1 Introduction

Audovia is a database application for making music on your laptop or PC. Songs can have up to fifteen instrumental voices and a percussion track. Instruments can be chosen from either the default soundbank of 128 instruments or other soundbanks of your choice.

Songs can be developed, tested and edited very quickly and easily by virtue of the database structure and the ***JFugue*** MusicString notation. Notes within a MusicString are specified by their name and octave or by their MIDI value and their durations are specified either by character code, or numerically. You can use notes from C0 to G10, corresponding to MIDI values 0 to 127. Middle C is C5. Notes can be entered manually or by picking from graphic Treble, Alto, Tenor and Bass staves within the MusicString editor.

For example, the opening phrase of Joy to the World can be written as:

```
C6h B5q. A5i G5h. F5q E5h D5h C5h.
```
(note durations will be explained later)

A MusicString consists of one or more tokens separated by spaces, as above. Sequences of MusicStrings can be assembled into Patterns, which can be nested to any level. Songs can be constructed from Patterns either timewise or by voice.

The *File/Template* menu item creates song templates by voice where each voice is a Pattern containing other Patterns and MusicStrings. Bars (or measures) can be MusicStrings or Patterns. Pattern bars can be used to contain sequences of MusicStrings and/or Patterns.

All MusicStrings and Patterns within a song are given unique names which makes it easy to keep track of them within a composition. A MusicString can be shared between any of the Patterns in a song and it only needs to be edited once for the changes to be effective wherever that MusicString occurs within the song.

Audovia will play back your music and also export to MIDI and WAV files. The MIDI files can be opened in ***LMMS*** for music processing and in ***MuseScore*** for music publishing. The WAV files can be opened in ***Audacity***, then exported to MP3.

1.1 Background

Audovia is written in Java with a Swing forms user interface. It uses Apache Derby as the default database with an embedded JDBC driver. The music is generated by ***JFugue***, a Java API for music programming.

Audovia can be used to produce backing tracks for playing or singing along to. It is also useful for creating background music for videos or ringtones for mobile phones.

The ***Audovia*** program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

The source code can be found at ***GitHub***.

2 Installation on Linux

To install and run ***Audovia*** on a Linux ***snapd*** enabled system:

sudo snap install audovia-lite
audovia-lite

A desktop shortcut can be found at:

/snap/audovia-lite/current/Audovia Lite.desktop

For a quick start you can use *File/Song Import* and open the *Demo* folder, then select a song and *Import Song*. Then, from Tree View select "Song", press Play, then Default Soundbank.

Audovia stores its files in directories under:

~/Documents/Audovia/

2.1 Soundbanks

Audovia uses the Gervill synthesizer to generate sound. Gervill can use the default soundbank, based on TimGM6mb.sf2 by Tim Brechbill, or any soundbank with a *.sf2* or *.dls* extension. These can be found by searching for SoundFont files on the Internet.

Soundbanks can be kept on file, in the SF2 directory, or you can use *Soundbanks/Manage Soundbanks* to upload these to the database.

2.2 Backing up your Songs

Use *File/Song Export* to export a song to an XML file or *File/Song Import* to import a song from an XML file. The entire song structure is exported or imported with the exception of any soundbank reference.

3 Tutorial

Audovia opens with the ***Songs*** form for managing your songs. If you are using numeric note durations within a song, use the **numeric durations** column to specify whether these are expressed as decimal values of a whole note or as pulses, where 32 pulses represents a quarter note.

To ensure that the different voices in a song are synchronized, all note durations should equate to a whole number of pulses.

Numeric durations should be used for triplets. A quarter note divided into three triplets should be given durations of 11, 11 and 10 pulses, or 12, 10 and 10 pulses if you want to give more emphasis to the first triplet.

To edit a cell, either select the cell and type something into it or double-click on the cell. To copy data in a cell, highlight the text to be copied and press Ctrl-C. To paste, press Ctrl-V. This only works if the cell is in edit mode (yellow).

