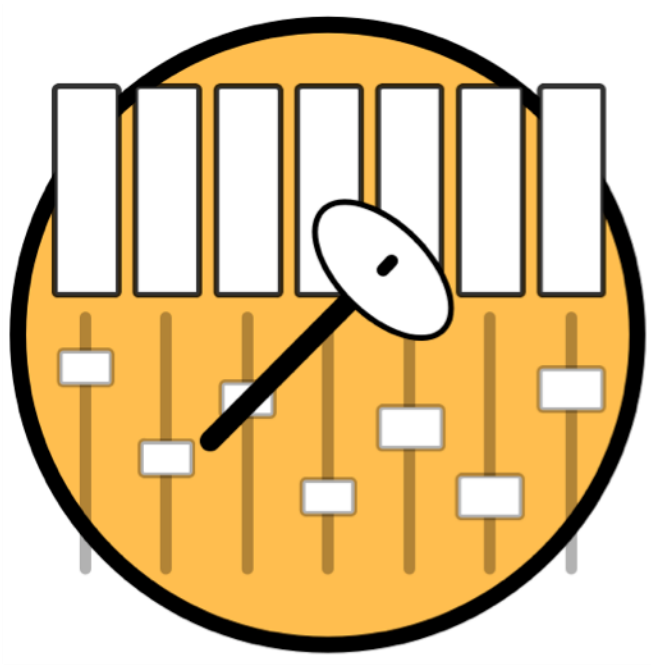
Pipilan

Composition and performance software for gamelan and electronics



http://www.augmentedgamelan.com/

Contents

Introduction

This software has been developed in Max/MSP and remains a work in progress. It has been designed as a composition tool for my own work, with certain elements to be used in performance. This iteration in particular has been designed to handle random sequences rather than traditional gamelan music.

Previous versions of this software have been used in the creation of both original compositions and arrangements of traditional Javanese material. I am sharing this primarily as a means of communicating some of the ideas behind my work, as a "sound toy" to experiment with, but also in the hope that it may provide the inspiration for more serious compositions.

The development of this software has been made possible through funding from the Arts and Humanities Research Council and formed part of a research degree in the Art and Design Research Institute at Middlesex University.

Further information

I have not attempted to provide an introduction to gamelan here. Readers interested in learning more about traditional Javanese gamelan may find the following links useful:

An Introduction to Javanese Gamelan (PDF) by Sumarsam:
http://sumarsam.web.wesleyan.edu/Intro.gamelan.pdf

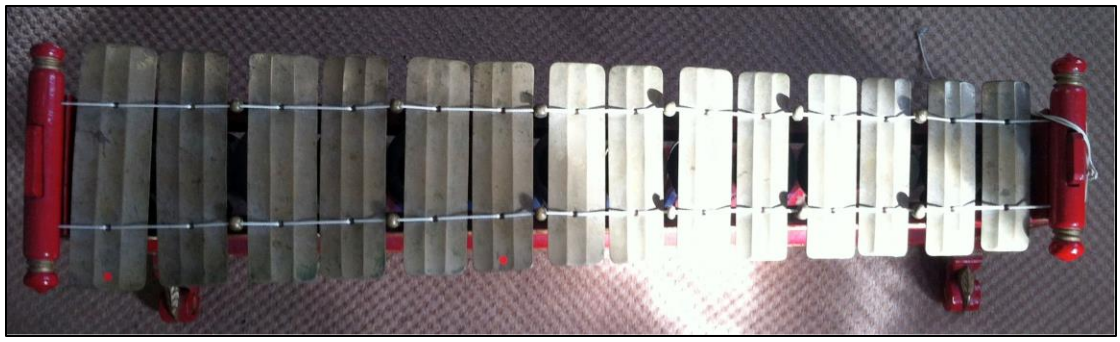
A Gamelan Manual by Richard Pickvance
http://www.amazon.co.uk/Gamelan-Manual-Players-Central-Javanese/dp/0955029503/ref=sr_1_1?ie=UTF8&qid=1351002207&sr=8-1

Southbank website for gamelan and education:
http://ticketing.southbankcentre.co.uk/gamelan-at-southbank-centre

Instruments

Pipilan is based around a set of instruments called *gender*, typically encountered as part of a full Javanese gamelan. The *gender* are often thought of as elaborating instruments, and their primary purposes are to accompany vocal parts in ensembles and the *dhalang* in shadow puppet theatre. There are three main types of *gender* in Javanese gamelan: the *gender barung* (14 keys, often referred to simply as *gender*), *gender panerus* (14 keys, higher-pitched), and *slenthem* (7 keys, sometimes referred to as *gender panembung*). There is generally only one each of these instruments for each tuning system in a full gamelan.

In ensemble performance the *gender* is used for elaborate patterns called *cengkok*, which are played as duo-phonic melodies using a soft mallet each in both hands. This software draws on a technique called *mipil* (or *pipilan*), which is more commonly associated with an instrument called the *bonang*. The instruments are also currently treated as monophonic. Although it is not possible at this stage to recreate typical *gender* repertoire, I hope that the patterns generated with this software might be used to generate material that maintains compatibility with more complex techniques.



gender (*gender barung*)

slenthem (*gender panembung*)

This software includes samples of two *gender barung* and a *slenthem*. Since the *gender barung* come from different sets, their tunings do not match exactly and produce beating tones when played together. This instrumental ensemble was originally put together for a performance in the Union Chapel, London, and has formed the basis of several subsequent compositions.

Terminology

The following Javanese terms are used throughout the software and documentation where there is no equivalent English term or where similar musical terms might be misleading. They are presented as used in this context and might vary in actual usage – see the links in the introduction for more details.

Balungan – central melody that is interpreted for all the parts in this software apart from the *gong*. This is also the part that the *slenthem* plays directly.

Gantung – (also *gantungan*, *gt*) can be translated as "hanging" – a pattern that plays around a single note.

Gatra – a phrase lasting four beats.

Gender – (pronounced with a hard "g") – a type of instrument consisting of several keys, typically brass or bronze, suspended over resonating tubes. These are typically played with padded beaters but can be struck or bowed in various ways in contemporary music. In Javanese music a *gender* typically has 14 keys (see *slenthem*)

Garap – the process of working out parts. In the case of this software this means how the *balungan* is interpreted, e.g. which order the notes are played in.

