User Guide

BITWIG STUDIO 5.1

The content of this user guide is subject to change without notice and does not represent a commitment on the part of Bitwig. Furthermore, Bitwig doesn't take responsibility or liability for errors or inaccuracies that may appear in this user guide. This guide and the software described in this guide are subject to a license agreement and may be used and copied only in terms of this license agreement. No part of this publication may be copied, reproduced, edited or otherwise transmitted or recorded, for any purpose, without prior written permission by Bitwig.

This user guide was written by Dave Linnenbank.

Updated for Bitwig Studio version 5.1, November 2023.

Bitwig GmbH | Schwedter Str. 13 | 10119 Berlin - Germany

contact@bitwig.com | www.bitwig.com

0. Welcome to Bitwig Studio

Welcome to **Bitwig Studio**! We are glad you have joined us and are excited to help you create, compose, polish, and perform your music.

And welcome also to our **Bitwig Studio Producer** and **Bitwig Studio Essentials** users! Most of Bitwig Studio's functions and resources are available in all of our products so this user guide applies equally to all programs.

If you are reading this user guide as a web page, the table of contents along with a search function and language selector is available either on the right of this text or at the bottom of this page (hello, mobile interface). And if you are viewing the PDF version, use your program's normal features for browsing sections, searching, etc.

The purpose of this document is to walk you thru most of Bitwig Studio's functions and show you how to operate the program. The chapters and topics are arranged progressively, with basic concepts appearing first and advanced ideas showing up later. And although this document does not attempt to explain fundamental audio and musical concepts, it is written for users of any stripe who want to use software to make music.

In addition to this document, other resources will be mentioned when appropriate, and you can always visit Bitwig's website [http://bitwig.com] for the latest information. And please share any feedback you have or issues you encounter by visiting our support portal [http://bitwig.com/support].

In this chapter, we will begin with links to sections that have changed in this version. We will move on to the **Dashboard**, which is more or less the command center of Bitwig Studio. Finally, we outline a few conventions that will be used across this document. But you will not make sound in this chapter; that is what the rest of this document is for.

0.1. What's New in Bitwig Studio v5.1

For those of you who are recent Bitwig users, hello! Here are some pointers to new and changed sections of this document. New features and some of the updates in Bitwig Studio v5.1 include:

10 New Filters & Waveshapers

› *New Grid/**Polymer**/**Filter+**/**Sweep** module: **Fizz** (Filter)*, a modern *Character* filter for spreading harmonic nodes around (see section 19.28.9.8).

› *New Grid/**Polymer**/**Filter+**/**Sweep** module: **Rasp** (Filter)*, a modern *Character* filter that can scream or whisper (see section 19.28.9.9).

› *New Grid/**Polymer**/**Filter+**/**Sweep** module: **Ripple** (Filter)*, a modern *Character* filter with hyper-resonance (see section 19.28.9.10).

› *New Grid/**Polymer**/**Filter+**/**Sweep** module: **Vowels** (Filter)*, an *Inspired* filter that produces vowel sounds (see section 19.28.9.7).

› *New Grid/**Filter+**/**Sweep** module: **Push** (Waveshaper)*, a *Character* soft clipper with a detailed curve (see section 19.28.10.10).

› *New Grid/**Filter+**/**Sweep** module: **Heat** (Waveshaper)*, a *Character* S-shaped clipper that starts soft but can drive hard (see section 19.28.10.11).

› *New Grid/**Filter+**/**Sweep** module: **Soar** (Waveshaper)*, a *Character* soft wave folder that makes the quietest parts loud (see section 19.28.10.12).

› *New Grid/**Filter+**/**Sweep** module: **Howl** (Waveshaper)*, a *Character* wave folder that puts different parts of the signal into loud focus (see section 19.28.10.13).

› *New Grid/**Filter+**/**Sweep** module: **Shred** (Waveshaper)*, a *Character* non-linear wave folder for subtle cancellation or big-time artifacts (see section 19.28.10.14).

› *New Grid/**Filter+**/**Sweep** module: **Diode** (Waveshaper)*, a *Parametric* shaper modeling the classic circuit in a modern way (see section 19.28.10.6).

Two New Grid-powered Audio Effects

› *New audio effect: **Filter+** (Filter)*, a dead-simple FX box, for deploying any waveshaper and filter from *The Grid* directly onto a track (see section 19.10.2).

› *New audio effect: **Sweep** (Filter)*, a performable effect device, combining and blending a waveshaper and two filters from *The Grid* (see section 19.10.6).

Voice Stacking Expansion

› *Voice Stacking can now provide up to 16 voices*, a creative approach to sound design that is also available to certain audio effects, note effects, and CLAP plug-ins (see section 16.2.5).

› *Individual voices can be soloed*, for clear and easy sound programming (see section 16.2.5).

› *The old **Voice Stack** modulator has been replaced*, with two new, focused modulators taking over and expanding its functions in the new *Voice Stacking* category:

*New modulator: **Stack Spread** (Voice Stacking)*, offering 12 spread modes including standard splits, special distributions, and randomize options (see section 19.27.8.1).

*New modulator: **Voice Control** (Voice Stacking)*, for direct adjustment of any of the up to 16 voices within a voice stack (see section 19.27.8.2).

*Projects using the old **Voice Stack** modulator open with the new modulator(s) instead*, preserving modulation connections and all.

› *Three new Grid modules have been added*, offering direct access to individual **Voice Stacking** signals and more:

*New Grid module: **Voice Stack Info** (I/O)*, provides both the polyphonic *Voice Stack Index* for creating your own spread functions, and the current *Voice Stack Size* (see section 19.28.1.24).

*New Grid module: **Voice Stack Mix** (Mix)*, a modulatable processor with standard mix controls for each voice in the stack, at any point within a patch (see section 19.28.12.14).

*New Grid module: **Voice Stack Tog** (Mix)*, a modulatable processor to toggle the signal for each voice in the stack, at any point within a patch (see section 19.28.12.15).

Audio Quantize and other Onset Controls

› *Audio Quantize is now available*, a high-level function that creates beat markers based on certain onsets and slides them toward the beat grid (see section 10.2.1.7).

› *Onset visualization and threshold controls are now in several function dialogs*, including *Quantize Audio*, *Slice In Place* (see section 10.2.1.7), *Slice to Multisample*, and *Slice to Drum Machine* (see section 13.3.1).

› *Onset-aware stretch algorithms now have an Onset Intensity Threshold*, letting you exclude weaker onsets from affecting playback (see section 10.2.1.2).

Mixer Improvements

› *Channel strips have been improved*, both in the mixer and other interfaces (see section 7.1.8).

› *Track headers have been redesigned*, showing connecting various layers visually and providing more color (see section 7.1.1).

› *Devices with an **Expanded Device View** can now be opened* from various mixer and **Inspector Panel** interfaces (see section 7.1.5).

› *Mixer panels become scrollable when possible* while still adapting to the available space as necessary (see section 7.1).

Other new things include:

› *Creation and support of DAWproject file format*, a modern interchange format for taking project files from one music software to another (see section 14.7).

› *New Grid/**Polymer** module: **Bite** (Oscillator)*, a *Techniques*-driven oscillator, offering exponential FM, hard sync, PWM, and ring mod from dual oscillator feedback (see section 19.28.5.8).

› *Make Legato function now takes chords into account*, treating notes that start within a short time of one another as entities that should be extended together (see section 11.2.2.5).

› *Updated **FX Grid** (The Grid) device:* now has an *Auto-gate* option (and accompanying *Auto-gate Release Time* parameter) to make an **FX Grid** patch note-activated without editing the patch (see section 17.3.4.1).

*Auto-gate is on by default for **FX Grid***, as well as in the Grid-powered **Filter+** and **Sweep** (Filter) devices.

› *New Grid module: **Toggle In** (Mix)*, a switch between two incoming signals, with a button directly on the module (see section 19.28.12.5).

› *New Grid module: **Toggle Out** (Mix)*, a switch between two outgoing paths, with a button directly on the module (see section 19.28.12.6).

› *New Grid module: **Pitch Buss** (Pitch)*, a pitch summing buss with semitone-based attenuators, for up to six inputs (see section 19.28.14.7).

› *New Grid module: **Invert** (Math)*, gives a button to reverse polarity (× -1) of the incoming signal, with *Stereo-ness* option (see section 19.28.15.2).

› *New Grid module: **Reciprocal** (Math)*, gives a button to flip (1/x) the incoming signal, with *Stereo-ness* option (see section 19.28.15.3).

0.2. The Dashboard

Once you have Bitwig Studio installed and launched, the first place you will land is a place you will return to again and again. The **Dashboard** is a central hub for finding your projects, configuring your settings, managing library content, and accessing help. Each of these four tasks

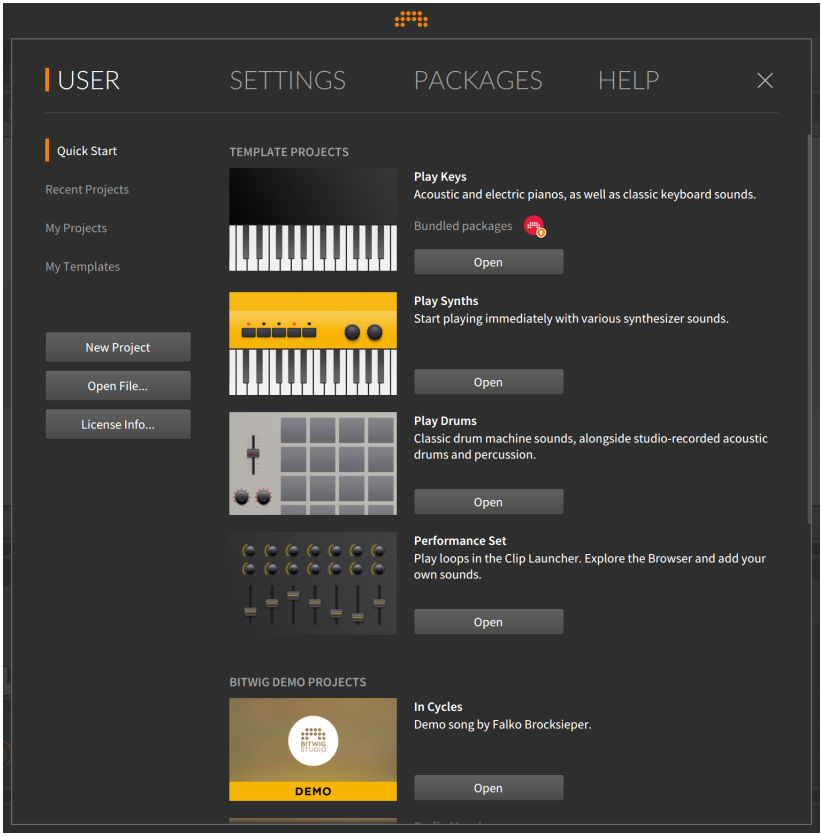
has its own tab for navigation, and we will walk thru each of them in turn in the following sections.

Note

If Bitwig Studio opens to a different view, you can call up the **Dashboard** at any time by clicking the Bitwig logo in the center of the window's header, at the very top of the screen.

0.2.1. User Tab

We call the first tab of the **Dashboard** the *user tab* because it displays the name you have registered with Bitwig. (If your username is too long, it will simply display *User*.)



The *Quick Start* page shows both *Template Projects* (that work as starting points) and demo project made either by Bitwig (found under *Bitwig Demo Projects*) and our partners (under *Partner Demo Projects*). Each demo project provides a short write-up, a list of any *Bundled packages* that are required to run it, and an *Open* button. Clicking *Open* downloads the project along with any used packages (which requires an internet connection), and then opens the project.

The next three pages show local content and are similar in format. The *Recent Projects* page shows the Bitwig Studio projects you have opened lately. The *My Projects* page displays all projects found in the *My Projects* path (which is defined in the *Settings* tab within the *Locations* page), and the *My Templates* page shows any template projects that you have saved.

Each of these three pages shows content in the same way. A search bar is provided at the top of the project list for winnowing down the projects being shown. When a project is selected (by single-clicking it), project information is display at the bottom of the window. This includes entries such as the last modification time and the file path to the project folder.

To open a listed project: either click the respective *Open* button or double-click the project name.

Finally, every page under the user tab shares three buttons on the middle left:

› *New Project* creates a blank project to let you begin working from scratch.

› *Open File…* provides a standard open dialog, in case you prefer locating a project that way.

› *License Info…* opens a window that displays your local license data and provides an option for registering a new serial number.

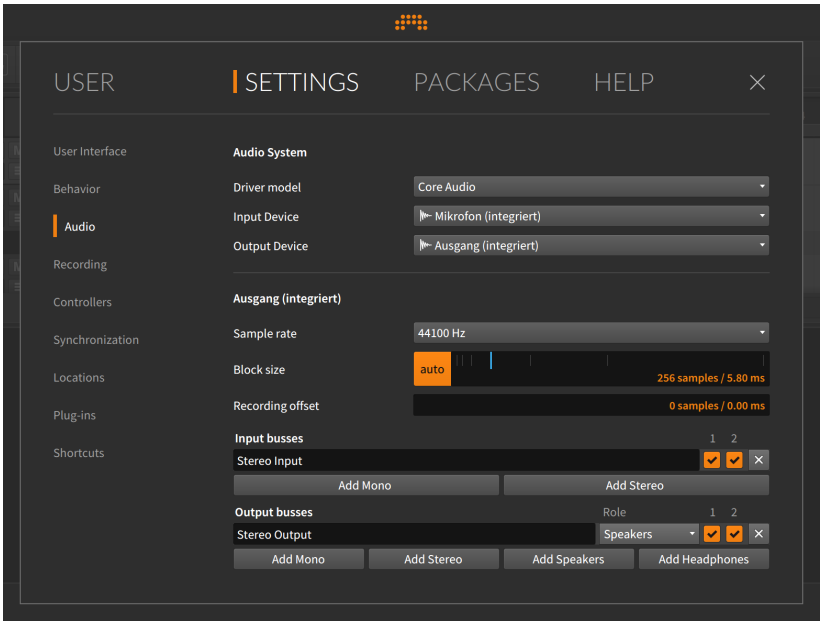
Because exiting the **Dashboard** requires that you have a project file open, trying to leave the **Dashboard** with no project open will send you to the *User* tab. The *New Project* button politely flashes in this case, indicating the quickest way to exit the **Dashboard** and get to work.

0.2.2. Settings Tab

The *Settings tab* is where Bitwig Studio's preferences generally live. We will look at a few of these pages in detail and then take the rest in the order that they appear.

0.2.2.1. Audio Settings

The *Audio* page defines a number of important settings for audio operation, including defining your audio interface and its inputs and outputs, as well as details such as the *Sample rate* and *Block size*.



To configure your audio hardware for the first time, begin by selecting the proper *Audio System* for your interface. The options available here vary based on your platform. If you are unsure of what to set, try the first option available (there may be only one option).

The *Input Device* and *Output Device* settings specify which audio interface you will be using for bringing audio signals into and out of the system, respectively. Whether you plan on using audio input or not, you must set the *Output Device* in order to hear anything out of Bitwig Studio.

Once the *Output Device* is selected, a section of the window is added with the same name. (In the image above, the driver's output device is named *Ausgang (integriert)* so a section also labeled *Ausgang (integriert)* follows.) Bitwig Studio will have created a stereo output pair that is mapped to the first two audio outputs of your interface. In the example shown above, the stereo output created by Bitwig Studio was named *Stereo Output* and is shown under the *Output busses* header.

Note

Names defined in the *Output Busses* and *Input Busses* sections will be used across Bitwig Studio to indicate audio routings. These names can be changed here at any time.

For more information, see section 7.3.2.

The *Output Device* selected in our case above has only two available audio outputs, and both of those are being used by *Stereo Output*, as indicated by the checked boxes labeled *1* and *2*. The fact that both boxes are checked means that they are being used for the *Stereo Output* path, which will be available in the program under that name.

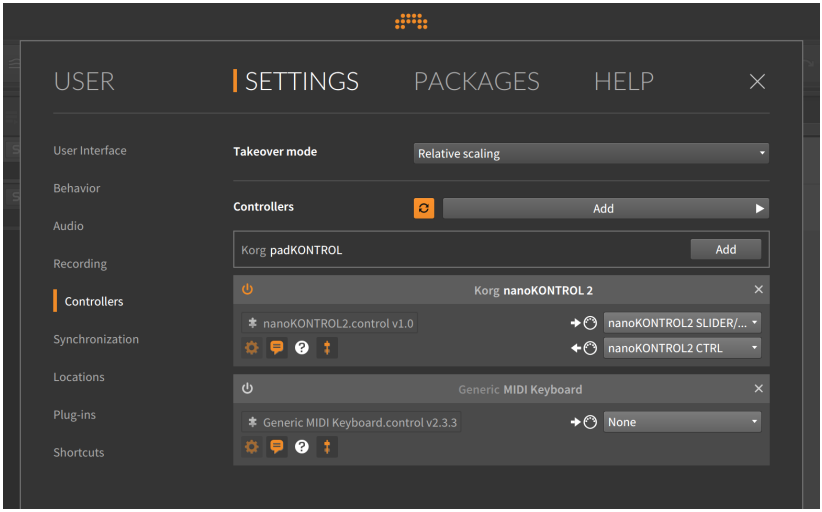
Finally, each output path has an assignable *Role*. The *Stereo Output* path has been defined as *Speakers*, making it an option for audio monitoring. The other *Role* settings are *Headphones* (also a monitoring option) and *Output*, which covers anything other than speakers or headphones.

If an *Input Device* is selected, a *Stereo Input* will be similarly created from the first two inputs.

Finally, the *x* button at the far right of each listed buss will delete that path. So if you create a buss by mistake, just click this button.

0.2.2.2. Controllers Settings

The *Controllers* page allows you to designate and configure any MIDI controllers that you will be using with Bitwig Studio.



The global *Takeover mode* setting determines how individual controls and their associate software parameters interact before their values match. Options include:

› *Immediate*, which fully applies any control message to its software parameter, moving it immediately.

› *Catch*, which waits to move the software parameter until the control message matches or passes the current parameter value.

› *Relative scaling*, which moves the software parameter incrementally in the same direction that the control is moving (for example, turning a knob up increases the parameter value, while turning a knob down decreases the value). This creates a relative motion based on your control gestures that will gradually meet the parameter value.

In the *Controllers* sections, the top row represents ways of adding controllers to your setup. The toggle with circular arrows represents auto-add mode. Enabled by default, this mode will automatically add any detected controller to your Bitwig Studio setup if a device-specific controller extension is also found.

The *Add* button allows you to add controllers manually. Clicking it calls up a menu of various controller manufacturers, each containing a submenu of models. If you do not find your device here, you can choose the top menu item, labeled *Generic*, and select the model best approximating your controller. Choices include:

› *Keyboard + 8 Device Knobs (CC 20-27)*, which is useful for a device with eight controllers that use continuous controller (CC) numbers 20 thru 27. These CCs are then used for soft control mappings.

› *MIDI Keyboard*, which is useful for a keyboard controller that you plan to use as a note input device. When specifying the source of MIDI/note messages via an input chooser, you can select all incoming MIDI channels (the default), or you can specify one MIDI channel to listen to.

As shown in the above image with the *Korg padKontrol* entry, you may see one or more unfilled rectangles with an *Add* button at the right. These entries appear when a controller that was previously setup and then manually deleted has been recognized by the computer. Since auto-add is not available in these cases, the manual *Add* button is here to let you quickly restore the device.

Below this top line are entries for individual controllers that are configured, usually named in their title bar with the controller manufacturer and the name of the extension (often matching the controller model). The "power" toggle at the title bar's left edge allows you to disable messages from the controller and extension without removing it. And the *x* icon at the right is for deleting the controller altogether.

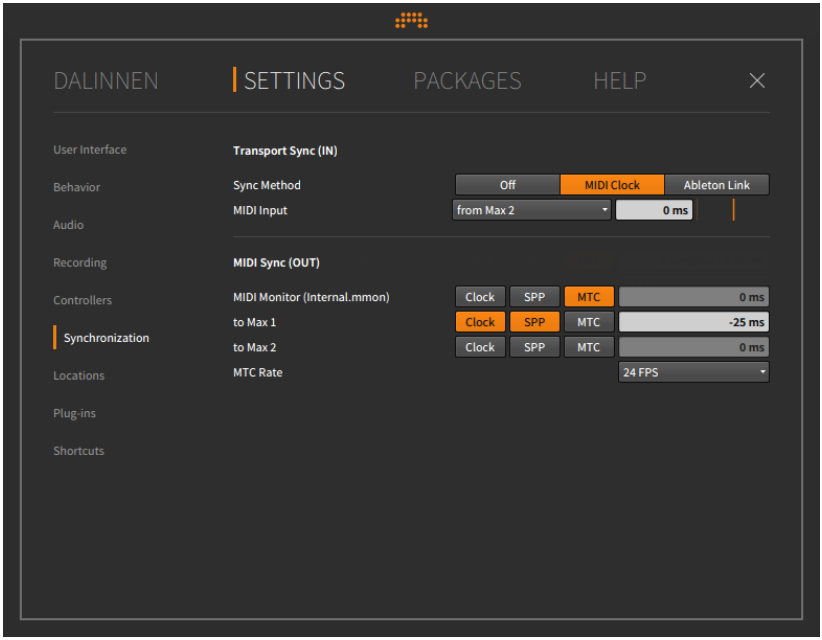
Just below the title bar is a puzzle piece icon with the name of the controller extension (or extension) following it. In the case that you have multiple extensions on your computer that work with this controller, this line becomes a menu, allowing you to swap one extension for another.

On the right side of each entry are menus for MIDI input and output ports (respectively) that the controller extension requires. If a device has gone offline or been disconnected, these ports may need to be set again before the power toggle can be enabled.

Finally, the bottom left of each entry contains a row of buttons related to the controller's performance (see section 15.2).

0.2.2.3. Synchronization Settings

The *Synchronization* page provides options for both controlling Bitwig Studio from external sources and for transmitting messages to synchronize other platforms/hardware to Bitwig.



The *Transport Sync (IN)* section allows you to select the *Sync Method* in use. The following three options are available:

› Bitwig Studio's *Internal* mode keeps the program's clock and transport independent from the outside world.

› The *MIDI Clock* mode synchronizes Bitwig Studio's clock to incoming MIDI clock messages from a selected *MIDI Input* port. For better synchronization, the *MIDI Input* signal can be shifted positively (to play a bit earlier) or negatively (pushing it later, into the future) in milliseconds.

Additionally, a vertical orange slider at the far right sets the responsiveness of Bitwig Studio to incoming tempo changes. Moving the slider to the left results in a quicker response to new tempo messages. Moving the slider to the right results is a more gentle response, which can be helpful when the tempo is largely static or the hardware in question is resulting in jittery behavior.

› *Ableton Link* connects Bitwig Studio to any and all other programs and devices on your local network that use Ableton's *Link* technology. (Compatible software running on your own machine alongside Bitwig Studio will be automatically found as well and can be synchronized in the same fashion.)

Note

A list of applications and devices that support Link can be found on this web page [https://www.ableton.com/en/link/apps/]. For additional information and support for these other products, visit the appropriate manufacturer's website or support center.

Link acts as a global time keeper, keeping track of and sharing the latest tempo and relative bar position for all "participants" (each application and device) in a "Link session." The rules are fairly simple:

1. When a new participant joins a Link session, its local tempo will automatically be set to the Link session's current tempo.
2. When a participant's transport is started, playback will wait until the Link session's relative bar position matches the participant's starting point. So if you hit play on a participant's transport from the top of bar one, the transport will wait for the Link session to arrive at the beginning of the next bar, thereby keeping everyone in relative sync.
3. When the tempo of any participant changes, the Link session's tempo is updated, and each participant's local tempo is automatically changed as well.

Note

A general troubleshooting Q&A on Link can be found on this web page [https://help.ableton.com/hc/en-us/articles/209073069-Link-Troubleshooting] from Ableton.

Finally, both the *MIDI Clock* and *Ableton Link* options add a dedicated button to the Bitwig Studio window, between the transport and display sections of the menu/transport area (see section 2.3). These buttons allow you to toggle the selected sync method on and off on the fly, and the Link button also reflects the number of other participants in the current Link session.

The *MIDI Sync (OUT)* section lets you set for each output path whether to:

› *Enable MIDI Clock* (the time clock icon)

› *Enable MIDI Clock Start/Stop messages* (the play triangle icon; available if MIDI clock is enabled)

› *Always send MIDI Clock*, even when the transport is stopped (the lock icon; available if MIDI clock is enabled)

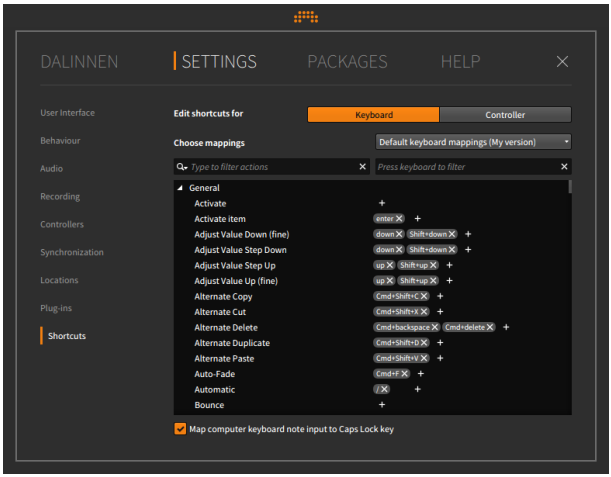
› Enable *SPP* (MIDI song position pointers; available if MIDI clock is enabled)

› Enable *MTC* (MIDI timecode)

And similar to the *MIDI Input* offset value, a *MIDI Out Clock Offset* can be set to fine tune each outgoing path separately. And a global setting for the *MTC Rate* can be set here as well.

0.2.2.4. Shortcuts Settings

The *Shortcuts* page allows the reconfiguration of Bitwig Studio's keyboard commands and the use of MIDI controller mappings to trigger these commands.



On this page, you can *Edit shortcuts for* both the computer *Keyboard* and via MIDI *Controller*.

To define a command mapping: locate the command you wish to map, and then click the *+* button to the far right of the command. You will then be prompted to trigger the desired mapping.

As can be seen in the image above, multiple mappings can be defined for each command.

To remove a command mapping: click the *x* button at the right of the mapping.

Once settings have been adjusted, the *Choose mappings* menu becomes a text entry box where new mapping sets can be named and a *Save* button appears.

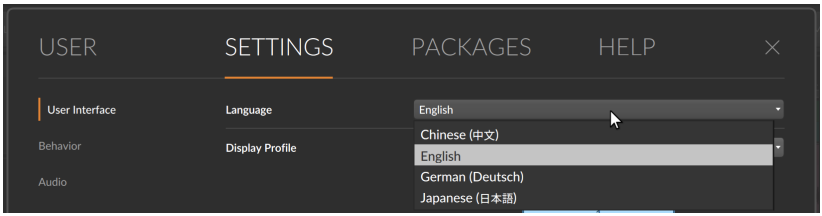
Note

When this manual refers to keyboard shortcuts, it is referencing the program's default shortcuts. Once you begin using your own shortcuts, the shortcuts in this document may be inaccurate for your use.

0.2.2.5. Other Settings

All other pages of the Settings tab are listed here in order.

› *User Interface* houses settings that visually alter Bitwig Studio. This starts with the *Language* chooser.



Device and parameters are still shown with their proper names, but most functions, labels, and **Interactive Help** (for the 300+ devices and modules) are translated to the language selected.

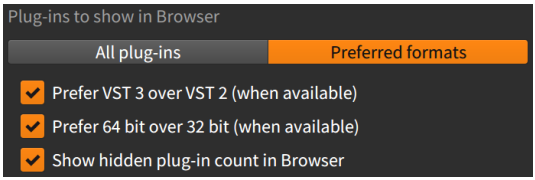
This page also includes the selected *Display Profile*, the program's *Scaling* level for each display in use, *Contrast* settings for getting the interface to look its best, the *Playhead follow mode* for how the window scrolls, and whether timeline audio's *Waveform display* is shown on a *Perceptual* scale or not.

› *Behavior* contains general settings, such as what to *Open on start*, whether a *Template* project should be used whenever you create a new project, and other editing and cache options.

› *Recording* provides general *Recording* settings, what type of tracks *Auto-Arm* when selected, the amount of *Pre-Roll* use (and whether the metronome is activated for that period), and the amount (if any) of *Record Quantization* to be used on notes.

› *Locations* defines several paths for Bitwig Studio, such as where *My Projects* live, where *My Library* is stored, where *My Controller Scripts* should be saved, and a number of other locations for the browsers to use.

The section for *Plug-in Locations* includes folders to be scanned for valid audio plug-ins, but it also contains preferences for which format(s) should be displayed when a plug-in is located in multiple formats.



When you have selected to show *All plug-ins*, the following options are irrelevant and dimmed. When *Preferred formats* is selected, the options below take effect:

Prefer CLAP over VST (when available) - When both a CLAP and VST version of the same plug-in are found (and can be matched), this option will hide the VST version by default.

Prefer VST 3 over VST 2 (when available) - When both a VST 3 and VST 2 version of the same plug-in are found, this option will hide the VST 2 version by default.

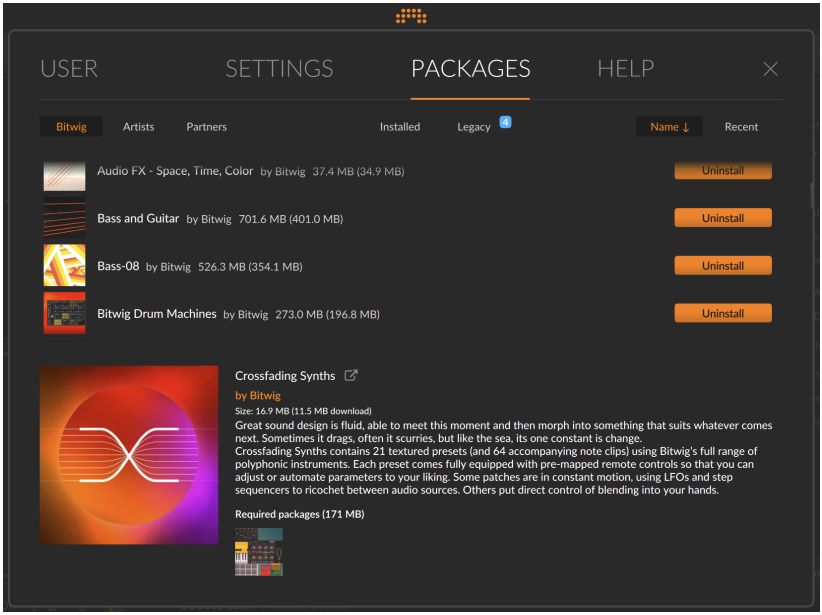
Prefer 64 bit over 32 bit (when available) - When both a 64-bit and 32-bit version of the same plug-in are found, this option will hide the 32-bit version by default.

Prefer native over emulated Intel on Rosetta (when available) - For Mac ARM, when both a native ARM and Intel version of the same plug-in are found, this option will hide the Intel version by default.

› *Plug-ins* provides options for how third party audio plug-ins are shown and handled. For more information, see section 16.3.

0.2.3. Packages Tab

The *Packages tab* is where supported library contents can be managed, downloaded, and updated from Bitwig.



Click on any package results in additional information popping up, as seen above. Otherwise, the top row of text buttons represents view filters for seeing and sorting packs differently.

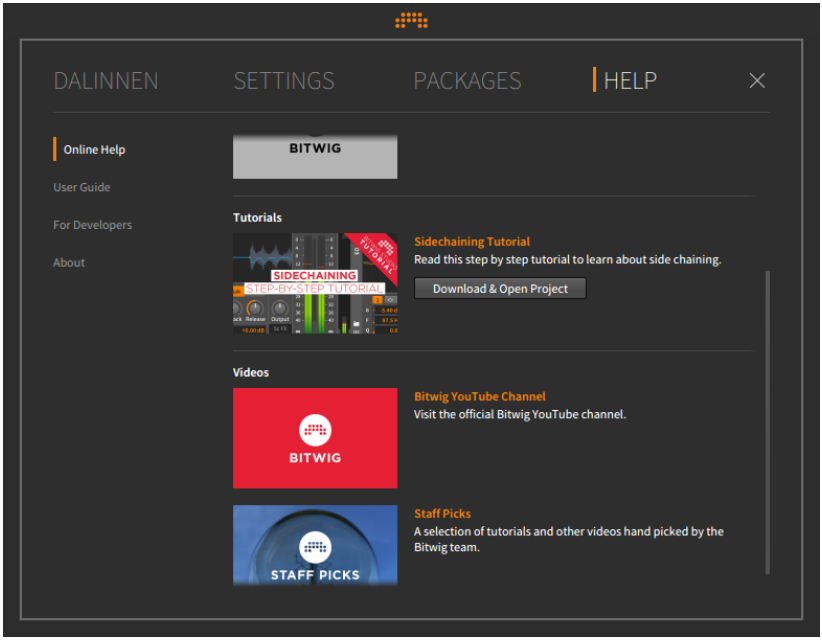
The first group of buttons offers to filter packages by their source, either by showing only those by *Bitwig*, only those by *Artists*, or only those from *Partners* (like sound design companies, etc.). Or simply turn off this filter to see packages from all sources.

The second group of buttons offers to filter packages by their status within you library, by either showing only the packages you have already *Installed* (meaning their content is available to use), or just showing the packages that aren't installed but are *Available*. Or, again, simply turn off this filter to see all packages in the list below.

Finally, the third group offers sort options. One option is to sort packages alphabetically with the *Name* ↓ button. Or choose to sort packages based on their release date with the *Recent* button.

0.2.4. Help Tab

The *Help tab* provides links to documentation and resources both within the application package and online.



Again, several pages exist within this tab:

› *Online Help* offers information about various resources, as well as either links to the online content or the option to *Download & Open Project*.

› *User Guide* provides links to this document in all available languages.

› *For Developers* contains links to various guide and references documents and other on-board tools.

› *About* presents the version of this Bitwig Studio installation. It may be useful for bug reporting, etc.

0.3. Document Conventions

Here are a few notes on the formatting of this document, particularly in relation to the platform you may be using:

› Whenever key commands are the same for Windows, OS X, and Linux, the command will be listed once without any comment. When the key command is different for the platforms, the Windows/Linux version will be listed first, and the Mac version will follow and be labeled. An

example for the copy function would be: press [CTRL]+[C] ([CMD]+[C] on Mac).

› If you are on a Mac, your [ALT] key might be labeled "option." In this document, it will always be called [ALT].

› If you are on a Mac, your "command" key might be labeled with an apple icon. In this document, it will always be called [CMD].

› If you are on a Mac, right-clicking can also be achieved by [CTRL]-clicking.

› Screenshots in this document were made with the Mac version of Bitwig Studio.

1. Bitwig Studio Concepts

This chapter is both an introduction to the program and an overview of its structure. Please start here to get acquainted with the fundamental concepts and related vocabulary used in Bitwig Studio.

1.1. Top-Level Concepts

Bitwig Studio is a modern digital audio workstation (DAW) that allows you to seamlessly compose, produce, perform, and expand your music.

A file created in Bitwig Studio is called a *project*. You can have multiple projects open at once, but audio will be active for only one of these projects at a time.

Bitwig Studio projects are organized into *tracks*, which can be thought of as either individual instruments or layers that should be handled similarly. Each track contains a signal path that results in audio and has common mixing board controls (such as volume, panning, solo, and mute).

Clips are containers for individual musical ideas. Clips store either notes or audio, as well as control and automation data.

Music is made in Bitwig Studio by creating a project and populating its tracks with clips, which you can then refine, arrange, and trigger.

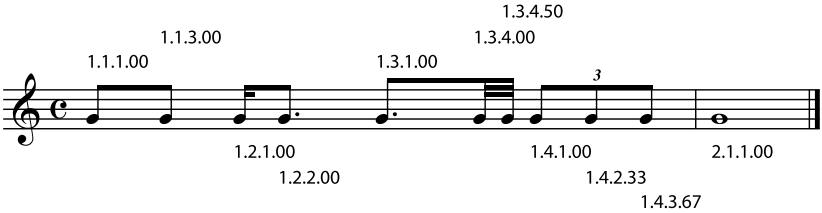
1.2. A Matter of Timing

As Bitwig Studio's primary task is to record and play back music, the element of time is crucial. The *transport* (most closely associated with the global play, stop, and record buttons) is the engine that drives all time functions in Bitwig Studio. This means that for any clip(s) to be played back, triggered or recorded, the transport must be active, propelling the Global Playhead forward.

Bitwig Studio works with time in musical units of bars, beats, and ticks (a set subdivision, which defaults to sixteenth notes). A final value is stored for finer resolution, which is a rounded percentage of the distance between the current tick and the next one. These four units are shown together with period spacers in this way: `BARs.BEATs.TICKs.%`

For example, with a default time signature setting of *4/4*, *1.3.4.50* would represent an event happening in the first bar, on the third beat, within

the fourth sixteenth note, exactly halfway to the next sixteenth note. The example below uses Bitwig Studio's counting system to label a rhythm in traditional musical notation:



1.3. One DAW, Two Sequencers

Within Bitwig Studio are two independent sequencers:

› The *Arranger Timeline* (or *Arranger*) is a linear sequencer that operates across a standard musical timeline. This is the place for sketching and producing full-length songs or other works.

› The *Clip Launcher* (or *Launcher*) is a nonlinear sequencer where you can accumulate a bank of musical ideas and then mix and match them. Clips in the Launcher can be organized into groups called *scenes*, either for triggering those clips together or for composing in blocks (such as verse, chorus, bridge, etc.).

The Arranger Timeline and Clip Launcher contain completely separate data. Editing clips on the Arranger Timeline has no effect on those stored in the Clip Launcher, and vice versa. But the Arranger Timeline and Clip Launcher do interact in several critical ways:

› Clips can be freely copied between the Arranger Timeline and Clip Launcher. When selected together, multiple clips can also be copied back and forth, and scenes can as well.

› The result of all triggered Launcher clips can be recorded directly to each Arranger track, allowing you to capture an improvisation that can be edited later.

› Except when recording the Clip Launcher's output to the Arranger Timeline, only one of these two sequencers is active at any given time. So on a track-by-track basis, you choose whether the Arranger Timeline or Clip Launcher is in control and can trigger its data.

› By default, the Arranger Timeline is the active sequencer for each track.

› Each track can play only one clip at a time.

1.4. Devices, Modulators, and Other Signal Achievements

Devices are special-function components that extend your signal paths by modifying or transforming incoming notes or audio signals.

Every track has a *device chain*. In terms of signal flow, this device chain falls between the incoming sequencer data and the track's mixing board section. In this device chain you can insert as many devices as you like. You can even use Bitwig's devices to create additional device chains.

Each device has *parameters*, which are settings that determine how that device operates. Parameters are set directly within the device's interface or via an assigned MIDI controller. Parameter values can also be sequenced via automation, adjusted via the device's remote controls, or manipulated by *modulators*, which are special-purpose modules that can be loaded within any device — or onto any track for control of all its contained devices and mixer controls.

Devices are grouped into several descriptive categories, including these:

› *Analysis.* Devices that merely visualize the signals that reach them. They make no effect on the audio chain they are a part of.

› *Audio FX.* Devices that manipulate incoming audio signals before passing them onward.

› *Container.* Utility devices whose primarily function is to host other devices.

› *Delay.* Delay line-based processors that operate on their incoming audio signals.

› *Distortion.* Shapers and other mangling processors that operate on their incoming audio signals.

› *Dynamic.* Processors that operate on their incoming audio signals, based off of those signals' amplitude levels and trends.

› *EQ.* Sets of frequency-specific processors that operate on their incoming audio signals.

› *Filter.* Frequency-specific processors that operate on their incoming audio signals.

› *Hardware.* Interface objects for sending signals and/or messages to devices beyond Bitwig Studio (such as hardware synthesizers and effect units, etc.). This can include transmitting and/or receiving audio signals, control voltage (CV) signals, and clock messages.

› *MIDI.* Transmitters for sending various MIDI messages via the track's device chain. This is useful for sending messages to plug-ins or to external hardware (when used in conjunction with Bitwig's *hardware* devices).

› *Modulation.* Processors that manipulate incoming audio signals with an LFO, etc. influencing their function.

› *Note FX.* Devices that generate or manipulate incoming note messages before passing them onward.

› *Reverb.* Time-based processors that operate on their incoming audio signals.

› *Routing.* Devices that divert a track's signal path, allowing signals to exit and/or reenter the track.

› *Spectral.* Devices that operate in the frequency domain, working with hundreds of individual frequency bands.

› *Synth.* Synthesizer instruments that either generate their audio from rudimentary source material or use audio samples. Incoming note messages are used to synthesize audio.

› *The Grid.* Devices utilizing **The Grid**, Bitwig's modular sound-design environment (see chapter 17).

› *Utility.* An assortment of devices sporting various generating, processing, and time-shifting functionality.

All device chains in Bitwig Studio support both audio and note signals. To keep these signals accessible, a few rules apply.

› Except for note FX devices, all devices receiving note signals pass them directly to their output. (Note FX process the incoming notes before passing them onward.)

› Except for audio FX devices, all devices receiving audio signals pass them to their output. (Audio FX process the incoming audio before passing them onward.)

› Many Bitwig devices possess a *Mix* parameter. Similar to a "wet/dry" fader, this control blends the raw audio that entered the device into the device's output.

In Bitwig Studio, all audio signal paths are stereo.

1.5. A Musical Swiss Army Knife

Bitwig Studio's various viewers and editors are called *panels*. These panels are the heart of the program and the places where all work happens.

The **Arranger Timeline Panel** lets you see all of your project's tracks, create an arrangement with timeline clips, and edit track automation.

The **Clip Launcher Panel** allows you to trigger clips both freely and in sync with the transport, copy clips into and out of the Arranger, and sort clips into scenes.

The **Inspector Panel** displays all parameters for any selected clips, notes, audio events, or tracks (and modulation parameters for any selected devices).

The **Detail Editor Panel** is the graphical editor for both notes and audio, and their affiliated data.

The **Automation Editor Panel** gives you detailed control over track automation, clip automation, and MIDI control messages.

The **Device Panel** shows the full device chain for the selected track, including an interface for each Bitwig device and VST plug-in in use.

The **Mixer Panel** presents the channel strip for each track and any subsidiary signal chains.

The **Browser Panel** allows you to preview, load, save, and tag content from your Bitwig Studio library and elsewhere on your machine.

The **Project Panel** manages your project's metadata, gives access to all Arranger cue markers and Launcher scenes, and shows the status of files and plug-ins being used.

The **Output Monitoring Panel** gives audio control options, such as routing the main audio buss to any pairs of speakers and headphones, solo and cue behaviors etc.

The **Mappings Browser Panel** allows you to make and edit project-specific connections of your computer keyboard and/or MIDI controller(s) to your project's parameters.

The **On-screen Keyboard Panel** provides visualizations of the selected track's playing and incoming note messages, pitch expressions, and timbre expressions, as well as an input method for these data streams.

The primary interfaces in Bitwig Studio are called *views*. Each view gives you access to a set of panels chosen to help you carry out a particular musical job.

› The **Arrange View** lets you focus on assembling music, particularly by recording and ordering clips. The **Arranger Timeline Panel** is central to this view along with the optional **Clip Launcher Panel**. All panels are available here, and all project tracks are viewed together.

› The **Mix View** focuses on mixing tracks and triggering clips. The **Mixer Panel** is central to this view along with the optional **Clip Launcher Panel**. Except for the **Arranger Timeline Panel**, all other panels are available here, and all project tracks are viewed together.

› The **Edit View** is for making detail edits to clips. The **Detail Editor Panel** is central to this view along with the optional **Automation Editor Panel**. Except for the **Arranger Timeline**, **Clip Launcher**, and **Mixer** panels, all other panels are available here.

Bitwig Studio offers several window arrangements called *display profiles*. These configurations adjust the placement of panels and even provide additional application windows when appropriate. This is all in the name of optimized workflows, allowing the program's layout to match your current screen arrangement and the task at hand.

› *Single Display (Large)* is intended for use with one monitor, using a single application window to focus on one of Bitwig Studio's views at a time. *This is the default display profile (and the one used for screenshots within this document).*

› *Single Display (Small)* is similar to the *Single Display (Large)* profile but is optimized for use on a smaller monitor.

› *Tablet* is intended for use with a supported tablet computer. This profile is optimized for touch-and stylus-based interfaces, allowing you to play and create notes thru a specialized **Play View**. (Depending on your operating system and hardware platform, this option may not be available.)

Note

Information on Bitwig Studio's tablet computer-specific features can be found in chapter 18.

› *Dual Display (Studio)* is intended for use with a two-monitor setup, such as a laptop screen and an external display. This profile keeps the **Arrange View** on your primary display and toggles your secondary display between the **Mix View** and the **Edit View**.

› *Dual Display (Arranger/Mixer)* is intended for use with a two-monitor setup. This profile is fixed, keeping the **Arrange View**on your primary display and the **Mix View** on your secondary display.

› *Dual Display (Master/Detail)* is intended for use with a two-monitor setup. This profile keeps the **Edit View** on your secondary display and toggles your primary screen between the **Arrange View** and **Mix View**.

› *Dual Display (Studio/Touch)* is intended for use with a two-monitor setup where one of the monitors is a touch-screen tablet. This profile provides one standard window (like the *Single Display (Large)* profile) for your standard monitor and a slightly modified *Tablet*-style window for interacting with Bitwig via your touch-screen interface.

› *Triple Display* is intended for use with a three-monitor setup. This profile is fixed, keeping the **Arrange View** on your primary display and the **Mix View** and **Edit View** on your secondary and tertiary displays.

1.6. User Interfacing

Finally, a few notes to help you interact with Bitwig Studio.

› Any interface control (like a knob or curve control) can be set with the mouse by clicking and dragging upward or downward. You can [CTRL]-click ([CMD]-click on Mac) on the control to set its value with the keyboard. Double-clicking on the control restores its default value.

› Any numeric control (one that directly shows you numbers) can be set with the mouse by clicking and dragging upward or downward. You can also double-click on the control to set its value with the keyboard.

› Any control at all can be fine-tuned with the mouse by [SHIFT]-clicking the control and dragging. If you have already clicked the control, you can also press [SHIFT] after the fact to engage this mode.

› When a button is tinted orange, that control is active. The inactive form of a control uses a neutral color, such as white, gray, or silver.

› Many key commands remain available while you are clicking and dragging an item. These include the commands for toggling panel visibility or switching the current view.

› Only one visible panel will ever have focus at a given time. Focus follows the panel that was last clicked or activated. Panel focus is indicated by the outer rounded rectangle being tinted silver. Key commands that target a specific panel are available only when that panel is in focus.

› Enabling [CAPS LOCK] causes your computer keyboard to transmit note messages. While this can be a quick way to enter notes, it will also disable many normal key commands. If your key commands are not working, make sure that [CAPS LOCK] is disengaged.

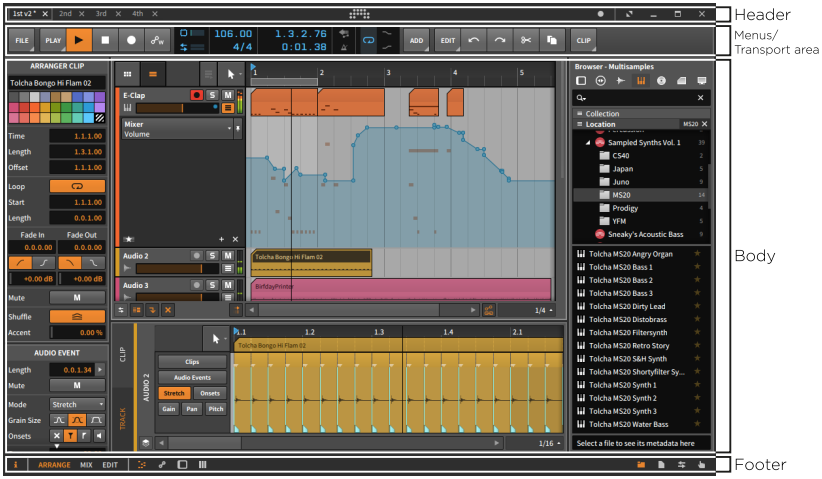
› Many of Bitwig Studio's functions already have computer keyboard shortcuts assigned, but you can modify these shortcuts and even assign them to MIDI controllers as well.

To globally make or modify keyboard and/or controller shortcuts: call up the **Dashboard**, click the *Settings* tab, and then click to load the *Shortcuts* page. From here, you can select between computer *Keyboard* and MIDI *Controller* assignments, and then scroll to browse the categorized program functions, or type to search them by action name or assignment. From this preference tab you can also save and switch between various keyboard mapping sets (via the *Choose mappings* menu).

To assign keyboard and/or controller shortcuts for a particular project: use the **Mappings Browser Panel** (see section 15.4).

2. Anatomy of the Bitwig Studio Window

All functions and controls of Bitwig Studio are accessible thru the application window. Each window can be thought of in four vertical slices: the header, the menus/transport area, the body, and the footer.



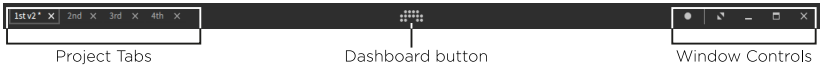
We will give them each their own turn: the reliable header, the pliant footer, the shifting menus/transport area, and finally the mercurial body.

Note

When using the *Tablet* display profile, some of the elements listed in this chapter are rearranged. For details on using a tablet computer, see chapter 18.

2.1. The Window Header

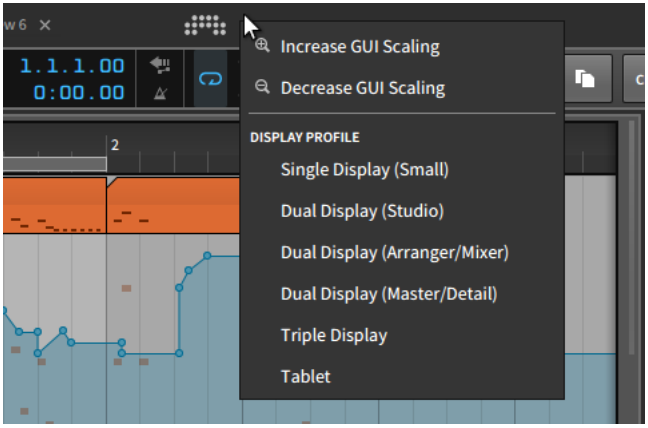
The header of each window contains two main sections: project tabs are found on the left, and window controls are found on the right.



The area just to the left of the window controls is also used controller status icons, if controllers are connected and configured. Otherwise, nothing appears here.

In the center is the *Dashboard button*. When clicked, the **Dashboard** will appear over the main window. For more information on the **Dashboard**, see section 0.2.

It is also worth noting that by right-clicking anywhere in the window header, a context menu with display options is called up.



The *Increase GUI Scaling* and *Decrease GUI Scaling* options allow you to resize Bitwig Studio's entire graphical user interface to be larger or smaller (respectively) on your monitor.

Note

By default, Bitwig Studio makes maximum use of your screen. As such, the *Decrease GUI Scaling* option may not do anything if you try it first.

Beneath the GUI options are a list of the available *Display Profile* choices (see section 1.5) for easy switching.

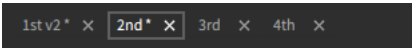
2.1.1. Project Tabs Section

On the far left are tabs for the Bitwig Studio projects which are currently open. Some notes on using these tabs:

› Bitwig Studio will display the contents of only one project at a time. This is true even if you are using a display profile that uses multiple application windows.

› To focus on any one of the open projects, click on its tab.

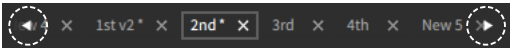
› The tab that is outlined with a box and whose name appears in bright white represents the currently viewed project. In the image below, this is the project named *2nd*.



› Only one project at a time is capable of producing sound. This allows you to view and even edit different projects without interrupting audio playback of the current one.

› You can click and drag any project tab to change its position.

› If there is not enough space to show all open projects together, left and right scroll arrows will appear around the project tabs.



› An asterisk (*) will be appended to any project's name if unsaved changes have been made.

› The *x* on the right side of each tab can be clicked to close that project.

2.1.2. Controller Status Section

When MIDI controllers are connected and configured, the area just before the window controls section is used to display one icon per controller (within reason).

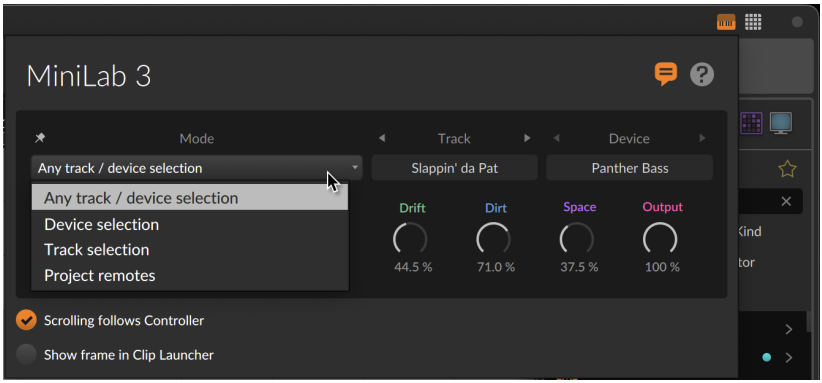
While the icons are suggestive of each device's layout — here showing one regular controller, and one pad-style controller — mousing over the icon will show the controller name.

Clicking on the icon offers a status view for that controller.

First, the icons at the top right and settings at the bottom are similar to what the **Dashboard** offers under *Settings* > *Controllers* (see section 0.2.2.2). The dark field in the middle offer some information and some control.

Informationally, we see exactly what this controller is currently looking at. In this case, a *Device* on a particular *Track* is being targeted, and the names and current values of the parameters in question are shown on the knobs.

Then there is this *Mode* menu, which determine what the controller will follow.



The *Mode* options include:

› *Any track / device selection* (the default setting) will focus this controller on the remote controls of any element selected in the software, including devices, tracks, and project remotes (when selecting the master track).

› *Device selection* will follow the remotes of only devices that are selected.

› *Track selection* will follow the remotes of only tracks that are selected.

› *Project remotes* will keep the controller focus on the project-level remote controls, regardless of what other project elements are clicked on.

This status page can also be used to navigate to other targets, by clicking the left and right stepper triangles around the *Track* and *Device* elements. And particular targets can also be "pinned" or locked so that they stay in focus.

Mousing over either the *Track* or *Device* element will hint at this option, showing a thumbtack icon while you hover.

To pin a controller's focus on a particular track or device: simply click that track or device in the controller status pop-up.

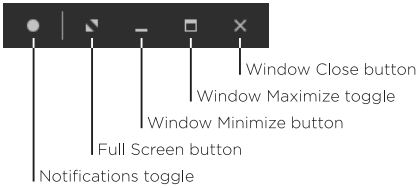
To unpin a device's focus from a particular track or device: either click the selected track or device again to toggle it off, or click to pin the controller to a different target.

2.1.3. Window Controls Section

Note

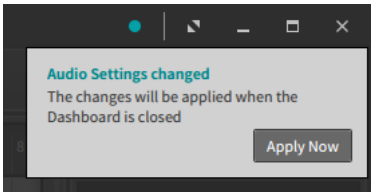
If your operating system has a different standard for window controls, then we try to use their preferred layout. For example on macOS, the *notification toggle* (shown below) will be alone in the top right corner of the window, and OS-standard close (red), minimize (yellow), and maximize (green) buttons appear on the left.

On the far right of the window header are options for controlling Bitwig Studio's window size, appearance, and notifications.



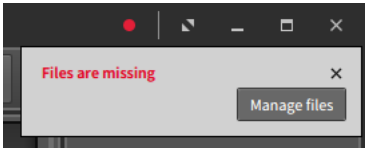
› *Notification toggle* allows you to show or hide event notifications from Bitwig Studio. The filled circle shown above represents that notifications are enabled, and an empty circle indicates that they will not pop up.

When notifications are enabled, any received message it will pop up below the window header.



Most often, notifications are presented with an action button (such as *Apply Now* in the above image). Notifications tinted blue-ish are largely

assistive. Notifications tinted red represent errors that could adversely impact the performance of your project or the program itself.



Finally, when a message is received but notifications are disabled from appearing, the ring of the empty circle icon is tinted based on the type of notification that has arrived.

› *Full screen button* switches Bitwig Studio into the full-screen mode provided by your operating system. Once you are in full-screen mode, the options available in the window controls section may decrease.

To exit full-screen mode: click the window maximize toggle, to the immediate left of the window close button.

› *Window minimize button* hides the Bitwig Studio window.

› *Window maximize toggle* alternates between maximizing the size of the window and restoring its original, smaller size.

› *Window close button* is the equivalent of quitting Bitwig Studio (by selecting *File* > *Quit*).

2.2. The Window Footer

The window footer contains various buttons that determine which parts of Bitwig Studio are visible, along with context-specific messages of available actions and controller visualizations.

Footers will differ based on the display profile being used. The image above — and all screenshots in this document — shows a footer from the default *Single Display (Large)* profile in **Arrange View**, where all panels and views are available.

2.2.1. Panel Icons

The small icons that appear in the window footer are panel icons. Each icon represents a panel that is available within the current view. The icons are also buttons, allowing you to toggle the visibility of each panel by clicking its icon. An icon that is illuminated in orange indicates an active panel.

For each cluster of icons, only one panel can be shown at a time. These icon clusters are located either on the far-left, far-right, or center-left of the window footer, indicating whether those panels would be displayed on the left, right, or center-bottom of the window, respectively.

The panel icons that you will encounter are:

The **Inspector Panel** icon is a seriffed, lowercase *i*. When available, you can focus on this panel and toggle its visibility by pressing [I] or [ALT]+[I].

The **Detail Editor Panel** icon is an arrangement of dashed lines, like a standard "piano roll" representation of notes. When available, you can focus on this panel and toggle its visibility by pressing [E] or [ALT]+[E].

The **Automation Editor Panel** icon is two circles connected by a line, like the breakpoints that build an automation curve. When available, you can focus on this panel and toggle its visibility by pressing [A] or [ALT]+[A].

The **Device Panel** icon is a rounded rectangle with a shaded left side, like the containing box for each device and its left-sided title bar and master controls. When available, you can focus on this panel and toggle its visibility by pressing [D] or [ALT]+[D].

The **Mixer Panel** icon is a series of three wide vertical lines, like the volume faders of a mixing console. When available, you can focus on this panel and toggle its visibility by pressing [M] or [ALT]+[M].

The **Browser Panel** icon is a folder icon, representing the library of content that is accessible in this panel. When available, you can focus on this panel and toggle its visibility by pressing [ALT]+[B].

The **Project Panel** icon is a file icon, representing the project file whose metadata is defined in this panel.

The **Output Monitoring Panel** icon is a pair of opposite-pointing arrows, representing the input and output paths that are addressed in this panel.

The **Mappings Browser Panel** icon is a right-hand with an extended index finger, representing the connections of yourself to your project that are made here.

The **On-screen Keyboard Panel** icon shows the common grouping of five piano-style keys, representing one of the note visualization and entry methods available in this panel.

2.2.2. View Words

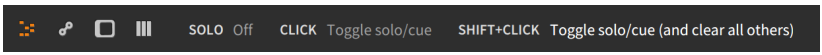
The capitalized, bold words that appear on the left side of the window footer represent all currently available views. To match the views' names, the labels used are *ARRANGE*, *MIX*, and *EDIT*.

A window with no view words indicates that your current display profile is fixed and has only one available view.

For the two-window display profiles (those whose name begins with *Dual Display*), available views are shown as compound names, such as *ARRANGE-MIX* or *MIX-EDIT*. In this situation both windows show the same view words, indicating the views shown on the primary and secondary windows, respectively.

2.2.3. Available Actions

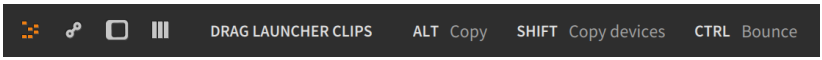
Available actions appear just to the right of all left-aligned view words and panel icons. As your mouse moves around the program, any interactive object that is hovered over will display information and available mouse functions here.



In the example above, a track *SOLO* button is hovered so the line starts with the object name and it's status (the solo button is switched *Off*

currently). Possible *CLICK* and modifier-click options follow. And since I was holding the [SHIFT] key, the *SHIFT+CLICK* option is shown more brightly as it will be used.

Available actions are also shown while you are interacting with the program, as in this example when actively dragging a Launcher clip.



While dragging a clip, I am free to move it to a different clip slot or even to an Arranger track, but additional modifiers are also available to change the basic move action into something more complex. Available actions are there to remind us of workflow variations for tasks that we are already doing.

2.2.4. Parameter Information

Parameter information will appear in the same area when mousing over various controls in the program. This is most commonly seen while working with devices. In the example below, the cursor is hovering over the cutoff control of the filter in **Polysynth**.

Here the footer show the full title of the parameter (*Filter Frequency*) and then the current parameter value (*2.33 kHz*).

Since this happens to be a frequency parameter, the following string shows the relevant pitch as MIDI note (*♩ D6*). Since an arbitrary frequency rarely matches a specific note value, the *tone bar* before the note name signifies the intonation to the note shown:

› ╕ indicates that the frequency is quite sharp.

› ╡ indicates that the frequency is somewhat sharp.

› ┤ indicates that the frequency is very close or in tune.

› ┘ indicates that the frequency is somewhat flat.

› ┘ indicates that the frequency is quite flat.

When a parameter has modulators mapped to it, that parameter's calculated value is also shown.

In the example above, the *Filter Resonance* knob position is set to *39.5 %*. The following bracketed value, *[27.1 %]*, shows the applied value of the parameter after all modulator signals are added.

Note

For information on using Bitwig's modulators to modulate any device or plug-in parameter, see section 16.2.

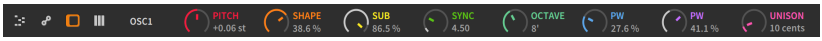
Additionally, parameters that consist of a list of possible settings (such as modes) often present additional information when hovered over.

For example, the *OSC Blend Mode* in **Polysynth** presents six discrete buttons with short mode names (*MIX*, *NEG*, *WIPE*, etc.). As shown in the

image above, mousing over the mode *SIGN* provides a short explanation of what this means in the window footer.

2.2.5. Controller Visualizations

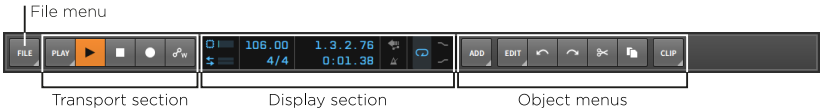
Controller visualizations also use the same middle portion of the footer. They show the current position of controls and the parameters that they are assigned to (for any controller that has visualizations enabled).



The layout and visual style is influenced by the controller script. And when non-immediate takeover modes (see section 0.2.2.2) are being used, the outer ring/indicator shows the current parameter value in white and the colored indicator shows the hardware control's current position. Once the parameter and control meet, both elements use the control color.

2.3. The Window Menus/Transport Area

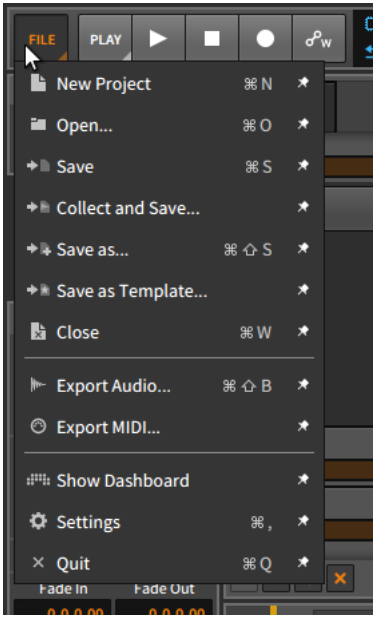
Beneath the window header is an area where Bitwig Studio's menus live, along with the transport and its associated displays.



Some of these elements are persistent, and some are transitory. This is a function of Bitwig Studio's unique menu system, which we will examine first.

2.3.1. The Menu System (via the File Menu)

The *File* menu itself contains only menu items that you would expect and/or those which will be covered in this document at the appropriate time. So we will take this opportunity to see Bitwig Studio's unique menu system at work.



Most items in the menu shown above have four distinct elements:

› An *icon* leads each entry, visually abbreviating the function of the menu item.

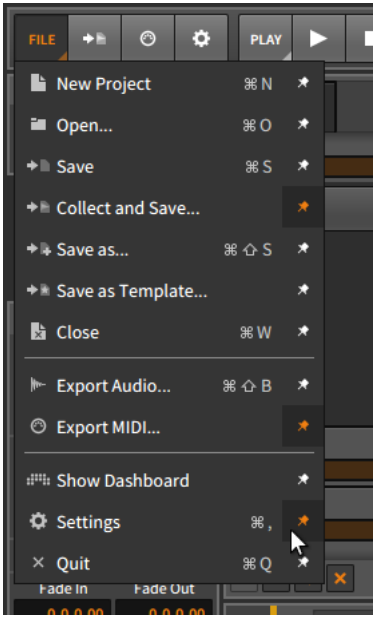
› The *menu item name* itself is always second.

› When defined, a *keyboard shortcut* follows. When more than one keyboard shortcut exists for a menu item, the first shortcut is shown.

> **Note**
>
> For information on making or altering shortcut assignments, see section 0.2.2.4.

› Finally, a *thumbtack toggle* appears at end of each line.

To anchor an item in the menu area: enable the thumbtack toggle beside the menu item. This will place a button with the menu item's icon beside the menu button itself.



In the image above, three menu items (*Collect and Save...*, *Export MIDI...*, and *Settings*) each have their thumbtack toggle enabled. And now to the right of the *File* menu are three shortcut buttons, each representing one of those menu items and showing their menu item's icon. Clicking one of these buttons is the equivalent of triggering the menu item.

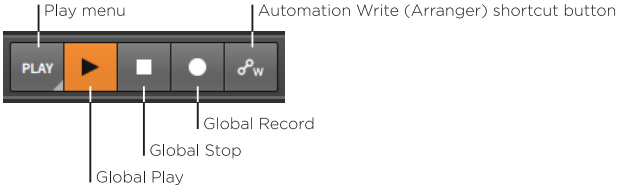
Like the *File* menu, each menu button is indicated with a dog-eared triangle in its bottom right corner, hinting that the button can be unfolded. Every menu in Bitwig Studio uses this system, allowing you to anchor any function that you please to the top level of the program.

Note

If your window is ever sized too narrowly to display all menu options, the program will prioritize by showing all menu buttons first, and then showing as many anchored buttons as will fit the current width.

2.3.2. Transport Section

The transport section appears deceptively simple at first glance.



Let's skip the *Play* menu for the moment and look at the four buttons that follow:

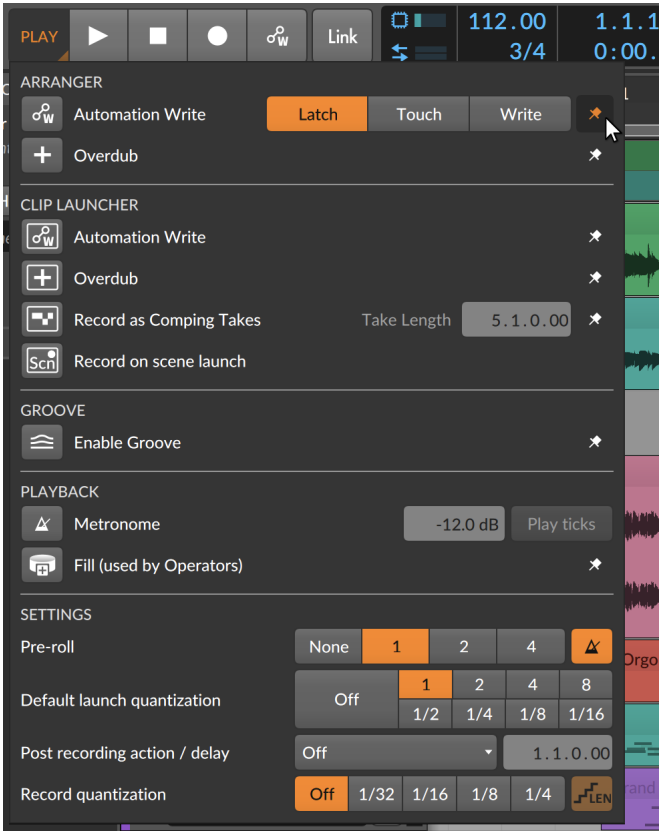
› *Global Play*: Toggles and indicates the state of Bitwig Studio's transport. When clicked to toggle the transport on, Arranger playback resumes from the Play Start Position and active Launcher clips are triggered in sync. When clicked to toggle off, the transport is stopped and the Play Start Position is moved to the current Global Playhead position.

› *Global Stop*: Deactivates the transport. When the transport is already inactive, clicking the global stop button returns both the Global Transport and the Play Start Position to the beginning (play position *1.1.1.00*).

› *Global Record*: Arms all record-enabled tracks. When the global record button is enabled, Arranger recording will begin the next time the transport is started.

› *Automation Write (Arranger) shortcut button*: Enables automation recording to the Arranger Timeline the next time the transport is started.

The three global buttons above will always be present. The shortcut button, however, is so named because you can toggle it in and out of existence. This is available for many more transport options within the *Play* menu.



The *Play* menu still makes use of the thumbtack toggle convention (when appropriate), but it also makes special use of knobs and other controls. There are five headers within this menu:

› The *Arranger* section provides settings that apply when working within the **Arranger Timeline Panel**.

› The *Clip Launcher* section provides settings that apply when working within the **Clip Launcher Panel**. Note the clip boxes around the icons in this section, helping to distinguish the Launcher functions from similar Arranger functions.

› The *Groove* section allows you to activate shuffle for all clips whose own *Shuffle* parameter is enabled. Other parameters here include the *Shuffle* amount and interval (*Rate*), as well as the *Accent* amount, interval (again, called *Rate*), and *Phase*.

Note

All controls in the *Groove* section can be mapped and/or automated.

› The *Playback* section provides parameters that take effect during project playback, such as the *Metronome* volume, whether sub-beats should also sound (*Play Ticks*), and the mappable *Fill* mode toggle, used by *Occurrence* Operator (see section 12.1.3) and available via the **Globals** modulator (see section 19.27.3.3).

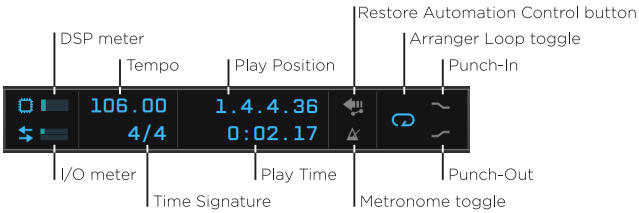
› The *Settings* section offers a mix of workflow parameters, including *Pre-roll* controls (for length and whether the metronome should be active), and whether you want *Record Quantization* applied to notes (and if so, whether you want their end times to be quantized as well).

Finally, note that Bitwig Studio's audio engine can be engaged for only one Bitwig Studio project at a time, no matter how many are open. So if your current project does not have audio enabled, the transport section will be replaced by a single button.

Simply click this button to rejoin the audible world. (Just realize that this will silence any other project that was previously using audio.)

2.3.3. Display Section

The menus/transport area's display section provides informational meters, numeric controls, and the odd automation-related setting.



This section contains the following items:

› *DSP meter*: Displays Bitwig Studio's current CPU usage. (Clicking the processor chip icon on the left will also load a *DSP Performance Graph window*, including various details and metrics.)

› *I/O meter*: Displays Bitwig Studio's current disk activity for data being read (input) and written (output), respectively.

› *Tempo*: A control for the project's current tempo, set in beats per minute (BPM).

› *Time Signature*: A control for the project's current time signature and an optional tick setting.

The time signature's numerator represents the number of beats in each bar. Common denominators are accepted (such as *2*, *4*, *8*, and *16*), each number representing the type of beat counted in each bar (half, quarter, eighth, and sixteenth notes, respectively).

The optional *tick* setting represents the primary beat subdivision to be used across the project (see section 1.2). If only a time signature is set (like *4/4*), a default tick setting of sixteenth notes is used. If the time signature is followed by a comma and an appropriate tick value (such as *4/4,8*), then that tick setting will be used. Values recognized by Bitwig Studio include *8* (eighth notes), *12* (triplet eighth notes), *16* (sixteenth notes), *24* (triplet sixteenth notes), *32* (thirty-second notes), and *48* (triplet thirty-second notes).

› *Play Position*: A control for the project's current play position, shown as `BARs.BEATs.TICKs.%`.

› *Play Time*: A control for the project's current play time, shown as `MINUTEs:SECONDs.MILLISECONDs`.

› *Restore Automation Control button*: Restores control of automation after a parameter is adjusted during playback. The Restore Automation Control button arms itself when the function is useful.