The **Strings** button opens a form for defining the ***JFugue*** MusicStrings in a selected song and the **Patterns** button opens a form for declaring the Patterns in the song.

The naming of MusicStrings is independent from the naming of Patterns. You may find it helpful to name MusicStrings in lower case and Patterns with an initial capital.

Use *File/Template* to give you a start when creating new songs.

Use the **Tree View** button to display the structure of Patterns and MusicStrings in a selected song (see below).

3.1 Strings

The ***Strings*** form is used to define the MusicStrings in the selected song. MusicString notation is described in the next section.

MusicStrings can be edited within this form or, if they are more than two lines long, the **Editor** can be used to provide an editing window.

MusicStrings can be imported from other songs by using *File/Import Strings*. Libraries of MusicStrings can be built and used in this way.

3.2 Patterns

The ***Patterns*** form is used to declare the Patterns in the selected song. A Pattern is a container for other Patterns and MusicStrings.

The **Components** button opens a form for picking the Patterns and MusicStrings that are to be contained within a selected Pattern.

The **Play** button will play the selected Pattern.

The *File* options are *Export to MIDI, Export to WAV* and *Clone.*

When exporting to WAV, you can set the amount of padding to be applied after a song to allow any reverb to die away.

If you want to create a Pattern that is very similar to an existing Pattern, use *Clone* and edit the clone.

3.2.1 Example of Cloning

Suppose you have a song which is 128 bars long and the first 12 bars are repeated with a different 12th bar on the repeat.

Use the template to create a song where the voices have 2 parts with 128 bars per part. Clone each Part 1 and call this the Repeat. Insert the Repeat after Part 1 in the voice patterns. Delete bar 13 from Part 1 and delete bar 12 from the Repeat. Bar 13 in the Repeat becomes the different 12th bar.

You can keep the existing bar numbers and since bar numbers are independent for each part you can start Part 2 at bar 13.

3.3 Components

The ***Components*** form is used to pick the Patterns and MusicStrings that are to be contained within the selected Pattern and to specify their position in the sequence in which they are to be played. Picking is done via a drop down list which is displayed when you click on a **Component** cell.

The **Insert** and **Renumber** buttons can be used to insert components into an existing sequence.

The **Drill Down** button allows you to drill down to the child components of a selected Pattern. Drilling down on a MusicString will open a window for editing that MusicString. When selecting a component for **Drill Down** it is advisable to click on the **Position** cell otherwise the component pick list will be displayed.

If you would like to enter an anonymous MusicString into your Pattern, leave the **Component** cell blank and type the MusicString into the **Anonymous String** cell. This column can be made wider by dragging the boundary between the **Component** and **Anonymous String** headers.

3.4 Tree View

The tree view may be used to display the structure of Patterns and MusicStrings in a selected song.

Clicking on a MusicString will open a window for editing that MusicString.

Clicking on a Pattern will enable the **Play** button and also the *File* options: *Export to MIDI, Export to WAV* and *Clone.*

The **Play** button should only be used to preview your music. For an accurate rendition you should *Export to WAV* and then open the *WAV* file in Audacity® and then *Export as MP3*.

4 MusicStrings

4.1 Notes

A C Major scale of quarter notes, starting at middle C, can be written as:

```
C5q D5q E5q F5q G5q A5q B5q C6q
```

or as:

```
C5/0.25 D5/0.25 E5/0.25 F5/0.25 G5/0.25 A5/0.25 B5/0.25 C6/0.25
```

or, if you have **numeric durations** set to pulses, as:

```
C5/32 D5/32 E5/32 F5/32 G5/32 A5/32 B5/32 C6/32
```

In addition to the note letters, A to G, you can use R for a rest. Sharps, flats and naturals can be added by placing the character #, b or n immediately after the note letter so B-flat above middle C is written as Bb5.

