BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

APPLICATION FOR ONLINE MUSIC NOTATION TYPE-SETTING

APLIKACE PRO ONLINE SAZBU NOTOVÉHO ZÁPISU

BACHELOR'S THESIS
BAKALÁŘSKÁ PRÁCE

AUTHOR JAKUB MACKOVIČ
AUTOR PRÁCE

SUPERVISOR Ing. RADEK BURGET, Ph.D.
VEDOUCÍ PRÁCE

BRNO 2016

Zadání bakalářské práce/18899/2015/xmacko09

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav informačních systémů Akademický rok 2015/2016

Zadání bakalářské práce

Řešitel:	**Mackovič Jakub**
Obor:	Informační technologie
Téma:	**Aplikace pro online sazbu notového zápisu**
	Application for Online Music Notation Typesetting
Kategorie:	Web

Pokyny:

1. Seznamte se s možnostmi tvorby interaktivních aplikací využívajících vektorovou grafiku ve webovém prohlížeči.
2. Prostudujte existující nástroje pro sazbu notového zápisu.
3. Navrhněte aplikaci pro interaktivní tvorbu notového zápisu ve webovém prohlížeči.
4. Implementujte navrženou aplikaci s využitím vhodných technologií.
5. Po dohodě s vedoucím implementujte možnost exportu vytvořeného notového zápisu do zvolených běžně používaných formátů.
6. Proveďte testování aplikace a zhodnoťte dosažené výsledky.

Literatura:

- Gutmans, A., Rethans, D., Bakken, S.: Mistrovství v PHP 5, Computer Press, 2012
- Žára, O.: JavaScript - Programátorské techniky a webové technologie, Computer Press, 2015
- Eisenberg, J. D.: SVG Essentials, O'Reilly Media, 2002
- Scalable Vector Graphics, The Wold Wide Web Consortium, http://www.w3.org/Graphics/SVG/

Pro udělení zápočtu za první semestr je požadováno:

- Body 1 až 3

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí:	**Burget Radek, Ing., Ph.D.**, UIFS FIT VUT
Datum zadání:	1. listopadu 2015
Datum odevzdání:	18. května 2016

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav informačních systémů
612 66 Brno, Božetěchova 2

L.S.

[illegible]

doc. Dr. Ing. Dušan Kolář
vedoucí ústavu

Abstract

In this thesis, an application for musical notation in the environment of web browsers using the SVG vector graphics format is proposed. We have a look at relevant standard notation practices as well as at the history of musical typesetting. Existing programs for musical notation are classified and examined and based on those a user interface for the application is designed along with some modifications. The implementation of the application is then described. A set of features is presented and their usefulness is with the user interface tested on users. The feedback of the testing is then evaluated and the application is improved based on the feedback. Finally, the future of the application is discussed. Because SVG can still be considered an experimental technology and many browsers lack its support or their behaviour differs, the application is only guaranteed to work on the Google Chrome browser.

Abstrakt

V tejto práci je predstavená aplikácia pre online zápis hudby s použitím SVG vektorovej grafiky v prostredí webových prehliadačov. Pozrieme sa na relevantné postupy pri notovom zápise a na históriu sadzby hudobného zápisu. Preskúmame a roztriedime existujúce programy na notový zápis. Na ich základe navrhneme užívateľské rozhranie pre aplikáciu s niekoľkými obmenami. Nasleduje popis implementácie aplikácie. Predstavíme tiež sadu funkcionalít, ktorých užitočnosť spolu s užívateľským rozhraním otestujeme na užívateľoch. Vyhodnotíme výsledky testovania a na jeho základe vylepšíme niektoré črty aplikácie. Nakoniec vyhodnotíme aplikáciu a jej budúci vývin. Keďže SVG sa stále považuje za experimentálnu technológiu a niektoré prehliadače ju stále nepodporujú alebo sa ich implementácia líši, bola aplikácia vytvorená tak, aby fungovala na prehliadači Chrome od spoločnosti Google.

Keywords

Musical typesetting, musical notation, typesetting, notation, music, SVG, vector graphics, JavaScript, interactive, online, web

Klíčová slova

Hudobná sadzba, hudobný zápis, sadzba, zápis, hudba, SVG, vektorová grafika, JavaScript, interaktívny, online, web

Reference

MACKOVIČ, Jakub. *Application for Online Music Notation Typesetting.* Brno, 2016. Bachelor's thesis. Brno University of Technology, Faculty of Information Technology. Supervisor Burget Radek.

Application for Online Music Notation Typesetting

Declaration

Hereby I declare that this thesis was independently developed under the supervision of Ing. Radek Burget, Ph.D., and that it is my original work. All referential literature was specified and properly cited.

. . . .
Jakub Mackovič
May 15, 2016

Acknowledgements

I would like to thank my parents for helping me develop my interest in music and my supervisor for his technical advice.

Contents

Chapter 1

Introduction

Musical notation is a way to store musical pieces in a readable form using combinations of various symbols and text. It can be passed to other people which, with enough knowledge of musical notation, are then able to reproduce it. In this paper, we briefly explain the history of musical notation, its current standards, the nomenclature and meaning of different symbols. Existing computer notation programs are examined and the creation of a new one is proposed. The description of its implementation follows, including the choice of used technologies and its various features. We will have a look at the application's user interface, the work flow it allows and the results of testing performed by selected users. Finally, conclusions are drawn based on the testing and possible future additions to the application's functionality are discussed.

The purpose of this paper is not to teach its reader musical theory. It is expected that the reader has basic knowledge about it; they should at least understand the connection between the positions of notes in a composition and the way they are played with an instrument. If unsure about the notation terminology in English language, the reader is advised to first read chapter 3 where the notation's various terms are explained.

Chapter 2

History of musical notation

Every society that has ever existed has developed music in some form regardless of the society's level of isolation from other societies or their geographical location. This fact is supported by many archaeological findings, such as a fraction of a bone flute whose age has been estimated to be about 44 000 years old[10].

As the human cultures started to become more advanced, their music also started to grow in complexity. It became more difficult to learn and pass songs from person to person by simply listening to their performances and any attempt at it could result in a loss of detail. Therefore, first means of notating music were developed.

An important precondition for musical notation is a society's development of its writing system. Although the Egyptians, the Hebrews and Chaldeans and other Semitic nations had arrived to a certain level of a writing system and musical culture, it is unknown whether they noted their music. The Persians used a system of numbers placed on a stave of nine lines, while the Chinese used specific symbols. The ancient Greeks are the only ancient European nation that made use of letters of the alphabet to differentiate between sounds of different pitch. Their system of notation gave foundations to any successor European notation system, up to our present system[13].

In the beginning of the sixth century A.D., the Latin letters began to be used to notate different degrees of a scale. Around the tenth century, with the evolution of polyphonic music, where multiple sounds are played simultaneously of various lengths and pitches, it was necessary to find a system that would precisely indicate the pitch and the length of notes to be sung. Thus, throughout the next few centuries, heavily influenced by Christian theologians[6], a satisfactory system was founded and the notation similar to the one used today was completed[13].

2.1 Score arrangement

The advance in polyphonic and choral music in the fifteenth century lead to the desire of arranging music for multiple voices or instruments. By the term score arrangement we understand the scheme of notating music where multiple voices are drawn underneath each other and simultaneously played notes are vertically aligned. This held true for polyphonic music, although there still were differences between this type of music and monophonic music, an arrangement for a single voice, as there were virtually no constraints for any alignment in it. The influence of classical music in the following centuries caused the polyphonic notation to be more prevalent and to be used in solo voice notation as well[6].

Prior to the invention of the printing press in the fifteenth century, scores must have been written by composers by hand on sheets of paper with pre-drawn staff lines. The invention made it easier to notate compositions, although early arrangers were facing the problem of proper alignment of text, notes and staff lines. An innovator in score printing was the sixteenth century's printer Ottavio Petrucci who invented a technique to print staves, notes and text separately in multiple steps. A few other techniques were developed where the whole score was engraved onto a piece of metal which was then transferred to paper[9].

These approaches had been in use until the invention of computers and scorewriters in the second half of the twentieth century. A scorewriter is a piece of software that allows users to create and notate musical compositions. Its set of features usually includes error correction such as changing a note's pitch, length or simply undoing an erroneous action, audio preview of the arrangement and printing the piece or exporting it to various serialized and graphical formats.

These programs can be divided into two groups based on the interaction with their users – WYSIWYG editors and text-based typesetters. This classification is closely described in chapter 4.

Chapter 3

Terminology and notation practices

The key to correct communication between two sides is common understanding. Because the reader of this text may not possess excellent knowledge in musical theory and the nomenclature of its various elements and aspects, this chapter will briefly explain those that are necessary in order to understand future chapters, pictured in today's notation standards.

3.1 Rhythmic symbols and the staff

3.1.1 Notes

The most important elements of musical notations are the notes which represent sounds that are supposed to be played. A note is music's elementary rhythmic element. It is associated with a certain length which means how long in time should it be played and a certain pitch that specifies how high should it be played. Among singers, men's voices are usually thicker then women's. Therefore, compositions for men often consist of lower notes and those for women of higher notes.

The length of a note is determined by its shape. A note typically consists of an oval head and optionally of a vertical line coming out of its head called the *stem* and of wings coming out of the end of the stem. The note of the reference length is called the *whole* note. The lengths of other notes refer to the length of this note. For example, the next shorter note after the whole note is the *half* note which is half as long as the whole note. The note of the next shorter length is twice as short as the previous one. The series would continue with a *quarter*, an *eighth*, a *sixteenth*, a *thirty-second* and so on. Going the other way, we can derive a *double* or a *quadruple* note. Notes of even longer durations do exist, although they are not commonly used in practice.

Notes are placed on, between, above or below a sequence of five equidistant horizontal lines called the *staff* (plural *staves*). A note's placement determines its pitch – a higher note is placed higher (vertically; relative to the middle staff line) than a lower note. Figure 3.1 shows six notes placed on a staff. The first one from the left side is the whole note and each next note is played for a half of the duration of the previous one. The last one is the *thirty-second*.