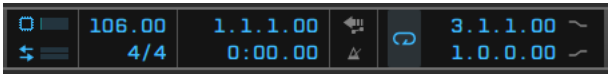
› *Metronome toggle*: Enables/disables the metronome whenever the transport is active.

› *Arranger Loop toggle*: Activates/deactivates Arranger looping within the Loop Selector's bounds. This toggle together with recording also enables "cycle recording" on the Arranger for comp recording (see section 5.3.3.3).

› *Punch-In*: Causes recording to begin at the start of the Arranger Loop Selector.

› *Punch-Out*: Causes recording to stop at the end of the Arranger Loop Selector.

From the **Dashboard** on the *Settings* page, the *User Interface* tab has a *Transport* parameter that can also *Show Loop Region* within the display area. This displays the Arranger Loop Selector's start time and length, both to the right of Arranger Loop toggle.



2.3.4. Object Menus

The far right of the window menus/transport area is reserved for the object menus.

Three menus generally appear here, each with their own set of anchored items:

› The *Add* menu is always present. It allows you to create new tracks and scenes.

› The *Edit* menu is always present. It provides standard "edit" commands for your current selection (like cut, copy, paste, duplicate, and delete), as well as to undo (or redo) recent actions taken across the program.

› The third menu is a *selection-sensitive menu*. If nothing is selected in your Bitwig Studio project, then no menu appears here. But if you have selected, say, a *Clip* or *Event*, then a menu with relevant functions will appear. This is essentially a context menu with the option to create shortcut buttons (using the menus' thumbtack toggles).

  For example, if we made a time selection, a *Time* menu would be provided in the third, selection-sensitive slot.

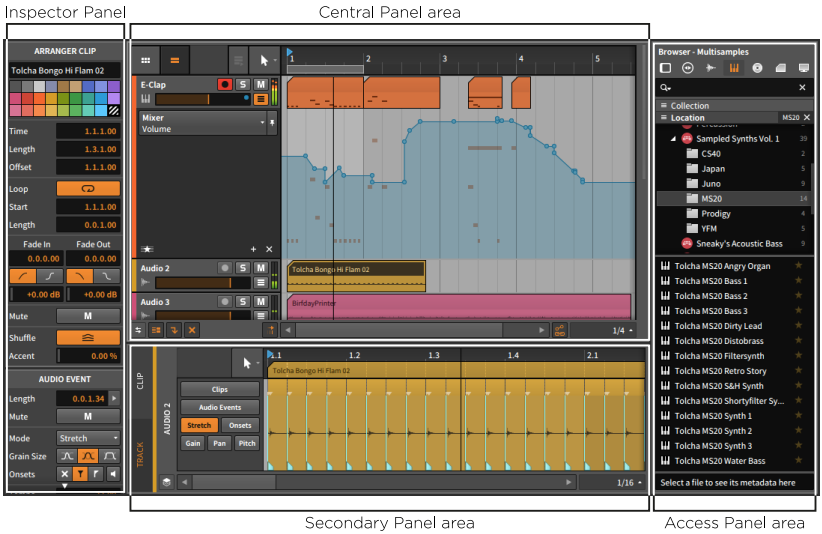
Also note in that last image that when a function is currently unavailable, its shortcut button appears grayed out. As the menu item would appear, so will the shortcut button.

2.4. The Window Body

So the window header is always the same (aside from the project tabs), and while the footer's content and arrangement depend upon the current display profile, the set of controls is consistent. These two areas

give you control of the program and its behavior so they are generally static. Not so with the window body.

The window body's purpose is to display your work so that you can edit it in different situations. To that end, the body's appearance is always changing, giving you the tools you need to perform specific tasks, but certain areas of the window body are designated for consistent usage.



The central portion of the Bitwig Studio window is reserved for the *central panel*. The panel(s) shown here is defined by the window's current view (either **Arrange**, **Mix**, or **Edit View**). The central panel cannot be hidden, so if all other panels were disabled, the central panel would take up the entire window body.

Below the central panel is the *secondary panel area*. This area is where a second panel can be loaded for editing your project's content. Again, the selection of available panels is determined by the window's current view and the display profile being used. Most secondary panels can be vertically resized.

On the right side of the window body is an *access panel area*. This area is usually reserved for panels that deal with things other than the content of your project. Typical access panels are the **Browser Panel** (which gives access to the Bitwig Studio library and outside files), the **Project Panel** (which gives access to the project's metadata and dependencies), the **Output Monitoring Panel** (which gives access to your hardware routings), and the **Mappings Browser Panel** (which gives access to both MIDI controller mappings and project-specific computer keyboard

mappings). Each of these panels can be horizontally resized. When no panel is loaded in this area, the central and secondary panels simply reclaim the space.

On the left side of the window body is an area usually reserved for the **Inspector Panel**. In certain display profiles, however, the **Inspector Panel** is included in the access panel area. This panel is not resizable.

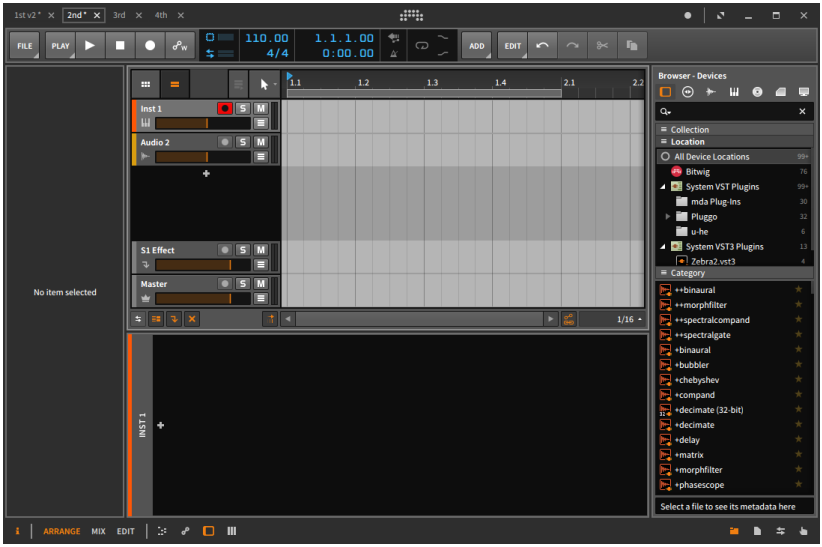
3. The Arrange View and Tracks

Now that we have examined all the fixed parts and dynamic possibilities of the Bitwig Studio window, let's enter the practical world of the **Arrange View**. We will start by looking at a few key sections of the **Arranger Timeline Panel** and their constituent elements. We will then examine the track types used by Bitwig Studio along with basic track editing functions. Finally we will get a brief introduction to the **Inspector Panel** for current and future use.

3.1. The Arranger Timeline Panel

Unlike sculpture, painting, and architecture, music is an art form appreciated over a defined length of time. That is to say, when we listen to a piece of music, either at home or out at a venue, it unfolds over the same amount of time and at the same pace for everyone in the audience. While music can definitely be performed or created with improvisation (see chapter 6), each performance has a rigidly defined structure to us listeners. And as most productions are still based around a fixed song structure, we will start with the **Arrange View** and its friend the **Arranger Timeline Panel**, which is made to lay out music arrangements in a precise way.

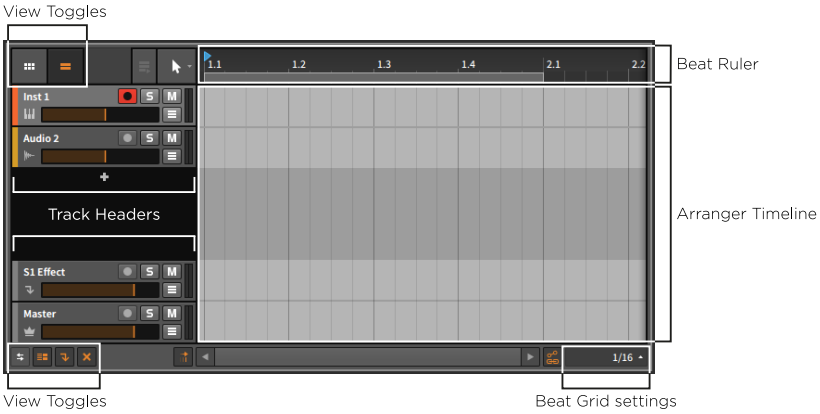
The **Arranger Timeline Panel** is unique in Bitwig Studio: it is available in only one view (the **Arrange View**), and it is available in this view only as the central panel. And as this panel is the only way to create a traditional, linear musical arrangement within Bitwig Studio, it is impossible to overstate the importance of the **Arranger Timeline Panel** — also called the *Arranger* — which is seen here after a new file has been created.



We will start by examining various sections of the **Arranger Timeline Panel**.

3.1.1. Arranger Area, Arranger Timeline, and Zooming

The most important element here is the actual *Arranger Timeline*, which is currently blank. As you may have seen here in earlier images (or from opening a demo project), this is the area where your song arrangements will take shape in the form of clips and track automation. Whenever we refer to an "Arranger clip," we mean a clip that is housed within this Arranger sequencer.



The Arranger is laid out horizontally, showing time progressing from the left side of the screen to the right. This can be seen in the *Beat Ruler* at the top of the Arranger. The integers here — *1*, *2*, *3*, etc. — show where each new bar begins.

To adjust the zoom level: place the mouse in-line with the bar numbers inside the Beat Ruler. The cursor will become a magnifying glass indicating that we are in *zoom mode*. Now click and hold the mouse button, dragging upward to zoom in or downward to zoom out. You can also drag the mouse from side to side to horizontally scroll within the Arranger Timeline.

Other ways to adjust the zoom level include:

› Press either [PLUS] or [CTRL]+[PLUS] ([CMD]+[PLUS] on Mac) to zoom in and either [MINUS] or [CTRL]+[MINUS] ([CMD]+[MINUS] on Mac) to zoom out.

› Hold [CTRL]+[ALT], and then click and drag anywhere within the Arranger area. If your mouse or trackpad supports a scroll function, you can also hold [CTRL]+[ALT] anywhere within the Arranger area and then scroll up and down.

› If you have a three-button mouse, click and drag the middle button anywhere within the Arranger area.

› If you have a trackpad (particularly on Mac), pinch/stretch two fingers diagonally on the trackpad.

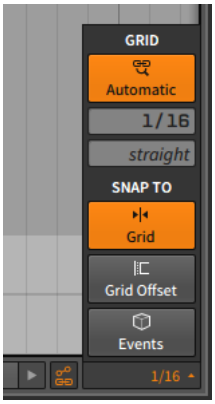
As you zoom in on the Beat Ruler, you may notice that the bar numbers start adding decimals. Depending on your zoom level, the timeline values will be represented as either `BARs`, `BARs.BEATs`, or `BARs.BEATs.TICKs`.

And within the Beat Ruler area, you can also right-click to show a *realtime ruler*, displaying `MINUTES:SECONDS.MILLISECONDS` of the project time.

3.1.2. Beat Grid Settings

As you adjust the Arranger Timeline's zoom level, you may also notice that the grid lines within the Arranger area begin to change. This has to do with the *beat grid settings*, which are found in the bottom of the **Arranger Timeline Panel** and to the right of the horizontal scroll bar.

Actually, the value shown represents the current value in use. By clicking on that value, the various *Grid* settings are exposed.



The *beat grid resolution* (shown above as *1/16*, for sixteenth notes) tells us what musical interval is being represented by the grid lines. In a new project, the *adaptive beat grid* setting (the button at top, with a linked magnifying glass and the word *Adaptive*) is turned on. When adaptive beat grid is enabled, changes to the zoom level also cause appropriate changes to the beat grid resolution. The beat grid resolution setting will update as the value changes.

To toggle the adaptive beat grid: click the adaptive beat grid button within the beat grid settings, or press [SLASH] .

Note

On a German keyboard, the key command is [HYPHEN] .

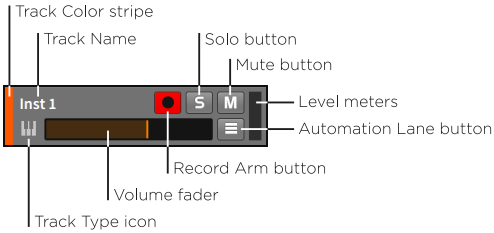
To manually set the beat grid resolution: first make sure that adaptive beat grid is disabled. Then manipulate the beat grid resolution by setting it with the mouse or by pressing [COMMA] to lower the grid resolution or [PERIOD] to raise it.

The beat grid resolution has an accompanying parameter right below it. The *beat grid subdivision* (shown above as *straight*) sets the rhythmic grouping used for the beat grid resolution setting. For example, the default *straight* value means that straight duple values are being used. Other available settings include *triole* or *3t* (triplets), *quintole* or *5t* (quintuplets, or fifth-lets), and *septole* or *7t* (septuplets, or seventh-lets).

To manually set the beat grid subdivision: first make sure that adaptive beat grid is disabled. Then manipulate the beat grid subdivision by setting it with the mouse or by pressing [ALT]+[COMMA] to lower the grid resolution or [ALT]+[PERIOD] to raise it.

3.1.3. Track Headers

The horizontal lines you see within the Arrange area are the dividers between each track lane. To the left of the Arrange area are the *track headers*.



Within each header are the following identifications, meters, and controls for that track:

› *Track Color stripe*: A swatch of the track's assigned color.

› *Track Type icon*: An icon to indicate the kind of track.

› *Track Name*: The title assigned to the track.

› *Volume fader*: A final level control for the track.

› *Record Arm button*: Record enables the track.

› *Solo button*: When any track has its solo button enabled, only tracks with solo enabled will output their audio.

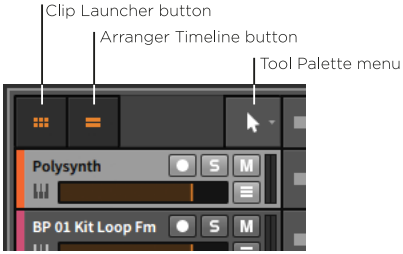
› *Mute button*: Disables the track's audio output.

› *Automation Lane button*: Toggles to reveal the automation lane section of the track (see section 9.1.1).

› *Level meters*: Stereo audio meters that display the track's output level.

3.1.4. Arranger View Toggles

Both above and beneath the track headers are the *Arranger view toggles*. Similar to the panel icons of the window footer, each of these icons is a toggle that adjusts what is displayed in the **Arranger Timeline Panel**.



The upper toggles are:

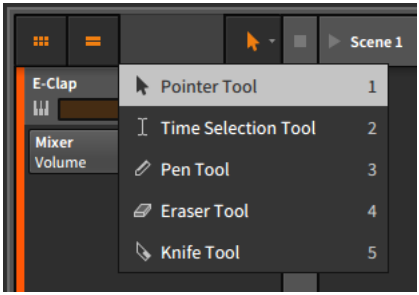
› *Clip Launcher button*: Toggles visibility of the **Clip Launcher Panel** (see section 6.1) within the **Arranger Timeline Panel**.

› *Arranger Timeline button*: Toggles visibility of the Arranger Timeline within the **Arranger Timeline Panel**

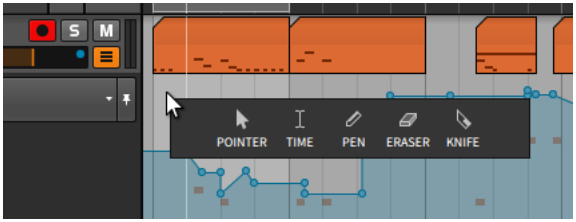
Note

Either the **Clip Launcher Panel** or the Arranger Timeline must be visible within the **Arranger Timeline Panel**. If only one of these is visible and you hide it, the other will automatically become visible.

› *Tool Palette menu:* This menu allows you to toggle between Bitwig Studio's various editing tools.



In fact, right-clicking within any timeline-based panel will give you the option to switch tools at the top of the context menu.



While the **Arranger Timeline Panel** is the first place we see the tool palette, each timeline-based panel has its own tool palette. This allows us to have a different tool selected for each individual panel.

› *Pointer tool* is for selecting and moving events. Clicking between automation points along the current curve will create a new point. And double-clicking in a blank area will create a new event of the appropriate kind. You can switch to this tool by pressing [1], or you can temporarily use the tool by holding [1].

Note

As this is the primary tool in Bitwig Studio, all editing functions described in this document presume you have the Pointer tool engaged. If a different tool is meant to be used, it will be specifically noted.

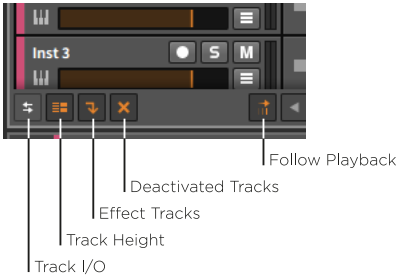
› *Time Selection tool* is for choosing an arbitrary section of time instead of particular events. Otherwise it generally acts like the Pointer tool. You can switch to this tool by pressing [2], or you can temporarily use the tool by holding [2].

› *Pen tool* is for drawing new events. You can switch to this tool by pressing [3], or you can temporarily use the tool by holding [3].

› *Eraser tool* is for deleting relevant events from the area of time that you select. You can switch to this tool by pressing [4], or you can temporarily use the tool by holding [4].

› *Knife tool* is for splitting a continuous event into two. You can switch to this tool by pressing [5], or you can temporarily use the tool by holding [5].

Finally, the Pointer tool engages in *smart tool switching*. This is to say that depending on where you hover over a clip or event, different tools will become available. Specific information will be provided within this document, but it is worth mentioning here as your cursor will tend to shift shapes as you mouse navigate around clips.



The lower toggles are:

› *Track I/O button*: Toggles visibility of the Track I/O section of all track headers (see section 5.3.1).

› *Track Height button*: Toggles the track height in the Arranger between normal and half size (shown below respectively). In half size, the same track header components are displayed with some minor adjustments.

› *FX Tracks button*: Toggles visibility of FX tracks within the **Arranger Timeline Panel**.

› *Deactivated Tracks button*: Toggles visibility of deactivated tracks within the **Arranger Timeline Panel**.

› *Follow Playback button*: Toggles whether to keep the Global Playhead on screen at all times in the **Arranger Timeline Panel** or not.

Note

From the *Settings* tab within the **Dashboard**, the *User Interface* page offers two settings for the *Playhead follow mode*:

› *Scroll by pages* will scroll once the Global Playhead reaches the edge of the current display area. This is the default setting.

› *Continuously scroll* will keep the Global Playhead centered in each timeline-based panel.

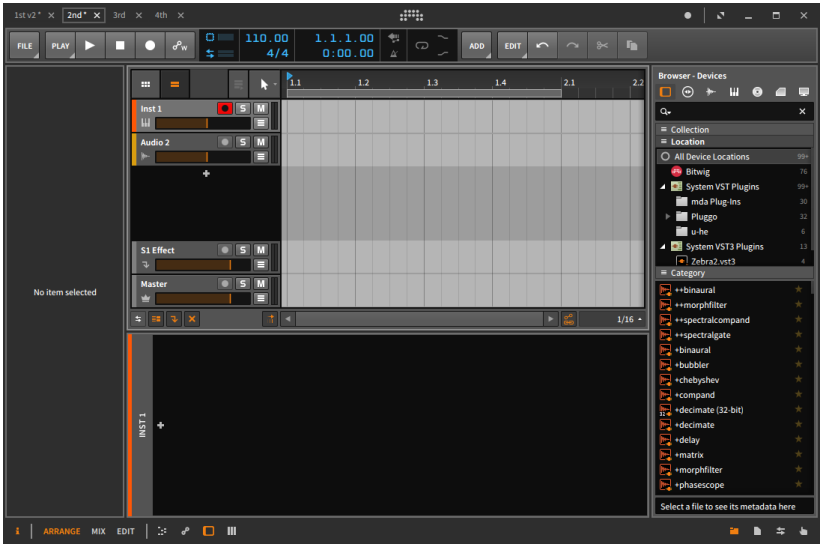
3.2. Intro to Tracks

As we have seen in the Arranger Timeline, Bitwig Studio projects are organized into *tracks*, and clips live on tracks. While clips are critical for expressing your musical ideas, tracks contain the signal paths that take clips out of the computer and into the audible world. Were there no tracks, there would be no sound either.

We will look at the kinds of tracks that exist in Bitwig Studio before discussing a few basic track operations.

3.2.1. Track Types

Bitwig Studio has five types of tracks. The four most common types are present in any new project you create. Here again is a blank new project.



As each type of track has its own designated icon, each track also has its own particular use:

An *instrument track* is denoted with a piano keys icon. The usual purpose of an instrument track is to record and hold note clips that will trigger an instrument and result in audio output.

An *audio track* is denoted with a waveform icon. The usual purpose of an audio track is to record and hold audio clips that will be played back.

A *hybrid track* is denoted with an icon that is half audio waveform and half piano keys. The usual purpose of a hybrid track is to record and hold both note and audio clips. A hybrid track is not present in a new Bitwig Studio project.

An *FX track* is denoted with a downward arrow icon. The usual purpose of an FX track is to receive portions of other tracks' audio output, then mix them together for further processing.

A *group track* is denoted with a folder icon. The usual purpose of a group track is to unite several component tracks (either instrument, audio, hybrid, FX, or other group tracks) into one higher-level track for streamlined mixing and editing. The track's folder icon appears open when its component tracks

are visible and closed when they are hidden from view. A group track is not present in a new Bitwig Studio project.

A *master track* is denoted with a crown icon. One and only one master track is present in each project, making him the king. The purpose of the master track is to sum all signals that are routed to the main audio buss. The master track also provides access to various transport parameters (such as tempo) for the sake of automation.

3.2.2. Creating and Selecting Tracks

As you develop any project, you will almost certainly need additional tracks.

To create a track: go to the *Add* menu and select either *Add Instrument Track*, *Add Audio Track*, *Add FX Track*, or *Add Group Track*.

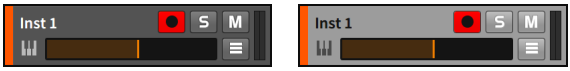
Other ways to create a track include:

› Use the appropriate key command as noted in the *Add* menu.

› Right-click a part of the Arranger where no tracks exist (such as the blank space between the track headers), and then choose the appropriate function from the context menu.

Before you can do anything with a track, it must first be selected, and the track header is key to this. Clicking anywhere else — including in the Arranger Timeline area — selects clips or automation, not an entire track.

When a track is not selected, the background of its header is charcoal gray, and its text and icon are light. When a track is selected, the background of its header is a light silver, and its text and icon are dark.



To select a track: click on the track's header.

When a track is already selected, you can press [UP ARROW] or [DOWN ARROW] to cycle thru the adjacent tracks.

To select or deselect additional contiguous tracks: either hold [SHIFT] and then click on the final track to be included in the selection, or hold [SHIFT] while cycling thru tracks with [UP ARROW] or [DOWN ARROW].

To select or deselect additional individual tracks: hold [CTRL] ([CMD] on Mac) and then click on the track to be added or removed from the selection.

To group tracks: select the tracks you wish to group and then press [CTRL]+[G] ([CMD]+[G] on Mac).

To toggle the visibility of a group track's encapsulated tracks: click on the group track's folder icon.

To unpack and remove a group track: select the group track(s) and then press [CTRL]+[SHIFT]+[G] ([CMD]+[SHIFT]+[G] on Mac).

3.2.3. Edit Functions and Moving Tracks

Once a track is properly selected, several standard edit functions can be used.

To copy a track: select the track and then press [CTRL]+[C] ([CMD]+[C] on Mac).

To cut a track: select the track and then press [CTRL]+[X] ([CMD]+[X] on Mac).

To paste a track: select a track as a reference and then press [CTRL]+[V] ([CMD]+[V] on Mac). The pasted track will be added after the track that was selected.

To duplicate a track: select the track and then press [CTRL]+[D] ([CMD]+[D] on Mac).

To delete a track: select the track and then press [DELETE] or [BACKSPACE].

Other ways to execute the above functions include:

› Select the track and then choose the appropriate function from the *Edit* menu.

› Right-click the track's header and then choose the appropriate function from the context menu.

To move a track: click and drag the track's header vertically.

3.2.4. Track Names

You may have noticed that when a track is created, it is automatically given a name to reflect the type of track it is and its track number.

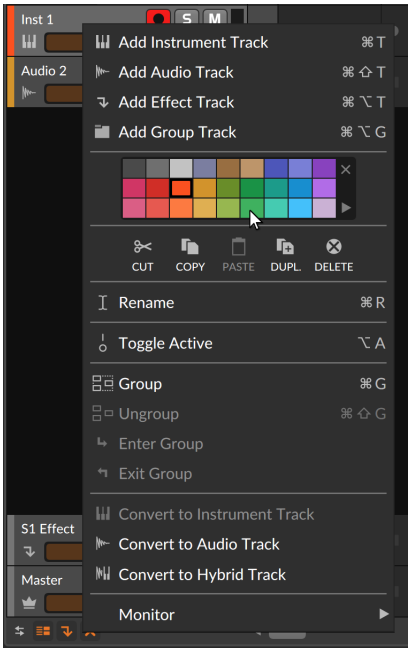
And when a track is moved around, the track number in its name is dynamically updated. By default, tracks are set to automatically name themselves based on certain factors. If you desire, you can override this functionality by renaming the track.

To rename a track: right-click the track's header and then choose *Rename* from the context menu.

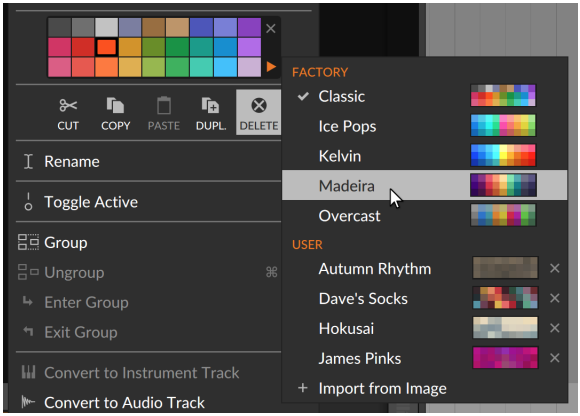
3.2.5. Track Colors and Color Palettes

Each track is assigned a color when it is created. Like the track name, the track color can also be changed.

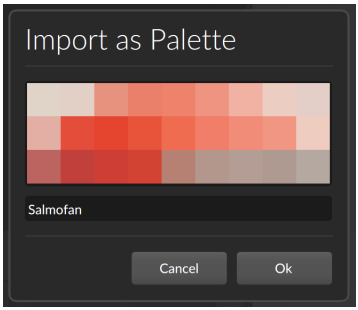
To change the color of a track: right-click the track's header and then select a different color from the palette that appears within the context menu.



To the right of the color palette are two additional options. Clicking the *x* icon clears the color from the current object, opting instead to 'inherit' the color provided. And clicking the right-facing triangle in the bottom corner exposes a menu of factory and user color palettes.



Selecting a different palette makes those colors available, and the most recent palette will be remembered while working on this project. To add a new palette of your own to the *User* category, simply drag a PNG or JPG file from your system's file manager onto the Bitwig window. The image will be resampled and previewed for you.



Change the name as necessary and click *Ok* to add this palette to your library.

3.2.6. Deactivating Tracks

There are various ways to silence a track. One useful option is to deactivate and subsequently (re)activate tracks. When a track is deactivated, not only is its output silenced, but any load it was placing on your CPU is also removed for the time being. From the standpoint of our limited computing resources, deactivating an object is as close as we can get to deleting it — and none of our data are lost in the process.

To deactivate an active track: right-click the track's header and then choose *Activate/Deactivate Track* from the context menu. Or select the track and then press [ALT]+[A] .

Any disabled track is visibly grayed out and certain interface items are removed.

To activate an inactive track: right-click the track's header and then choose *Activate/Deactivate Track* from the context menu. Or select the track and then press [ALT]+[A] .

Note

The deactivate and (re)activate functions can be applied to tracks, devices, and top-level chains/layers of the **Drum Machine**, **Instrument Layer**, and **FX Layer** container devices. And any plug-ins that are deactivated will also stop accruing latency to your project.

Similarly, clips and notes can be muted and unmuted with the same respective key commands.

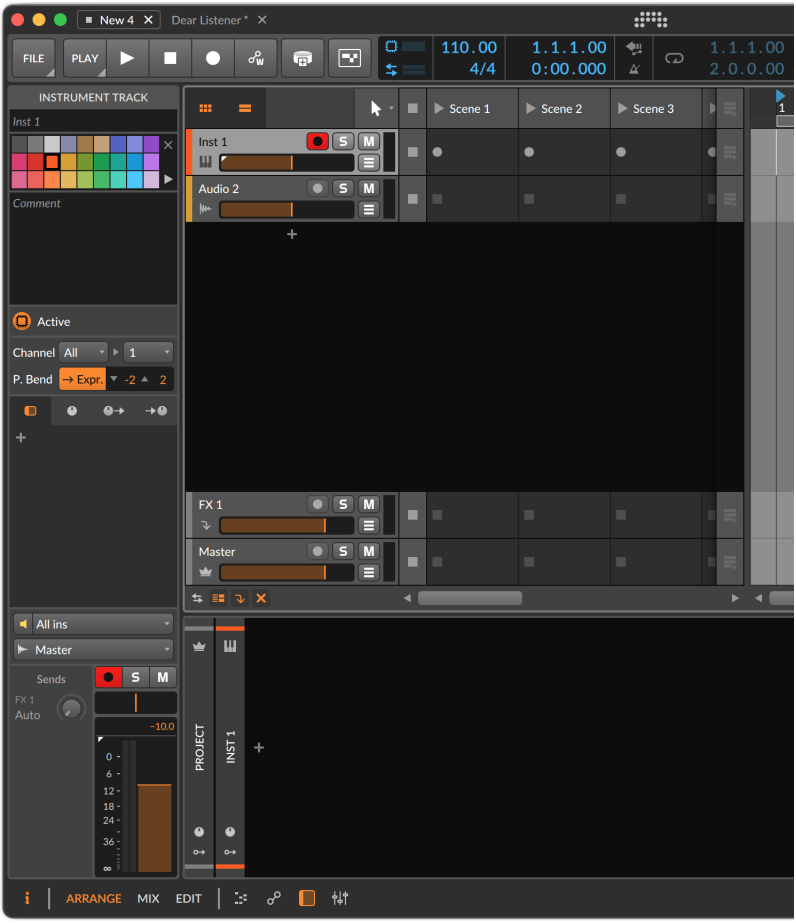
3.3. Meet Inspector Panel

A context menu is available across Bitwig Studio. By right-clicking on an item (practically any object or event), relevant actions that can be taken will be shown along with certain properties of that item. For a fuller list of the available properties, we also have the **Inspector Panel**.

To toggle the visibility of the Inspector Panel: click the view toggle for the **Inspector Panel** (the *i* icon), located in the window's footer.

The **Inspector Panel** follows the active panel's selection, displaying all properties of that selection. As there are many types of items in Bitwig Studio (clips, notes, audio events, devices, automation points, and tracks), the parameters displayed in the **Inspector Panel** can change dramatically depending on what you have clicked on.

By selecting a track, the **Inspector Panel** displays relevant parameters of that track.



The text entry box at top displays the current track name (shown in italics when the name is provided by Bitwig Studio). The color palette is identical to the one from the track header context menu, a *Comment* can be left for viewing here or in the mixer interfaces, and the *Active* toggle controls whether the selected track is currently running or deactivated.

Plenty of other parameters are shown within the **Inspector Panel**, including nearly all of the meters and controls from the track header. And we will get to the parameters that are now unfamiliar in the appropriate sections of this document.

The main idea is that the **Inspector Panel** is an ideal way to see all the parameters of most selected items. A context menu is also available for most items and window areas. Going forward, we will primarily use the **Inspector Panel** for viewing or altering parameters and the context menu for executing functions. So this isn't "goodbye" to either option, but rather "nice to meet you."

4. Browsers in Bitwig Studio

In some ways, the best analogy for a digital audio workstation is a traffic cop. A primary task of the modern DAW is getting your computer and software to play well with everyone, including any controllers, plug-ins, and audio equipment you may have. The hardware side of this is a bit more obvious and flashy — working with MPE controllers and their fluid note streams; offering our controller API for dynamic and customized interactions between hardware and software; multitouch support, including alternate workflows for editing, mixing, and performing; various playback sync options; specialized display profiles for two or three monitor setups; and natively speaking control voltage (CV) for Eurorack modules and beyond.

While the software side might seem like the easier part of the equation, it includes all of your files. And the list of file formats you might browse is only growing. As of today, it includes: WAV, AIFF, MP3, FLAC, OGG, OPUS audio files (and more); WT wavetable files; MULTISAMPLE, SFZ, and SoundFont 2 (SF2) multisample files; CLAP, VST 2, and even VST3 plug-ins; BWPRESETS, H2P, as well as FXP, FXB, VSTPRESET, and any vendor-specific formats that CLAP preset discovery offers; BWIMPULSE files and any other audio for use as convolution impulse files; BWCLIP files, MIDI files, DAWPROJECT files (for project interchange with other music programs; more information here [https://www.bitwig.com/support/technical_support/dawproject-file-format-faqs-62/]), and other sequence formats with some import support (FLP and ALS), as well as BWPROJECT and BWTEMPLATE files; and Bitwig's internal devices, modulators, and modules.

The purpose of Bitwig Studio's browsers is to connect your current idea to a relevant musical materials from that mountain of files and formats. This means providing clear ways to narrow a large pile of results, and also nudging you back on track when you might be looking for something in the wrong place. And as with any search, you will find a great sound at the wrong time so making it easy to file things away for later is important too. In short, it's better to save time each day, both for today and tomorrow.

We say "browsers" plural because there is the omnipresent **Browser Panel** anchored to the right side of the window, as well as the dynamic **Pop-up Browser** that appears when a plus icon (+) or folder button is clicked. Their structures are largely identical, and their few differences will be noted.

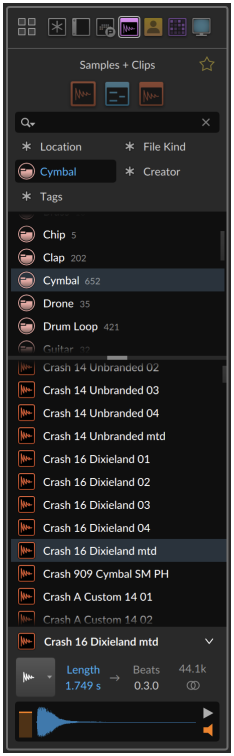
One procedural note: key commands will be mentioned all thru this chapter, and they reference Bitwig's *Default keyboard mappings*. If you are working with your own key commands, most functions can be found and mapped as you like (see section 0.2.2.4).

So let's dive into browsing. We'll generally look at features in isolation — sources, filters, key commands, autocomplete suggestions, customization options, and more — but when working on music, you will use these tools together. Which is great because then you'll spend less time selecting sounds and more time bringing them to life.

4.1. All Sources

Browsing in Bitwig Studio is centered around *sources*. Each source is just a way to group searchable content, providing windows thru which you can approach your files. When any browser is loaded, a source is selected.

In the **Browser Panel**, the current source is shown by the title above the various filters. In this image, *Samples + Clips* is the selected source.



And in any variation of the **Pop-up Browser**, the area above the filters also shows the current source along with its icon. Shown here is the *All*

Instruments source and its keyboard icon, hinting that note input will be required.

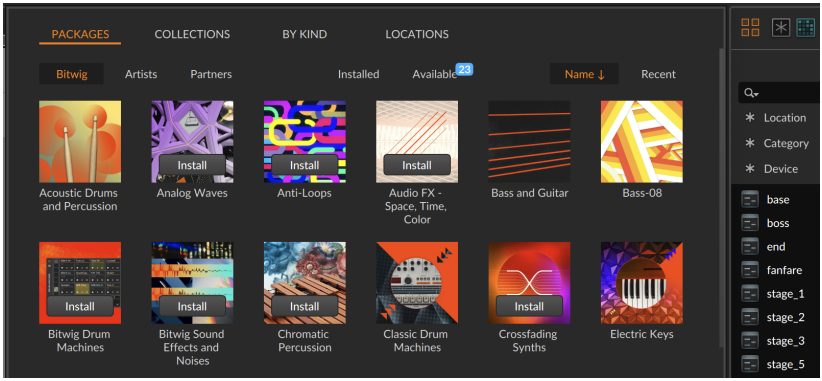
In both of these views, the top left corner holds a button (with an icon of four little squares) for switching to the *All Sources* page, where all available sources can be seen. Clicking on any source returns to the browser with that source selected, so every available source can be browsed from the *All Sources* page. Or press [CTL]+[0] ([CMD]+[0] on Mac) to toggle between the *All Sources* page and the regular browser view.

We will look at each of the four tabs in order. And for now we will use the perspective of the **Browser Panel**, where having no context means that everything is always available.

Just know that each source only appears once, so knowing the concept of each tab will help you know where to look later.

4.1.1. Packages Tab

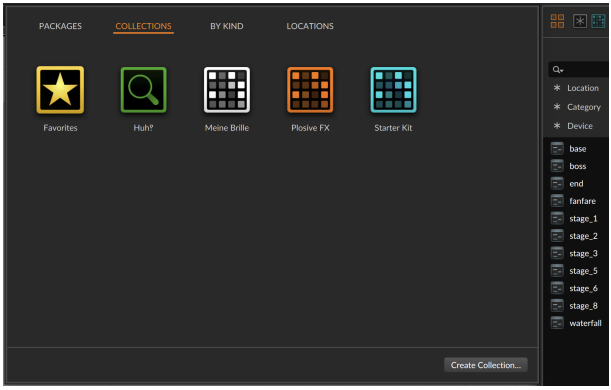
The *Packages* tab offers a source for each sound package you have from Bitwig, as well as a way to acquire content you haven't installed yet.



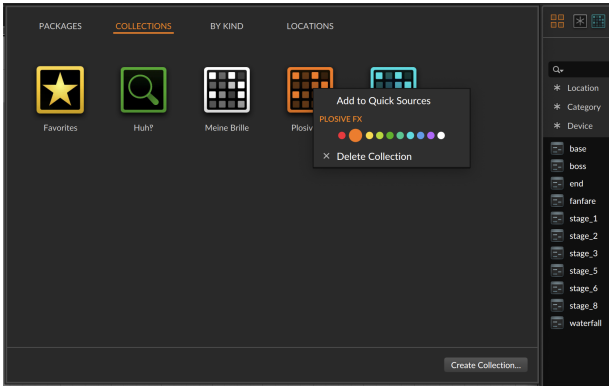
Unique to the *Packages* tab is a row of view and sort options, all shown as small text buttons just above where the packages start. They are identical to those in the *Packages* tab of the **Dashboard** (see section 0.2.3).

4.1.2. Collections Tab

The *Collections* tab displays all user-saved groups. This definitely includes *Favorites*, which contains every item you have marked as a favorite. And any fixed *collections* (with the colorful grid icons) of yours will be here too, as well as dynamic *smart collections* (with the magnifying glass icons) that you might have created.



Right-clicking on any collection or smart collection provides a context menu with various options, including to change the color of its icon or to *Delete Collection*.

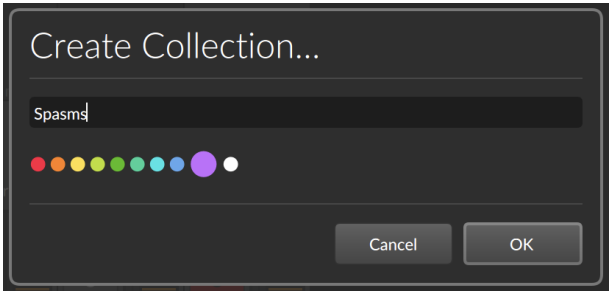


To rename a collection or smart collection: click on its name, which will make the text editable.

Both collections and smart collections are ways for you to organize your content. But contrary to their names being so similar, they represent two distinct concepts.

A *collection* starts empty and waits for you to insert content into it. In this way, the *Favorites* source is a special collection. For some users, this single collection will be enough, but you can create others.

To create a collection from the All Sources page: on the *Collections* tab, click the *Create Collection...* button in the bottom right corner of the window. Then choose a name and color for the collection.

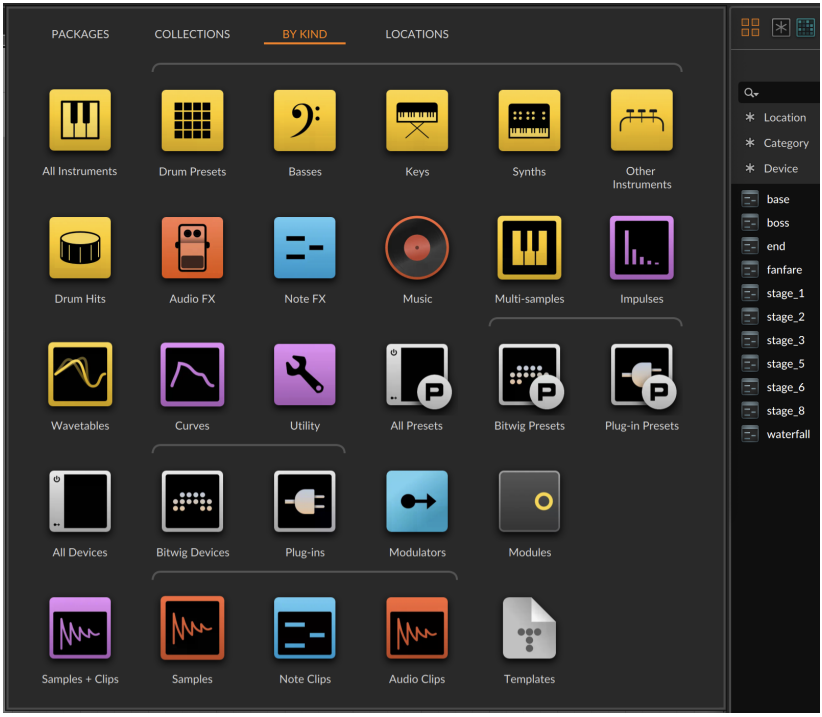


Items can be added to the collection either from the results list (see section 4.2.3), from the file area (see section 4.2.4), or from the Quick Sources (see section 4.3.1).

A *smart collection* is a saved set of filters that can be viewed as a source. As it doesn't contain individual items but rather search parameters, its content will be dynamic (see section 4.3.4).

4.1.3. by Kind Tab

The *by Kind* tab offers sources organized by file type — and sometimes by category as well. Since these sources are always available, this list is the longest to start with.



All Instruments contains all instrument devices, plug-ins, and presets. It is the parent source of these individual sound-descriptive sources:

› *Drum Presets* contains devices, plug-ins, and presets in known drum/percussion categories (including *Clap*, *Cymbal*, *Drum Kit*, *Hi-hat*, *Kick*, *Percussion*, *Snare*, and *Tom*).

› *Basses* contains devices, plug-ins, and presets in known bass categories (including *Bass* and *Synth Bass*).

› *Keys* contains devices, plug-ins, and presets in known keyboard categories (including *Electric Piano*, *Organ*, and *Piano*).

› *Synths* contains devices, plug-ins, and presets in various synthesizer-type categories (including *Bell*, *Chip*, *Drone*, *Ensemble*, *Lead*, *Monosynth*, *Pad*, *Pipe*, *Plucks*, *Sound FX*, *Synth*, and *Synth Bass*).

› *Other Instruments* contains devices, plug-ins, and presets in other various categories (including *Brass*, *Ethnic*, *Guitar*, *Mallet*, *Orchestral*, *Strings*, *Vocal*, and *Winds*).

Drum Hits is a hybrid source comprising e-drum instruments, drum sound presets, and sample files identified as individual drums.

Audio FX contains all audio FX devices, plug-ins, and presets.

Note FX contains all note FX devices, plug-ins, and presets.

Music contains all audio files from your chosen music locations.

Multi-samples contains all MULTISAMPLE, SFZ, and SF2 files (available to both the **Sampler** device and Grid module), either in the Bitwig library or from your chosen sound content locations.

Impulses contains all BWIMPULSE files (used by the **Convolution** device), either in the Bitwig library or from your chosen sound content locations.

Wavetables contains all WT files (used by the Grid/Polymer **Wavetable** module and also the **Wavetable LFO** modulator & Grid module), either in the Bitwig library or from your chosen sound content locations.

Curves contains all BWCURVE files (used by any of the various "curve"-based devices, modulators, and modules), either in the Bitwig library or from your chosen sound content locations.

Utility contains devices and presets in the *Utility* categories, as well as devices in other special function categories (*Analysis*, *Container*, *Hardware*, *MIDI*, and *Routing*).

All Presets is the parent source of these sources:

› *Bitwig Presets*, for BWPRESET files in the Bitwig library or from your chosen sound content locations.

› *Plug-in Presets*, for H2P files, as well as FXP, FXB, VSTPRESET, and any vendor-specific formats that CLAP preset discovery offers.

All Devices is the parent source of these sources:

› *Bitwig Devices*, for our internal devices within the Bitwig Studio application.

› *Plug-ins*, for CLAP, VST 2, and VST3 plug-ins, installed in one of your chosen plug-in locations.

Modulators are available in the **Browser Panel** for loading Bitwig's internal modulators. It allows dragging one or more modulators into a device's modulator pane.

Modules are available in the **Browser Panel** for loading Bitwig's internal modules. It allows dragging one or more modules into a Grid device's editor window (see'.

Samples + Clips is a parent source for all audio and timeline materials, including these sources:

› *Samples*, for all audio files, either in the Bitwig library or from one of your chosen sound content locations.

› *Note Clips*, for all note-based BWCLIP files as well as MIDI files, either in the Bitwig library or from one of your chosen sound content locations.

› *Audio Clips*, for all audio-based BWCLIP files, either in the Bitwig library or from one of your chosen sound content locations.

Templates is available in the **Browser Panel** for loading BWTEMPLATE files, either in the Bitwig library or from one of your chosen sound content locations.

4.1.4. Locations Tab

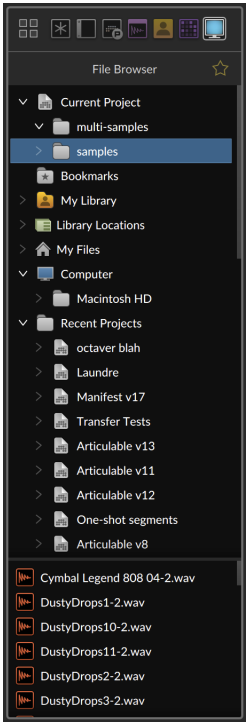
The *Locations* tab combines sources tied to particular disk locations and a few special sources.

Everything is a catch-all source that is the first available source in all browsers. It is useful for searching across all applicable items at once.

My Library contains files found in your local Bitwig user library.

Each of your chosen sound content, music, and plug-in location folders appear as individual sources here. Both sound content and plug-in locations are shown with a folder icon, and music locations appear as vinyl records. You can right-click any of these sources to *Remove* the location, and you can click any of the bottom three *Add* buttons to create a new location and source.

And the *File Browser* is available in the **Browser Panel** as a view for browsing your files and computer generally.



A word on each top-level entry.

› *Current Project* allows you to unfold the file structure of the current project, giving access to any of the contained files (as shown above).

› *Bookmarks* is a place for any disk folder locations you have saved from within this *File Browser* view.

To add a bookmark a folder for the File Browser source: navigate to the folder via the *File Browser*, then right-click on the folder and select *Add as Bookmark*.

› *My Library* provides access to your local Bitwig user library folder.

› *Library Locations* contains all folders added as sound content, music, and plug-in locations.

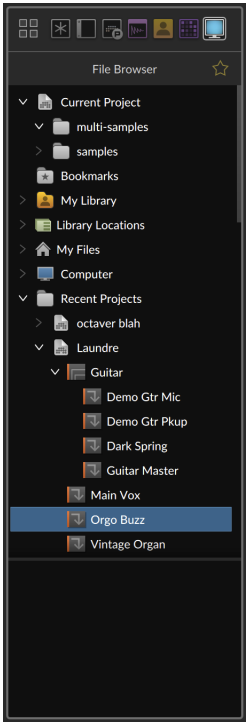
› *My Files* provides access to your computer user's home folder.

› *Computer* provides access to all disks attached to your computer.

› *Recent Projects* offers recently opened projects in order, starting with the one currently open.

Finally, the *File Browser* has two super powers when it comes to Bitwig project files. The first is that you can drag a full project from the *File Browser* into your current project. This will create a group track for that project's master track, with all possible content being inserted within it.

Second and unique to the *File Browser* is that projects can be unfolded here to see the individual tracks (and group tracks can be further unfolded as well).

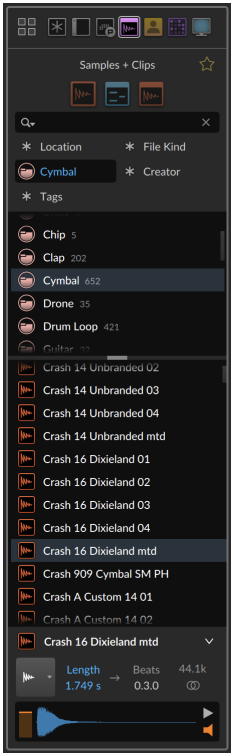


To import one or more tracks from another project: locate the project from the *File Browser* (in the **Browser Panel**). Then unfold the project, select one or more tracks, and drag them into the current project.

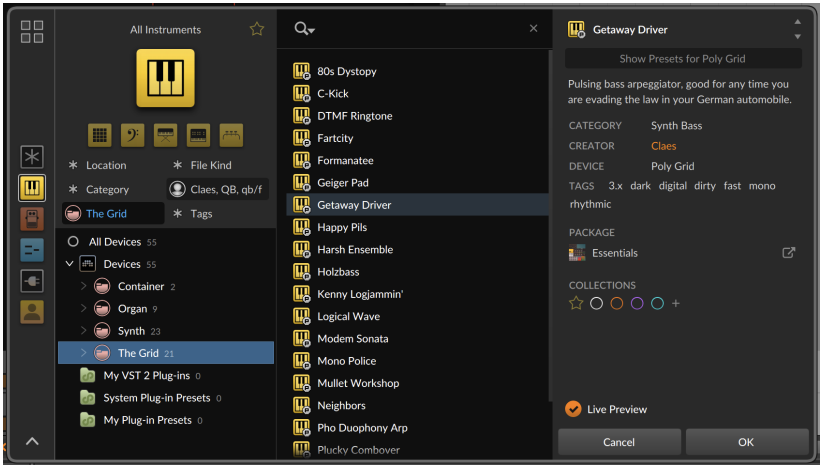
4.2. Common Browser Elements

Whether using the anchored **Browser Panel** or the dynamic **Pop-up Browser**, most elements are shared by both browsers, albeit with different orientations.

Living along the left edge of the window, the **Browser Panel** is vertical and thin by design.



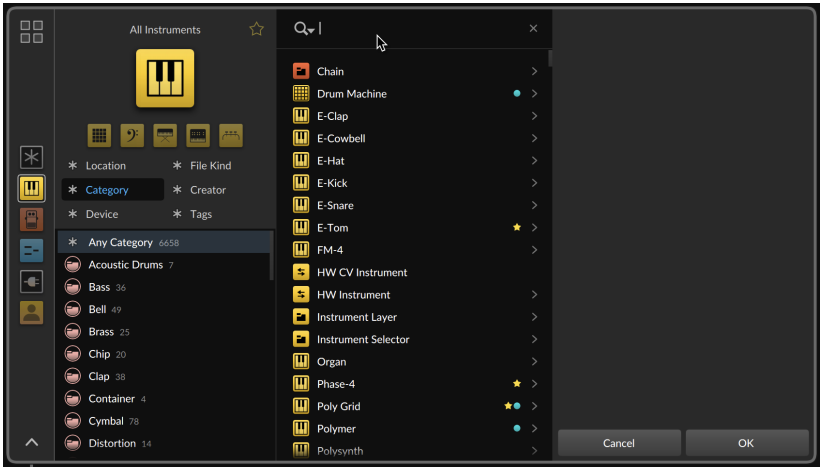
Made to appear only when called, the **Pop-up Browser** has a larger, horizontal layout, like most computer screens.



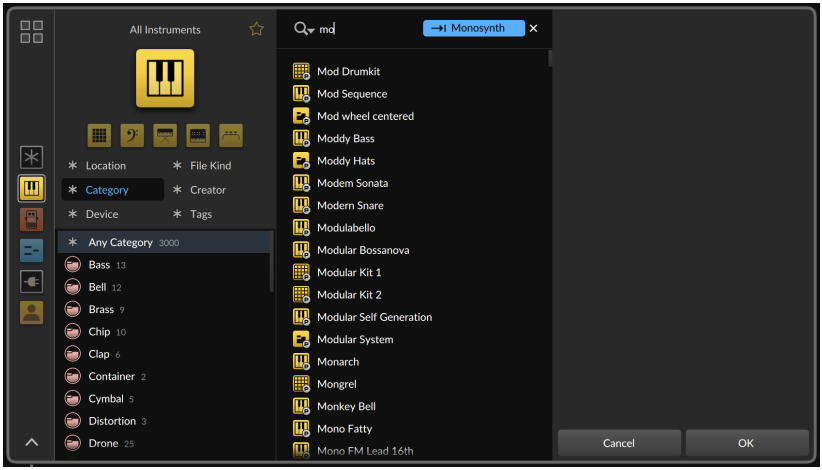
For this section, we will take the perspective of the **Pop-up Browser** as it has a few additional touches.

4.2.1. Search Field

While *browsing* can feel like the opposite of *searching*, Bitwig Studio's browsers allow both workflows to go hand in hand. The search field inside each browser is marked with the magnifying glass icon.

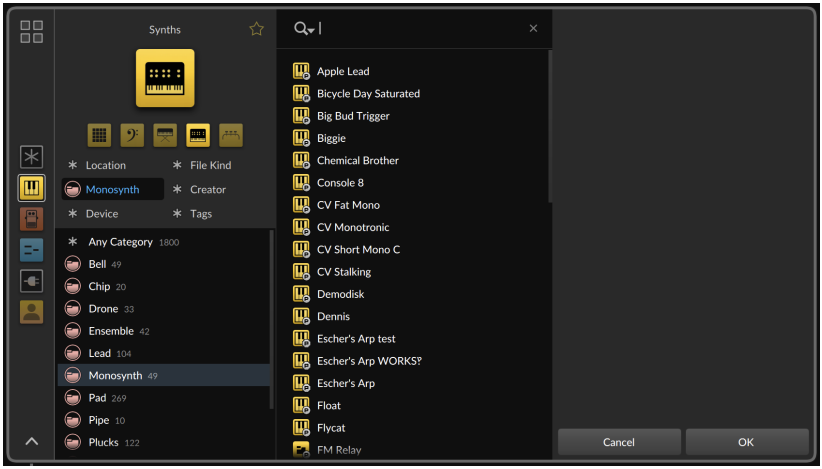


Often times, opening the browser puts keyboard focus on the search field immediately so just start typing. And actually, that might be the best advice about the browser: whatever you are thinking, *just start typing*. It often works out because as you start typing, any matching sources, collections, creators, tags, and more will be offered as suggestions in a blue "autocomplete" button that appears.

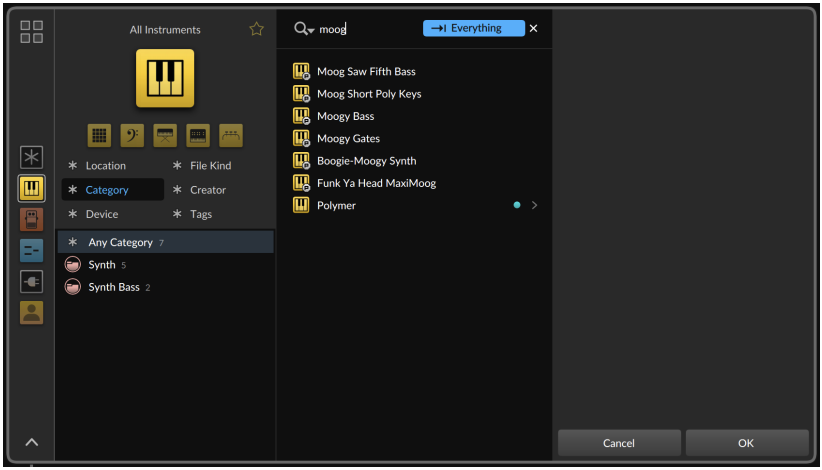


To accept a browser suggestion: either press the [TAB] key, or click the blue button.

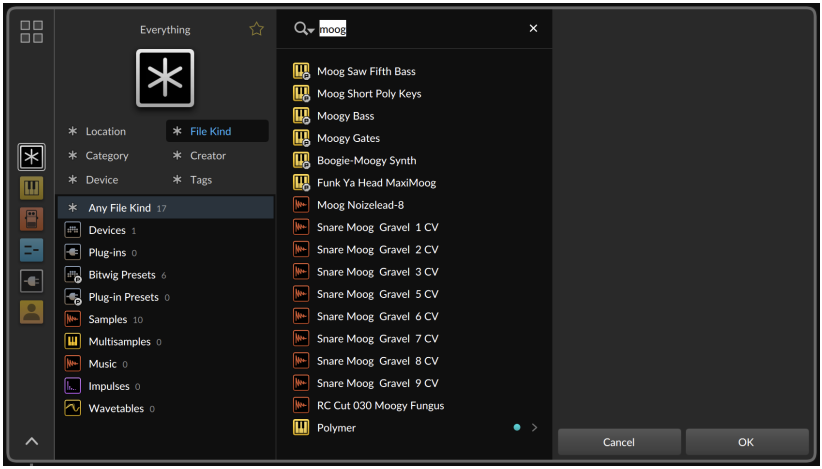
This will switch to that source or add the filter offered, etc. As you can see below, the *Category* is now set to *Monosynth*, and the search field remaining in focus so you can just start typing (again).



If your search yields few results, the browser will suggest switching to the *Everything* source, where more content is likely available.

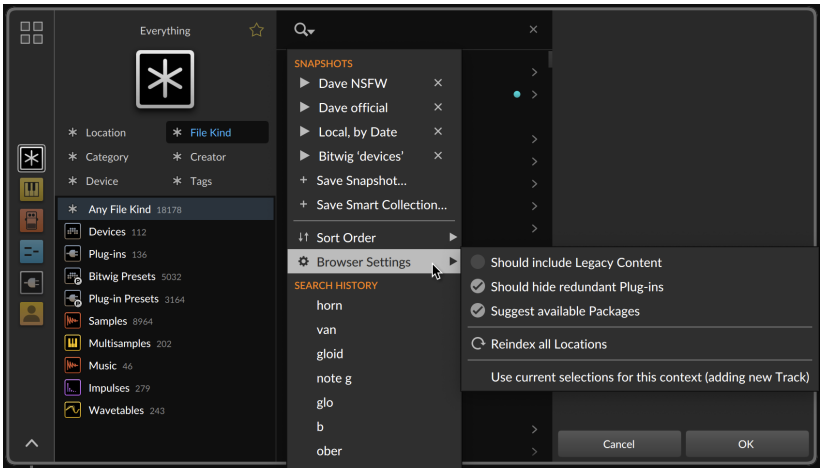


Making the switch to the *Everything* source will also preserve your search so you can see the change in your result list immediately.



Whenever focus is in the search field, pressing [DOWN ARROW] will move focus into the results list, which selects the first item if nothing was already selected. And if focus is on the first result item, pressing [UP ARROW] will return focus to the search field (in the **Pop-up Browser**, [PAGE UP] also works). Or press [S] no matter where focus is in the browser to return to the search field and continue typing.

Finally, the magnifying glass at the left edge of the search field is also clickable, offering various options, including general *Browser Settings*.



Among the various *Browser Settings* is the option to have the browser *Suggest available Packages*. On by default, this option will offer a

notification in the bottom of the browser when your search terms match a package that is available and not yet installed.

4.2.2. Filters Area

Together with searching, applying filters will help narrow your search results to a manageable and thematic pile. All available filters are shown below the current source, and clicking any filter will select it, unfolding its entries in the space below.

Clicking an entry within a filter activates it. Multiple filter entries can also be selected in the standard way — by [CTL]-clicking ([CMD]-clicking on Mac) to add/remove additional entries.

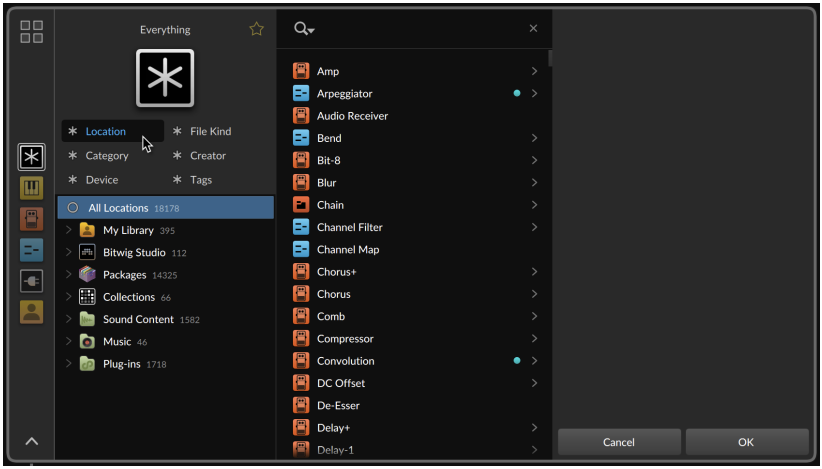
When a filter is active, the name of the filter is replaced with its current selection(s). So regardless of which filter is currently open, an active filter will always be visible. And hovering over an active filter header also presents an *x* icon on its right edge for easily clearing it. (Pressing [X] will also clear the currently selected filter.)

As the **Browser Panel** is oriented vertically, pressing [CTRL]+[ALT]+[UP ARROW] and [CTRL]+[ALT]+[DOWN ARROW] ([CMD]+[ALT]+[UP ARROW] and [CMD]+[ALT]+[DOWN ARROW] on Mac) will move to the previous or next filter available. In the horizontal **Pop-up Browser**, this translates to [CTRL]+[ALT]+[LEFT ARROW] and [CTRL]+[ALT]+[RIGHT ARROW] ([CMD]+[ALT]+[LEFT ARROW] and [CMD]+[ALT]+[RIGHT ARROW] on Mac). These commands will work anywhere in the browser, including from the search field.

Finally, in the **Pop-up Browser**, the width of the source and filter area is resizable and will be remember for similar contexts (for example, the width you used when browsing for devices versus clips, and so on).

4.2.2.1. Location

The *Location* filter is practically always visible, offering both disk locations and virtual ones. Pressing [L] from most browser locations will move focus to this filter.



As with all filters, the top-level items can always be clicked on to only show results from that entire location, or each entry can be unfolded for additional specificity.

My Library points to your local Bitwig user library, including subgroups that correspond to its folder structure.

Bitwig Studio points to any relevant internal content from inside the application, including subgroups of *Devices*, *Modulators*, and Grid *Modules*.

Packages points to installed package content, including subgroups for each individual package (and its folder structure) that contains relevant items.

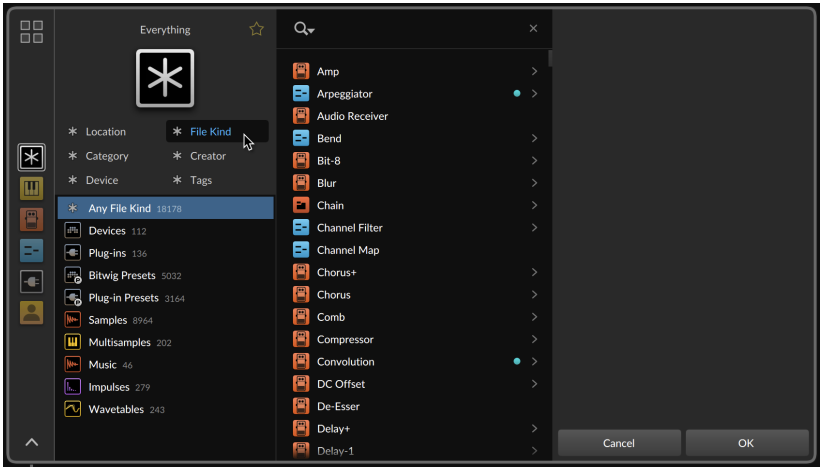
Collections points to all of your user collections (including *Favorites*). When unfolded, all collections will be shown regardless of whether they have relevant content or not. This makes it possible to select one or more items from the result list and drag them into a collection via the *Location* filter.

Last comes with *Added Locations Group*. It is actually three top-level items, allowing you to select or unfold your *Sound Content*, *Music*, and *Plug-ins* locations for any relevant items.

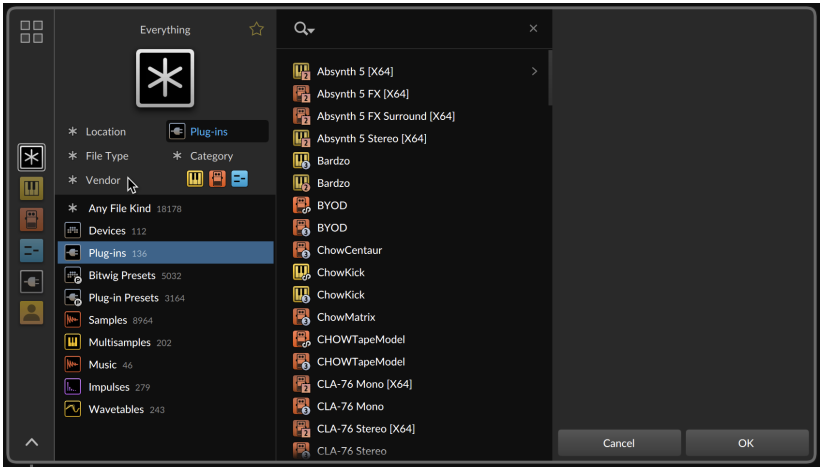
4.2.2.2. File Kind

The *File Kind* filter allows isolating the results list to a particular kind(s) of file. Pressing [F] from most browser locations will alternate focus

between this filter and the more specific *File Type* filter, discussed below.



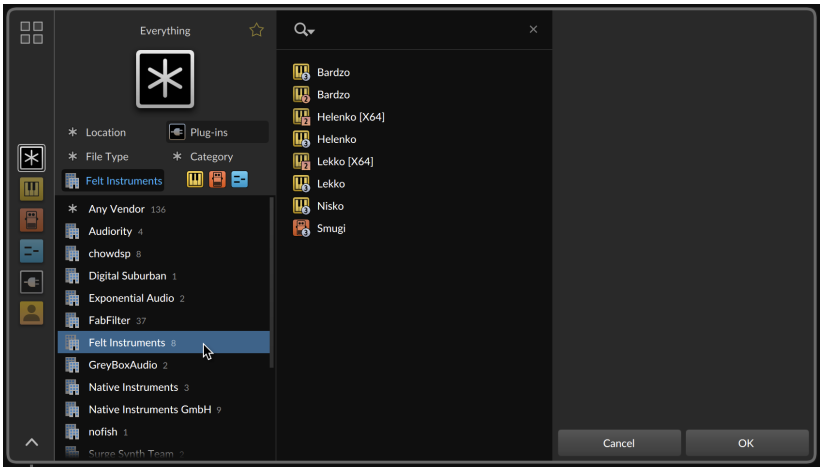
All of these kinds are also available as sources of their own and were described above (see section 4.1.3). With certain *File Kind* selections, additional filters will also appear.



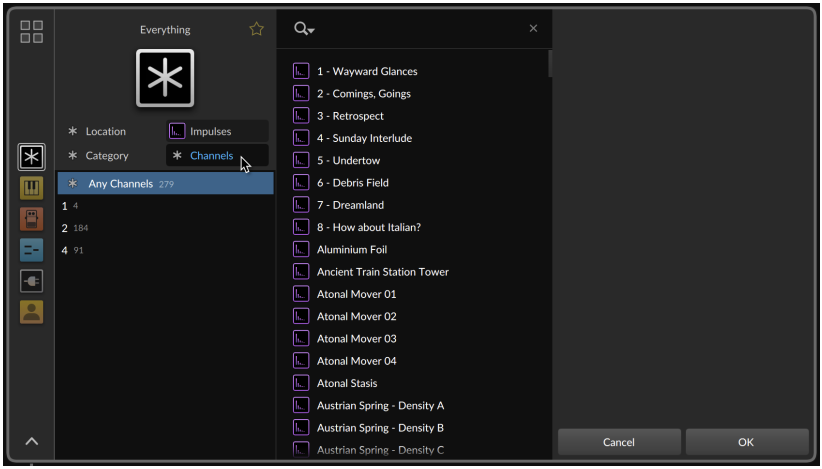
Shown above, a special icon chooser has appeared at the end of the filter section. For any device- or preset-based selection, these icons will appear so you can limit your results to just instruments (yellow), audio FX (red), or note FX (blue).

Also shown above, a *File Type* filter will appear for any *File Kind* selection that includes more than one format of file. When both filters are present, pressing [F] from most browser locations will alternate between selecting the *File Type* and *File Kind* filters.

And a special *Vendor* filter will appear for *Plug-ins* and other device-based selections. Pressing [V] from most browser locations will switch to this filter, where you can narrow your search by selecting one (or more) plug-in manufacturer.

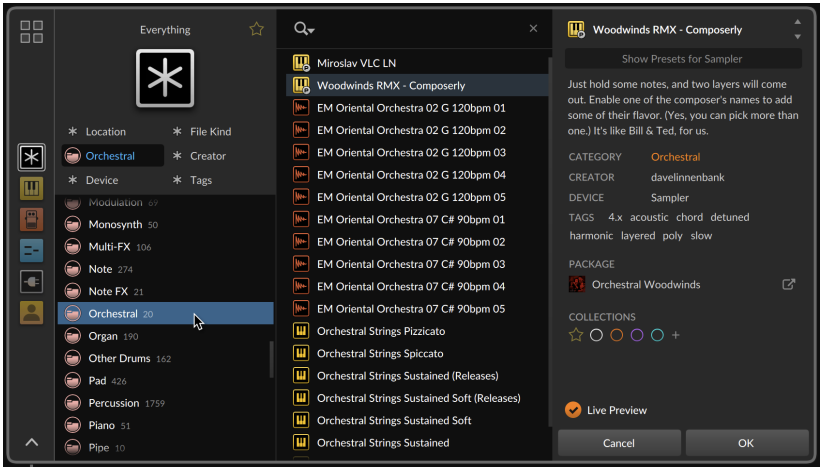


Finally when *Impulses* is selected for *File Kind*, an additional *Channels* filter appears for limiting your results to files with a certain number of audio channels.



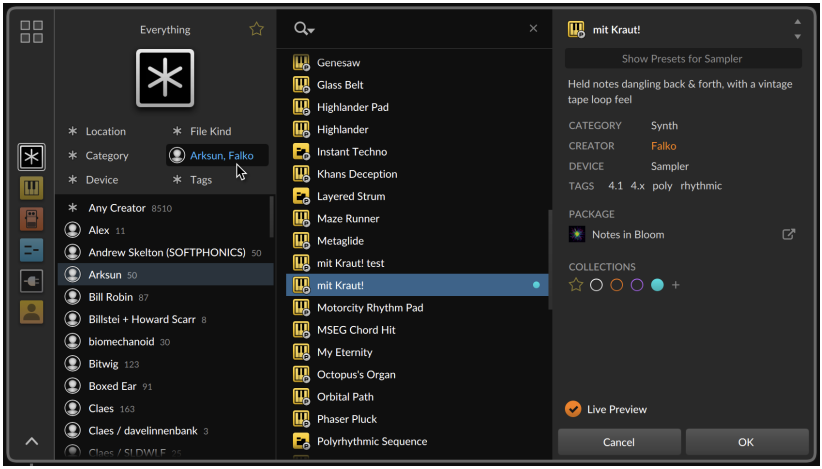
4.2.2.3. Category

The *Category* filter allows limiting results to a specific category. Pressing [C] from most browser locations will alternate focus between this filter and the *Creator* filter.



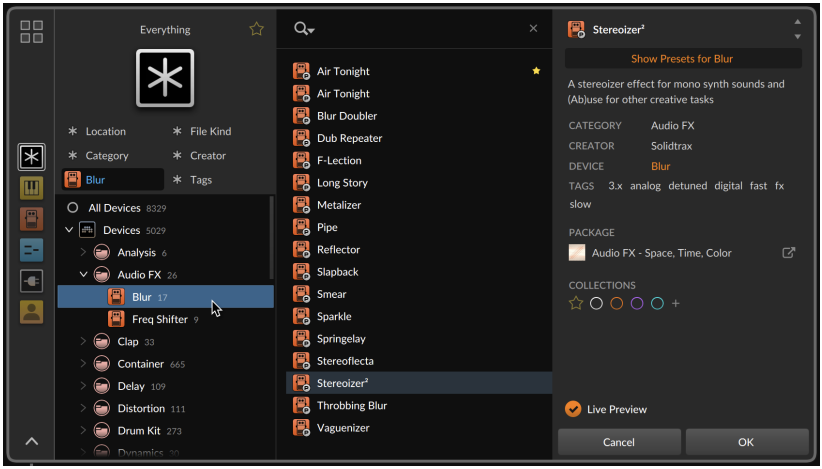
4.2.2.4. Creator

The *Creator* filter allows limiting results to those made by a certain person. Pressing [C] from most browser locations will alternate focus between this filter and the *Category* filter.