Please note that, if you are transcribing music, accidentals in ***Audovia*** apply only to the immediately following note and not to the end of the bar as in conventional music notation.

MusicStrings can optionally be split into bars (or measures) by using the vertical bar character (|):

```
C5q D5q E5q F5q | G5q A5q B5q C6q |
```

As an alternative to note letters, MIDI values, enclosed in square brackets, may be used:

```
[60]q [62]q [64]q [65]q | [67]q [69]q [71]q [72]q |
```

4.1.1 Durations

The duration characters are:

w	whole note
h	half note
q	quarter note
i	eighth note
s	sixteenth note
t	thirty-second note
x	sixty-fourth note
o	one-twenty-eighth note

Dotted duration can be achieved by putting the period character (.) immediately after the duration character.

4.1.2 Chords

Chords are formed by adding the constituent notes together. A C Major chord can be written as:

```
C5q+E5q+G5q
```

4.1.3 Ties

Two or more notes of the same pitch can be tied together by using the hyphen character (-). Place the hyphen immediately after the duration of the note at the start of the tie and immediately before the duration of the note at the end of the tie. Notes in the middle of the tie have hyphens placed immediately before and after the note duration, as below.

```
C5q D5q E5q F5q- | F5-w- | F5-q G5q A5q B5q |
```

4.2 Tempo

A tempo of 120 beats per minute can be expressed as:

```
T120
```

or as:

```
T[allegro]
```

Note the use of a predefined numeric constant within the square brackets. More tempo constants are available from *Insert/Tempo*.

4.3 Constants

Constants are defined using the $ character followed by the constant name.

4.3.1 Numeric Constants

Numeric constants can be used anywhere that a number would appear in a MusicString. In addition to the predefined constants available from the *Insert* menu, you can define your own constants. For example, a bagpipe scale could be defined as:

```
$HA=70
$G=68
$F=67
$E=65
$D=63
$C=62
$B=60
$A=58
$LG=56
```

and played as:

```
[LG]q [A]q [B]q [C]q [D]q [E]q [F]q [G]q [HA]q
```

4.3.2 String Constants

Suppose you wanted to use the following arpeggio several times in your music.

```
F3i A3i C4i F4i C4i A3i
```

You could define a string constant as:

```
$arpeggioFoctave3=F3i~A3i~C4i~F4i~C4i~A3i
```

Then, in your music, you could refer to it as:

```
{arpeggioFoctave3}
```

Note the use of curly brackets for string constants.

4.4 Voices

Voices are specified by the V character followed by a number from 0 to 15. Note that V9 is the percussion voice and has its own set of instruments.

Voices can be subdivided into layers by using the L character followed by a number from 0 to 15. This is a way to get multiple melodies out of a single track and is particularly useful for the percussion channel.

4.5 Key Signatures

Key signatures are specified by the K character followed by a note letter (or a note letter followed by # or b) followed by maj or min to indicate a major or minor scale so the key of G Major is written as KGmaj.

4.6 Instruments

Instruments are specified by the I character followed by a number from 0 to 127. You can use *Insert/Instrument* to pick one of the predefined values.

4.7 MIDI Controller

MIDI controller events can be specified by the X character followed by the controller number followed by the equals sign (=) followed by a value. You can use *Insert/Controller* to pick one of the predefined controllers.

For example, if you want to set the volume of the current voice to a value of 12000, out of a possible 16383, use X[volume]=12000.

4.8 Pitch Wheel

A change of pitch can be specified by the & character followed by a number from 0 to 16383. This affects all following notes.

&0	lowers the pitch by a full tone;
&8192	returns the pitch to no change;
&16383	raises the pitch by a full tone.

