri večjih. To se dogaja, ker se za upodabljanje uporabljajo večinoma velikosti od 1 do 10, zato je tukaj ponujena večja izbira velikosti. Velikost 10 predstavlja že 20% platna, 15 že 45%, 20 pa 100%. Ko je premer čopiča velikosti 20, ta čopič še vedno ne pokrije celotne površine, saj ne doseže prostorov v kotih platna in zato tu nastane problem. Temu je namenjena velikost 21, ki predstavlja 200% velikosti platna, vsako orodje bo s to velikostjo pokrilo celotno površino platna.

S klikom na želeno velikost orodja se ozadje za številko kliknjene velikosti spremeni iz belega v črno in sama barva številke iz črne v belo. Tako uporabnik ve, katera je izbrana velikost. Kot pri barvi orodja se tudi tu v odjemalčevem JavaScriptu uporablja globalna spremenljivka *drawSize*, ki hrani delež velikosti platna, torej velikost orodja, ki jo bo uporabila funkcija *draw* za izris orodja.

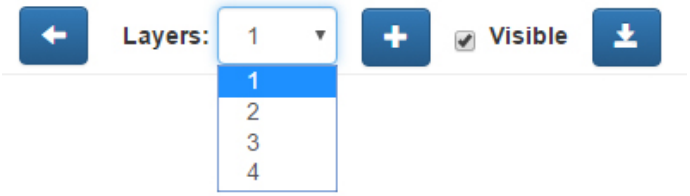


Slika 3.22: Prikaz različnih velikosti orodij: na levi strani je najmanjše platno, ločljivosti 640x480, na desni pa največje platno, ločljivosti 1280x720. Uporabljeno orodje je *čopič* in sicer najprej velikosti 21, črne barve, kliknjeno na sredino platna. Nato spet kliknjeno na sredino platna, tokrat velikosti 20 in bele barve, zatem velikosti 19 in črne barve in tako naprej, do velikosti 1, ki jo predstavlja najmanjša črna pika na sredni platna.

3.4.4 Dodatne možnosti solo načina

Kot že omenjeno, ima v solo načinu uporabnik na voljo določene možnosti, ki jih v live načinu ni. Ena od teh možnosti je upodabljanje na več različnih plasti, vsaka plast se lahko na primer uporabi za en instrument. Druga možnost je razveljavi, ki izbriše najnovejšo akcijo z orodjem. Ti dve akciji ter akcija za konec seje so na voljo v t.i. „solo meniju“, ki se nahaja v glavi strani, zgoraj desno. Seveda se meni pojavi le, če smo v solo načinu, v live načinu je namreč skrit.

Prva stvar v meniju, gledano od leve proti desni, je gumb *Undo*. Nato sledi padajoči menu *Layers* (plasti), kjer lahko s pomočjo potrditvenega polja *Visible* izberemo če bo izbrana plast vidna na platnu ali ne. Zraven je še gumb za novo plast, ki naredi novo plast, na katero lahko potem uporabnik upodablja glasbo. Na desni strani menija pa je gumb *End Session*, ki zaključi solo sejo in vrne uporabnika na domačo stran.



Slika 3.23: Solo meni: gumb *Razveljavi* na levi strani, sledi mu padajočim meni *Layers*, kjer vidimo, da so v tej seji štiri plasti končane, uporabnik pa riše na peto. Zraven menija je gumb *New Layer*, potrditveno polje *Visible* in gumb *End Session*.

Možnost razveljavi

Razveljavi (undo) je pogosta in še kako uporabna možnost v grafičnih urejevalnikih. Strežnik v live načinu shranjuje akcije vsakega uporabnika. Enako je treba narediti pri solo načinu, le da to naredimo na strani odjemalca. Možnost razveljavi deluje namreč tako, da se vedno, ko uporabnik razveljavi

akcijo, čez celotno platno izriše bel ali črn štirikotnik, odvisno od izbrane barve ozadja, preko katerega se nato izrišejo vse akcije, razen zadnje. Vsakič, ko uporabnik klikne na platno, si funkcija namreč zapomni trenutno stanje in čas. To se nato uporabi za mejno točko, do koder se izrišejo akcije. Vse, kar je izrisano za to točko, se razume kot nazadnje izvedena akcija.