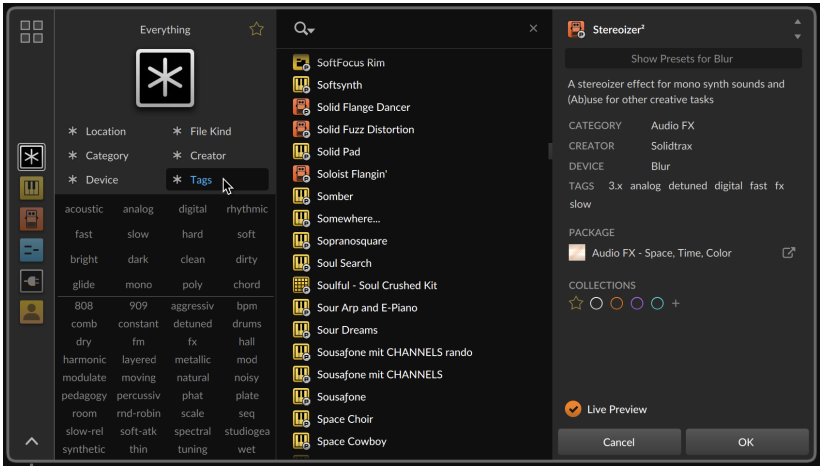
4.2.2.5. Devices

The *Device* filter is useful when searching presets, limiting your search to only presets made with certain devices (or categories of devices). Pressing [D] from most browser locations will move focus to this filter.



4.2.2.6. Tags

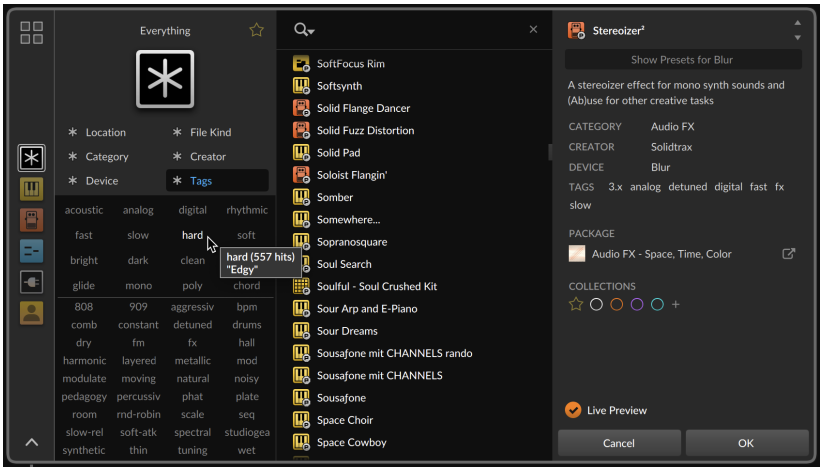
The *Tags* filter is special, helping narrow your search with various assigned keywords. Pressing [T] from most browser locations will move focus to this filter.



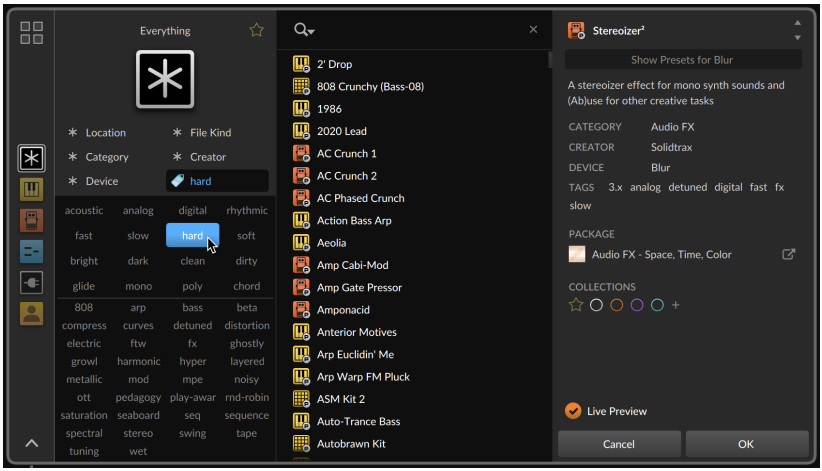
With a potentially limitless number of tags in use, the interface here uses four columns to show much more at a time. The top 16 slots are fixed with a consistent layout, using tags from Bitwig's content that

may apply to all types of presets. This full top block of 16 will always be shown. Below the divider, the space is filled with as many relevant tags as possible, picking the most common ones in use for your current search, and then sorting them alphabetically.

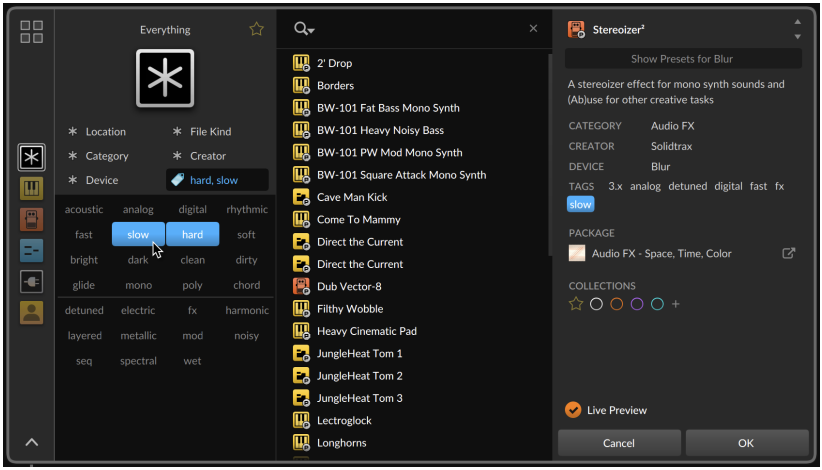
In this abbreviated layout, you must hover over tags to see how many results match that tag, as well as a short, subjective description of the tag's meaning.



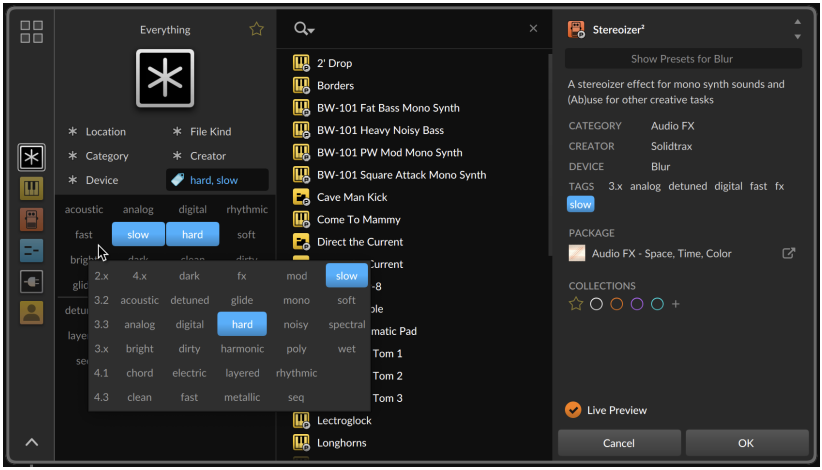
Once a tag is clicked on and selected, your results list will have changed and so will the bottom section of tags.



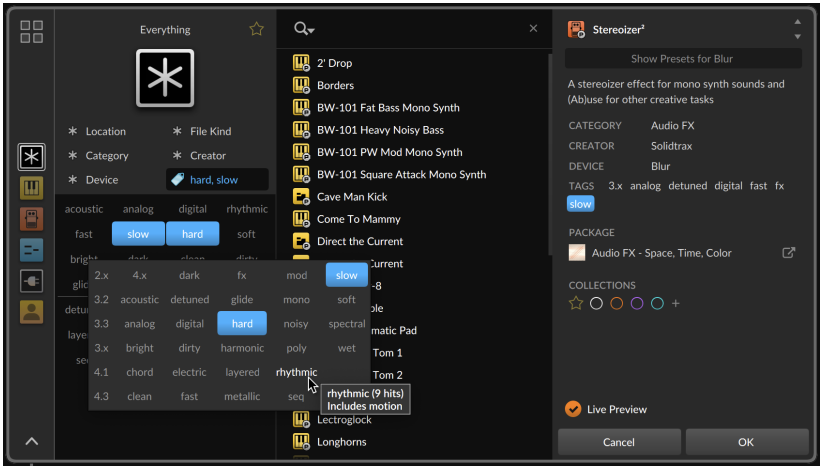
Additional selections will further narrow the results and the tags available.



And in case this view is too limited, or you would just rather see all tags in alphabetical order, you can right-click anywhere in the tags area for a special pop-up menu.

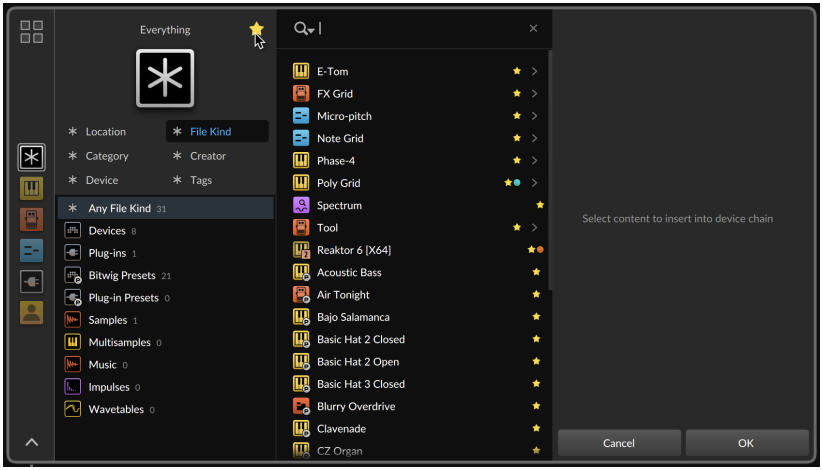


Hovering over items will even work the same in this view.



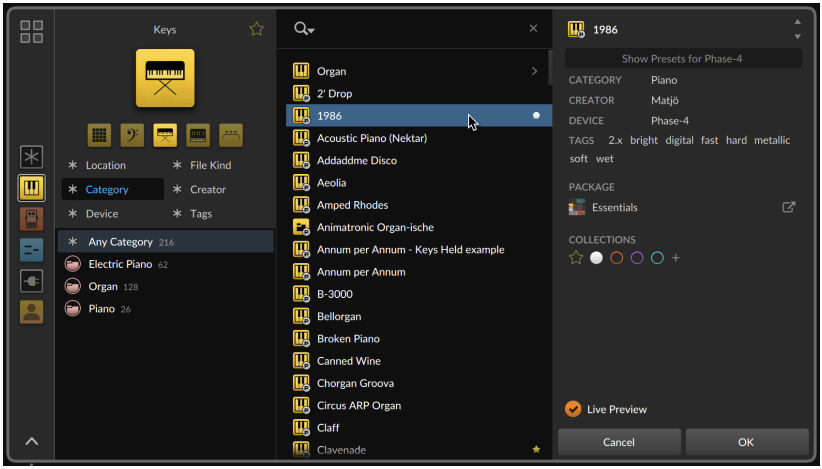
4.2.2.7. Favorites

The special *Favorites* filter is always available via the hollow star to the right of the current source name. Clicking it shows only results that you have also marked as your favorites.



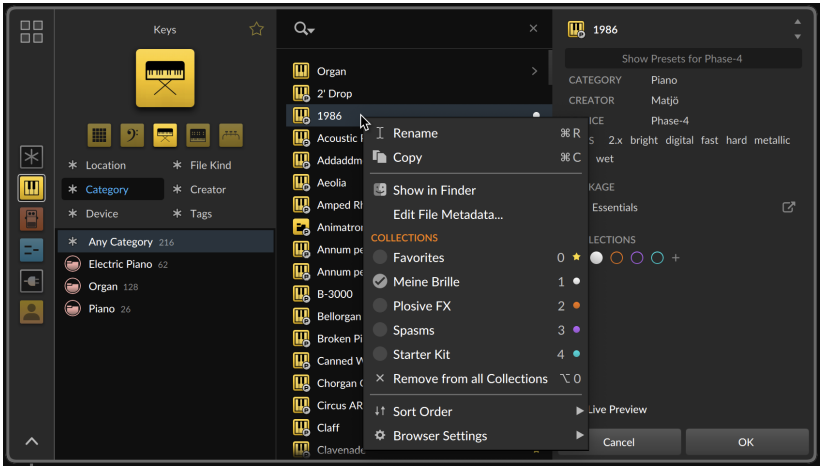
4.2.3. Results List

A list of results is always visible in the browser, presenting each item that matches your current search terms and filters with an icon and its name.



Any small colored circles represent collections that that item belongs to (or a star, in the case of it being a favorite). The [DOWN ARROW] and [UP ARROW] keys provide the simplest navigation.

Right-clicking any item also reveals a useful context menu.



Clicking one of the listed *Collections* will toggle the item's status, either adding it to or removing it from that collection. But the numbers beside each collection represent a key command that can be used without entering the context menu.

To add/remove an item from a collection from the results list: select the desired item(s), and press the number key associated with the collection.

So in the **Pop-up Browser** with its *Live Preview* mode, or when going thru samples in either browser with the *Auto-Preview* option enabled, filing items is as easy and pressing [DOWN ARROW], instantly auditioning a preset or sound, pressing a number to send it to a collection ([0] to mark it a favorite), and then pressing [DOWN ARROW] again.

In the *Sort Order* submenu are options for how the results list should be sorted. Options include:

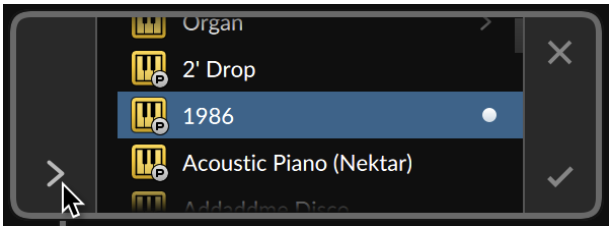
› *by Kind / Name* (default), which separates various file kinds (such as Bitwig devices, then plug-ins, then Bitwig device presets, and so on) and alphabetizes each kind.

› *by Name*, which alphabetically sorts the entire list, regardless of file kind.

› *by Date*, which sorts files by their modification date, with most recently touched files coming first.

In the *Browser Settings* submenu are preferences for omitting content from the results list, including whether the results *Should include Legacy Content* and whether the list *Should hide redundant Plug-ins*, which is based on your settings (see section 0.2.2.5).

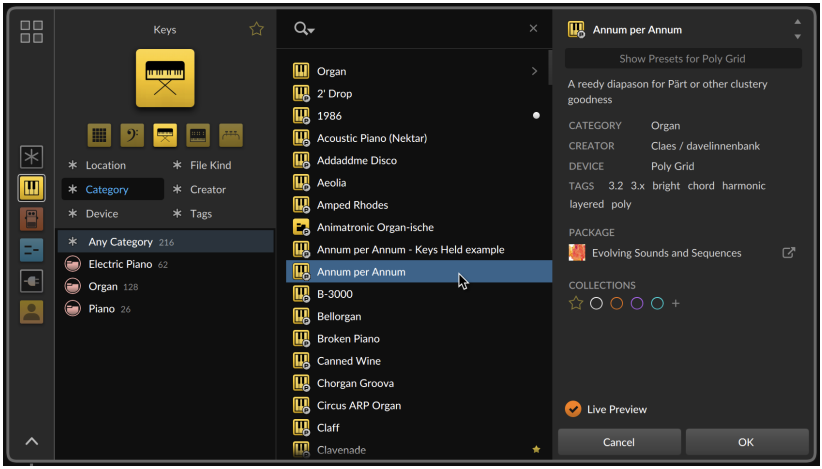
Finally, the **Pop-up Browser** also has a unique collapsed view, available by clicking the folding triangle frame in the bottom left corner.



All key commands listed above (for moving thru the results list and even adding items to *Favorites* and other collections) still work in the small view, making it a nice way to try out content while keeping the rest of the Bitwig window visible.

4.2.4. File Area

When an item is selected, the file area on the right tells you about it.



Many options and much information are available here.

› Beside the item's icon and name are a pair of up and down arrows, which can be clicked or tapped to move to the previous or next result, respectively.

› All labeled data (shown below the item description) are clickable, effectively toggling those filters.

› When a device (that has presets) or a preset itself is selected in the **Pop-up Browser**, a button to *Show Presets for* that device appears near the top of the area. Clicking it will select the target device in the *Device* filter, limiting your current search to only presets from this device. Clicking the button again will return you to your previous search.

In both browsers, devices that have accessible presets will show a small greater-than symbol (>) at the right edge of their result list entry (see **Organ** in the image above). Clicking that icon or pressing [RIGHT ARROW] will toggle on the *Show Presets for* mode. And pressing [LEFT ARROW] will then exit that mode.

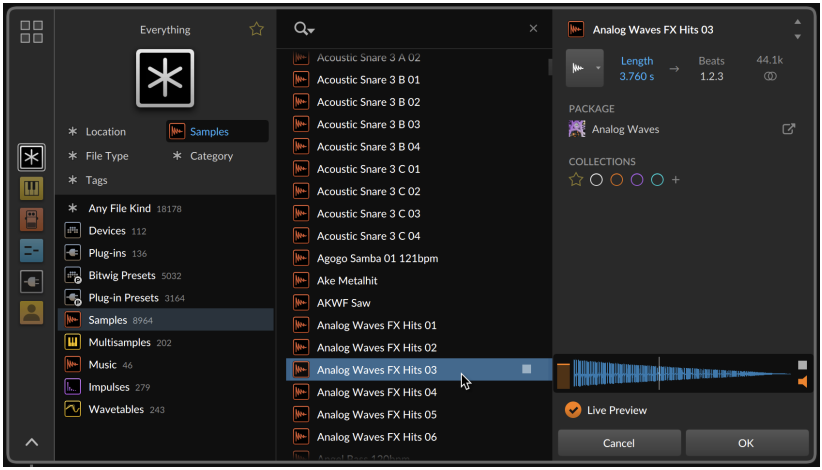
› In the **Pop-up Browser**, the width of the file area is resizable and will be remembered for similar contexts (for example, the width you used when browsing for devices versus clips, etc.).

› In the **Pop-up Browser**, entries for the item's *Location* (or *Package*, etc.) and clickable buttons for its *Collections* are always present. In the **Browser Panel**, these items can be enabled from the *Browser Settings* menu.

› In case space is tight in the **Browser Panel**, the folding triangle frame to the right of the file name can be clicked to fold away the rest of the file area, except for the preview player…

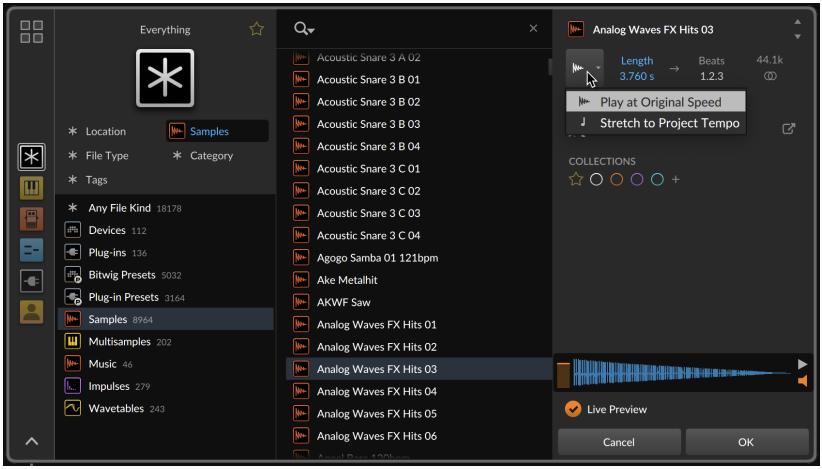
4.2.4.1. Preview Player

For audio files, clips, and other timeline-friendly content, special information about the file will be present, as well as a preview player at the bottom of the browser.



The preview player offers a volume fader at the left edge, as well as a play/stop button on the right. The speaker icon in the bottom right corner toggles the *Auto-preview When Selected* setting. If you would rather trigger (or just stop) each selection manually, [RIGHT ARROW] also alternates the play/stop state.

For audio samples, the file information section starts with a drop-down chooser for the *Audio Preview/Import Mode* preference (also found in the **Dashboard** under *Settings* > *Behavior*).



Two modes are available here, each defining how any audio will be previewed and imported.

› *Play at Original Speed* previews the audio at its original length and speed, regardless of the project tempo. If then inserted as clip material, that clip will be set for playing back neutrally at the project's current tempo.

› *Stretch to Project Tempo* previews the audio at the project's current tempo. If then inserted selected as clip material, that clip will be set to play back at the project's current tempo as well.

4.2.5. Visual Browsers

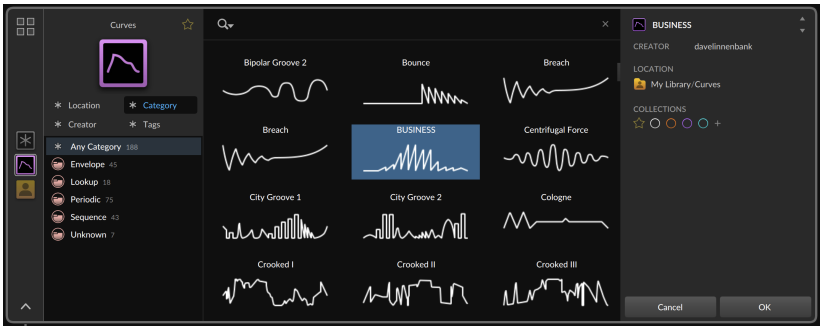
The **Pop-up Browser** can be invoked in numerous contexts. Some content is better visualized than read about.

The primary difference with these special browsers is that results are presented as rows and columns, so all four arrow keys are used for moving between results.

4.2.5.1. Curve Browsers

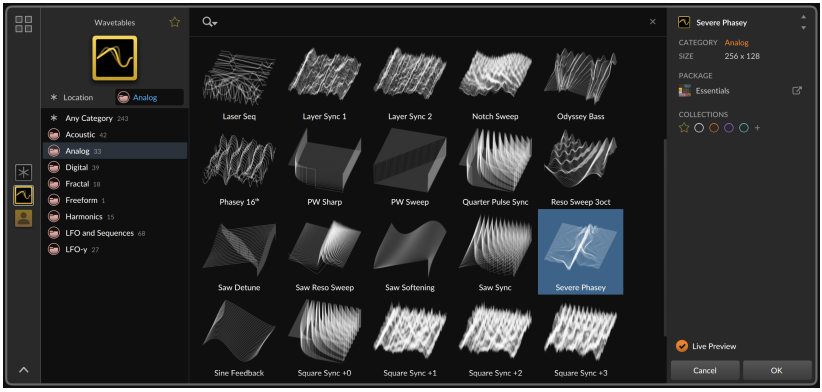
Bitwig has a family of curve-based devices, currently numbering three modulators and five Grid modules (with two of those modules also present in **Polymer**, and the **Transfer** waveshaper also available in **Filter+**

and **Sweep**). They can all load and save the same BWCURVE files, whose free-form shapes are visualized in the *curve browser*.



4.2.5.2. Wavetable Browser

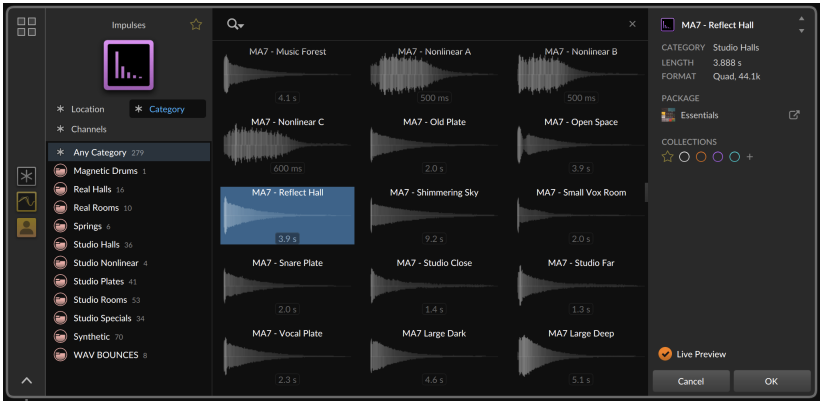
WT-format wavetable files can be loaded by the **Wavetable** oscillator (as a Grid module or in **Polymer**), as well as by the **Wavetable LFO** (in either its modulator or Grid module incarnation). A tilted 3D layout of each file's tables is presented by the *wavetable browser*.



4.2.5.3. Impulse Browser

Bitwig's **Convolution** device can load any audio as an impulse response. Once loaded, the audio is saved to your library as a BWIMPULSE file, joining the hundreds of files available in Bitwig's factory library.

The *impulse browser* visualizes the amplitude of these files, along with their original length listed beneath them.



4.3. Customizing the Browsers

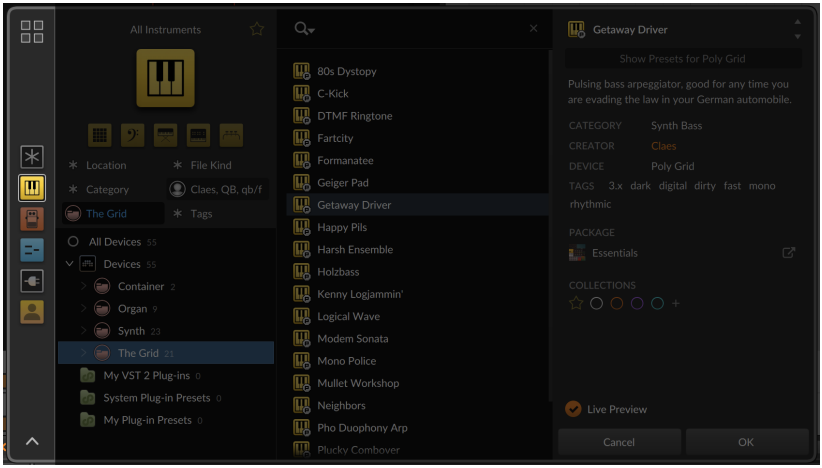
The browsers in Bitwig Studio endeavor to give you good starting points wherever they are invoked. But if you find that something else might suit you better, you should change it. This includes the idea of which *Quick Sources* you would like to have access to in different scenarios, as well as configuring a different default source, filters, and more for each context that Bitwig keeps track of. Additionally, snapshots offer a way to save and restore search sessions. And smart collections add a variation on that idea.

Let's look at these various ways of making the browsers your own.

4.3.1. Quick Sources

The one browser section that hasn't been mentioned so far is the row of miniature source icons that are across the top of the **Browser Panel**.

On the **Pop-up Browser**, they run along its left edge.



This group of icons represents the *Quick Sources* for your current context. By keeping them docked nearby, these sources are accessible with a single click.

On the first click to switch sources, we will try to preserve your search terms and filters. This is true both when you click on one of the *Quick Sources*, if you select a different source from the *All Sources* page, or when you follow an autocomplete suggestion to another source (see section 4.2.1).

And if you are already on a *Quick Source*, clicking its icon again will clear all search criteria, letting you start cleanly from this source.

You can move between the *Quick Sources* with key commands as well. The always-first *Everything* source is mapped to [F1], and the sources that follow take [F2] thru up to [F9].

Since the **Pop-up Browser** presents it *Quick Sources* in a vertical row, pressing [CTRL]+[ALT]+[UP ARROW] and [CTRL]+[ALT]+[DOWN ARROW] ([CMD]+[ALT]+[UP ARROW] and [CMD]+[ALT]+[DOWN ARROW] on Mac) will select the previous or next source. And as the **Browser Panel** has these in a horizontal row, [CTRL]+[ALT]+[LEFT ARROW] and [CTRL]+[ALT]+[RIGHT ARROW] ([CMD]+[ALT]+[LEFT ARROW] and [CMD]+[ALT]+[RIGHT ARROW] on Mac) will switch sources here.

To add a source to the current Quick Sources: drag a source between or beyond any source in the current *Quick Sources* palette. Any source icon can be dragged in, whether it is the current source on the regular browser view, or from the *All Sources* page.

You can also replace one source with another by dragging it on top of the old one.

To remove a source from the current Quick Sources: right-click the source and then select *Remove from Quick Sources*.

Also note the right-click option to *Restore Quick Sources to Default*, in case you want to return a context's *Quick Sources* to the program's default.

4.3.2. Contexts

The word contexts has come up several times already. The browsers in Bitwig can appear when adding new content in various places, and several of these *contexts* can be saved to have their own:

› Set of *Quick Sources*

› Selected source

› Settings for all filters, including which filter is visible (and what subfolders within it are unfolded)

› *Sort Order* (see section 4.2.3)

The *Quick Sources* will be remembered instantly when changed (see section 4.3.1). For the other settings, you have to actively re-save them.

To change the browser settings for this context: click on the magnifying glass icon, then go to the *Browser Settings* submenu and select *Use current selections for this context*.

Note

When the *Use current selections for this context* function is not available, the context you are in either cannot save an independent default state, or you got here indirectly. For example, clicking the folder icon to swap content re-enters the previous search session so this is a local context.

Browser contexts that can have their own defaults include:

› The **Browser Panel** in general (it has only one context; all others are for where the **Pop-up Browser** is invoked)

› When inserting a new track

› When adding to an empty instrument track

› When adding to an empty audio track (including FX tracks)

› When inserting a note FX (for example, clicking *+* between other note FXs, or between a note FX and instrument)

› When inserting a note FX or instrument (for example, clicking *+* after a note FX with nothing following it)

› When inserting an audio FX (for example, clicking *+* after an instrument, or between audio FX)

› When inserting an audio FX or note detector (for example, clicking *+* before a note FX or instrument on an audio track)

› When inserting a new note FX layer

› When inserting a new instrument layer

› When inserting a new audio FX layer

› When inserting into a blank **Drum Machine** cell

› When inserting/browsing content within a **Sampler**

› When inserting into a Launcher clip on a note track

› When inserting into a Launcher clip on an audio track

› When inserting into a Launcher clip on a hybrid track

› When inserting into a *Periodic* curve device (for example, browsing in **Curves** or **Scrawl**)

› When inserting into an *Envelope* curve device (for example, browsing in **Segments**)

› When inserting into a *Sequence* curve device (for example, browsing in **Slopes**)

› When inserting into a *Lookup* curve device (for example, browsing in **Transfer** or **Keytrack+**)

4.3.3. Snapshots

It is possible to save your current search session as a *snapshot*. This will include:

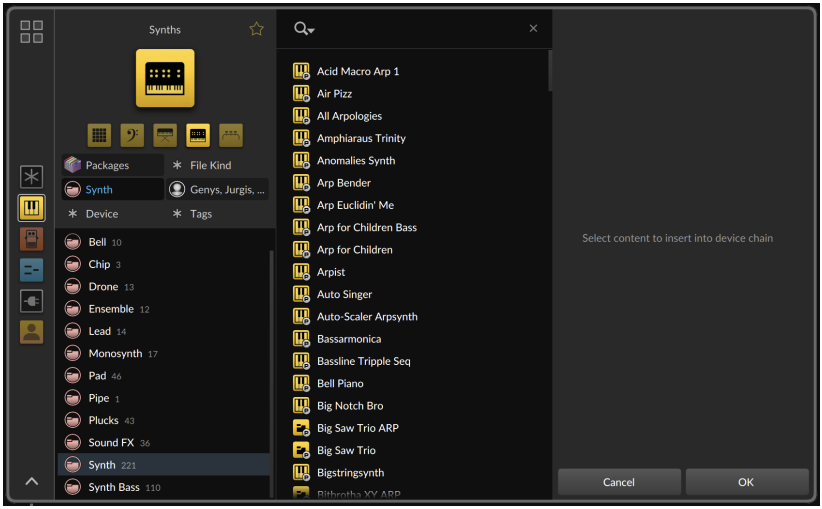
› The selected source

› All selected filters, including which one is visible (and any subfolders that are unfolded there)

› Any text search terms

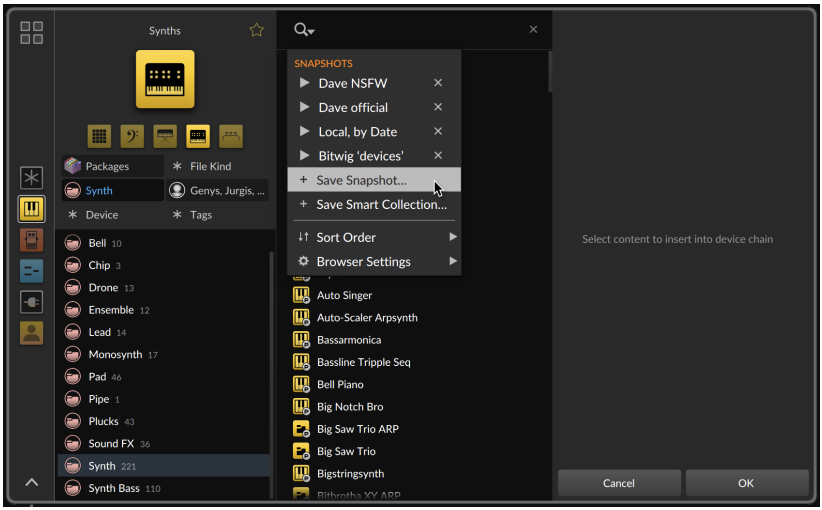
› The *Sort Order* setting

So in this example, I will start in the *Synths* source.

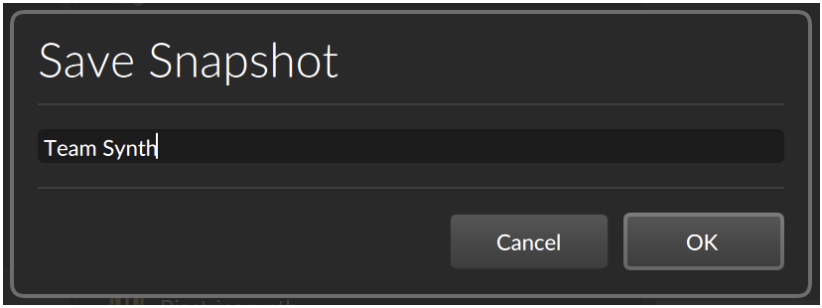


Additionally, the *Location* filter is set to *Packages* (so I will only see installed content instead of my local library); the *Category* filter is set to *Synth*; and for *Creator*, I have selected a few preset makers who I enjoy.

To save a snapshot of your search configuration: click the magnifying glass icon in the search bar, and then select *Save Snapshot…*.

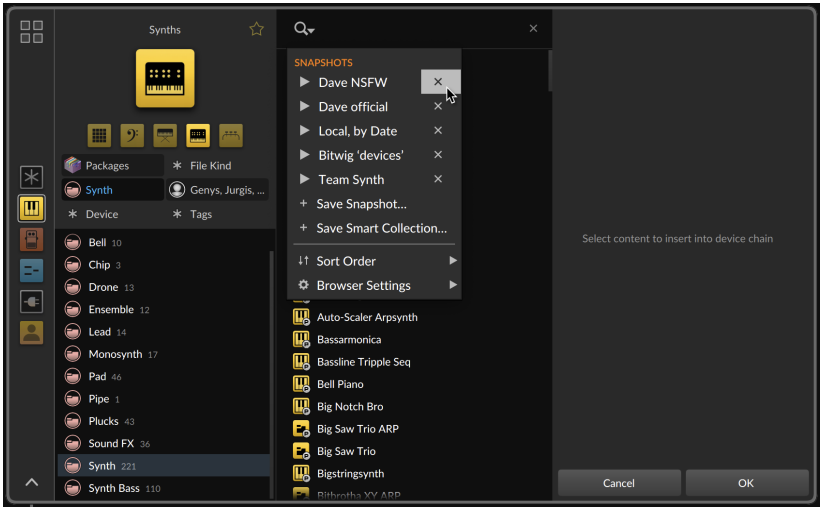


A dialog will appear so you can name your snapshot.



To recall a snapshot: click the magnifying glass icon in the search bar, and then click either the name of the snapshot or the play triangle icon beside it. Everything saved (that is available in the current search context) will be restored, allowing me to continue and modify my search.

To delete a snapshot: click the magnifying glass icon in the search bar, and then click *x* icon to the right of the particular snapshot's name.



4.3.4. Smart Collections

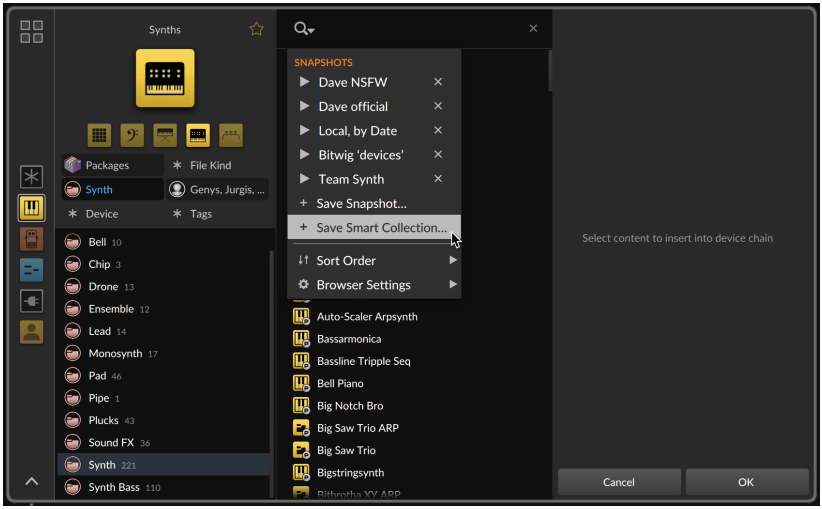
A *smart collection* is a saved set of filters that creates a dynamic collection. This is similar to snapshots, with a couple key difference.

While both snapshots and smart collections offer dynamic results, a smart collection is, well, a collection. This makes it a real entity and allows it to be a source of its own.

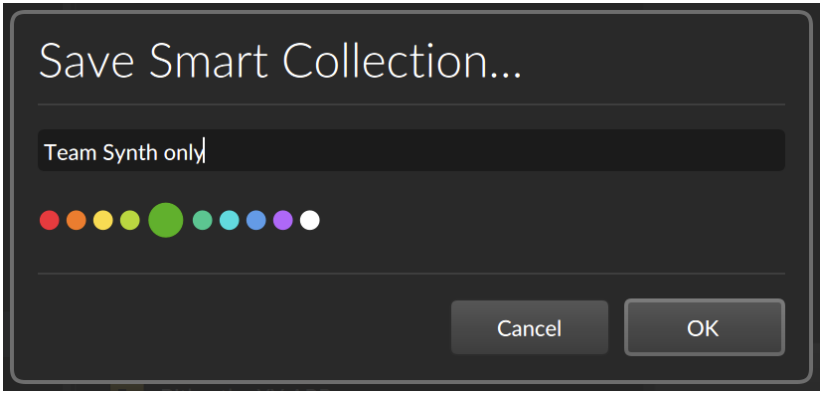
And while filters are completely editable after recalling a snapshot, smart collections preserve your selected filters, making the universe look like the limitations you requested.

As an example, I'll start with the exact same settings as we did with snapshots (see section 4.3.3).

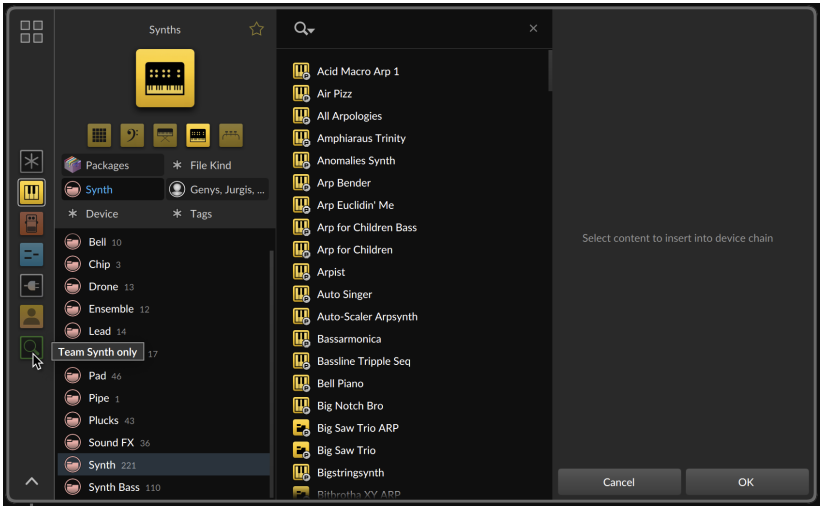
To save a smart collection from your current search: click the magnifying glass icon in the search bar, and then select *Save Smart Collection…*.



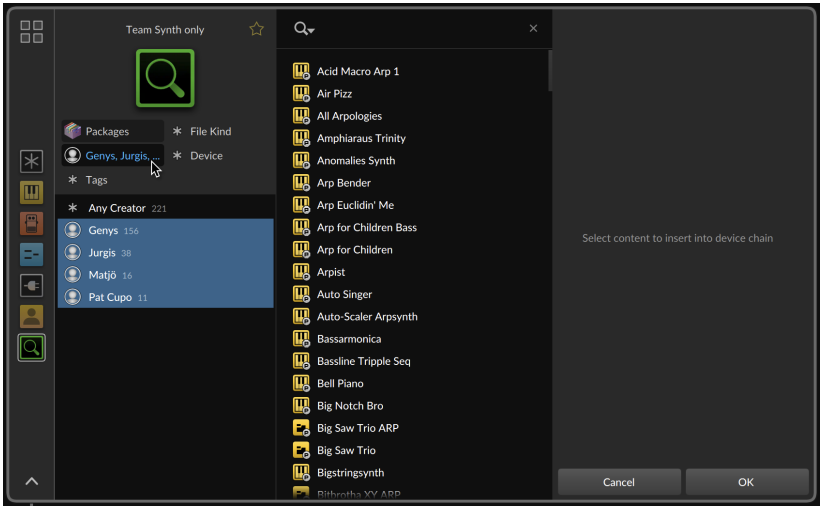
A dialog will appear so you can name and select a color for your smart collection.



Once you've selected *OK* from the dialog, the smart collection will be saved and also added to your current context's *Quick Sources*.



And if we select the new smart collection, we will see the difference between snapshots and smart collections.



From the *Creator* column, we can now clearly see that selecting *Any Creator* will be limited to only those that were selected when the smart collection was saved. And the source of *Synths* and the *Category* filter of *Synth* are now permanent as well.

5. Arranger Clips

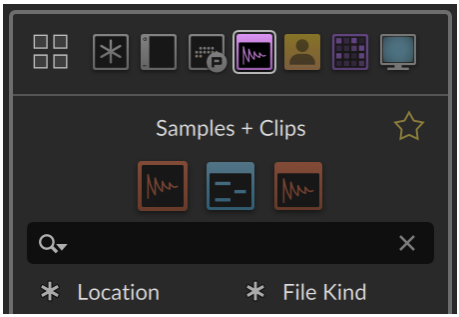
Clips are the heart of any music that you will create in Bitwig Studio. Since they are the smallest unit we will work with for arranging tasks, clips can be thought of as our musical atoms. Put a different way, a clip is the smallest musical idea that you might consider looping.

In this chapter, we will continue working with the **Arrange View**. Taking our knowledge of the browser (see chapter 4), we will see how to drag in clips and move them around. Then we'll adjust their basic parameters in the **Arranger Timeline Panel**, which leads us to playing back Arranger contents and understanding basic transport functions. Finally, we will see how to record new clips.

If our music is made of clips, then creating and capturing our music starts here.

5.1. Inserting and Working with Arranger Clips

While many browser sources can lead you to the same material, the best place to start now is the purple *Samples + Clips* source of the **Browser Panel**.

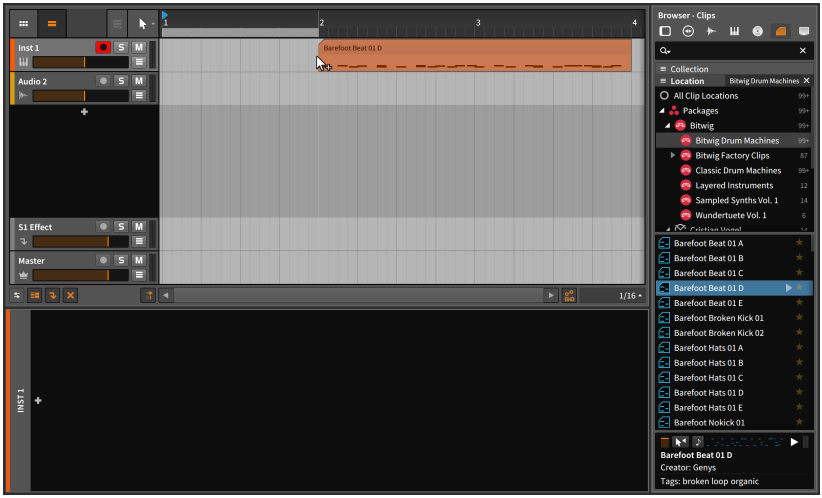


The purposes of this hybrid source is to have all timeline-friendly content in one place. So however you search or filter, you will just find material that can make sense on the Arranger.

And being a parent source, the icons beneath are clickable for isolating one of the included sources, like the blue *Note Clips* source in the center.

5.1.1. Inserting Clips

To insert a clip on an Arranger track: click and drag the clip from the **Browser Panel** to the desired timeline position on the appropriate track.



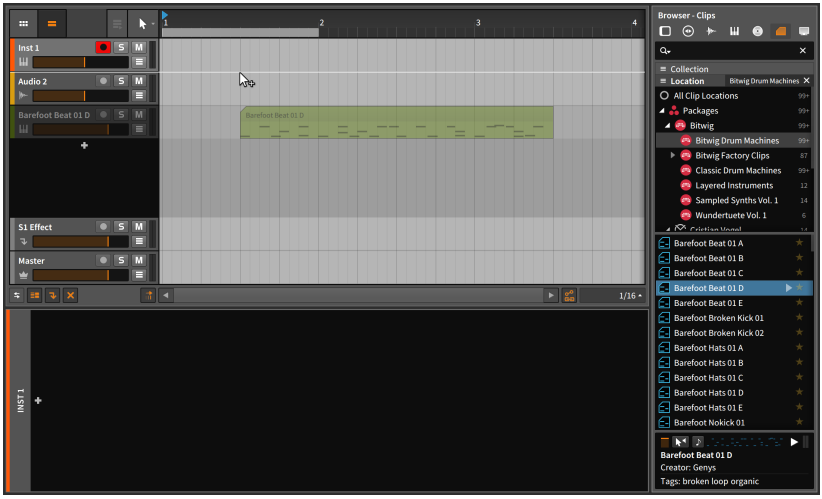
Note

Since we are dragging a note clip, it made the most sense to place it on a note track, but we could have dragged it to any track. As the concept of hybrid tracks may have indicated, Bitwig Studio is rather free with the idea of track types.

If you drag a note clip to an empty audio track, the track will be converted to an instrument track. If you drag a note to an occupied audio track, the track will be converted to a hybrid track. In both cases, the converse is true as well.

So inserting clips from the browser is as simple as dragging them into the Arranger Timeline.

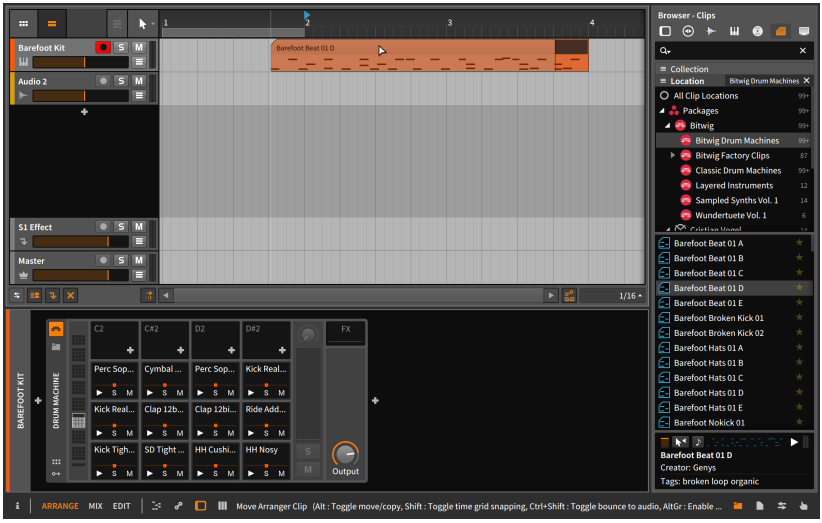
To insert a clip on a brand new Arranger track: click and drag the clip from the **Browser Panel** to the desired timeline position between existing tracks.



This method of inserting clips will work from the **Browser Panel** with any content that can be placed on tracks. And the same method will work when dragging appropriate files from your file manager application (i.e., File Explorer on Windows, Finder on Mac, etc.) directly onto the tracks.

5.1.2. Moving Clips and Snap Settings

To move a clip within the Arranger Timeline Panel: click and drag the clip with the mouse.



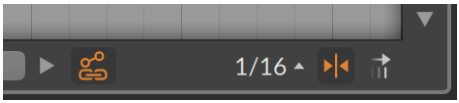
The result will be similar to when the clip was originally inserted from the **Browser Panel**. But also note that as you begin dragging the clip to move it, a status message appears in the window footer with several additional options. (This is shown in the image above; note that the order of options varies by platform, and your screen may not match the sequence in this image.)

Note

Do look for status messages whenever you are clicking and dragging items in Bitwig Studio. This document will not necessarily cover all variations that are shown within the program.

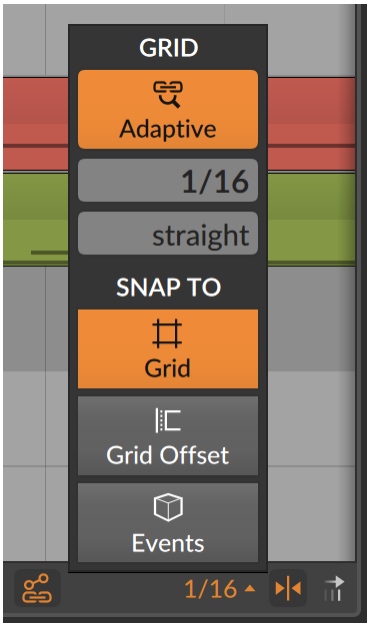
The first option — that adding [CTRL] ([ALT] on Mac) while dragging a selection toggles between moving and copying — was mentioned in a previous chapter.

The second option is new and indicates that [SHIFT] temporarily inverts the snapping behavior, offering to *Disable* it when it is currently enabled, and vice versa. To know the current setting, we should examine the bottom right corner of the **Arranger Timeline Panel**.



Most of these options live on the bottom right of any timeline editor. In the above image, the enabled icon to the right of *1/16* shows arrows coming from the left and right toward a center line (>|<, more or less). This toggle shows that snapping is currently enabled for this editor.

Whether and how clips conform to the beat grid is governed by the more detailed *snap settings*, which are found by clicking on the beat grid settings menu, which is that *1/16* that we saw above.



Found under the *Snap To* header, three independent options determine which elements clips will or will not snap to as you drag them across time. As each option only provides additional anchor points, the options have no effect on one other.

› The *Grid* option causes clips to snap to the current beat grid.

› The *Grid Offset* option uses the current beat grid resolution, but it thinks of a grid in relation to the clip's current start time. So if the clip does not start exactly on the beat grid, the amount that the clip is offset will be preserved when it is moved.

› The *Events* option causes clips to snap to the start and end of other clips within the Arrangement Timeline.

If only one of these options is enabled, only that snapping rule applies. If multiple options are enabled, clips will momentarily snap into place for each and every rule that applies.

These settings will apply not just to moving clips, but to any other editing action in the panel. We will touch upon some of those actions in a moment, but one other option is worth mentioning here.

In the above image, note the *automation follow button*, to the left of the beat grid settings menu. Toggling this function determines whether automation is moved along with clips or not. So if you are moving clips around, be sure to check the status of this button.

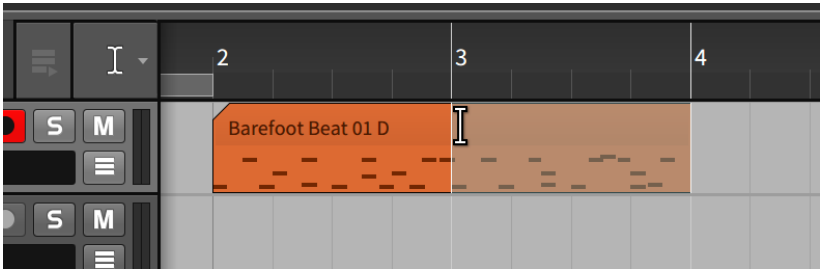
5.1.3. Adjusting Clip Lengths

To demonstrate working with the Bitwig Studio's various tools in the **Arranger Timeline Panel**, we will start with the task of removing the second half of a clip.

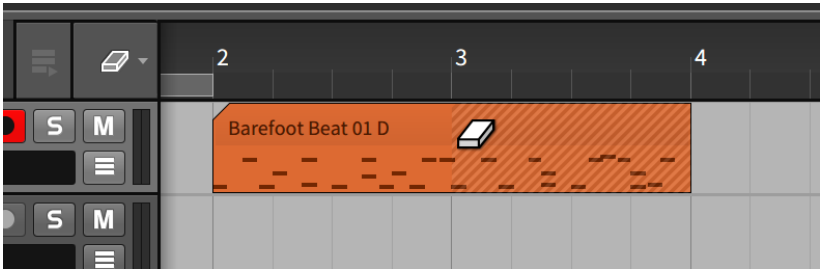
To shorten an Arranger clip: mouse over the top right edge of the clip so that a half-bracket cursor appears. Then click and drag to the left.

Other ways to shorten an Arranger clip include:

› With the Time Selection tool, click and drag over the time area that should be removed. Then clear the time by pressing [DELETE] or [BACKSPACE].



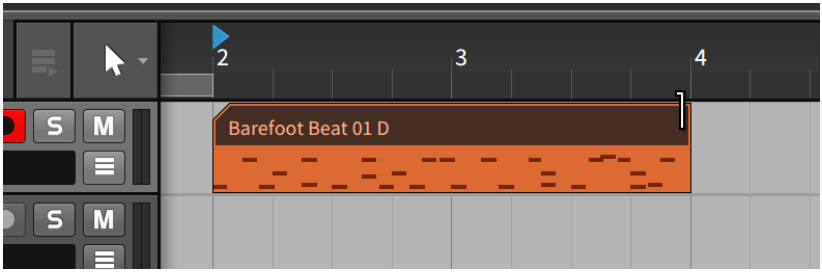
› With the Eraser tool, click and drag over the portion of the clip to be removed.



› With the Knife tool, click the position where the clip should be separated. Once the clip is divided, select and delete — [DELETE] or [BACKSPACE] — the unwanted clip.

All of these methods achieve the same effect. And while it may seem like the second half of our clip is now gone forever, this is not the case. Bitwig Studio still remembers the full contents of our clip in case we need it back later.

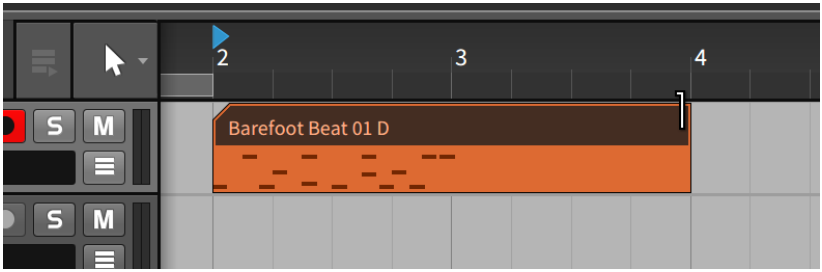
To lengthen an Arranger clip: mouse over the top right edge of the clip so that a half-bracket cursor appears. Then click and drag to the right.



Bitwig Studio acts rather nondestructively, internally preserving data whenever practicable. But you can always ask the program to stop considering data that is not currently visible by using the *consolidate* function, which essentially solidifies a clip for various purposes.

To remove unseen data from a clip: right-click the clip and then choose *Consolidate* from the context menu.

After consolidating the previous clip, extending it would now work differently.

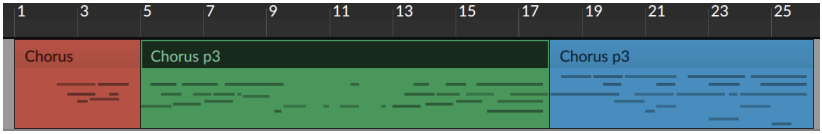


To consolidate multiple clips: select all of the clips. Then right-click one of the clips and choose *Consolidate* from the context menu.

For all of the above purposes, the consolidate function is also available by selecting *Edit* › *Consolidate* or by pressing [CTRL]+[J] ([CMD]+[J] on Mac).

5.1.4. Free Content Scaling

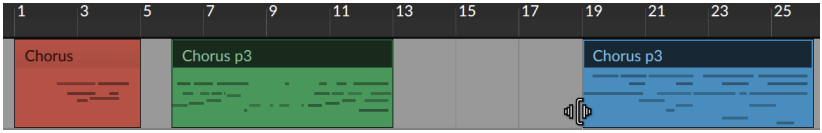
While the normal bracket options (shown above) allow for growing or contracting clips based on their underlying content, clips can also be freely scaled or stretched. And this concept is also the same for note and audio events as well.



To freely scale a clip: hold [ALT] and then click and drag from the left or right edge of the clip.

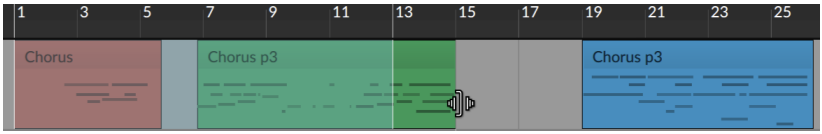
If the right edge is dragged, then the left edge of the clip will be the anchor for scaling, and vice versa.

To freely scale multiple clips: with multiple clips selected, hold [ALT], and then click and drag from the left or right edge of the clip.



Note that when scaling from the edge of a clip, all selected clips are treated individually and are scaled in their original place.

To freely scale time: with a time selection made, hold [ALT] and then click and drag from the left or right boundary of the selection. This will stretch the entire time, shifting any clips that are not lined up with the time selection's start or end.

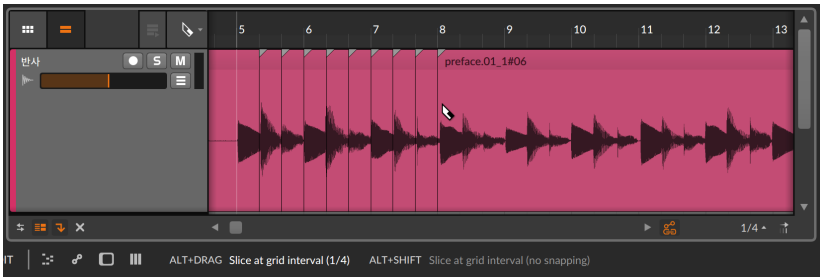


Using a time selection allows anything selected to be scaled, including automation and even partial clips or events.

5.1.5. Slicing and Quick Slice

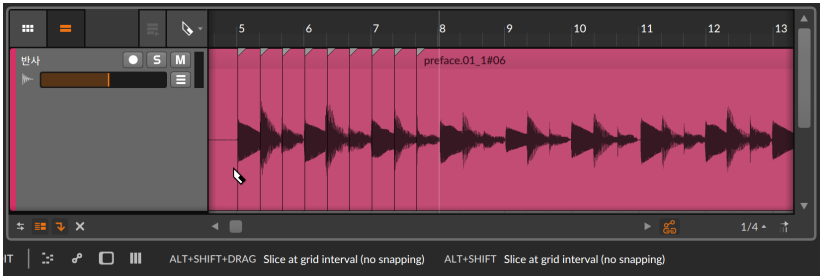
As shown earlier in passing, the Knife tool can be used to slice clips. It can be used in the same fashion to slice notes and audio events. And for any of these types of objects, these is also a **Quick Slice** function, which allows making multiple cuts with one gesture when the Knife tool is selected.

To put successive cuts in a clip, audio event, or note: hold [ALT], and then click at the position of the first cut and drag to the position of the last cut.



The current beat grid value (*1/4* notes, above) will set the distance between cuts and will snap the position of the first cut onto the beat grid. You may sometimes need to freely place (without snapping) the position of the first cut. That is also possible.

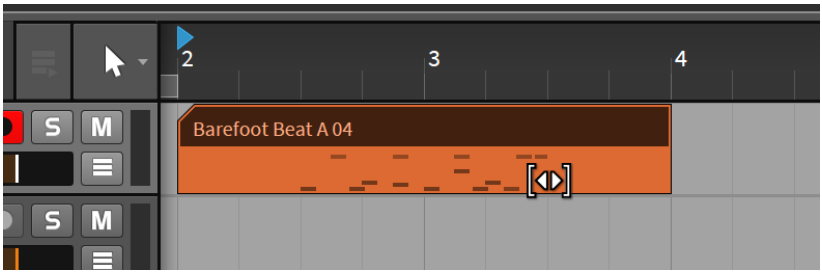
To put successive cuts in a clip, audio event, or note without snapping: hold [SHIFT]+[ALT] and click, to initiate **Quick Slice** mode without quantization. Then drag either to the right or left to insert successive cuts.



5.1.6. Sliding Arranger Clip Content

The content of one or multiple clips can also be shifted left and right from the **Arranger Timeline Panel**. Sliding content in this fashion preserves the boundaries of each clip, simply sliding the contained note or audio events (including any associated expressions) earlier or later in time.

To slide the content of a clip: mouse over the top half of the waveform. Then [CTRL]-click ([CMD]+[ALT]-click on Mac) and drag horizontally.

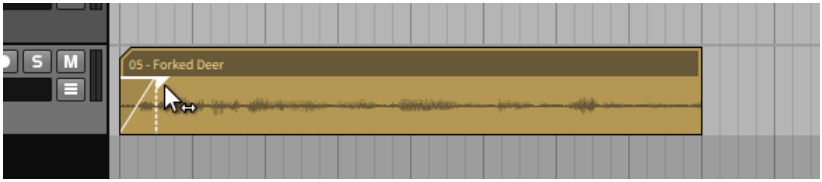


You can optionally add the [SHIFT] key while dragging to toggle the snapping behavior.

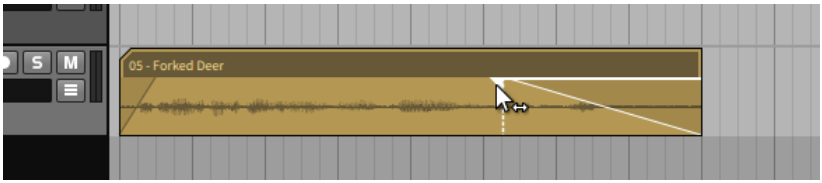
5.1.7. Applying Fades and Crossfades to Audio

While most functions in this chapter are applicable to both note and audio clips, the options to fade in, fade out, and crossfade are only relevant to audio clips.

To create a fade in: mouse over the middle of the clip's left edge, at the top of the waveform display. Once a white triangle appears, click and drag the triangle toward the center of the clip. Release the mouse where you would like the fade to end.

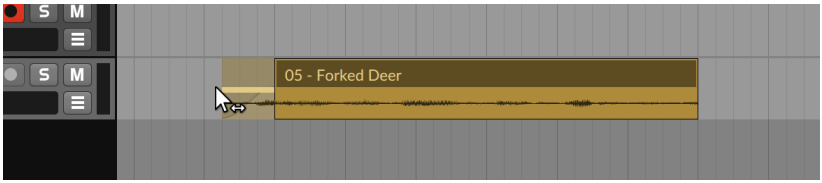


Fade outs can be created in the same way by mousing over a clip's right edge.



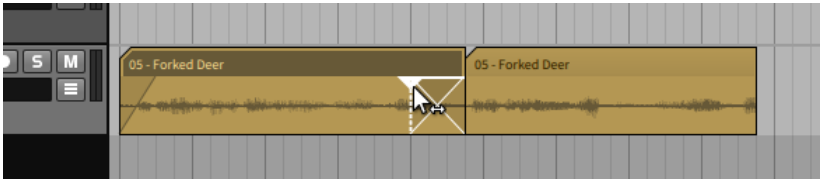
Additionally, *pre-fades* can be created on audio clips. Pre-fades preserve your original clip start at full amplitude, fading in any earlier audio material before your original clip edit.

To create a pre-fade: mouse over the middle of the clip's left edge, at the top of the waveform display. Once a white triangle appears, click and drag the triangle to the left of the clip. Release the mouse where you would like the pre-fade to end.



Creating a crossfade requires audio clips that are overlapping and have material extending beyond their own boundaries.

To create a crossfade: mouse over the middle of the intersection of the clips, at the top of the waveform display. Once two white triangles appear, click the triangle where you would like the crossfade to boundary to be, and then drag across the clips' border. Release the mouse where you would like the boundary of the crossfade to be.

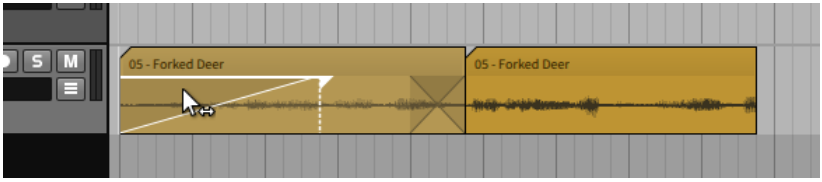


That is a bit of a mouthful so let's take a moment unpack the process.

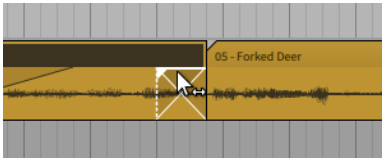
If you click on a clip's edge and drag toward its center, you are creating a fade in or fade out for that single clip. So creating a crossfade requires clicking on one of the overlapping clips and then dragging the fade past its boundary and onto the other one.

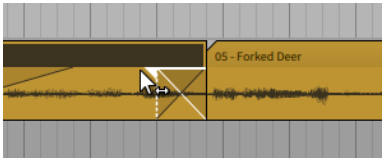
If you start by clicking in clip 1 and then drag across to clip 2, the crossfade will begin where the boundary was and will end wherever you release the mouse. If you start by clicking in clip 2 and then drag across to clip 1, the crossfade will end where the boundary was and will end wherever you release the mouse.

To adjust the boundaries of any fade: mouse over the top portion of a fade so that its white triangle(s) appears, and then click and drag to move the fade's boundary relatively.

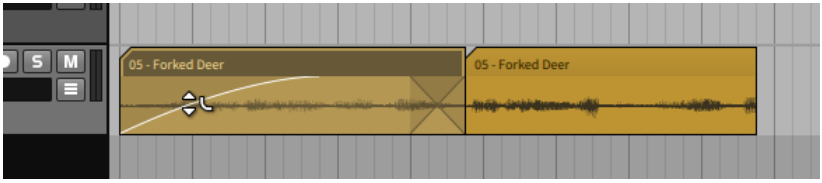


Note that for a crossfade, dragging an inner boundary will select both curves (shown as highlighted in white) and let you adjust them together. Dragging an outer boundary will let you adjust the closest fade by itself.



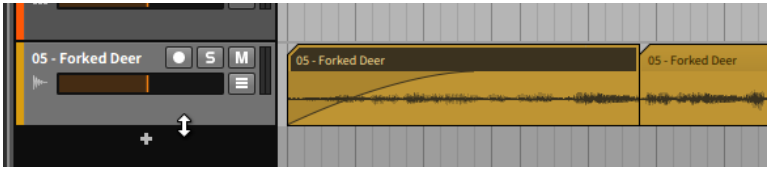


To adjust the slope of a fade: mouse over the fade's curve, and then [ALT]-click and drag the mouse up or down.

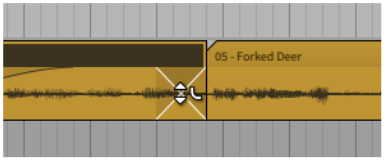


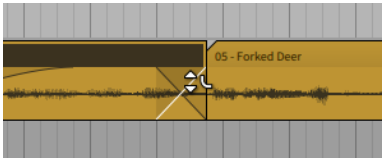
Note

If may be difficult to get the curvature cursor to appear if your track height is small. If you are working a good amount with fades and their curves, you should make your track's height larger than the minimum by clicking and dragging the bottom of the track header.

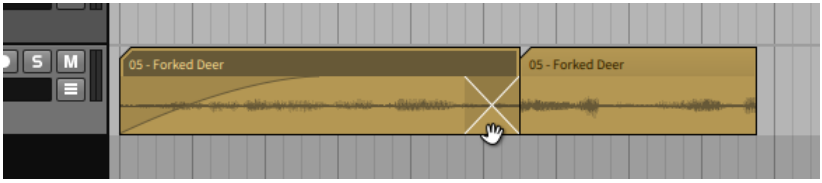


Note again that with a crossfade, you can either mouse over both fade curves to manipulate them in tandem, or you can adjust each fade by itself — just hold the [ALT] key and drag your target(s).





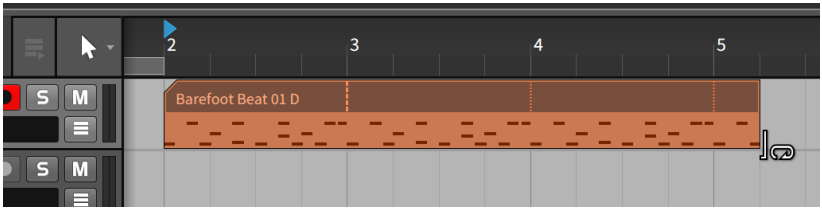
To shift an entire crossfade: mouse over the bottom of the crossfade, and then click and drag backward or forward in time.



5.1.8. Looping Clips

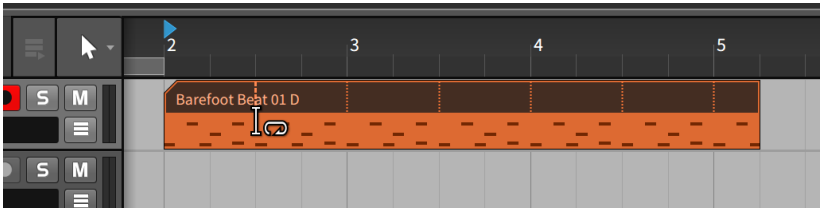
As clips are intended to be the smallest practical musical idea, you may want to loop clips.

To loop an Arranger clip: mouse over the bottom right edge of the clip so that a half-bracket cursor appears with a looping oval. Then click and drag to the right.



After you drag the clip beyond its full length, additional copies will be generated. The first copy starts with a dashed vertical line, marking the loop length being used. All subsequent repetitions of the loop are marked with dotted vertical lines. Once the clip is looping, you can do the same using any of the "bracket" tools, either at the end or beginning of the clip.

To adjust the loop length of an Arranger clip: mouse over the clip's first repeat marker (the dashed vertical line) so that an I-beam cursor appears with a looping oval. Then click and drag in either direction.

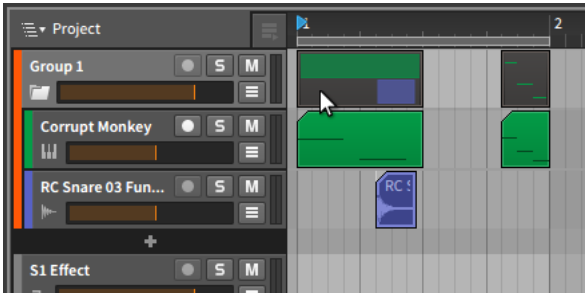


The length of the clip itself remains the same while the section of the clip that loops — and accordingly the number of repetitions — has changed.

5.1.9. Meta Clips and Group Tracks in the Arranger

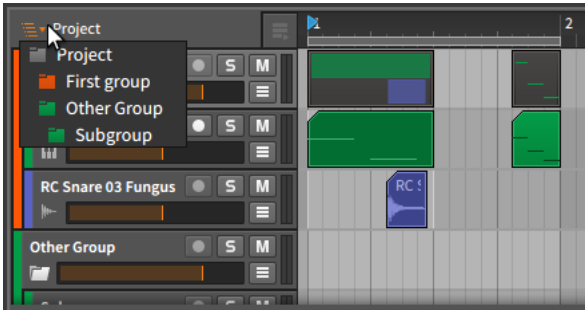
When working with a group track, the contents of its enclosed tracks are summarized in the Arranger Timeline. When no clips within the group track are overlapping, these *meta clips* are essentially direct representatives of their contained clips.

When the enclosed track do have overlapping clips, affected meta clips adapt to show colorized summaries of the track contents.