5 Predefined Constants

5.1 Instrument Names

PIANO	0	PIZZICATO_STRINGS	45
ACOUSTIC_GRAND	0	ORCHESTRAL_STRINGS	46
BRIGHT_ACOUSTIC	1	TIMPANI	47
ELECTRIC_GRAND	2	STRING_ENSEMBLE_1	48
HONKEY_TONK	3	STRING_ENSEMBLE_2	49
ELECTRIC_PIANO	4	SYNTH_STRINGS_1	50
ELECTRIC_PIANO_1	4	SYNTH_STRINGS_2	51
ELECTRIC_PIANO_2	5	CHOIR_AAHS	52
HARPSICHORD	6	VOICE_OOHS	53
CLAVINET	7	SYNTH_VOICE	54
CELESTA	8	ORCHESTRA_HIT	55
GLOCKENSPIEL	9	TRUMPET	56
MUSIC_BOX	10	TROMBONE	57
VIBRAPHONE	11	TUBA	58
MARIMBA	12	MUTED_TRUMPET	59
XYLOPHONE	13	FRENCH_HORN	60
TUBULAR_BELLS	14	BRASS_SECTION	61
DULCIMER	15	SYNTHBRASS_1	62
DRAWBAR_ORGAN	16	SYNTH_BRASS_1	62
PERCUSSIVE_ORGAN	17	SYNTHBRASS_2	63
ROCK_ORGAN	18	SYNTH_BRASS_2	63
CHURCH_ORGAN	19	SOPRANO_SAX	64
REED_ORGAN	20	ALTO_SAX	65
ACCORDIAN	21	TENOR_SAX	66
HARMONICA	22	BARITONE_SAX	67
TANGO_ACCORDIAN	23	OBOE	68
GUITAR	24	CHANTER	68
NYLON_STRING_GUITAR	24	ENGLISH_HORN	69
STEEL_STRING_GUITAR	25	BASSOON	70
ELECTRIC_JAZZ_GUITAR	26	CLARINET	71
ELECTRIC_CLEAN_GUITAR	27	PICCOLO	72
ELECTRIC_MUTED_GUITAR	28	FLUTE	73
OVERDRIVEN_GUITAR	29	RECORDER	74
DISTORTION_GUITAR	30	PAN_FLUTE	75
GUITAR_HARMONICS	31	BLOWN_BOTTLE	76
ACOUSTIC_BASS	32	SKAKUHACHI	77
ELECTRIC_BASS_FINGER	33	WHISTLE	78
ELECTRIC_BASS_PICK	34	OCARINA	79
FRETLESS_BASS	35	LEAD_SQUARE	80
SLAP_BASS_1	36	SQUARE	80
SLAP_BASS_2	37	LEAD_SAWTOOTH	81
SYNTH_BASS_1	38	SAWTOOTH	81
SYNTH_BASS_2	39	LEAD_CALLIOPE	82
VIOLIN	40	CALLIOPE	82
VIOLA	41	LEAD_CHIFF	83
CELLO	42	CHIFF	83
CONTRABASS	43	LEAD_CHARANG	84
TREMOLO_STRINGS	44	CHARANG	84

LEAD_VOICE	85	BRIGHTNESS	100
VOICE	85	FX_GOBLINS	101
LEAD_FIFTHS	86	GOBLINS	101
FIFTHS	86	FX_ECHOES	102
LEAD_BASSLEAD	87	ECHOES	102
BASSLEAD	87	FX_SCI-FI	103
PAD_NEW_AGE	88	SCI-FI	103
NEW_AGE	88	SITAR	104
PAD_WARM	89	BANJO	105
WARM	89	SHAMISEN	106
PAD_POLYSYNTH	90	KOTO	107
POLYSYNTH	90	KALIMBA	108
PAD_CHOIR	91	BAGPIPE	109
CHOIR	91	FIDDLE	110
PAD_BOWED	92	SHANAI	111
BOWED	92	TINKLE_BELL	112
PAD_METALLIC	93	AGOGO	113
METALLIC	93	STEEL_DRUMS	114
PAD_HALO	94	WOODBLOCK	115
HALO	94	TAIKO_DRUM	116
PAD_SWEEP	95	MELODIC_TOM	117
SWEEP	95	SYNTH_DRUM	118
FX_RAIN	96	REVERSE_CYMBAL	119
RAIN	96	GUITAR_FRET_NOISE	120
FX_SOUNDTRACK	97	BREATH_NOISE	121
SOUNDTRACK	97	SEASHORE	122
FX_CRYSTAL	98	BIRD_TWEET	123
CRYSTAL	98	TELEPHONE_RING	124
FX_ATMOSPHERE	99	HELICOPTER	125
ATMOSPHERE	99	APPLAUSE	126
FX_BRIGHTNESS	100	GUNSHOT	127