Plasti (layers)

V solo načinu ni različnih uporabnikov za različne instrumente, zato je treba enemu uporabniku dati možnost, da loči različne instrumente. Pri tem zelo pomagajo plasti. Vsaka plast predstavlja skupek akcij, izrisanih na platno. Ko uporabnik prvič začne upodabljati na platno v solo seji, upodablja na prvo plast. Če želi ustvariti novo plast, pritisne na gumb *New Layer* z ikono plusa. V tem trenutku se v padajoči meni *Layers* doda številka 1, ki predstavlja prvo plast, ki se je dodala h končanim plastem. Ta plast vsebuje celotno sliko, ki jo je uporabnik izrisal do trenutka, ko je pritisnil na *New Layer*. Predvajanje skladbe se ustavi in postavi na začetek, tako je vse pripravljeno, da uporabnik začne upodabljati na novo ustvarjeno plast. Tu se lahko uporabnik odloči, ali naj bo plast 1 v ozadju vidna ali ne. To naredi tako, da obkljuka ali odkljuka potrdivetno polje *Visible*, medtem ko je v padajočem meniju *Layers* označena plast, kateri želi spremeniti vidnost. Enako lahko naredi za vsako končano plast posebej.

Zaključek seje

Ko se uporabnik odloči, da so vse plasti končane in je zadovoljen s končnim izdelkom, lahko konča sejo s pritiskom na gumb *End Session*. Zatem se vse akcije vsake plasti združijo v en skupen seznam in razvrstijo po času izrisa. V takem vrstnem redu morajo namreč biti, da jih lahko potem izrišemo kot posnetek te seje. Celoten seznam se nato pošlje strežniku, ki ga shrani v še večji seznam *Records*. To je arhiv vseh posnetkov, narejenih na strani.

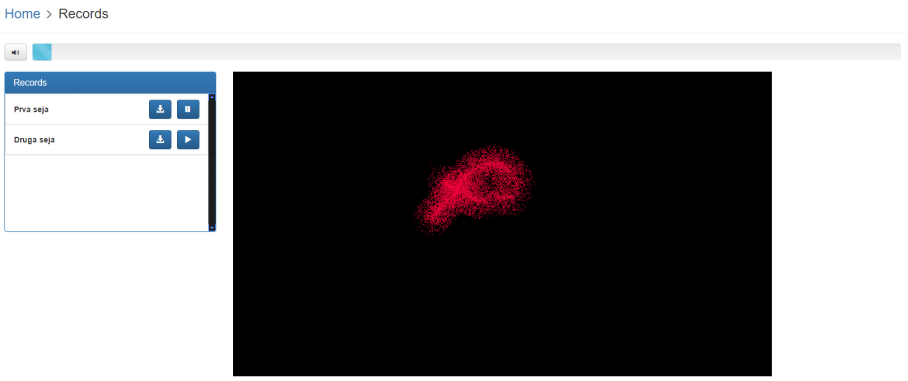
3.5 Posnetek seje

Po koncu vsake live ali solo seje se vse akcije, ki so bile uporabljene za upodobitev glasbe, pošljejo na strežnik. Ko uporabnik klikne na povezavo Records, ki se nahaja na domači strani, pošlje strežniku zahtevo za imena posnetkov. Strežnik mu odgovori s seznamom imen vseh posnetkov, ki so na voljo za ogled. Odjemalčev JavaScript nato uporabniku prikaže ta seznam v oknu *Records*. Vsak posnetek je v tem oknu predstavljen v svoji vrstici, ki vsebuje ime posnetka in 2 gumba:

- gumb *Download* za prenos posnetka iz strežnika na uporabnikov disk,
- gumb *Play* za predvajanje posnetka kar na strani.

V glavi strani sta časovnica za prikaz predvajanja posnetka in pa gumb *Mute*, ki utiša predvajano skladbo.