Gong – generally a large disc-shaped instrument capable of producing low frequencies, which is typically used to mark out important points in a melodic/rhythmic cycle. The *gong* used in this software is based on the *gong kemodhong*, which consists of two gender-type keys tuned a few Hz apart to create a deep tone with prominent beating.

Irama – can be translated as "rhythm". In this software this describes the relative density of the *gender* A part to the *balungan*. If the *balungan* is played slowly then there is space for more notes to be played on the *gender*. There are two levels of *irama* in this software.

Mipil – (also *pipilan*) can be translated as "picking". Used to describe the act of taking several notes as reference and playing them in a different order, often anticipating where they are played in the *balungan*.

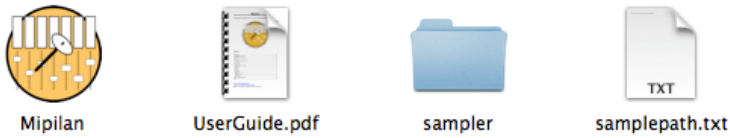
Pin – a rest. In the case of the *balungan* this is generally seen as a continuation of the previous note rather than a silence.

Seleh – a goal tone – here the last note in a *gatra* or half-*gatra* that is not a *pin*.

Slenthem – (pronounced *slentem*) – a gender-type instrument with seven keys that typically plays a part almost identical to the *balungan*.

Installation and setup

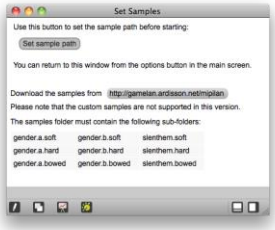
Prior to loading the software, create a *Pipilan* folder in the applications folder or your preferred location. Download the associated sample pack and place it within this folder along with the application. Unzip the files by double-clicking on them – you may then delete the original zip files.



contents of pipilan folder (samplepath.txt will be created in next step)

The sample pack has been created as a separate download in order to minimize download size between versions. Please note that this patch is not designed to work with third party samples at this stage, but this may be achieved by providing a set of .aif files following the same folder structure and naming convention.

Upon loading the application for the first time a dialog will appear (see image on the right). This will create a text file in the same folder called [samplepath.txt]. The application may crash upon the first attempt, but you will not need to repeat this step provided the text file has been created.



When updating the software to a new version you may leave the samples folder and samplepath.txt file in place, and replace the application file. If you wish to use a different set of samples you may either replace the folder, giving it the same name, or repeat the step above by accessing the window from the options screen.

To check that audio is working, press play or click on the representations of instruments in the main screen. If no sound follows, click the options button at the top of the screen (see **1B** in the screenshots on the following page). Here it is possible to test audio through generation of a burst of white noise, or access the audio status window to assign a sound card.

For more details and troubleshooting of the audio status window please see the Max/MSP documentation here:

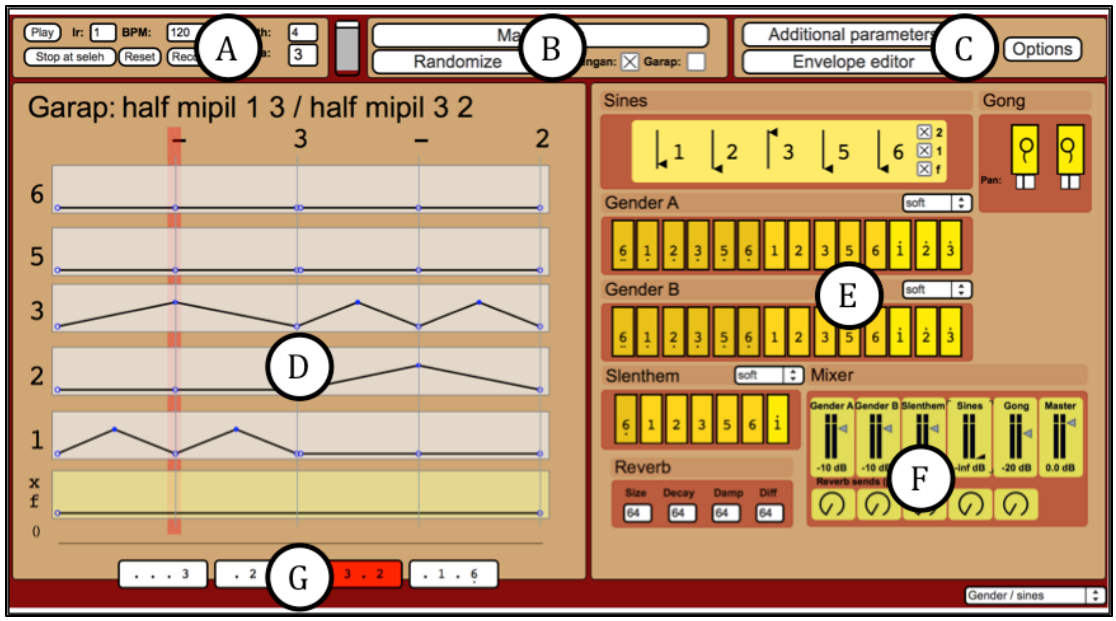
http://cycling74.com/docs/max5/vignettes/core/dsp_status.html

Please note that this software may be CPU intensive. A Core2Duo processor and at least 4GB of RAM is recommended. In the event of audio breaking up or general unresponsiveness please close any other active applications, including web browsers.

Layout

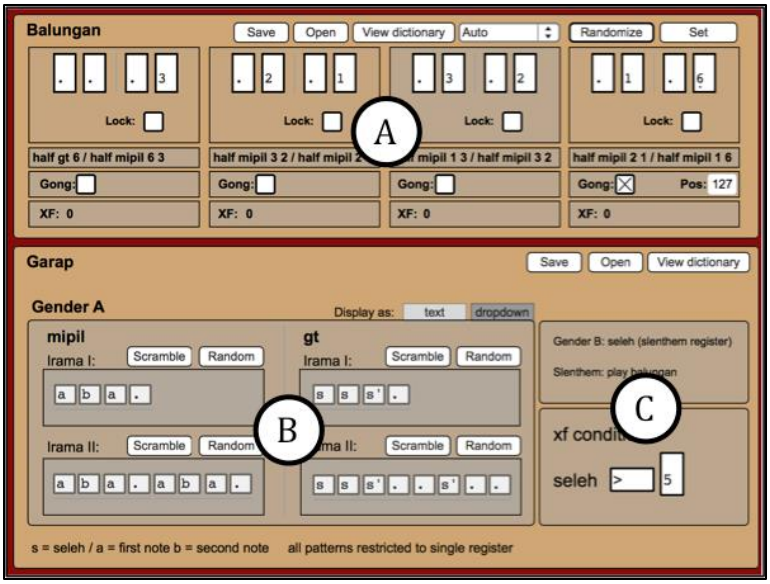
Pipilan consists of two main windows: **performance** and **editor**.