5.2 Percussion Names

ACOUSTIC_BASS_DRUM	35	RIDE_BELL	53
BASS_DRUM	36	TAMBOURINE	54
SIDE_STICK	37	SPLASH_CYMBAL	55
ACOUSTIC_SNARE	38	COWBELL	56
HAND_CLAP	39	CRASH_CYMBAL_2	57
ELECTRIC_SNARE	40	VIBRASLAP	58
LOW_FLOOR_TOM	41	RIDE_CYMBAL_2	59
CLOSED_HI_HAT	42	HI_BONGO	60
HIGH_FLOOR_TOM	43	LOW_BONGO	61
PEDAL_HI_HAT	44	MUTE_HI_CONGA	62
LOW_TOM	45	OPEN_HI_CONGA	63
OPEN_HI_HAT	46	LOW_CONGA	64
LOW_MID_TOM	47	HIGH_TIMBALE	65
HI_MID_TOM	48	LOW_TIMBALE	66
CRASH_CYMBAL_1	49	HIGH_AGOGO	67
HIGH_TOM	50	LOW_AGOGO	68
RIDE_CYMBAL_1	51	CABASA	69
CHINESE_CYMBAL	52	MARACAS	70

SHORT_WHISTLE	71
LONG_WHISTLE	72
SHORT_GUIRO	73
LONG_GUIRO	74
CLAVES	75
HI_WOOD_BLOCK	76
LOW_WOOD_BLOCK	77
MUTE_CUICA	78
OPEN_CUICA	79
MUTE_TRIANGLE	80
OPEN_TRIANGLE	81

5.3 Controller Names

BANK_SELECT_COARSE	0
MOD_WHEEL_COARSE	1
BREATH_COARSE	2
FOOT_PEDAL_COARSE	4
PORTAMENTO_TIME_COARSE	5
DATA_ENTRY_COARSE	6
VOLUME_COARSE	7
BALANCE_COARSE	8
PAN_POSITION_COARSE	10
EXPRESSION_COARSE	11
EFFECT_CONTROL_1_COARSE	12
EFFECT_CONTROL_2_COARSE	13
SLIDER_1	16
SLIDER_2	17
SLIDER_3	18
SLIDER_4	19
BANK_SELECT_FINE	32
MOD_WHEEL_FINE	33
BREATH_FINE	34
FOOT_PEDAL_FINE	36
PORTAMENTO_TIME_FINE	37
DATA_ENTRY_FINE	38
VOLUME_FINE	39
BALANCE_FINE	40
PAN_POSITION_FINE	42
EXPRESSION_FINE	43
EFFECT_CONTROL_1_FINE	44
EFFECT_CONTROL_2_FINE	45
HOLD_PEDAL	64
HOLD	64
PORTAMENTO	65
SUSTENUTO_PEDAL	66
SUSTENUTO	66
SOFT_PEDAL	67
SOFT	67
LEGATO_PEDAL	68
LEGATO	68
HOLD_2_PEDAL	69
HOLD_2	69
SOUND_VARIATION	70
VARIATION	70
SOUND_TIMBRE	71
TIMBRE	71
SOUND_RELEASE_TIME	72
RELEASE_TIME	72
SOUND_ATTACK_TIME	73
ATTACK_TIME	73
SOUND_BRIGHTNESS	74
BRIGHTNESS	74
SOUND_CONTROL_6	75
CONTROL_6	75
SOUND_CONTROL_7	76
CONTROL_7	76
SOUND_CONTROL_8	77
CONTROL_8	77
SOUND_CONTROL_9	78
CONTROL_9	78
SOUND_CONTROL_10	79
CONTROL_10	79
GENERAL_PURPOSE_BUTTON_1	80
GENERAL_BUTTON_1	80
BUTTON_1	80
GENERAL_PURPOSE_BUTTON_2	81
GENERAL_BUTTON_2	81
BUTTON_2	81
GENERAL_PURPOSE_BUTTON_3	82
GENERAL_BUTTON_3	82
BUTTON_3	82
GENERAL_PURPOSE_BUTTON_4	83
GENERAL_BUTTON_4	83
BUTTON_4	83
EFFECTS_LEVEL	91
EFFECTS	91
TREMULO_LEVEL	92
TREMULO	92
CHORUS_LEVEL	93
CHORUS	93
CELESTE_LEVEL	94
CELESTE	94
PHASER_LEVEL	95
PHASER	95
DATA_BUTTON_INCREMENT	96
DATA_BUTTON_INC	96
BUTTON_INC	96
DATA_BUTTON_DECREMENT	97
DATA_BUTTON_DEC	97

