DJII

Pronounced like the letter *G*. A modular software instrument for supporting an improvising instrumentalist.

Dana Jessen and Ted Moore

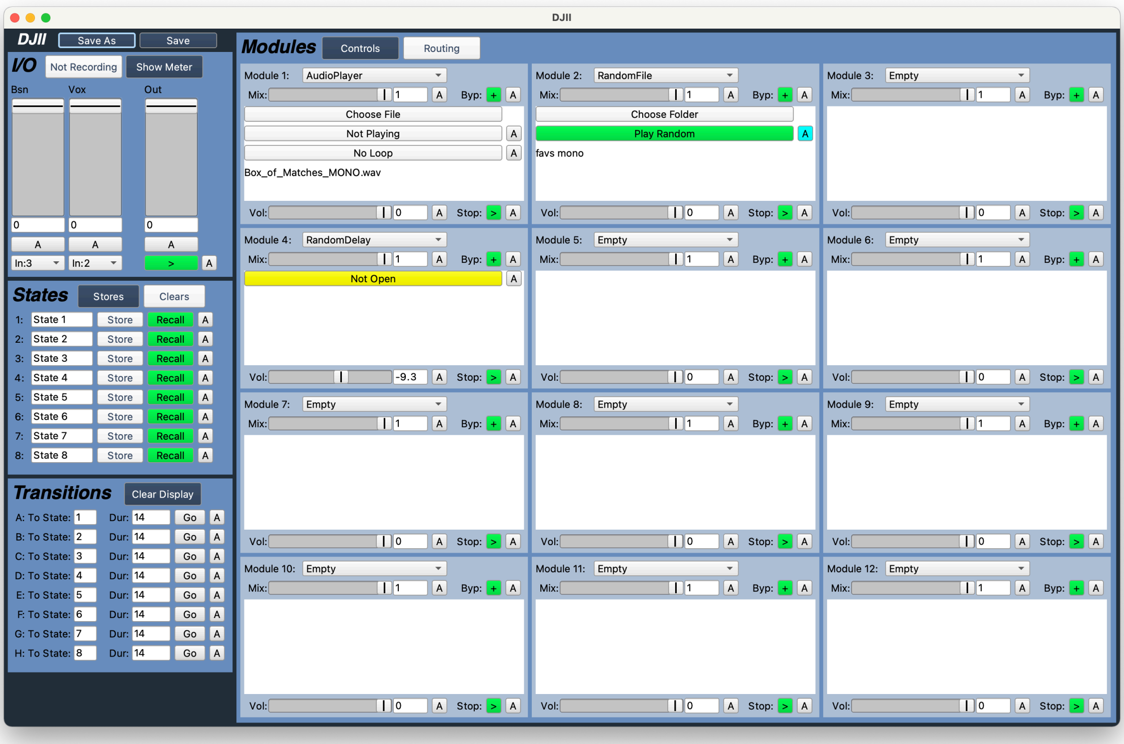
Contents

1 Introduction

Pronounced like the letter "G"

The *Dana Jessen Improvisation Instrument*, or *"G"* is designed by Dana Jessen and Ted Moore and coded by Ted Moore. The software is designed to be used by an improvising solo musician as supporting electronics material.

- The modular GUI design allows for intuitive exploration of sound possibilities including flexible matrix-based routing.
- Software-wide state save and recall enables performer navigation through different preplanned sections of a performance.
- A Transitions panel allows for time-variable, interpolated transitions between states.
- MIDI and QWERTY Learn functionality allows for all of the appropriate controls to be custom-mapped onto controllers accessible to the soloist during performance.
- Saving and loading the software allows the performer to prepare all of the above functionalities before the performance, which:
  - greatly reduces setup time at the gig,
  - allows a performer to work on multiple software configurations in parallel, possibly for different performances or contexts, and
  - allows a performer to rehearse with the same configuration of the software over time, developing an intuitive relationship with it and access to the musical expressivity that comes through practice.



2 Download the Latest Version

Download the latest version of the software from the Google Drive folder. The zip files are marked with a timestamp indicating when they were zipped. The timestamp is *YYMMDD_HHMMSS* meaning year, month, day, hour, minute, second respectively. For example February 21, 2022 at 8:04pm and 46 seconds will appear as: *220221_200446* (note that hours appear as a number out of 24). These timestamps are useful for knowing which version is the most recent and also for returning to past versions for any necessary troubleshooting or downgrading.

Unzip the zipped file and put that folder in SuperCollider's Extensions folder. You can easily open the Extensions folder by running this line of code inside SuperCollider:

```
Platform.userExtensionDir.openOS;
```

You can run this line of code by putting your text cursor on this line and hitting `command+return`.

You will also need to download four dependencies, all of which also need to go in the Extensions folder:

- sc3-plugins: http://supercollider.github.io/sc3-plugins/
- SelectFiles: https://github.com/tedmoore/SelectFiles
- EZConv: https://github.com/davidgranstrom/EZConv
- FluCoMa: https://github.com/flucoma/flucoma-sc

Once all these folders are inside the SuperCollider Extensions folder, you will need to reboot the SuperCollider Library. you can do this one of two ways: (1) close and reopen SuperCollider or (2) go to the `Language` menu at the top and choose `Recompile Class Library`.