It is also possible to examine the pitches of these notes. The leftmost one has the lowest pitch of them and the difference of pitches of a note and the note after it is equal to one except for the last two notes where it is equal to two. We can also see that if a note's pitch is greater or equal to the pitch given by the middle staff line, the note's stem

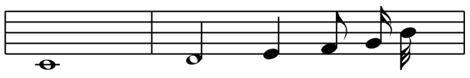


Figure 3.1: A series of notes with varying qualities.

grows downwards and any wings that it contains will be turned upside down but they will maintain their direction of growth to the right.

Let us also notice the changing distances between notes. A general rule states that the longer the duration of a note is, the larger space is left after it. The relation between a note's length and the spacing should not be linear in order to achieve a high level of a composition's visual quality. For example, there should be less then twice as much space after a whole note than after a half note[12]. This is observable in figure 3.1.

The height of a note head is very closely related to the amount of space between the staff's lines. If a note is placed in between two lines, its head should exactly fit there with its height and if it is placed on a line then the line will go through the head's centre. We can see this in figure 3.1 where the third note is placed on the fifth line and the fourth note is placed in between the fourth and the fifth line.

It is also possible to see that the first note of the series in figure 3.1 is located below the staff and it is not touching any of the staff's lines. Therefore, additional lines are needed to notate the note's pitch. These lines are called *ledgers* and they extend the staff lines with the same spacing as at the staff lines' spacing and the same thickness. Ledger lines only slightly extend past the note head, sufficiently enough to be seen[12]. An example can be seen in figure 3.2 where four ledger lines are added to reach the first and the second whole note.

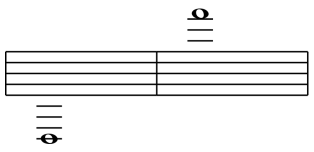


Figure 3.2: Ledger lines are drawn to extend the staff lines.

3.1.2 Rests

The other essential rhythmic element is a *rest* or a *break*. It specifies that at a particular point in a composition, nothing will be played by an instrument for a specific amount of time. Various shapes of rests determine their length and the set of rules about length of notes as described in 3.1.1 applies to the rests as well. Rests are also placed on staves and, because they have no pitch, their vertical placement does not influence any of their qualities.

Rests are typically placed on the staff around the middle line. Figure 3.3 shows six rests placed on the staff with their lengths equivalent to the notes in figure 3.1.

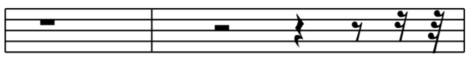


Figure 3.3: An array of rests of decreasing length.

3.2 Measures

The rhythmic symbols in a staff are divided into sections called *measures*. Each measure is associated with a certain total length of rhythmic symbols that can fit into it and the total length must be satisfied in order for the measure to be valid. The way the total length is determined is described in section 3.4.3.

3.2.1 Measure separation

Once a composer fills a measure with notes and rests and wants the melody to continue, they need to put the new notes into a new measure. The new measure is separated from the previous measure using a vertical line called the *bar line*. The bar line is as high as the staff is and it is at most as thick as the staff's lines are[12]. Figure 3.4 shows an example of dividing a simple melody into measures.

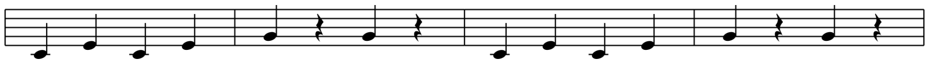


Figure 3.4: A simple melody with its notes distributed over four measures.

Measures can be added next to each other while there is free space in a line in which to put them. Once the measures form a full line, the composition may continue with the next measure on the next line. Figure 3.5 displays a continuation of the melody from figure 3.4 that needed to be continued on a new line.

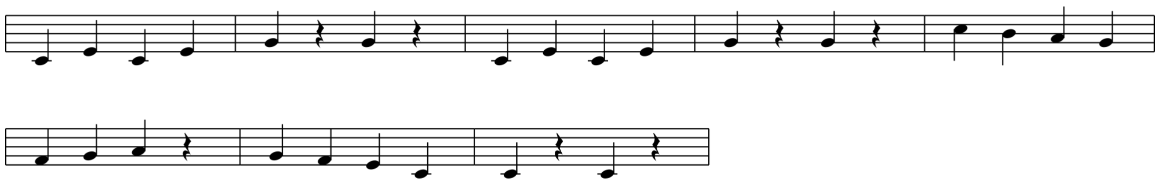


Figure 3.5: A simple melody spanning two lines.

3.2.2 Measure alignment

When a composition incorporates multiple instruments, the measures of these instruments need to be vertically aligned in order to make their synchronization possible and to make concurrently played notes be easier to detect.

Thus, across all instruments throughout the whole composition, the beginnings of measures and the notes that are played at the same time must be vertically aligned. Moreover, the first measures of every instrument in a line should have their beginning bar lines joined into one, connecting all instruments.

Figure 3.6 displays two instruments playing their melodies at the same time. It should be noted that left edges of their measures are aligned and that the second instrument's second, third and fifth note is pushed to the right to match its position with the respective note of the first instrument. On the left side, these instruments are joined by a common bar line.

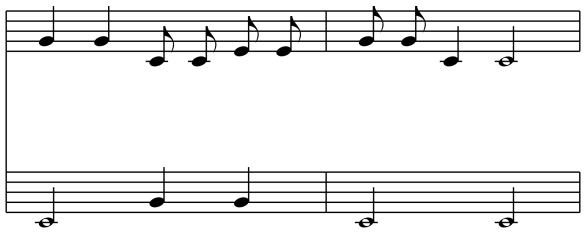


Figure 3.6: Two instruments having their notation aligned.

3.3 Accidentals

So far, we have been describing a note's pitch as being directly correlated with its vertical position on a staff. Although this may be true for the majority of instances, the real pitch of a produced physical tone may differ between two notes that are drawn identically on the staff.

To change a tone's pitch without having to alter the corresponding note's vertical position, one may add an *accidental* next to the note. The so-called *sharp* raises a tone by a semitone and a *double sharp* by a whole tone. A *flat* lowers the tone by a semitone and a *double flat* by a whole tone. Finally, a *natural* cancels any effect of a previous accidental on a particular pitch. These symbols are shown in figure 3.7 in the order of mentioning.

The double flat symbol acts as a single symbol in spite of being created by joining two flat symbols. An accidental stays in effect for all notes of the pitch to which it was applied and its effect lasts until the end of the measure.

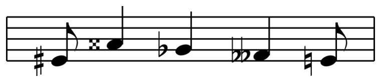


Figure 3.7: Accidentals attached to their note.

After looking at the first and the last note, we discover that even though both of them are placed on the fifth staff line, their real pitch is different, as the second note's natural accidental cancels the first one's sharp accidental.

3.4 Signatures

A musical *signature* is a symbol or a set of symbols that may be located at the beginning of a measure to alter the way a part of a composition should be reproduced. Their effect lasts until the end of the composition or until another signature of the same kind is defined.

If specifying multiple kinds of signatures at the same time, the following order must be obeyed: a clef, a key signature and a time signature.

3.4.1 Clefs

A *clef* sets the pitch of a reference note so that it is possible to tell which tone a note corresponds to. Let us have a look at three basic kinds of clefs and which reference notes they set in their base position.

The *treble* clef, also called the G clef, specifies the position of the tone G^4, the *alto* clef specifies the position of the tone C^4 and finally, the *bass* clef specifies where the F^3 tone is located.

Figure 3.8 shows an arrangement of three instruments where each of them uses a different clef for notation. From top to bottom, a treble, alto and a bass clef is used. If this arrangement was reproduced, every instrument would play the notes G^4, C^4 and F^3.

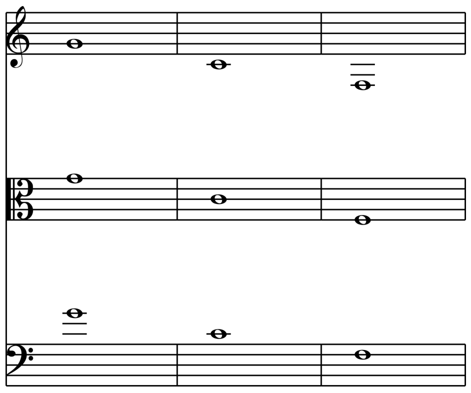


Figure 3.8: Different clefs simultaneously notating tones of equal pitch.

To maintain a visually appealing composition, certain dimensions of the clefs must be met. The treble clef's bottom curl must touch the fifth staff line and the preceding staff line must go through the curl's centre. Its height should slightly exceed the height of the staff. The alto clef's height must be equal to the height of the staff. The bass clef's height should be approximately equal to three spaces of the staff lines and its two dots must surround the staff line that is specified by this clef to be the F note.

The main reason to use different clefs is to reduce the amount of ledger lines per note and to increase the readability of the notation. At least one clef per instrument in the whole composition is required, otherwise the arrangement would not be reproducible. Generally, clefs are drawn at the beginning of each line as a reminder for the players.

3.4.2 Key signature

Using the same symbols as for accidentals (3.3), a *key signature* changes the pitch of individual notes not only for one measure, but for every measure from the key signature's definition until the end of the composition or until an another key signature is specified. It consists of zero to seven sharps or flats and, if necessary, it is immediately preceded or followed by an amount of naturals. Those must be used if the key signature of the composition changes from one to an another.

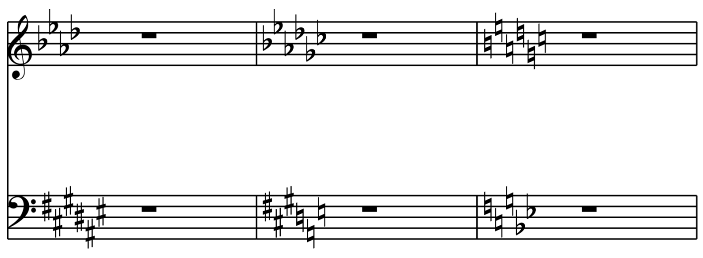


Figure 3.9: Changes of key signatures.