Če uporabnik želi predvajati posnetek kar v brskalniku, pritisne gumb *Play* pri želenem posnetku. Ta pošlje strežniku zahtevo za prenos akcij upodabljanja in skladbe, ki pripada temu posnetku. Ko uporabnik prejme zahtevane podatke, inicializira platno po enakem principu kot pri samem upodabljanju glasbe. Nato začne istočasno predvajati skladbo in izrisovati akcije na platno v pravilnem časovnem zaporedju, kot so bile shranjene. Uporabnik lahko tudi ustavi posnetek z gumbom *Pause*, ki je, ko se posnetek predvaja, na istem mestu, kot je prej bil gumb *Play*. Če pritisne gumb *Pause*, se ta nato spet spremeni v gumb *Play*, s katerim lahko uporabnik nadaljuje predvajanje posnetka od trenutka, ko ga je ustavil.



Slika 3.24: Predvajanje posnetka v brskalniku: na levi je okno *Records*, z vsemi posnetki, ki so na voljo. Vidimo, da se predvaja posnetek „Prva seja“, saj ima namesto gumba *Play* prikazan gumb *Pause*.

Možno je tudi prenesti posnetek s strani, v tem primeru se od strežnika zahteva enake podatke kot prej, le da zdaj odjemalčev JavaScript namesto da predvaja posnetek s pomočjo platna, le-tega pripravi za prenos v formatu *MP4*. To naredi tako, da najprej na platnu predvaja celoten posnetek, medtem pa snema platno in tako dobi video del formata *MP4*, ki je zaenkrat v formatu *WebM*[1]. Ta *WebM* se nato z zvokom združi v skupen *MP4* format s pomočjo knjižnice *ffmpeg*, ki je JavaScriptova knjižnica za obdelavo *videa* in zvoka. Uporabi se seveda skladba, ki pripada temu posnetku.

Uporabnik mora nato počakati kar nekaj časa, odvisno od dolžine posnetka, da kodirnik opravi svoje delo. Medtem je na strani prikazano pojavno okno, ki uporabnika obvešča o celotnem postopku. Ko je postopek končan, se na pojavnem oknu pojavi gumb *Download MP4*, s katerim lahko datoteko *„ime seje“.mp4* tudi prenesemo na svoj lokalni disk.

[1]WebM je odprtokodni video in avdio format, zasnovan za splet. [2]

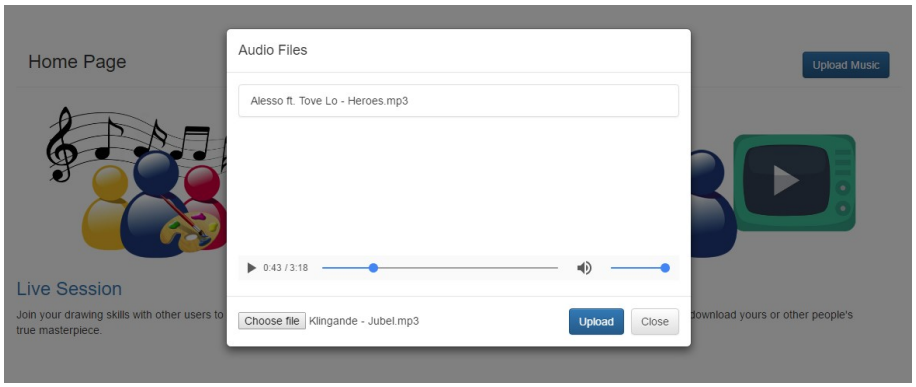
3.6 Nalaganje glasbe na strežnik

Pred začetkom upodabljanja glasbe je treba na strežnik naložiti skladbo, ki jo bomo upodobili. Za to imamo možnost *Upload Music*, ki je v desnem zgornjem kotu glave domače strani. V tem primeru se prikaže pojavno okno *Audio Files*, v katerem lahko poslušamo skladbe, ki so na lokalnem disku in jih nato po želji tudi naložimo na strežnik. V telesu pojavnega okna sta dva elementa in sicer seznam, ki prikazuje katere skladbe so že naložene na strežniku in pa element HTML *audio*, ki služi kot predvajalnik, s katerim lahko predvajamo skladbe iz svojega računalnika:

```
<audio id='audio-player' autoplay controls></audio>
```

Autoplay pomeni, da bo začel samodejno predvajati glasbo in *controls* pomeni, da so vidni gumbi za manipuliranje s predvajano skladbo. Na desni strani noge pojavnega okna sta dva gumba in sicer *Upload*, s katerim naložimo skladbo na strežnik in *Close*, ki preprosto zapre pojavno okno.