3 Booting the Software

3.1 Setting the Audio Interface

In order to boot the software, you will run the bit of code that is found in the `DJII.scd` file included in the downloaded zip (below). First, you'll want to set the `~input` and `~output` variables equal to the audio interface you plan to use. If you're not sure what to put here, one good way to check what your options are is to boot the SuperCollider server (hit `command+b`) so it shows you what audio devices are connected in the post window (which is usually to the right side of the screen). You could then copy and paste the name of the device you want to use into the appropriate place in the code (set it = to `~input` and `~output`).

```
// open DJII:
( // command+return
s.options.sampleRate_(44100);

~input = "MacBook Pro Microphone";
```

```
~output = "MacBook Pro Speakers";
// ~output = "External Headphones";

DJII(s,~input,~output);
)
```

For performance, `~input` and `~output` will pretty much always be equal to the same device: the audio interface being used. Here they are offered separately in case one wants to play around with the software with just the laptop, in which case `~input` should equal `"MacBook Pro Microphone"` and `~output` can be either `"MacBook Pro Speakers"` or `"External Headphones"`. Make sure there semicolons at the end of the lines in which you're setting these devices equal to `~input` and `~output`. For convenience, one can keep the various devices that one might use in the code here by "commenting out" the ones not in use. Putting two forward slashes in front of the line of code with an option will make it a "comment" in the code, meaning that it won't be executed by SuperCollider, and therefore won't set anything equal to `~input` or `~output` (whichever is on that line.)

```
(
s.options.sampleRate_(44100);

~input = "Fireface UC Mac (24006457)";
~output = "Fireface UC Mac (24006457)";

// ~input = "MacBook Pro Microphone";
// ~output = "MacBook Pro Speakers";
// ~output = "External Headphones";

DJII(s,~input,~output);
)
```

Once the `~input` and `~output` devices are set, you can run this bit of code by putting your text cursor somewhere in this block and hitting `command+return`.

4 `.djii` Files

4.1 Opening a Saved `.djii` File

When the software is first booted up (by running the "bit of code" mentioned above) an open panel file dialog will appear. This gives the user the opportunity to load a saved `.djii` file. If the user clicks "Cancel" on this open panel, **DJII** will open in the default state, with no saved settings.

Alternatively, a user could use **DJII**'s `.load` class method to load a file, thereby bypassing the open panel file dialog. This class method call is similar to `.new` but includes a fourth argument: the path to the `.djii` file to load after boot. For example:

```
DJII.load(s,~input,~output,"/path/to/the/file.djii");
```

Both the `.new` and `.load` methods also have an optional `action` argument which can be passed a function that will execute after boot and load (if loading is to occur). This function will be passed the **DJII** instance as its only argument.

4.2 Saving `.djii` Files

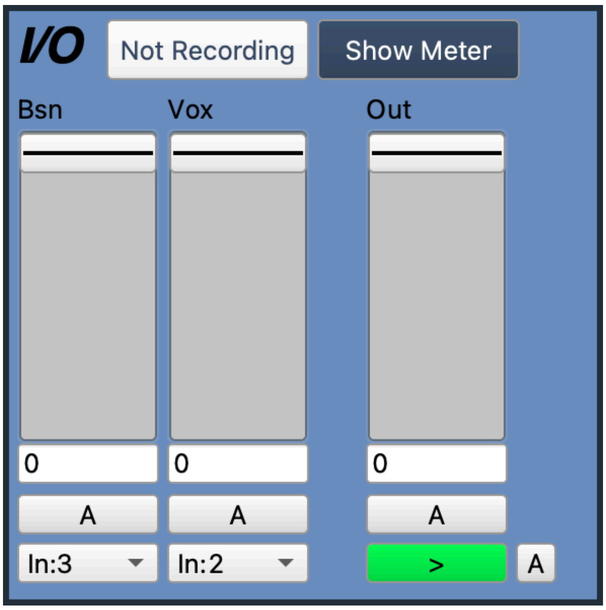
The state of the entire software (including controls *not* stored in the *States* Panel) can be saved in a `.djii` file stored anywhere on the computer.

Pressing the "Save As" button as the top left will open a save panel allowing the user to provide a location and name save the current state of the software. Pressing the "Save" button will overwrite whatever `.djii` file was opened when the software was booted (this is a permanent ov overwrite, so be careful!). If the "Save" button is pressed after the software was loaded in to the default state (no `.djii` file was selected), a open panel will allow the user to specify a name and location.

`.djii` File Internal Structure

This information is not necessary for using **DJII** or the `.djii` save files, but may be useful to know in some circumstances. The `.djii` files are constructed internally as a Dictionary of nested Dictionaries and Arrays. It is saved to disk using SuperCollider's `Object.writeArchive` method. Unfortunately, this makes them not "human readable" in a text editor, but they can be inspected (and even edited) by loading the file directly into SuperCollider with the `Object.readArchive` method. They can also be somewhat visually inspected in this way by calling `.gui` on the loaded Dictionary.