BUTTON_DEC	97	OMNI_MODE_OFF	124
NON_REGISTERED_COARSE	98	OMNI_OFF	124
NON_REGISTERED_FINE	99	OMNI_MODE_ON	125
REGISTERED_COARSE	100	OMNI_ON	125
REGISTERED_FINE	101	MONO_OPERATION	126
ALL_SOUND_OFF	120	MONO	126
ALL_CONTROLLERS_OFF	121	POLY_OPERATION	127
LOCAL_KEYBOARD	122	POLY	127
ALL_NOTES_OFF	123		

5.4 Combined Controller Names

(index = coarse_controller_index * 128 + fine_controller_index)

BANK_SELECT	16383	BALANCE	1064
MOD_WHEEL	161	PAN_POSITION	1322
BREATH	290	EXPRESSION	1451
FOOT_PEDAL	548	EFFECT_CONTROL_1	1580
PORTAMENTO_TIME	677	EFFECT_CONTROL_2	1709
DATA_ENTRY	806	NON_REGISTERED	12770
VOLUME	935	REGISTERED	13028

5.5 Values for some controllers

ON	127
OFF	0
DEFAULT	64

5.6 Tempo Values

GRAVE	40	ANDANTINO	80
LARGO	45	MODERATO	95
LARGHETTO	50	ALLEGRETTO	110
LENTO	55	ALLEGRO	120
ADAGIO	60	VIVACE	145
ADAGIETTO	65	PRESTO	180
ANDANTE	70	PRETISSIMO	220

6 XML Specification

Here is the XML specification for the *.sbxml* files used in *File/Song Export* and *File/Song Import*.

```
<?xml version="1.0"?>

<songs>

  <song>

    <song_name>Song Name</song_name>
    <numeric_duration_type>decimal or pulses</numeric_duration_type>

    <components>

      <component>

        <component_type>pattern or string</component_type>
        <component_name>Pattern or String Name</component_name>
        <string_value>a JFugue MusicString</string_value>

      </component>

    </components>

    <pattern_components>

      <pattern_component>

        <pattern_name>Pattern Name</pattern_name>
        <component_position>an integer value</component_position>
        <component_type>pattern or string</component_type>
        <component_name>Pattern or String Name</component_name>
        <anonymous_string>a JFugue MusicString</anonymous_string>

      </pattern_component>

    </pattern_components>

  </song>

</songs>
```