Let us examine some rules for key placement, referring to figure 3.9. If no key is previously specified and we would like to set one, we can simply place the accidentals on the staff right away, as can be seen in the first measure of both instruments. If a new key signature possesses more accidentals of the same kind than the previous one, they may also be placed on the staff immediately (see second measure of the first instrument). If it possesses fewer accidentals then those must be followed by naturals to cancel the remaining accidentals from the previous signature (second instrument, second measure). If the signature changes the type of accidentals, then the previous signature must be completely cancelled with naturals which precede the new signature (second instrument, third measure). Finally, if we would like to set the key with no accidentals explicitly, we would need to cancel the previous signature with naturals (first instrument, third measure)[2].

It is also a general habit to place the key signature of a composition on the beginning of each line as a reminder.

3.4.3 Time signature

A *time signature* (also simply called *time*) indicates the sum of the lengths of rhythmic symbols per measure. If used along with other types of signatures, it appears at the end of the measure's part with signatures. Again, its effect lasts until the end of the composition or until a different time signature is specified.

This signature is typically represented by two numbers written on the staff underneath each other. The first number fits with its size exactly between the first and the third staff line and tells us the amount of beats per measure, the second number fits between the third and the fifth staff line and tells us how long are the beats in terms of note lengths – the number 2 represents a half note, 4 represents a quarter and so on.

For instance, a 3/4 time constraints measures to contain rhythmic symbols with the sum of their lengths equal to the length of three quarter notes and is well known for its frequent usage in waltz.

For historical reasons, alternative means of time notation are possible. A symbol that resembles the capital letter C of the Latin alphabet can be used as a short-cut for the 4/4 time and that same letter with a vertical line striking the letter through its centre can be used instead of the 2/4 time[6]. These symbols are placed on the staff to fit between the second and the fourth staff line. The times are named the *common time* and the *cut common time*, respectively.

Examples of the mentioned time signatures may be found in figure 3.10.

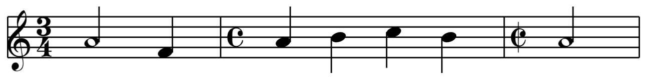


Figure 3.10: Changes of the time signature in a composition.

Chapter 4

State of the art

In this chapter, we will analyse some of the current means of musical notation. We will classify and describe these means and present examples of existing solutions.

Musical typesetting has gone through many centuries of innovations and changes. Today, it is still entirely possible to compose using only pen and paper and some may prefer this way. However, if one wants to publish arrangements of exceptional visual quality and precision, they need to exploit modern typesetting techniques – using computer software.

The way every typesetting program operates could be simply described in two steps:

1. Receive user input
2. Output the composition

What these programs have in common is the second step – they are able to display the final product to the user using a screen or to export it to a standardized graphical or a serialized format. Nonetheless, the process of user input may be vastly different. One program may receive user input through the use of a graphical user interface and a computer's peripherals, while the other one could accept one or more files which it would then process.

Therefore, based on how they obtain user input, modern typesetting programs can be divided into these following groups.

4.1 WYSIWYG editors

A WYSIWYG (what you see is what you get) program is a program whose user interface provides immediate visual feedback in the form of a product that precisely or almost exactly resembles the final product[11]. A classic example are word processors which display the documents the way they would look like if they were printed at the given moment.

This exact principle applies to the first group of musical notation editors. A user is presented with a visual representation of the composition in the form of a paper which can be modified with the use of an array of tools. These editors employ a graphical user interface and the computer's peripheral devices such as mouse or keyboard and, of course, the monitor as the primary method of user's interaction.

Most notable features of these applications are precision of symbol placement, immediate auditory feedback, being able to undo actions, MIDI keyboard input and the option to print or export the composition.

These programs commonly make use of a modal user interface – an interface where two same user actions may cause a different result.

4.1.1 MuseScore

MuseScore is an open source and free desktop scorewriter written in C++ and Qt. It places great emphasis on an easy and fast note entry, visual appearance even at high levels of scaling, modal interface, playback using software synthesizer and alternative input approach [3].

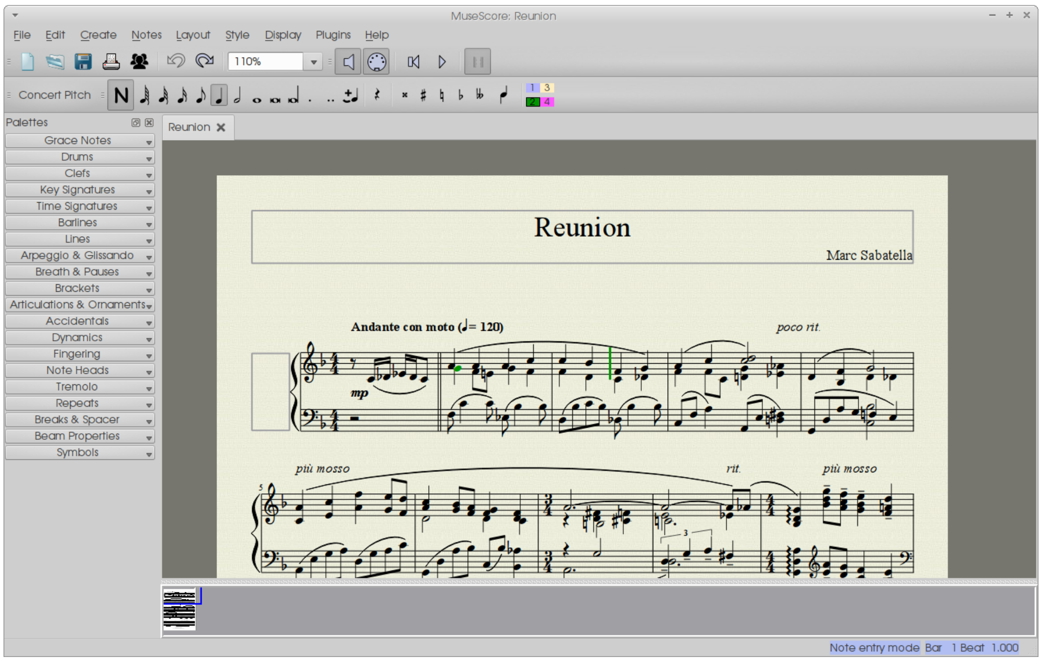


Figure 4.1: An overview of MuseScore's GUI (version 1.3).

As we can see on figure 4.1, the user is presented a virtual sheet of paper. On the left side there is a panel with tools for manipulation of different aspects of the composition. The panel just above the sheet provides a list of symbols that are assumed to be frequently used. In the bottom right corner the current mode is displayed along with playback information.

4.1.2 Flat

Another notable exhibit of this type of scorewriters is *Flat*[5]. It operates on the platform of modern web browsers that support scalable vector graphics and scripting using JavaScript.

Akin to MuseScore's user interface, a sheet of paper with the composition is presented, surrounded with a list of tools and frequently used symbols. Because Flat is a web-based application, it allows the user to publish their score on-line, letting other users view it or interact with it. Auditory feedback is also present with a palette of sounds for instruments or drums. A view of its interface may be seen on figure 4.2.

Flat also maintains an information system for storing its users' arrangements and offers a free version with limited capabilities and a paid version with a wide repertoire of functionalities.

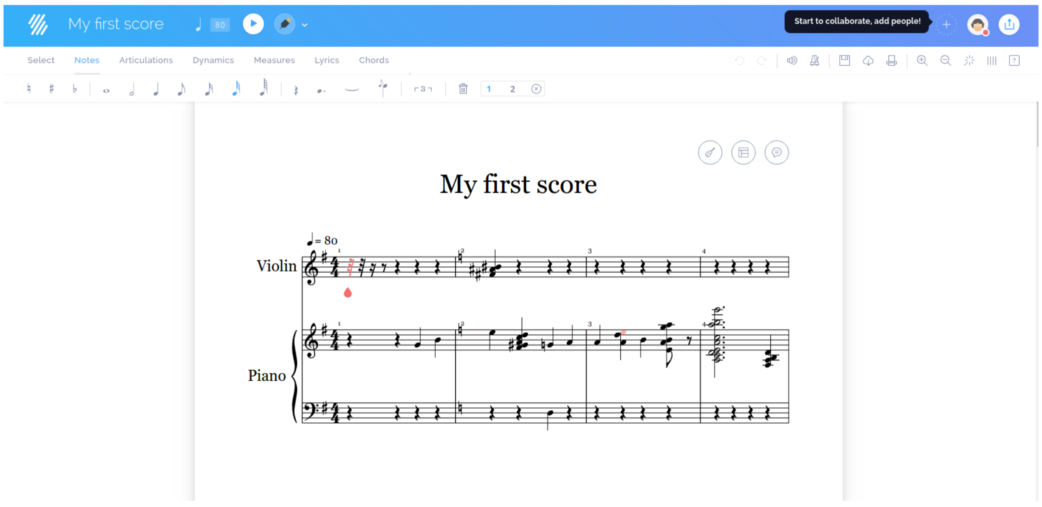


Figure 4.2: An overview of Flat's GUI.

4.2 Text-based typesetters

This category of typesetters allows the creation of a composition without the use of a graphical interface. Their only source of input are text files which contain a set of instructions to define the composition. The instructions are often similar to those of a programming language and the scorewriters must interpret or compile the files to produce a graphical representation of the score.

Their user interfaces may not be as user-friendly as those of the WYSIWYG editors, changes to the input files are not immediately reflected on the final product and the users have to learn a new language for the sole purpose of score creation. Nevertheless, they provide a platform independent way of musical notation and determine an optimal symbol placement. The latter results in a more pleasurable reading experience as the programs are are not constrained to an immediate visual feedback, allowing them to take some time for this action.

4.2.1 LilyPond

A noteworthy example of this class of scorewriters is *LilyPond*. Created with the goal to achieve the visual notation style that mimics the finest hand-engraved scores of the pre-computer age[7], it works as a command line tool accepting files of its own file format similar to the one of the LaTeX typesetting environment.