Ko izberemo skladbo z lokalnega diska, JavaScript preveri, ali je to res glasbena datoteka in če je, jo posreduje predvajalniku, ki jo samodejno začne predvajati. Uporabnik lahko nato po želji ustavi, nadaljuje, spreminja glasnost in se premakne na katerikoli časovni trenutek v skladbi. Uporabnik lahko nato klikne na gumb *Upload*, ki bo sprožil nalaganje skladbe na strežnik. Med nalaganjem se levo od gumba *Upload* prikaže indikator, ki prikazuje, da poteka nalaganje, gumb *Upload* pa je ta celoten čas onemogočen. Medtem lahko uporabnik po želji še naprej posluša skladbo, ko pa se nalaganje zaključi, se ime skladbe prikaže na seznamu. To pomeni, da je skladba dostopna na strežniku. Glasbo lahko nalaga več uporabnikov hkrati in vsakemu se bodo na seznamu glasbe pokazale vse skladbe, ki so bile naložene na strežnik.



Slika 3.25: Pojavno okno *Audio Files*: na sliki vidimo, da je na strežniku ena zvočna datoteka in sicer „Alesso ft. Tove Lo - Heroes.mp3“, uporabnik pa preko predvajalnika glasbe posluša „Klingande - Jubel.mp3“. To datoteko lahko naloži z gumbom *Upload*, lahko pa izbere drugo z gumbom *Choose File*.

Poglavje 4

Zaključek

Programska vizualizacija glasbe ni tako popularna kot bi pričakovali. Je sicer zanimiva za opazovanje, to pa je bolj ali manj vse. Ljudje pri opazovanju različnih učinkov vizualizacije glasbe, ki se radi ponavljajo, kmalu izgubimo interes. Navada je poslušanje glasbe v ozadju, medtem ko se primarno ukvarjamo z drugimi stvarmi, tudi če gre za vizualizacijo. Upodabljanje glasbe, ki je v celoti postavljeno v roke uporabnika, kot je prikazano v tej diplomski nalogi, niti ni prav priljubljeno. Vendar pa je razlog za to lahko tudi dejstvo, da za tako upodabljanje, enostavno ni na voljo nobene ali vsaj dovolj dobre programske opreme.

Mogoče bi lahko k temu prispeval prav ta sistem, trenutno je sicer mišljen za uporabo ožje skupine ljudi, ki se z upodabljanjem glasbe intenzivno ukvarja, vendar bi se ga dalo pripraviti tudi za širšo uporabo preko medmrežja. Izbrati je treba le gostitelja spletnih strani, ki bi gostoval celoten sistem in rezervirati domeno, kamor bi se lahko povezal kdorkoli in začel upodabljati glasbo.

Ena izmed možnih izboljšav sistema je več funkcij pri solo načinu, na primer poljubno premikanje po časovnici in popravljanje napak na platnu v točno določeni časovni točki. Prav tako bi lahko enako funkcijo implementiral pri posnetku, torej da bi lahko med ogledom posnetka uporabnik časovno kjerkoli spreminjal akcije in s tem samo upodobitev glasbe. Potem je tu

seveda možnost celotne uporabniške baze, kjer bi bil vsak uporabnik dejansko registriran na strežniku in imel svoj arhiv upodobitev, lahko bi imel tudi možnost ocenjevanja upodobitev drugih uporabnikov ipd. Implementacija foruma, kjer bi se uporabniki pogovarjali o upodabljanju in drugih stvareh, bi bila ena od mnogih možnosti, ki bi pripomogle k razvitju celotne skupnosti okoli te spletne strani.

Vse te funkcije so sicer še daleč od delujočih, vendar se jih lahko, tako kot vse ostalo, z malo volje in dela, spremeni v resničnost.

Literatura

[1] Bootstrap. Bootstrap buttons and navigation. http://www.bootstrapcss.com/images/bootstrap.png.

[2] The WebM Project. WebM. https://www.webmproject.org/about/faq/.

[3] W3 Schools. HTML Head. http://aiweb.techfak.uni-bielefeld.de/content/bworld-robot-control-software/.

[4] B. Svetek. *Ustvarjanje svetlobe: optična podoba glasbe*. Slovenska filharmonija, 2014.

[5] Wikipedia. HTML. https://en.wikipedia.org/wiki/HTML.