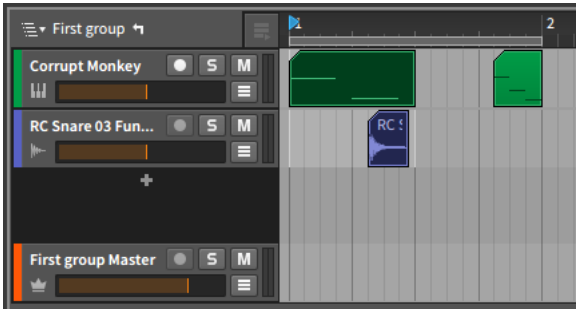


Regardless of the display style, each meta clip acts as an alias of the clip (or clips) that they represent. As with any regular Arranger clip, meta clips can be moved by dragging and dropping, they can be cut or copied or pasted in the normal ways, they can be deleted, and they can even be split with the Knife tool. Taking any of these actions on meta clips directly affect the clips that they represent.

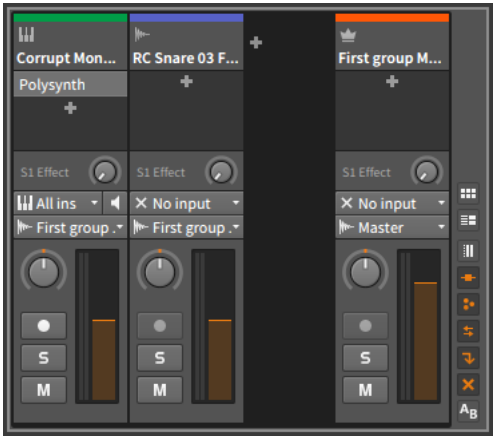
When working with group tracks, a project navigation menu appears at the top of the **Arranger Timeline Panel** within the **Arrange View**.



Clicking on this menu exposes a hierarchy of the current project, including the top level of the *Project* and all group tracks that are present. Selecting one of these group tracks changes the context which the **Arranger Timeline Panel** displays.

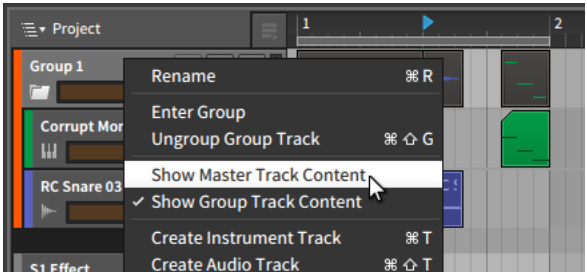


To the right of the project navigation menu, a "left turn" arrow has now appeared. Clicking this arrow navigates upward into the parent level of the current context. It is also worth noting that the context selected in the **Arranger Timeline Panel** is preserved if you switch to the **Mixer Panel**.



Finally, back in the **Arranger Timeline Panel**, you can toggle between viewing each group track's meta clips or a representation of the group track's master track.

To view the contents of the group track's internal master track: right-click on the group track's header, and then select *Show Master Track Content* from the context menu.

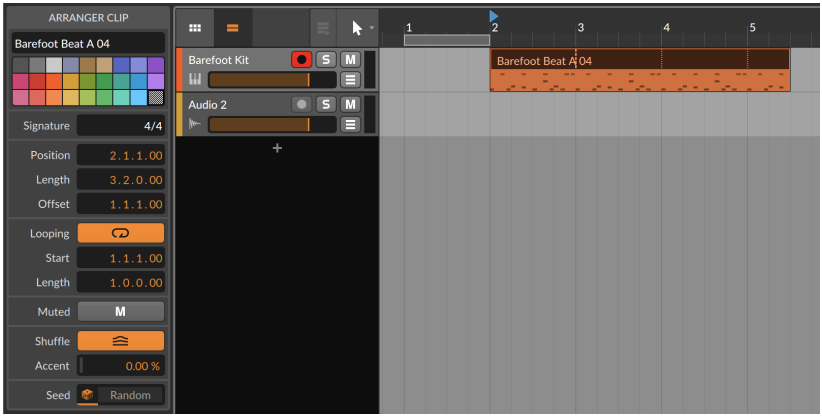


You can switch back to displaying the meta clips by calling up that same context menu and then selecting *Show Group Track Content*.

5.1.10. The Inspector Panel on Arranger Clips

While the Arranger Timeline is a convenient, graphical view for working with the length and loop settings of a clip, all of those mouse movements are really just triggering parameter changes in the **Inspector Panel**. By investigating these parameters (along with the associated functions available in the *Clip* menu), we will get a clearer understanding of what is possible in Bitwig Studio in general and the Arranger in particular.

We will start by focusing the **Inspector Panel** on the same clip looping example we just finished.



For the time being, we are just paying attention to the parameters in the *ARRANGER CLIP* portion of the **Inspector Panel**. We have already seen

the name (see section 3.2.4) and color options (see section 3.2.5) for tracks. The remaining sections offer additional parameters.

5.1.10.1. Signature Section

Signature sets the time signature of the selected clip. Along with an optional tick setting (see section 2.3.3), this reflects how the clip is displayed for editing.

5.1.10.2. Time (Position) Section

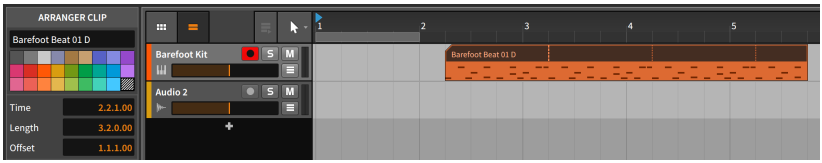
These settings relate to the musical time or position of the selected clip:

› *Time* sets the start of the clip in the Arranger Timeline. Adjusting this position will simply move the clip exactly as it exists, the same as clicking and dragging the entire clip in the Arranger.

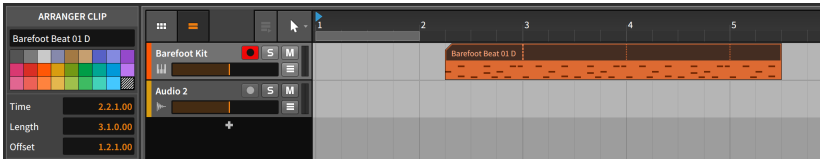
› *Length* sets the duration of the clip in the Arranger Timeline. Adjusting this duration will simply lengthen or shorten the clip, the same as using the bracket cursor to adjust the right edge of the clip.

› *Offset* preserves the position and length of the clip, but shifts its internal content by the set amount. This is the same as using the bracket cursor to move the left edge of the clip forward in time.

Taking the previous image as an example, I could increase the *Time* from *2.1.1.00* to *2.2.1.00*. The entire clip is now happening a quarter note later.



But if I wanted the clip to stay in time and simply skip the first beat it was playing, I would increase the *Offset* from *1.1.1.00* (no offset) to *1.2.1.00*.

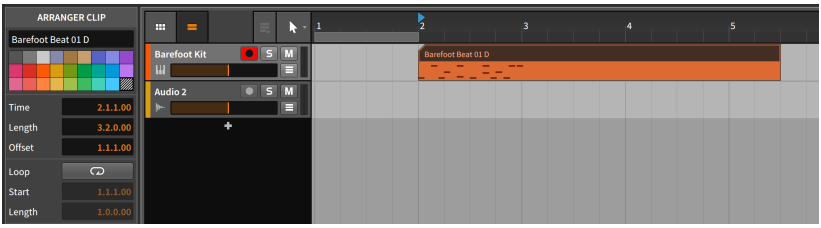


Note that the first beat is included in subsequent loops.

5.1.10.3. Loop Section

These settings relate to the looping of the selected clip:

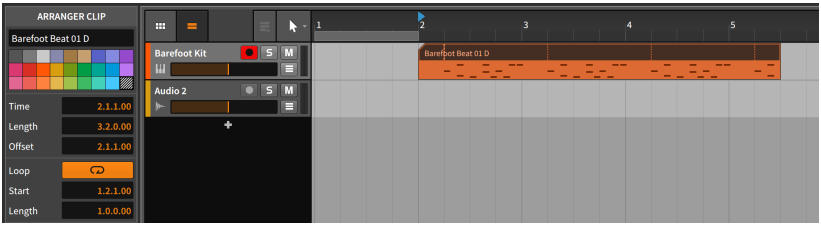
› *Loop* toggles whether or not the clip loops with the Arranger. When disabled, the clip will play only once. If the size of the clip is longer than its contents, the later portion of the clip will be empty.



If *Loop* is off, the other settings here are ignored.

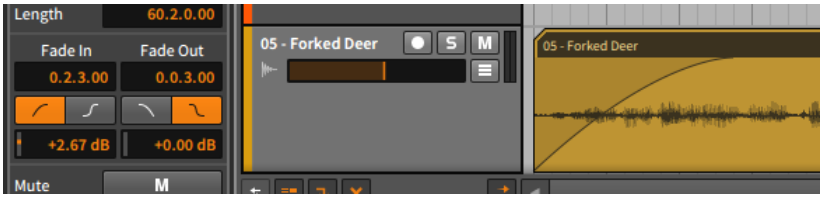
› *Start* is the looping equivalent of the *Offset* parameter, keeping the clip contents in their place but delaying the point at which each loop repetition starts.

Taking the same example from above, I could increase the *Start* from *1.1.1.00* (no loop offset) to *1.2.1.00*, causing each one-bar loop to end in the same place but start a quarter note late.



› *Length* sets the duration of the clip that is being repeated. This is the same as using the I-beam cursor with a looping oval to graphically adjust the loop length.

5.1.10.4. Fade Section



As stated earlier, fade actions and parameters apply only to audio clips. So these twin sets of parameters represent controls for any *Fade In* and *Fade Out* applied to the selected audio clip. Taken from top to bottom:

› The musical time value represents the length of the fade. If it is set to zero (*0.0.0.00*), then no fade is applied regardless of the other settings.

› The buttons allow toggling the fade's curvature type between a standard linear curve and an S-curve, respectively.

› The level value sets the amplitude at the fade's midpoint, effectively shaping the fade's curve.

As shown earlier in the Arranger, crossfades are really comprised of two separate fades (a fade out from the first clip, and a fade in on the second). As such, their settings can be coordinated or handled completely independently.

5.1.10.5. Mute Section

Mute toggles whether or not the selected clip is disabled on playback. This is in contrast to the track mute button, which disables all contents of the track.

5.1.10.6. Shuffle Section

These settings relate to the groove of the selected clip:

› *Shuffle* toggles whether or not the Global Groove parameters are applied to the clip. If *Shuffle* is off, the other setting here is ignored.

› *Accent* sets the percent of the Global Groove's accent *Amount* that should be applied to this clip.

For example, if the Global Groove's accent *Amount* is set to *100%* (the default setting) and the clip's *Accent* setting is at *30%*, then the clip will apply an accent at 30% strength (30% of 100%).

Or if the Global Groove's accent *Amount* is set to *50%* (the default setting) and the clip's *Accent* setting is at *50%*, then the clip will apply an accent at 25% strength (50% of 50%).

Since this is a scaling function, either parameter being set to zero (*0%*) results in no accent.

5.1.10.7. Seed Section

The clip *Seed* setting relates to randomized parameters in Bitwig Studio. This includes any expression *Spread* values (see section 10.1.3) and *Chance* Operators (see section 12.1.1).

When "random" numbers are being generated, the *seed* shapes the sequence that follows. When that seed is randomly selected, so are the values produced. This is the default behavior for clips in Bitwig.

The die on the left is selected, reading out as *Random* because a new seed is picked each time the clip begins playing. But if the *same* seed value is used each time, then playing the clip will produce the same series of numbers — and sounds .

To generate a Seed value for a clip: click on the right side of the *Seed* field (where *Random* showed in the picture above).

The die is deselected, and a visualization of the current *Seed* value is shown. You can now play the clip and hear the pattern that this seed produces for any randomized elements. If you like the results, keep it; the same result will be produced when you trigger the clip again.

Note

Alternatively, you could print these randomized elements by using the *Consolidate* function (see section 12.2.3). Or to choose what is

made permanent and generate a new, longer clip, you could use the Launcher's *Expand* function (see section 12.2.2).

To generate a new Seed value for a clip: click on the right side of the *Seed* field again (where the current value is visualized in the last picture).

Different seed, different pattern on playback. You can also right-click on the right side of the field to copy the current seed value or paste in one from another clip.

And to return to randomized playback, simply click the die icon.

Note

One technical detail. A defined *Seed* value makes the full sequence repeatable, including all additional loop cycles that follow. So the results are not identical for every loop, but rather the values picked *for each loop* are reproducible.

To borrow the idea of a die, if the clip's set *Seed* produces a *5* on the first cycle, a *6* on the second loop, and a *2* on the third pass, retriggering the clip will produce *5*, then *6*, then *2*, and so on, again. And again. And...

5.1.10.8. Clip Menu Functions

These functions take the specified action on the selected clip:

› *Consolidate* merges all selected clips (on a track by track basis) into single, contiguous clips.

› *Double Content* makes the selected clip twice its current length and duplicates its non-looping contents.

› *Reverse* flips the order and positions of the clip's contents, causing them to play "backwards."

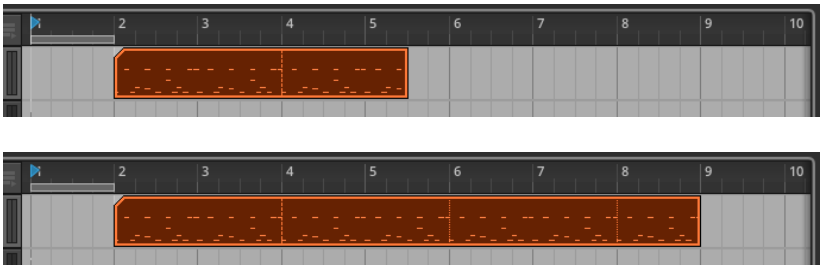
› *Scale Each 50%* and *Scale 50%* both halve the length of each selected clip as well as each contained event's duration and position, effectively causing the clip to play back twice as fast.

The following images demonstrate a selected clip both before and after either *Scale 50%* function is applied:

The difference between the two functions comes when multiple clips are selected. In this case, *Scale Each 50%* preserves the start time of each selected clip, while *Scale 50%* uses the first clip's start time and moves each following clip 50% closer to the first clip.

› *Scale Each 200%* and *Scale 200%* both double the length of each selected clip as well as each contained event's duration and position, effectively causing the clip to play back half as fast.

The following images demonstrate a selected clip both before and after either *Scale 200%* function is applied:



The difference between the two functions comes when multiple clips are selected. In this case, *Scale Each 200%* preserves the start time of each selected clip, while *Scale 200%* uses the first clip's start time and moves each following clip 200% further away from the first clip.

› *Scale...* stretches the selected clip by an *Amount* that you type in. An additional option for whether to *Scale each (keep position)* — to preserve the start time of each Arranger clip — is also available.

› *Bounce In Place* replaces the selected clip with a new audio clip. When the selected clip was an audio clip, the sound source is the audio itself,

which will be printed into a solid clip. For a note clip, the sound source is the first instrument device in the track's device chain.

Note

For additional information on this function, see section 13.2.2.

› *Bounce* prints the sound source of the selected clip into a new, solid audio clip (the functional equivalent of a "consolidated" clip). For an audio clip, the sound source is the audio itself, which will be printed into a solid clip. For a note clip, the sound source is the first instrument device in the track's device chain.

Note

For additional information on this function, see section 13.2.1.

› *Slice In Place...* divides the selected clip into multiple clips, slicing regularly at a note interval (*on Beat Grid*). With audio clips, slicing can also be done at *Onsets* (the detected transients) or *Beat Markers* (defined stretch points that you may have changed). This can be an extremely efficient way to do audio edits.

Note

For additional information on this function, see section 10.2.1.7.

› *Slice to Drum Machine...* produces a new instrument track loaded with a **Drum Machine** device, which contains a series of audio clips (loaded in **Sampler** devices) representing the original clip's content. The track is loaded with a note clip that is configured to trigger the **Drum Machine** in a fashion that reproduces the original clip.

Note

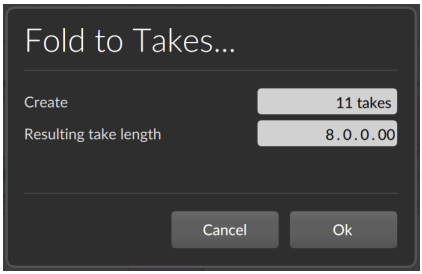
For additional information on this function, see section 13.3.2.

› *Slice to Multisample...* produces a new instrument track loaded with a **Sampler** device, whose multiple samples represent the original clip's content. The track is loaded with a note clip that is configured to trigger the **Sampler** in a fashion that reproduces the original clip.

Note

For additional information on this function, see section 13.3.1.

› *Fold to Takes...* takes any audio clip and wraps its material into successive take lanes. Once selected, a dialog appears allowing you to set either the number of takes the clip should be folded into, or the *Resulting take length* for each take. As these parameters are connected, changing one will change the other too.



Note

The function can also be used to fold the contents of a single take lane (see section 10.1.4.2).

› *Reset Fades* removes any applied fades from the selected audio clips.

› *Auto-Fade* applies a quick, relative fade in and fade out to all selected audio clips.

› *Auto-Crossfade* applies a quick, relative pre-fade and fade out to all selected audio clips, creating crossfades between adjacent clips.

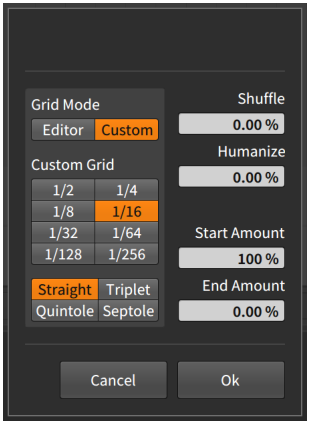
› *Transpose a Semitone Up* shifts the pitch up by one half step either (by adjusting the pitch of each note events or the pitch expression of each audio events).

› *Transpose a Semitone Down* shifts the pitch down by one half step (by adjusting the pitch of each note events or the pitch expression of each audio events).

› *Transpose an Octave Up* shifts the pitch up by twelve semitones (by adjusting the pitch of each note events or the pitch expression of each audio events).

› *Transpose an Octave Down* shifts the pitch down by twelve semitones (by adjusting the pitch of each note events or the pitch expression of each audio events).

› *Quantize...* moves the start and/or end times of all events in the selected clip(s) in relation to a beat grid. A parameter pane appears after this function is selected.



Note

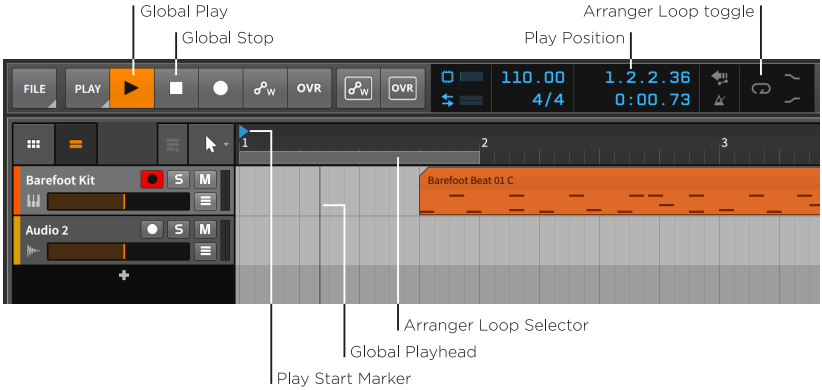
For additional information on the parameters available for the quantize function, see section 11.2.2.5.

› *Make Legato* adjusts the length of each event in the selected clip(s) so that it ends immediately before the next event begins. This creates a continuous series of events by both extending events beyond rests to the beginning of the next event and by shortening events which overlapped their successor.

› *Save Clip To Library...* stores the selected clip in your library, allowing you to first set various tags for the clip.

5.2. Playing Back the Arranger

How to play Arranger clips is simple enough: you play the Arranger. But there are a few details worth getting into at this point. Let's begin this discussion with the elements that enable basic playback.



To play the Arranger timeline: engage the transport by pressing either [SPACE BAR] or [P], or by clicking the Global Play button.

To stop the Arranger timeline: disengage the transport by pressing either [SPACE BAR] or [P], or by clicking the Global Stop button.

The *Global Playhead* is an indicator of where the transport has most recently played. In the Arranger Timeline, it is represented with a vertical black line. Whenever the transport is active, the Global Playhead progresses thru the Arranger tracks, and its location is noted by the play position display in the window header.

The *Play Start Marker* is the blue, right-facing triangle within the Beat Ruler that indicates where the transport will play from the next time it is engaged.

To move the Play Start Marker: single-click in the top half of the Beat Ruler.

Other ways to move the Play Start Marker include:

› Single-click anywhere within the Arranger Timeline with the Pointer tool.

› Click and drag the play position in the window header's display section.

› Select a single Arranger clip to move the Play Start Marker to the beginning of that clip.

To play the Arranger timeline from the beginning: press [ALT]+[SPACE BAR] or [ALT]+[P] .

To play the Arranger timeline from the Global Playhead's position: press [SHIFT]+[SPACE BAR] or [SHIFT]+[P] .

To stop the Arranger timeline and advance the Play Start Marker: click the Global Play button.

The *Arranger Loop Selector* sets the region of the Arranger Timeline that will be looped during playback. This region is also used for several other functions.

To toggle the Arranger Loop function: click the Arranger Loop toggle in the window header.

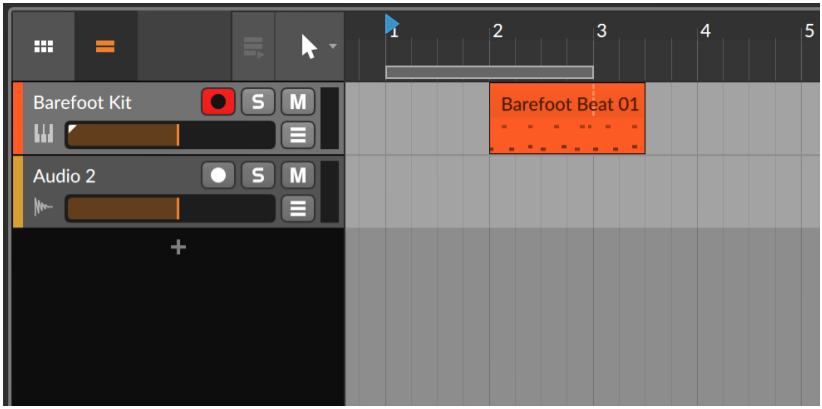
The Arranger Loop function affects all tracks as it literally picks up and moves back the Global Playhead when the end of the region is reached. This is a playback function, while clip looping is an arrangement function.

To move the Arranger Loop Selector's position: click the center of the Arranger Loop Selector and drag it in time.

To change the Arranger Loop Selector's length: mouse over the left or right edge of the Arranger Loop Selector so that a bracket cursor appears. Then click and drag in either direction.

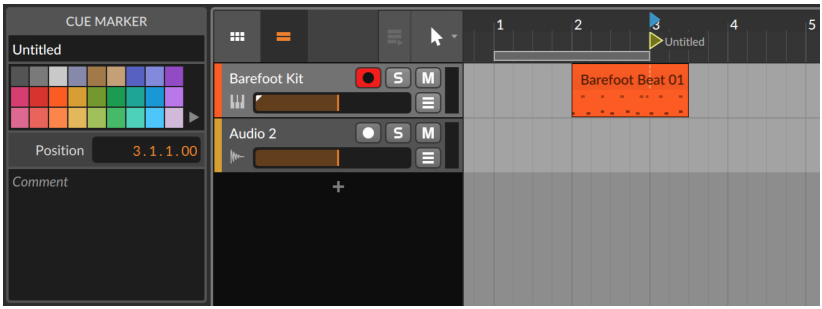
5.2.1. Cue Markers

You also have the option of using *cue markers* in the Arranger, which store play positions along the Arranger Timeline for easy triggering. To use Arranger cue markers, first right-click within the Beat Ruler, and then enable *Show Cue Markers* from the context menu. This will make the Beat Ruler slightly taller.



To create a cue marker: right-click the Beat Ruler, and then select *Insert Cue Marker*. A yellow play icon and the cue marker's current name (likely

Untitled) will appear in the Beat Ruler. Or use the *Insert Cue Marker Here* function, which can be freely assigned to a keyboard or MIDI command (on the *Shortcuts* page of the **Dashboard**).



The left edge of a cue marker's play button icon aligns with its location.

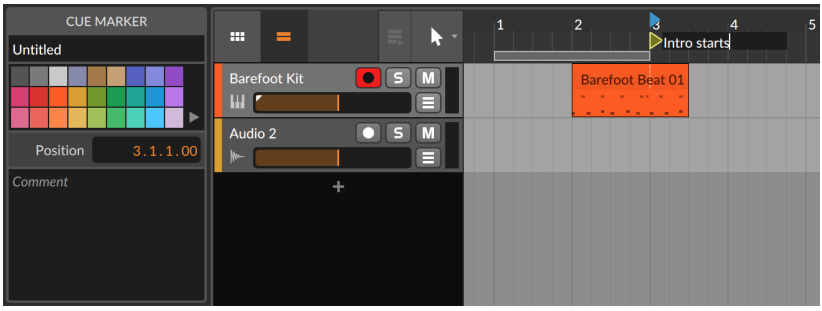
To trigger playback from a cue marker: double-click its play icon.

If the transport was inactive, playback will start immediately from the cue marker. If the transport was already going, playback will move to the cue marker's position after the *Default Launch Quantization* interval (see section 6.2.5.2).

Note

If you want the same playback behavior without creating a cue marker, simply double-click the desired playback position from the top of the Beat Ruler (between the numbers).

To rename a cue marker: double-click its name.

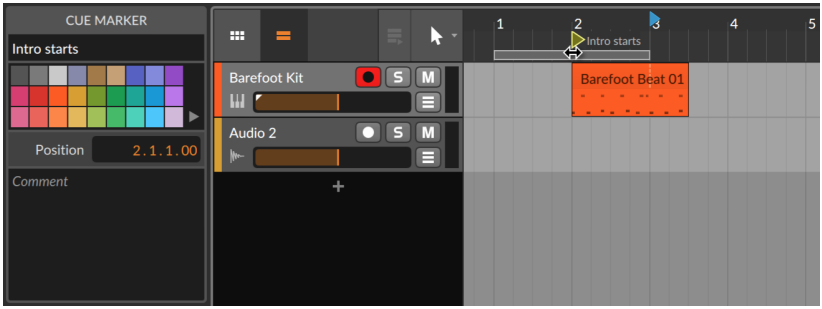


The *Comment* area in the **Inspector Panel** allows for adding any text you find relevant, including lyrics. Additionally, a list of all Arranger cue

markers — and their comments — can be seen, selected, or triggered from the *Sections* tab of the **Project Panel** (see section 14.2.4).

To change a cue marker's color: right-click either the cue marker's icon or name, and then select a different color from the palette that appears within the context menu.

To move a cue marker: click either the cue marker's icon or name, and then drag it to the desired position. Or click the cue marker to select it, and then change its position in the **Inspector Panel**.



To delete a cue marker: click the cue marker to select it, and then press [DELETE] or [BACKSPACE].

5.2.2. Time Signature Changes

Like cue markers, time signature changes can also be inserted along the Arranger Timeline.

To insert a time signature change: right-click the Beat Ruler, and then select *Insert Time Signature Change*. An orange triangle appears beside the new time signature change, indicating that it is selected and its parameters can be edited from the **Inspector Panel**.

Note

For more information on time signatures and how the ticks parameter is handled, see section 2.3.3.

To move a time signature change: click either the time signature change's triangle or name, and then drag it to the desired position. Or click the time signature change to select it, and then change its position in the **Inspector Panel**.

To delete a time signature change: click the time signature change to select it, and then press [DELETE] or [BACKSPACE].

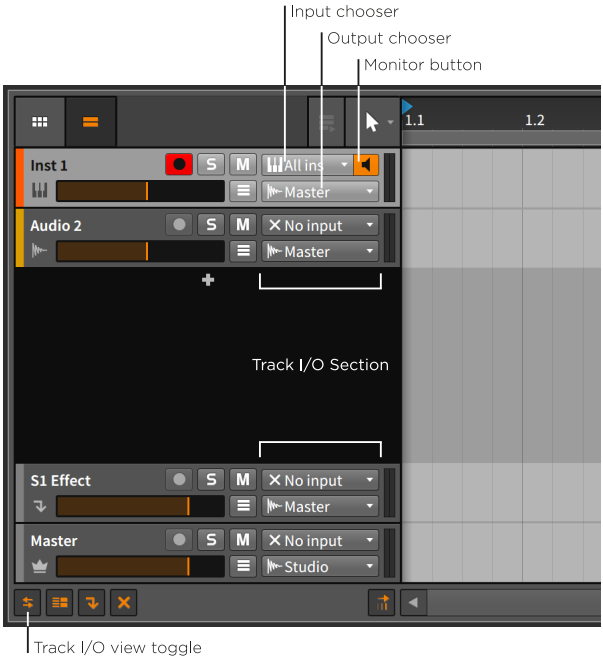
5.3. Recording Clips

Since we can now edit Arranger clips in the most fundamental ways, it is a good time to examine recording new note and audio clips. This begins with getting the right signals routed into our tracks.

Before we deal with this on a track level, make sure that any audio and MIDI interfaces/controllers you are using have been set up properly (see section 0.2.2).

5.3.1. Track I/O Settings

To assign input and output paths for each track, we must first have access to the Track I/O section within each track header. This section's visibility is toggled by clicking the Track I/O view toggle.



This section contains the following controls:

› The *input chooser* lets you select which signals are getting routed into the track.

  For instruments tracks, the options are incoming MIDI sources. The default selection is *All inputs* so that every MIDI source should make it to the track.

  For audio tracks, the options are both incoming audio sources and the audio outputs of all other tracks. The default selection is *No input*.

Note

If a desired MIDI source is missing, select the *Add Controller…* option, which will open the *Controllers* tab on the *Settings* page of the **Dashboard** (see section 0.2.2.2).

Similarly, *Add Buss…* can be selected from any audio input or output chooser, which goes to the *Audio* tab instead (see section 0.2.2.1).

› The *output chooser* lets you select where the track's final audio is getting routed to. The default selection is *Master*, which will serve us well in nearly all situations.

Additionally all tracks show available hardware *Note Outputs*, allowing you to route notes and other MIDI directly out from any track.

Note

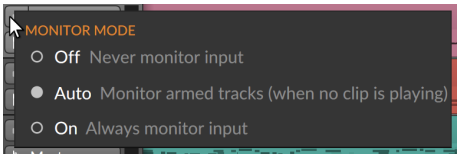
If you want to send out MIDI and return audio back into Bitwig with appropriate delay compensation applied, you should probably use the **HW Instrument** device (see section 19.11.5).

› The *monitor button* is now a three-state toggle on the left of the input chooser.

An empty icon represents monitoring set to *Off*.

An filled icon represents monitoring set to *Auto*.

An encircled icon represents monitoring set to *On*. Or you can right-click on any monitor switch to see a list of all modes.



The default setting for all tracks is *Auto*.

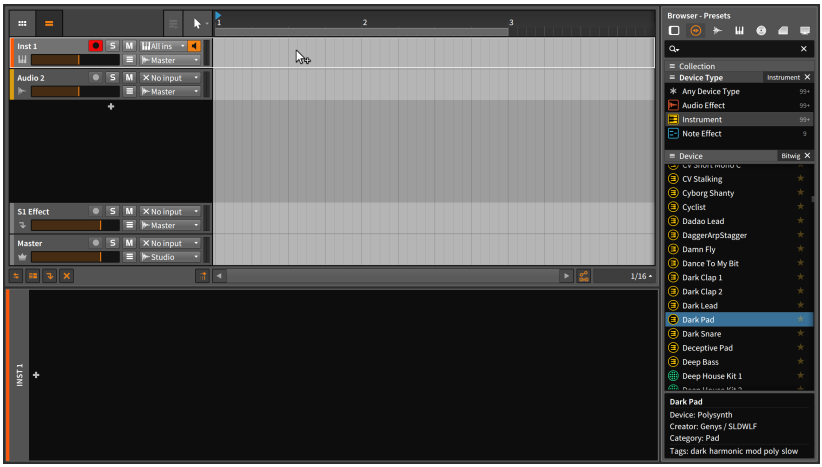
5.3.2. Recording Note Clips

A few steps are needed to successfully record a note clip. First, we need a sound source for our audio. Second, we need a MIDI source to record as notes. And then, we can hit record.

5.3.2.1. Loading an Instrument Preset

Note clips in Bitwig Studio — not unlike MIDI — are really just instructions to be interpreted by an instrument device. Notes themselves do not produce any sound. So before we record any notes, we should load an instrument preset so that our notes can be realized.

To load an instrument device: go to the **Browser Panel** and select the *Bitwig Presets* source. Under the *Category* filter, browse down to the *Synth* category, or something that sounds playable and fun to you. From the selection pane, drag any preset into the **Arrangement Timeline Panel**.



If you do not like the first device preset you load, repeat the above steps until you find one you appreciate.

5.3.2.2. Setting a MIDI Source

If you have a MIDI keyboard connected and already made Bitwig Studio aware of it, then it should be working already. By playing the keys, the instrument track's level meters should start showing audio.

Note

By default, all incoming MIDI channels will be received and written on record. A couple *Channel* settings are also available from the **Inspector Panel** when you select the header of the instrument track in question.

If you want a track to receive messages *From* a particular channel and/or want to record all incoming data *To* a single channel, just change these track settings here.

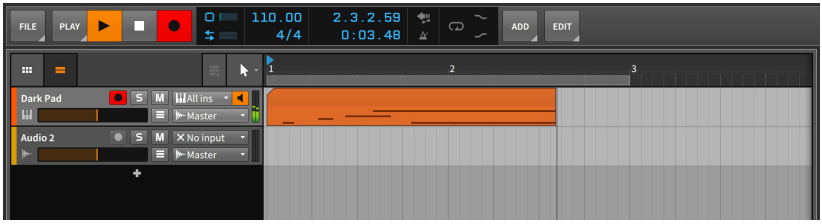
If you do not have a MIDI controller — or your MIDI device is all knobs and no keys — press [CAPS LOCK] to temporarily transform your computer keyboard into a MIDI keyboard. Pressing letters in the top two rows should trigger notes and cause the audio meters to dance.

Note

While [CAPS LOCK] is active, most key commands will not work.

5.3.2.3. Recording Notes

To record an Arranger note clip: enable the track's record arm button, enable the Global Record button, and then activate the transport and begin playing notes.



5.3.3. Recording Audio Clips

Unlike notes, the audio events that make up audio clips do not require any devices. They are already audio. So once we determine the audio source to be recorded, we should be good to go.

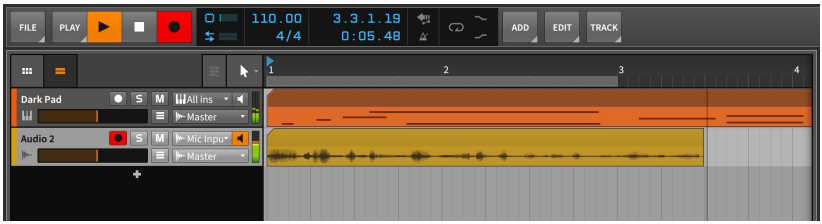
5.3.3.1. Setting an Audio Source

Whether you are using an external audio interface or the internal interface of your computer, you first need to set the desired input source in the track's input chooser (they will be at the top of the chooser list). If you enable the monitor button of the track and then send audio to this input, you should see the input in the track's audio meters.

Before recording, you probably want to disable the record arm buttons on all other tracks. Otherwise, you could trigger multiple tracks to record at once and alter or erase other clips in the process.

5.3.3.2. Recording Audio

To record an Arranger audio clip: enable the track's record arm button, enable the Global Record button, and then activate the transport.



5.3.3.3. Comp Recording in the Arranger

If the Arranger Loop is enabled, this does affect playback as previously described (see section 5.2). But it also affects recording, enabling a "cycle recording" mode that is ideal for capture audio for comping.

To do cycle recording in the Arranger: enable the Arranger Loop toggle, with the desired period set (by the Arranger Loop Selector). Then enable the track's record arm button, enable the Global Record button, and then activate the transport.

Note

Regardless of the Arranger Loop being on or off, there are two possible recording behaviors when the playhead encounters Arranger audio clips:

› If the audio clip *was* actively looping (with its *Looping* parameter enabled and with some amount of looping on the timeline), the section where recording happens will remove the old clip and record a new one.

› If the audio clip *was not* actively looping, new audio recordings will be added into the clip as comping takes and selected for playback.

So if you want to record new comping takes into a looping Arranger clip, you might consider using *Consolidate* for the relevant portion of that clip before recording.

Note

For information on editing comping expressions within a clip, see section 10.1.4.1.

6. The Clip Launcher

We have spent the last couple chapters working within the Arranger Timeline. And while the Arranger is absolutely crucial to music creation in Bitwig Studio, it is only half of the story.

The **Clip Launcher Panel** — also called the *Launcher* — is the logical Arranger's artistic brother. While the Arranger is an excellent way to lay out the fixed "story" of a song, the Launcher allows you to freely improvise with your clips. More on that soon.

We will start by getting an overview of the **Clip Launcher Panel** and its constituent elements. Next we will revisit some of the same concepts we saw with Arranger clips as they apply to Launcher clips. We will then investigate how Launcher clips relate to the transport and Arranger clips and see how Launcher clips are triggered. Finally, we will record Launcher clips and learn to capture the Clip Launcher's output on the Arranger Timeline.

Bitwig Studio is just one DAW, but it is the two sequencers within that provide limitless musical possibilities.

6.1. The Clip Launcher Panel

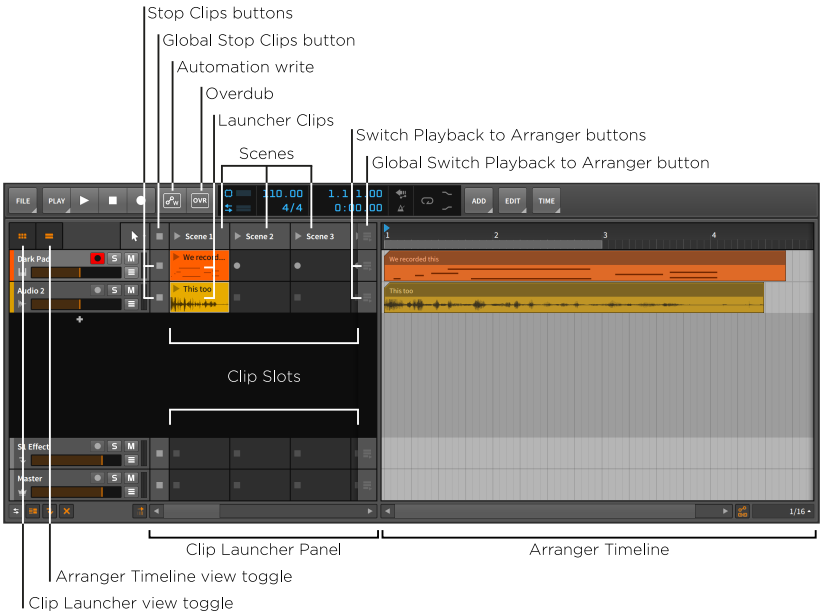
Charting out music from beginning to end is the way nearly all productions take place. But even from the earliest music, improvisation has been an important source of variation, inspiration, and life. Balancing these two poles — the programmed and the spontaneous — has been a central concern, all the way from Bach's time and his (literally) sacred music, up to the present day and our attempts to make electronic music engaging from the stage.

Aside from its unique perspective and purpose, the **Clip Launcher Panel** is also the only panel that loads directly into another panel. In this chapter, we will be learning about the Launcher within the **Arranger Timeline Panel**, but it can also be called up inside the **Mixer Panel** of the **Mix View** (see section 7.1.2).

The key difference between Arranger clips and Launcher clips is their purpose. Arranger clips are played back precisely at the designated time. But Launcher clips must be available whenever you want them, either for section-based composition (verse, chorus, bridge), or as pieces for a live performance, or however else you might use them. Arranger clips must be rigid, and Launcher clips must follow your whim.

6.1.1. Clip Launcher Layout

Let's begin by examining the **Clip Launcher Panel** beside the Arranger Timeline that we were just using.



What we see here is the same **Arranger Timeline Panel** as before, but now the view toggles for both the Clip Launcher and the Arranger Timeline are engaged. As a result, we see these two sequencers side by side within the panel.

The **Clip Launcher Panel** appears as a series of *slots* that are arranged across each track. Since tracks in the **Arrange View** are oriented horizontally, the **Clip Launcher Panel** is also arranged from left to right. In case more slots exist than can be shown at one time, the horizontal scroll bar at the bottom of the panel allows you to scroll to view all the slots.

The slots are made to house clips and have no functionality of their own. Whenever we refer to a "Launcher clip," we mean a clip that is housed within this Launcher sequencer.

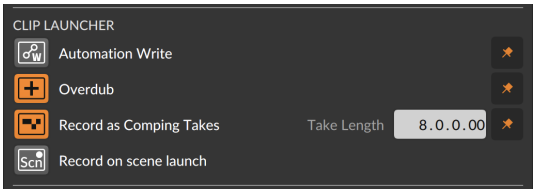
On each track before the clip slots begin is a *Stop Clips button*. Each of these buttons halts all clips that were playing on its track. And on each track after the last visible clip slot is a *Switch Playback to Arranger*

button. Each of these buttons restores the Arranger as the active sequencer for this track. The last section of this chapter will explain this relationship in detail.

Each vertical column of clips is a group called a *scene*. These groupings can be used for triggering or working with the constituent clips all together. If additional slots are needed, additional scenes can be created to provide them. Also note that each scene can be resized horizontally, providing more space to show the contents of its component clips and their playheads.

Similar to each track, the displayed scenes begin and end with the *Global Stop Clips button* and the *Global Switch Playback to Arranger button*, respectively. Each global button is the equivalent of triggering all track buttons of that kind. Again, the last section of this chapter will cover these functions in more detail.

Finally, various *Clip Launcher* settings are grouped within the *Play* menu.



› *Automation Write*: Enables automation recording to the **Clip Launcher Panel**.

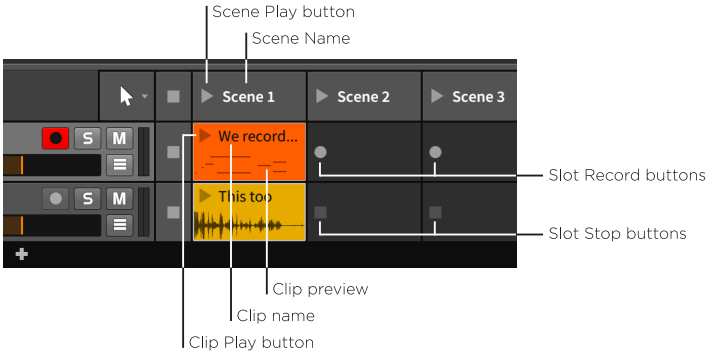
› *Overdub*: Merges incoming notes onto active clips the **Clip Launcher Panel** the next time the transport is started. Otherwise, note data is overwritten.

› *Record as Comping Takes*: Enables "cycle recording" on empty launcher slots at the defined *Take Length* (set within the *Play* menu). This begins writing comping data after the first cycle completes.

› *Record on scene launch*: Causes a scene launch to trigger recording into empty slots on all record-enabled tracks.

6.1.2. Within Launcher Clips, Scenes, and Slots

As for the appearance of Launcher clips themselves, there are only a few things to note.



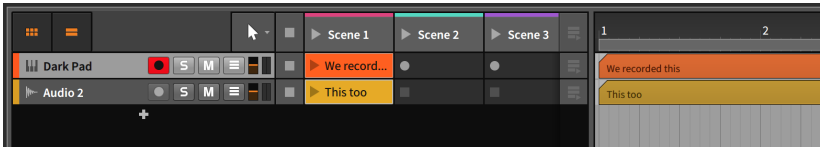
The crucial item within each clip and scene is the *play button*. This is the means by which you trigger the clip or scene. These play buttons also serve as indicators of which clips are active.

The top of each clip and scene also leaves space for that item's *name*, which is optional. As can be seen in the image above, scenes without names may be given automatic ones which you can always replace manually. And the *color stripe* at the top of the scene reflects the scene's color, just as the background of each clip shows its set color.

Note

In addition to a name and color, each scene can be given a text *comment* for your own use. These parameters are all shown for the selected scene in the **Inspector Panel**, or for all scenes in the *Sections* tab of the **Project Panel** (see section 14.2.4).

Below the play button and name of a clip may be a *preview* of the clip's contents. Clips that contain either notes or audio events will always have a preview, but the preview can be shown only when the track height is set to normal. When the **Arranger Timeline Panel** has tracks set to half size (as shown below), there is no room for the preview.



Finally, a couple of different buttons can appear within empty slots.

If the track is record-enabled, a *slot record button* will appear about where the play button would within a clip. Clicking this record button activates recording within the clip.

If the track is not record-enabled, a *slot stop button* will appear instead. This button is just an alias to the track's Stop All clips button, performing the exact same function.

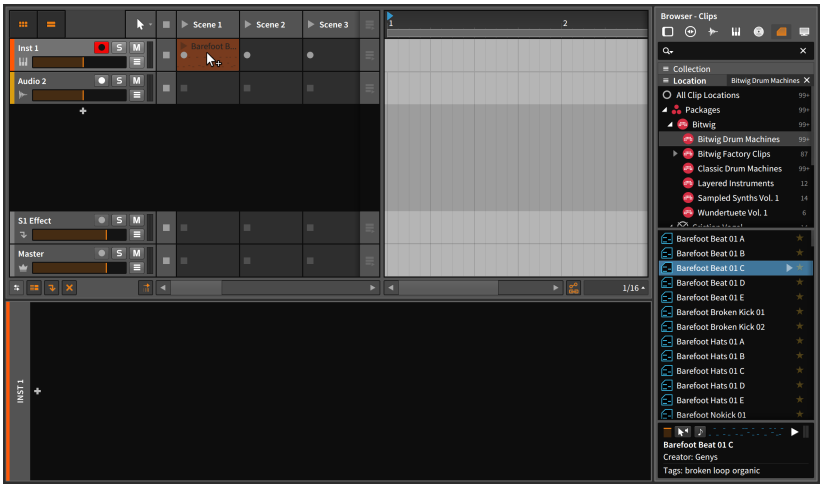
6.2. Acquiring and Working with Launcher Clips

Before manipulating clips in familiar ways, we must first get clips into the Launcher. We will start by recapping inserting and recording clips, and then look at moving clips between the Arranger and Launcher. Finally, we will see how length and looping adjustments are handled in the **Clip Launcher Panel** with the help of the **Inspector Panel**.

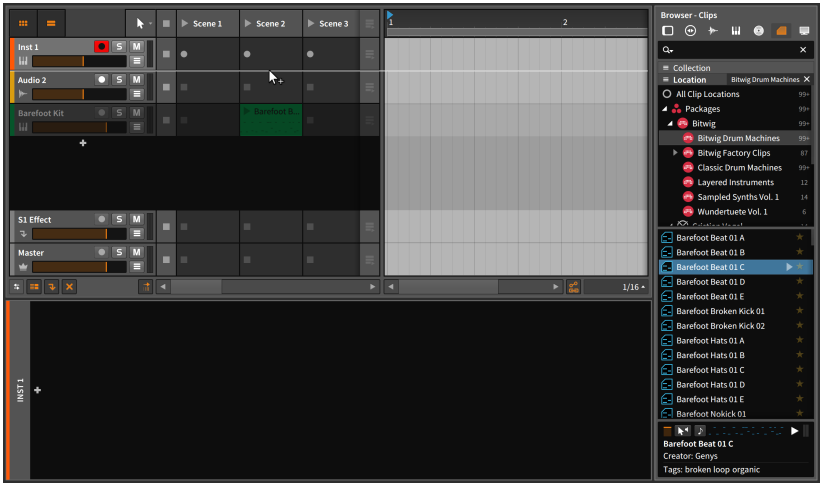
In the **Clip Launcher Panel**, we will recap inserting clips from the **Browser Panel**, look at moving clips between the Launcher and the Arranger, and see the options available for Launcher clips in the **Inspector Panel**.

6.2.1. Getting Clips from the Browser Panel

Getting clips from the **Browser Panel** onto a track is almost identical for the **Clip Launcher Panel** and the Arranger Timeline (see section 5.1.1). The only difference is where you drop the clip off.



And if the clip is dragged between two tracks, a new track will be created automatically as well.



Additionally, an empty Launcher clip slot has a + icon appear when the slot is hovered over. As in most other situations, clicking + opens the **Pop-up Browser**, in this case with a special configuration that offer clips and samples within your preset library, as well as any defined music locations.

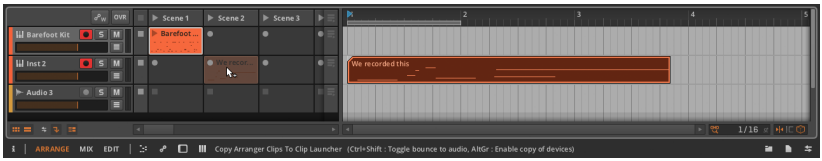
Note

For more information on working with the **Pop-up Browser**, see chapter 4.

6.2.2. Copying Clips Between the Arranger and Launcher

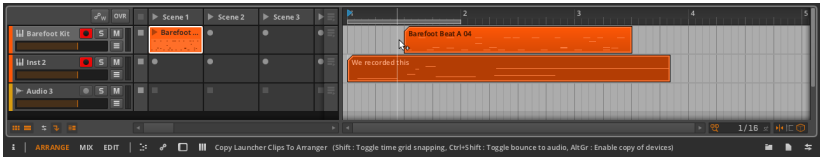
Copying a clip from one sequencer to the other follows the same pattern as all other movements that we have made.

To copy an Arranger clip to the Launcher: click and drag the clip from the Arranger Timeline to the desired slot on the appropriate track.



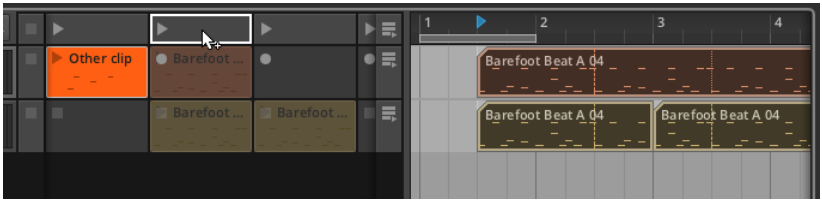
If multiple Arranger clips are selected, the clips will be copied into successive slots.

To copy a Launcher clip to the Arranger: click and drag the clip from the Launcher to the desired timeline position on the appropriate track.



If multiple Launcher clips are selected, the clips will be placed into the Arranger consecutively.

Scenes can also be copied from the Launcher to the Arranger Timeline. And conversely, any combination of Arranger clips can be copied to a scene by dragging them over.

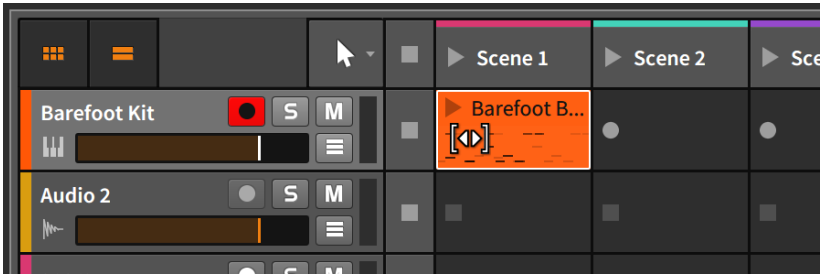


All of these copy functions can also be done into new tracks.

6.2.3. Sliding Launcher Clip Content

The content of one or multiple clips can also be shifted left and right from the **Clip Launcher Panel**. Sliding content in this fashion preserves the length of each clip, simply sliding the contained note or audio events (including any associated expressions) earlier or later in time.

To slide the content of a clip: mouse over the top half of the waveform. Then [ALT]-click ([CMD]+[ALT]-click on Mac) and drag horizontally.



You can optionally add the [SHIFT] key while dragging to toggle the snapping behavior.

6.2.4. Sub Scenes and Group Tracks in the Launcher

When working with group tracks in the Arranger, we encountered the idea of meta clips (see section 5.1.9). In the Launcher, a similar idea exists in the form of *sub scenes*.

Each group track has its own row of sub scenes. Each sub scene uses color blocks to identify which contained tracks have clips that fall within that sub scene. Just as a scene allows you to trigger a set of Launcher clips across your project, a sub scene allows you to trigger Launcher clips contained by that group track's component tracks. And while clips within a sub scene are playing back, miniature clip playheads are shown within the sub scene to indicate the current playback position of each of its clips.

Also similar to meta clips in the Arranger, sub scenes act as aliases for the clips they contain. Sub scenes can be moved by dragging and dropping, they can be cut or copied or pasted in the normal ways, they can be deleted, and they can even be sources or destinations for dragging clips between the Launcher and Arranger.

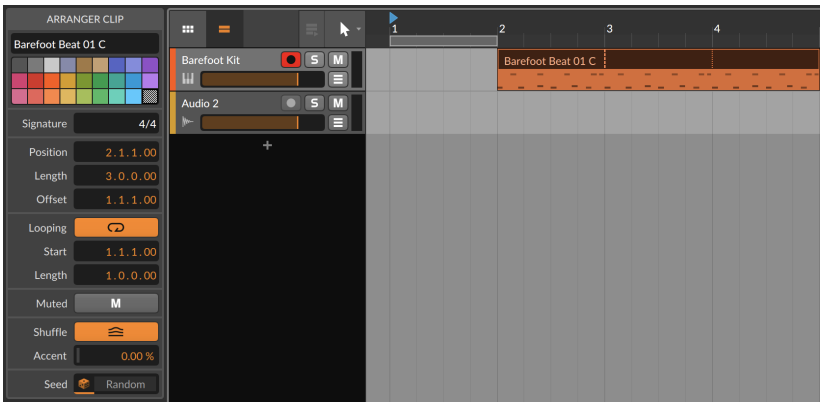
Note

Like regular scenes, sub scenes can also have colors assigned to them. These color stripes will be shown on screen when you navigate into that group track (see section 5.1.9).

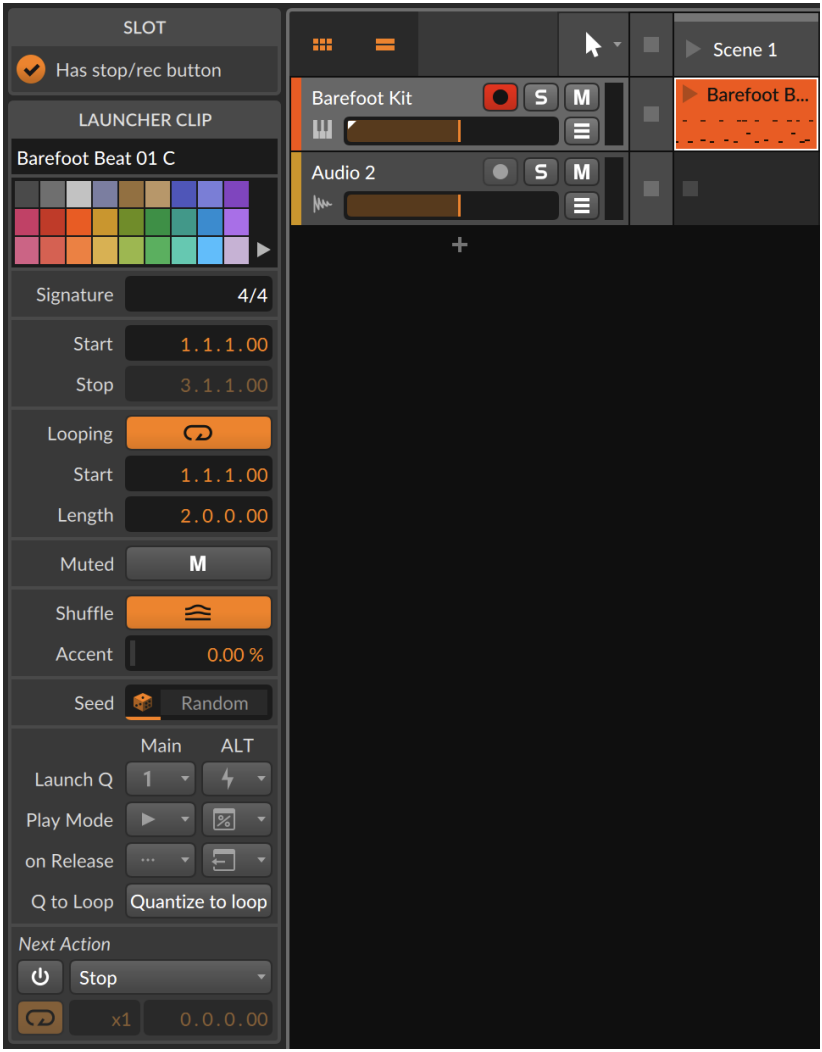
6.2.5. Launcher Clip Parameters

The Arranger Timeline had a convenient graphical view for visualizing the length and loop settings of a clip. While the **Clip Launcher Panel** does not have its own graphical editor, we will always have the **Inspector Panel**.

Launcher clip parameters are generally similar to Arranger clip parameters with a few important differences. In order to see how the **Inspector Panel** represents Launcher clip information, let's revisit the Arranger clip looping example from the last chapter.



In this case, I have copied the example Arranger clip into a Launcher slot. The resultant Launcher clip gives us these settings in the **Inspector Panel**.



We can see that the *Signature*, *Loop*, *Mute*, *Shuffle*, and *Seed* sections are identical to their Arranger clip counterparts, and that we have also seen all the functions available here already (see section 5.1.10).

We can also see that the initial *Start/Stop* section differs from the Arranger clip's *Time* (Position) model, and that the Launch and *Next Action* sections are completely new.

6.2.5.1. Start/Stop Section

Arranger clips had the *Time* (Position) section because they are always triggered at the exact position where they reside. Since Launcher clips do not share this sense of predestination, their parameters simply describe what portion of the clip should be played when triggered.

This section contains the following controls:

› *Start* sets the location within the clip that should be played first. This is very similar to adjusting the *Offset* of an Arranger clip, changing only which part of the Launcher clip should play back first.

› *Stop* sets the end of the clip contents that should be played. This setting is available only when *Loop* is disabled.

6.2.5.2. Launch Section

The Launch section controls how and when Launcher clips are triggered, and what should happen when a clip is released. Clips are usually triggered with a performance gesture, such as a click of the mouse or the push of a button, so quantization can be crucial here to keep clips in time around the Global Playhead.

When any of the parameters in this section are set to *Use Project Setting*, the same value from the *Settings* page of the **Project Panel** is in control (see section 14.2.1). When in this state, the icon of the current project setting will appear here so you know the behavior, but it will be slightly dim, hinting that the value is being set elsewhere. In a new project, the launch parameters of clips are set to *Use Project Setting*, allowing you to control the whole project's behavior in the **Project Panel**.

A cluster of three parameters define each launch behavior.

› *Launch Q(uantization)* sets the interval at which this particular clip will be triggered. This forces the clips we trigger to enter at the next appropriate beat grid. Since things that have already started cannot be shifted backwards in time, we must trigger the clips ahead of time so they can land on the next beat. (You can think of launch quantization as a performance-based version of absolute grid snapping.)

Note

While we can't go back in time, we can slightly delay the entire sequencer, giving you a window within which to trigger things "on time". This amount of delay / margin for error is set in the **Dashboard** under *Behavior* > *Sequencer* > *Latency*.

A beat-level setting (for example, *1/2*, *1/4*, *1/8*, or *1/16*) will play all newly triggered clips when the Global Playhead reaches the next grid line of that interval.

A bar-level setting (for example, *1 bar*, *2 bars*, *4 bars*, or *8 bars*) will play all newly triggered clips when the Global Playhead reaches the next bar of this interval. For example, a setting of *1 bar* would wait for beat 1 of the next bar to play, while a setting of *4 bars* would wait for the next fourth bar (e.g., bar 1, bar 5, bar 9, etc.) to be reached.

Off disables clip quantization, meaning the clip will begin playback the moment it is triggered.

› *Play Mode* determines where this clip will begin playback from. All "legato" modes aim to move from any already playing music's position to the same relative position in this clip — like jumping from beat 3 of the playing clip straight to beat 3 of this clip.

Four options are available:

Trigger from Start - Plays the clip from its beginning

Legato from Clip (or Start) - Starts relative to the playing clip's position (or when nothing was playing, starts from clip start)

Legato from Clip (or Project) - Starts relative to the playing clip's position (or when nothing was playing, starts relative to the global transport's position)

Legato from Project - Starts relative to the global transport's position

› *on Release* defines what you want to happen once you let go of your triggering gesture, whether it was the mouse you clicked or the controller pad you pressed. Four options are available:

Continue - Let the clip play and do nothing

Stop - Stops the clip

Return - Returns to the previously playing clip, or to the Arrangement if it was playing last

Next Action - Trigger the clip's *Next Action* immediately on release

You will notice that these three settings exist twice, once in a column labeled *Main* and again right after, labeled *ALT*. By defining two behaviors for how clips are launched (and released), you can create a more compelling performance by deciding how clips should play in the moment (see section 6.3.2).

Note

Scenes also have this identical layout, of the three *Main* launch settings and their twinned three *ALT* launch settings. When the scene's *Override Launch Settings* option is enabled, then triggering the scene will force its settings on all clips being launched. But when you directly trigger a clip, its own settings are always used.

One additional parameter lives in this section.

› *Q(uantize) to Loop* toggles clip quantization to be based on the loop start point instead of the clip start. This allows you to trigger a clip with a lead-in that plays once, like musical pick-up notes.

6.2.5.3. Next Action Section

Next Action is the option to determine what should happen after this clip has played for a set amount of time. The *Next Action Function* is set by the drop-down menu (all options are described in the next section), and if the *Enable Next Action* toggle is turned on, the function will be executed at a set time after the clip started playing.

Note

Even if the *Enable Next Action* toggle it disabled, the *Next Action Function* may still be used if set as one of the clip's release actions (see section 6.2.5.2).

The timing of the next action is set in one of two ways, with its row offering a toggle to tie timing to the length of the clip. When enabled, a parameter called *Loop Count before Next Action* appears, letting you set a number of whole times that the clip should loop before its next action is fired. (If the clip is not set to *Loop*, the next action will simply fire when the clip finishes playing back once.) And a dim display will appear at the right, indicating the effective number of bars and beats.

And if the clip timing toggle is disabled, you get a simple *Next Action Time* parameter for manually setting the bars, beats, etc. before the next action fires.

6.2.5.3.1. Local and Global Next Action Functions

The following *Do* actions are listed at the top of the action list. They relate either to the clip itself or any clip on the same track:

› *Stop* simply stops the clip.

› *Return to Arrangement* resumes playback of the Arrangement Timeline for this track.

› *Return to Last Clip* resumes playback of the Launcher clip that was playing immediately before the current clip. If no clip was playing when this one was triggered, the clip is stopped.

› *Play Next* triggers the next available Launcher clip. If the current clip is the last clip on the track, the clip is stopped.

› *Play Previous* triggers the previous available Launcher clip. If the current clip is the first on the track, the clip is stopped.

› *Play First* triggers the first Launcher clip on the track.

› *Play Last* triggers the last Launcher clip on the track.

› *Play Random* triggers a Launcher clip from the track at random, which could potentially retrigger this clip.

› *Play Other* triggers a different Launcher clip from the track at random. The current clip will not be retriggered.

› *Round-robin* triggers the next available Launcher clip. If this is the last clip on the track, the first Launcher clip on the track is triggered.

6.2.5.3.2. Using Clip Blocks with Next Actions

The second half of the *Do* actions list makes use of *clip blocks*, which are groups of clips that sit side by side with blank slots around them.

In the image above, the *Drums* track has three clip blocks (which I have manually colored), each containing two clips. The number of clips in a block is completely up to you, and each block needn't contain the same number of clips.

These functions include:

› *First in Block* triggers the first Launcher clip in the current clip block.

› *Last in Block* triggers the last Launcher clip in the current clip block.

› *Random in Block* triggers a Launcher clip from the current clip block at random, which could potentially retrigger this clip.

› *Other in Block* triggers a Launcher clip from the current clip block at random. This clip will not be retriggered.

› *Round-robin in Block* triggers the next available Launcher clip in the current clip block. If the current clip is the last in the block, the first Launcher clip of the clip block is triggered.

› *First in Next Block* triggers the first Launcher clip in the next clip block. If the current clip is within the last clip block, this will act like the *Stop* function.

› *Random in Next Block* triggers a Launcher clip from the next clip block at random. If the current clip is within the last clip block, this will act like the *Stop* function.

› *First in Previous Block* triggers the first Launcher clip in the previous clip block. If the current clip is within the first clip block, this block's first clip will be triggered.

› *Random in Previous Block* triggers a Launcher clip from the previous clip block at random. If the current clip block is the first block, a clip will randomly be triggered from this block.

› *First in Other Block* triggers the first Launcher clip from a different clip block.

› *Random in Other Block* triggers a Launcher clip at random from a different clip block.

6.3. Triggering Launcher Clips

Just as the last chapter looked at playing the Arranger and its clips, we should now discuss triggering Launcher clips. But now that we have two sequencers in play, we must first discuss the relationship between the Arranger and Launcher. Understanding their alliance will allow you to get the most — and possibly most interesting results — out of Bitwig Studio.

6.3.1. How the Arranger and Launcher Work Together

When thinking about Bitwig Studio's two distinct sequencers, it helps to consider the following concepts:

› The transport drives all timing functions, whether it is the playback of Launcher clips, the recording of Arranger clips, or vice versa.

› The Arranger Timeline's Beat Ruler also has influence over the **Clip Launcher Panel**. Launcher clips may be played back whenever you choose, but the launch quantize feature described above is regularly

used for the sake of coherence and musicality, aligning launched clips with arranged ones according to your wishes.

› On each individual track, either the Launcher or Arranger will be active at any given time.

› By default, each track starts with the Arranger Timeline active. The Launcher will take over for a track after a Launcher clip is either triggered or recorded, or the track's Stop Clips button is pressed. The Arranger will regain control only after the track's Switch Playback to Arranger button is pressed.

› All tracks can be toggled in unison from the Arranger to the Launcher and back. The Launcher will take over all tracks when either the Global Stop Clips button is pressed or a scene is triggered. The Arranger will regain control of all tracks when the Global Switch Playback to Arranger button is pressed.

The takeaway is that you can act like Bitwig Studio has just one sequencer, by using only the Arranger Timeline (to create a completely composed song, for example) or only the Clip Launcher (to take elements you have made and freely improvise a structure). You could also keep most tracks playing what you programmed in the Arranger, and occasionally shift some tracks to the Launcher for the sake of improvisation.

Once the two sequencers make sense to you, there is no "right way" to use them. Only options.

6.3.2. Triggering Launcher Clips

To trigger a Launcher clip normally: click the play button in its top left corner. Or press a controller pad that is mapped to this clip. This will trigger the clip using its *Main* launch behaviors (see section 6.2.5.2).

To trigger a Launcher clip with its alternate (ALT) behavior: hold [ALT] and then click the play button in its top left corner. Or hold the controller's [SHIFT] button (see your controller's documentation for support information) and then press the pad that is mapped to this clip. This will trigger the clip using its *ALT* launch behaviors (again, see section 6.2.5.2).

If the transport was stopped, triggering a clip immediately activates the transport. (Otherwise, no clip could play.)

Once a clip is triggered, a black box appears around the play button to mark this as an *active clip*. A clip remains active until either a different clip on that track is triggered, the track's (or the Global) Stop All Clips button is triggered, or the track's (or the Global) Switch Playback to Arranger button is pressed. When the transport is activated, all active clips resume playing.

In the image above, you may also notice a vertical line going thru the active clip. Each active clip has its own *clip playhead* that indicates the play position within the clip while the transport is active.

To release a Launcher clip normally: simply let go of the mouse button (or controller pad) that you pressed to trigger the clip. Whenever you let go, the *Main Release Action* will be executed immediately.

To release a Launcher clip with its alternate (ALT) behavior: hold down the [ALT] key, and then let go of the mouse button that you pressed to trigger the clip. Or if using a controller, hold the controller's [SHIFT] button (etc.), and then release the pad that triggered the clip. Whenever you let go, the *ALT Release Action* will be executed immediately.

Note

Triggering a clip is distinct from releasing it; these are two separate actions. So you can actually mix and match a normal *Main* trigger with an *ALT* release (by adding [ALT] or the controller's [SHIFT] button while holding the pad). Or use an *ALT* trigger, then release the computer keyboard's [ALT] key (or controller's [SHIFT] button) first, and finally let go of the clip to make a *Main* release. It may sound complicated, but your fingers will get used to these new gestures quickly.

To trigger a scene: click the play button in its top left corner. This will trigger all clips that exist within the scene and Stop All Clips for tracks that contain no clip for the scene.

Identical to clips, a normal click/press will execute a *Main* trigger of the scene; holding [ALT] will make an alt trigger; and so on with releases, etc. etc.

Note

If the *Record on scene launch* setting is enabled, empty slots on record-enabled tracks will also begin recording (see section 6.1.1).

And if the scene's *Override Launch Settings* option is enabled, all clips will launch with the scene's defined *Main* or *ALT* trigger behaviors. This can be especially useful to align the *Launch Q(uantization)* timing for all of a scene's clips, etc.

To stop all clips on a track: click either the track's Stop All Clips button or a stop button within an empty slot.

This stops Arranger clips as well since the Launcher is given control of the track. Each Stop All Clips button will take effect at the default launch quantize interval.

To stop all clips: click the Global Stop All Clips button.

While this will stop all clips after the default launch quantize interval, the transport remains active.

To return control of a track to the Arranger: click the track's Switch Playback to Arranger button.

This will take effect immediately, regardless of the default launch quantize setting.

To return control of all tracks to the Arranger: click the Global Switch Playback to Arranger button.

This will take effect immediately, regardless of the default launch quantize setting.

6.3.3. Launching Time Signature Changes

Just as you can insert time signature changes in the Arranger Timeline (see section 5.2.2), you can also trigger time signature changes from the Clip Launcher. This method of automation can be achieved by placing a Launcher clip with a set *Signature* parameter (see section 5.1.10.1) on the master track. Whenever a clip like this is triggered, the transport's time signature will be overridden.

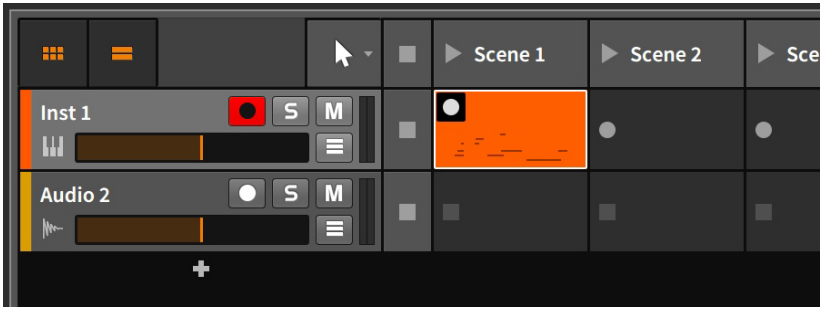
6.4. Recording Launcher Clips

We will finally return to recording with the Clip Launcher, both to record new Launcher clips and to print the results that come out of the Launcher.

6.4.1. Recording Clips

All the same requirements apply for recording Launcher clips as Arranger clips (see section 5.3).

To record a Launcher note clip: enable the track's record arm button, click a blank slot's record button, and then begin playing notes.



If the transport was inactive, it will automatically start once you click the slot record button. If the transport was already active, it will continue moving, and recording will commence after the default launch quantize interval.

Note

The scene play buttons can also trigger empty Launcher slots to record clips when the *Record on scene launch* setting is enabled (see section 6.1.1).

6.4.2. Comp Recording in the Launcher

When recording audio in the Launcher, you can also do comp recording. This mode works like "cycle recording," where new recording is broken up into takes at a set interval.

To do cycle recording in the Launcher: enable *Record as Comping Takes* from the *Play* menu, and set the desired *Take Length* right beside it (see section 6.1.1). As long as your audio source is selected and the audio track is armed (see section 5.3.3.1), clicking a blank slot's record button will now start "cycle recording."

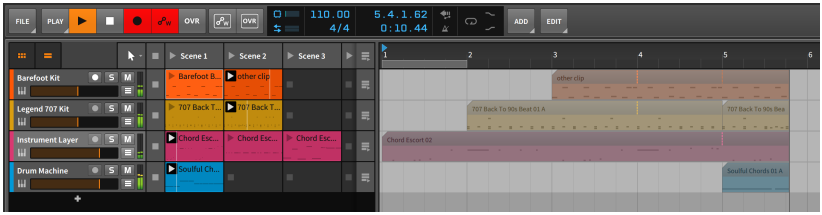
Note

For information on editing comping expressions within a clip, see section 10.1.4.1.

6.4.3. Record to Arranger Timeline

As one more form of interaction between the Launcher and Arranger, the result of all triggered Launcher clips can be recorded directly to each Arranger track. This is a way to capture an improvisation, whether from an early production phase, a stage performance, or whatever else you can imagine.