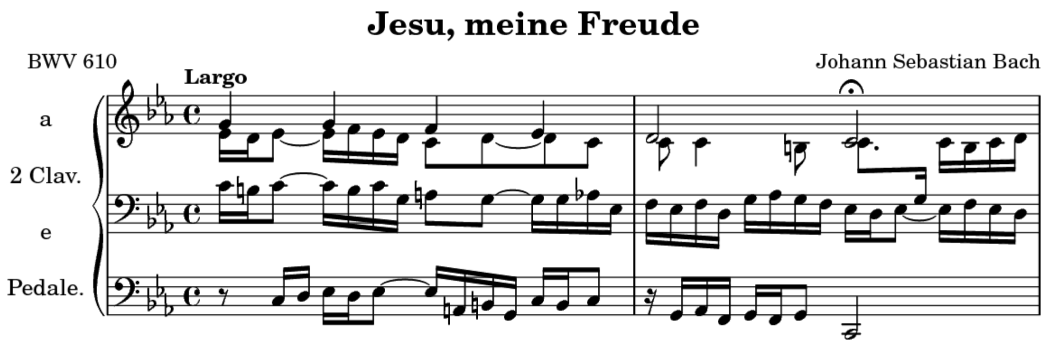


Figure 4.3: An example of a LilyPond score, taken from [1].

Chapter 5

Application design

The central task of this thesis is the creation of a musical typesetting application in the environment of web browsers.

Therefore, its specifications, goals, design and implementation shall be described in the following chapters.

5.1 Goals

As it has been mentioned, scorewriters allow various ways of user input to let users notate compositions to their needs. With the enormous amount of musical aspects, rules and symbols, it is nearly impossible for an individual to devise a program whose tool set would be comparable to those of the professional programs.

Therefore, a set of goals that would characterize this application was compiled. They are presented here in no particular order of significance.

5.1.1 Measure validation

The application must make sure that at any moment, every measure's sum of lengths of rhythmic symbols is equal to the total length given by the appropriate time signature. If a measure does not satisfy this constraint, the application must perform a series of steps that result in a valid measure.

5.1.2 Interface functionality

The user interface should behave in a uniform manner – if the user is used to how one tool works, they should expect other tools to function similarly to the first one – to raise the interface's degree of user-friendliness. It should provide an immediate visual feedback to user's action and indicate what the outcome of an action would be.

Moreover, the user should be able to easily correct their mistakes in the stages of simple symbol insertion and actions that could incorrectly modify many parts of the composition should either be prohibited or should prompt for the user's confirmation.

Finally, the interface should help the user create a composition more quickly compared to when that same composition would be drawn by hand.

5.1.3 Visual appeal

The application must provide a visually appealing score as its final graphical product. To achieve this, the ability to correctly align measures and notes must be reached (described in section 3.2.2) and the rule of non-uniform rhythmic symbol spacing must be satisfied (as described in figure 3.1).

5.1.4 Export

Users must be given the option to export their composition to a standardized file format or be able to print it.

5.2 User interface

The design of the application's user interface is based on the graphical interface of MuseScore (see section 4.1.1). It will consist of two main elements – a virtual sheet of paper and a list of tools called the *toolbox*.

The paper will take up most of the space. It will be slightly off-centred to the right to allow the toolbox's positioning. The toolbox will be located on the left side of the interface, starting in the top left corner. A tool of the toolbox will consist of a label stating which aspect of the composition will the tool be associated with. The actual actions this tool offers will be shown under the label after the user selects this tool (by clicking on the label, for instance). This will make this tool *active*. However, at most one tool will be in an active state. When the user selects another tool, the previously active tool will become *inactive* and will be hidden.

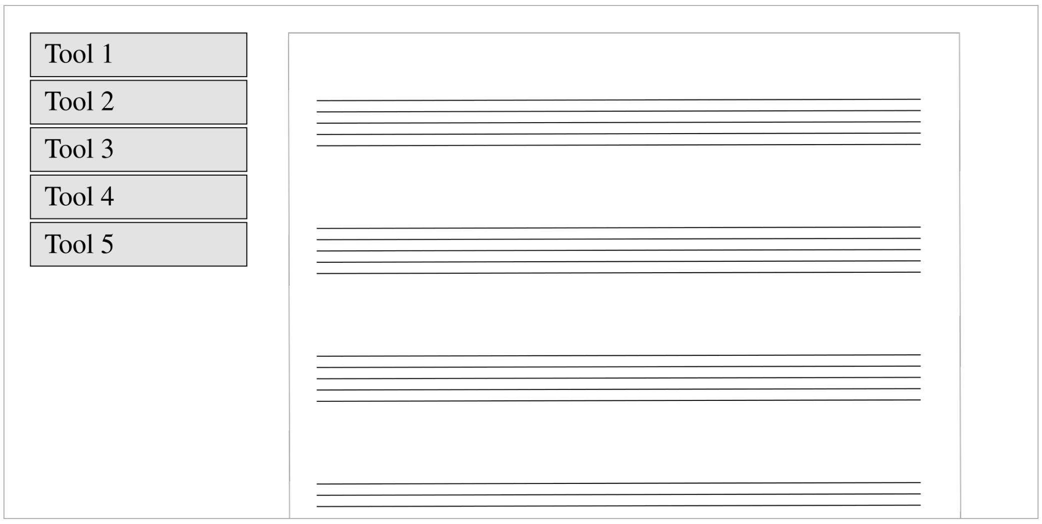


Figure 5.1: A sketch of the interface with a collapsed toolbox.

A sketch drawn in figure 5.1 displays the toolbox with no active tools. Next to it there is the sheet and its bottom part is cut off because it can not fit into the interface's viewport. Figure 5.2 follows and we can observe that the second tool has been selected and expanded to show the actions it offers.

Users will be able to navigate to the parts of the composition they can not see by scrolling up or down using the mouse. While the users are scrolling, the toolbox keeps its

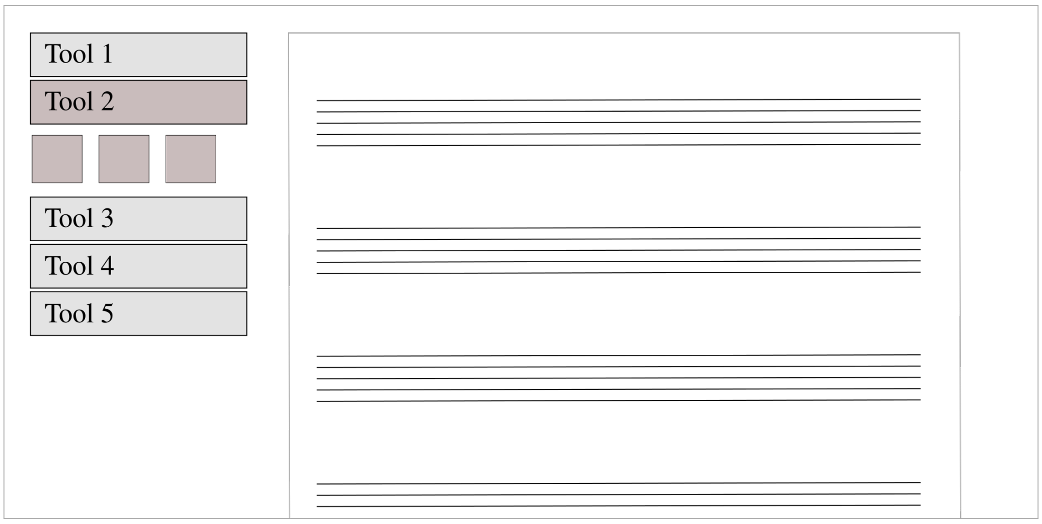


Figure 5.2: A sketch of the interface with an active tool.

position so that it is available at all times. New sheets are added below the last one.

Chapter 6

Implementation

I have implemented an application for online musical typesetting that follows the design and meets the goals of the previous chapter. In this chapter, we will have a closer look at its implementation details.

6.1 Choice of implementation languages

The choice of a programming language is a critical step in the early stages of software development. The target architecture must be considered and it limits the amount of languages to choose from. If there are more candidate languages, then they are, as a tie breaker, evaluated based on some criteria. The programmer may consider the speed of execution, the set of built-in functions, the availability at the end users or the programmer's familiarity with the languages. In this work, the choice of languages was conditioned by the target platform of web browsers.

6.1.1 Front and back end

HTML (Hypertext Markup Language) is the building rock for online documents. Based on SGML, it defines a structure and content for the documents using tags, is platform independent and the document's content may be accessed and modified using the DOM (Document Object Model) interface.

JavaScript is a prototype-based, dynamically typed interpreted scripting language that is run in the user's web browser and is used to create dynamic web pages exploiting the DOM interface to add, modify or remove parts of the HTML document. The core of this application is written in this language and it also handles user interaction.

CSS (Cascading Style Sheets) is a language that alters how HTML elements are displayed to the user. It is used in this application's user interface to make it look the way it was designed.

This scorewriter makes no use of an information system as it does not store its users' composition, although PHP is used on the server side to generate the HTML document that is sent to the users.

6.2 Composition representation

A composition can be logically divided into many units. The whole composition consists of pages, the pages contain lines of measures for every instrument, the measures are made up of notes, rests and so on, effectively creating a tree data structure. Let us have a look at how compositions are represented in this application.

6.2.1 The document

To represent and display musical symbols, SVG is used. SVG (Scalable Vector Graphics) is an application of XML (Extensible Markup Language) which is also based on SGML. It is a text-based vector graphic file format that uses tags similar to those of HTML to define graphical objects, creating a hierarchical structure. Thanks to this can SVG be directly embedded into the HTML tree and accessed the same way using the DOM interface.

A composition's page is represented by a root SVG tag. This tag contains tags for the representation of sheet lines. Those are represented by grouping G tags, containing a line element to connect the left edges of all instruments' first measure of that sheet line and a nested G tag. The nested G tag then contains tags for measures (from now on, assume that everything is represented by the G tag unless stated otherwise). A measure contains its staff lines, an ending bar line, its attributes and rhythmic symbols. The lines are represented by a line tag. The measure's attributes tag always contains a tag for a clef, a time and a key signature and those may be empty if no signatures are specified for that measure. Notes and rests are contained in the rhythmic symbols tag. If a rhythmic symbol is a note, then it nests its note heads (in case it is a chord), accidentals of those note heads, a stem line and ledger lines.