5 I/O Panel



5.1 Record Button

The Record Button has two states, "Not Recording" and "Recording," showing the current state of whether the software is recording. To start a recording, press the button while it shows "Not Recording" so that it changes to the state "Recording." To stop recording press the button while it is showing "Recording" so that it changes to the state "Not Recording." When recording is stopped three audio files will be put into a folder created within the SuperCollider recordings directory. The SuperCollider recordings directory can be found by executing this line of code in SuperCollider (hit shift `shift+return` when your cursor is on it):

```
Platform.recordingsDir
```

The directory will be shown in the post window (which is usually on the right side of the screen). Inside of this SuperCollider recordings directory will be a directory called "DJII" (if there isn't one there currently, SuperCollider will create it). Inside of the **DJII** directory will be a folder named with a timestamp (structured the same as the zip files). The folders are named this way so that if you do many recordings, perhaps over many days, these timestamped folders inside the "DJII" will appear chronologically when sorted "alphabetically."

Inside of one of these timestamped folders will be three files:

1. A mono file that of just the audio coming in the Bsn input (which ever hardware input is specified in the "Bsn" input drop down menu)
2. A mono file that of just the audio coming in the Vox input (which ever hardware input is specified in the "Vox" input drop down menu)
3. A stereo file of the master output

The names of these files also contain the timestamp of the folder they are in, an index number (`00`, `01`, and `02`) to keep them in "order," and the name of the recording, either `bsn`, `vox`, or `out`.

5.2 Show Meter Button

This button will open a new window with a VU meter showing the audio levels of the inputs and output for SuperCollider. This new, smaller window can be closed anytime and will not affect the rest of **DJII** at all. Pressing it more than once will just open multiple windows: they're all the same. This window may be useful for setting levels with the audio interface. One can look at these meters and turn the gain on the audio interface to an appropriate level. Notice that on the VU meter the channels are indexed from zero, so the four input channels are name 0, 1, 2, and 3, while the channel options in the Bsn and Vox input drop down menus are labeled 1, 2, 3, and 4. This inconsistency is unfortunate, but since the user's audio interface will *very* likely label the channels 1-4, the input selection drop down menus are intended to be consistent with that. Also, visually the VU meter is very clear which channel would be considered input "1-4" and output "1-4", except for the very small numbers at the bottom.

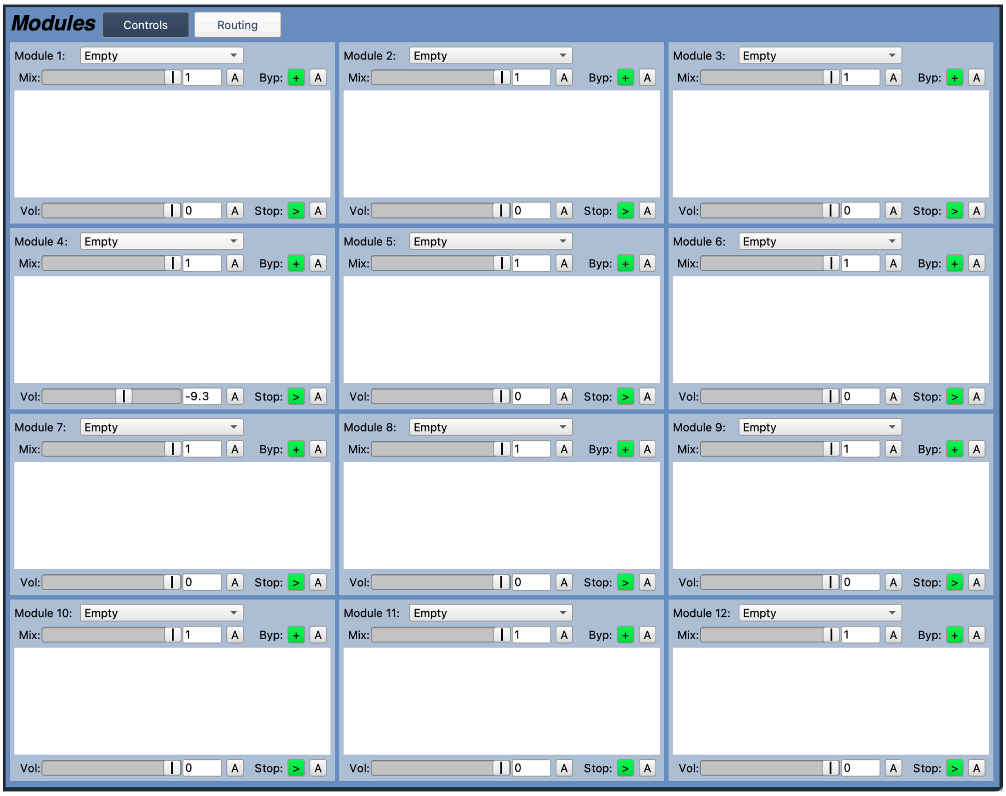
5.3 Bassoon and Voice Inputs

The Bassoon and Voice inputs each consist of a volume slider (in decibels), a number box showing the position of the slider, and a drop down menu for selecting which audio interface input is receiving input for that instrument. Learn about how to send these audio inputs to the output and modules in the Routing Matrix section. These volume sliders control the volume of the input signal before it is sent to the Routing Matrix. These sliders *are* controlled by the Transitions System.