To capture clips and/or scenes triggered from the Launcher into the Arranger: enable the Global Record button, activate the transport, and then trigger the clips/scenes.



A few notes that may be helpful here.

› If you activate the transport by triggering your first clip or scene, recording will begin from the Play Start Marker.

› If you deactivate the record arm buttons of individual tracks, you will avoid recording empty clips to the Arranger tracks.

› Control changes can also be captured, making for a fully editable transcription.

› All Launcher clips recorded into the Arranger will create clips with defined *Seed* values (see section 5.1.10.7). If the Launcher clip had a set *Seed* value, that value is maintained. And if the Launcher clip was fully *Random*, the seed value used during recording will be set in the new Arranger clip. The result is that randomized elements connected to the *Seed* value will be replayed exactly as you heard them while recording.

7. The Mix View

For three of the last four chapters, we have dealt exclusively with the **Arrange View**, and more particularly with the **Arranger Timeline Panel** that is housed there. And while we are not done with the **Arrange View** (it will be back), it is time to see another of Bitwig Studio's views.

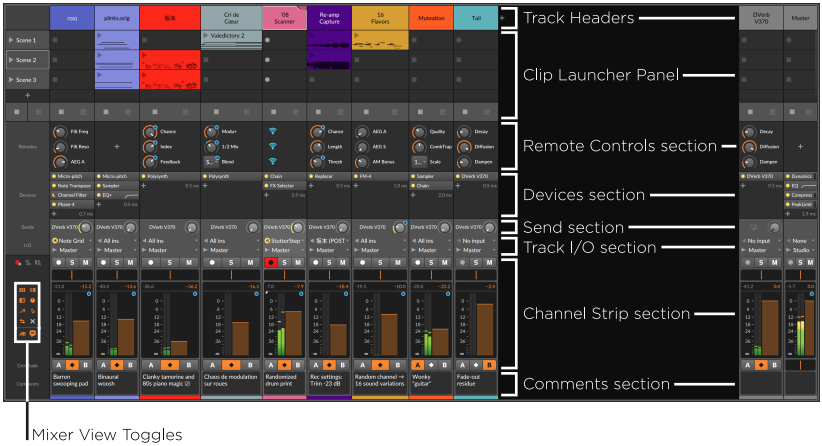
In this chapter we will take up the **Mix View** and its central **Mixer Panel**. As the purpose of each view is to provide tools organized around a musical task, the clear task of this view is *mixing*, the necessary art of adjusting and blending your tracks so that they play well together. This happens first at the master track, and then on to the real world, in headphones and on speakers.

We will begin by taking stock of the **Mixer Panel**, examining various functional details along the way. We will also look at places outside of the **Mix View** where mixing functions crop up. Finally, we will see how the master track's output can be easily controlled with the **Output Monitoring Panel**.

7.1. The Mixer Panel

We will begin our examination with the **Mixer Panel** itself. Within the **Mixer View**, the **Mixer Panel** is the lone central panel.

The **Arranger Timeline Panel** was oriented horizontally, which is perfectly sensible for viewing the left-to-right timeline of your music. Just as sensibly, the **Mixer Panel** is laid out vertically like a traditional mixing board, with each available section stacked one atop the other.

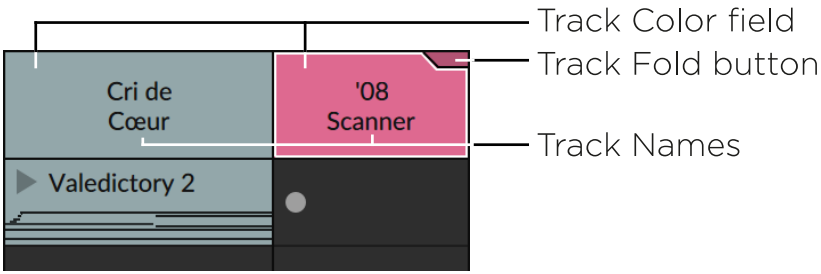


The first and next to last sections (track headers at top, channel strip sections near the bottom) will always be visible. The *View Toggles* on the bottom left allow you to decide whether each of the eight other sections are shown or hidden, with another two options for whether the FX tracks and deactivated tracks should be displayed.

We will take the sections of the **Mixer Panel** in order, starting at the top.

7.1.1. Track Headers

The *track headers* in the **Mixer Panel** contain the same information as the track headers of the **Arranger Timeline Panel**.



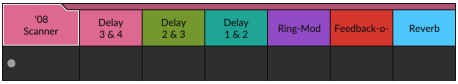
Each track header consists of at least three parts:

› *Track Color field*: The track's assigned color.

> **Note**
>
> You can also right-click in the Mixer view toggles toggle for a *Fill header backgrounds* option. Disabling this view preference will change the painting style to just show a track color stripe at the top of each channel.

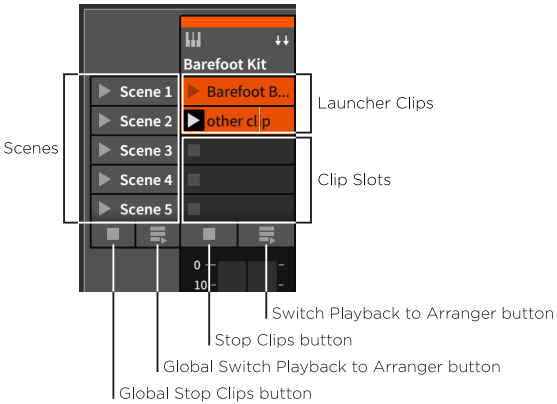
› *Track Name*: The title assigned to the track.

› *Track Fold button:* Available for tracks whose primary signal path includes certain container devices (such as **Drum Machine**, **Instrument Layer**, or **FX Layer**). These devices all contain *layers*, which have some of the attributes of tracks — channel strip elements when appropriate (volume, pan, sends, etc.) and comments. When a fold button is enabled, the track's channel strip expands to the right, exposing all layers in the top-level of the container.



7.1.2. Clip Launcher Panel

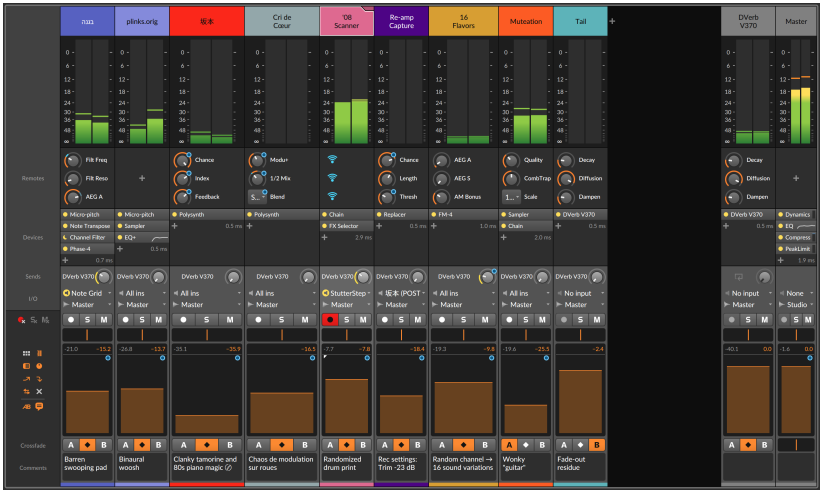
The **Clip Launcher Panel** contains all of its usual elements and functionality when loaded into the **Mixer Panel** (see chapter 6).



Its elements have just been rearranged to fit the vertical orientation of tracks in this view. Also note that each track can be resized horizontally to provide more screen space for viewing the track's clips.

7.1.3. Big Meters Section

These high-resolution stereo audio meters — aka the *big meters* — liberates each channel's output level meters from the channel strip section (see section 7.1.8).



Note that the big meters section is only available when the **Clip Launcher Panel** is disabled inside the **Mixer Panel**.

7.1.4. Track Remotes Section

The *track remotes section* gives you the remote controls for each track directly on the **Mixer Panel**.

This is a unique opportunity to have parameter controls from different tracks appear side by side, both for visualization and interaction. Right-clicking in this section offers relevant options.

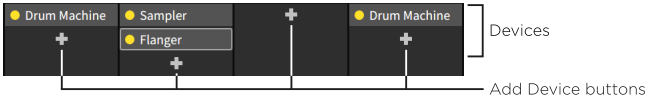
Delete Remote Control will remove the control you clicked on, replacing it with the Wi-Fi icon to map a different parameter in its place.

Alias preset remotes on tracks sets whether tracks should borrow certain device remote control pages until you create your own track remote controls. (Track remotes will always win, but if you want to work with an aliased device, just keep it.)

Track remotes shown in mixer sets the number of remote controls you want displayed here. So if this was set to *3*, only the first three remotes would be displayed for each track.

7.1.5. Devices Section

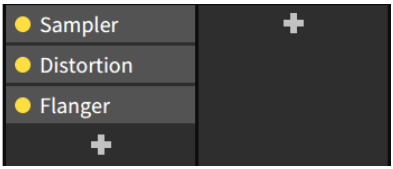
The *devices section* provides a list of all the top-level devices on each track.



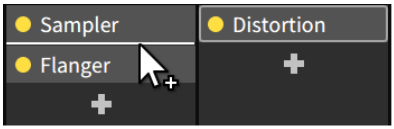
This is not to be confused with the **Device Panel** (see section 8.1), where parameters can be accessed and edited. This section can be used to call up the **Device Panel**, move/copy the devices present, and add new devices.

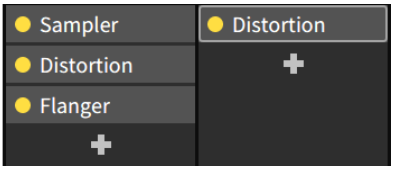
To focus on a track's device within the Device Panel: double-click the device.

To move a device: click and drag the device to the desired location.

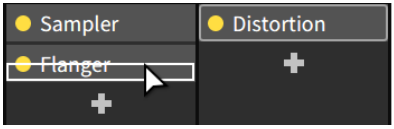


You can also hold [ALT] to copy the device.



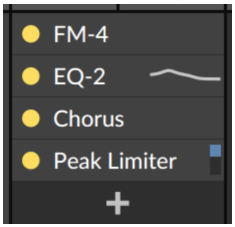


To layer a device with another: [SHIFT]-click and drag the device over top of the device where the layer should be inserted.

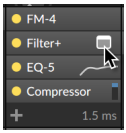


To add a device: click the track's *Add Device button* (the + icon) to pull up the *Pop-up Browser* (see chapter 4).

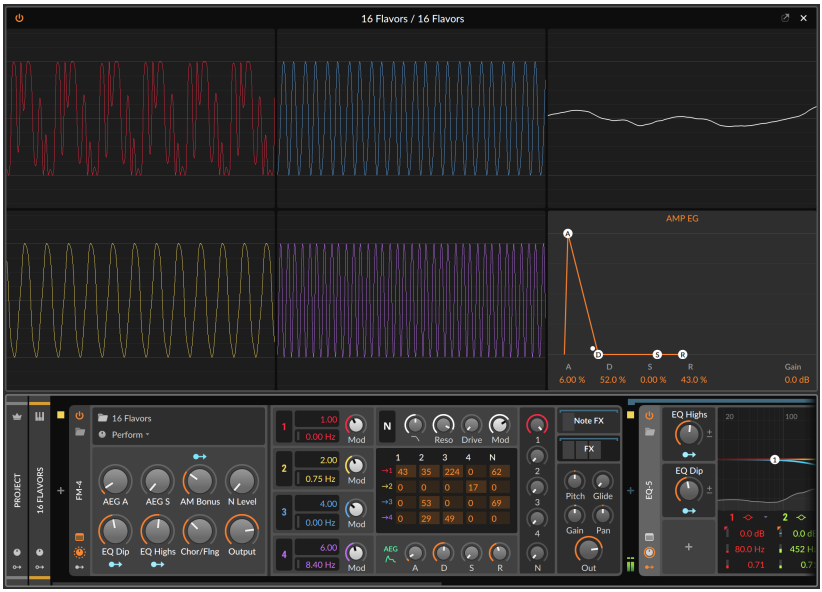
Also note that certain devices include mini displays within this interface. This includes EQ curves (for **EQ+**, **EQ-5**, and **EQ-2**) or gain reduction amounts (for **Compressor**, **De-Esser**, **Dynamics**, **Gate**, and **Peak Limiter**).



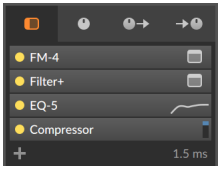
Finally, when mousing over the devices section, any device with an **Expanded Device View** (see section 8.1.4) will offer a button for opening it.



*To access a device's **Expanded Device View** from a mixer interface:* hover over the device and click the **Expanded Device View** button. Bitwig will then reveal the **Device Panel**, scroll the device chain so the selected device is on screen, and open the **Expanded Device View** in the window's central panel area.



The same is true when viewing the devices within a track's **Inspector Panel**, except most **Expanded Device View** buttons will always be shown.



The exception is devices with both mini displays *and* an **Expanded Device View** (like **EQ+** and **EQ-5**), which must still be hovered over.

7.1.6. Send Section

The *send section* provides a level knob for each FX track in your project. Other than the master, this section is available on all tracks and any visible layers.

The send section is the block of right-aligned knobs beneath the device section. In the image above, there are four FX tracks present, with corresponding send knobs on each track. These sends allow us to pass a portion of each track's audio into the various FX tracks. Using a send does not affect a track's main output level.

For each individual send, you can decide whether the audio being sent is taken before the track's volume fader has been applied or after. Since this setting is relative to the track's fader, the settings are called *Pre* (for pre-fader) and *Post* (post-fader). A third choice of *Auto* is selected by default, permitting the FX track targeted to decide whether *Pre* or *Post* should be used (see section 7.2.3). To make this immediately readable on the mixer, the indicator ring around each send knob is painted correspondingly — normal *Post* sends are colored yellow, and *Pre* sends are blue.

To set a send's source setting: right-click the send, and then select the appropriate setting from the context menu.

Finally, each send can also be disabled. This can be a useful way to 'bypass' a routing without losing your level setting, and it is also a CPU saver.

To toggle a send: click on the name of the particular send. It will toggle between bright text and a regular knob (when activated) and dim text and a dim knob (when disabled).

To toggle all sends on a particular track: [SHIFT]-click any send of that track. If the particular send you clicked on was activated, then all of the track's sends will be disabled, and vice versa.

There is also a *Disable All Unused Sends* function, to save CPU — and to make the mixer easier to read. (For assigning shortcuts to functions, see section 0.2.2.4.)

Note

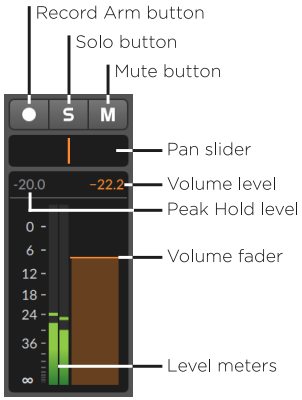
New projects started in Bitwig Studio v4.3 or later have all sends disabled by default. The first adjustment of any send knob (even clicking on it) will automatically activate the send. This keeps all sends available with just one click, and keeps CPU usage to a minimum until each send is needed.

7.1.7. Track I/O Section

The *Track I/O section* allows you to assign the input and output paths for each track. This is exactly the same as it appears in the **Arranger Timeline Panel** (see section 5.3.1).

7.1.8. Channel Strip Section

The *channel strip section* contains most of the same control items as the track headers of the **Arranger Timeline Panel**.



This section contains the following controls:

› *Record Arm button*: Record enables the track.

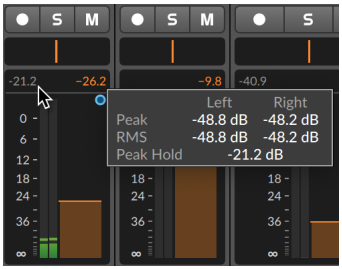
› *Solo button*: When any channel has its solo button enabled, only those with solo enabled will output their audio.

› *Mute button*: Disables the channel's audio output.

› *Pan knob:* A stereo placement control for the channel.

› *Peak Hold level:* A readout of the strongest momentary level received since transport play started. Clicking the peak hold level on any track will reset this value for all tracks (as will stopping and restarting the transport).

Also note that hovering over this indicator will present a pop-up showing current *Peak* and *RMS* levels in stereo.



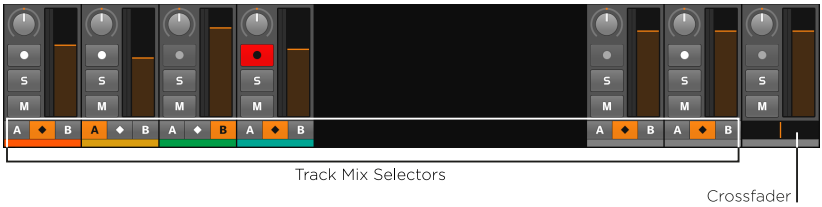
› *Volume level*: A readout of the channel's current volume setting.

› *Level meters*: Stereo audio meters that display the channel's output level.

› *Volume fader*: A final level control for the channel.

7.1.9. Crossfader Section

The *crossfader section* contains a *Global Crossfader* on the master track. Every other track has a *Track Mix Selector*, which allows you to designate whether that track belongs to the *A*mix, both mixes, or the *B* mix, respectively.



› When a track mix selector is set to the *A* position, that track will be unaffected when the Global Crossfader is anywhere between the leftmost and center positions, but that track's level will be gradually faded out as the Global Crossfader moves from the center position to the far right.

› When a track mix selector is set to the *B* position, that track will be unaffected when the Global Crossfader is anywhere between the rightmost and center positions, but that track's level will be gradually faded out as the Global Crossfader moves from the center position to the far left.

› When a track mix selector is set to the both mixes option (the diamond button at center), that track is completely unaffected by the Global Crossfader.

Note

Realize that the crossfader settings are active regardless of whether the crossfader section is visible or not.

In addition, the Global Crossfader's current position is also available as a modulator signal for any device on any track to use (see section 19.27.3.3).

7.1.10. Comments Section

The *comments section* shows all track and unfolded layer/drum chain comments side-by-side. These could be used for recording settings, content reminders, mixing notes, or keeping track of to-do items — it is your choice.

Clicking in the comment area allows for adding comments on that object, or you can select and edit text as usual.

7.2. Other Mixing Interfaces

While the functions offered by the **Mix Panel** within the **Mix View** are extensive, a subset of these options can be found both in the secondary **Mixer Panel** and within the **Inspector Panel** when tracks are selected.

7.2.1. The Secondary Mixer Panel

Unlike the **Arranger Timeline Panel**, the **Mixer Panel** can be loaded as a secondary panel in other views. We will briefly examine this version of the panel within the **Arrange View**.

To load the secondary Mixer Panel: click the **Mixer Panel** button in the window footer, or press either [M] or [ALT]+[M] .

Note

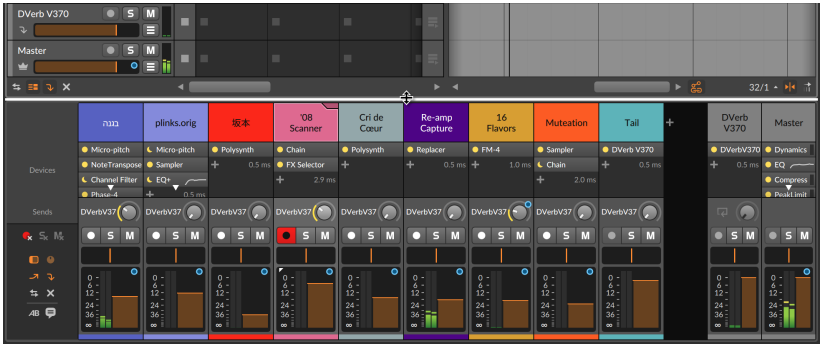
Not every view supports every panel. The available panels within a particular view will have their buttons shown in the window footer.

For a review of these buttons and how to load the various panels, see section 2.2.1.

Again the left edge of the panel includes the Mixer view toggles. But while all the toggles appear enabled, there are curiously few sections being displayed.

By looking closer at the view toggles, you will notice that they are mostly enabled but also grayed out. Bitwig Studio is acknowledging that you have these sections enabled, but is also letting you know that there isn't enough vertical space to display them all. While not all panels are resizable, this one is.

To resize a panel: mouse over the panel's border that faces the middle of the Bitwig Studio window. When the cursor becomes a bidirectional arrow, click and drag the border.



More of the enabled sections will progressively become visible (only the track remotes section is missing in the image above), each working the same as they did in the central **Mixer Panel**.

The only difference in this secondary version of the panel is that the **Clip Launcher Panel** and the big meters section are unavailable.

7.2.2. Mixing in the Inspector Panel

Finally, the **Inspector Panel** will also display certain mixing parameters whenever a track is selected. Whether in the **Arranger Timeline Panel** or the **Mixer Panel**, clicking on the track header will focus the **Inspector Panel** on that track.

The device section is available in the central panel, as well as the track remotes section if you switch to the second tab. And the track I/O and channel strip sections below are largely as they were in the **Mixer Panel**.

The send section is also similar, offer a clickable menu for each send's source setting (again, either *Auto* which inherits the FX tracks preference, or an explicit *Pre*- or *Post*-fader setting) just below the send's name.

7.2.3. Inspecting FX Tracks, and FX Track Sends

Everything just shown regarding the **Inspector Panel** holds true for FX tracks, but one additional parameter is worth noting: the button labeled *Pre-fader (Cue)*.

We have spoken of FX track's having their own send source preference, hence the *Auto* option on track sends. FX tracks default to the post-fader model, which is more common for mixing.

To switch an FX track's preference to pre-fader: simply enable an FX track's *Pre-fader (Cue)* button. Whether this is for cueing, monitor mixing, or some other special effect, any send to this track using the *Auto* source will follow and immediately start sending its signal pre-fader.

Finally, FX tracks (and FX layers within **Drum Machine**) also have sends. And since this means FX tracks can be routed to other FX tracks, the visualization is a little different on these tracks, and the logic could use a little explanation.

To avoid feedback mayhem, one simple rule is in place: FX tracks sending to their right are processed normally, and FX tracks sending to their left ('backwards') are delayed by one audio buffer. So let's walk thru the previous image as an example case.

The highlighted FX track (named *Delay-2*) is the second of four FX tracks in that project. The bottom two sends, labeled *Distortion* and *Reverb*, are sending to the right, so they appear the same as all other tracks' sends 3 and 4. The send to FX 1 (*Pasture*) is going to the left, however. So the little left facing triangle is reminding us of this relationship and the delay added to keep us safe. Finally, the send to FX 2 is indeed a feedback routing, hence the feedback icon. This routing

also uses a one buffer delay, and like all other send labels, the icon is clickable to enable or disable this send.

7.3. Master Track Routing

We mentioned earlier that the default output assignment of all tracks is *Master* (see section 5.3.1). This is referring to the name of the project's master track, which defaults to *Master*. If we rename the master track, the output choosers will follow suit.

As you can also see in the image above, the default output of the master track is set to *Studio*, which refers to the output set in the **Output Monitoring Panel**. We will now examine this panel and then see an example setup where a multichannel audio interface is used.

7.3.1. Output Monitoring Panel

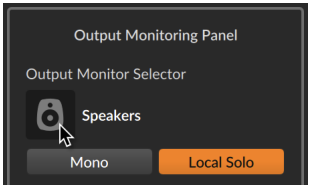
Clicking the **Output Monitoring Panel** view toggle (the speaker icon) in the window footer will call up the panel.

The top area of the panel displays the following audio settings:

› The *Output Monitor Selector* lets you select which pair of speakers and/or sets of headphones are being used for any track whose output is set to *Studio*.

  The monitoring options are those you have defined under *Settings* > *Audio* in the **Dashboard** (see section 0.2.2.1) using the same interface.

  To toggle a monitor: click the monitor's icon.



Only one pair of speakers can be active at a time, and any number of headphones may be used.

A fuller example with multiple monitoring options is presented in the next section.

› The *Mono* button toggles your studio output(s) from stereo to a mixed-down mono output.

› The *Local Solo* button applies when working with container devices that have discrete layers that include their own solo buttons, such as **Drum Machine**. When this function is enabled, solo logic is applied at the local device level. In this case, soloing one instrument layer/chains only mutes that device's other layers. This is the default behavior.

When this function is disabled, solo logic is applied at the global level. In this case, soloing one chain of a **Drum Machine** would effectively mute all other tracks in your project.

The middle area of the panel displays the following cue and preview settings:

› The *Cue / Preview* output menu sets the monitoring destination for both cue signals (when the *Solo as Cue* button is active) and **Browser Panel** previews.

This is particularly useful for performance situations. For example, this allows you to cue up certain signals in your headphones before adding them to the main mix.

› The *Solo as Cue* button alters how solo works. When this function is enabled, all solo-enabled tracks are also routed to the cue output, and all other tracks are routed as usual. Solo buttons themselves will be switched from *S* to *C* to reflect this.

When this function is disabled, normal solo rules apply (see section 3.1.3). When enabled, two additional parameters available:

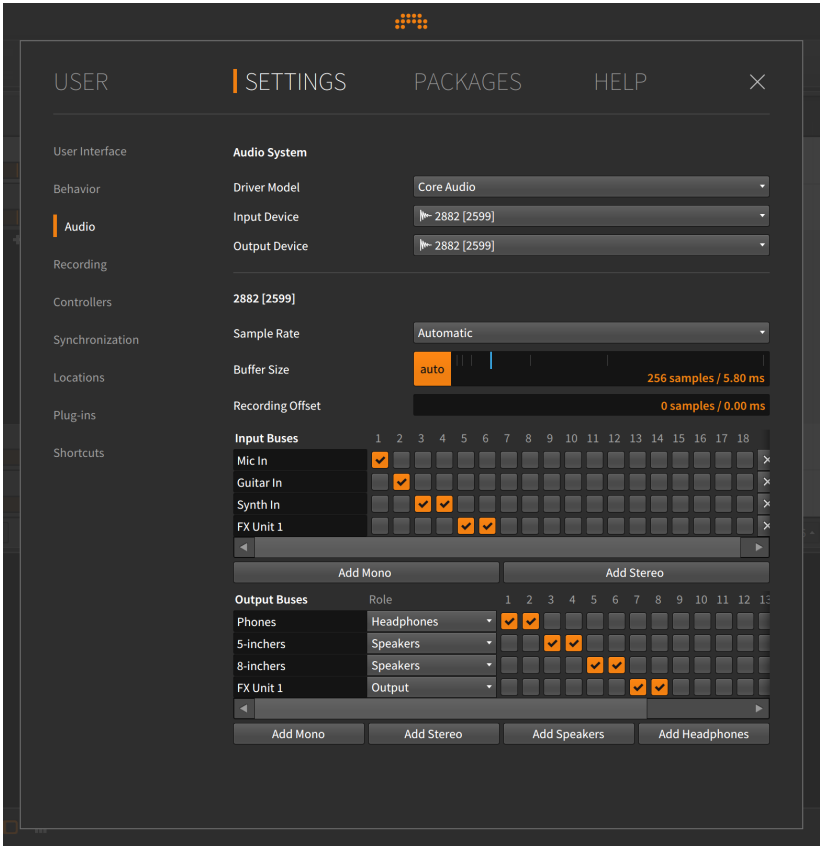
The *Post-Fader* toggle applies each track's volume level before routing the signal to the set cue path.

The *Cue Mix* control is a crossfader, blending between the output *Studio* buss output on the left and the cue signals on the right. This allows you to set a mix of the two on your cue monitor.

› The *Cue Level* controls sets the volume out for cue monitoring. As this buss is also used for previews in the browser panels, it is relevant even when the same audio path (say, headphones) is used for both your *Studio* and *Cue / Preview* outputs.

7.3.2. Multichannel Audio Interface

Most audio settings in the **Output Monitoring Panel** are only useful when you have more than one audio output option. To show one use case, I have connected a multichannel audio interface and made the following configuration in the *Audio* tab of the *Preferences* window.



Let's walk thru the example shown above.

Under *Audio Inputs*, three paths have been set up:

› *Mic In* is a mono input path that uses input 1 of our audio interface.

› *Guitar In* is a mono input path that uses input 2.

› *Synth In* is a stereo input path that uses inputs 3 and 4.

› *FX Unit 1* is a stereo input path (for a hardware effects unit) that uses inputs 5 and 6.

Under *Audio Outputs*, four paths have been set up:

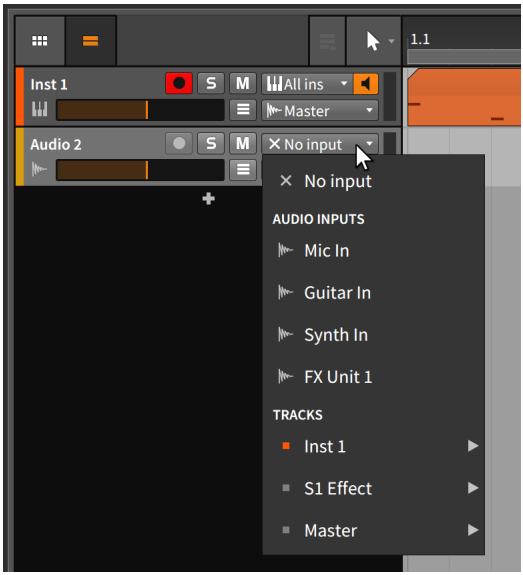
› *Phones* is a stereo output path that uses outputs 1 and 2 of our audio interface. This path has a role of *Headphones*.

› *5-inchers* is a stereo output path (for my 5" speakers) that uses outputs 3 and 4. This path has a role of *Speakers*.

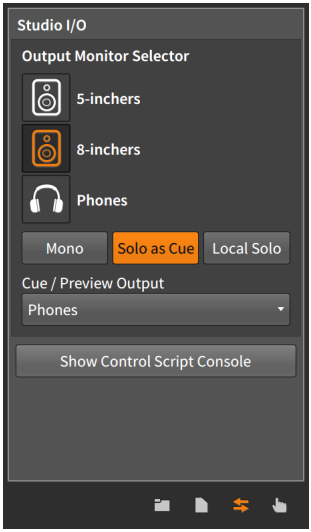
› *8-inchers* is a stereo output path (for my 8" speakers) that uses outputs 5 and 6. This path has a role of *Speakers*.

› *FX Unit 1* is a stereo output path (for a hardware effects unit) that uses outputs 7 and 8. This path has a role of *Output*.

The audio input paths will now be available in various places in the program, such as audio tracks' input choosers.



The audio output paths will be available from every track's output chooser, but they will also appear in the **Output Monitoring Panel**.



You will notice that only the monitoring options (*Speakers* and *Headphones*) are available here. Setting a path to an *Output* role makes it available for signal routing, but not for monitoring.

So in this example, my project's master track is routed to *Studio*. Because the *Output Monitor Selector* of the **Output Monitoring Panel** is set to *8-inchers*, anything reaching the master track is getting passed to my 8-inch speakers. And because *Solo as Cue* is enabled, any track that is solo-enabled (and any **Browser Panel** content being previewed) is routed to *Phones*.

If you have a simple setup and never click any of these options, audio will be routed to the right place. But if you have more sophisticated requirements, the settings shown here and Bitwig Studio's routing options will cater to your needs as well.

8. Introduction to Devices

The word "devices" has come up a few times now. For one thing, we have already been using them on instrument tracks and know how to load them (see chapter 4). For another, we have seen how other Bitwig Studio interfaces give us access to devices we were already using (see section 7.1.5). But in this chapter, we are finally dealing with the nuts and bolts of loading and using devices. This small exploration will benefit users of all levels.

Note

More "advanced" device concepts are covered in chapter 16, which assumes familiarity with the concepts found in this chapter.

The purpose of this chapter is not to teach you the particulars of any device. Instead, it is to acquaint you with accessing devices, their general interface concepts, and the layout of the **Device Panel**. A short section about the Bitwig devices themselves can be found at the end of this document (see chapter 19).

To expand slightly on chapter 1, each track in Bitwig Studio is equipped with a *device chain*. Each track passes all played-back audio, note, and MIDI signals to this device chain, which passes the messages from one device to the next, like a bucket brigade. The final device in the chain returns its audio output back to the track so that the mixing board controls (volume, panning, etc.) can be applied before the audio is passed to the track's assigned output buss.

Devices are grouped into the following descriptive categories:

› *Analysis.* Devices that merely visualize the signals that reach them. They make no effect on the audio chain they are a part of.

  Examples include **Oscilloscope** and **Spectrum**, which both have mini views and **Expanded Device View** options.

› *Audio FX.* Devices that manipulate incoming audio signals before passing them onward.

  Examples include **Blur**, **Freq Shifter**, **Ring-Mod**, and **Treemonster**.

› *Clap.* Clap drum element instruments that use incoming note signals to synthesize audio.

  Examples include the electronic drum emulator, **E-Clap**.

› *Container.* Utility devices whose primarily function is to host other devices.

Examples include **Instrument Layer** (for stacks), **Instrument Selector** (for cycling notes [via Round-robin, Keyswitches, etc.] to various instruments), and **Multiband FX-2** (for multiband audio processing).

› *Delay.* Delay line-based processors that operate on their incoming audio signals.

Examples include various configurations of single tap delay lines (**Delay-1** and **Delay-2**) and multitap delay lines (**Delay-4**).

› *Distortion.* Shapers and other mangling processors that operate on their incoming audio signals.

Examples include **Amp**, **Bit-8** (a signal degrader) and **Saturator**.

› *Drum Kit.* Drum kit-oriented devices that work with other instruments.

Examples include the container-style **Drum Machine** (for separate chains trigger by each note pitch that comes in).

› *Dynamics.* Processors that operate on their incoming audio signals, based off of those signals' amplitude levels and trends.

Examples include **Compressor**, **Gate**, **Peak Limiter**, and **Transient Control**.

› *EQ.* Sets of frequency-specific processors that operate on their incoming audio signals.

Examples include various configurations of equalizers (such as **EQ+** and **EQ-DJ**).

› *Filter.* Frequency-specific processors that operate on their incoming audio signals.

Examples include **Filter+** (for combining one of ten filter modules with any of 14 waveshapers), his performance-friendly companion **Sweep** (with two filter slots and clever macro controls), a layered **Resonator Bank**, and an endlessly configurable **Vocoder**.

› *Hardware.* Interface objects for sending signals and/or messages to devices beyond Bitwig Studio (such as hardware synthesizers and effect units, etc.). This can include transmitting and/or receiving audio signals, control voltage (CV) signals, and clock messages.

Examples include **HW Clock Out**, **HW CV Instrument**, and **HW FX**.

› *Hi-hat.* Hi-hat drum element instruments that use incoming note signals to synthesize audio.

Examples include the electronic drum emulator, **E-Hat**.

› *Kick.* Kick drum element instruments that use incoming note signals to synthesize audio.

Examples include the electronic drum emulator, **E-Kick**.

› *MIDI.* Transmitters for sending various MIDI messages via the track's device chain. This is useful for sending messages to plug-ins or to external hardware (when used in conjunction with Bitwig's *hardware* devices).

Examples include **MIDI CC**, **MIDI Program Change**, and **MIDI Song Select**.

› *Modulation.* Processors that manipulate incoming audio signals with an LFO, etc., influencing their function.

Examples include the high-level **Chorus+**, **Flanger**, and **Phaser+**-type processors, as well as moving **Rotary** and **Tremolo** effects.

› *Note FX.* Devices that manipulate incoming note signals before passing them onward.

Examples include **Arpeggiator** (for animating held notes), **Multi-note** (for using single notes to trigger multiple notes), and **Note Repeats** (for repeating held notes at a timing interval, with optional chance, accents, Euclidean rhythmic pattern, and more).

› *Organ.* Yes, organ emulators that use incoming note signals to synthesize audio.

Examples include the drawbar-based **Organ**.

› *Percussion.* Percussion instruments that use incoming note signals to synthesize audio.

Examples include the electronic drum emulator, **E-Cowbell**.

› *Reverb.* Time-based processors that operate on their incoming audio signals.

Examples include the eponymous **Reverb** device and the open-ended **Convolution**.

› *Routing.* Devices that divert a track's signal path, allowing signals to exit and/or reenter the track.

Examples include **Audio Receiver** (for bringing in audio signal from other track or input) and **Note Receiver** (which does the same for note signals).

› *Snare.* Snare drum element instruments that use incoming note signals to synthesize audio.

Examples include the electronic drum emulator, **E-Snare**.

› *Spectral.* Devices that operate in the frequency domain, working with hundreds of individual frequency bands.

Examples include **Transient Split** (for separating the percussive, noisy parts of a sound from the tones [see section 19.22.4]), **Loud Split** (for adjusting the quiet, mid, and loud parts of any individual moment [see section 19.22.3]), and **Harmonic Split** (for taking odd harmonics, even harmonics, and non-harmonics into three different signal paths [see section 19.22.2]).

› *Synth.* Synthesizer instruments that either generate their audio from rudimentary source material or use audio samples. Incoming note signals are used to synthesize audio.

Examples include **Polysynth**, **FM-4**, and **Sampler**.

› *The Grid.* Devices utilizing **The Grid**, Bitwig's modular sound-design environment (see chapter 17).

Examples include **FX Grid** (for building audio effects, etc.), **Note Grid** (for creating note processors or even note generators) and **Poly Grid**.

› *Tom.* Tom drum element instruments that use incoming note signals to synthesize audio.

Examples include the electronic drum emulator, **E-Tom**.

› *Utility.* An assortment of devices sporting various generating, processing, and time-shifting functionality.

Examples include signal generators (such as **Test Tone**), processors (such as **Tool**), and the unique **Time Shift** device, for moving audio and notes signals either later or (relatively) earlier in time.

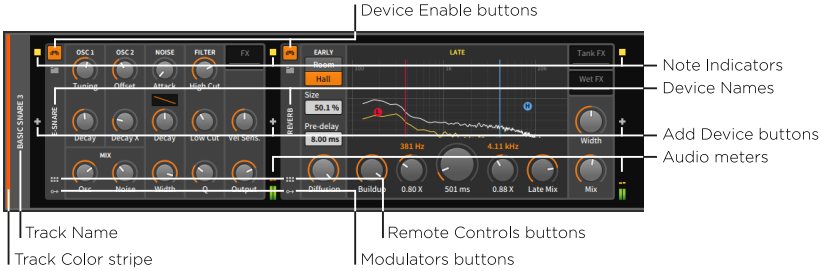
So while devices aren't always necessary, they can make things a whole lot more interesting and open up possibilities that you may not have previously thought of.

8.1. The Device Panel

As we've seen with the browsers in Bitwig Studio (see chapter 4), presets and devices can be found and searched in several ways. Whether we load our devices from other panels or not, the **Device Panel** is where all direct interaction with devices will occur. So once we are ready to work with devices, we must explore the **Device Panel** and see what it has to offer.

8.1.1. The Panel Itself

Let's take a simple example of a track that contains two devices: one instrument and one audio FX.



Note that the above image shows the instrument on the left and the audio FX on the right. In the **Device Panel**, signal always flows from left (input) to right (output). While you could swap the position of these devices, you probably would not get the desired outcome.

Starting with the outer rounded rectangle, we find on its left edge an abbreviated, vertical track header. Included here are the familiar *track color stripe* and *track name*.

Other than the group (including project) and track headers, all space in the **Device Panel** is reserved for devices. But before the first device (and after every device) comes a vertical column containing three items:

› The *note indicators* light up when at least one note signal is active at that stage. (This is similar to a MIDI "note on" message that has not yet been followed by a corresponding "note off.")

› The *Add Device button* calls up the **Pop-up Browser** window.

› The *audio meters* indicate the presence and level of audio signal being received and transmitted by each device.

The Add Device button is present in all these locations so that you can insert additional devices at any point within the device chain. The note indicators and audio meters are present at every device handoff to visually inform you of signals that are changing as the signal flow progresses. As relevant texts and your own experimentation will teach you, the order in which devices are connected is critical to the outcome.

Each device has its own vertical header on its left edge. Common elements in device headers are:

› *Device Enable button*: Toggles the device between on (engaged) or bypass mode (temporarily disabled).

› *Device Name*: The official name of the device, or a substitute name that you have selected (see section 16.2.4).

› *Remote Controls button*: Toggles to reveal the Remote Controls pane for the device (see section 15.1.1).

› *Modulators button*: Toggles to reveal the Modulators pane for this device.

Finally, the body of each device contains its own various parameters. They can take the form of knobs, sliders, numerics, text and graphical lists, buttons, curve controls, clickable graphic interfaces, and more. All parameters can be set with the mouse by simply clicking and dragging.

8.1.2. Player Mode

Some instruments and other devices are limited to specific versions of Bitwig Studio. But even so, your license may include presets that make use of these devices. In this case, the preset will be loaded in a *player mode*, with the preset's remote controls available as the way to manipulate the sound.

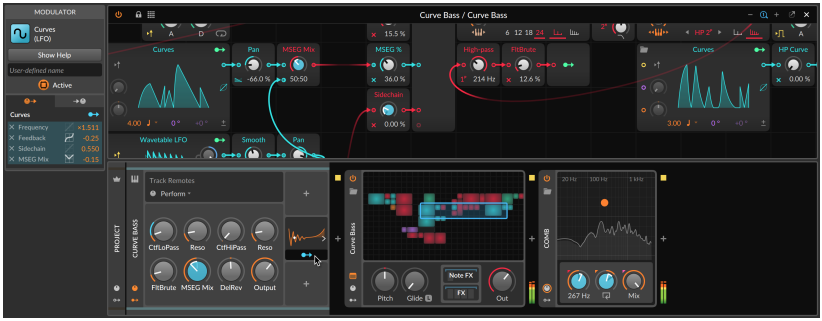
As with this **Poly Grid** patch from Bitwig's *Essentials* package, notes can be sequenced, remote control automation draw and edited, audio bounced, etc.

Additionally, this makes it possible to collaborate with any Bitwig Studio user because project files work in a similar way — giving you can access to remotes and free sequence editing for devices that are not part of

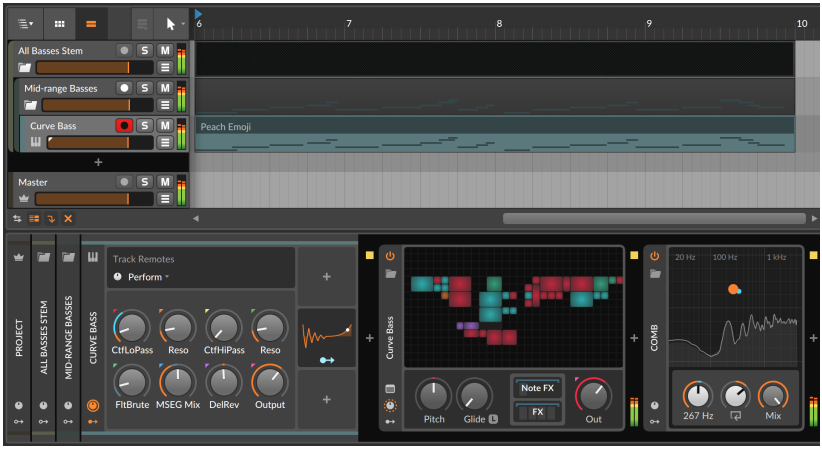
your license. You can then do work, save your changes, and send the project back and forth.

8.1.3. Track Headers in the Device Panel

The **Device Panel** also contains headers for each level from this track up to the master track. This will usually just include one project header and one for the track, allowing remote controls and modulators to be added at either level.



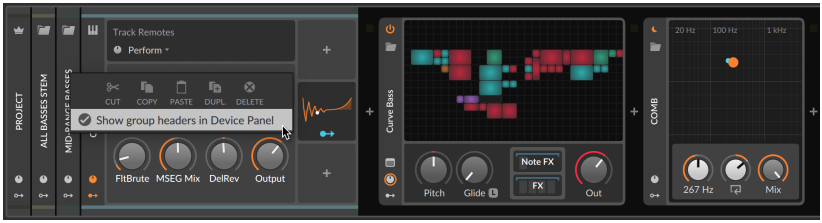
If group tracks are involved, you will also get a header for each level within the hierarchy.



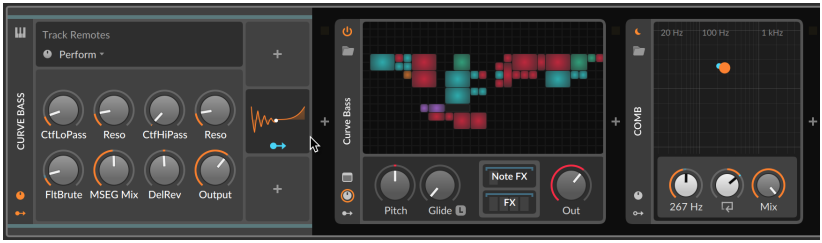
*To show the **Inspector Panel** for any track in the **Device Panel**:* select that track's device header. Since the track inspector includes the track's

meter and mixer controls, this can be a very helpful short for viewing the master output level, or focusing on any other track.

*To show/hide the higher-level track headers from the **Device Panel**:* right-click on any device header to expose the *Show group headers in Device Panel* option.



When disabled, every **Device Panel** will simply start at the local track level.



8.1.4. The Expanded Device View

Certain devices have an optional *Expanded Device View*. This list currently includes several instruments (**FM-4**, **Phase-4**, **Polysynth**, and **Sampler**) and some audio effects or analyzers (**EQ+**, **EQ-5**, **Resonator Bank**, as well as **Oscilloscope** and **Spectrum Analyzer**), and all devices that give access to *The Grid* (**Poly Grid**, **FX Grid**, and **Note Grid**), as well as devices that are Grid-powered (including the **Polymer** synth, and the audio FXs **Filter+** and **Sweep**). Each of these devices has an *Expanded Device View button* in its device header.

Clicking the **Expanded Device View** button covers the central panel area with additional controls and visualizations for the device.

The **Expanded Device View** can also be loaded as a separate floating window by clicking the undocking button (the box with an arrow ascending out of it) in the top right.

Once floating, the **Expanded Device View** will remain visible regardless of what track is selected. You can always close the window or click the re-docking button (the box with an arrow, propelling the window back to Earth) to rejoin the view within the main window.

Note

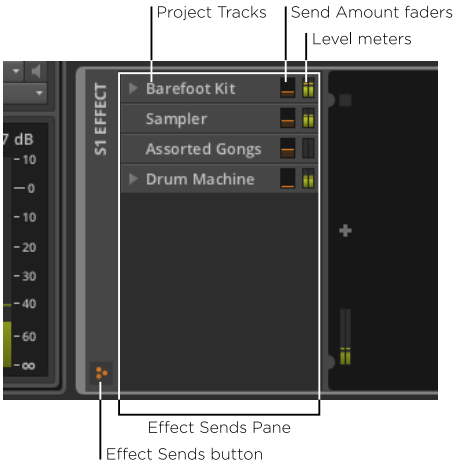
This behavior can be changed with *Floating windows follow current track* setting. This preference can be found in the **Dashboard** under the *Settings* tab on the *Behavior* page in the *Device* category. Enabling this setting will hide any floating **Expanded Device View** windows when a different track is selected and restore them when you select their track again.

Additionally, this setting provides a thumbtack toggle at the top right of the window, allowing you to make some floating windows persistent while the others only appear when their track is selected.

These views can also be accessed via mixer and **Inspector Panel** interfaces in Bitwig (see section 7.1.5).

8.1.5. FX Tracks and Send Amounts

FX tracks have one unique feature in the track header of the **Device Panel**.



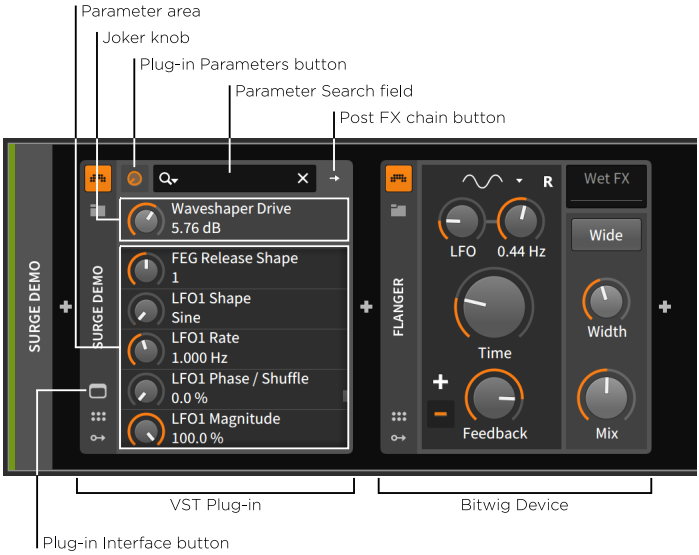
When the *effects sends button* is enabled, the *effect sends pane* is visible within the track header area. This resizeable pane shows a list of all instrument, audio, hybrid, and nested group tracks in your current project. Each track is listed along with a meter showing its current output level and a control for the send amount targeting this FX track.

Essentially, this is a "mixer" view of the buss that feeds the FX track. And tracks that have track fold buttons on the mixer (see section 7.1.1) have a similar fold button here.

Additionally, clicking on the name of any send toggles whether it is active or not, saving processing when unneeded or allowing for a 'bypass' while keeping your level setting.

8.2. Plug-ins

The other kind of devices that can be used in the **Device Panel** are plug-ins, such as VST or CLAP plug-ins. Aside from setting up Bitwig Studio to recognize the plug-ins you own (see section 0.2.2.5 for information on the *Locations* page of the **Dashboard**), we haven't talked much about them. They operate side by side with Bitwig devices, and both generally function in the same way, but the interface for plug-ins is a bit different.



The bulk of the panel is reserved for the plug-in's *Parameter area*, but the parameters are in the form of a long scrollable list of knobs. And above this list is a single *joker knob*, which is really an alias (or wild card) which follows the last plug-in parameter that you touched. So after you scroll halfway down a very long parameter list, the last parameter you adjusted will still be accessible just above the list.

The top row of most plug-in devices has three important controls:

› The *Plug-in Parameters button* (with a knob icon) is lit up whenever the joker knob and list of parameters are being shown below.

› The next button varies depending on the type of plug-in your have loaded:

Most plug-ins then have a *Post FX chain button* (with a single right-facing arrow for an icon), as was shown in the above image. Clicking

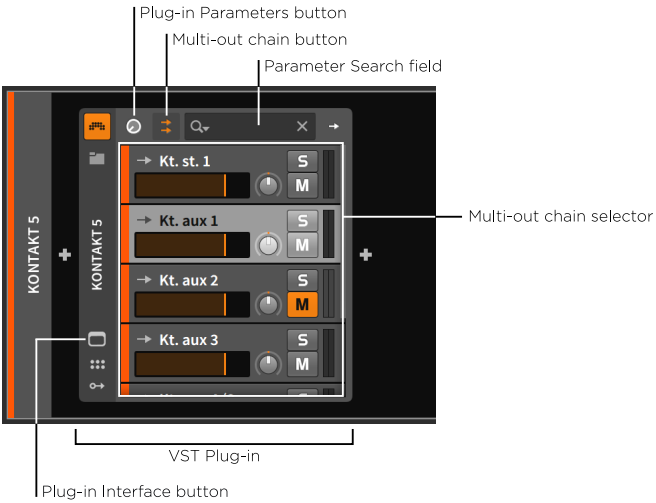
this button expands the right edge of the plug-in interface to display a chain where other devices and plug-ins can be loaded.

The advantage of loading devices within a plug-in's Post FX chain is that when you store a preset for this plug-in, that preset will include all attached devices as well as their settings. So in the example above, saving a preset for **Surge** would include the **Chorus** device and the **MasterVerb 5** plug-in along with all of their current settings, but the **Blur** device would not be included.

Note

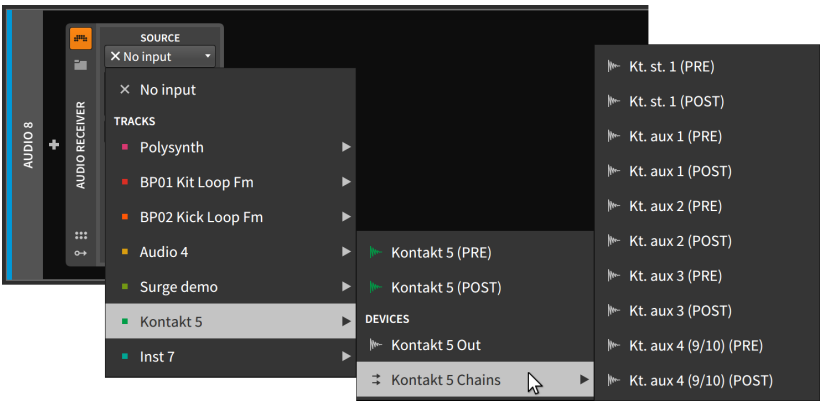
For more information on nested device chains, see section 16.1. And for specific information on Post FX chains, see section 16.1.3.

Multichannel plug-ins do not have a Post FX chain button and corresponding chain. Instead, they have a *Multi-out chain button* (with two right-facing arrows for an icon). Clicking this button toggles the parameter area below to instead show the *Multi-out chain selector*.



This chain mixer gives you mixing controls for all the various outputs of this multichannel plug-in within the current stereo track. Clicking on the plug-in parameters button will return the parameter area to its normal joker knob and parameter list.

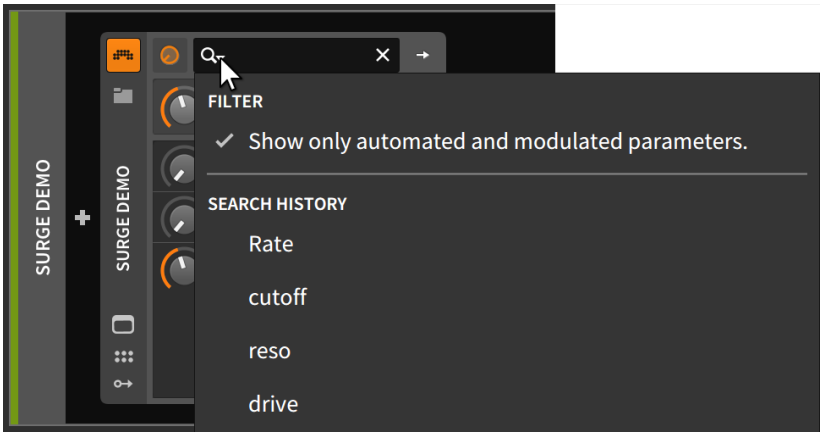
To access audio channels from a multichannel plug-in on a different track: either from a track's audio input chooser or from an **Audio Receiver** device's *SOURCE* menu, select the track of the multichannel plug-in, then select its *Chains* submenu, and finally select the desired audio source.



› The *Parameter Search field* is provided in the top of the plug-in area to let you filter the parameter list and find what you are looking for.

This is useful as the parameter list for a complex plug-in can be exceedingly long.

The parameter search field's magnifying glass icon also doubles as a menu. By clicking on this icon, you can thin the parameter list to *Show only automated and modulated parameters*. You can also revisit your recent *SEARCH HISTORY* from this menu.



If remote controls are configured for your plug-in (see section 15.1.1), an active controller may show parameter mappings here using small colored circles in both panes.

Finally, in the device header for any plug-in is a *Plug-in Interface button*. Clicking this button calls up the plug-in's custom interface in a floating window.

(As all plug-ins have their own custom interface, please don't expect anything else to look like **Surge**, shown as the example above.)

8.3. Working with Devices

Earlier in this chapter, we covered both adding devices and loading presets. Before moving on, here is a list of other basic functions you may wish to execute with the **Device Panel**.

To minimize/restore a device's interface: double-click the device header.

This is a change in appearance only and does not affect the operation of any device.

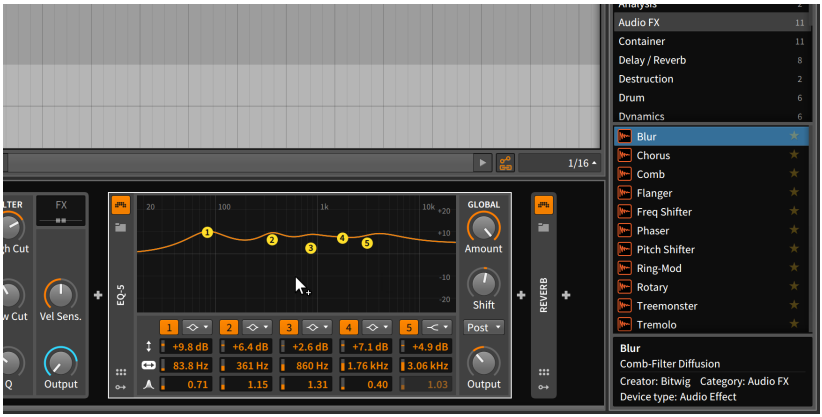
To select a device: single-click its track header.

The currently selected device is indicated with a dashed white border. Once selected, all regular *Edit* functions apply, such as cut, copy, duplicate, and delete.

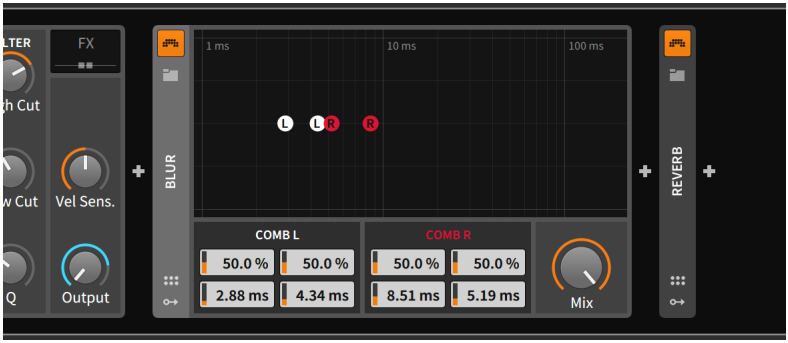
To move devices around: click and drag the device header to the desired position within the **Device Panel**.

As the status message in the footer suggests, [CTRL] ([ALT] on Mac) can be added to toggle the move to a copy function.

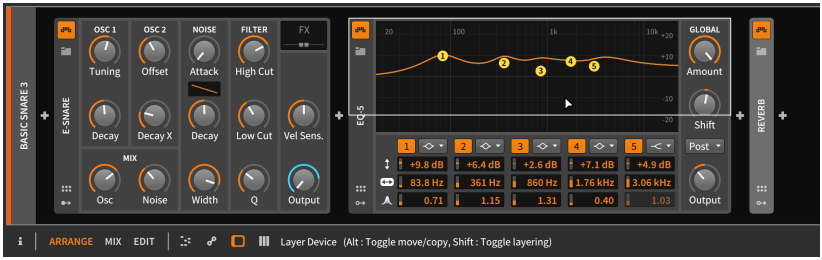
To replace one device with another: drag the desired device or preset from the **Browser Panel** onto the device to be replaced.



Once the mouse is released, the device will be replaced.



To layer a device with another: [SHIFT]-click and drag the device over top of the device where the layer should be inserted.



Depending on the type of devices being layered, an appropriate container device will be created and populated with your selections.

Note

For more information on container devices and other advanced device concepts, see chapter 16.

To rename a device: select the device and then change its name from the **Inspector Panel** (see section 16.2.4).

9. Automation

With the mixer interface (chapter 7) and our introduction to devices (chapter 8), we examined both track and device parameters that you will want to set as your own tastes dictate. But fixing these parameters to certain values is probably not enough.

If you can think about how a song develops — from the arrangement growing as parts gradually fade in and find their place in the stereo field, to instruments becoming more animated as their tones morph and brighten, to parts gradually fading away by both losing volume and increasing reverb — then you can visualize the series of long and short curves that represents a piece of music and its structure.

Automation is the animation of any defined parameter over time. It is usually thought of as narrative and rigid (in the same way the Arranger Timeline defines a particular musical progression), but Bitwig Studio also supports both a clip-oriented approach to automation, and techniques for having multiple layers of control cooperate to shape individual parameters in a relative way.

We will start our look at modulation back in the **Arranger Timeline Panel**, where we can work directly with traditional track-based automation. Then we will meet the **Automation Editor Panel**, whose sole purpose is displaying and manipulating automation. Finally, we will see how clip-based and relative automation can enhance our workflows and music in ways both novel and powerful.

Let's get those parameters dancing.

9.1. Automation Basics

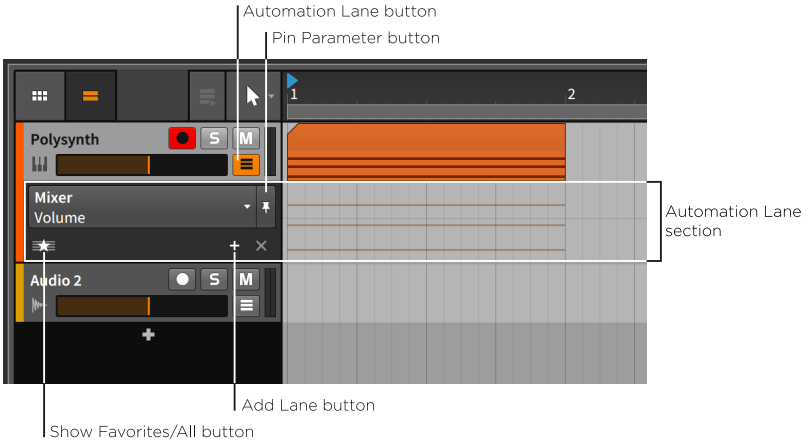
If you work with music software and are used to only one type of automation, it is *track automation*. With this kind of automation, values for a parameter — volume, cutoff frequency, reverb amount, etc. — are stored as fixed values. So when the playback head reaches an automation point of either *-9.43 dB* or *2.88 kHz* or *124 %*, that exact value will be set and preserved until the automation dictates otherwise.

Bitwig Studio can accommodate this kind of automation, and it can be achieved with our old friend, the **Arranger Timeline Panel**.

9.1.1. The Arranger's Automation Lane Section

The one item in the Arranger that we have not looked into yet is the *Automation Lane button* within each track header. When a track has this

button enabled, the *Automation Lane section* for that track becomes visible.



The Automation Lane section appears just below the track header and extends across the Arranger Timeline area as a place to show its own time-based data. Like all automation lanes, this one is resizable.

This track header section contains the following controls:

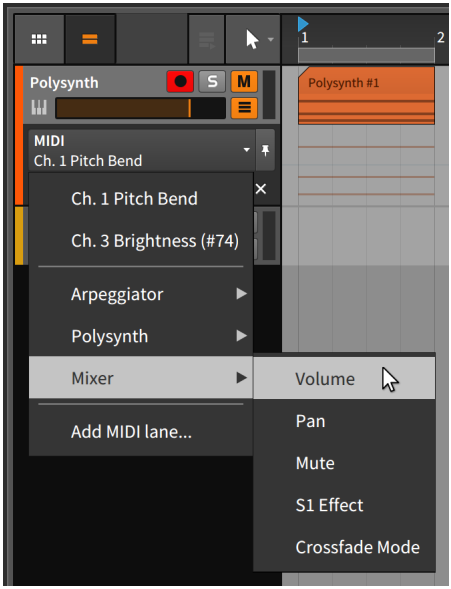
› *Parameter chooser*: Indicates and selects which parameter is displayed in this primary lane.

› *Pin Parameter button*: Maintains this lane's focus on the current parameter. This is disabled by default, which causes focus to follow the last clicked parameter.

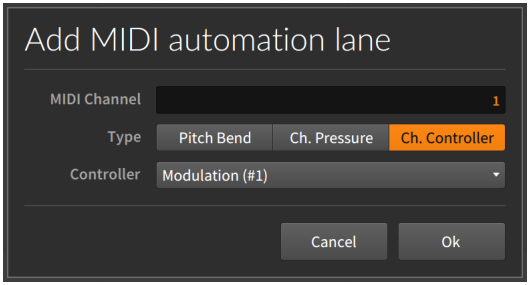
› *Add Lane button*: Creates an additional automation lane that is fixed on the currently selected parameter.

› *Show Favorites/All button*: Toggles between displaying additional lanes for either your favorite parameters or for all parameters that are automated.

By clicking on the Parameter chooser, we will see a list of all automation targets for the selected track.



The list is displayed in signal-flow order, starting with any MIDI automation lanes that are present. Listed next in order are all devices directly on the track's device chain. (Nested devices appear within their parent device's menu.) After that are *Mixer* elements, including track *Volume* and other parameters shown above. The last item, *Add MIDI lane...*, calls up a window.

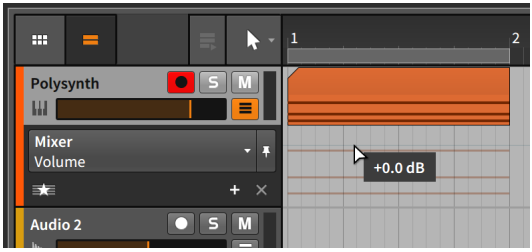


In order to add a MIDI automation lane, you have to set the *MIDI Channel* and the *Type* of message for this lane. Message types include *Pitch Bend*, *Ch. Pressure* (sometimes called *aftertouch*), and *Control Change* (which also requires a *Controller Number*).

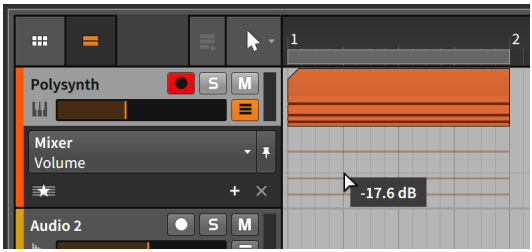
The background of the Automation Lane in the Arranger Timeline faintly hints at notes or audio events on the current track. These cannot be

selected or edited; they are just illustrations to help you define your automation in relation to the track's contents.

This area is where our automation functions will be defined. And while this lane might seem empty, one subtle datum is present.



As the picture above shows, there is a light gray line just above the note outlines. This is the current automation curve of the track's volume. And since there are no additional automation points, that curve is a flat line at the current setting of *+0.00 dB*. If we were to grab the volume fader in the track header and make it quieter (by dragging it to the left), the gray line would follow.

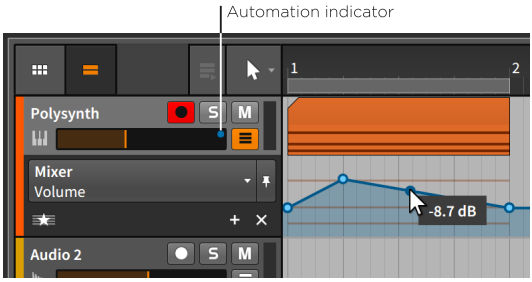


9.1.2. Drawing and Editing Automation

We will start with manipulating single points of automation. Similar processes will also work when multiple values are selected.

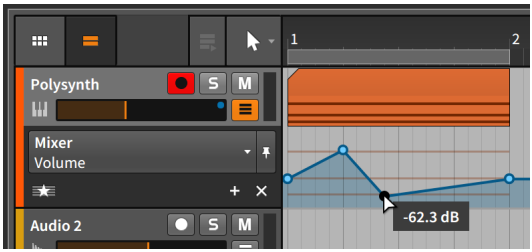
To create a single point along the automation curve: click in an area along the curve, and then drag the point to the desired value and position. Or single-click anywhere within the automation lane with the Knife tool.

We can repeat this a few times to create a small shape.

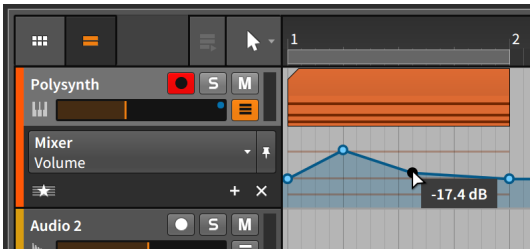


Note that dragging your mouse along the automation curve displays the parameter value beside your cursor for that song position. Also note the blue circle that has appeared near the top of the volume fader's range. This *automation indicator* — which looks like a misplaced automation point — indicates that the parameter in question is under the control of automation.

To create a single point outside the automation curve: double-click any area of the automation lane.



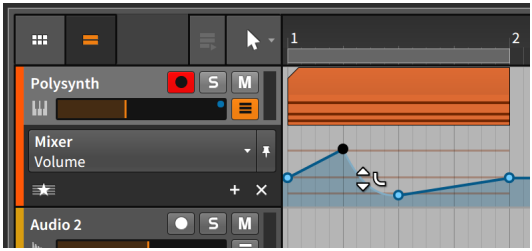
To move an automation point: click and drag the point with the mouse.



Note

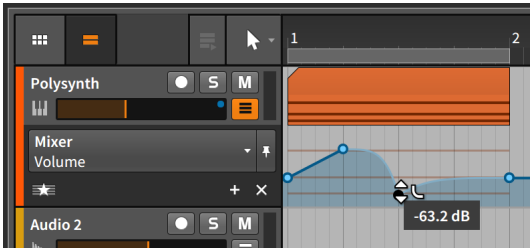
The absolute grid setting constrains the movement of automation points. To temporarily toggle this setting off, hold [SHIFT] while placing points.

To adjust the transition between two automation points: [ALT]-click and drag the curve between two points.



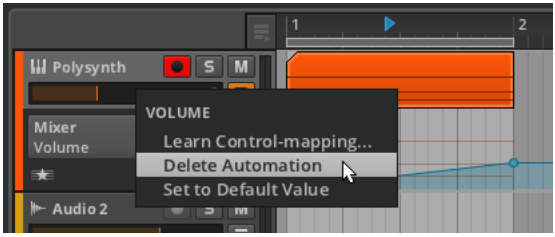
To reset a transition (to linear interpolation): [ALT]-double-click the transition.

To shape both transitions around an automation point: [ALT]-click and drag the point.

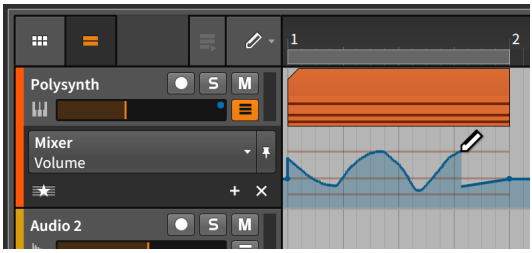


To delete an automation point: double-click it. Or single-click the point to select it, and then press [DELETE] or [BACKSPACE].

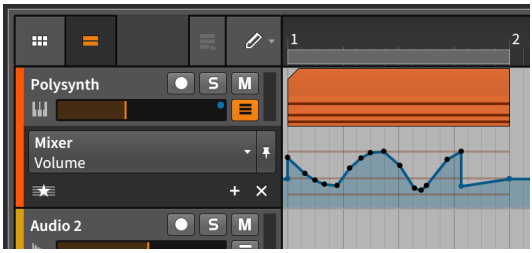
To delete all automation for a parameter: right-click on the parameter and select *Delete Automation* from the parameter's context menu.



To redraw an automation curve: click and drag horizontally with the Pen tool.

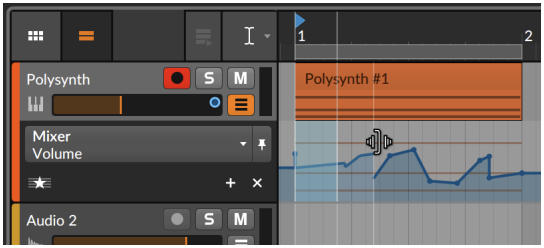


Once you release the mouse, the curve will be optimized to maintain its shape with the minimum number of points.



To select multiple points: either click and drag a selection rectangle around the points of interest, or switch to the Time Selection tool and click and drag horizontally.

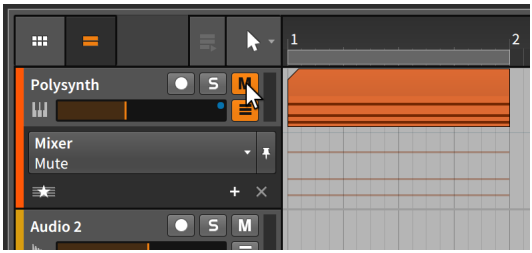
To time scale a range of automation points: first make a time selection (with the Time Selection tool), and then [ALT]-drag on the left or right boundary of the selection.



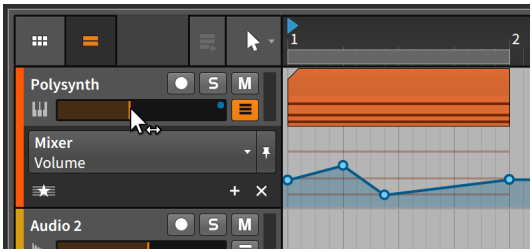
9.1.3. Parameter Follow and Automation Control

While you could use the Parameter chooser every time you need to find a parameter, the chooser can help you. Its default behavior is to focus on whichever parameter you select with the mouse. We call this initial automation lane the *joker lane* because like a wild card, it takes on whatever function you want it to.

For example, clicking the track's mute button will now focus on the primary lane for that parameter.