1. Performance window



1A: transport | **1B**: editor | **1C**: additional parameters and options | **1D**: envelope display | **1E**: instrument and sine wave part representations | **1F**: mixer | **1G**: current position in *balungan* (display only)

2. Editor



2A: *Balungan* editor | **2B:** *garap* editor | **2C:** additional garap and crossfade

Quick Start

Playback and mixing

The performance screen (see **layout 1**) is dominated by a set of envelopes on the left (**1D**) – these represent the volume of a set of sine waves and are not editable directly. On the right hand side are representations of the instruments (**1E**) - these can be clicked to audition sounds and in some cases include parameter controls. Situated next to these are a mixer for each channel and reverb sends and controls (**1F**).

In order to make sound, first a pattern must be generated; this may be achieved by clicking the randomize button or accessing the editor window (**1A**). The transport controls (play and stop, BPM, and record*) can be found at the top-left of the window (**1A**).

When the BPM crosses a certain threshold (100BPM by default), a change in *irama* is triggered: a second, denser set of patterns are then used in the *gender* part.

Editing

To start editing patterns, click on the *editor* button in the transport controls (**1A**). This will bring up the editor window (**layout 2**). Try clicking the randomize button in the top left hand corner of the screen (**2A**), or edit the main pattern by clicking on the number boxes. The instrumental and sine wave parts will change accordingly. Check the comments at the bottom of each box to see which patterns are being used.

To change the way in which *gender* A interprets the parts, try editing the garap boxes (**2B**). By default these patterns are editable through dropdown menus, but may be switched to text input for quick access. Use the randomize or scramble buttons for quick variations.

Patterns in the *gt* column must consist of combinations of [s] [s'] and [.]. Try these examples:

***Ir* I:** [s s s s] [. s . s'] [. . . s] ***Ir* II:** [. s . s . s' . s'] [. s s . . s' s' .] [. . . s . . . s']

Patterns in the *mipil* column should consist of combinations of [a] [b] and [.]. e.g.:

***Ir* I**: [a b a b] [a a a b] [a . b .] ***Ir* II**: [a a a b a a a b] [a a b b a a b b]

* In lieu of a save dialog, recordings are pre-named and saved to the same folder as the application.

Notation and tuning

Kepatihan notation

Gamelan music is often notated with numeric representation of the *balungan* tones and other notes. The examples presented here use the *Kepatihan* system of notation, which has been rendered in font form as *Kepatihan Pro*. Registers ("octaves") are indicated by dots above or below the numbers. The perceived pitch of any given note is relative to the general pitch of the instrument; e.g. a "note 1" played on a *slenthem* is a register lower than the equivalent "note 1" on the *gender*. The complete range available for notation is as follows:

Low: 1 2 3 4 5 6 7 1 2 3 4 5 6 7

Middle: 1 2 3 4 5 6 7

High: 1 2 3 4 5 6 7 1 2 3 4 5 6 7

Laras slendro (tuning)

Gamelan music typically uses one of two tuning systems at a time, called *slendro* and *pelog*. These are occasionally mixed together in contemporary pieces, and very rarely in traditional pieces. The tuning system used in this software is *slendro*: a five-tone scale notated with the numbers one to six (four is omitted).

The basic range of the *slendro* tuning system is as follows:

3 5 6 1 2 3 5 6 1 2 3

The range of a *slenthem*, which forms the constraints for sequences written in this software, is:

6 1 2 3 5 6 1

Variation between ensembles

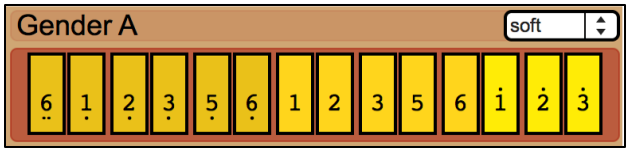
There is no fixed frequency reference for tuning in gamelan, although the relative intervals are recognizable across ensembles. As the instruments presented here are from two different ensembles, when played together the two *gender* create beating tones through the differences in tuning.

Instrument functions

The bundled samples represent three basic techniques of playing:

- **Soft**: striking keys with traditional padded beater.
- **Hard**: striking keys with a vibraphone beater.
- **Bowed**: keys bowed with violin or *rebab* bow.

These may be selected through a drop-down menu at the right hand side of each instrument:



In this version of the software the roles of each instrument are fixed, with *gender A* playing a user-definable part (see the *garap* section below for more details):

Gender A	***Gender B***	***Slenthem***	**Sine waves**
Mipil / gantung	*Seleh*	*Balungan*	Amplitude envelopes based on *pipilan / gantungan*

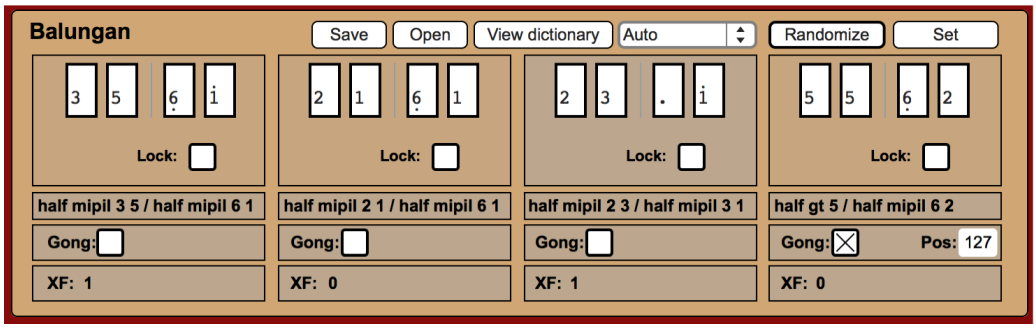
Balungan / slenthem

In Javanese gamelan *balungan* is the name given to a central melodic structure that is often used as a reference point for instrumental parts. Some parts play something along the lines of the *balungan* directly, while others play more elaborate parts (*cengkokan*) that generally coincide with the most important notes, or might pick notes from it in anticipation (*mipil*). In this version of the software the *slenthem* plays the *balungan* directly, and so editing the *balungan* may be seen as effectively editing the *slenthem* part.