5.4 Output Slider

This slider controls the master output volume (in decibels) it is the final manual volume control before the audio is sent to the audio interface for output. Also, *after* this volume slider **DJII** has a compressor to keep any unruly sounds from exploding a sound system or ear drums (hard knee, threshold = -10dB, ratio=4:1, attack = 10ms, release = 100ms). This slider is *not* controlled by the States Panel or Transitions Panel.

6 Modules Panel



6.1 Module Containers

By selecting the "Controls" button next to the "Modules" label the Modules Panel will display the 12 Module Containers. Each of these containers can hold any one of the Modules that appears in the drop down list. If the dropdown list says "Empty", no audio will flow through this container, even if audio is routed to it in the Routing Matrix. Except for the module selection drop down menu, all the controls in each module container will respond to changes in state executed in the States Panel or Transitions Panel.

6.1.1 Mix Slider

This slider controls the wet/dry mix of the container. If at 1 (completely wet), only the processed sound will be heard out of the container. If at 0 (completely dry), only the unprocessed sound (which is routed to the container in the Routing Matrix) will be heard. This may be useful for balancing the dry sound of an audio input (such as voice or bassoon) with an FX such as reverb.

6.1.2 Bypass Toggle

When green with a `[+]` sign, the module is not bypassed: it will output the sound of the module in the container (the Mix Slider still applies). When yellow with a `[0]` sign, the module is bypassed completely. If there is no input to this container, it will output silence. If there is audio input to this container, the input will be sent through to the output (the Mix Slider is bypassed and has no effect). The Volume Slider still applies to the output.

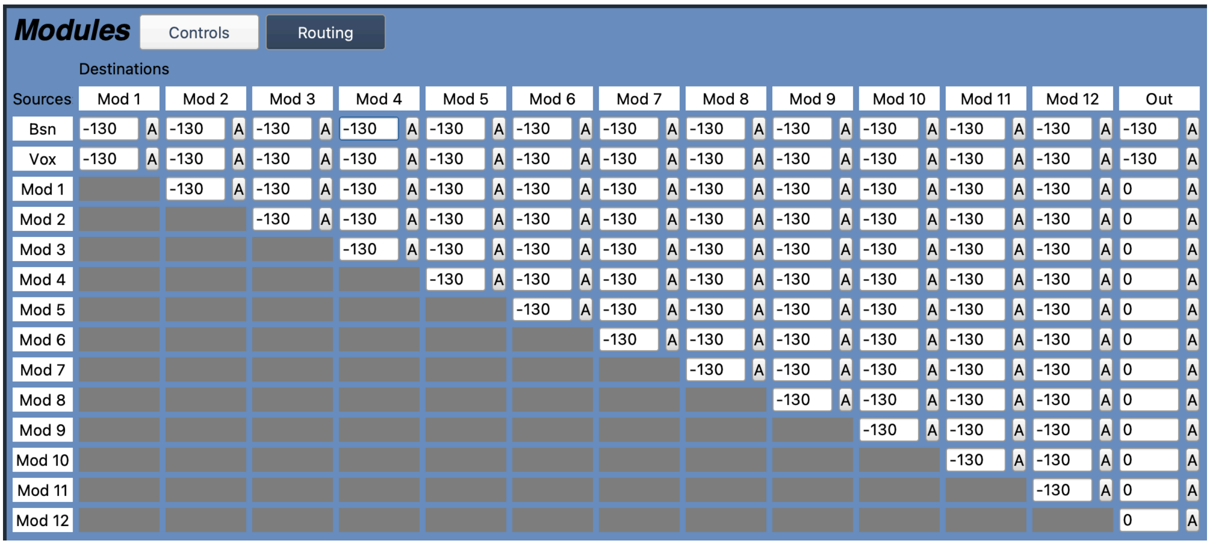
6.1.3 Volume Slider

Controls the volume (in dB) of the module output. This applies to the signal regardless of the Mix Slider and Bypass Toggle.

6.1.4 Stop Toggle

When green with a `[>]` arrow, the module's sound is routed to the output of the container. When red with an `[X]`, the container is "stopped" (muted), so no sound will come out. This includes any sound from the module, as well as any sound that might be routed from the container's input either through the module itself or via the Mix Slider.