Finally, the notation symbols appear at the leaf level of nesting. Symbols are defined in a separate hidden SVG tree using a defs tag and each of the definitions has a unique identifier. SVG's use tags are used to display the symbols and their href attributes specify the identification of the symbol's definition. The browser then looks up the correct symbol definition. These use tags are always nested in an extra G tag to allow a symbol's easier positioning.

Although SVG is mostly supported by modern browsers, it is still an experimental technology and the browsers are not always unified on how to display these vector images. Therefore, this application has been developed on the Google Chrome browser, version 49, and it is guaranteed to work on this browser as intended.

6.2.2 Internal representation

The composition's internal representation is quite similar to the document's representation. There are objects for pages, instruments (which are not present in the document), sheet lines, measures, attributes of measures, rhythmic symbols. All of these objects hold a reference to their HTML document's equivalent.

However, this representation is different in its tree structure which it forms – it is a result of various rendering optimizations.

The whole composition is contained in a singleton object. This object has references to an array of instruments, an array of sheet pages and an array of sheet lines. An instrument contains all of its measures throughout the composition. A page consists of sheet lines. However, it does not refer to the sheet line objects directly. It only stores information about the position of the lines in the composition object's collection of sheet lines. The same applies to the sheet lines. A sheet line object contains a single collection of indexes of measures that appear in the line, regardless of how many instruments are there in the arrangement.

A measure consists of its attributes (signatures) and a collection of rhythmic symbols. If a measure sets or changes one of the signatures, it holds a reference to an attributes object which encodes the signatures.

There are two types of rhythmic symbols – notes and rests. A rhythmic symbol is bound to a certain musical length. We model notes as so called *note stacks*, as they may contain more than one note head (in the case of chords). Therefore, a note stack contains a collection of one or more note heads and a collection of accidentals that may appear at the note heads.

A note head is associated with a certain pitch. The pitch is closely related to the musical pitch (see section 3.1.1) and its value is used to determine the note head's position on the staff.

6.3 Rendering

The composition need to be, ultimately, drawn on the virtual sheet. With the exception of a sheet page, every part of the composition is associated with x and y coordinates of the SVG canvas. SVG allows us to position our symbols either by setting the x and y attributes of the elements or by setting their transform attributes[8]. In the application, the latter option is used.

The displaying and the drawing part of an element is handled by the browser as soon as the element's use tag is inserted in the document tree. The browser places the element in the canvas the way it is defined in the definitions and it could be of an arbitrary size. Thus, we need to scale the element up or down to the desired size using the transform attribute. Although the element can be moved and scaled at the same time using the attribute, the order of these operations must be taken in consideration as a different order of transformations will lead to different results. Therefore, we only allow one transformation function per element. The use element is for this reason first scaled applying the transform attribute's scale function and then wrapped in a parent g tag to which the translate function is applied.

The application's role in symbol rendering is to calculate the symbol's exact target coordinates and size as the correct placement of symbols is crucial for the score's readability.

6.3.1 Bounding box

In order to understand how the application places the symbols, we need to explain what an element's bounding box is and how does the application make use of it.

A bounding box (or a fitting box; figure 6.1) of an element is the tightest, smallest rectangle in which the element still fits. The browser can calculate this box and it provides us with information regarding the box's dimensions and its coordinates either in the whole document or in the browser's viewport.

With the use of bounding boxes, we can find an element's coordinates in the SVG coordinate system by subtracting the element's bounding box's top left corner coordinates from the SVG root element's top left corner coordinates.

6.3.2 Symbol placement

Once we know where a symbol should be placed and what its height should be, its placement is performed as described in the following steps.

Figure 6.1: The bounding box of a treble clef.

First, we take the symbol's use element and strip it off any previous transformation attributes (or create the element if it does not exist yet). It is then inserted in the document tree so that the browser draws it on the SVG canvas. We measure the drawn symbol's height using its bounding box and use the height to calculate the parameters of the transform attribute's scale function as the ratio between the desired and current height values.

Afterwards, this symbol is positioned to the target coordinates by setting the translate function parameters of the transform attribute in the wrapping g element. The process of calculating the values of the parameters is similar to the scaling process. The element is stripped off its transform attribute. We cannot rely on the browser to draw the element on both zero coordinates because the symbol's definition may be shifted to any direction. The translate values are then calculated as the difference between the desired coordinates and the coordinates of the bounding box of the symbol in its base position.

Centering

Let us keep in mind that the above mentioned technique causes the top left corner of the symbol's bounding box to be located on these coordinates.

In many instances, this behaviour is not desired. For example, we may want the symbol's centre or its bottom right corner to be placed on those coordinates. Thus, we define the *centering* parameter of a translation. It specifies, in which part of the symbol's bounding box will the target coordinates land. Two real numbers which indicate the fraction of the box's width and height are used.

For example, a centering of $(0.0, 0.0)$ causes the coordinates to be located in the box's top left corner and $(0.5, 0.5)$ to be in the box's centre. $(1.0, 0.5)$ would make the coordinates be in the middle of the box's right edge, $(0.5, 1.0)$ in the middle of its bottom edge and $(1.0, 1.0)$ in the bottom right corner. The influence of these centering values can be seen in figure 6.2.

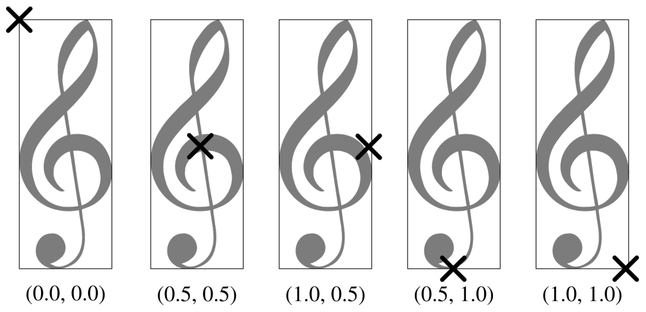


Figure 6.2: The connection between the centering parameter values and a symbol's position (X marks the coordinates; values in brackets indicate the x and y centering).

6.4 Composition assembling

Before the composition's symbols can be rendered, the application needs to assign them their final coordinates and centering (6.3.2). The next sections describe how the application decides where to put them and in what order to render them.

6.4.1 Rhythmic symbol alignment

The position of a rhythmic symbol in a measure depends on many factors, such as the measure's signatures, the amount of rhythmic symbols in the measure or the amount of instruments in the composition. Each measure is associated with its top left corner x and y coordinate and the value of the rhythmic symbol's x coordinate is relative to the measure's x coordinate. We will call this value an *x offset*.

As it has been discussed in section 3.2.2, concurrently played measures and notes must be vertically aligned for better readability. The reason there is an issue like this is caused by the non-linear dependence between a rhythmic symbol's length and the empty space after it (discussed in 3.1.1). We will call this empty space *space after*.

The process of alignment is described by algorithm 1.

The algorithm's input are two arrays containing rhythmic symbols of the concurrent measures. One is sorted by time (in the order as the rhythmics should be played) and the second one is sorted by their unaligned x offset in an ascending order.

The output of the algorithm is the second array in which the rhythmic symbols have their x offsets pushed back along the measure. The following statements will be true:

For any two rhythmics $r1$, $r2$ where $r1$ is played at the same time as $r2$, the x offset of $r1$ is equal to the x offset of $r2$. For any two rhythmics $r1$, $r2$ where $r2$ is played after $r1$, the x offset of $r2$ is greater than the x offset of $r1$.

As a result, the rhythmics in both arrays will be sorted in the same order.

First, the algorithm picks the earliest rhythmics in the time array (multiple are picked if more rhythmics are played at the same time) and stores them in a set called $CurrentTime$. The same is done for the x offset array, rhythmics are stored in the $CurrentX$ set. Then, in the set $Found$, there will be those rhythmics that appear both in $CurrentTime$ and $CurrentX$. $Missing$ will contain those rhythmic symbols that appear in in the $CurrentTime$ set but don't in the $CurrentX$ set and $Needless$ will contain those rhythmics that do appear in the $CurrentX$ and do not appear in the $CurrentTime$ set.

The $Missing$ set represents those rhythmics that should be played simultaneously with the rhythmics of the $Found$ set but have yet not been found in $AllX$. If there are any missing, the algorithm attempts to find them by picking a next symbol from the $AllX$ array one by one until there are no missing rhythmics. The $Needless$ set represents those rhythmics that were examined but are not supposed to be played along with the rhythmics in $Found$.

After finding all missing symbols, the symbols in the $Found$ set are aligned with the symbol whose x offset is the largest. Any symbol that increases its x offset causes all other symbols played after it to have their x offset increased by the same value (they are pushed back). Every symbol from the $Needless$ set is also pushed back behind the furthest $Found$ symbol based on their starting time difference.

Finally, the x offset array must be sorted because of the offset changes and the process repeats until both arrays are equally sorted.

Algorithm 1 Measure alignment

Input

```
1: AllTime  ▷ An array of rhythmic symbols of concurrent measures, sorted by time.
2: AllX  ▷ An array of rhythmic symbols of concurrent measures, sorted by base X offset.
```

Steps

```
1: procedure AlignMeasures
2:     $i \leftarrow 0$
3:     while $i < AllX.length$ do
4:         $CurrentTime \leftarrow AllTime.pickNextRhythmics(i)$
5:         $CurrentX \leftarrow AllX.pickNextRhythmics(i)$
6:         $Found \leftarrow CurrentTime \cap CurrentX$
7:         $Missing \leftarrow CurrentTime - CurrentX$
8:         $Needless \leftarrow CurrentX - CurrentTime$
9:         $j \leftarrow i + Found.length$
10:        while $Missing.length > 0$ do
11:            $rhythmic \leftarrow AllX[j]$
12:            if $Missing.contains(rhythmic)$ then
13:                $Missing.remove(rhythmic)$
14:                $Found.add(rhythmic)$
15:            else
16:                $Needless.add(rhythmic)$
17:            end if
18:            $j \leftarrow j + 1$
19:        end while
20:        $Found.alignX()$
21:        $Needless.pushX()$
22:        $AllX.sort()$
23:        $i \leftarrow i + CurrentTime.length$
24:    end while
25: end procedure
```

6.4.2 Fitting measures to lines and lines to pages

Once it is determined what size the gaps between notes and rests in measures are, can those measures be distributed into sheet lines. A sheet line is represented by a series of concurrently played measures and during the process of assignment of measures to sheet lines we make use of the fact that concurrent measures have, after their alignment, the same width. Thus, it is safe to consider only the widths of the first instrument's measures.