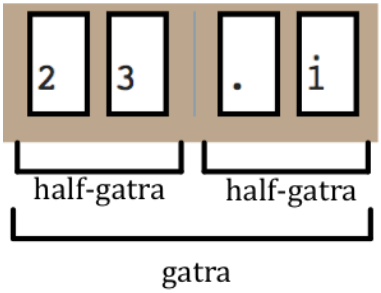
Editing the *balungan*

The *balungan* edit window has been built around the auditioning of random sequences, but may be used for composing *balungan* as well. The editor allows for the creation of patterns using low 6 and high 1 notes to be played on the *slenthem*, but these are converted to a single register when approached through *garap* for the *gender*.

The *balungan* is divided into sets of four notes called *gatra*, represented by the four boxes in the edit window. When randomizing patterns each *gatra* may be locked in place by ticking the **lock** box.



Each *gatra* displays the collective *garap* for *gender A* and the sine wave part at the bottom of its box (note that each pattern is split into halves). These are updated upon editing or randomizing the *balungan*.



Gongs

The gong section is based on the *gong kemodhong*, a pair of *gender*-type keys placed over resonating tubes that is often used when there is no space for a hanging gong. In a traditional context the keys may be struck at the same time or with a slight delay. As this is the part with the least variation in timbre and melodic information I have introduced parameters for creative input, based on how I have approached the emulation of the role of a gong in my own compositions: see **additional parameters**.

The gong tones may be tested by clicking the circles in each of the graphic representations of the instruments. Note that the default frequencies assigned to this instrument are likely to be inaudible on laptop speakers.

One gong stroke may be placed per *gatra* by checking the appropriate box below the *balungan*. The position of each stroke is editable through the **pos** value (0-127).

Gender A

Garap

Garap is the process of working out parts, typically from a central framework. In traditional Javanese gamelan there are a variety of ways this might be performed. In the case of this software it has been simplified into two primary patterns that make direct references to the *balungan*. These are called *mipil* and *gantung*, which mean "picking" and "hanging" respectively. When broken down into pairs, the notes will be picked at one by one if they are different; if they are the same or contain spaces, the part will hang around the shared note.

The various notes of the *gatra* are analyzed and classified as follows:

a	first note of a half-*gatra*
b	second note of a half-*gatra*
s	*seleh* (strong note, or in this case the only note possible to refer to)
.	*pin* (rest)

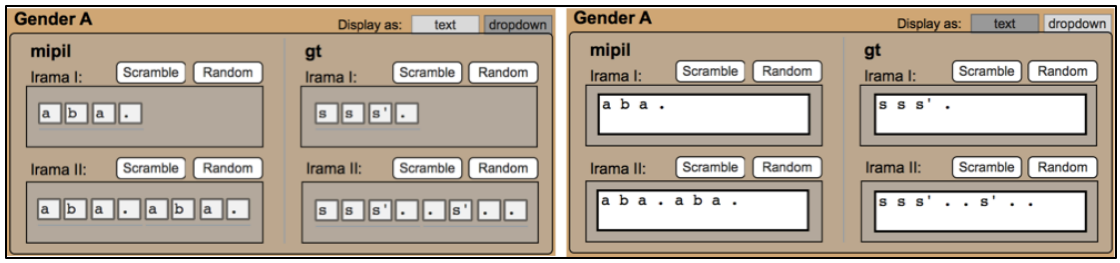
All notes in the *balungan* are currently converted to a single register when applying *garap* for *gender* A. As a result an adjacent low and medium register "6" are treated as the same note and generate a *gantung* pattern (see the flow chart below).

Irama

Irama can be translated literally as "rhythm". The term is commonly used in Javanese gamelan to denote levels of rhythmic density. In this version of the software there are two levels of *irama* that apply to the *gender A* part:

Irama	Ratio to *slenthem* part
1	2 : 1
2	4 : 1

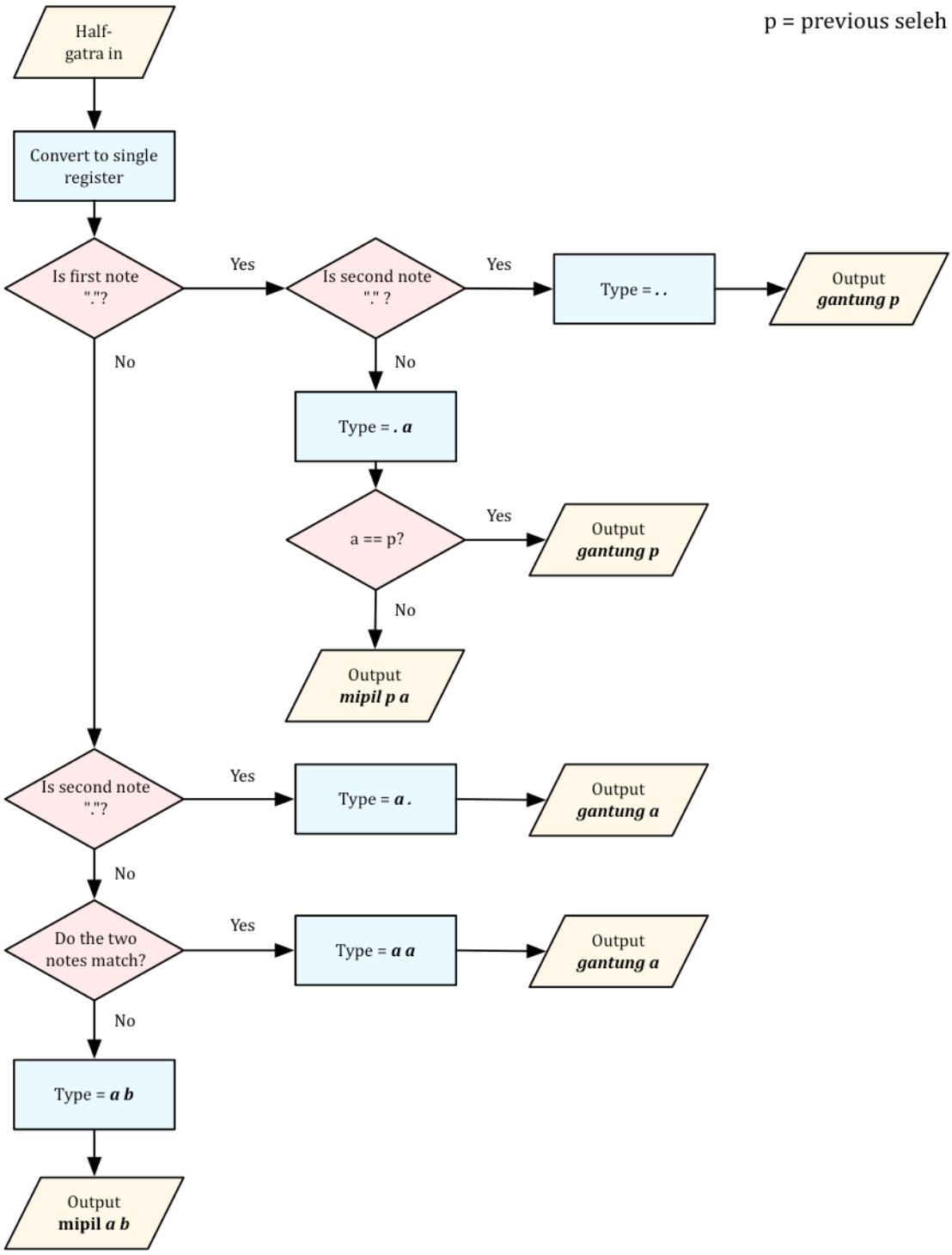
Each level has an associated set of patterns, which must be triggered by the appropriate BPM threshold. These may be edited using the text boxes or drop-down editors in the ***garap* editor**:



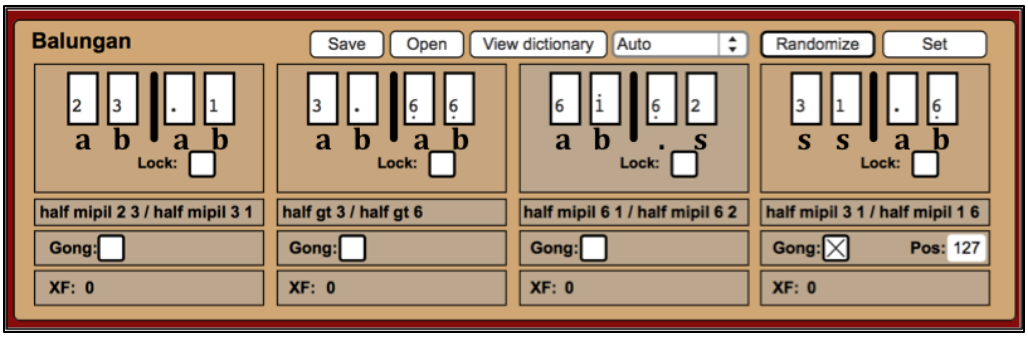
Editing *garap*

The parts for *gender A* refer to the *balungan* at the level of half gatra – each half gatra is either split up into notes *a* and *b* or refers to a *seleh*.

Each half-*gatra* is classified as *mipil* or *gantung* by applying the following process:



The following screenshot demonstrates the way that the value a b or *seleh* is allocated to represent each note*:



This randomly generated *balungan* will be used throughout subsequent examples in this section.

Mipil

Mipil patterns refer to notes **a** and **b** of every half-*gatra*. These patterns anticipate the *balungan* by one or two notes depending on the current *irama* setting. The default patterns for *irama* I and II, [**a b a .**] and [**a b a . . b a .**] respectively, will generate the following results when applied to the *gatra* [2 3 . 1]:

In the first half-*gatra* (**2 3**), **a** = **2**, **b** = **3**:

```
a b a . | a b a . a b a .

2 3 2 . | 2 3 2 . 2 3 2 .
```

In the second half-*gatra* (**. 1**), **a** must take its value from the last note of the previous phrase: **a** = **3**, **b** = **1**:

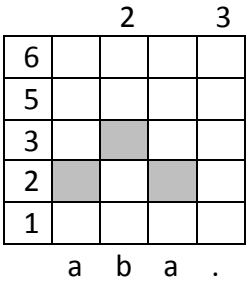
```
a b a . | a b a . a b a .

3 1 3 . | 3 1 3 . 3 1 3 .
```

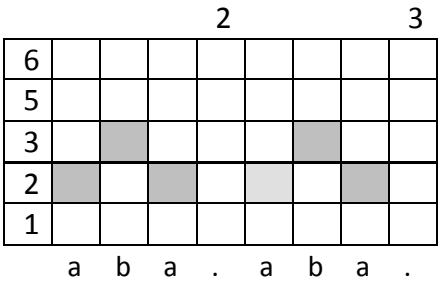
The following examples represent the timing in relation to the *balungan* (*slenthem* part) the note played is marked by grey boxes, with the *balungan* above and reference note underneath:

* The determination of *seleh* notes from a *balungan* and vice versa in traditional gamelan music is generally a more complicated process. Further information on *seleh* allocation specific to this software may be found in the ***gender* B** section.

Mipil **2 3** (*irama* I):



Mipil **2 3** (*irama* II):



Gantung

Gantung patterns are designed to refer to the *seleh* note (*s*). A modifier, denoted by " ' ", may be added to take the note into the next register up or down depending on the availability set out by the range of the *gender*. Since the *balungan* is currently converted to a single register before processing, gantungan patterns for low and high sixes and ones will trigger the same note:

Gt note	**Note with ' modifier applied**
1	$\dot{1}$
2	$\dot{2}$
3	$\dot{3}$
5	$\underset{\cdot}{5}$
6	$\underset{\cdot}{6}$

In the following graphic examples the note played is marked by grey boxes; notes with modifiers are marked with blue boxes:

In the first half-*gatra* (3 .) the second note is empty, and so the first note is used as the *seleh*. **s** = **3** and **s'** = **high 3**:

s s s' . | s s s' . . . s' .