6.2 Routing Matrix



Modules Controls Routing

Destinations

Sources	Mod 1	Mod 2	Mod 3	Mod 4	Mod 5	Mod 6	Mod 7	Mod 8	Mod 9	Mod 10	Mod 11	Mod 12	Out
Bsn	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A
Vox	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A
Mod 1		-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	0 A
Mod 2			-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	0 A
Mod 3				-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	0 A
Mod 4					-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	0 A
Mod 5						-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	0 A
Mod 6							-130 A	-130 A	-130 A	-130 A	-130 A	-130 A	0 A
Mod 7								-130 A	-130 A	-130 A	-130 A	-130 A	0 A
Mod 8									-130 A	-130 A	-130 A	-130 A	0 A
Mod 9										-130 A	-130 A	-130 A	0 A
Mod 10											-130 A	-130 A	0 A
Mod 11												-130 A	0 A
Mod 12													0 A

The Routing Matrix is used to send audio between different sources and destinations in the software. Along the left side is a list of the **Sources**: all the places that audio can *come from*. This includes not only the output of all the modules, but also the inputs for the Bassoon ("Bsn") and Voice ("Vox"). Along the top are all of the **Destinations**: any part of the software that can be *sent to*. This includes all of the Module Containers, as well as the master output.

In order to route audio from one place to another, begin by finding the row that corresponds to which **Source** is producing the audio you wish to route. Next, move across that row to the column that corresponds to the **Destination** you wish to send the audio to. The box at this intersection is a volume adjustment (in decibels full scale) for *how loud* the audio should be routed from the selected **Source** to the chosen **Destination**. 0 dB indicates "full volume": the audio will be routed to **Destination** at the full volume that it is coming out of it's Module Container (the Module Container's mix, volume, bypass, and stop controls are all applied *before* the Routing Matrix). -130 dB is essentially silent: no audio will be routed from the **Source** to the **Destination**.

If the box at the intersection is just a gray rectangle (such as the intersection of **Source**: Module 2 and **Destination**: Module 1), this means that the audio cannot be routed in that way.

All of the routing matrix volumes can be assigned to a MIDI handle for easier control using the `[A]` button, however, none of the volume values will be controlled by changes in the States Panel or Transitions Panel.

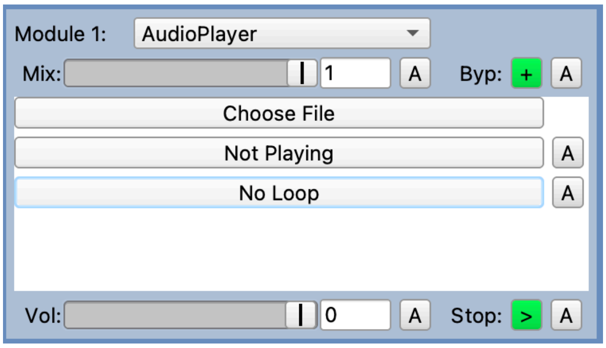
Notice (in the screenshot above) that the default routing is to have all the modules routed to the output (0 dB) but with no interconnections between modules. Also, both inputs (Bsn and Vox) are not routed anywhere by default.

7 Modules

7.1 AudioPlayer

Very simply choose an audio file and play, pause, or loop it.

- **Choose File** button will present an open panel file dialog for selecting an audio file.
  - It must be `.wav` or `.aiff`.
  - It cannot be more than 2 channels.
  - If a mono file is selected it will play centered in the stereo field.
- **Not Playing / Playing** toggle. Indicates whether the sound file is playing. When the sound file gets the end this will change from **Playing** back to **Not Playing** (as long as **Loop** is not on).
- **No Loop / Loop** toggle specifies what **AudioPlayer** should do when a playing sound reaches the end of the file. This can be set at any time, including while a sound is playing to determine what should happen at the end of the file. This might be useful if a sound file is looping and at some point (towards the end of a performance perhaps) you want the file to continue playing to the end of the file, but then *not* loop this time: you could change the toggle to **No Loop** while it's playing and then it will just stop the next time it gets to the end of the file.



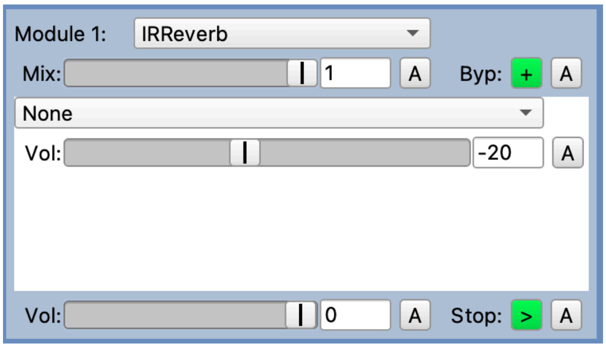
7.2 IRReverb

IRReverb stands for "Impulse Response Reverb", which is also known as Convolution Reverb. This reverb module uses "impulse response" recordings of real-world acoustic spaces to model or imitate what it might sound like to make sound in that space. These recordings are essentially `.wav` file recordings of an "impulse" created in the space (think balloon pop) or they're constructed from a sine tone sweep analysis made in the space.

The drop down menu offers many different options to choose from (including some more experimental ones such as cymbals and vases). Many of the options have an "L", "R", and "M-to-S" ("Left", "Right", and "Mid/Side"). These are all stereo files but the "L" and "R" are going to have a more pronounced reverb on that side, while the "M-to-S" files will have a more balanced stereo image.