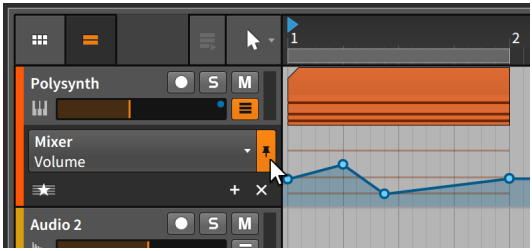


If you then click on the track's volume fader, focus will return to the volume parameter.



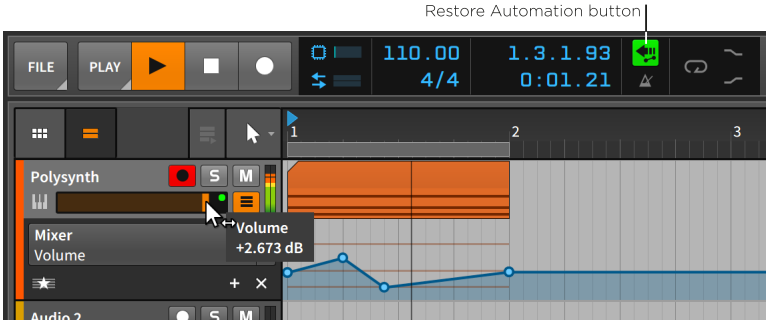
As you can see, the automation that was drawn a minute ago has not been lost. This primary lane is simply shifting its focus with each mouse click.

To lock the Parameter chooser to its current selection: enable the Pin Parameter button.



In the example shown, the Parameter chooser will now stay focused on the *Volume* parameter even if you click on the track mute button or any other parameter.

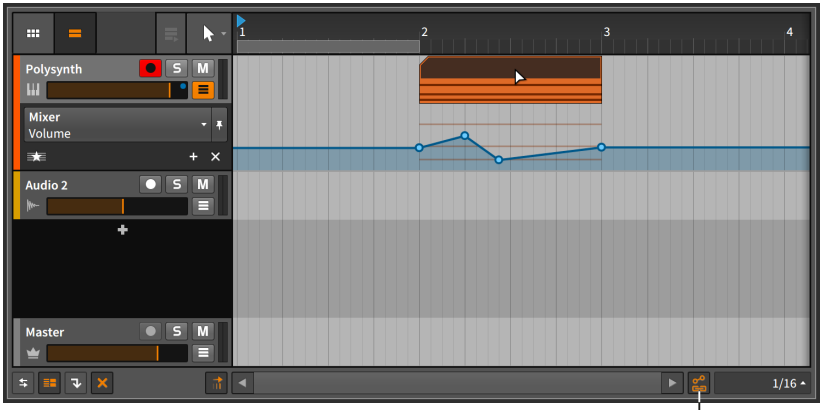
Additionally, Bitwig Studio will let you temporarily override the automation values you have set. This will automatically occur whenever you grab an automated parameter and adjust it.



The automation indicator for the volume parameter has switched from blue to green, indicating that the automation's control of this parameter has been broken for the time being. At the same time, the *Restore Automation Control button* within the display section of the window header is now tinted green, indicating that it is armed.

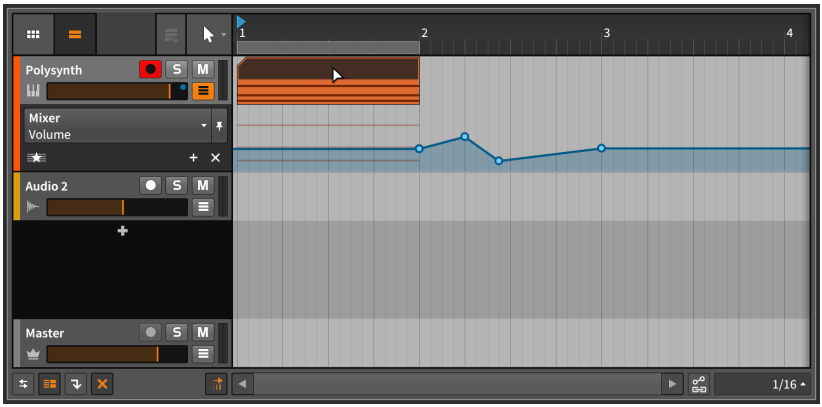
To restore the control of automation over all parameters: click the Restore Automation Control button.

The *Automation Follow button*, beside the beat grid settings in the bottom right corner of the panel, is worth mentioning here. This button toggles whether track automation is moved in tandem with Arranger clips or not. The setting is enabled by default so moving a clip would have the following effect.



Automation Follow button

Disabling the button and moving the clip back would leave any and all track automation behind.

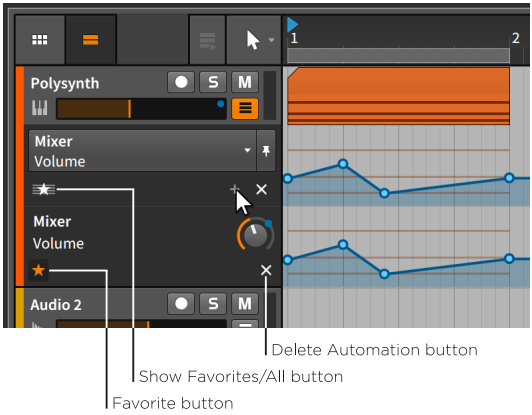


The would hold true for movement functions, such as copy, duplicate, etc.

9.1.4. Additional Automation Lanes

At times it will be useful to view the automation curves of several parameters at once. To achieve this, Bitwig Studio supports fixed automation lanes that appear beneath the dynamic primary lane.

To create a fixed automation lane for a parameter: select the desired parameter in the chooser, and then click the *Add Lane button*.



While it looks as though the lane just duplicated itself, there are some key differences here.

Only the top lane has a parameter chooser. The new lane — and any subsequent lanes — only has a text label indicating the device and parameter being automated so it cannot change focus.

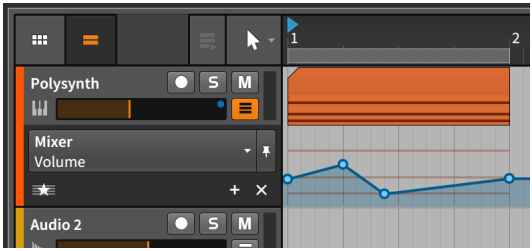
You will also notice that the new lane has two slightly different interface buttons beneath.

› *Favorite button*: Marks the parameter to be displayed in the favorites list.

› *Delete Automation button*: Deletes all automation for the lane's parameter and removes the lane.

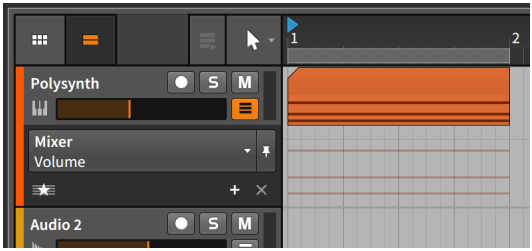
As the *Show Favorites/All button* above is indicating with its star icon, tracks default to displaying favorite parameters. When favorites are being shown, clicking the Add Lane button both creates the fixed lane and automatically marks this parameter a favorite. The enabled Favorite button of our new lane demonstrates its status.

To remove a fixed lane's favorite status: disable the lane's Favorite button.

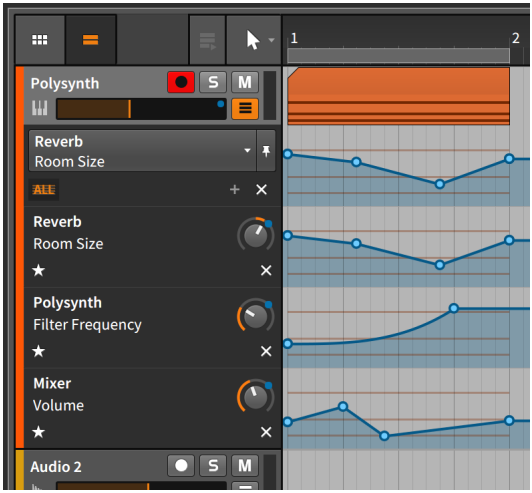


This essentially puts us back to where we started.

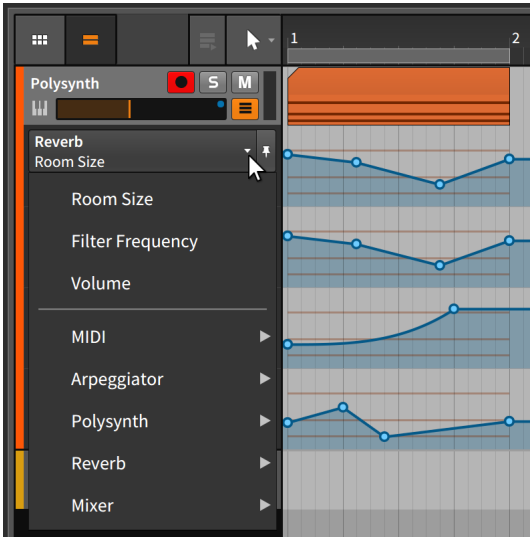
Please do not confuse the *Delete Automation button* for a "close" button. Clicking it instead of the Favorite button will collapse the additional lane, but it will also delete the automation for that parameter.



To display all parameters that have automation: toggle the Show Favorites/All button to the *All* setting and icon.

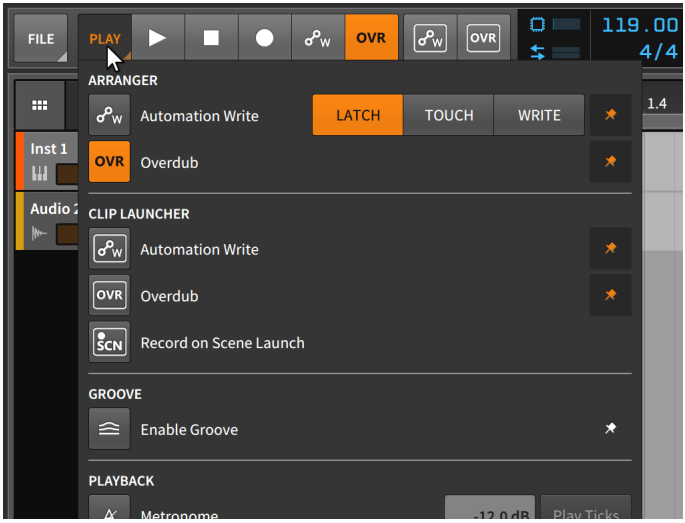


The list of automated parameters can also be accessed from the top of the Parameter chooser list.



9.1.5. Recording Automation

The automation write mode is set within the *Play* menu in the window header's transport section.



There are three modes for recording automation.

› *Latch* mode begins recording automation values as soon as you begin changing parameters. Recording then continues until the transport is stopped.

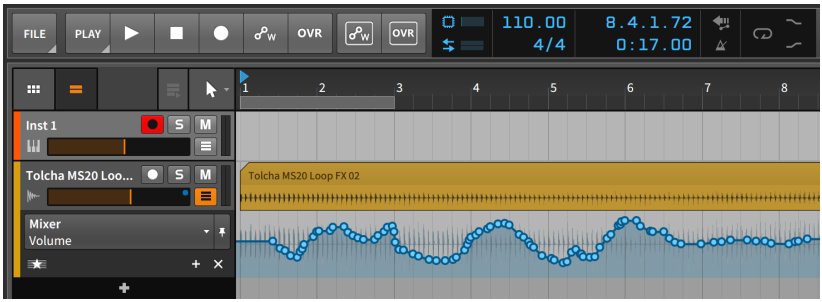
› *Touch* mode also waits until you have begun changing parameters to begin recording automation values, but once you stop interacting with a parameter, recording is halted and any preexisting values are preserved.

› *Write* mode is the most destructive, recording automation values from the time the transport is launched until it is stopped. Any preexisting automation points that are passed will be overwritten.

Automation recording is separately included in both the **Arranger Timeline Panel** and the **Clip Launcher Panel**.

To record automation in the Arranger Timeline: enable the Automation Record button in the transport controls section of the window header, and then start the transport.

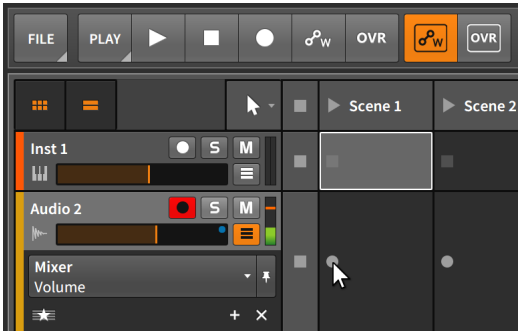
Whether the transport is playing or recording, any parameter adjustments made on this track will be printed as automation. Once the transport is stopped, the automation curve will be optimized and the Automation Record button will be disabled.

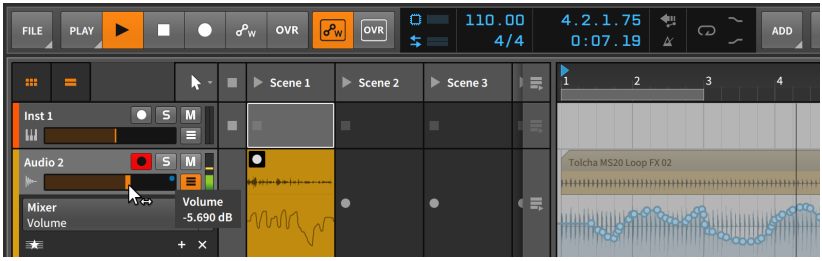


Note

In the **Dashboard** under the *Settings* tab on the *Recording* page in the *Recording* section is an option called *Write Automation on Record*. If this option is enabled, the Arranger's Automation Record button will automatically be enabled whenever the Global Record button is armed.

To record automation in the Clip Launcher Panel: enable the track's record arm button and the Launcher's Automation Record button, and then begin recording a clip.





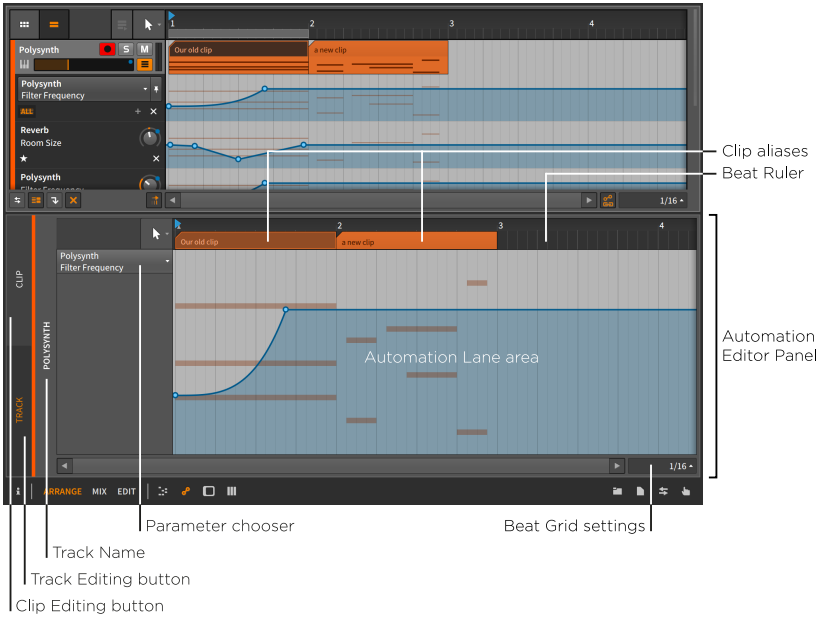
If the Automation Lane button is enabled for the track, the clip's automation will be displayed in the bottom of the clip.

9.2. The Automation Editor Panel

Each panel in Bitwig Studio is focused as narrowly as possible on a specific function. The **Arranger Timeline Panel** is, by necessity, the broadest of our editors. While it also supports working with automation, that is not its primary purpose. Working with automation is, however, the only purpose of the **Automation Editor Panel**.

9.2.1. Track Editing Mode

When the **Automation Editor Panel** is initially called up within the **Arrange View** (by clicking the **Automation Editor Panel** button in the window footer), it opens in *track editing mode*.



In this mode, the interface should look quite familiar. Due to the presence of the *Beat Ruler* (see section 3.1.1), unique *beat grid settings* (see section 3.1.2), and unique *snapping settings* (see section 5.1.2), this looks a lot like the **Arranger Timeline Panel**. The difference is that the general purpose Arranger Timeline area has been replaced with the *Automation Lane area* for our currently selected track.

And the Automation Lane area is essentially an enlarged version of the primary automation lane we just saw in the **Arranger Timeline Panel**. This one also has a Parameter chooser on the left side, and the Automation Lane area is being used to display the automation curve of this parameter over a backdrop of the track's contents.

All of the automation drawing and editing functions we learned in the Automation Lane section of the **Arranger Timeline Panel** will work identically here. But there are a couple differences.

› The **Automation Editor Panel** contains only one automation lane. If you are looking to view multiple parameters from one track, the **Arranger Timeline Panel** is the way to go.

› The *clip aliases* (that float above the Automation Lane area in the Beat Ruler) are indicators of where the track's clips are starting and ending. But these aliases are also editable.

In the same way that Arranger clips can be moved (see section 5.1.2), edited (see section 5.1.3), and looped (see section 5.1.8), these same actions will work on the clip aliases. Just remember that the Automation Follow setting (see section 9.1.3) will determine how automation is affected by any clip movements or edits.

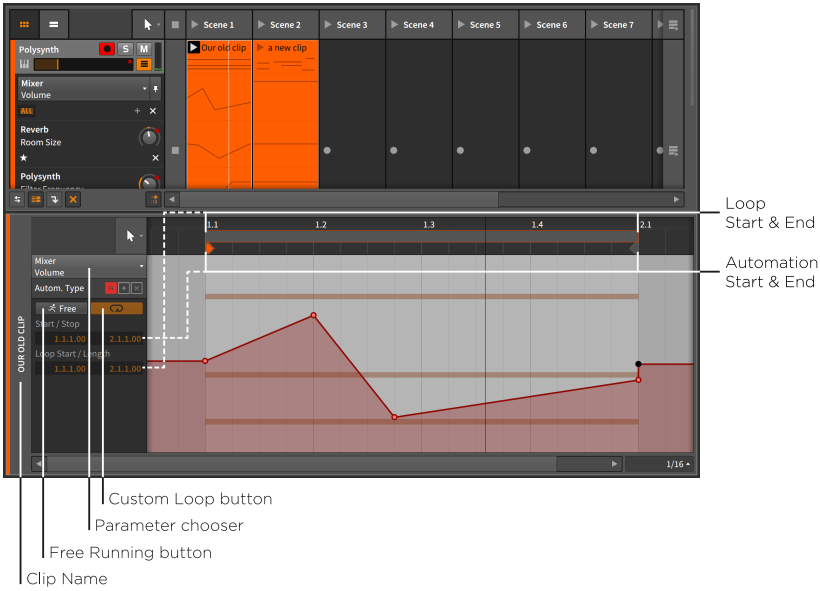
So this track editing mode of the **Automation Editor Panel** is a focused way to work with standard track-based automation. And for less standard, less track-based automation, there is the *Clip Editing button* in the top left of the panel.

9.2.2. Clip Editing Mode

At times it will be useful to have automation attached to a clip rather than to a track's timeline. This is ideal, for example, whenever you want the automation to repeat each time the clip does, or when you are working with the Clip Launcher.

When you want automation to be attached to an Arranger clip instead of the track's timeline, you can switch the **Automation Editor Panel** from track editing mode to *clip editing mode* by enabling the *Clip Editing button*.

When you are working with Launcher clips, all automation is done in clip editing mode with the **Automation Editor Panel**.



Once we break out of the track-based mindset, the same considerations come up as when we talked about the **Clip Launcher Panel** originally. Without the context of a track, our clips are essentially untethered to any fixed timebase or duration. And for this reason, clips viewed here generally use position *1.1.1.00* (often spoken as "bar 1, beat 1") as the relative start of the clip.

This is also where the Launcher's notion that clips should loop by default comes into play. In the *clip editing mode* of the **Automation Editor Panel**, we now get to decide if a clip's automation data should be tethered to its musical content or should play more freely.

The *Free Running button* contains an icon of a man running with the word *Free*. Once enabled, the clip's automation data can now be adjusted to play back differently from the clip's notes/audio. Once the Free Running button is enabled, the *Start* parameter below can now be adjusted, determining which part of the clip's automation will play back first.

Beside the Free Running button is the *Custom Loop button*. When enabled, this allows you to set different values for the automation's *Loop Start* and loop *Length* settings. When disabled, the clip's automation will loop just as the clip's musical content does.

These options can create some very dynamic situations, as in the example shown below.

Aside from the Free Running and Custom Loop buttons being enabled, the only change made was increasing the automation's loop *Length* from *1.0.0.00* (one bar) to *1.1.0.00* (one bar and one quarter). By making the automation loop repeat every five beats while the clip's notes repeat every four beats, the automation and notes will only line up in every fifth bar (every 20 beats).

Note

When any of these parameters are changed, you will need to retrigger the clip for the changes to be registered.

This example is just one way to create rich variation among a single clip's musical content and automation. With the options available, you are free to find your own preferred usage.

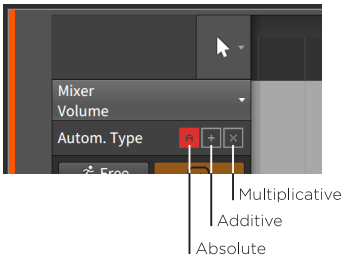
9.2.3. Relative Automation

All the work we have done so far involved *absolute automation*. In this paradigm, the automation values specified map to exact values in the parameter's units. A series of examples was already given at the top of this chapter: *-9.43 dB*, *2.88 kHz*, and *124 %*.

Bitwig Studio also has the capability to adjust most parameters in a relative way. With *relative automation*, you can move a parameter ±50%

of its total range (*additive automation*), or scale a parameter toward zero, anywhere from 100% of its current value to 0% (*multiplicative automation*).

When we started working in clip editing mode, three buttons appeared beside the *Autom. Type* label.



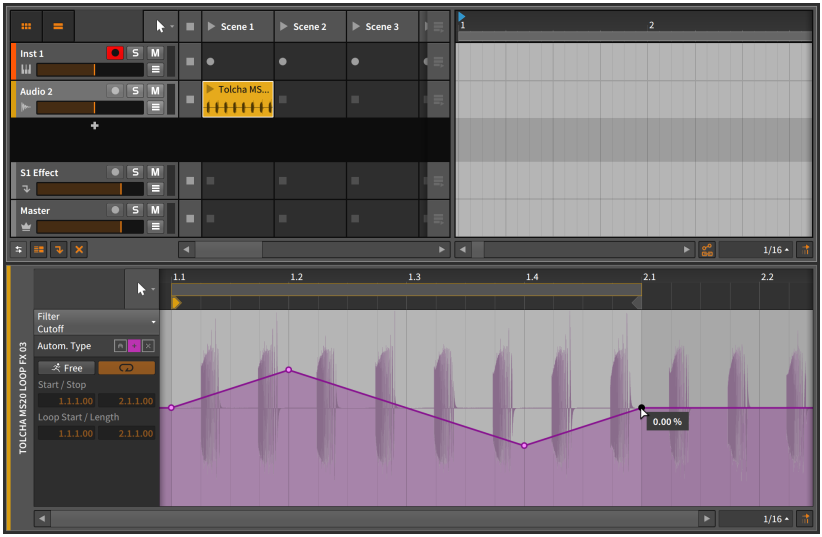
These three icons represent our automation mode choices of *absolute automation* (*A*), *additive automation* (*+*), and *multiplicative automation* (*x*).

When any of these icons are shaded in, this indicates the presence of that type of automation. So the image above is displaying that absolute automation was present for the selected parameter. An unshaded icon suggests that none of that automation type is present.

Note

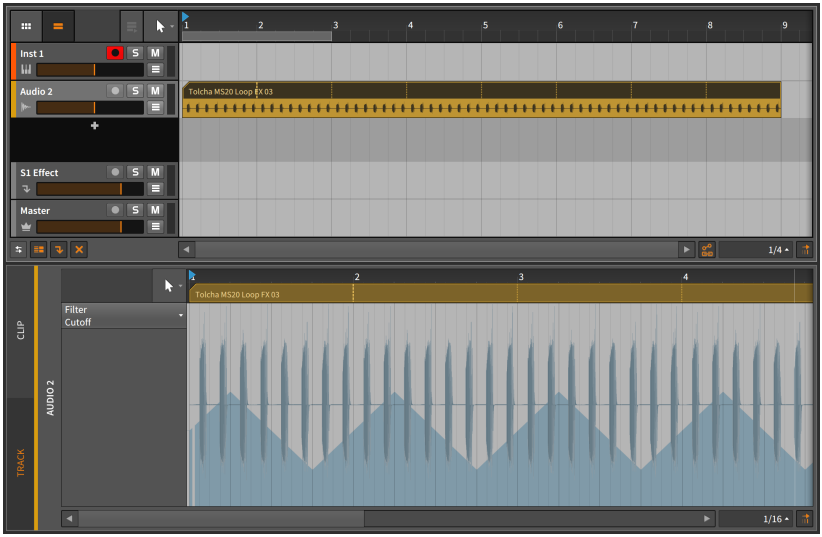
All forms of automation can be present for a single parameter. In this case, the absolute automation is applied first and then modulated by the additive automation. Multiplicative automation is applied last and has the final word, as multiplication always does.

For one example use, I will take a one-bar Launcher clip. I want its filter cutoff to move up a little, down a little, and then back to the middle in each bar. I can draw this with additive automation.

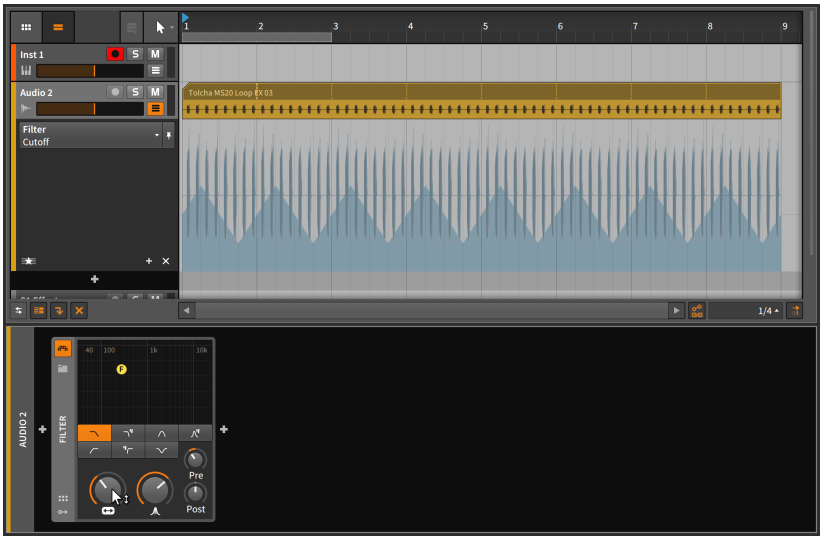


We can see that this automation is ending at *0.00%* so this additive automation is bipolar, moving up to about *20.0%* and dipping evenly to about *-20.0%*. We can also see that the additive modulation icon is the only one shaded so it is currently the only form of automation for this parameter.

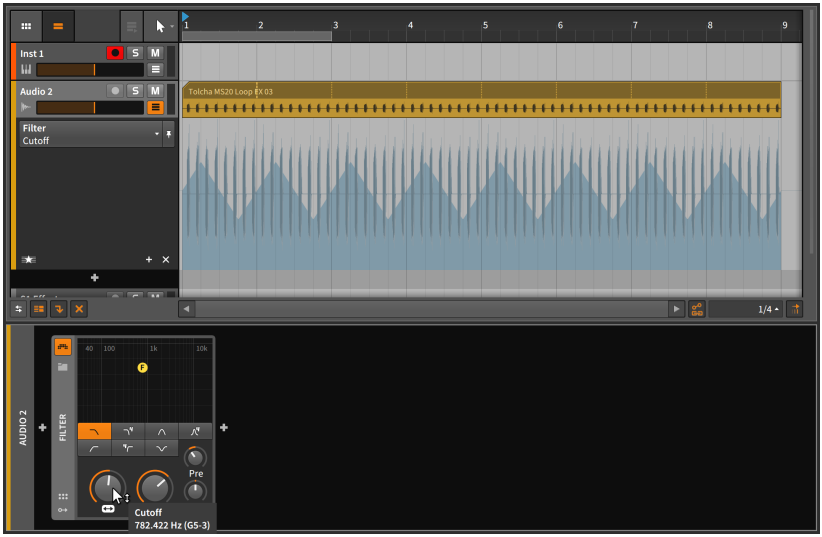
Next, I will drag this Launcher clip into the Arranger and loop it so that it lasts for eight bars.



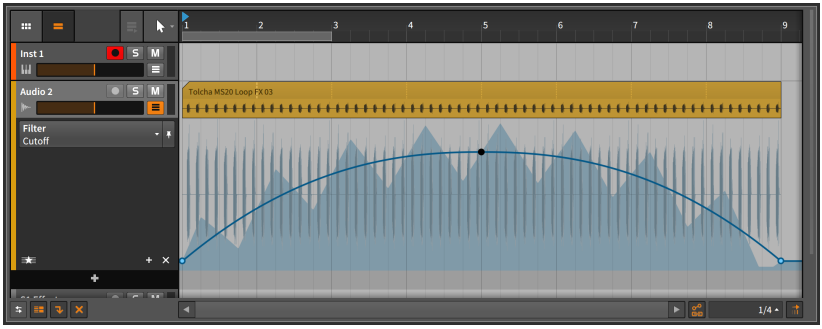
By viewing the absolute automation — the track automation here, since we are back in the Arranger — the automation curve has been extend for our eight bars, but it doesn't appear to be balanced around zero anymore. Let's look at both our automation and **Filter** device together.



We can see now that the default value of the *Cutoff* parameter is a good deal below the center of the range. Since the automation is relative, we can move the *Cutoff* knob to recenter where the automation lands.



I will leave you here with two ideas. The first idea is to now draw absolute automation over the course of these eight bars, taking the *Cutoff* from low to high and then back to low. I will do this by double-clicking to add three automation points, and then [ALT]-clicking and dragging the center point to reshape the curve.



The solid blue line represents the absolute automation curve. The shaded curve is showing the final parameter value, which is the result of both the absolute and relative automation together. By activating the transport, you would see the *Cutoff* control animated to match the

absolute automation curve, and the *Cutoff* knob's indicator ring would be moving to match the final parameter value.

The second idea is to not use absolute automation. Instead, use relative automation to give a sense of motion. And then during playback move the parameter control itself in realtime, perhaps with a MIDI controller (see chapter 15). This could be a very strong performance technique.

Note

Whenever a parameter's level indicator is moving separately from its control (as with the *Cutoff* knob and its indicator ring in the previous example), *modulation* is taking place. Relative automation is one form of modulation, and several others are discussed in section 16.2.

10. Working with Audio Events

We spent a healthy amount of time in the early chapters of this document talking about clips and their centrality to music production in Bitwig Studio. Even as the last few chapters have focused on other facilities of Bitwig Studio, clips are still a central part of the conversation. They are the vessels which hold our musical ideas, allowing us to manage, manipulate, copy, and vary these fragments into something greater.

And while we can call the clip our "musical atom," science tells us that atoms are made up of even smaller pieces and particles. In this chapter and the next, we will discuss the audio events and note events that clips are made of. (Whenever we refer to the "musical content" of clips, we are referring to the same audio events and notes.)

We have already examined the various capabilities for manipulating whole clips, whether they are Arranger clips (see section 5.1.10) or Launcher clips (see section 6.2.5). By using the **Detail Editor Panel**, we will begin working at the event level and seeing what tools are available to us at this deepest level of musical arrangement. And once we couple that interface with the **Inspector Panel**, most of the editing options and optimized workflows offered by Bitwig Studio will now be at our fingertips.

So let us begin the detail work of creating and preparing music. Next stop: *audio events*.

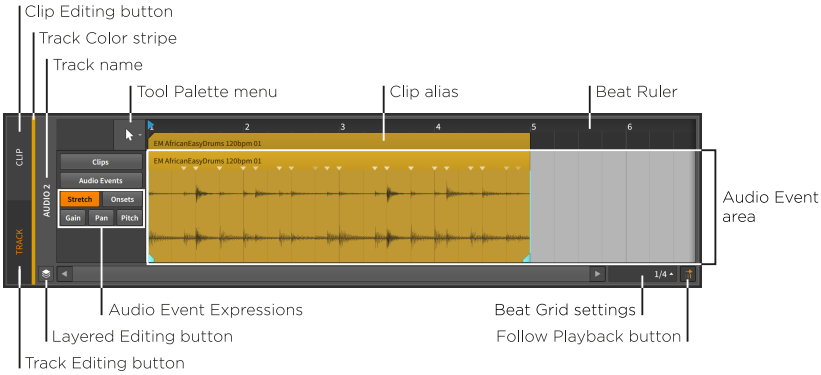
10.1. The Detail Editor Panel, Audio Clip Edition

All music is assembled in clips in Bitwig Studio. Just as a primary purpose of the **Automation Editor Panel** is to work with various kinds of clip automation, the purpose of the **Detail Editor Panel** is to work with the musical content of clips.

As you engage with the **Detail Editor Panel**, remember the subtle note earlier that every timeline-based panel has its own tool palette menu (see section 3.1.4). This allows each of these panels to have its own tool selection. This may seem like a small gain, but it really adds up. For example, if you find yourself making selections in the **Arranger Timeline Panel** and then going straight back to the **Detail Editor Panel** for making fine touches, you could be saving several mouse clicks (and a modicum of sanity) per edit.

10.1.1. Layout of the Detail Editor Panel

By double-clicking a clip, the **Detail Editor Panel** will be called up with its focus on that clip. For the examples in this chapter, we will use audio clips, and we will start by double-clicking an audio clip from the Arranger Timeline.



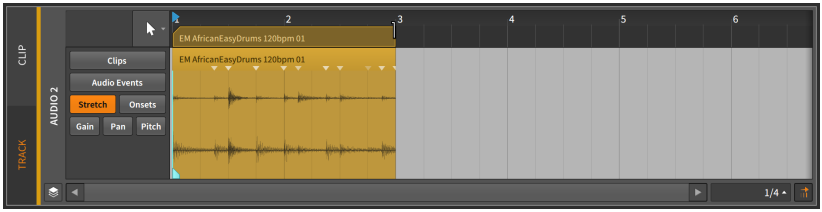
After working with the **Arranger Timeline Panel** and the **Automation Editor Panel**, many of these interface elements should be familiar, including the *Beat Ruler* (see section 3.1.1), and the *clip aliases* (see section 9.2.1), as well as this panel's own *beat grid settings* (see section 3.1.2), *snapping settings* (see section 5.1.2), and *Follow Playback button* (see section 3.1.4). Even the currently inactive *Clip Editing button* (see section 9.2.2) is here, indicating that we are starting in *track editing mode*.

But as with the previous timeline-based panels, the sections that have changed are substantial and unique to the operation of this panel.

The central *Audio Event area* is where all audio events are displayed in this panel. Audio events that appear here have their own headers, which can look redundant right below the clip's alias.

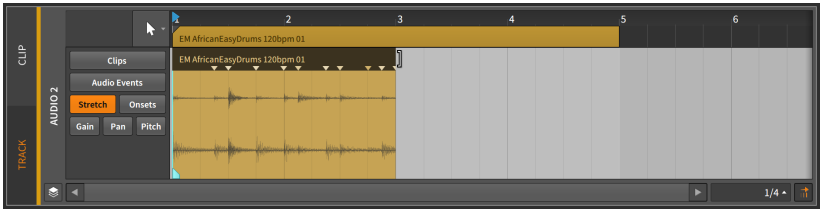
One example will illustrate the relationship between the clip and the contained audio event.

To adjust the length of a clip: mouse over the top right edge of the clip alias so that a half-bracket cursor appears. Then click and drag the mouse horizontally.



By shortening the clip, you can see that the audio event is also shortened. The clip is the parent in this relationship, and the children (audio events, in this case) can exist only where the parent is there to allow it.

To adjust the length of an event: mouse over the top right edge of the event so that a bracket cursor appears. Then click and drag the mouse horizontally.



By shortening the event, you can see that the clip itself is unaffected. You can do anything you want with this empty clip space: insert a short audio event/sample, duplicate as much of the previous event as will fit, or leave it blank as a rest. Nothing placed in the clip will be allowed to go beyond its boundaries, but all the available space can be used.

As you may also have noticed, no looping cursor appears when navigating the audio event's header. Clips are the smallest units where most arranging tasks are carried out. Accordingly, looping can be applied as an arrangement gesture for clips, but not for audio events (or notes). But fades can be applied to individual audio events, just as they were applied to audio clips (see section 5.1.7). And event stretching also works the same as it does for clips (see section 5.1.4).

10.1.2. Audio Event Expressions

To the left of the Audio Event area is a space for specifying which *audio event expression* is being displayed — and potentially edited. The images shown a moment ago displayed a menu in this area. But if you prefer a list of all available audio event expressions, drag the top border of the **Detail Editor Panel** so that it grows.

Audio event expressions — also called *expressions* — are parameters that can be set within each individual audio event. Several of these parameters can change over the course of the event, making them like specialized automation curves. Others are a series of location markers that are used to affect the playback of the audio event.

Only one expression can be focused on at a time, and you pick which expression to view by clicking its name in the list. We will examine them in order, starting at the top of the list. We will then see how programmable expression points can be given a random *Spread* range, and finally look at comping in Bitwig Studio.

Note

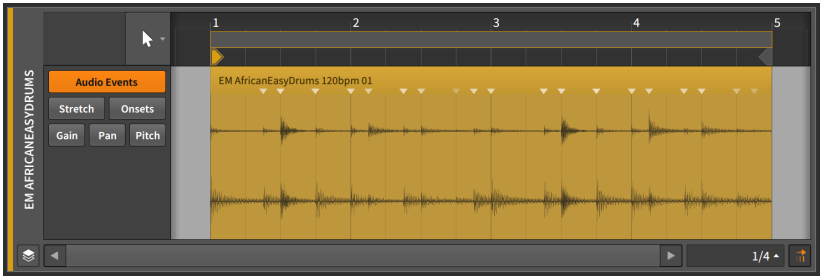
Two expression are not covered here as they are not always available.

When in track editing mode, a *Clips* expression view is available first, which is largely similar to working with clips on the Arranger Timeline (see section 5.1).

When in clip editing mode, a *Comping* expression view is available second. This unique mode for weaving a pile of takes into the perfect performance is covered in its own section (see section 10.1.4).

10.1.2.1. Event Expressions

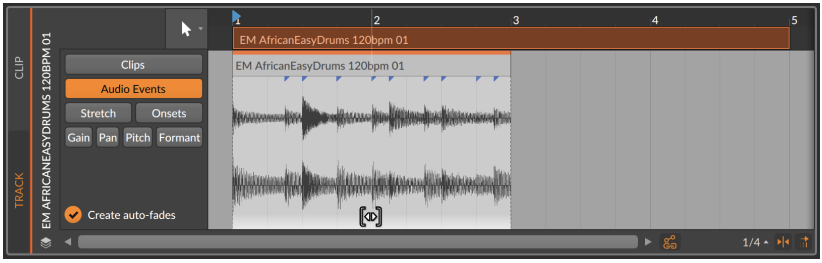
Audio Events presents a simple display of all audio events.



No actual expression curve or other data is shown here. This allows you to freely move and edit the audio events themselves without inadvertently changing other values.

Audio events are moved and adjusted in the same way as clips (see section 5.1.3) except that the range of motion is limited to the length of the parent clip. When compared to the **Arranger Timeline Panel**, all tools function equivalently in this panel except for the pencil tool. And a quick sliding gesture is also available.

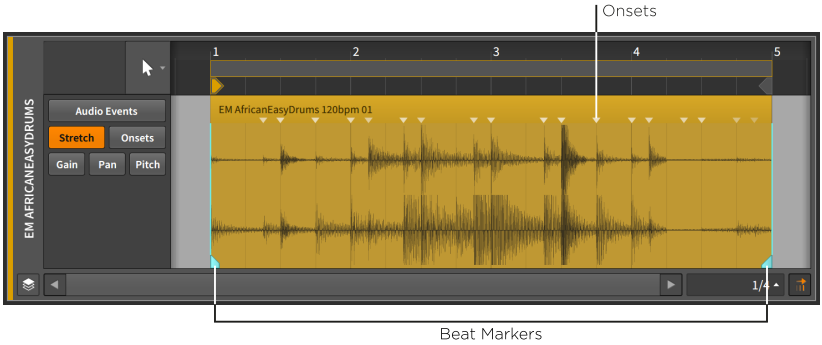
To slide the content of an audio event: mouse over the bottom edge of the waveform and drag horizontally. Or hold [ALT] and drag horizontally from any point on the waveform.



You can optionally add the [SHIFT] key while dragging to toggle the snapping behavior.

10.1.2.2. Stretch Expressions

Stretch expressions determine how the playback speed is altered, thereby stretching the audio file.



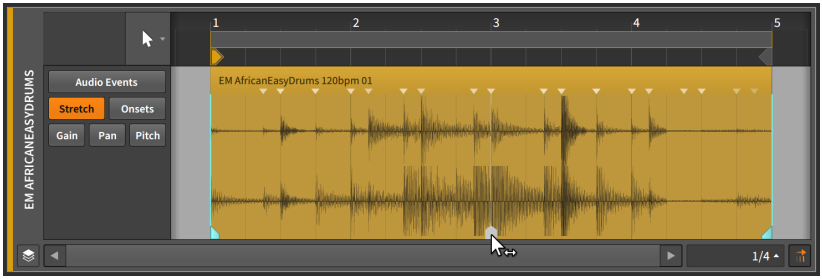
Note

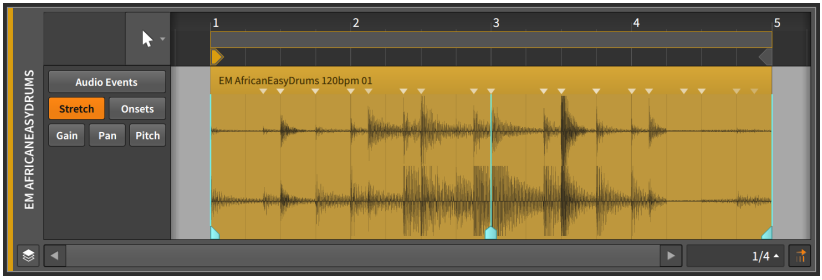
This expression will take effect only with certain audio event playback modes (see section 10.2.1.2).

The stretch function of this expression is achieved by inserting *beat markers*, which dictate the points in the audio event that are locked to their position. The playback speed of the area between beat markers is then altered to ensure that those beat markers occur at their assigned times.

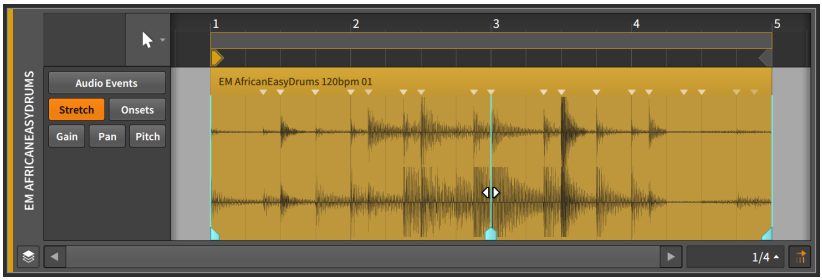
By default, only the start and end times of each event are given beat markers, but the stretch expression makes it easy to create a beat marker where an onset already exists.

To create a beat marker: double-click any area of the event. Or mouse around the bottom of the event, and then single-click any white marker that appears.

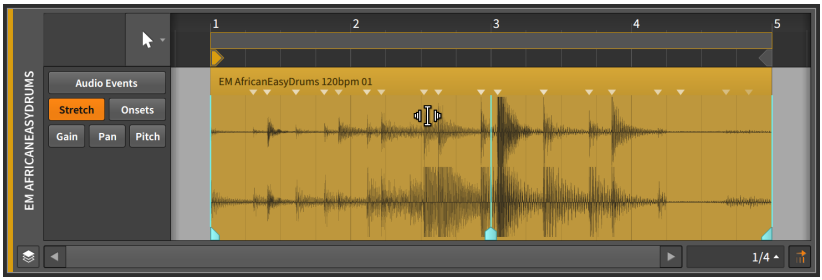




To move a beat marker and its surrounding audio: along the bottom half of the event, click and drag a beat marker with the simple, double-arrow cursor.

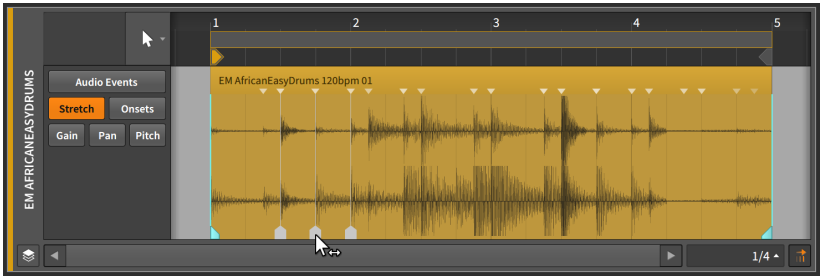


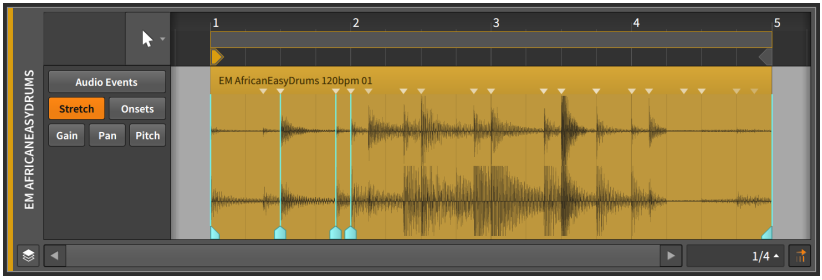
To keep a beat marker in place and fine-tune the position of the audio around it: [ALT]-drag on any beat marker. A cursor with radiating audio on either side will appear for this gesture.



The combination of moving beat markers and then "sliding" them precisely will speed up any workflow involving audio stretching.

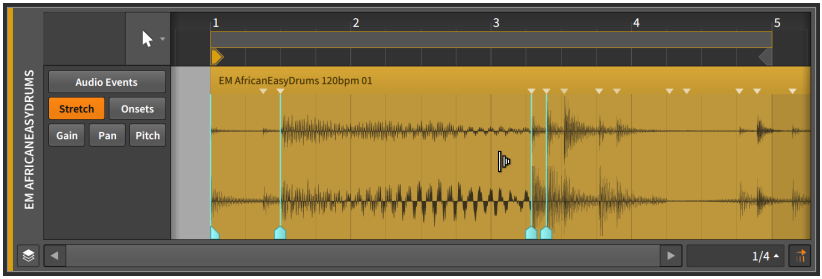
To convert a trio of onsets to beat markers: hold [ALT] and mouse around the top of the event until the desired three markers appear. Then click and drag the mouse horizontally.





This allows you to stretch a particular area of your audio event while keeping the rest of the event unaffected.

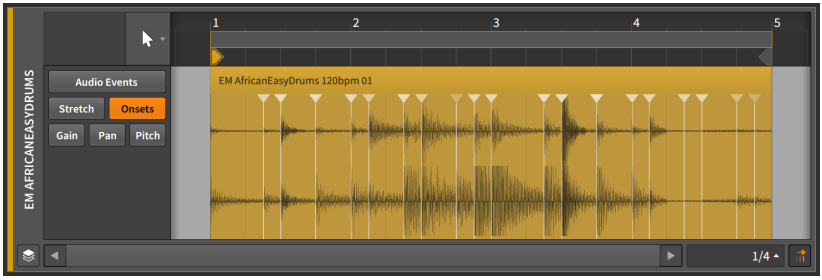
To freely stretch the size of a region: [ALT]-click a region while in *Stretch* view and drag horizontally.



And note that just as with clips, [ALT]-dragging a border of a time selection will scale the entire selection, and [ALT]-dragging an event edge will scale that side — the start or end — of all selected events (see section 10.1.1).

10.1.2.3. Onsets Expression

The *Onsets* expression represents locations in an audio event where the sound's envelope substantially changes, often where individual sounds occur.



Onsets are used both as data to help preserve the sound quality of single audio events, and as demarcations when splitting the component parts of one event into multiple, individual events.

When a sample is initially dragged into a Bitwig Studio project, it is analyzed for its tempo, its musical length, and where onsets occur in the file. Each onset is represented by a vertical blue line that reaches a small blue triangle at the top of the event.

Note

When using a stretch *Mode* that is set to follow *Onsets* and an *Onset Intensity Threshold* that is greater than zero (see section 10.2.1.2), onsets that are below the threshold will be dimmed in the *Onsets* expression view. (In other views, these lower onsets will simply be hidden.)

You can also manually insert or manipulate onsets, either because the automatic results were imprecise or to manipulate how stretching is done during playback (see section 10.1.2.2), etc.

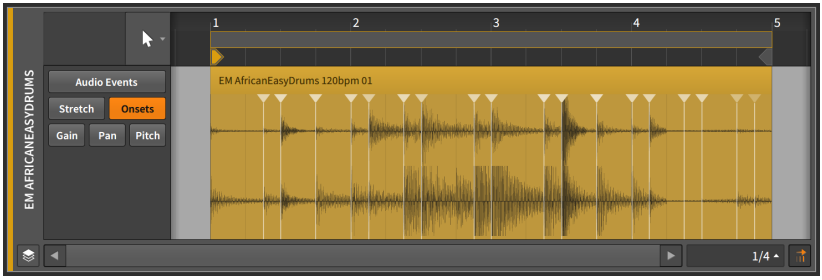
To insert an onset: double-click any area of the event away from a current onset.

To move an onset: click and drag the point with the mouse.

Note

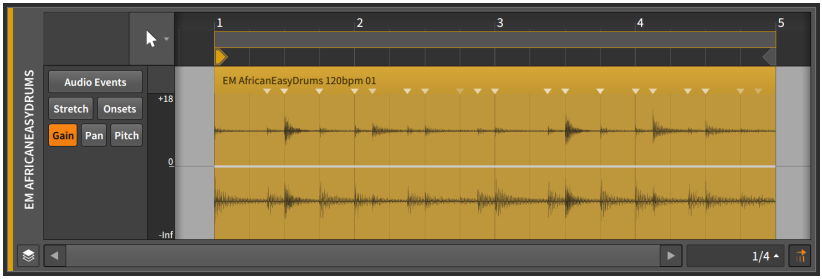
Onsets are colored blue. The more vivid the shade of blue, the stronger the onset. Selected onsets are tinted white.

To delete an onset: double-click it. Or single-click the point to select it, and then press [DELETE] or [BACKSPACE].



10.1.2.4. Gain Expressions

Gain expressions represent a level control for the audio event.

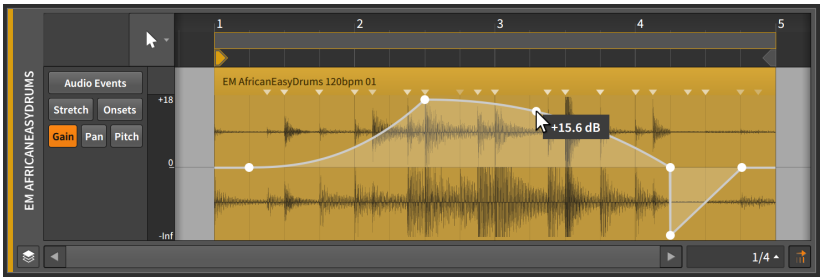


This expression can be made up of a series of points that are created and edited in the same way that automation points are (see section 9.1.2).

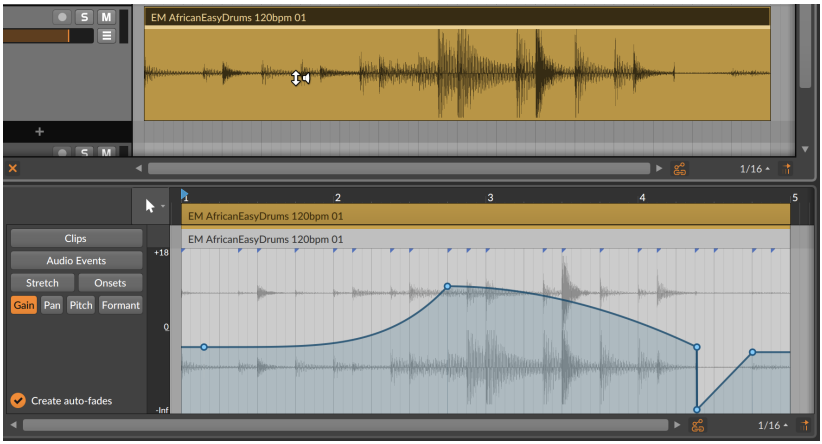
The gain expression is measured in units of decibels with the center line representing zero decibels of change (unity gain).

A gain expression is identical in function to volume automation. The difference is that the expression is applied to the audio source itself, and volume automation is applied as the last stage of a track's signal flow (after the track's device chain and everything else).

Since the gain expression affects the source material, the waveform is helpfully redrawn to show the effect of this expression.

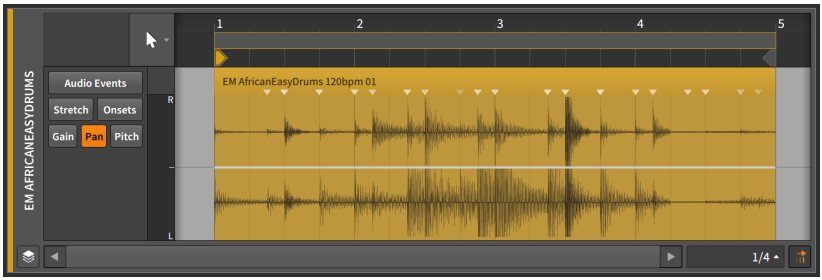


Finally, you can also quickly access a gain handle when working with *Audio Events* in the **Detail Editor Panel** by mousing just beneath the event's title and then clicking and dragging up or down. This handle is also available when working with *Clips* either in the **Detail Editor Panel** or directly in the **Arranger Timeline Panel**.



10.1.2.5. Pan Expressions

Pan expressions represent a stereo placement control for the audio event.



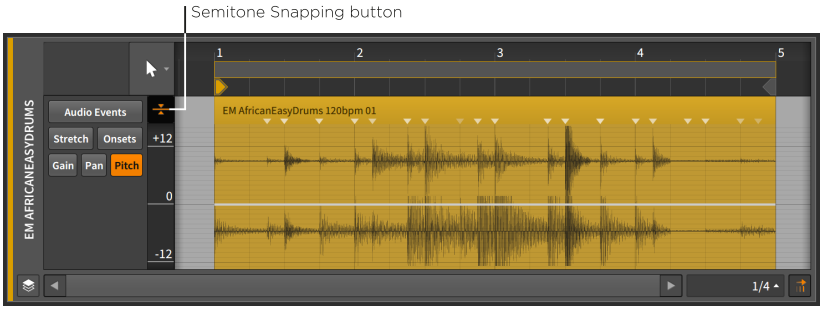
This expression can be made up of a series of points that are created and edited in the same way that automation points are (see section 9.1.2).

A pan expression is measured as a bipolar percentage with the center line at *0.00%* (center placement, or no panning adjustment), *100%* for hard right, and *-100%* for hard left.

As with the gain expression, the *pan expression* is applied to the audio source itself. This has no direct interaction with *pan automation*, which is applied by the track mixer after the device chain.

10.1.2.6. Pitch Expressions

Pitch expressions represent a frequency transposition control for the audio event.



Note

This expression will take effect only with certain audio event playback modes (see section 10.2.1.2). When an incompatible playback mode is selected, any expression data will be stored but shown with very small dots, to indicate it is not currently being used.

This expression can be made up of a series of points that are created and edited in the same way that automation points are (see section 9.1.2).

A pitch expression is measured in semitones (or half steps) with the center line at *0.00* (zero semitone shift for no transposition), a maximum of *24.00* (two octaves up), and a minimum of *-24.00* (two octaves down).

Note

Unlike the other expressions, the pitch expression's vertical axis is scrollable and zoomable (by clicking and dragging it). Because of this, it will not automatically compact itself to fit a small **Detail Editor Panel**.

The *semitone snapping* option causes pitch point changes to snap to whole number semitones. As with the position snapping options (see section 5.1.2), holding [SHIFT] will toggle this behavior.

10.1.2.7. Formant Expressions

Formant expressions represent a shift of the formants in audio event via the selected playback mode. Like Pitch, it is set in semitones and has a scrollable editor.

Note

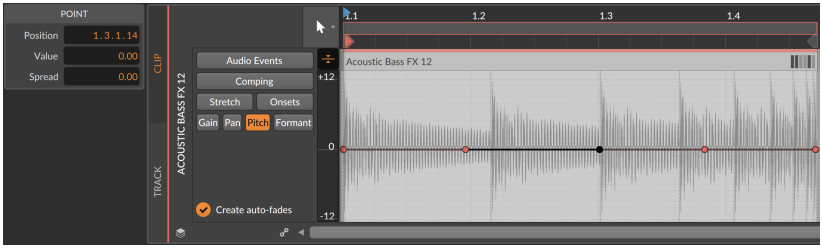
This expression will take effect only with certain audio event playback modes (see section 10.2.1.2). When an incompatible playback mode is selected, any expression data will be stored but shown with very small dots, to indicate it is not currently being used.

10.1.3. Expression Spread

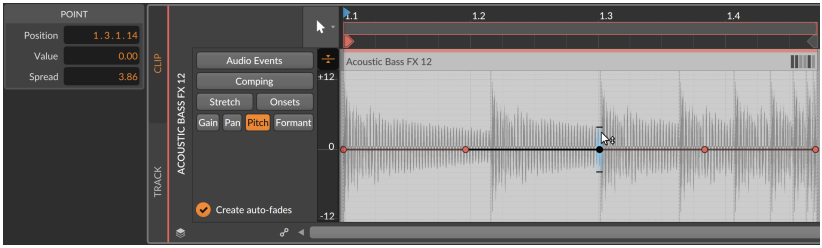
The *Spread* option offers randomization of expressions points, so it is available for those that are programmed like automation — for audio

events, this includes Gain, Pan, Pitch, and Formant. This turns any defined point into a range of possible values.

For this example, we'll start with some pitch expression points in a single audio event.

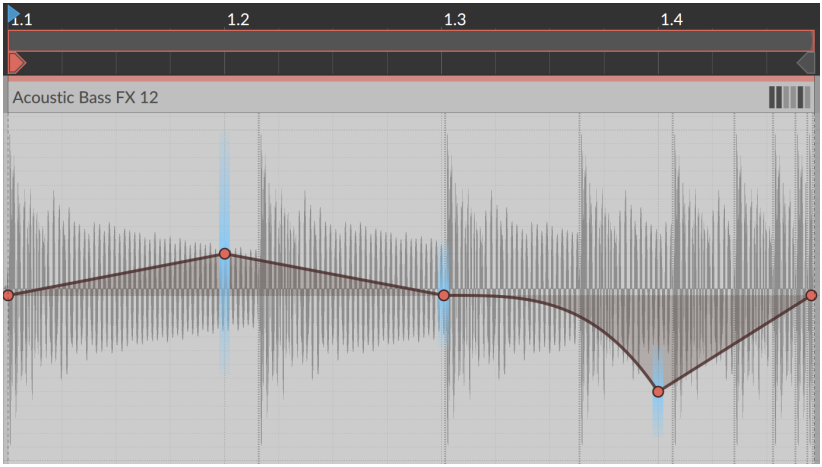


To define a Spread range for any expression point: [ALT]-drag the expression point up and down.

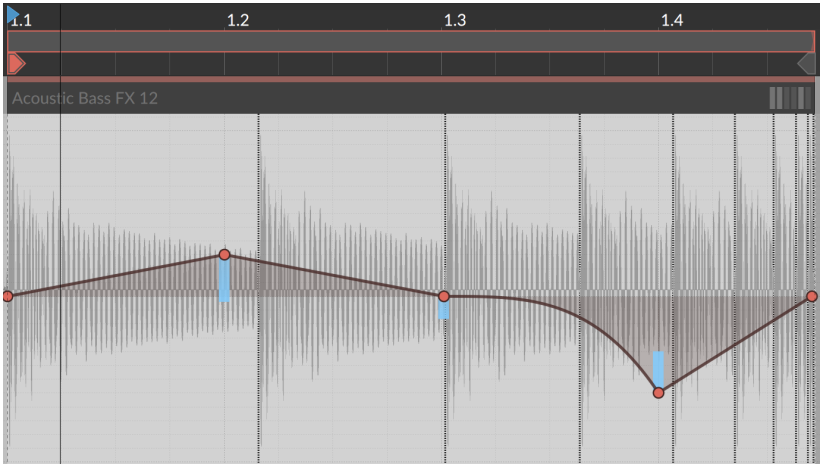


Note that the *POINT* section of the **Inspector Panel** allows you to see and type the *Spread* value, or to use the **Histogram** when multiple values are selected (see section 10.2.2.2).

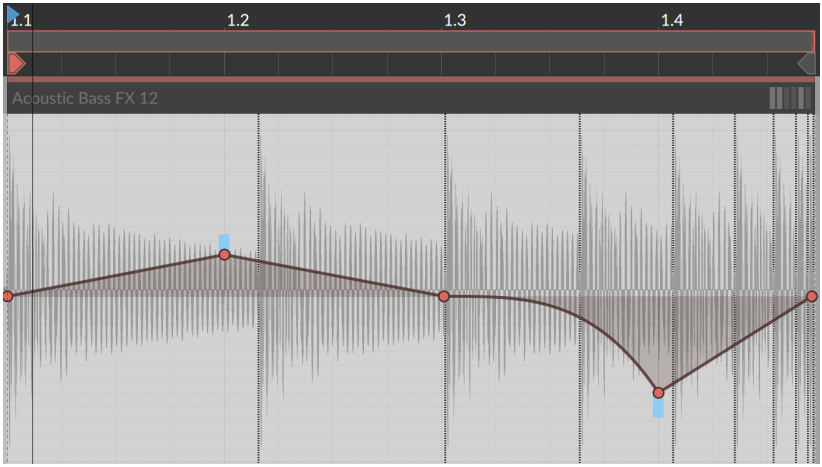
While interacting with expression points, the black horizontal dashes are shown above and below, clearly indicating the extremities of the range. When no points are selected, the highlighter-style gradients remain.



When the parent clip begins playback, the values selected for playback during that cycle will all be visualized immediately.



And new random values will be selected for the next cycle (assuming the clip loops), or the next time the clip starts to play again.



Finally, a few notes on the nature of random playback.

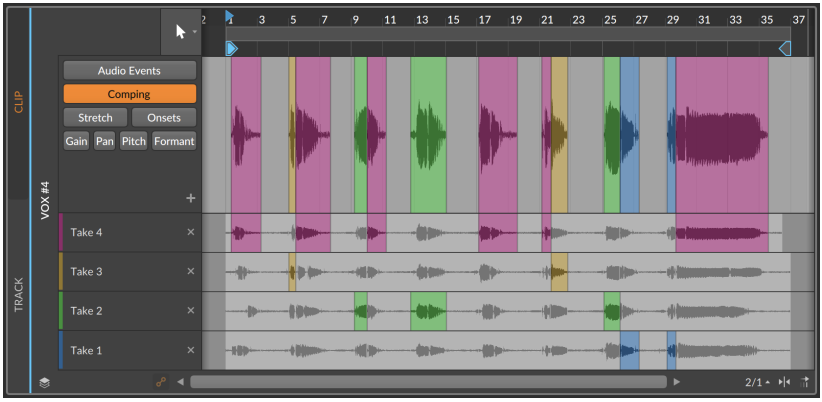
› Randomized expression points are traveled to smoothly, as if they had been drawn manually with a slope connecting it to the previous and following points. So even in the pitch examples at the top of this section (which look like a flat line), the slope value between sections is still used.

› As these values are randomized, they are tied to the clip's *Seed* setting (see section 5.1.10.7). If *Seed* is set to *Random*, then new values will be selected each time the clip restarts, which includes each loop cycle. If a *Seed* value is set, then the random pattern produced will repeat for each playback.

› If you want to print the randomness into a clip, you could try the *Consolidate* function (see section 12.2.3). And if you want to generate new and/or longer clips from the original, you could try the *Expand* function (see section 12.2.2).

10.1.4. Comping in Bitwig Studio

When in clip editing mode, audio clips offer a *Comping* expression view. If you are recording audio, you can "cycle record" takes straight in, whether working in the Arranger (see section 5.3.3.3) or in the Launcher (see section 6.4.2). The material may look flat at first, but after a few swipes, drags, and arrow key taps, the composite will clearly and pleasantly indicate the source takes used.

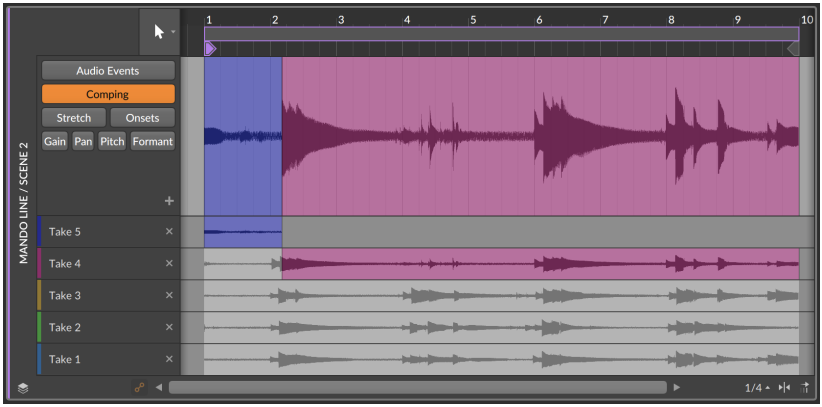


The following sections cover the expansive workflow for comping, as well as some ways to insert and work with takes themselves.

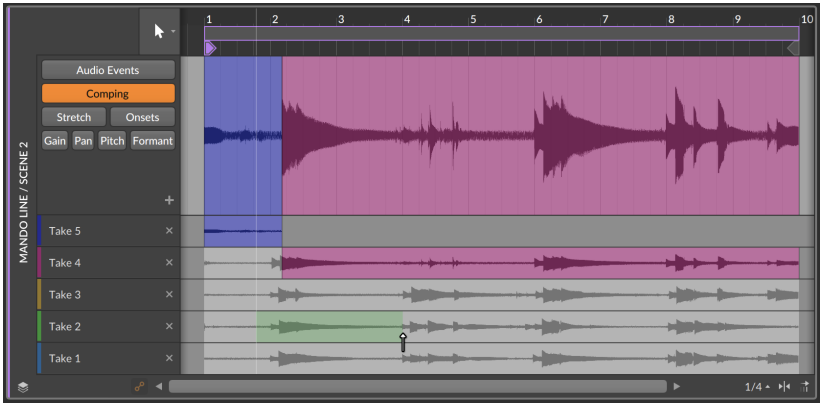
10.1.4.1. Comp Editing Workflow

Comping in Bitwig Studio is based on the idea of defining *comp regions*, and then selecting which of the available take lanes (if any) is played within that region.

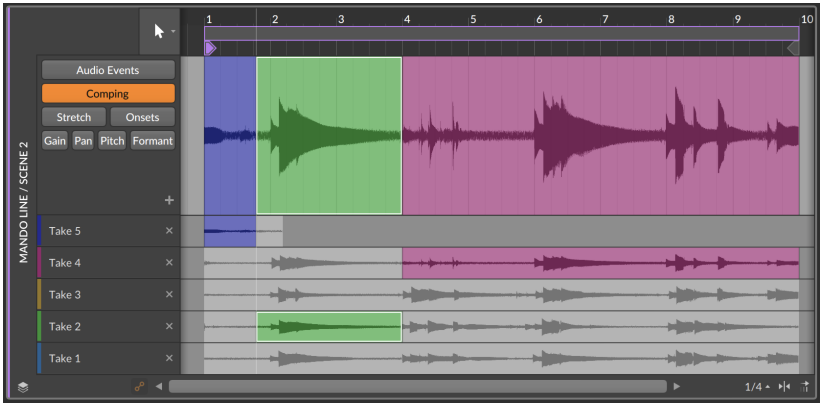
When using "cycle recording" to create takes, newly recorded comping material will tend to look like this at first.



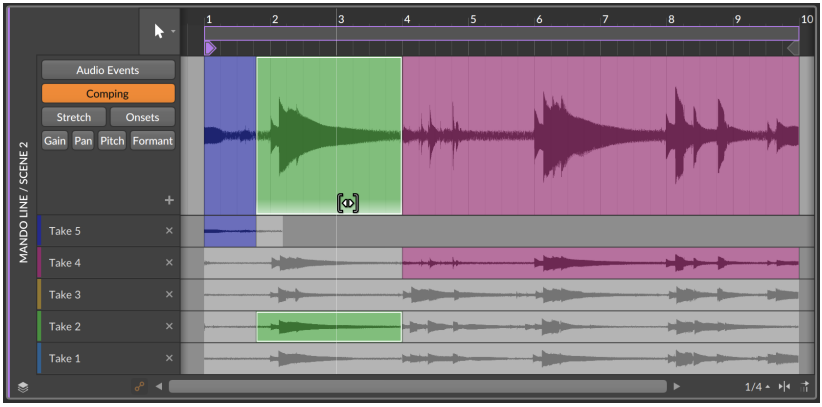
To define a comp region: click and drag over a portion of any take lane.



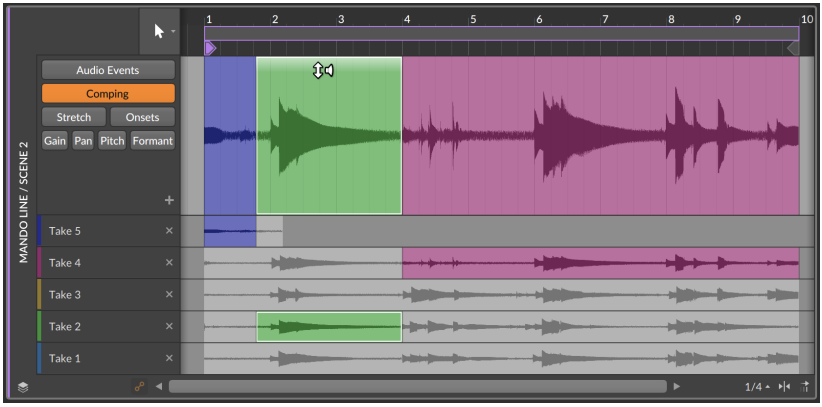
Once the click is released, the region will be shown as active in its take lane and painted into the composite lane at top.



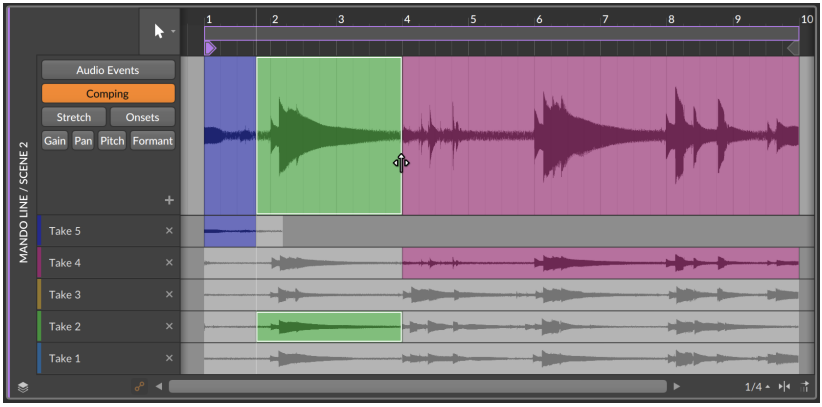
To slide a comp region: mouse over the bottom of the region's waveform in the composite lane. Then click and drag left or right.



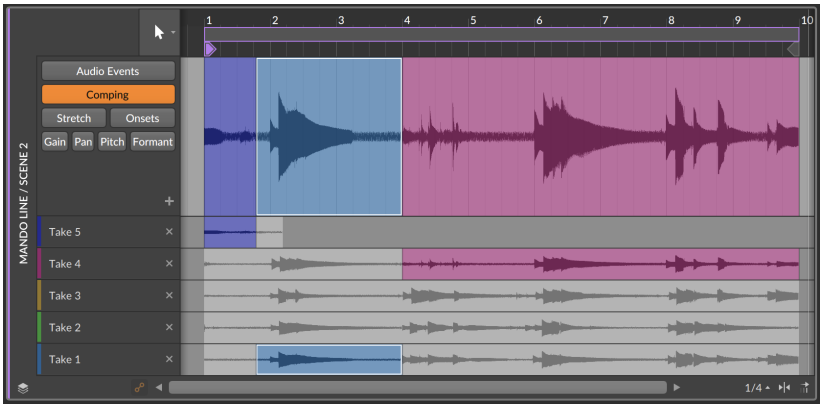
To change the gain of a comp region: mouse over the top of the region's waveform in the composite lane. Then click and drag up or down.