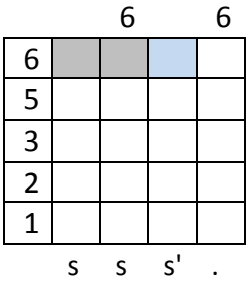
3 3 3̇ . | 3 3 3̇ . . . 3̇ .

In the second half-*gatra* (**6 6**), the only note available to refer to is **6**. Since there is no high 6 available on the *gender*, the modifier uses a low note instead: **s** = **6** and **s'** = **low 6**:

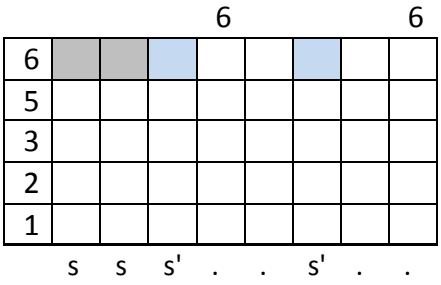
s s s' . | s s s' . . . s' .

6 6 6̣ . | 6 6 6̣ . . . 6̣ .

Gantung 6 (*irama* I):



Gantung 6 (*irama* II):

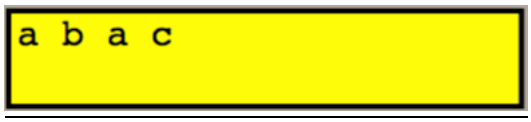


Randomization

Two types of randomization are available: **random** and **scramble**. **Random** will create an entirely random pattern using any values available ("a", "b", and "." in the case of *mipil*, and "s", "s'", and "." in the case of *gantungan*). **Scramble** takes the existing pattern and changes the order of its constituent elements.

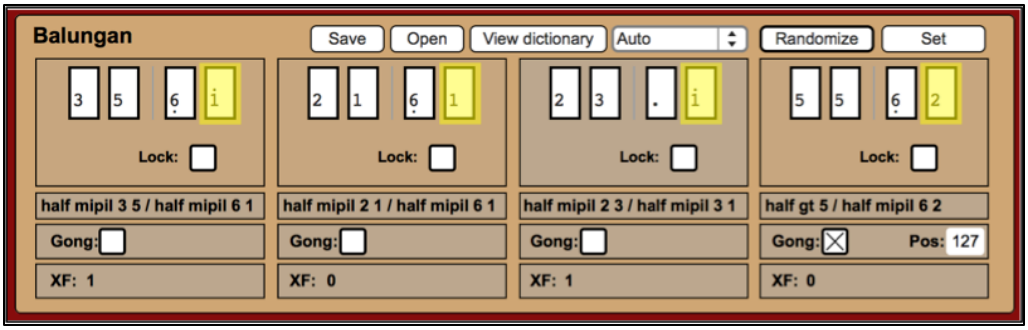
Errors – status colours

If unexpected input is found in the text boxes in the *garap* window the colour will change accordingly:

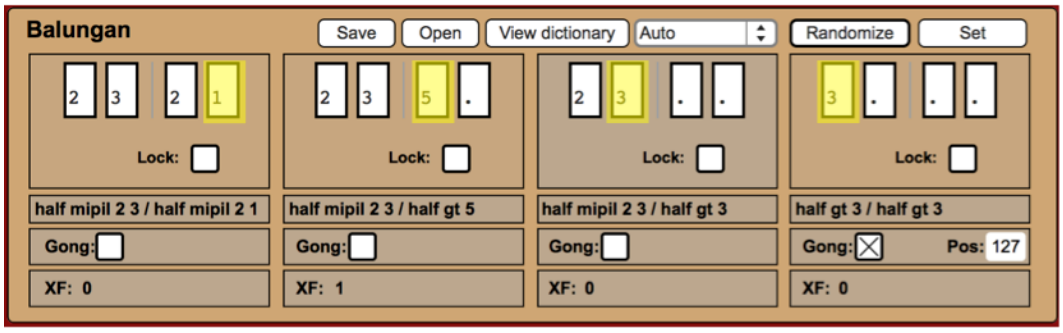
White  This is the normal state. The text box contains the correct number and type of characters.	Yellow  The text box contains at least one unpermitted character (in this case "c").
Pink  The text box contains too many or too few characters (in this case one too many).	Red  The text box contains too many characters and at least one of them is unpermitted (in this case one too many, and one of them is "c").

Gender B

The part for *gender* B is fixed in this version of the software, playing the *seleh* note of each *gatra* in the same range as the *slenthem* in order to generate a beating texture. The *seleh* is generally the last note of the *gatra*, as shown in yellow here:

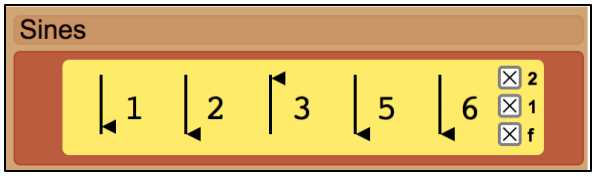


In the event that the last note is a *pin*/rest (e.g. [**2 3 . .**], the last note filled note value will be used as the *seleh*.



Sine waves and patterns

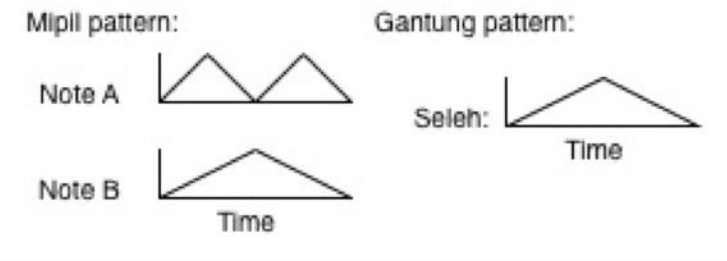
Each of the sine waves have been tuned by ear to complement the *gender* with the intention of creating some beating, and do not conform to standard *slendro* tuning. The frequencies are fixed and the interaction/control for each note is based around modulation of the amplitude. This may be tested using the sliders in the sine wave section of the ensemble:



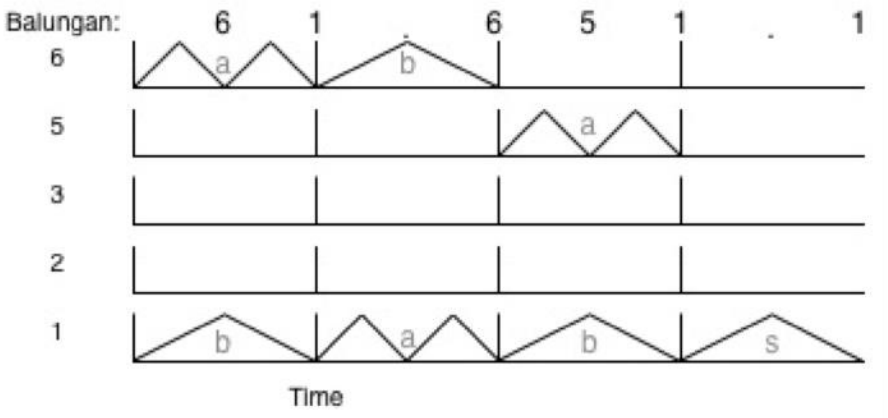
In addition to the fundamental frequencies I have added prominent partials from analysis of another instrument, a *saron*. The partials to be played for the set of sine waves may be selected by checking the **f**, **1** and **2** boxes next to the volume sliders. An additional slider to the right of these boxes sets the minimum value for all envelopes during playback.

Envelopes

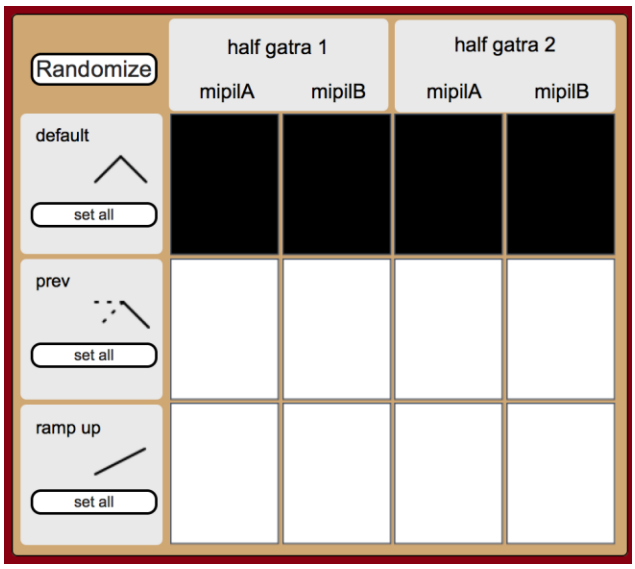
The volume envelopes themselves are not editable directly at this stage of development, and are linked to the *garap* used by *gender A*. As in the *gender* part, there are two types of pattern: *mipil* and *gantung*. These are currently fixed independently of *irama*, and take shape as follows:



In context these patterns may appear as follows:



Variations on these basic envelopes may be explored through the experimental editor accessible from the main screen:

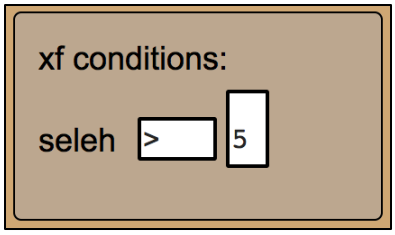


The two alternative envelopes, **ramp up** and **prev**, may be used in various combinations to create patterns spanning across *gatra*. **Ramp up** creates a fading-up gesture lasting the duration of the default envelope. **Prev** is essentially the same as the default envelope, but uses a value taken from the previous *gatra* to create the start point; if a **prev** envelope follows a **ramp up** then an extension of the pattern is created. The shape of these envelopes may be modified further in performance through the **additional parameters** window.

XF

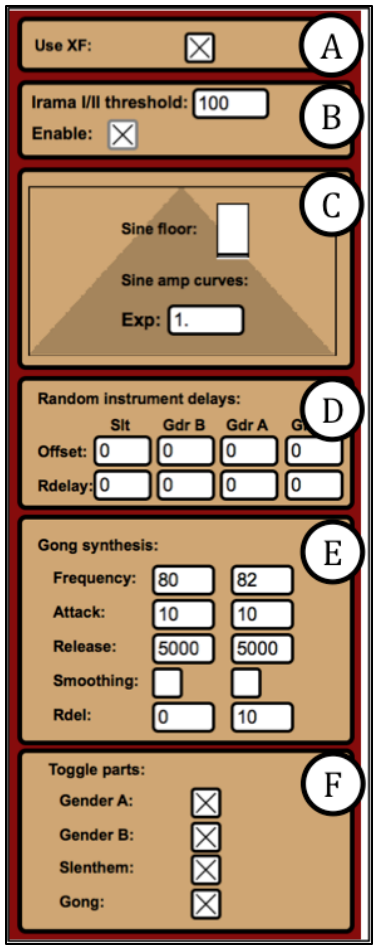
The sine wave generators are based on wavetable synthesizers, and are driven by two sets of oscillators: bank **A** uses a bank of plain sine waves, whereas bank **B** uses a wavetable with a slightly richer spectrum. The balance of the two banks is controlled by a crossfade parameter, which functions as an envelope alongside the other note-based events.

The value of the crossfade is determined by the *seleh* note exceeding a threshold; a ramp is created automatically from the previous *gatra* value. The threshold and operator may be set from the editor window:



Additional parameters

A selection of parameters may be edited for further manipulation of sound. These are listed in reference to the image shown on the right:



A) Use XF

This checkbox turns the crossfade parameter on and off (on by default).

B) *Irama* threshold

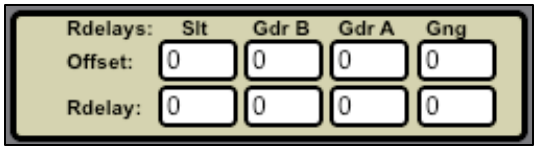
The number value sets the BPM value at which the *gender* A pattern changes from *irama* I to II; this can be disabled altogether using by unchecking the accompanying **enable** checkbox.

C) Envelope shapes

Sine floor and **amp curves** affect the shape of the envelopes. The **floor** sets the minimum value of the envelopes, while the **exp** value can be used to create a sharper ramp. Click towards the right-hand side of the number box to fine-tune this value.

D) Random instrument delays

Each instrument can have a random delay applied to its notes – this represents a crude "humanization" in terms of timing but can be used to more creative ends.