Because the convolution reverb algorithm can produce unexpectedly loud volumes, **IRReverb** has a personal volume slider inside the module that defaults to -20 decibels (which will be an appropriate adjustment for most of the impulse response files to choose from).

In order to get an appropriate dry/wet reverb mix (for a live instrument perhaps), use the Mix parameter on the module's Container.



7.3 Pitch2Synth

Pitch2Synth plays fat/lush synth tones based on pitch analysis of the signal it receives. The three sliders control three thresholds:

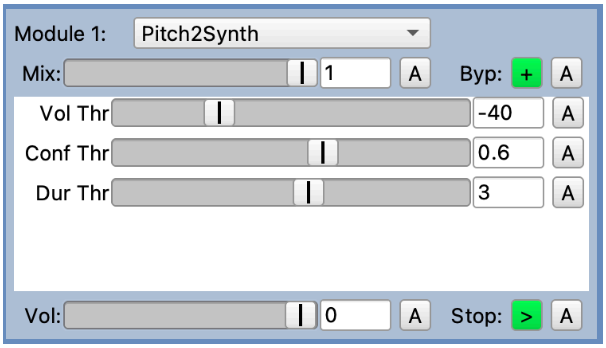
1. **Vol Thr:** Volume threshold measured in decibels.
2. **Conf Thr:** Pitch confidence threshold ranging from 0 to 1. This is an output of the Pitch detection algorithm that is used. It indicates the confidence the algorithm has *that there is* a pitch. It essentially is an indication of "noisy" (closer to 0) or "harmonic" (closer to 1) the sound is.
3. **Dur Thr:** Duration threshold measured in seconds.

In order for one of these synth tones to be triggered a few criteria must be met:

- The analysis of the incoming pitch must remain stable (within 1 semitone) for at least *duration threshold* seconds.
- The incoming audio must be above the *volume threshold* for at least *duration threshold* seconds.
- The pitch confidence must be above the *confidence threshold* for at least *duration threshold* seconds.

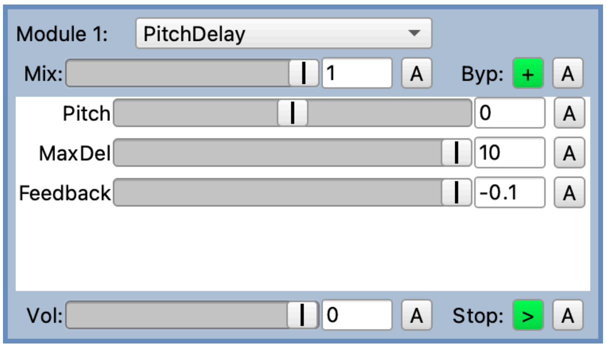
If all these criteria are met, a synth tone will be triggered. After a trigger occurs another will not occur for the next 6 seconds. Even if triggers are not being cause by analysis, there will be a few randomly triggered synth tones, roughly every 30 seconds *as long as* both the pitch confidence and volume are above their threshold (this check is done when one of the random triggers occur–determining if it will actually cause a synth tone to sound).

When a synth tone is triggered it uses weighted randomness to choose to transpose the pitch by some amount ([down 3 octaves, down 2 octaves, down 1 octave, down P4] with weights [0.25, 0.25, 0.375, 0.125] respectively). The synth tone fades in and out, lasting between 24 and 60 seconds.



7.4 PitchDelay

PitchDelay is a live processing module consisting of a granular-based pitch shifter followed by a cubic-interpolated delay, the output of which is fed back into the input of the module (causing pitch shifting / delay feedback routing). The pitch shift is granular-based with a grain size of 0.5 seconds. The time dispersion of the grains is randomly modulated (with a frequecy of 1) between 0.1 seconods and 0.5 seconods. The pitch shift is set by the slider which has a range of -24 semitones to +24 semitones. The delay time is randomly modulated (with a frequency of 0.05 Hz) between 0.01 seconds and the maximum delay time set by the slider. The volume of the output signal that is fed back into the input is specified with the "Feedback" slider, measured in decibels full-scale.



7.5 RandomDelay

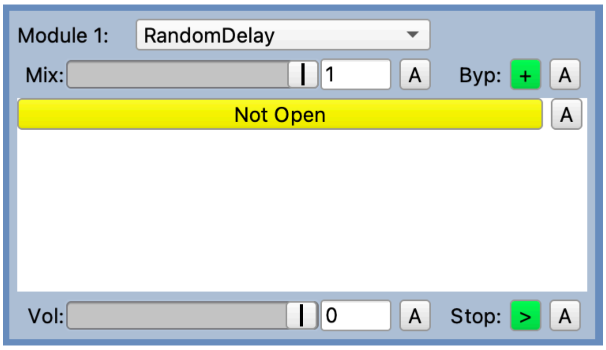
RandomDelay has a 60 second buffer for storing incoming audio. If the *Open* / *Not Open* toggle is *Open* then the incoming audio will be written into this buffer. If the toggle is *Not Open* then silence will be written into this buffer (RandomDelay is always *recording*, *Not Open* just means that the recording will writing silence into the buffer).