The measures are assigned to a line one by one until they exceed the line's maximum width. These measures then need to be shrinked (or stretched in case the last assigned measure is too wide) to fit the line's width. Certain thresholds of ratios between the line's width and the sum of measures' widths are in use to determine whether it is safe to scale the measures. A measure that is shrinked too much could have collisions in rhythmic symbols and would be less readable; a measure that is stretched too much could also be unappealing to the reader's eye. The latter might occur in the last line of the composition where there could be only one measure left and therefore it is preferred not to stretch it.

Each measure is then stretched based on the calculated ratio. However, the fixed parts of the measures are not stretched, it only applies to the gaps after rhythmic symbols. The factor of scaling the gaps of rhythmic symbols is computed as the equation 6.2 shows.

$$W_f + W_n * c = W_t \tag{6.1}$$

$$c = \frac{W_t - W_f}{W_n} \tag{6.2}$$

In the equation, W_f represents the width of a measure's fixed parts such as the width of its signatures or any intentional blank space, W_n is the width of gaps after the measure's rhythmic symbols, W_t is the target width to which the measure needs to fit and c represents the value by which the gaps after the rhythmics must be multiplied in order to make measure's width be equal to W_t.

Afterwards, it is determined which sheet lines will fit into a page based on their height. The height of a sheet line is calculated as a sum of the heights of measures of all instruments in that line. Various spaces between the measures are also taken into account. The lines are then assigned one by one to a page until the sum of their heights exceeds the available height in the page. Unlike in the measure assignment, sheet lines are not stretched to fit the page and the sheet line that causes the excess height is assigned to the next page. However, in the special case when the first line of the page is too high for the page, it does shrink in order to fit the page.

6.4.3 Signatures

If a measure is associated with any types of signatures, it is safe to render that signatures at this point. The rules about the order of the signatures (as mentioned in section 3.4) are satisfied.

The width of the signatures influences the x offset of the measure's rhythmic symbols. Because the rhythmics need to be aligned, they are pushed back in all other concurrent measures as if the signatures of their measure were as wide as the signatures of the measure with the widest signatures. This step may be executed before the first alignment phases as it is simpler to measure the signatures' rendered width than to perform various calculations.

6.4.4 Notes

Finally, the positions of notes' components (more precisely *note stacks*' components; defined in 6.2.2) need to be resolved. Their coordinates are dependent on the x offsets of the notes and the containing measures.

Stem and note heads

We define two arrangements of note heads in a note stack – *upstem* and *downstem*. An upstem arrangement occurs when the note head that is most distant from the middle staff line is located below it; a downstem arrangement occurs otherwise. The arrangement influences how is the note stack's stem drawn and on which side of the stem are the note heads placed. In the case of an upstem arrangement, the stem starts at the lowest note, grows upwards and note heads are placed on its left side, a downstem arrangement causes the stem to grow from the highest note downwards and the note heads are placed on its right side. It then grows past the note on the other end of the note stack and stops after the distance of 3.5 spaces between sheet lines (in musical theory, this is one octave above or below the last note). This may be observed in figure 6.3, where the note stacks' arrangements alternate between upstem and downstem, starting with an upstem.

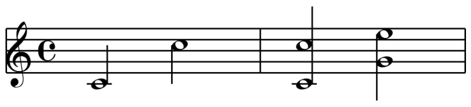


Figure 6.3: Downstem and upstem arrangements of notes.

If the stem does not reach the middle staff line, it is made longer to reach it. We can see this in figure 6.4 where notes are placed far above or below the staff and their stems still reach the middle staff line.

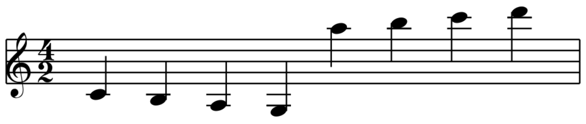


Figure 6.4: Stems of notes are lengthened to reach the middle staff line.

If there are notes with one pitch difference in a note stack then one of these notes is placed on the other side of the stem. The second note head (of the conflicting note heads) in the direction of stem growth is always chosen to be relocated (figure 6.5, measure 1). If there is a series of note heads with one pitch difference then the placement of the note heads alternates from side to side (figure 6.5, measure 2).

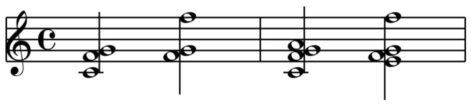


Figure 6.5: Note head collision handling on different stem growth directions.

Position of accidentals

The placement of note heads' accidentals must be as precise as possible. Collisions of accidentals with note heads and other accidentals need to be avoided and the positions of multiple accidentals in a note stack should reflect the relative positions of the respective note heads.

An accidental is placed at a default position next to its note head. If there is a collision with an accidental or a note head, this accidental is moved to the left behind the colliding object. This adjustment continues until the accidental does not collide with any other object. The collisions are detected by comparing the objects' bounding boxes.

The order in which the accidentals are placed is as follows: First, the accidentals of the note heads on the right side of the stem are placed, then the accidentals of the note heads on the left side. Both sides are then placed in the stem growth direction.

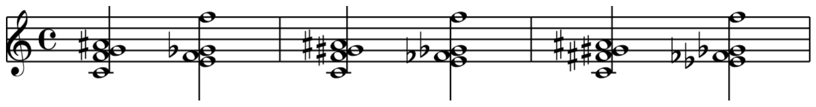


Figure 6.6: Placement of increasing amounts of accidentals.

In figure 6.6, there are three measures with two chords whose amount of note heads with accidentals increases. We can see how the accidentals adjust their position until the third measure where they mimic their respective note heads' constellation. For example, the note on the right side of the stem in the last chord with sharps has its sharp accidental on the rightmost position among the accidentals. This is also in effect in the last chord, except the leftmost note's flat accidental has the leftmost position among the accidentals.

Ledger lines and note wings

Ledger lines are drawn to reach any note heads above or below the staff lines. In the case of note heads of one pitch difference in a note stack, the respective ledger lines are extended as necessary to reach the note heads placed on the other side of the stem to help identify their pitch.

Note wings grow always from the end of the stem (if they are needed) always towards the notes and to the right side.

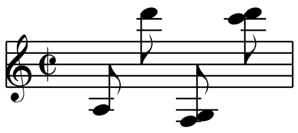


Figure 6.7: Length of ledger lines and orientation of stem wings.

6.5 User interface

The user interface was implemented as a single-page web site and is characteristic for its modality and consistent behaviour. It follows the proposal of the user interface from 5.2.

6.5.1 Web page layout

The web page consists of two components – a virtual sheet and a toolbox. The virtual sheet takes up about three quarters of the page's width and is centered a bit to the right to leave some space for the toolbox. In the beginning, a user is presented with a single sheet of paper containing a composition consisting of one instrument and an empty measure with a treble clef and the common time signature. The sheets expand downwards and the user is able to navigate around the composition by scrolling the mouse's wheel or by using the scrollbar to the right of the composition. An overview of the application after accessing the web page is shown in figure 6.8.

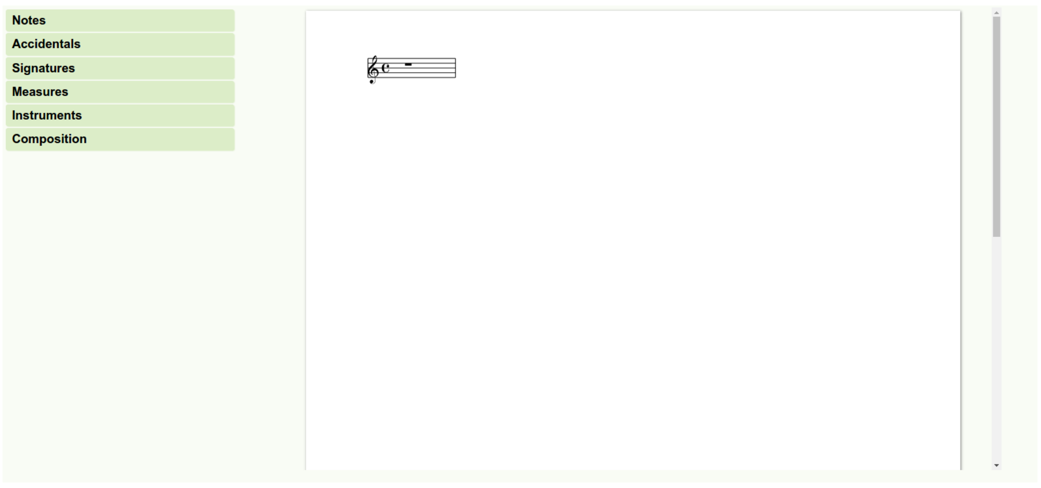


Figure 6.8: The application's user interface with a new composition.

The toolbox is bound to the page's left edge so that it is always at hand and it offers a variety of tools for the composition creation. A tool is by default collapsed to its label. Clicking on the tool's label expands the actions the tool offers and it collapses all other tools. Clicking on the label again causes the tool to collapse.

6.5.2 Modes

Modality allows the application to be associated with a *mode* and based on the mode to react to the same user input differently. Each tool in the toolbox is bound to a mode. By default, the application is in mode where it does not react to the user's interaction with the composition. After expanding a tool, the application changes its mode to the respective tool's mode.

The explanation of the tools and their modes in their order of appearance in the toolbox follows.