To adjust a comp region border: mouse over the boundary and then click and drag. This moves both adjacent regions together, and it can also be done on the edge of a region in any take lane.

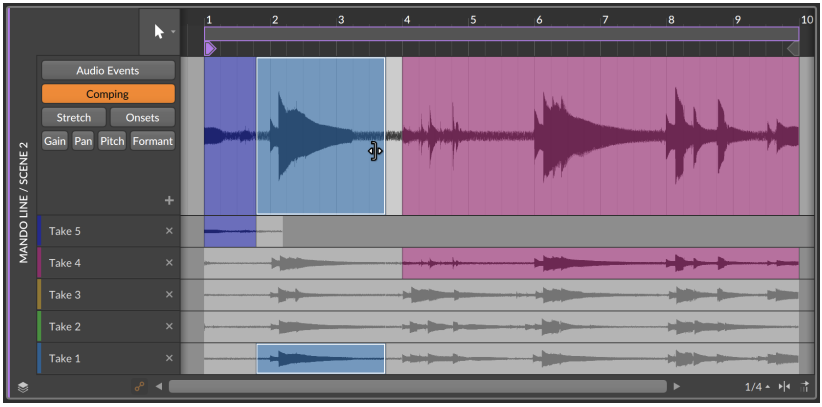


To point a comp region to a different take lane: click on any inactive portion of a take lane.

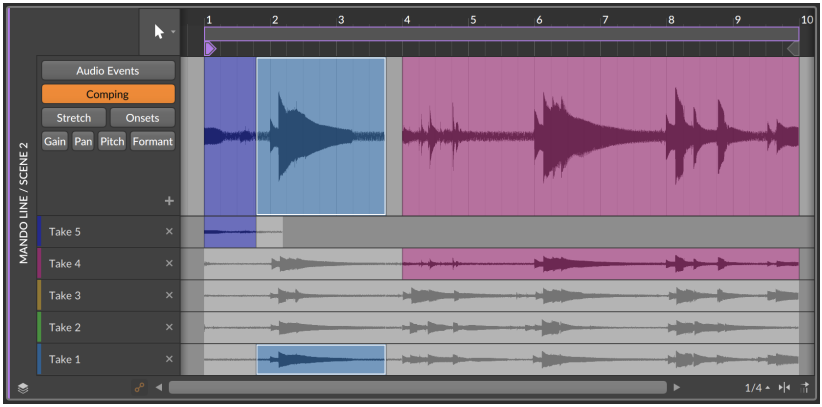


Or when a comp region is already selected, you can press the [UP ARROW] and [DOWN ARROW] keys to activate one of the nearest take lanes. The [LEFT ARROW] and [RIGHT ARROW] keys also move selection to the previous or next comp region. So once your comp regions are defined, a lot of auditioning and editing can be done with just the arrow keys.

To adjust a comp region border in one direction: mouse near the boundary so that a one-sided bracket cursor appears, and then click and drag.



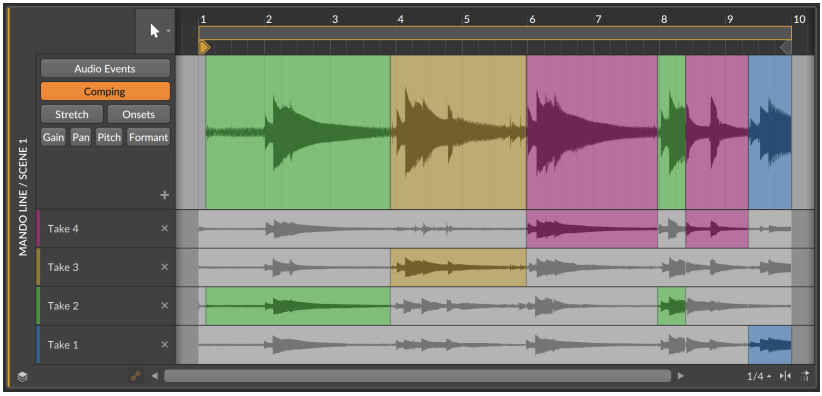
When the click is released, the excluded portion will be deleted from the composite.



All comping gestures can be applied to multiple comps, keeping them in sync. This is available in layered editing mode (see section 11.1.5).

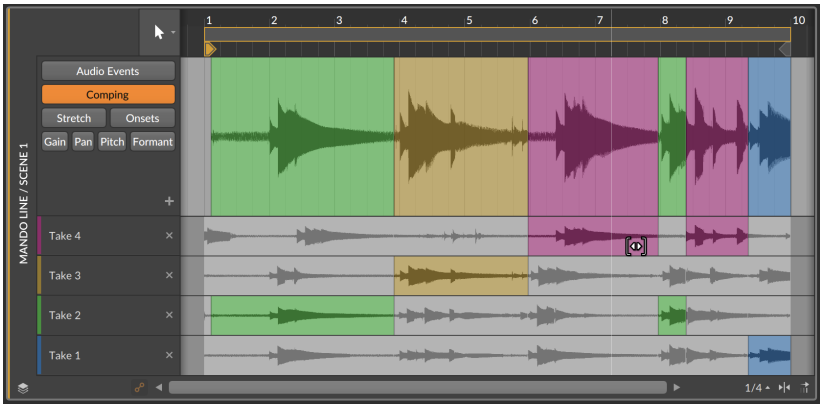
10.1.4.2. Adding and Working with Takes

Some comping functions are provided within the take lanes.

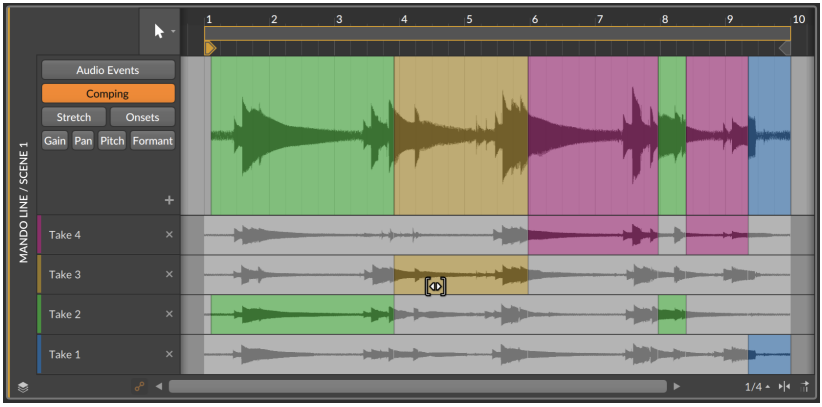


While slide is available for single comp regions, it is also available for a full take, or even to shift all takes.

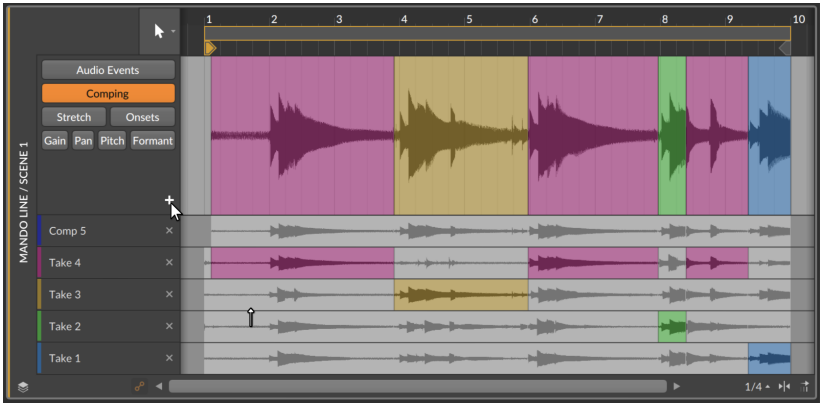
To slide a take: hold [ALT] and drag any part of the take lane horizontally. In this example, the red take lane is being dragged to be later.



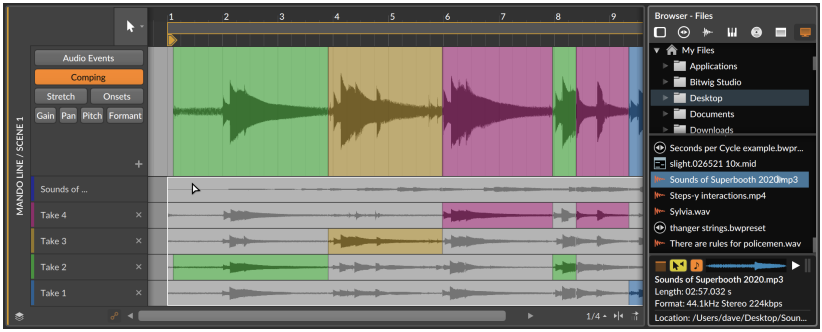
To slide all takes: hold [SHIFT]+[ALT] and drag any take lane horizontally. In this example, all take lanes are being dragged to be earlier.



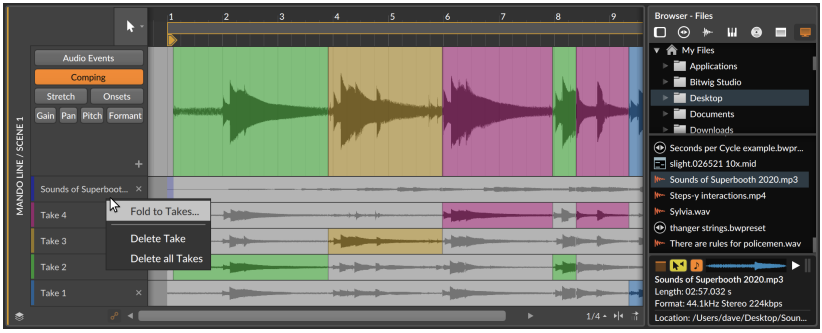
To copy the current composite as a unique take: click the plus (+) button.



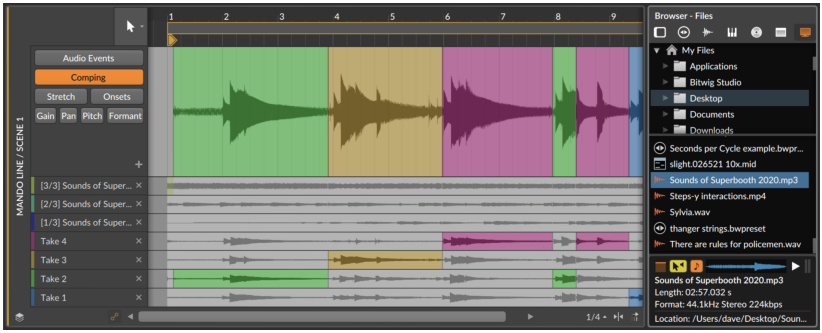
To add an audio file to a comp as a new take: navigate to the desired audio file in the **Browser Panel**, and then drag it into the comp.



And finally the *Fold to Takes...* function for wrapping an audio clip into successive take lanes (see section 5.1.10.8) is available on take lanes as well. Start by right-clicking on the take you want to divide.



After selecting *Fold to Takes...* and filling out the dialog as desired, successive takes will be placed at the top of the comp.



10.2. Inspecting Audio Clips

As was said in this chapter's introduction, we have been using the **Inspector Panel** to examine clips for quite some time. In addition to the clip settings we have already examined, any non-empty clip has a large section at the bottom of the **Inspector Panel** for dealing with its musical contents.

10.2.1. The Inspector Panel on Audio Events

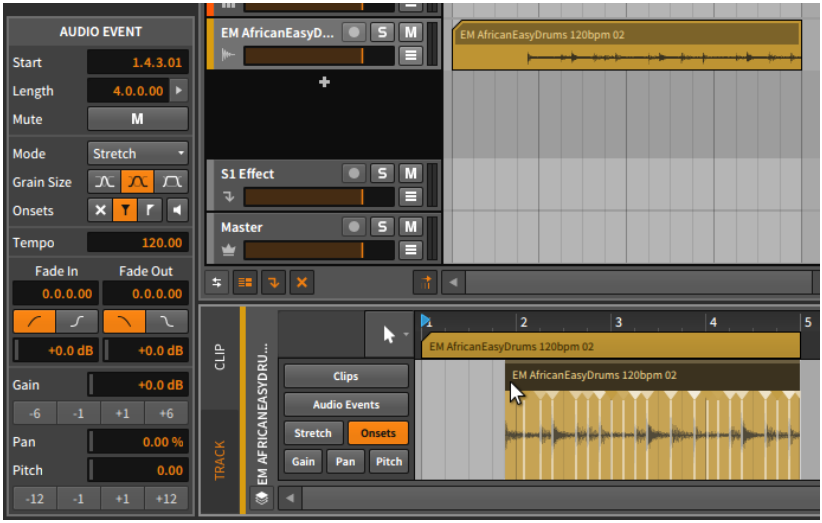
By selecting a clip, certain parameters are revealed in the *AUDIO EVENT* section, but when selecting an audio event itself (by single-clicking the audio event's header in the **Detail Editor Panel**), the **Inspector Panel** provides all settings relevant to the selected event(s).

Several of these settings will be familiar. Since there are many of them, we will take them one section at a time. And we will also look at the functions available in the *Event* menu when audio events are selected.

10.2.1.1. Timing Section

These settings generally relate to the musical position of the selected event and its optional fades:

› *Start* sets the start position of the event within its parent clip or track. Adjusting this position will move the audio event as it exists, the same as clicking and dragging the event within the **Detail Editor Panel**.



Note

Remember that audio events will be always be truncated by the boundaries of their parent clip.

› *Length* sets the duration of the event within its parent clip. Adjusting this duration will simply lengthen or shorten the event, the same as using the bracket cursor to adjust the right edge of the event's header.

› *Mute* toggles whether or not the event is disabled on playback.

10.2.1.2. Stretch Section

These settings relate to the behavior of Bitwig Studio's audio playback.

› *Mode* sets the audio playback algorithm for the audio event. The settings are grouped under categories that describe the general method being used to produce audio stretching.

› *GRANULAR* modes work in the time domain, allowing independent control of pitch and time.

Stretch is an optimized algorithm that time-stretches audio to match the project's tempo. When your settings match the original audio (targeting the original pitch and tempo), this algorithm is completely neutral, preserving your original audio at output and lowering your processor's load.

Stretch HD is a similar algorithm to *Stretch* but is applied in a multiband fashion, splitting the original signal into several frequency areas and stretching those.

Slice divides audio into chunks and then stretches those chunks (when appropriate) using the method set in the *Tail* parameter.

Cyclic adds overlaps to stretched audio in the fashion of classic hardware samplers.

Elastique Solo syncs its grain size to the wavelength of the audio. This makes it especially useful for voice or other monophonic sound sources. But any source material may yield interest results and/or robots.

› *SPECTRAL* modes work in the spectral domain, allowing independent control of pitch and time.

Elastique preserves transients, making it appropriate when rhythmic accuracy is important.

Elastique Eco focuses more on harmonic content, making it more useful for less rhythmically-active sounds (like pads).

Elastique Pro also preserves transients but has formant controls as well. This comes at the cost of additional CPU resources.

› *UNSTRETCHED* modes do not provide independent control over pitch and time.

Raw ignores all stretch expression data. Events are played back at their original speed, regardless of the project tempo or any other considerations.

Repitch ties pitch and playback speed together (as a tape recorder would). Stretch expression data is respected while pitch expressions are ignored.

Each stretching mode has up to three of the following parameters available:

› *Grain Size* adjusts the length of each audio segment that is stretched in the selected audio event. The three relative options are for short, medium, or long portions of the audio to get processed at a time.

› *Transients* controls how the onsets expression (see section 10.1.2.3) is used to adjust playback. There are three options to choose between and one optional mode:

› The first option is *off*, represented by an *x* icon. In this mode, the onsets expression is completely ignored for playback purposes.

› The second option is *soft*, represented by a centered vertical line with both a "fade out" triangle on the top left and a "fade in" triangle on the top right. This mode emphasizes smoothness by blending the audio before an onset with that that comes after.

- The third option is *hard*, represented by a centered vertical line with only a "fade in" triangle on the top right. This mode emphasizes rhythmic accuracy by focusing on the audio that comes after the onset.

- The separate button with the speaker icon represents preview mode. When toggled on, this mode plays the audio at each onset, but turns the volume down for all other parts of the event. This is a useful audible indicator of where the onsets are currently placed.

- *Rate* sets the interval at which audio is divided for processing and stretching. Options include:

  Transient Rate connects processing to the occurrence of *Onsets* by default (shown with a small calendar icon). But his can also be set to regular beats intervals (such as every *1/16* note), which could be useful for material without clearly discernable onsets.

  Onset Intensity Threshold is available when *Transient Rate* is set to follow *Onsets*. This threshold control is set in percentage, allowing the exclusion of onsets that are weaker than this set level.

- *Tail* sets a method for overlapping audio tails when stretching is needed. Settings include *None*, standard *Granular* stretching, and stacked *Ping-Pong* delays (as used by some vintage samplers).

- *Formant* offers two controls for shifting the formants of the affected audio:

  The button showing a keyboard with outward-facing arrows toggles the automatic shifting of formants based on the pitch expression.

  The numeric control allows you to set a fixed shift amount in semitones.

> **Note**
>
> When a *Formant* parameter is available for the current playback mode, that value can also be automated via the audio events' formant expressions (see section 10.1.2.7).

- *Resolution* sets the relative size of the spectral envelope used for formant shifting. Larger values create larger windows (which are better tuned for lower frequencies), etc.

- *Play Stop* allows you to set an end time (in `MINUTEs:SECONDs.MILLISECONDs`) for the audio event. Regardless of

clip length and anything else, the audio event will not play beyond this point. (Setting the value in time keeps tempo changes from interacting with this playback value.)

10.2.1.3. Tempo Section

Tempo defines the original tempo of the audio event. Knowing this enables Bitwig Studio to properly play back the data in any circumstance.

When an audio file is brought into a project, the program first checks the filename for an indication of tempo (such as the word *154bpm*). If nothing is found there, the program determines the tempo as best it can.

This value can be corrected at any time, but changing it will impact the placement and timing of the audio event.

10.2.1.4. Fades Section

The *Fade In* and *Fade Out* parameter sets allow you to define independent fades at the beginning and end of each audio event. When set in tandem with an overlapping clip, you can also create crossfades in this way.

All of the parameters and methods of operation are the same as when applied at the clip level (see section 5.1.7).

10.2.1.5. Operators Section

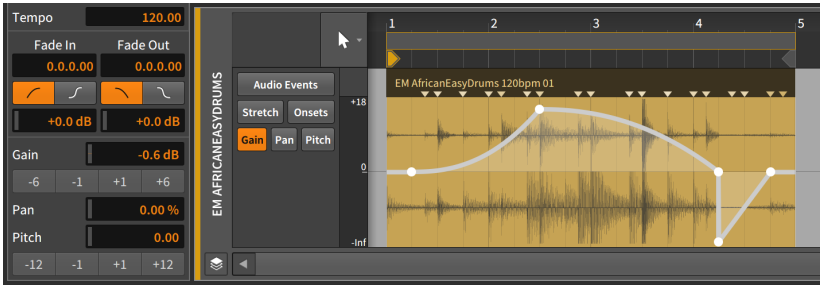
Unlike the other sections in the **Inspector Panel**, the section displaying **Operators** is only shown when events (and not clips) are selected. **Operators** are covered extensively in their own chapter (see chapter 12).

10.2.1.6. Expressions Section

This section exposes three of the expressions we have covered: *Gain* (see section 10.1.2.4), *Pan* (see section 10.1.2.5), and *Pitch* (see section 10.1.2.6). While these expressions have completely different functions, they are programmed in the same fashion.

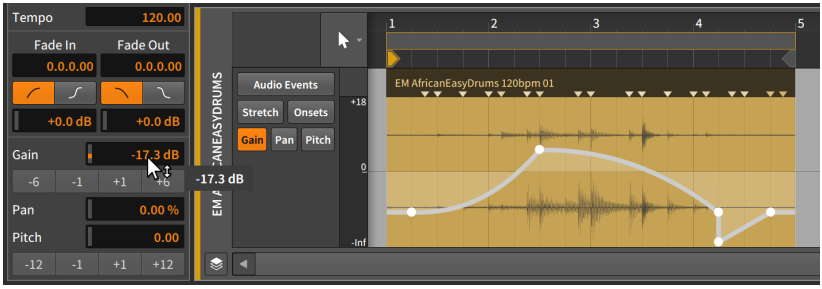
Following the *Gain* and *Pitch* numeric controls are incrementer and decrementer buttons that will adjust the expression value by the declared amount. For the *Gain* expression, these buttons express decibel changes. For the *Pitch* expression, the unit is semitones.

These are the automation-type expressions, each able to be defined by a curve made from several values. Because of this possibility, each value in this section of the **Inspector Panel** is actually representing the average of points in that expression. Let's examine the *Gain* expression as an example.



The *Gain* value listed of *-0.58 dB* is an average of the five points defined in this audio event expression.

To adjust an expression curve: change its listed average value, or click one of the expression's incrementer/decrementer buttons.



This method will work for any expression in this section, whether it is defined by a curve or a single value.

Finally, while right-arrow buttons at the edge of parameter fields are normally reserved for the **Histogram** interface for working with multiple events selected (see section 10.2.2.2), these buttons are also present when any number of expression points or even a clip is selected.

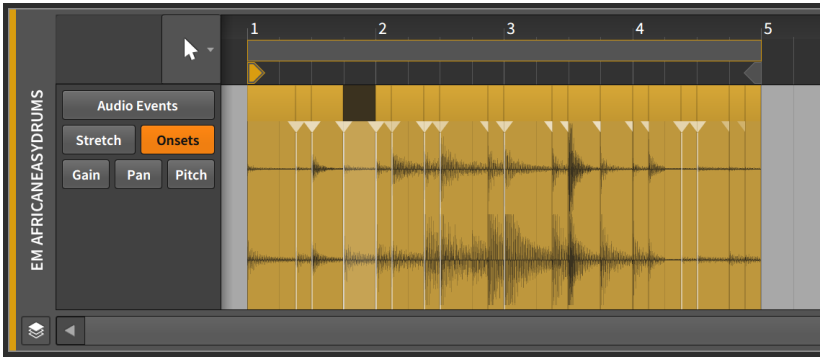
This provides a way to both see the average *Spread* value for all selected points, and a way to adjust them relatively.

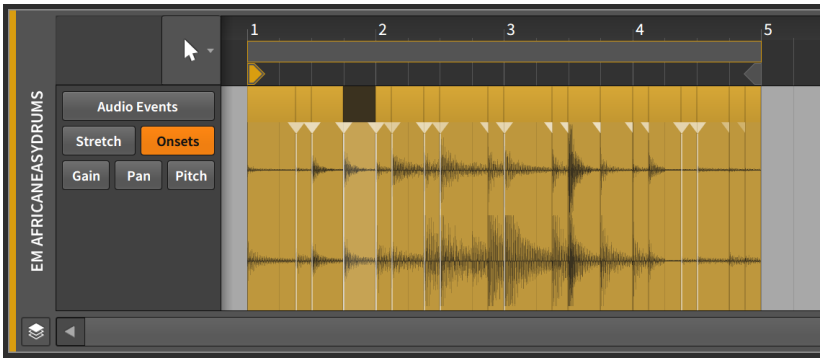
10.2.1.7. Event Menu Functions

These functions take the specified action on the selected audio event(s):

› *Slide Waveform to Previous Onset* moves the selected event so that it begins at the previous onset marker, effectively shifting this area to play earlier material. This affects only the content of the selected event.

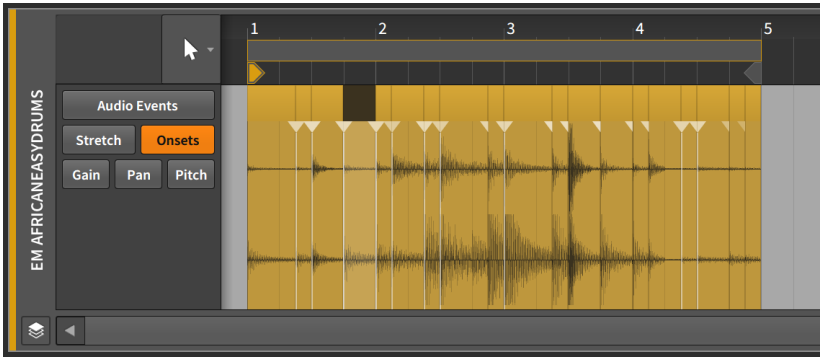
The following images demonstrate a selected event both before and after the *Slide Waveform to Previous Onset* function is applied:

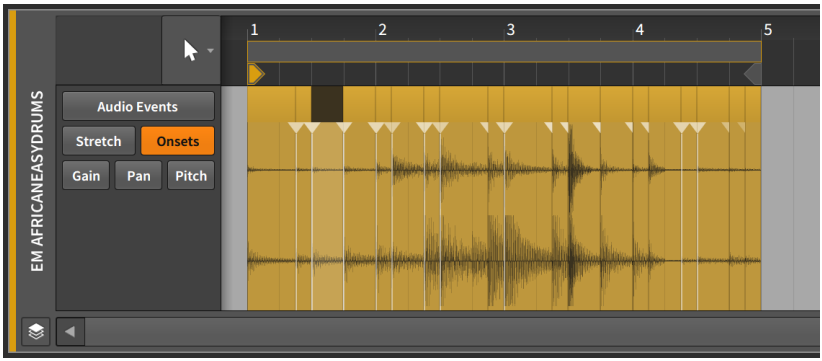




› *Slide Waveform to Next Onset* moves the selected event so that it begins at the next onset marker, effectively shifting this area to play later material. This affects only the content of the selected event.

The following images demonstrate a selected event both before and after the *Slide Waveform to Next Onset* function is applied:





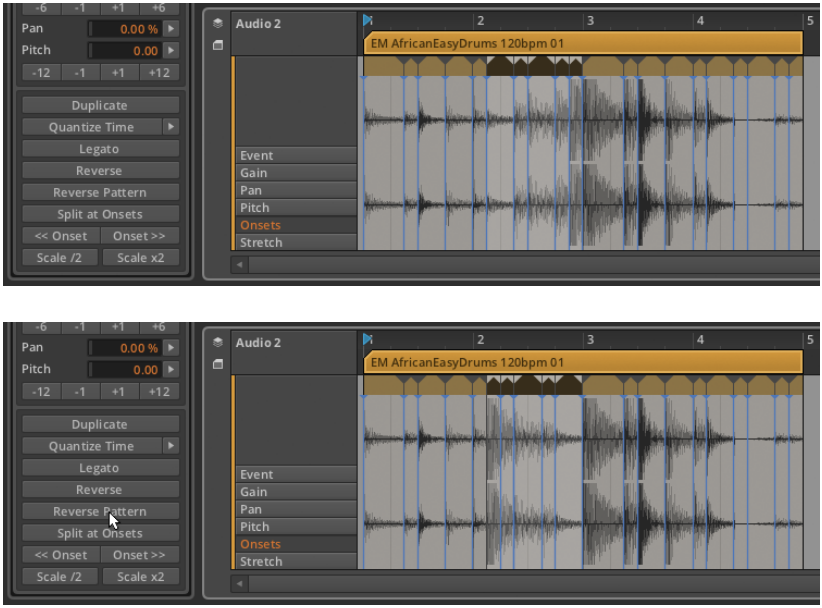
› *Reverse* flips the selected event around, causing it to play backwards. This also flips any event expression curves.

› *Reverse Pattern* flips the order of a group of selected events. This does not cause each event and its expressions to play backwards, but rather causes the last event to be played first, etc.

Note

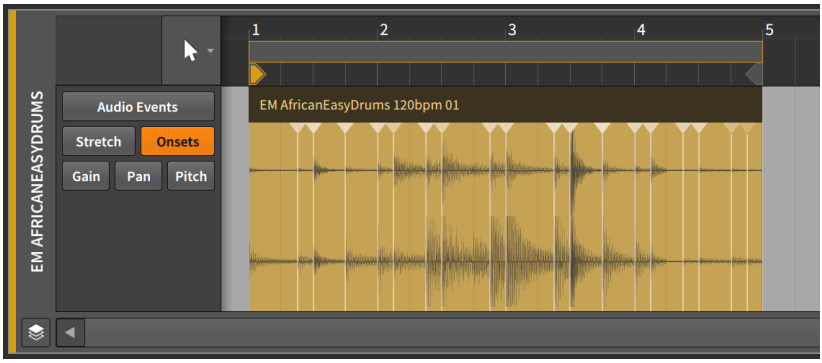
This function will work only when multiple events are selected.

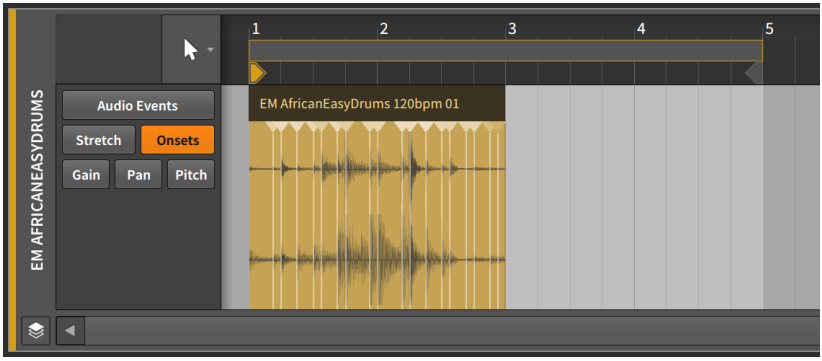
The following images demonstrate a group of selected events both before and after the *Reverse Pattern* function is applied:



› *Scale 50%* halves the length of the selected event, effectively causing it to play back twice as fast. All onset and beat markers are also proportionally shifted.

The following images demonstrate a selected event both before and after the *Scale 50%* function is applied:

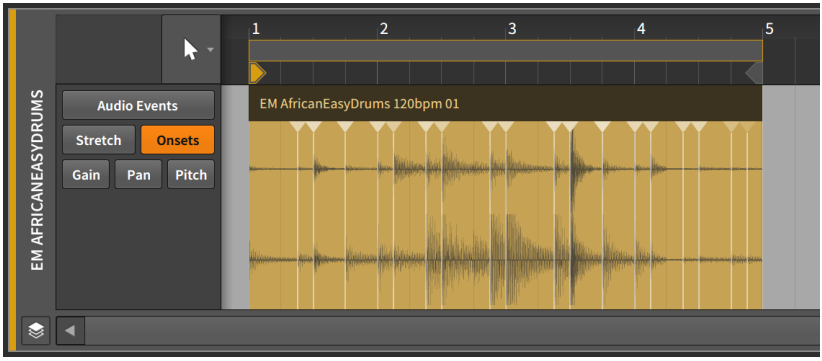


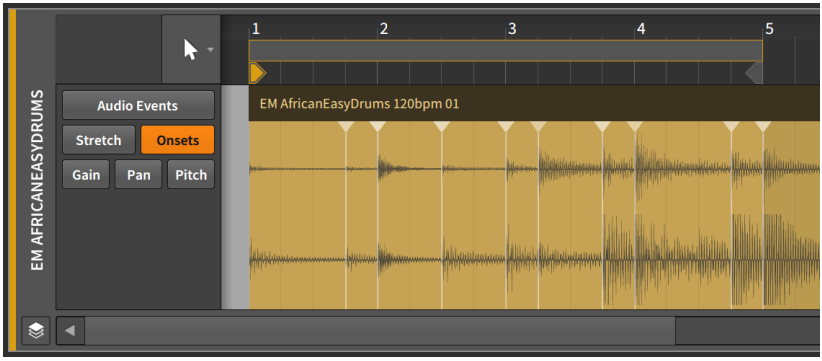


› *Scale Each 50%* is similar to *Scale 50%*, except the start time of each selected audio event is preserved.

› *Scale 200%* doubles the length of the selected event, effectively causing it to play back half as fast. All onset and beat markers are also proportionally shifted.

The following images demonstrate a selected event both before and after the *Scale 200%* function is applied:





› *Scale Each 200%* is similar to *Scale 200%*, except the start time of each selected audio event is preserved.

› *Scale...* requires a set *Amount* of scaling to be typed in, along with an option to *Scale each (keep position)*, which preserves the start time of each selected audio event.

› *Unstretch* removes all stretch markers from the selected audio events to restore their original character.

› *Slice In Place...* divides the selected event into multiple events. A dialog allows slicing either at *Onsets* (the detected transients), at *Beat Markers* (defined stretch points that you may have changed), or at a regular note interval (*on Beat Grid*). This can be an extremely efficient way to do audio edits, especially by splitting at onsets and then working with the returned audio events.

If *Onsets* is selected, an optional *Onset Threshold* parameter is provided, which will default to the event's current playback setting. And like the *Quantize Audio...* dialog, an open **Detail Editor Panel** will remain bright to preview which onsets will used.

Note

When an audio event is split (either by this function with the Knife tool), fade ins and outs will be added to split points if the option *Automatically create fades on audio clip/event edits* is enabled. This preference can be found in the **Dashboard**, under the *Settings* tab, on the *Behavior* page, in the *Fades* section.

› *Slice At Repeats* splits any selected audio event using the *Repeats* Operator into individual events (see section 12.2.1). When a selected event does not have *Repeats* enabled, no change is made.

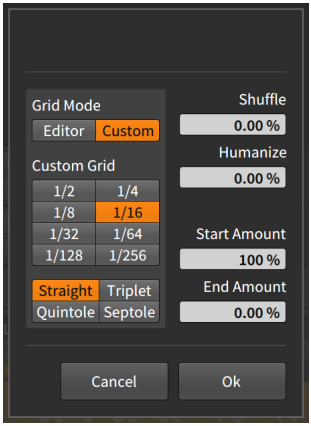
› *Reset Fades* removes any applied fades from the selected audio events.

› *Auto-Fade* applies a quick, relative fade in and fade out to all selected audio events.

› *Auto-Crossfade* applies a quick, relative pre-fade and fade out to all selected audio events, creating crossfades between adjacent events.

› *Quantize* is identical to the following *Quantize...* function except that the most recently set parameters are used for the function.

› *Quantize...* moves the start and/or end times of selected events in relation to a beat grid.



› *Grid Mode*: Determines whether to adopt the grid settings from the current *Editor* or to allow *Custom* grid settings.

› *Custom Grid*: Exclusive *beat grid resolution* and *beat grid subdivision* settings (see section 3.1.2) for the quantize function.

Note

This is available only when *Grid Mode* is set to *Custom*.

› *Shuffle*: Amount of swing/groove (see section 2.3.2) applied to the beat grid for the quantization function.

› *Humanize*: Amount of randomness added to the quantize function, with the intention of mimicking human imperfection.

› *Start Amount*: Amount of quantization applied to each selected event's start position.

For example, a setting of *50.0%* would move a selected event's start position halfway to the closest grid point. A setting of *100%* places the event exactly on the closest grid point.

› *End Amount*: Amount of quantization applied to each selected event's end position.

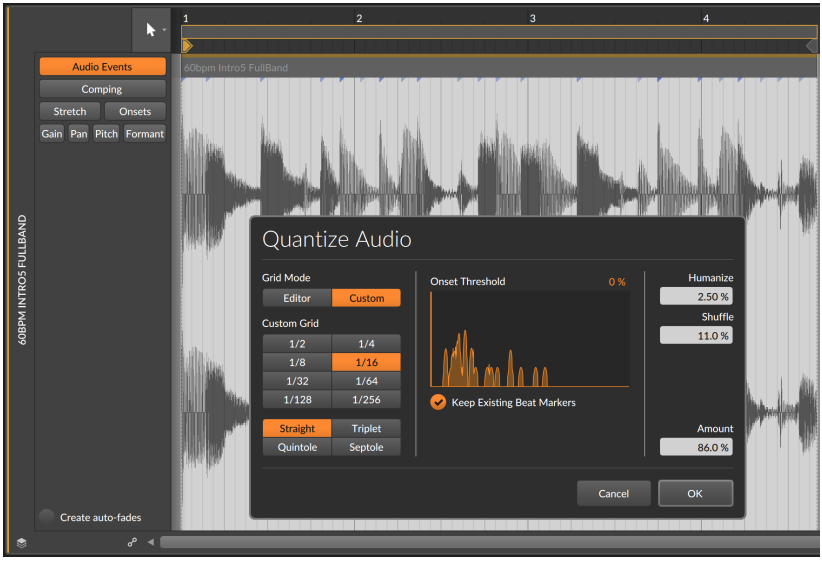
Note

Humanize is the last factor applied in the quantize function. So even *Start Amount* of *100%* might not place events directly on the grid if *Humanize* is enabled.

The quantize function can be executed by either clicking the *Apply* button at the bottom of the parameter pane, or by clicking the *Quantize Time* button itself.

› *Quantize Audio* is identical to the following *Quantize Audio...* function except that the most recently set parameters are used for the function.

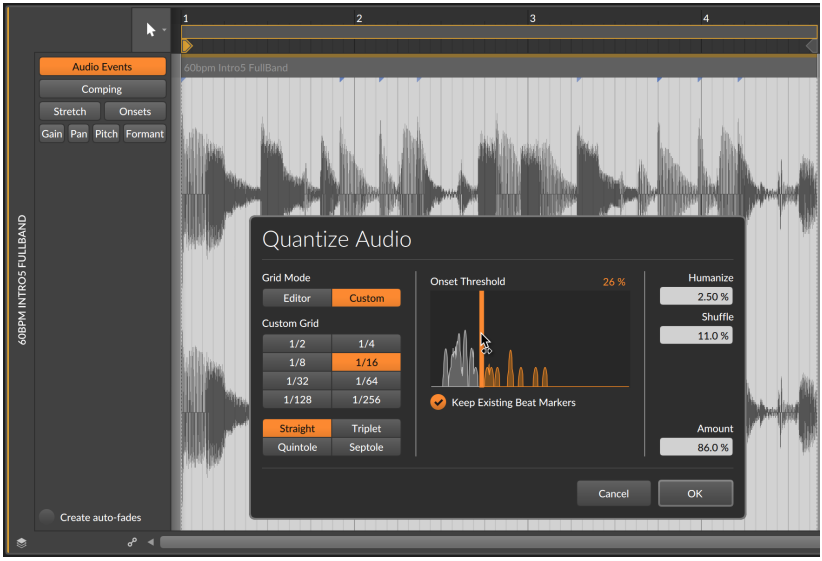
› *Quantize Audio...* is a high-level variation on the *Quantize...* function. The basic *Quantize...* function shifts the start/end times of discrete audio events toward the beat grid. *Quantize Audio...* goes inside of whole events, creating beat markers from certain onsets and then shifting those closer to the beat grid, effectively quantizing the audio. The dialog has three sections.



The dialog's left section sets the beat grid interval to aim for. The options here are identical to other quantization functions, choosing between the current *Editor* grid interval or definable *Custom* grid settings.

The dialog's central section focuses on the *Onset Threshold* parameter, which limits the process to only use the strongest onsets. If an *Onset Intensity Threshold* for set for this event (see section 10.2.1.2), then that value will be used. A setting of *0 %* will use all onsets.

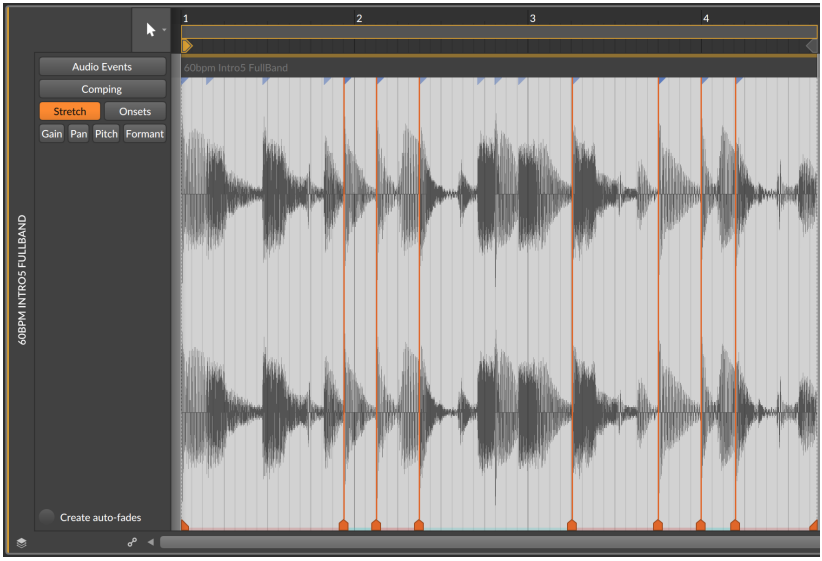
If the **Detail Editor Panel** is onscreen, it will stay bright even when the dialog is open to visualize which onsets will be used. If the dialog's *Onset Threshold* is changed — by adjusting the numeric control or dragging the vertical slider within the histogram representation — the shown/dimmed onsets in the **Detail Editor Panel** will update.



And the option to *Keep Existing Beat Markers* partially overrides the quantize process, preserving any present beat markers in their current place.

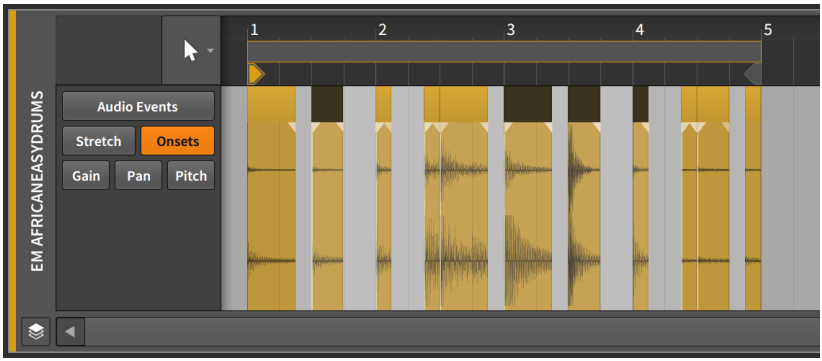
The dialog's right section offers similar *Humanize*, *Shuffle*, and *Amount* controls as other quantize functions.

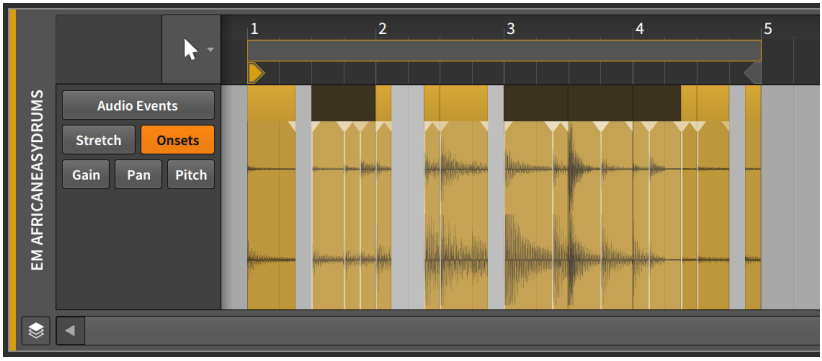
Once the function is committed via the *OK* button, an onscreen **Detail Editor Panel** will switch to the *Stretch* expression view, showing the beat markers that now exist.



› *Make Legato* adjusts the length of each selected event so that it ends immediately before the next event begins, creating a continuous series of events.

The following images demonstrate a group of selected events both before and after the *Legato* function is applied:





10.2.2. Working with Multiple Audio Events

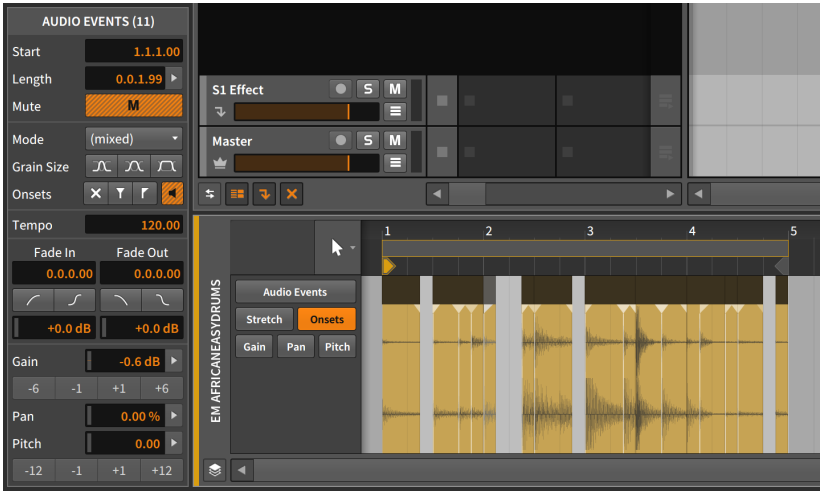
The **Inspector Panel** also works with selections of multiple events.

Functions are straightforward, as most of them listed in this chapter allow the selection of multiple events. (In the case of *Reverse Pattern*, it is not available *unless* you have multiple events selected.)

Parameters can be a little trickier when several events are selected at once. Bitwig Studio has a couple tricks of its own for both displaying and working with chunks of parameter data.

10.2.2.1. Mixed Settings

We saw expressions summarized earlier with a single average of all their points. That works well when you are dealing with numbers, but some parameters simply toggle on and off. For these discrete parameters, the **Inspector Panel** will diagonally stripe any indicator whose settings are mixed.



In the above image, the *Mute*, fade *IN*, fade *OUT*, and both of the *Onset* buttons (*Preserve* and preview) have the orange and gray striping to suggest that some of the selected events are enabled, some are not.

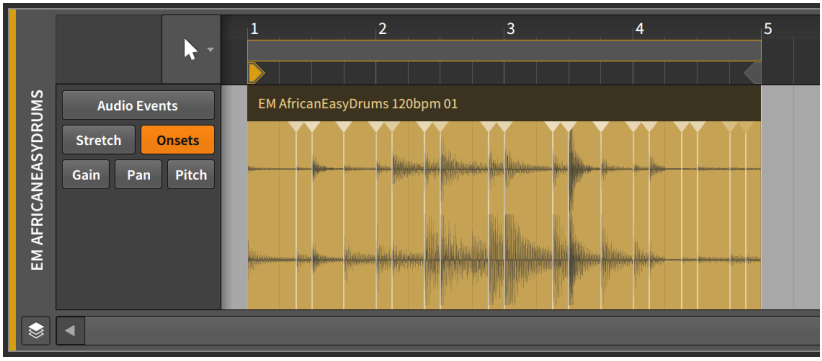
Additionally, the *Mode* menu is listed as *(mixed)*, which is its way of suggesting that not all selected events have a uniform setting.

10.2.2.2. Using the Histogram

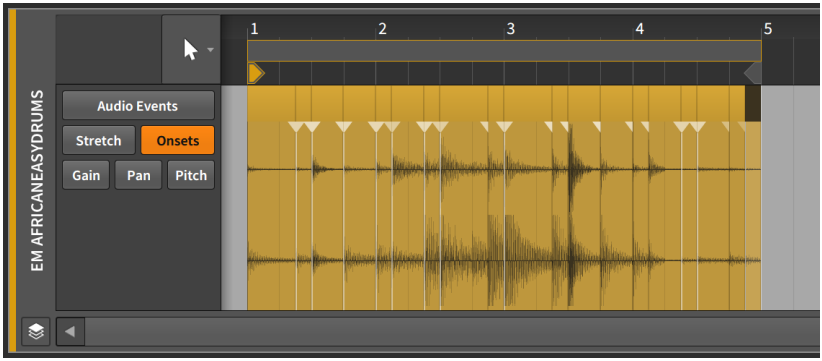
Finally, Bitwig Studio provides a special interface called the **Histogram** for working with a selection of multiple numeric values. The purpose of a histogram is to display the number of times that different possibilities occur over a span of time. In our case, the span of time being considered is the length of the current selection and the possibilities being considered are different values of the targeted parameter.

But our **Histogram** can also modify values, or even produce them from scratch. We will now demonstrate the option of creating values and then tweaking them.

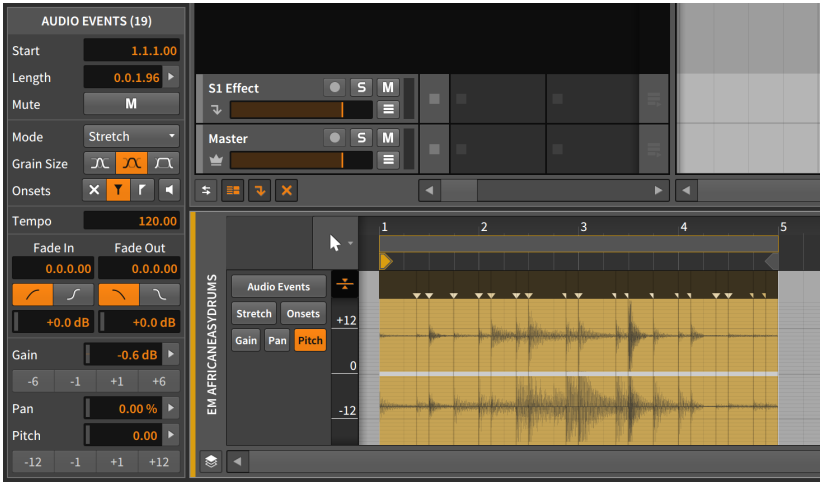
I will begin with the drum loop you have seen all across this chapter.



By applying the *Split at Onsets* function, this single event will now be divided at each onset point, giving us a collection of events that add up to the same loop.



From here, I will select all of the events. This can be done in the standard ways, by either pressing [CTRL]+[A] ([CMD]+[A] on Mac), or by choosing *Select All* either from the *Edit* menu or from the context menu. And once all events are selected, I will switch the **Detail Editor Panel** to focus on the *Pitch* expression.



A few things to note before we proceed.

First, the **Inspector Panel** now labels this section of the panel as *AUDIO EVENTS (19)*. The 19 in the title is indicating exactly how many audio events are currently selected and will be acted upon when changes are made here.

Second, the event headers are now reflecting fades where each onset point was split. This is because I have *Automatically create fades on audio clip/event edits* enabled, which is the default setting. (This preference is found in the **Dashboard**, under the *Settings* tab, on the *Behavior* page, in the *Fades* section.)

The only places where fades do not exist are at the start of the first event and at the end of the last one because no splitting occurred at these two places. And because these events lack a fade of each kind, both of the *Fades* buttons are now striped.

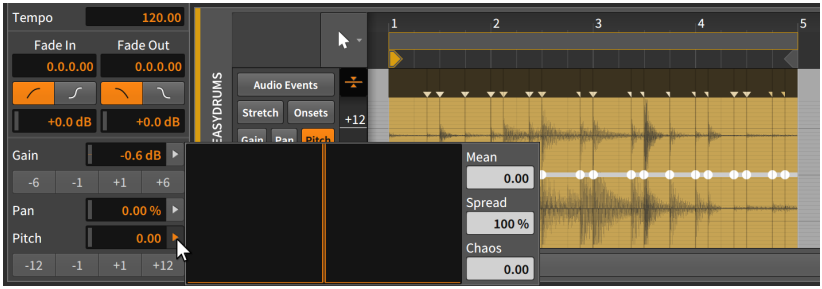
Third, in the expression section of the **Inspector Panel**, each numeric control is now followed by a right-arrow button. Since we now have multiple events selected, these arrows appear to give us access to the **Histogram**.

With these few observations made, we can now proceed.

The pitch expression is currently empty, containing no points. Now I will simply single-click on the *Pitch* parameter control. I am not changing the setting, just clicking on it once.

By just clicking on this parameter, an expression point has been created at the start of each event. So even though every point is currently set to *0.00* (semitones), we now have something to work with.

By clicking the right-arrow button beside the *Pitch* parameter, we can now see the **Histogram**.



The **Histogram** is comprised of four elements:

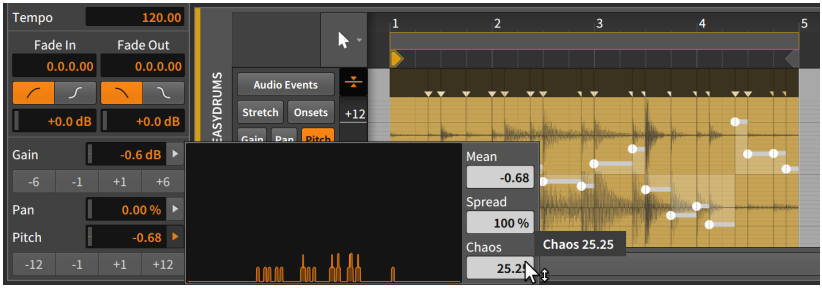
› The large *display* on the left is the actual histogram, which will present a count of the different values occurring across our selection. It is blank right now as we don't have any values yet.

› *Mean* represents the average of all selected values.

› *Spread* is a control for modifying the range of the selected values.

› *Chaos* is a control for injecting random variations to the selected values.

Adjusting the *Spread* of these points would do nothing as they are all currently identical. And adjusting the *Mean* would only adjust them all by an identical amount keeping them the same. So I will click the *Chaos* control and drag it upward.

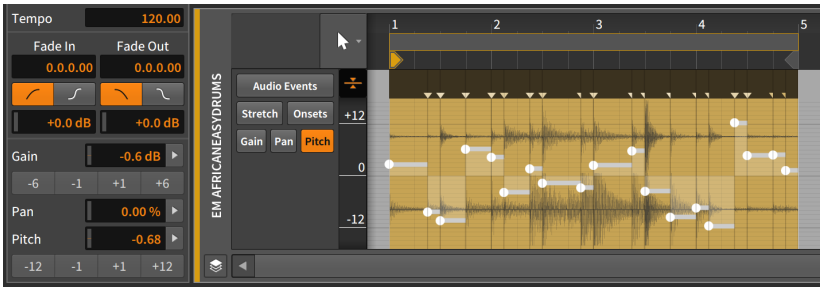


And now we have some variation in this expression.

You can see that the **Histogram** display now has some life in it. The horizontal positions are indicating the pitch values for various events — from -24 semitones on the left, to zero semitones in the middle (no pitch shift), to +24 semitones on the right. The vertical position of the chart roughly indicates the number of events found near that value.

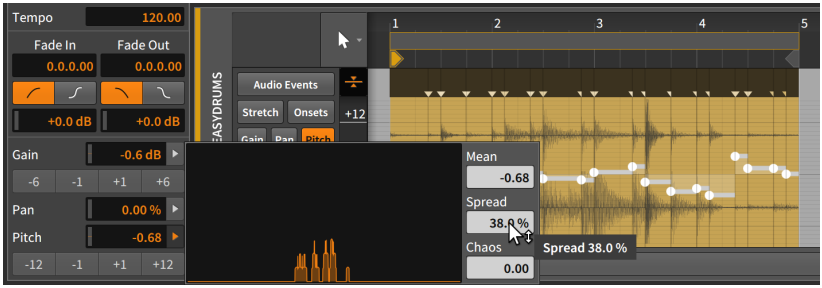
The distribution shown here is weighted toward the left (negative) side, and indeed, the *Mean* is telling us that *-1.31* semitones is the current average of all values. The **Inspector Panel** displays an identical *Pitch* value, showing that these two controls are identical.

The *Chaos* value is set in the units of the selected parameter, so it is *25.25* semitones of shift in this case. And because the pitch expression has a bipolar range, *25.25* semitones represents a distribution between -12.125 and +12.125 semitones.



Looking at the newly formed *Pitch* expression in the **Detail Editor Panel**, you can see that the highest point is right around +12 semitones (in the second audio event), and that the lowest point is right around -12 semitones (in the fourth event).

If we liked the shape of the expression but felt it was a little too extreme, we could call the **Histogram** back up and bring down the *Spread* value to narrow the overall range.

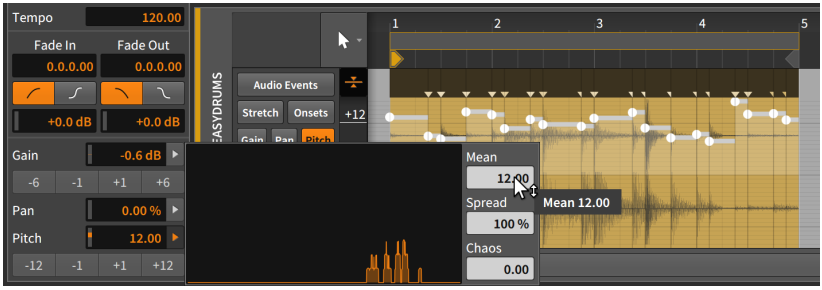


As the *Spread* value goes below *100%*, the range is indeed being shrunk, causing the histogram curve to become narrower and grow upwards — an indication that more of our 20 points are landing close to each other. But the shape of the curve is comparable to where it started.

Interestingly, the *Chaos* value was back at *0.00* when we brought the **Histogram** back up. Actually, this happened immediately after the *Chaos* setting was made and the mouse was released. And the same was true of the *Spread* function just now, as it will return to *100%* once you let go.

Each of these values represents an amount to change the current distribution of points. Unlike *Mean*, these values reflect only the future action and nothing about the present situation.

Finally, we can indeed use the *Mean* function to shift the whole expression so that zero is no longer near the center.



By moving the *Mean* to *12.00*, the average value is now a shift of one octave up with all variation landing just around that. (Again, we could have used the *Pitch* parameter to make the exact same adjustment.)

So that is a brief overview of how the **Histogram** works and an example of what you can do with it. We have spent this much time on it because the **Histogram** is available all across Bitwig Studio, whenever a group of numeric values can be selected together.

11. Working with Note Events

As we work with Bitwig Studio to assemble music, there are two forms of source material that we can use. One form is audio events, which was covered thoroughly in the last chapter. The other is *note events* — or simply *notes* — which we will investigate in this chapter.

As the introduction to the last chapter suggested, these two chapters are really parts one and two of working with the contents of clips. Accordingly, the format of this chapter is highly similar to the previous one, with many of the same issues and concerns being presented from the perspective of notes. And consistent with the rest of this document, ideas that reappear will reference the section where they were first discussed.

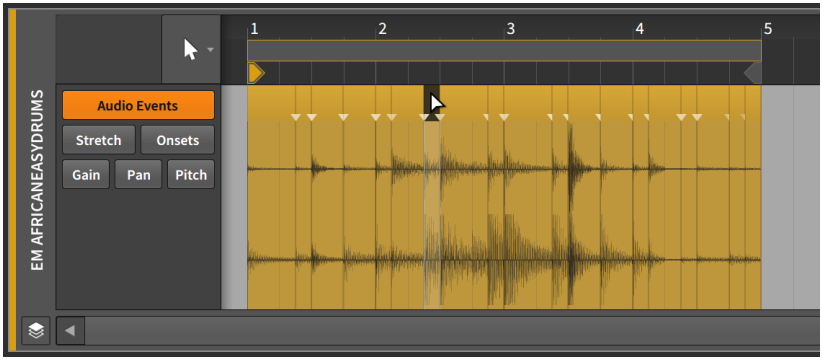
We will begin by revisiting the **Detail Editor Panel** to see how it works with note events, as well as the vast per-note modulation capabilities of Bitwig Studio. We then will see the last face of this panel as it allows us to work with multiple clips and tracks simultaneously. And after revisiting the **Inspector Panel** in the context of notes, we will take a look at the **Edit View**, the third and final panel set.

Let's sharpen our tools for working with that other type of musical content: *note events*.

11.1. The Detail Editor Panel, Note Clip Edition

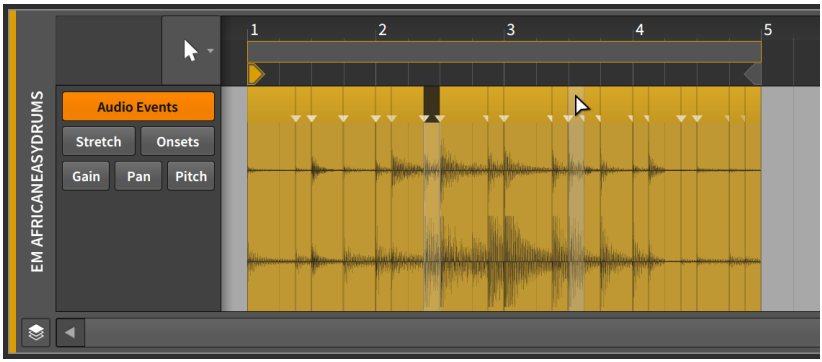
The utility of the **Detail Editor Panel** should be clear by now, but the truth is that we have covered only half of it at best. We will start again with this panel because when it is focused on note clips, the same **Detail Editor Panel** adapts and provides slightly different options that are appropriate to the situation.

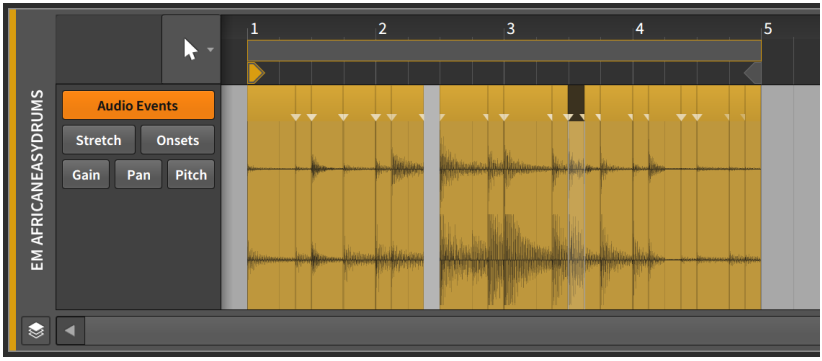
To better understand the incarnations of the **Detail Editor Panel**, let's take a moment to differentiate the structure of audio events and notes. (They are clearly made of different materials, but the way they are stored and structured is critical here.) The most important distinction is that while audio events are all of one kind, note events have pitches that allow us to distinguish them and make them overlap.



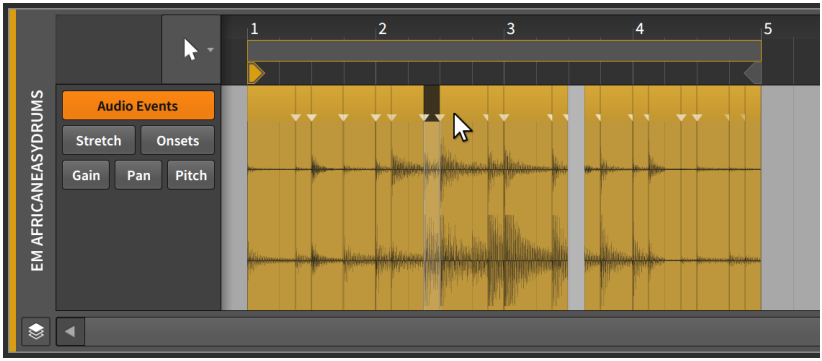
Only one audio event can occur at a time within a single clip, so while audio events can be arranged sequentially, they cannot be played simultaneously. And because no audio event has inherent priority over another, the last event placed in a certain position will "win."

If you move an audio event to a position already occupied by another event, the new event will effectively clear the position that it now occupies, leaving behind no trace of what was here.

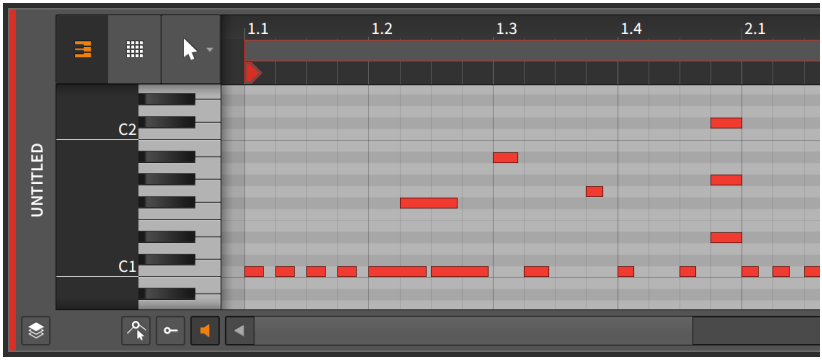




This is because audio events cannot coexist. (Clips of all kinds behave in exactly the same fashion.) To illustrate this, moving the new event back to its original position will leave a hole where you had placed it.



The most important characteristic of each note is its *pitch*. This characteristic immediately gives us a way to distinguish notes from one another. And once we can distinguish notes by type, we can now have overlapping notes.

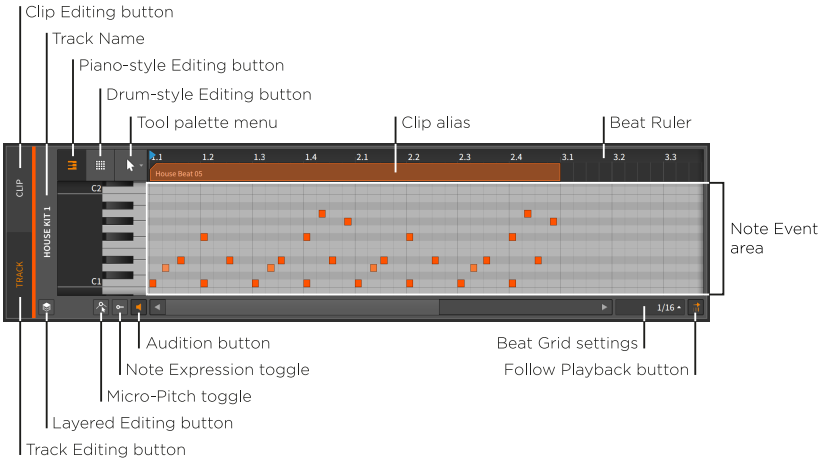


Chords and other overlapping gestures are a part of music, and note clips support them by allowing notes of different pitches to overlap. So while audio events are the smallest workable unit (and have their own headers to work with them), individual notes are the fundamental units here.

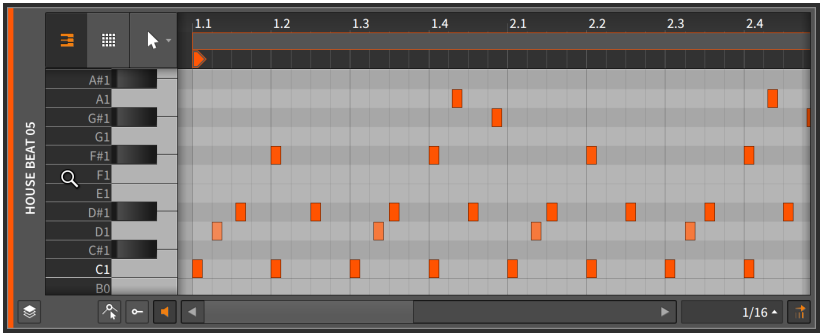
We will discuss the many similarities between how audio events and notes are edited. And they start in the **Detail Editor Panel**.

11.1.1. Layout of the Detail Editor Panel

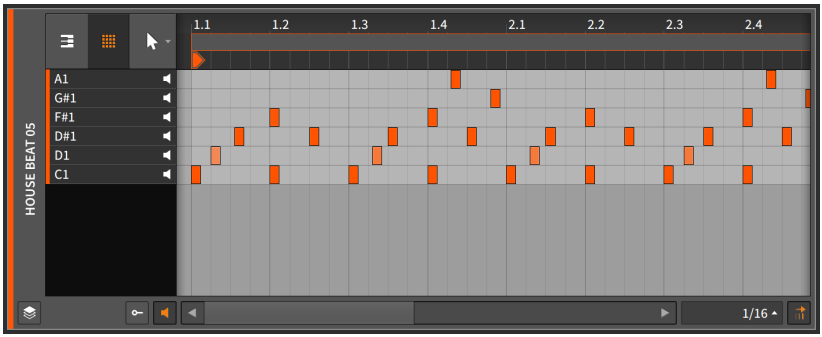
Double-clicking a note clip in either the **Clip Launcher Panel** or the Arranger Timeline will call up the **Detail Editor Panel** and place its focus on that clip.



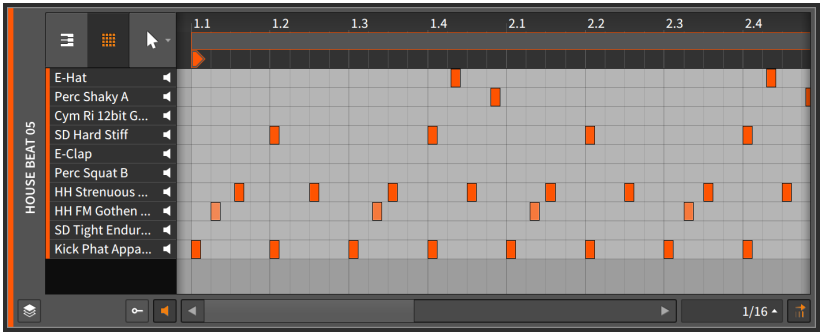
Much of this is familiar, such as the *Beat Ruler* (see section 3.1.1), the *clip aliases* (see section 9.2.1), and the *Clip Editing button* (see section 9.2.2), as well as this panel's own *beat grid settings* (see section 3.1.2), *snapping settings* (see section 5.1.2), and *Follow Playback button* (see section 3.1.4). The panel itself can still be vertically resized, but the y-axis can also be zoomed by clicking and dragging in the dark gray field just to the left of the piano keyboard.



The view we are seeing above is the *piano-style* view. Clicking the drum pads icon will switch to a *drum-style* editor. For nearly all instruments, only notes which are used on the current track (while in *track editing mode*) or for the current clip (while in *clip editing mode*) will be shown.



If the track's primary instrument is **Drum Machine**, then all notes with populated drum cells will be shown.



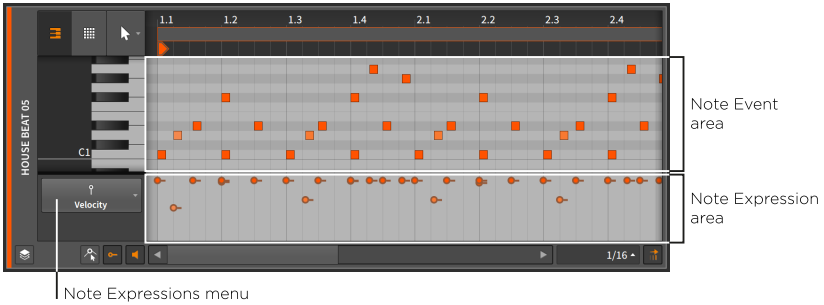
In either case, everything else about the panel continues to work as usual.

Three other new buttons have also appeared in the bottom left corner of the **Detail Editor Panel**.

› When the *Audition button* is enabled, clicking and dragging any note to a new pitch will send a corresponding note to the track's device chain. This provides an audible preview of the action being considered.

Additionally, clicking the piano keyboard to the left of the *note event area* will trigger a note when the Audition button is enabled.

› When the *Note Expression toggle* is enabled, the *Note Expression area* becomes visible below the Note Event area.



› When the *Micro-pitch toggle* is enabled, *Micro-pitch editing mode* will be active (see section 11.1.3). Note that this button and mode are only available with the piano-style editor.

11.1.1.1. Drawing Notes and Quick Draw

In addition to recording or importing note clips, you can also draw notes into a clip from the **Detail Editor Panel**.

To draw an individual note within a note clip: either double-click while the Pointer tool is selected, or switch to the Pen tool and then single-click within a note clip.

Notes will be given a velocity of *78.7 %* (the equivalent of *100* out of 127) and a length of the beat grid value. You can also adjust these values while drawing each note.

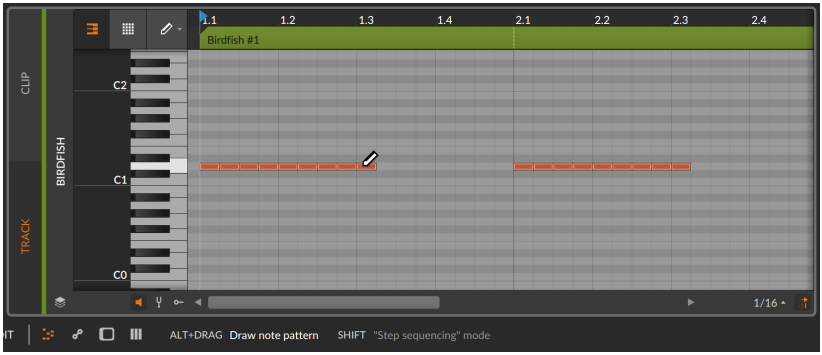
To set velocity while drawing a note: continue to hold the mouse down, and then drag up or down to adjust the velocity.

To set note length while drawing a note: continue to hold the mouse down, and then drag left or right to shorten or lengthen the note.

After drawing a note with an adjusted velocity or note length, these values will be the new defaults for notes drawn into this particular clip.

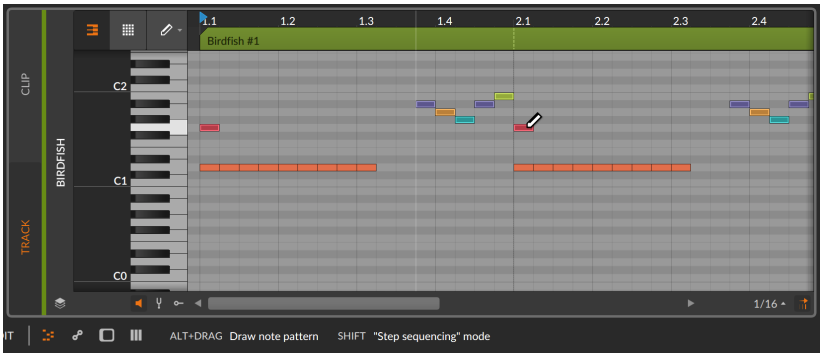
Quick Draw is a feature that allows you to draw multiple notes at once. The requires the Pen tool to be selected.