At the same time, this module is continuously playing random excerpts from this buffer. It triggers a new excerpt about 3 times per second and each excerpt is between 7 and 12 seconds long. At first (or anytime) while the buffer is mostly filled with silence, these excerpts will seem sparse, as most of the triggered excerpts are playing back silent parts of the buffer. After some time where this module has been able to fill most of the buffer with sound, most of the triggered excerpts will contain audio, filling out the texture provided by this module.

The playback rate of each excerpt is chosen randomly using probabilities as follows:

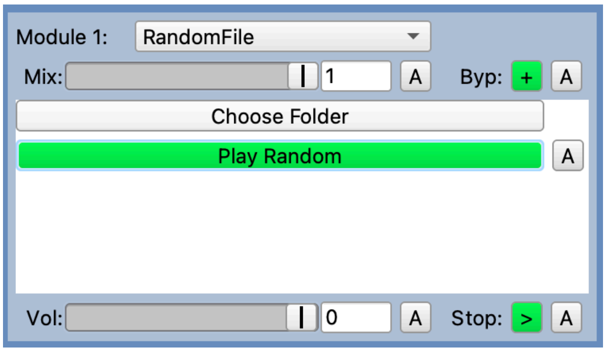
Pitch Shift (by playback rate)	Probability
down 2 ocatves	4/21
down 1 ocatve	6/21
down a justly tuned P4	2/21
no change in playback rate	6/21
up a justly tuned P5	1/21
up 1 octave	2/21

The recording buffer is mono but the excerpts play back at a random position in the stereo field.



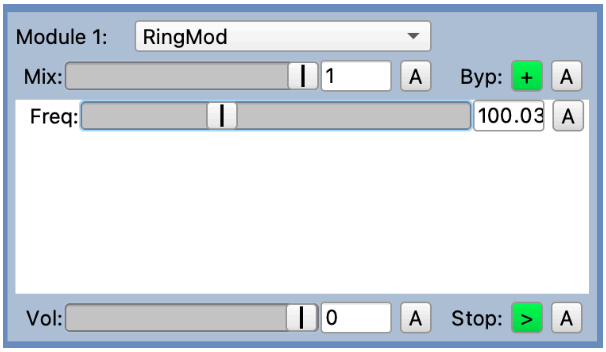
7.6 RandomFile

Clicking on the *Choose Folder* button will create an open dialog to select which folder of sound files this module should choose from. After loading a folder, pressing the *Play Random* button will choose a random file to playback, however the same file will never be chosen twice in a row. If *Play Random* is pressed while a previously chosen file is still playing, the playing file will continue to play and the new file will also start playing. All files must be either mono or stereo.



7.7 RingMod

Perform basic ring modulation on the input signal using a sine tone. The sine wave frequency can range between 20 Hz and 2,000 Hz. The frequency slider is controlled by the States Panel and Transitions Panel.

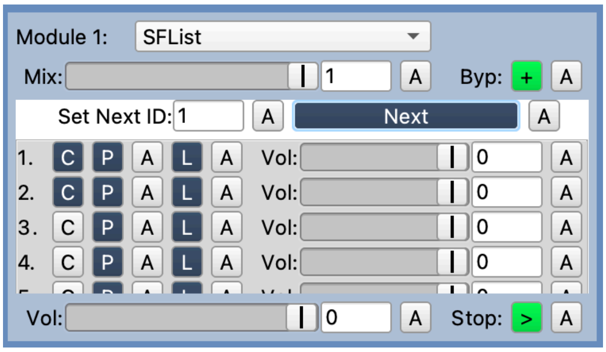


7.8 SFList

SFList (which stands for sound file list) enables pre-loading up to 20 different sound files that can either be played in sequence using the *Next* button or played individually. To the left of the next button, the *Set Next ID* number box shows what item in the list will play when the *Next* button is pushed. This updates automatically as the sequence of sound files is stepped through, providing visual feedback on what sound file the module has cued up. It can also be set by clicking in the box and typing the integer of the sound file you wish to play next. **The value in the *Set Next ID* number box is stored by the States Panel and Transitions Panel, so when storing a state, it may be very important to check what number this is set to.** *(I think that this may be a very undesirable feature, because it might be difficult to remember to check when setting states. Perhaps this number box should not be controlled by the States and Transitions panels.)*

In the list, each sound file has four controls:

- The [C] (for "Choose") button creates an open dialog for selecting which sound file should be held at the given number in the list. When a position in the list has a sound file assigned to it, its [C] button is blue. When there is no sound file assigned to a list position, the [C] button is gray.
- The [P] (for "Play") button will playback the sound file. While the sound file is playing the [P] button is green. When the sound file is not playing, the [P] button is blue. Pressing the button while the sound file is playing will stop playback.
- The [L] (for "Loop") button sets whether or not the sound file should loop when it gets to the end. When looping is on, the [L] button is green. When looping is off, the [L] button is blue. If loop is turned off while the sound file is playing, the file will continue to its end and then not loop.
- The *Vol* slider adjusts the volume (in dBFS) of the sound file at the given position in the list.