Note insert

Notes is the main tool for note insertion and manipulation. It offers a list of note lengths ordered by ascending length. By default, the quarter note is selected. The note lengths are represented by pictures of notes, drawn the way they would look like if placed onto the sheet. The selected item has a full opacity to distinguish it from the other items whose opacity is low but high enough to be seen. A note length can be selected simply by clicking on it.

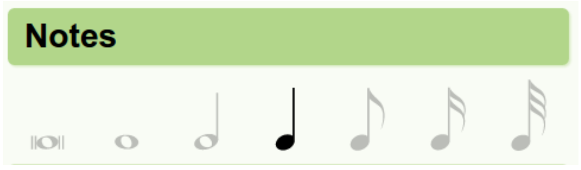


Figure 6.9: The notes tool, having a quarter note selected.

The user may then insert a note to the composition by clicking on the appropriate place in a measure on the sheet. The application then finds the closest rhythmic symbol. If it is a note stack then a note head will be added to it with a pitch determined by where the click occurred. For instance, clicking between the first and the second staff line will cause the note to be placed between those lines. The note head's musical length will be equal to the length of existing note heads in the stack. A note head is always added to the note stack unless there already is a note with that pitch.

If the closest rhythmic is a rest then the application attempts to substitute it with a note of the selected length. If the rest's length is too short then the length of consecutive rests beginning with that rest is considered. If the sum of those rests is still less than the selected length, no note is inserted. Otherwise the rests are removed, substituted with a new note stack with one note head and new rests may be added after it to fill the difference between the removed rests' length and the new note's length.

A note head can be removed in this mode by right-clicking while hovering over a note head. If the last note head is removed from a note stack, it is replaced by a rest or a series of rests to maintain the correct sum of lengths of rhythmics in that measure. Quick change in note length selection can be done by pressing the right and left arrow keys.

Accidental insert

The accidentals tool allows the user to add an accidental of any kind to a note head by left-clicking on that note head. Although the tool shows only three kinds of accidentals, as shown in figure 6.10, the double sharp and double flat accidentals can be added by clicking on the note head for the second time. A third click will remove the accidental. The accidental can be also removed by right-clicking on the accidental. To maintain a consistent behaviour across modes, right-clicking on a note head will remove the note head.

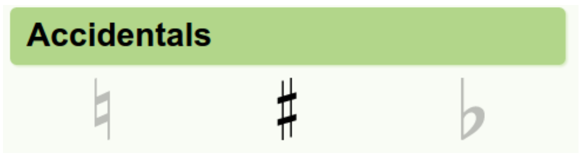


Figure 6.10: The accidentals tool expanded.

Signatures

This tool is used for inserting various signatures to measures. It contains three sections, each for one type of signatures. Each option in these sections contains an image that represents what the signature would look like if picked. Every section contains also a “none” option which is selected by default. When the user chooses their signatures, they may then click on a measure to which the combination of the three kinds of measures will be applied. A measure is highlighted when the mouse pointer hovers over it to indicate that, after clicking, this measure's attributes will change.

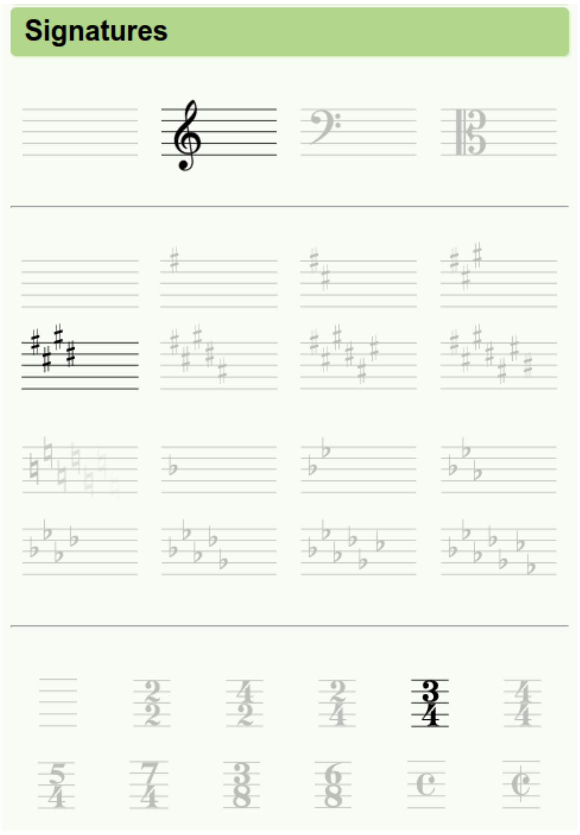


Figure 6.11: The signatures tool expanded.

In figure 6.11, the signatures tool is shown after the user selected a treble clef, a four-sharp key signature and a three-four time signature.

Measure manipulation

This simple tool serves two purposes: to add and remove measures. For that, two pictures, one consisting of a plus sign and the other of a minus sign, are used for these purposes respectively. By default, the add option is selected. After hovering with a mouse over a measure in the composition, the same picture appears at all concurrent measures because every instrument must contain the same amount of measures.

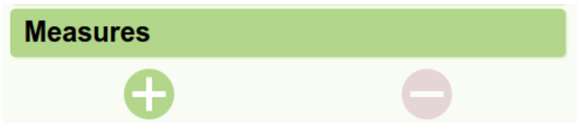


Figure 6.12: Caption

In the case of the adding action, plus signs appear between measures (with their bottom edges touching the uppermost staff line) where a new measure would be added after pressing a mouse button (shown in figure 6.13). If the mouse pointer is located in the left half of the measure, the sign appears above the measure's left bar line, otherwise it appears above its right bar line. In the case of deleting measures, a minus sign appears in the middle of the appropriate measures. The user does not have to click on the images for the action to be performed, clicking anywhere on the measure while the icons are displayed is sufficient.

The new measures contain only rests of the length given by the current time signature.

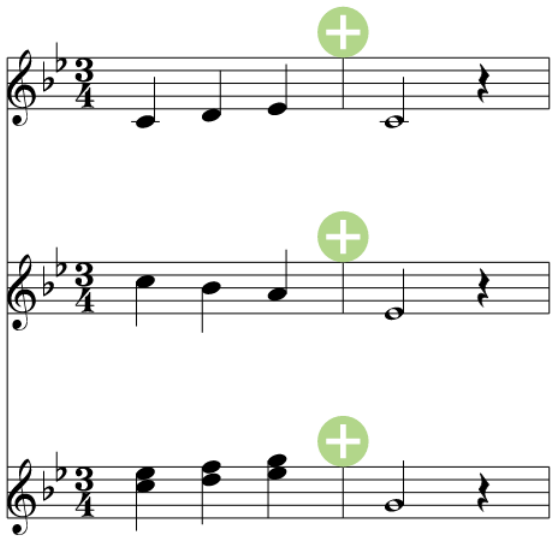
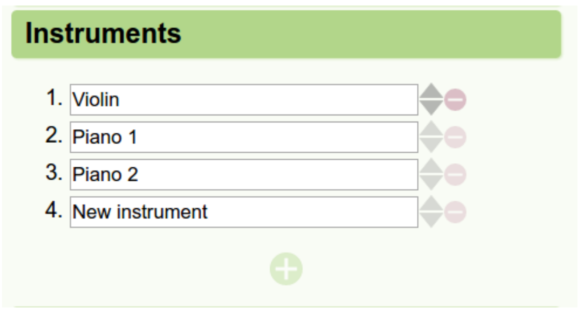


Figure 6.13: Add icons shown between appropriate measures.

Instrument manipulation

The instrument tool offers actions to add, remove or swap instruments. For adding and removing instruments, same icons are used as in the measures tool. The instruments are displayed in a list in the same order as they appear in the composition. Clicking on the triangle buttons causes the instrument to be exchanged with the instrument above or below it. The instruments may also be labelled, although their name will not appear in the composition.

A new instrument immediately contains as many measures as the other instruments do. These measures are filled with rests and none of these measures contain any signatures.



Instruments

1. Violin
2. Piano 1
3. Piano 2
4. New instrument

Figure 6.14: The instruments tool after adding fourth instrument.

Composition

Finally, the composition tool lets the user perform actions regarding the whole composition. The tool offers an option to export the composition to MusicXML format or to print the composition, displaying the browser's dialogue window for printing.

In order to print the sheet pages, CSS media queries are used to hide the unnecessary document elements.

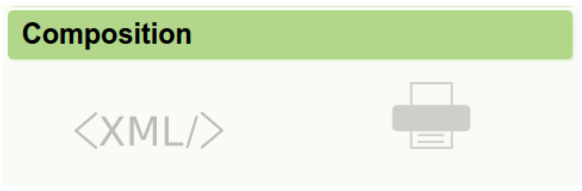


Figure 6.15: The composition tool with its exporting actions.

6.6 Features

In the application, various features are introduced to improve the user's comfort and speed of composition creation.

6.6.1 Sum of rhythmic symbols

The application makes sure that the sum of the musical lengths of every measure's rhythmic symbols satisfies the measure's current time signature. When the user adds or removes a note, new rests are inserted to the measure on appropriate positions to satisfy the sum. This way, the user does not need to manually calculate the amount of rests needed to fill the spaces between or around notes.

For this reason, users cannot explicitly insert rests into the composition the same way as they would insert notes. However, because rests substitute deleted notes, a rest of desired length can be inserted to the composition by removing a note of that length.

6.6.2 Changes of signatures

When the user changes signatures of measures, they usually want to change the signatures of the concurrent measures of all other instruments as well.

Therefore, when the user applies a time signature to a measure, that time signature is also applied to the simultaneous measures. Only the changing of the time signature causes this effect as the users may want different clefs or key signatures per instruments (although the latter is rare).

Inserting a time signature might change the total length of rhythmic symbols of the affected measures. The application makes sure that the total is satisfied either by adding rests to the end of those measures (if the signature changes to a longer one) or by removing rhythmic symbols from the end of the measures.

Moreover, setting a clef to a signature where a key signature is already present causes the key's accidentals to be moved up or down to match the reference pitch set by the type of the clef. The clef setting does not remove or alter the accidentals of the notes in the affected measures.