Rdelay stands for "random delay", and specifies the range of the random number generated each time a note is played. Each delay has a fixed offset time.

E) Gong parameters

Pan (in main interface): stereo positioning for each of the two constituent tones.

Frequency: limited to a range of 20-1000Hz. Pairs of frequencies around 80-100Hz spaced a couple of Hz apart are recommended for a traditional gong sound.

Attack: attack portion of the volume envelope (ms) – longer values will cause the sound to fade in.

Release: release portion of the volume envelope (ms) – longer values will cause the sound to fade out over a longer time.

Smoothing: might also be classified as "legato". Tick this box to prevent a new gong tone from being sounded if it overlaps with previous tones – the tone will be continued, creating a bassy drone if used in combination with a long release time.

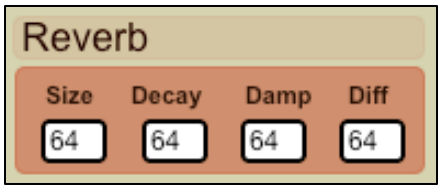
Rdel: set a random delay (ms) for each of the two constituent tones in addition to the delay specified in the **Rdelay** section.

F) Toggle parts

Note events for each instrument may be turned off; inactive instruments are coloured black in the main interface.

Reverb

Reverb can be added to each instrument through a pre-fader send.



Size: size of simulated room

Decay: length of reverb

Damp: damping – higher values will result in a brighter sound

Diff: diffusion – "breaking up" of the sound – higher values may result in a more natural sound.

Advanced / work in progress

Supplementary patches

Supplementary Max/MSP patches may be loaded from the file menu, allowing for expansion of the software. Patches featuring alternative sound generators may be used by choosing "no output" from the dropdown menu on the bottom-right hand corner of the performance window.

Supplementary patches available for this version of *Pipilan* include:

- **Slowdown.maxpat**: enables automatic modulation to the main BPM to provide a slow down into the final beat, either for every *gatra* or only *gatra* in which a gong stroke is present.
- **Timbre.maxpat**: provides access to the wavetables for oscillator A and B, along with tuning options for both oscillators.
- **Grid_edit.maxpat**: supplements the *kepatihan*-based editor with a more visually oriented grid interface.
- **midi_outputs.maxpat**: provides MIDI output with on a fixed scale based on approximations of the tuning of the *gender*.
- **Additional_gender_parts.maxpat**: enables patterns for the unused virtual hands on *gender* A and B, including mirroring of the *slenthem* part and *kempyang*/*kethuk* style patterns.
- **xf_rdelays.maxpat**: assigns crossfade parameter to instrumental random delays
- **balungan_wave.maxpat**: shared control for *balungan* generation and wavetable drawing.

Saving and reading files

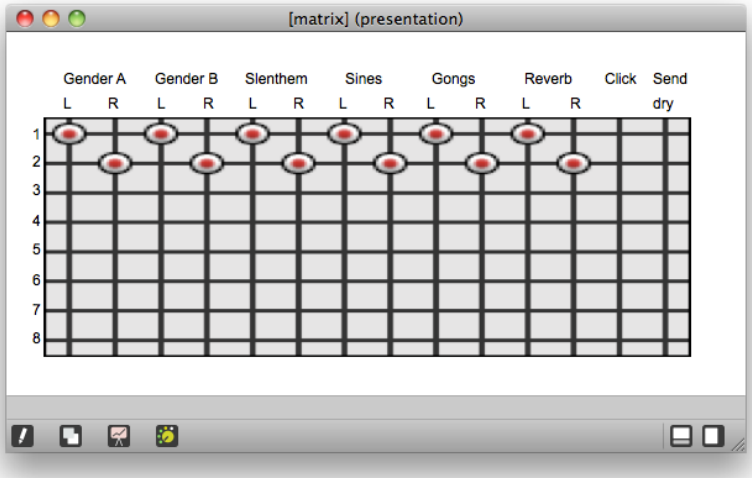
JSON files for *balungan* and *garap* may be loaded and saved from dedicated buttons in the editor window; the file menu is reserved for the supplementary patches described above.

Saving of the *balungan* is currently limited to a single *gongan*. The settings for the mixer and other devices for sound generation will not be saved upon leaving the software.

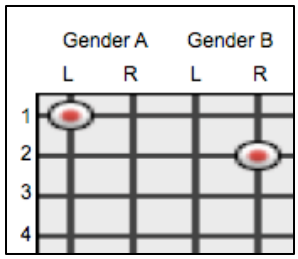
Routing matrix

The routing matrix is accessible from the options screen has been designed to send audio to multiple outputs of a soundcard or Soundflower for processing or multi-track recording. Place dots on the grid to make audio connections. Audio channels are represented vertically, while instruments are represented horizontally. Channels 1 and 2 represent the main stereo buss – sounds will not be recorded if they are not assigned here.

An additional two paths have been added specifically for multichannel output – a click track (intended for keeping time if recording live instruments) and a dry version of the effects send on each instrumental channel.



The matrix may also be used for panning sounds hard-left and hard-right (this functionality will be available in the mixer at a later date). In the following example, *gender* A and B are panned left and right respectively.



OSC control

All send/receive paths from the original Max/MSP patch may be accessed via OSC messages sent to port 2321. A full list of paths will be included in a future version.

Example commands:

```
/rate [integer]
/mainslider [0-127]
```

Third party objects and fonts

The sine wave synthesizer uses list-interpolate by Adrian Freed and Matt Write: http://cnmat.berkeley.edu/patch/4064

With the exception of some original JavaScript code everything within this software is based on objects bundled with Max/MSP. Reverb in the default *gender* setup is created via the *yafr2* object by Randy Jones.

Notation is based on the *KepatihanPro* font, which has been bundled for convenience. Please visit the American Gamelan Institute website – http://www.gamelan.org/library/ - fonts – for more information and an independent download link.

Known issues

Many timed events create an index out of range error (observable in the Max window by pressing *apple-M*). This is due to a change from the default behaviour of Max's dictionary objects from version 6.0 in which this software was initially developed, whereby unmatched keys were ignored. At the time of writing it is unclear whether this issue affects general performance.

Due to an unresolved bug it is not possible to set empty *gatra* (. . . .); upon doing so these sequences will be randomized automatically.