8 States Panel

The *States* Panel allows for storing eight software-wide states, or presets. All of the parameters in the software that have the [A] button next to it will be stored in these states, *except* the ones listed below. Whatever state the parameters are in when the button is pressed is what will be stored in that *State*.

Pressing the *Stores* or *Clears* button will toggle whether the panel is showing the *Store* buttons for saving the current state of the software into a *State* or the *Clear* buttons which will clear a state. If one of the eight states doesn't have anything saved to it, its *Clear* button will just be blank. If a state does have something stored in it, the *Clear* button will be yellow and say *Clear*.

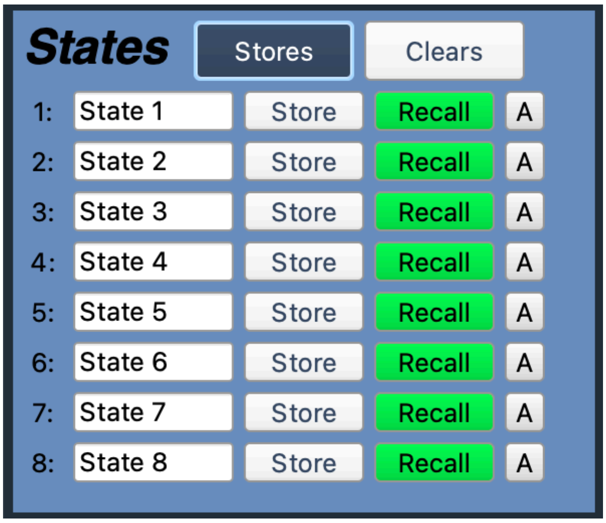
The state names can be edited from their default (such as "State 1"). This has no impact on the functionality of the software but may be useful for remembering what state of the software is stored there.

Just like all the other parameters with an [A] button next to it, the *Recall* buttons can be assigned a MIDI control so that these states can be switched between during live performance by using a MIDI controller. Note that the *Store* and *Clear* options for each state are not assignable as these operations are not intended to occur during live performance.

When a state is selected its number will appear (in large font) *over* the Transitions Panel indicating the current state. Hopefully this will be large enough to potentially be useful to a performer during performance.

Parameters *not* controlled by the *States* and *Transitions* Panels

- Master Volume
- Hardware Input selections for Bsn and Vox
- Module Container module selector drop down menus (this should not be being changed during performance!)
- All of the volumes in the Routing Matrix

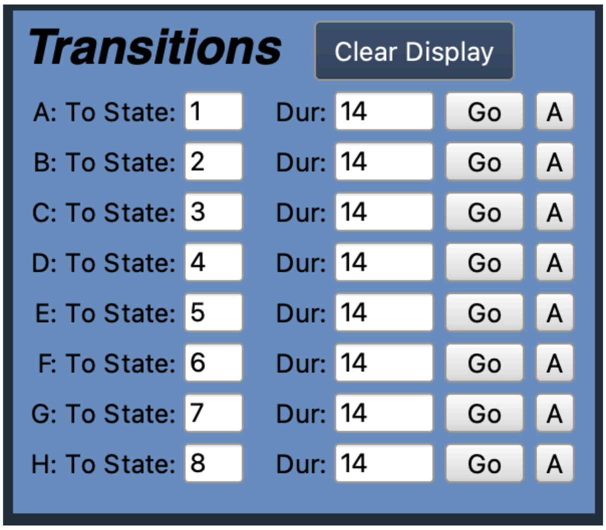


9 Transitions Panel

The *Transitions* Panel enables a users to trigger transitions of specified durations to a state set in the *States* Panel. For any of the transitions A through H, the user can specify which state (that is stored in the *States* Panel) to transition to and the duration in seconds the transition should take. Any continuous controls, such as sliders, number variables, etc., will use *Dur* seconds to transition from wherever they currently are *to* the value they were stored as in the specified state. Any toggles (such as the *Byp* and *Stop* buttons) will *immediately* jump to their position stored in the specified state (for this reason, using the *Mix* and *Vol* sliders to control different sonic spaces might be preferable).

During a transition the target state's number will fade in (in large font) over the Transitions Panel indicating the target and progress of the transition. If a previous state's number was displayed that will fade out.

Pressing the [Clear Display] button will clear any number that is showing over the Transitions Panel, allowing the controls to be accessed.



10 MIDI and QWERTY Learn

Every control in the software that has an [A] button next to it ("A" stands for "Assign") can be assigned to a MIDI control or QWERTY key.

Toggles and buttons are best *not* assigned to physical knobs, sliders, or continues foot pedals like expression pedals. Conversely, continuous controls (such as sliders or continuous number variables) are best *not* assigned to physical buttons or QWERTY keys.

To assign a control, click the [A] button and the press the physical button or wiggle the physical control that you want to assign. If successful, the [A] button will turn cyan. If not successful and you're trying to assign a physical control on a MIDI device, the MIDI device might not be successfully connected. Try checking the connection and rebooting DJII.

To unassign a control, hold shift and click on the cyan [A] button. The assignment will be nullified and the button will turn back to gray.