6.6.3 Default tool action selection

This feature causes actions of specific tools to be automatically selected when the tools are expanded. The feature applies to the signatures tool where the „no signature“ options are automatically selected and to the measures tool where the „add measure“ action is automatically selected. The reasoning behind this is the possibly destructive nature of those tools, as setting the wrong time signature accidentally while the user only wants to change a key signature might lead to note removal at the end of the affected measures.

6.6.4 Note insertion preview

When the user hovers their mouse over a measure in the notes tool, a grey note head is displayed on the position in the measure where a new note would be added if the user clicked at that moment. No grey note head is shown if clicking would not lead to an insertion. This can be seen in figure 6.16.

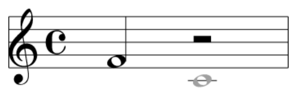


Figure 6.16: A grey stemless note head indicates where a note would be added after clicking.

6.6.5 Composition export

The application allows, in the *composition* tool, to export the composition in the serialized format of MusicXML, letting the users save their arrangement locally for later use.

MusicXML is an application of XML that with the use of a large variety of tags allows not only the logical representation of the composition (order and length of notes) but also the visual design representation (text, note spacing and many embellishments).

This type of files can be then imported to other notation software where the arrangement can be further edited. This has been successfully tested on an online notation application called NoteFlight[4].

Chapter 7

Testing

The application was, after being implemented, tested on users. As the application's target group of users consists of musicians and composers, people with at least basic musical background were chosen to test the application.

7.1 Tasks

The users were first presented the application, were explained its layout and then were let to find out how the tools work. After they became familiar with the interface they were given various tasks that emulated some of the use cases of the application. The tasks were simple at first (to add a chord or to create a sequence of notes) and their difficulty slowly graduated over time to the tasks that required them to operate on multiple instruments at the same time, to set a specific combination of signatures or to insert a series of notes and rests of varying lengths and pitches.

Finally, the users were asked to reproduce an existing composition with this application with the advice that if they are unable to recreate an aspect of it, they should approximate that aspect with what the application's tools allow.

7.2 Results

The users did not have any major issues understanding how to use the interface, although some of them attempted to drag the items from the toolbox onto the sheet multiple times before finding out that it has no effect.

When asked to remove a note, the users were unable to find an „action“ in the tools to remove a note. However, after giving them a hint that they can only manipulate the notes using their mouse, they figured out how to accomplish the removal within a few seconds.

The signatures tool's design was found out to be ineffective. Most of the time, the users wanted to set only one type of signatures and when they wanted to apply it to a measure with more kinds of signatures, they had to select the same signatures in the tool to avoid their removal in the measure. This fact inspired the creation of the feature described in section 6.6.3, however a different approach for the signature manipulation would be more suitable.

Most users pointed out the usefulness of the preview note by themselves (section 6.6.4) and the users with experience in composing appreciated the features about the automatic rests and signatures insertion (sections 6.6.1 and 6.6.2).

The experienced users performed very well on the last task. For example, one of them was asked to transcribe first few measures from a minuet by Johann Sebastian Bach. The first two lines of their composition are shown in figure 7.1 and the original can be seen in figure 7.2.

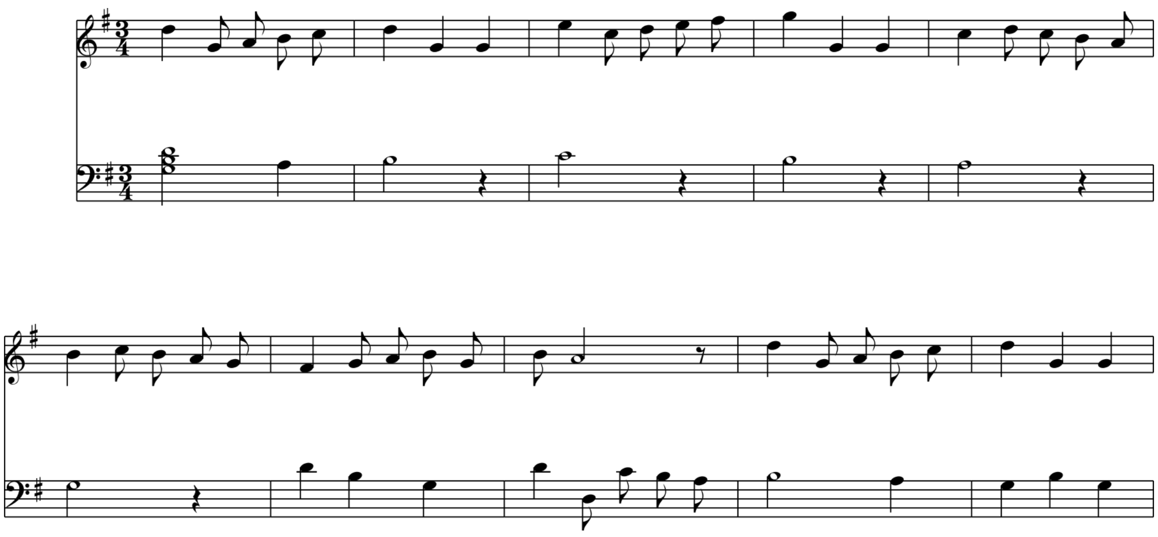


Figure 7.1: A user's transcription of a composition.

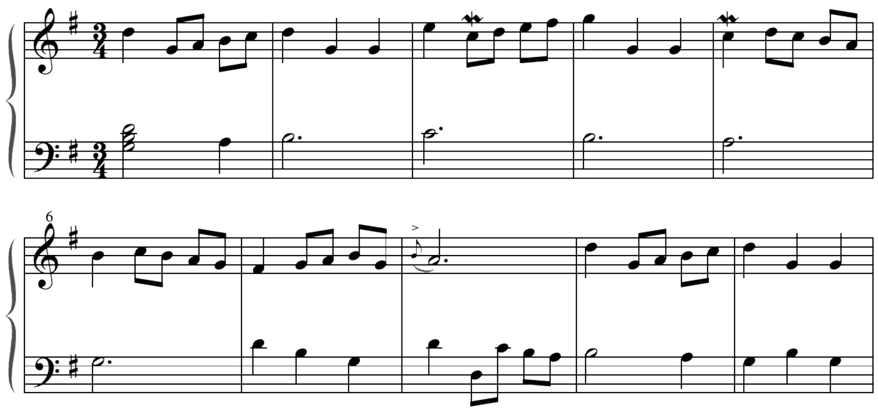


Figure 7.2: The original composition.

Another one requested that they arrange an intro to a pop music song. They successfully created it from memory and the first measures of the composition can be seen in figure 7.3.

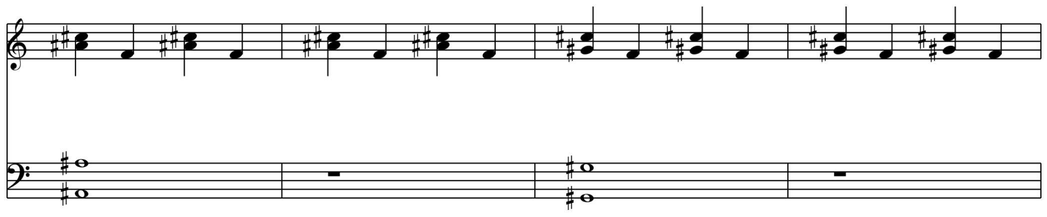


Figure 7.3: A user's arrangement of a pop song's intro.

Chapter 8

Conclusion

In this thesis, an application for online musical typesetting was proposed, designed and implemented. Several goals regarding its functionality were set and they were successfully met. A web-based user interface with a unique approach was created and tested on users. A set of features was introduced for the application which proved to be useful during the testing. The testing also provided feedback from the users and based on the feedback some aspects of the application and its interface were able to be improved.

This work's author gained a fair amount of experience not only in the fields of client side browser scripting and end user interaction, but also in the areas of musical theory, history and typesetting.

8.1 Future development

The thesis gave solid foundations for the application's further improvement. Auditory feedback, notation of lyrics and connecting notes of short duration with beam lines (thick lines connecting ends of the affected notes' stems) were the features suggested by most of the testing users.

In comparison to the existing programs, this application lacks the ability to notate tones of arbitrary length and various embellishments regarding the notes. However, turning this application into one with a tool set comparable to the existing ones is being seriously considered.

Bibliography

[1] Lilypond example scores. URL `http://lilypond.org/examples.html`. [Online; accessed 2016-05-05].

[2] *Standard music notation practice.* Music Publishers' Association of the United States, 1966, 1993.

[3] Musescore, the free music composition and notation software, 2008 - 2016. URL `https://musescore.org/`. [Online; accessed 2016-05-04].

[4] Noteflight llc, 2008 - 2016. URL `https://www.noteflight.com/`. [Online; accessed 2016-05-12].

[5] Flat, 2014 - 2016. URL `https://flat.io/`. [Online; accessed 2016-05-04].

[6] Willi Apel. *The Notation of Polyphonic Music, 900-1600.* Medieval Academy of America, 5 edition, 1961. ISBN 0910956154.

[7] The LilyPond development team. *Essay on automated music engraving*, 2002 - 2012. URL `http://lilypond.org/doc/v2.18/Documentation/essay.pdf`. [Online; accessed 2016-05-05].

[8] J. David Eisenberg. *SVG Essentials.* O'Reilly Media, 2002. ISBN 9780596002237.

[9] Alec Hyatt King. *Four Hundred Years of Music Printing.* British Library, 1979. ISBN 0904654338.

[10] Steven Brown Nils L. Wallin, Björn Merker. *The origins of music.* The MIT Press, new edition edition, 2001. ISBN 9780262731430.

[11] Eric S. Raymond. *The New Hacker's Dictionary.* The MIT Press, 3 edition, 1996. ISBN 9780262680929.

[12] Linda Lusk Tom Gerou. *Essential Dictionary of Music Notation: The Most Practical and Concise Source for Music Notation.* The Essential Dictionary Series. Alfred Publishing Co., Inc., 1996. ISBN 0882847309.

[13] C. F. Abdy Williams. *The story of notation.* London, The Walter Scott publishing co., ltd.; New York, C. Scribner's sons, 1903.