To draw successive notes within a note clip: hold [ALT], and then click at the position of the first note and drag to the position of the last note.



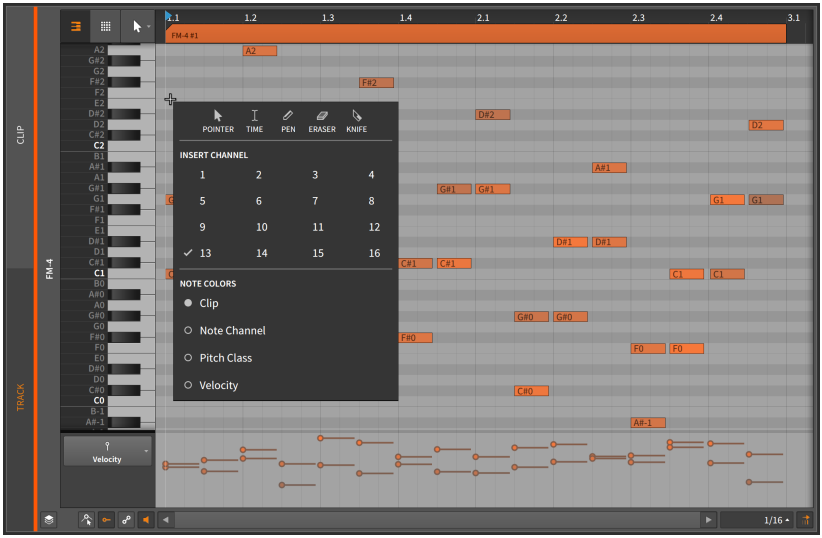
The current beat grid value (*1/16* notes, above) will set the length of each note and quantized start position of the series. And again, dragging up and down will adjust the velocity used for all notes. If you would prefer instead to draw notes on different pitches (kind of like step sequencing pitches), you can do that too.

To draw successive notes with different pitches within a note clip: hold [ALT] and click to initiate **Quick Draw** mode. Then add the [SHIFT] key to free the pitches being drawn.

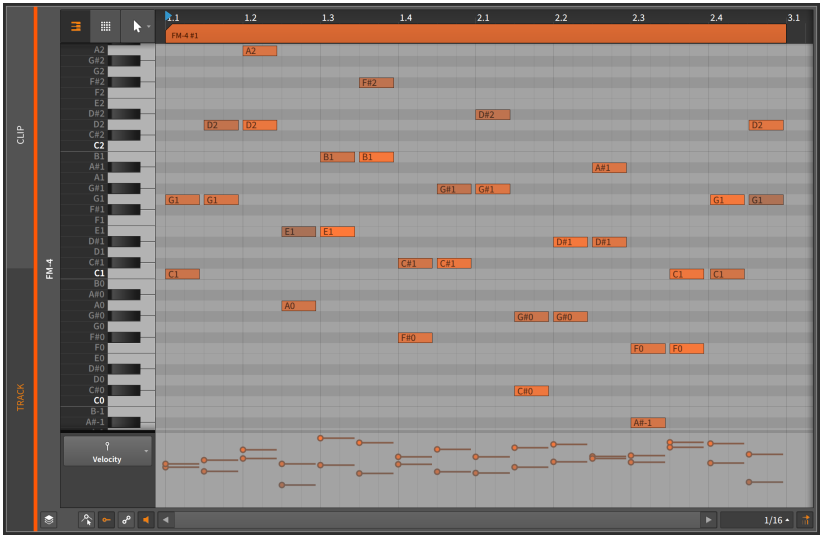


11.1.1.2. Note Color Options

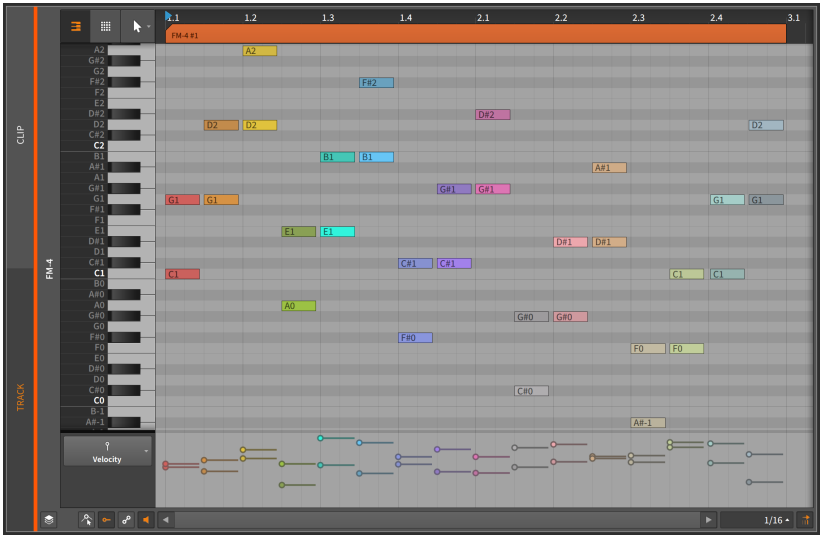
When dealing with notes in the **Detail Editor Panel**, various options for how notes are colored are available from the panel's context menu. Right-click a blank area of the editor to see the *NOTE COLORS* options.



› *Clip* uses the color of the parent clip for each note, and the velocity of each note scales the relative saturation.

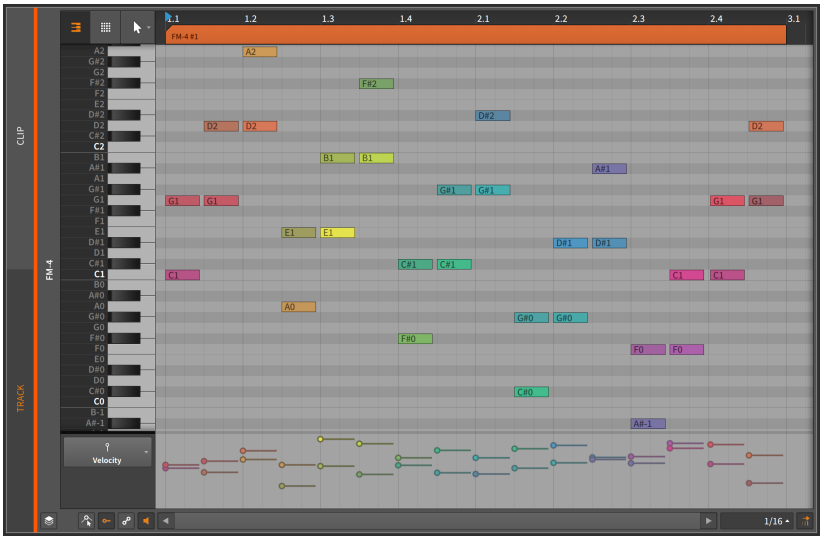


› *Note Channel* colors each note by the channel it is on, and the velocity of each note scales the relative saturation. To illustrate this range in the image below, the chords shown are spread across all 16 channels with the chord on the far left on channel 1 and that at the far right on channel 16.



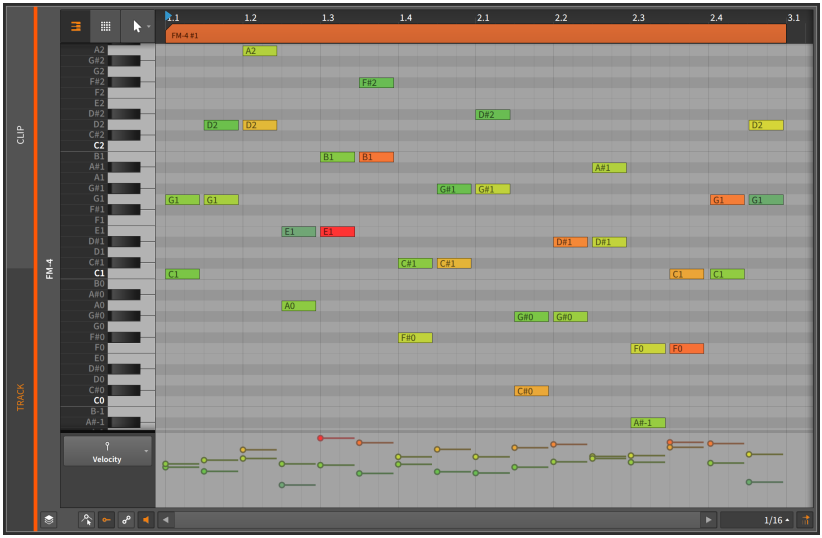
› *Pitch Class* colors each note by its pitch class (for example, all *C*s are treating the same, as are all *C#*s, *D*s, etc.) and the velocity of each note scales the relative saturation. The colors are based on the musical circle of fifths, with harmonically-related intervals colored similarly and more dissonant intervals using contrasting colors.

To illustrate this range in the image below, each chord is a succession of fifths (*C*-*G*, then *G*-*D*, then *D*-*A*, and so on). This shows how consonant intervals, such as fifths, looks alike and how more tense intervals, such as half-steps (for example, *G* to *G#*) and tritones (*C#* to *F*), contrast strongly.



› *Velocity* colors each note exclusively by its velocity. This provides a clear contrast, particularly when doing detail work on your notes.

The range used is similar to a level meter on a mixing board, with the velocities progressing from pale green to solid green, then yellow, orange, and eventually red. The range of velocities in the image below help illustrate this.



11.1.2. Note Event Expressions

Like audio event expressions, *note expressions* are parameters that can be set for each individual note. Many of these parameters can change over the course of the note, making them like specialized automation curves.

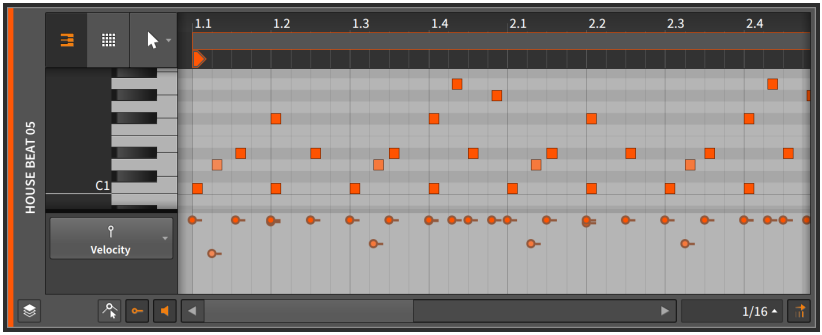
Only one note expression can be focused on at a time, and you pick which expression to view by clicking its name in the list. We will take them from top to bottom.

Note

For all of the note expressions including Micro-pitch (see section 11.1.3), *Spread* is available for each expression point (see section 10.1.3).

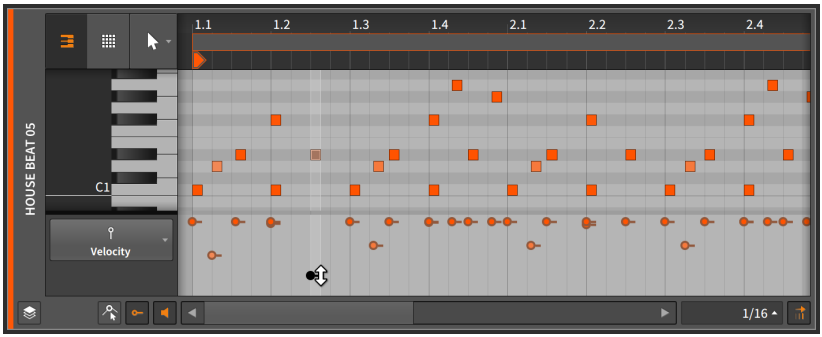
11.1.2.1. Velocity Expressions

Velocity expressions represent the strength with which each note should be triggered.



Similar to the MIDI specification, a *velocity expression* consists of a single value that is transmitted at the note's start. Each device determines how velocity will be used. Any device or plug-in can use the **Expressions** modulator device to route velocity expressions. See section 16.2.1 for information on using the modulator devices and section 19.27.6.2 for more on the **Expressions** device.

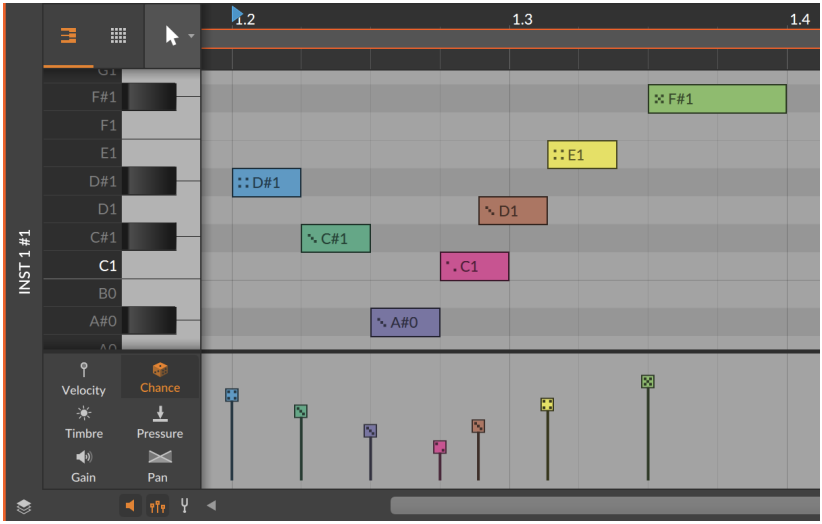
To adjust a velocity expression: mouse over the velocity expression so that a double-arrow cursor appears. Then click and drag the expression vertically.



Notes are colored to match their clip's color, with the saturation of each note set relative to the strength of the note's velocity. A note at full velocity (*100 %*) will be shown as the full color of the clip. As a velocity lowers, the color of that note will change.

11.1.2.2. Chance Expressions

Chance expressions represent the likelihood that any note will be played (see section 12.1.1).

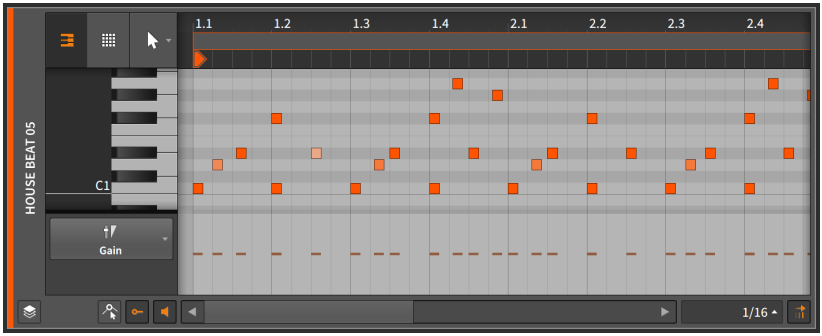


Like Velocity, Chance is only set at the beginning of each event.

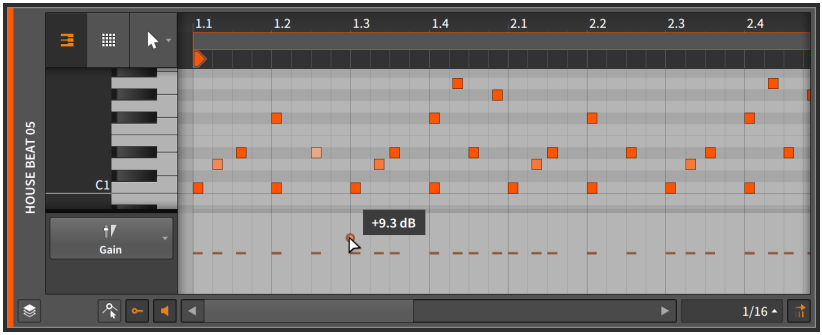
All other expressions can be programmed like automation across the length of each note, which we will look at next.

11.1.2.3. Gain Expressions

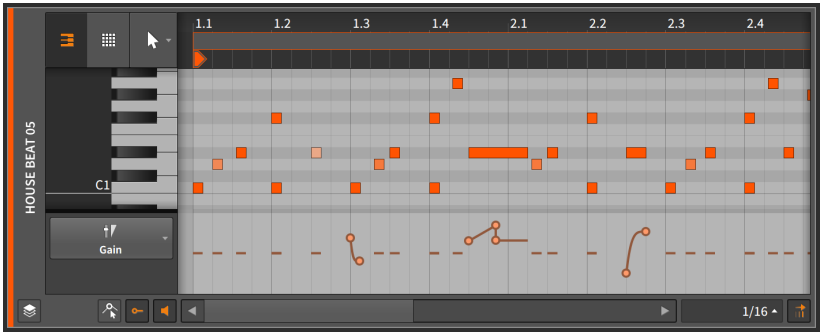
Gain expressions represent a level control for each note event.



To start with, each note's expression contains no individual points. By initially clicking and dragging an expression, you are both creating an initial point within the expression and defining the entire expression's value.



Once an initial point has been defined, additional expression points can be created and edited in the same way that automation points are (see section 9.1.2).

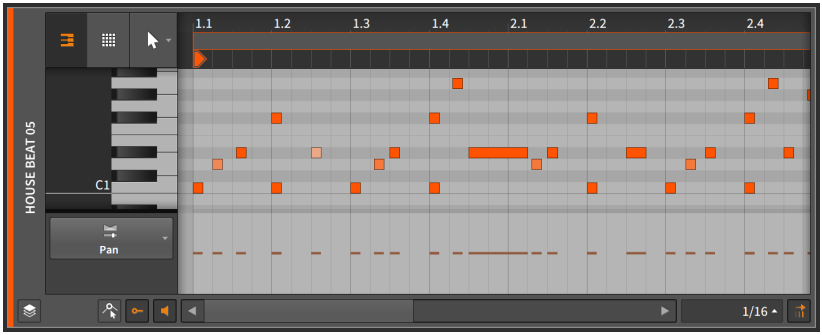


A *gain expression* is measured in units of decibels with the center line representing zero decibels of change (unity gain).

A gain expression is identical in function to volume automation. The difference is that the expression is applied at the beginning of the audio signal path — in this case, at the output of the instrument device (pre-FX Chain) that initially synthesizes audio signal. Volume automation is applied as the last stage of a track's signal flow (after the track's device chain and everything else).

11.1.2.4. Pan Expressions

Pan expressions represent a stereo placement control for each note event.



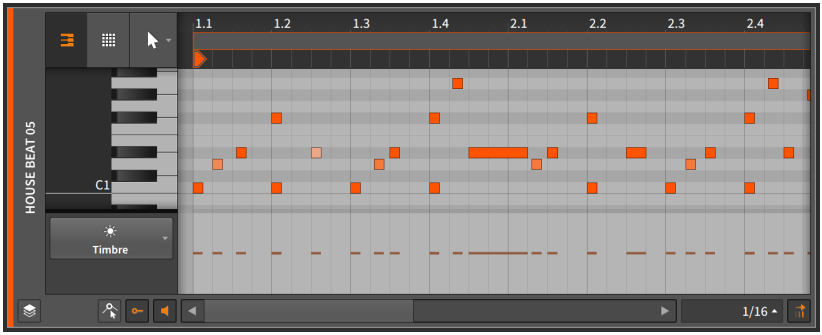
Once an initial point has been defined, additional expression points can be created and edited in the same way that automation points are (see section 9.1.2).

A pan expression is measured as a bipolar percentage with the center line at *0.00 %* (center placement, or no panning adjustment), *100 %* for hard right, and *-100 %* for hard left.

As with the gain expression, the *pan expression* is often applied at the beginning of the audio signal path. The pan expression has no direct interaction with *pan automation*, which is applied by the track mixer after the device chain.

11.1.2.5. Timbre Expressions

Timbre expressions represent an assignable modulation source for each note event.



Once an initial point has been defined, additional expression points can be created and edited in the same way that automation points are (see section 9.1.2).

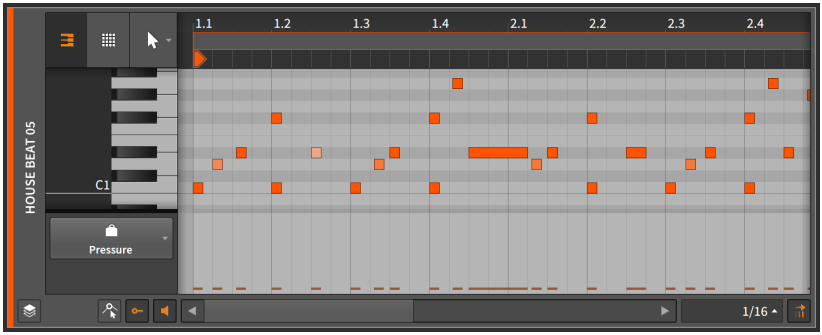
The word *timbre* refers to a sound's tone color, but the timbre expression here has no fixed purpose. Rather, it can be used to freely modulate one or more parameters of the track's instrument device (see section 16.2). Mapping is done with the *TMB* modulation source, which is available on any device or plug-in via the **Expressions** modulator device. See section 16.2.1 for information on using the modulator devices and section 19.27.6.2 for more on the **Expressions** device.

A timbre expression is measured as a bipolar percentage with the center line at *0.00 %* and the extremes at values of *100 %* and *-100 %*.

Similar to the gain and pan expressions, the *timbre expression* is often applied within the instrument at the beginning of the audio signal path.

11.1.2.6. Pressure Expressions

Pressure expressions represent an assignable modulation source for each note event.



Once an initial point has been defined, additional expression points can be created and edited in the same way that automation points are (see section 9.1.2).

As the word *pressure* suggests, this expression is similar to the idea of *polyphonic key pressure* (or *aftertouch*) from MIDI. But the pressure expression here has no fixed purpose. Rather, it can be used to freely modulate one or more parameters of the track's instrument device (see section 16.2). Mapping is done with the *PRES* modulation source, which is available on any device or plug-in via the **Expressions** modulator device. See section 16.2.1 for information on using the modulator devices and section 19.27.6.2 for more on the **Expressions** device.

When working with external MIDI via the **HW Instrument** device (see section 19.11.5), any pressure expressions are directly transmitted as polyphonic key pressure MIDI messages.

A pressure expression is measured as a percentage with default values set at *0.00 %* and a maximum level of *100 %*.

Similar to the gain, pan, and timbre expressions, the *pressure expression* is often applied within the instrument at the beginning of the audio signal path.

11.1.3. Micro-pitch Editing Mode

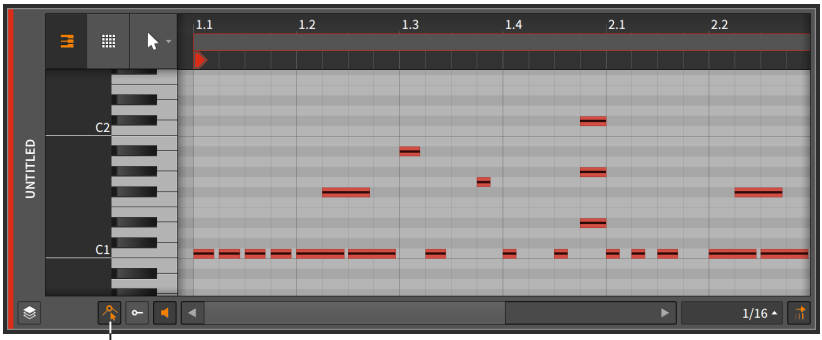
When working with notes, the **Detail Editor Panel** appears as a standard "piano roll" editor, with notes placed on their vertical pitch at the appropriate horizontal time. The notes can be created and edited in the exact same fashion as clips are (see section 5.1.1, section 5.1.2, and section 5.1.3).

By default, the **Detail Editor Panel** works with notes in the standard, discrete semitone fashion. But by enabling the *Micro-pitch toggle*, we enter *Micro-pitch editing mode*.

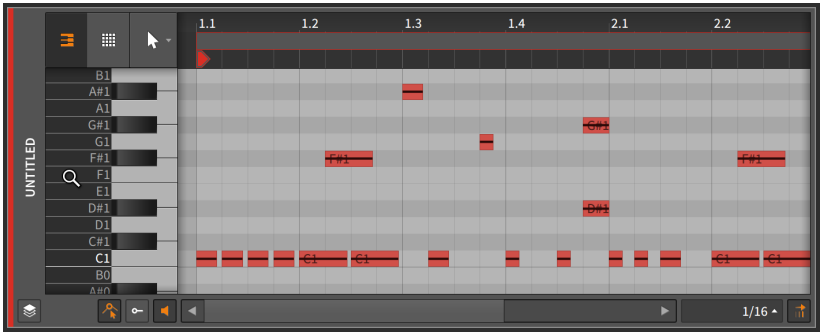
Note

Micro-pitch editing relies on Bitwig Studio's unique per-note modulation capabilities. Micro-pitch expressions will function properly with Bitwig's instrument devices, and they can work with CLAP plug-ins as well.

Micro-pitch editing mode is not available while the Fold Notes button is enabled.



Thin lines are now drawn across the center of each note event. We can zoom in to make this easier to work with.



These lines are *Micro-pitch expressions*. Like all other note expressions, Micro-pitch expressions are per-note events, allowing the specific pitch of each note to be set precisely, or even to change the pitch of the note while it is played. You can think of Micro-pitch expressions as a precise, polyphonic version of MIDI pitch bend, where each note played has its own pitch curve.

Micro-pitch expressions are measured in semitones, with the center line at *0.00* (for no pitch shift), a maximum of *24.00* (two octaves up), and a minimum of *-24.00* (two octaves down).

Just a few examples of how this might be used:

› Building a chord with one of its notes bent while all others are held steady.

› Shaping a lead line with graceful transitions, where each note fades (perhaps with a gain expression) while gliding to the pitch where the next note will begin one.

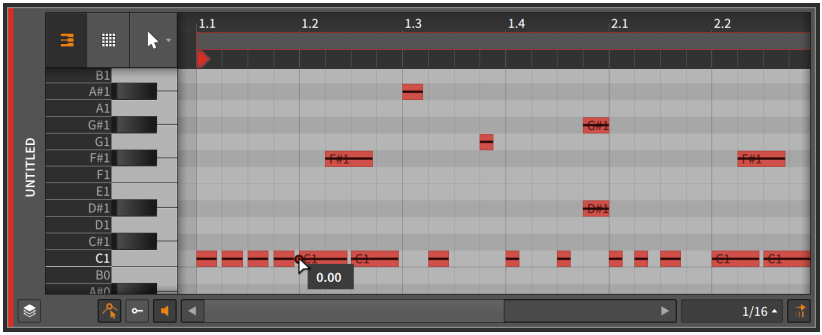
› Carving out a solo, where the shape of the vibrato is precisely drawn.

› Structuring a microtonal part, where each note's pitch is meticulously defined.

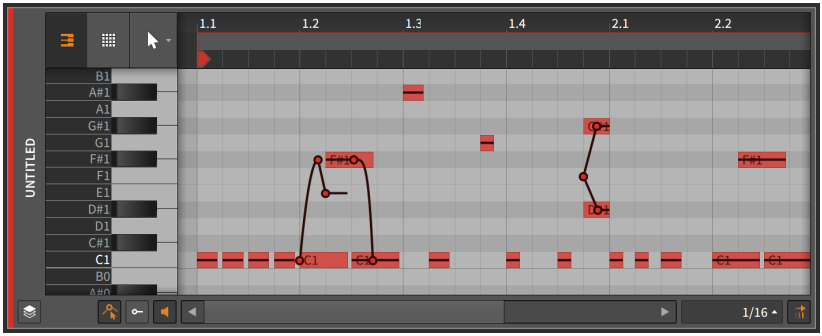
› Creating a part that combines any of these ideas, or something else altogether.

Like the other note expressions that can be automated, each Micro-pitch expression is blank to begin with. The centered line represents that the note is tuned only by its standard pitch assignment.

By initially clicking and dragging the Micro-pitch expression, you are both creating an initial point within the expression and defining the entire expression's value. In most cases, you will want to single-click the expression to start.



Once an initial point has been defined, additional Micro-pitch expression points can be created and edited in the same way that automation points are (see section 9.1.2).



The *semitone snapping* option causes Micro-pitch expression points to snap to whole number semitones. As with the position snapping options (see section 5.1.2), holding [SHIFT] will toggle this behavior. Semitone snapping is enabled by default.

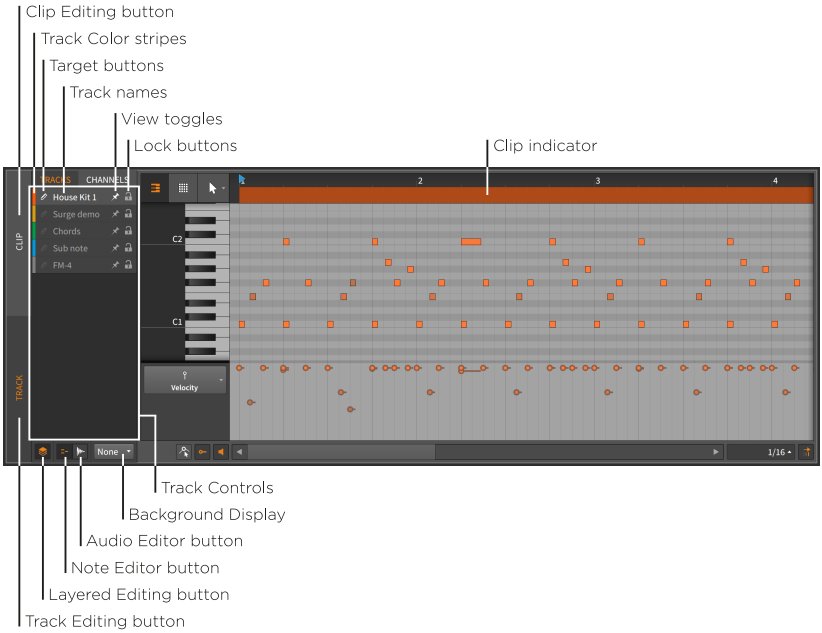
11.1.4. Layered Editing Mode

We have seen the **Detail Editor Panel** work at various levels. We examined the panel while it focused on a single clip at a time in *clip editing mode*. We have also (and primarily) explored the panel while it focused on all contents of a track in *track editing mode*. And now there is one, larger level left to explore.

Layered editing mode still has a *clip editing button* for letting us toggle between clip or track editing mode. But once we have chosen that

mode, entering layered editing mode allows us to view and edit several clips or tracks together. So once we pick the clip or track paradigm, we can then zoom out and work with several of those side by side.

We enter layered editing mode by enabling the *Layered Editing button*.



In the image above, we are in track editing mode, as set by the vertical *track editing button*.

Note

In the image above, the button at the top of the left column labeled *TRACKS* must also be selected. This indicates that layers are being shown by track content, and reads *CLIPS* when the *clip editing button* is enabled instead.

The alternate option, *CHANNELS*, is available when editing notes show layers by note channel (see section 11.1.4.3).

When we were previously in track editing mode within the **Detail Editor Panel**, the top of the panel displayed a *clip alias*. While track editing in layered editing mode, we now have a *clip indicator* instead. This indicator still shows us the start and end times of displayed clips, but

the clip's name is no longer present and its length and position can no longer be manipulated.

Other than that, the right side of the panel is unchanged. The left side of the panel, however, contains several new items.

On the top left edge of the **Detail Editor Panel** are two buttons — the Track Editing button and the Clip Editing button — which are already familiar. And if as in the image above the Clip Editing button is disabled, below it will be two new buttons that form a toggle pair.

If the *Note Editor button* is enabled, the **Detail Editor Panel** will focus on note containers as we have examined in this chapter. If the *Audio Editor button* is enabled, the **Detail Editor Panel** will focus on audio containers as we examined in the previous chapter. Only one of these can be enabled at a time so clicking either button toggles the current selection.

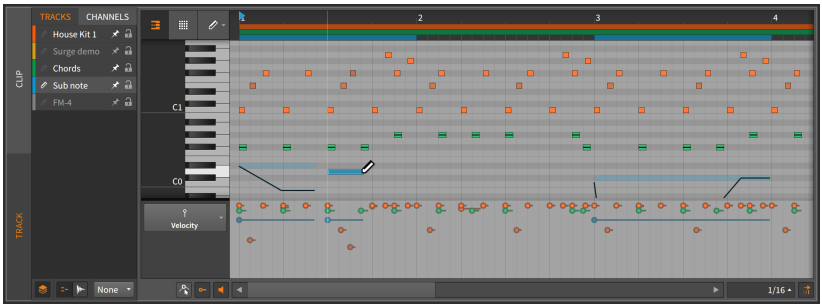
Taking all this together, we must select whether we want to use clip or track editing mode, and also choose whether we want to work with note or audio clips. For the current example, we will continue with note clips in track editing mode.

11.1.4.1. Layered Editing in Track Mode

Now that our modes are set, the resizable *track controls* section houses editor parameters for each instrument and hybrid track in the current project. These controls include:

› *Track Color stripe*: A swatch of the track's assigned color.

› *Target button*: A pencil icon that sets this track as the *target layer*, making it the destination for newly drawn or pasted notes. Also note that clicking on a layer's name or editing its contents will make that layer the target layer.



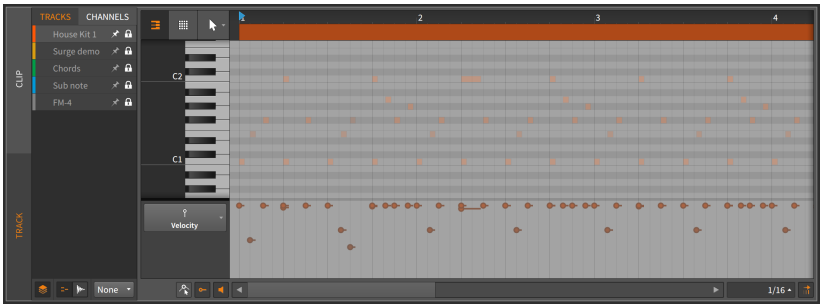
Note

By right-clicking anywhere in the track controls area and toggling on the *Selected Layer Becomes Target Layer* option, clicking a layer's header will no longer make it the target layer.

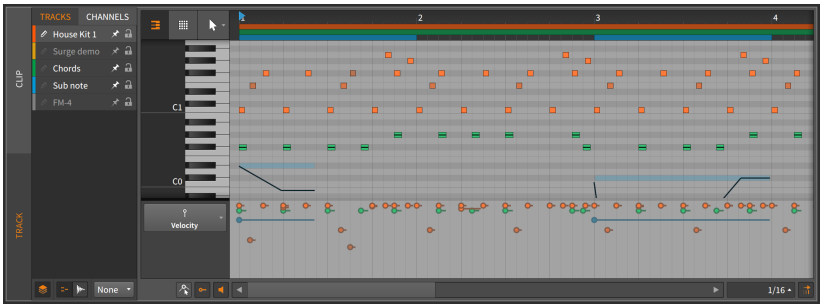
› *Track Name*: The title assigned to the track.

› *View toggle*: This thumbtack icon keeps the layer visible, even when it is not selected.

› *Lock button*: When enabled, the layer's data is protected from being selected or altered. When a locked track is visible, its contents are still shown but significantly dimmed.



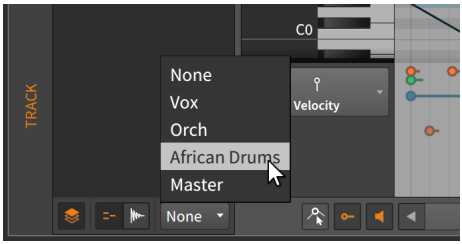
To make a layer visible: either select it or enable its view toggle.

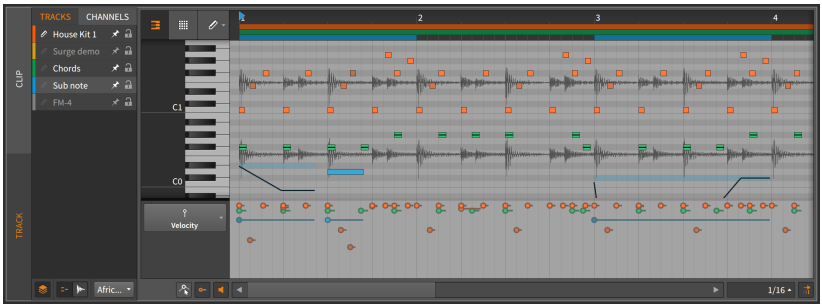


All aspects of unlocked visible tracks are editable with the techniques we have seen. Data from various tracks can also be edited together in this fashion, and objects can even be placed in relation to one another with *object snapping* (see section 5.1.2).

Any clip indicators for the target track will also shade the note event area to indicate both the boundaries you are working within and how those boundaries might change by moving notes into empty space.

While in the Note Editor, the *background display* setting is the final interface item. The menu labeled *Background* appears below the track controls and allows you to pick a background for display behind the note event area. The choices are either *None* (for no background) or any of the audio or hybrid tracks in the current project.





This setting is purely visual but can serve as a helpful reference.

11.1.4.2. Layered Editing in Clip Mode

Switching from track editing mode to clip editing mode presents a few structural differences.

Again, the right side of the **Detail Editor Panel** is largely unchanged from its standard clip editing mode layout.

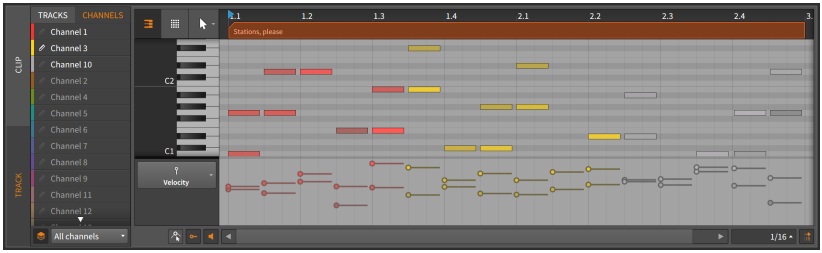
On the left side of the panel, the track controls have been replaced by *clip controls*. The primary difference here is that only clips which are currently selected in the active sequencer (either the Arranger Timeline or the **Clip Launcher Panel**) will be shown as options.

Because your selection is made in the sequencer, no view toggles are needed. Also the Note Editor and Audio Editor buttons will appear only when both clip types are selected.

Otherwise, this configuration works as expected.

11.1.4.3. Layered Editing by Channel

When working with note events, you can also layer them by their *channel* for editing purposes.



Note that the vertical track and clip editing buttons are still present on the far left, allowing you to specify whether you are viewing one entire track or one clip at a time.

The interface itself is mostly a streamlined version of what we have seen already. The layers here are listed by channel, with used channels shown at the top of the list in a bright white. The only real difference is

the menu of editing/display modes, just below the layer listing. Options include:

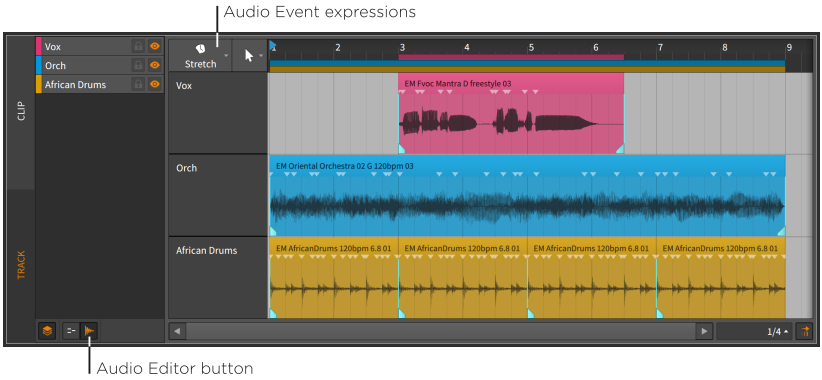
› *All channels* displays all notes, keeping them all editable.

› *Selected channels* allows only selected channels to be edited. Non-selected channels are still displayed but greatly dimmed.

› *Selected channels (hide others)* allows only selected channels to be edited. Non-selected channels hidden.

11.1.4.4. Layered Editing with the Audio Editor

Switching from the Note Editor to the Audio Editor also presents a few structural differences.



In track editing mode, audio events can be freely worked with as described in the previous chapter. In clip editing mode, both audio events and clips can be worked with.

Audio expressions can also be worked with in both modes. A single *audio event expression* menu appears above the track headers to determine which expression is globally displayed.

And again, events and/or expressions can even be set in relation to one another with *object snapping* (see section 5.1.2).

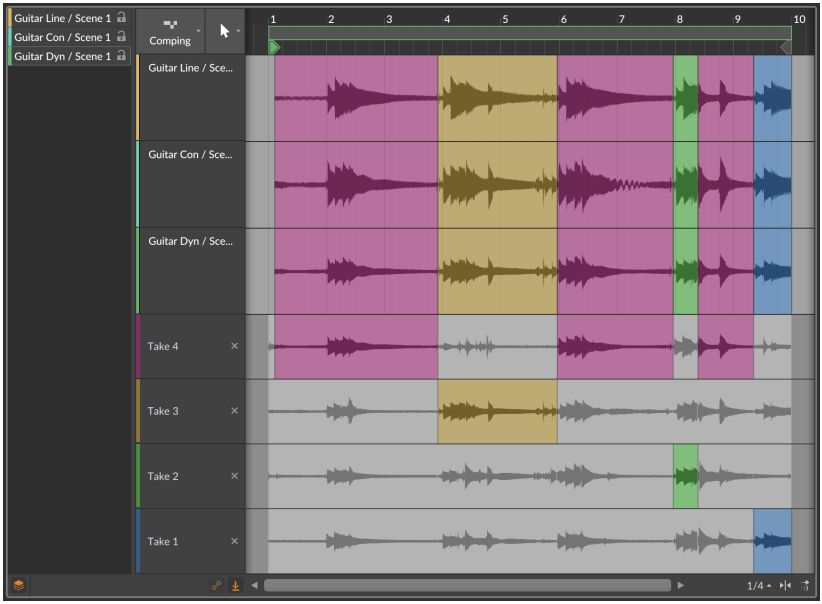
The last new interface option is the *Lane Resize toggle*. When enabled, resizing the **Detail Editor Panel** also tries to resize each individual track/clip lane in order to fit the available space.

Otherwise, this editor works as expected.

11.1.5. Layered Comping

Layered editing mode also serves as a way to perform *layered comping*, or to use all comp editing gestures (see section 10.1.4) on multiple comps at once. This is ideal for comps that were recorded simultaneously, but it can work on other material of similar length and configuration.

To work in layered comping mode: select multiple clips that contain comping data, then open the **Detail Editor Panel** and click the layered editing button.



All composite tracks will be shown up top (three in this case), with the takes visible for only one of the comps.

To edit only one comp while in layered editing mode: hold [CTRL] ([CMD] on Mac) and start your edits on the desired comps.

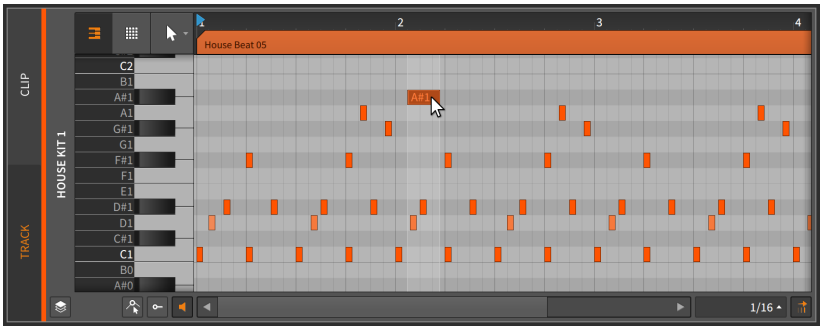
11.2. Inspecting Note Clips

As with audio events, the **Inspector Panel** is a critical way to both access the details of note events and edit them most effectively. To focus the

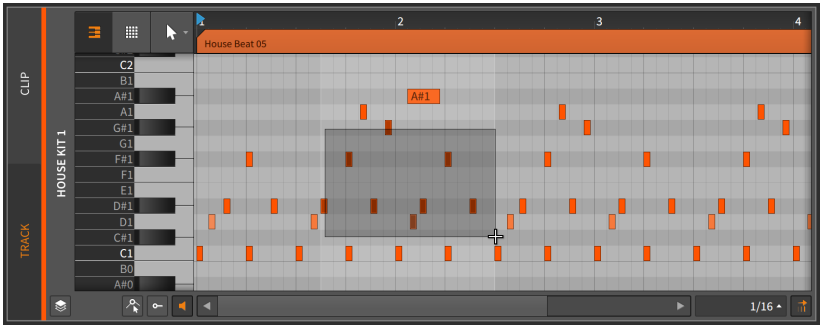
Inspector Panel on notes, we must first select them within the **Detail Editor Panel**.

11.2.1. Selecting Notes

To select a single note: single-click it.



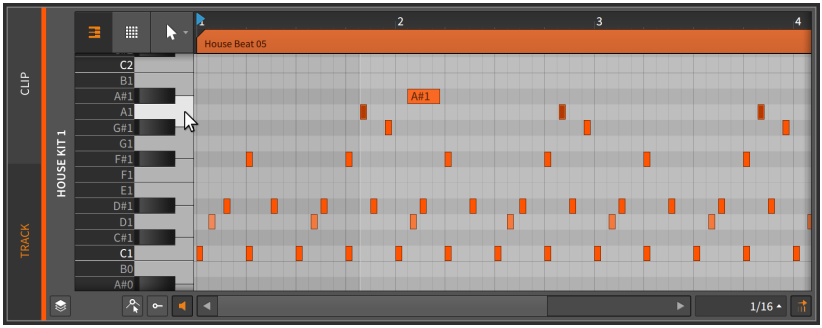
To select multiple notes: click a blank area and drag a rectangle around the desired notes.



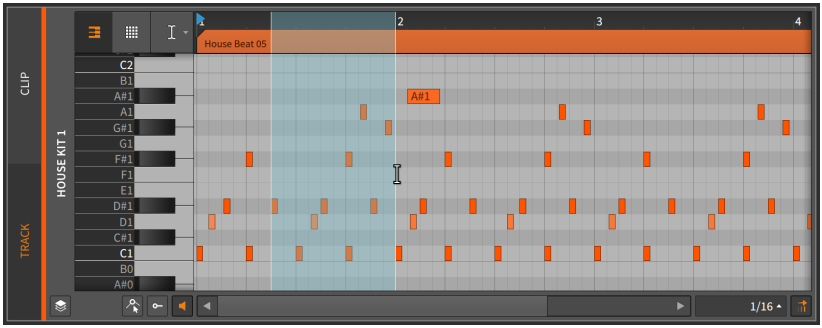
Other ways to select multiple notes include:

› After selecting one note, [CTRL]-click ([CMD]-click on Mac) additional notes to grow the selection.

› Click a note on the piano keyboard to select all displayed notes of that pitch.



› With the Time Selection tool, click and drag over the time area for which all displayed notes should be selected.



(To normally click and drag the notes after they are selected in this way, you can switch back to the Pointer tool.)

To select the next note: press [ALT]+[RIGHT ARROW].

To select the previous note: press [ALT]+[LEFT ARROW].

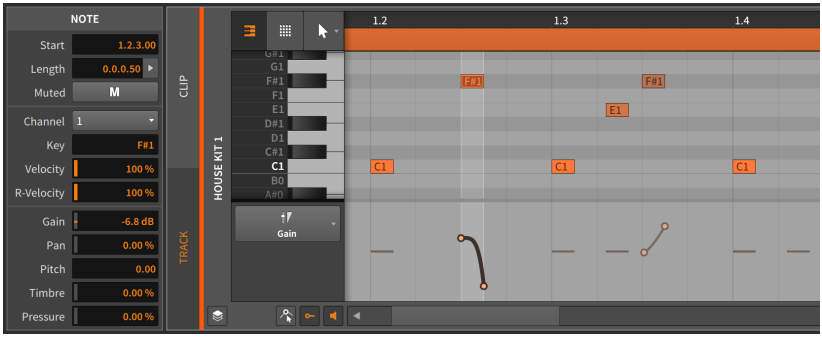
If you have one note selected, you can similarly grow the selection by pressing [SHIFT]+[ALT]+[RIGHT ARROW] or [SHIFT]+[ALT]+[LEFT ARROW] .

Once a note selection is made, the **Inspector Panel** will display relevant settings and functions.

11.2.2. The Inspector Panel on Note Events

As with audio clips and events, selecting a note clip makes certain parameters and functions available in the *NOTE* section of the **Inspector**

Panel, but by selecting a note event itself, the **Inspector Panel** provides all settings relevant for the selected event(s).

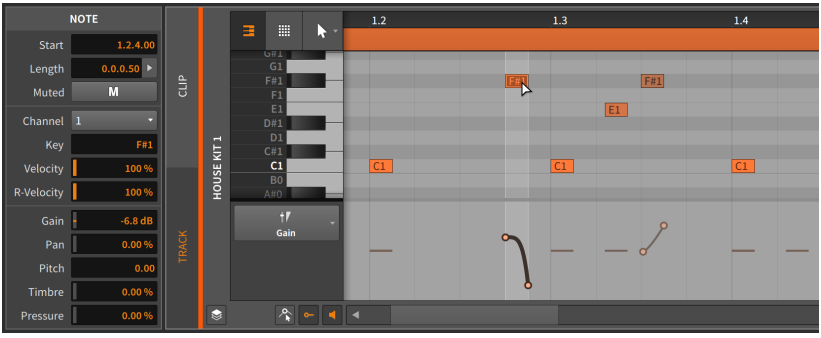


We will take these one section at a time and also examine the functions available in the *Event* menu when note events are selected.

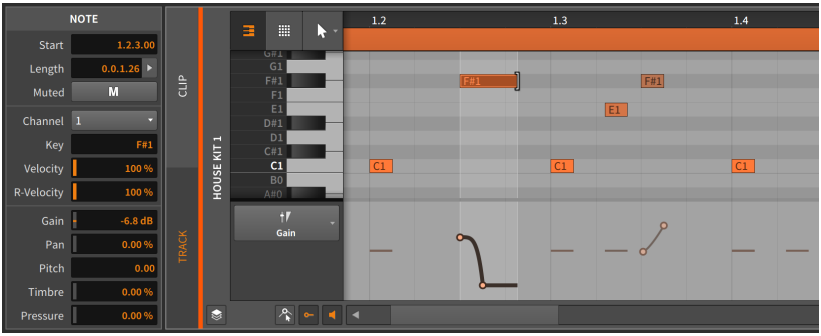
11.2.2.1. Timing and Mute Section

These settings relate to the musical position of the selected note and whether it is muted:

› *Start* sets the start position of the event within its parent clip or track. Adjusting this position will move the note event as it exists, the same as clicking and dragging the event within the **Detail Editor Panel**.



› *Length* sets the duration of the event within its parent clip. Adjusting this duration will simply lengthen or shorten the note event, the same as using the bracket cursor to adjust the right edge of the note.



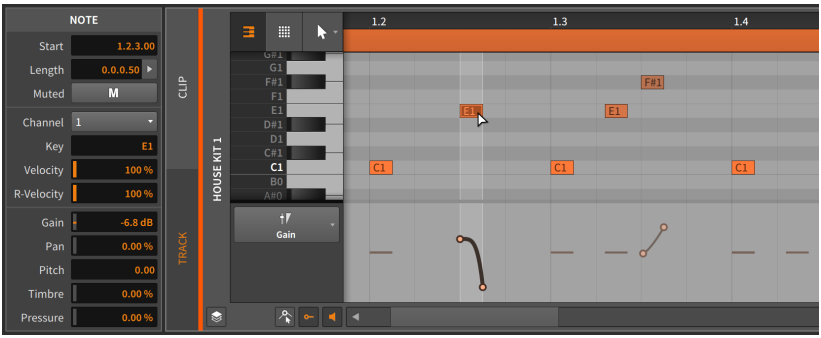
› *Muted* toggles whether or not the event is disabled on playback.

11.2.2.2. Note Properties Section

These parameters relate to how each selected note is sounded:

› *Channel* sets the internal channel that the note will play back on. This can act as a routing control within an **Instrument Layer** device, or when being sent directly to a VST plug-in or a hardware MIDI device that respects multiple channels.

› *Key* sets the root pitch that the note is set to. This is shown as a MIDI note value, where *C3* is roughly 261.262 Hertz ("middle C") and *A3* is 440 Hertz. Adjusting this value is the same as moving the note higher or lower.



Any Micro-pitch expressions are applied relative to the note's *Key* setting.

› *Velocity* sets the strength with which the note should be initially triggered. It is set on a scale from *0.00 %* to *100 %*, and this is just another representation of the note's velocity expression (see section 11.1.2.1).

› *Vel Spread* sets the bipolar spread range for the note (see section 10.1.3). So if a note has a *Velocity* of *78.7 %* and a *Vel Spread* of *10.0 %*, the note will trigger with a velocity between *68.7 %* and *88.7 %* each time it plays.

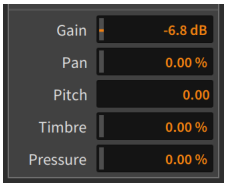
› *R-Velocity* stands for *release velocity*, and it sets the speed with which the note should be released. It is set on a scale from *0.00 %* to *100 %*. This parameter is implemented in whatever way the instrument device desires.

11.2.2.3. Operators Section

Unlike the other sections in the **Inspector Panel**, the section displaying **Operators** is only shown when notes (and not clips) are selected. **Operators** are covered extensively in their own chapter (see chapter 12).

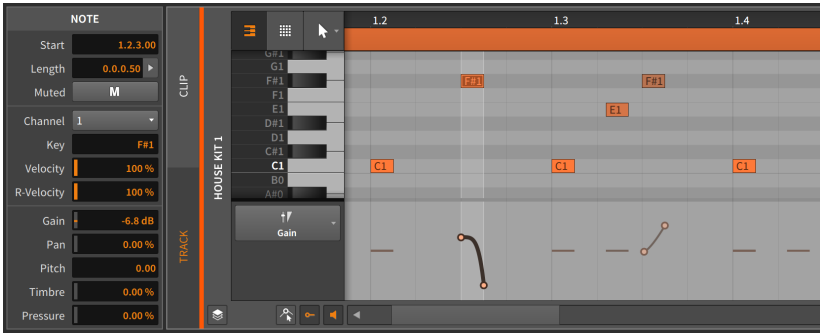
11.2.2.4. Expressions Section

This section exposes five of the expressions we have covered: *Gain* (see section 11.1.2.3), *Pan* (see section 11.1.2.4), *Pitch* (also known as the Micro-pitch expression; see section 11.1.3), *Timbre* (see section 11.1.2.5), and *Pressure* (see section 11.1.2.6). While these expressions have completely different functions, they are programmed in the same fashion.



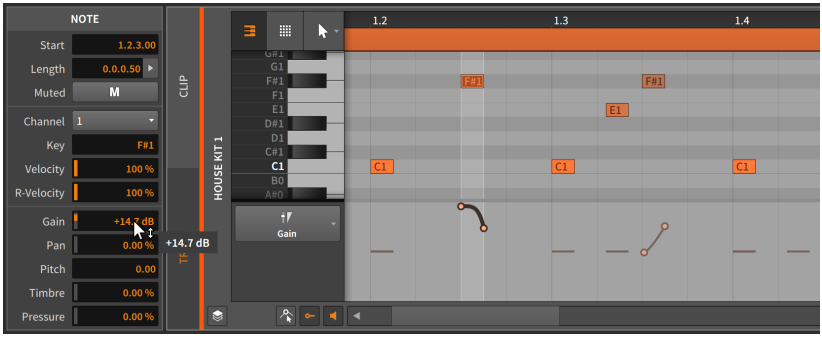
Most of these expressions have their units defined, with *Gain* set in decibels, and both *Pan* and *Timbre* set with bipolar percentages. The unlabeled *Pitch* is set in semitones, indicating the relative shift.

These are all automation-type expressions, so each is able to be defined by a curve made of several values. Because of this possibility, each value in this section of the **Inspector Panel** actually represents the average of points in that expression. We can see this in action with the *Gain* setting.

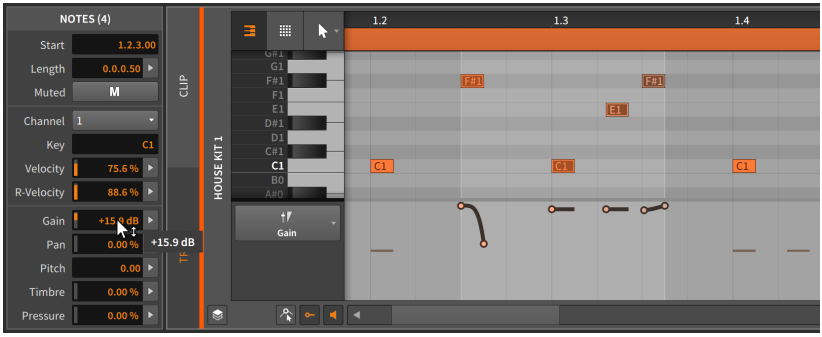


This note has a gain expression consisting of two points and a curve. The *-6.81 dB* listed for the *Gain* parameter is an average of these two points.

To adjust a note expression curve: change its listed average value.



This would work similarly if multiple note events were selected.

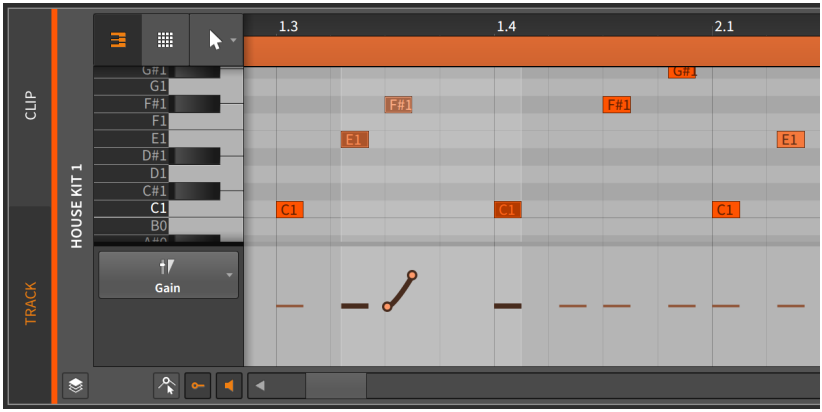


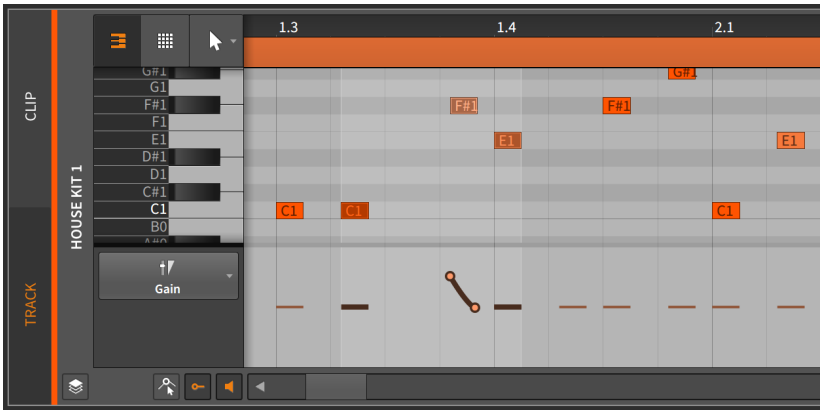
11.2.2.5. Event Menu Functions

These functions take the specified action on the selected note event(s):

› *Reverse* flips the selected event around, causing it to play backwards.

The following images demonstrate a group of selected events both before and after the *Reverse* function is applied:





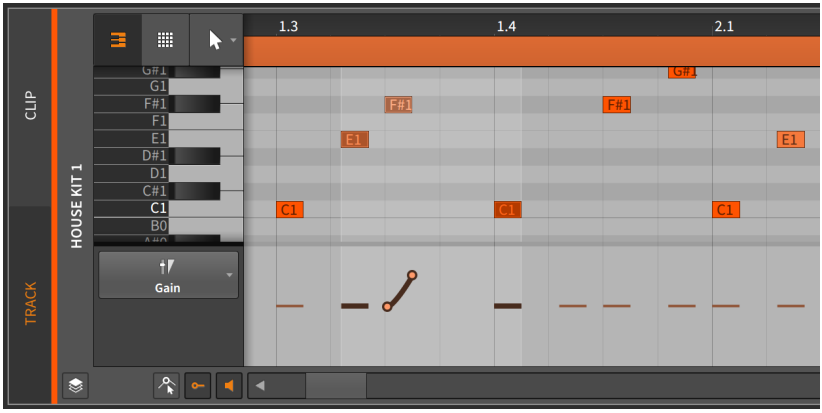
Notice that the expressions are also reversed.

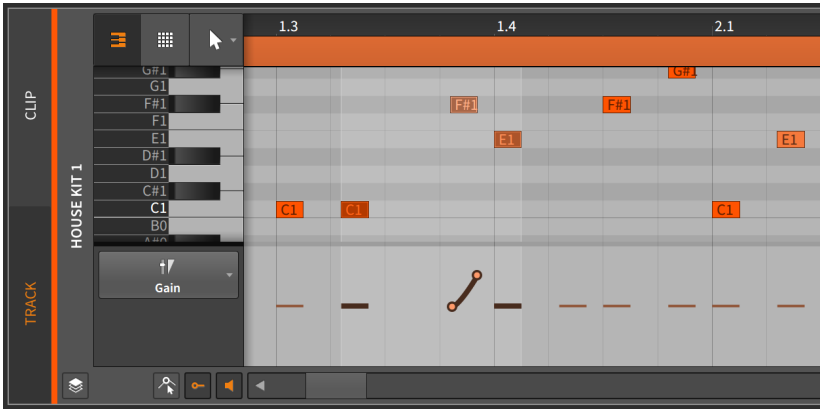
› *Reverse Pattern* flips the order of a group of selected events. This does not cause each event and its expressions to play backwards, but rather causes the last event to be played first, etc.

Note

This function will work only when multiple events are selected.

The following images demonstrate a group of selected events both before and after the *Reverse Pattern* function is applied:

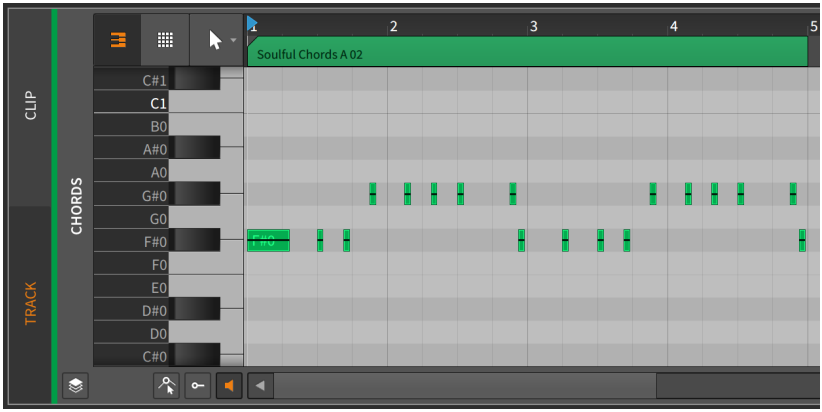


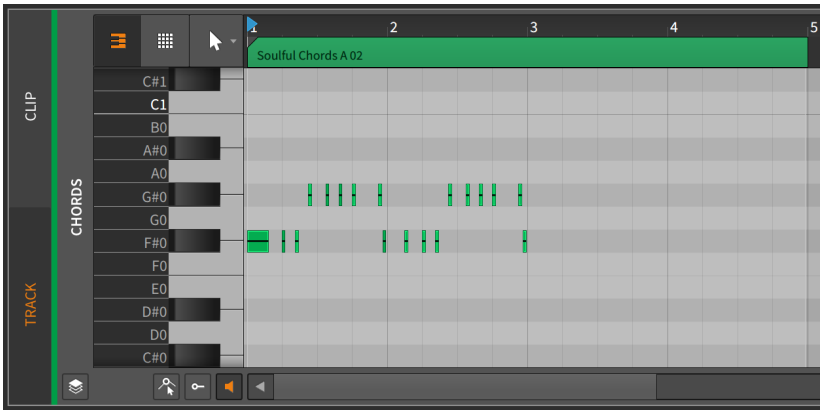


Notice that the expressions are preserved.

› *Scale 50%* halves the length of the selected event, effectively causing it to play back twice as fast. All expressions are also proportionally adjusted.

The following images demonstrate selected note events both before and after the *Scale 50%* function is applied:

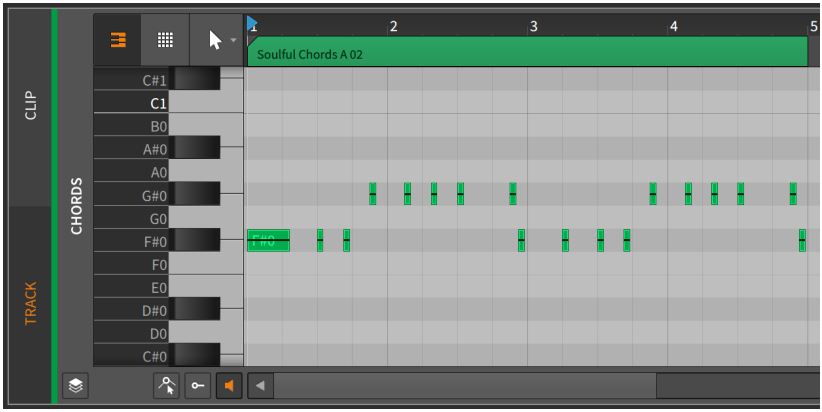


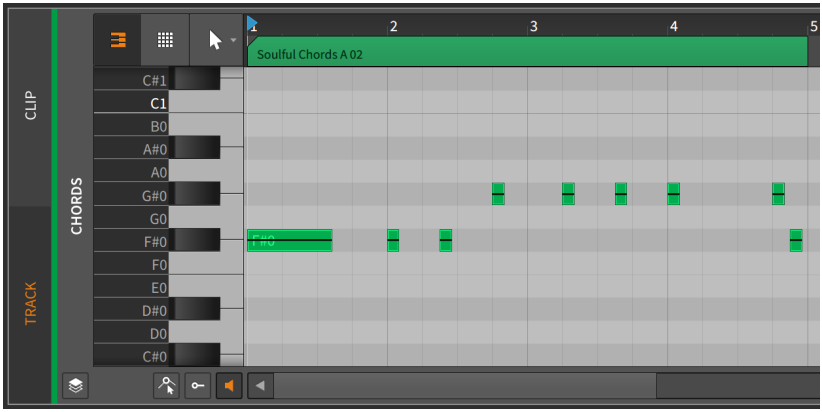


› *Scale Each 50%* is similar to *Scale 50%*, except the start time of each selected note event is preserved.

› *Scale 200%* doubles the length of the selected event, effectively causing it to play back half as fast. All expressions are also proportionally adjusted.

The following images demonstrate selected note events both before and after the *Scale 200%* function is applied:





Note

Remember that events must fit within their parent clip.

› *Scale Each 200%* is similar to *Scale 200%*, except the start time of each selected note event is preserved.

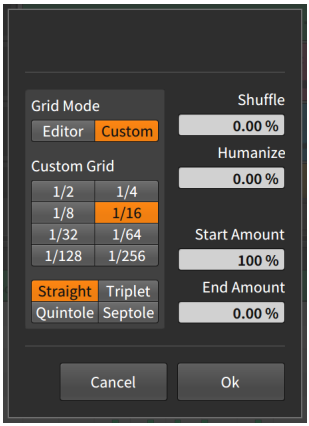
› *Scale...* requires a set *Amount* of scaling to be typed in, along with an option to *Scale each (keep position)*, which preserves the start time of each selected note event.

› *Slice In Place...* divides the selected event into multiple events at a selected regular note interval (*on Beat Grid*).

› *Slice At Repeats* splits any selected audio event using the *Repeats* Operator into individual events (see section 12.2.1). When a selected event does not have *Repeats* enabled, no change is made.

› *Quantize* is identical to the following *Quantize...* function except that the most recently set parameters are used for the function.

› *Quantize...* moves the start and/or end times of the selected note in relation to a beat grid. The parameter pane for this function appears when the right-arrow button is clicked.



› *Grid Mode*: Determines whether to adopt the grid settings from the current *Editor* or to allow *Custom* grid settings.

› *Custom Grid*: Exclusive *beat grid resolution* and *beat grid subdivision* settings (see section 3.1.2) for the quantize function.

Note

This is available only when *Grid Mode* is set to *Custom*.

› *Shuffle*: Amount of swing/groove (see section 2.3.2) applied to the beat grid for the quantization function.

› *Humanize*: Amount of randomness added to the quantize function, with the intention of mimicking human imperfection.

› *Start Amount*: Amount of quantization applied to each selected event's start position.

For example, a setting of *50.0 %* would move a selected event's start position halfway to the closest grid point. A setting of *100 %* places the event exactly on the closest grid point.

› *End Amount*: Amount of quantization applied to each selected event's end position.

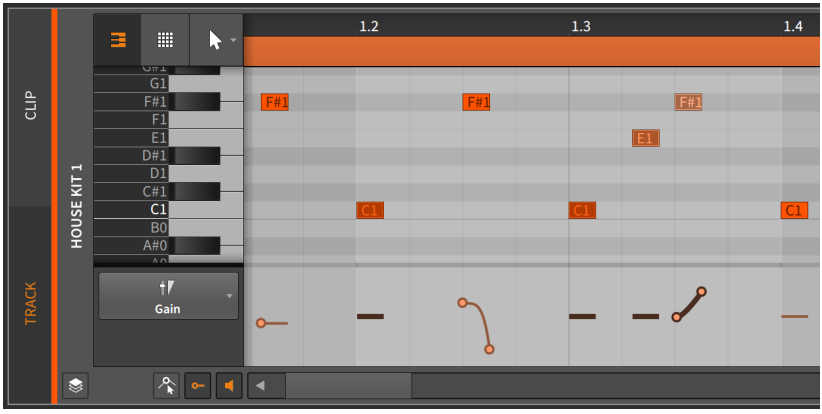
Note

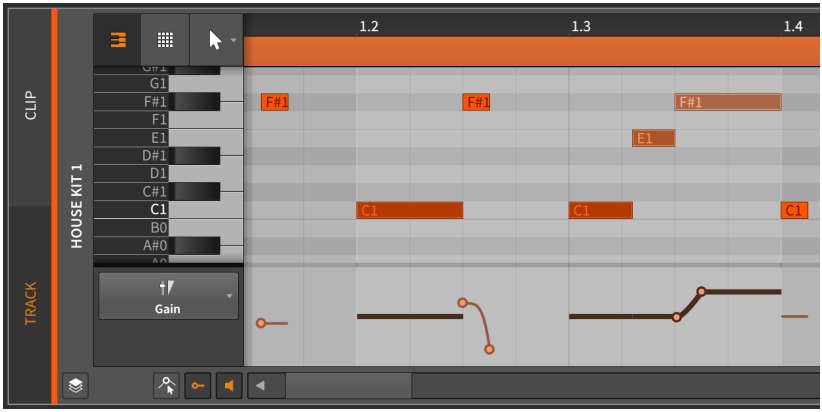
Humanize is the last factor applied in the quantize function. So a *Start Amount* of *100 %* might not place events directly on the grid if *Humanize* is enabled.

The quantize function can be executed by either clicking the *Apply* button at the bottom of the parameter pane, or by clicking the *Quantize Time* button itself.

› *Make Legato* adjusts the length of each selected note event so that it (or the chord it is a part of) ends immediately before the next event begins, creating a continuous series of events.

The following images demonstrate a group of selected events both before and after the *Legato* function is applied:





› *Transpose a Semitone Up* slides the selected event(s) up a semitone.

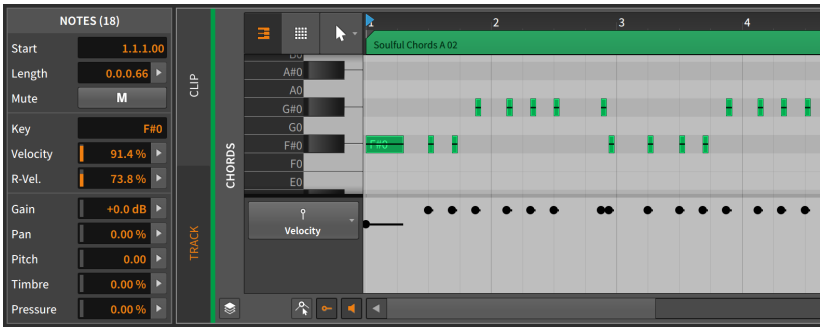
› *Transpose a Semitone Down* slides the selected event(s) down a semitone.

› *Transpose an Octave Up* slides the selected event up 12 semitones (in musical notation, *8va*). This function is also available by pressing [SHIFT]+[UP ARROW] .

› *Transpose an Octave Down* slides the selected event down 12 semitones (in musical notation, *8vb*). This function is also available by pressing [SHIFT]+[DOWN ARROW] .

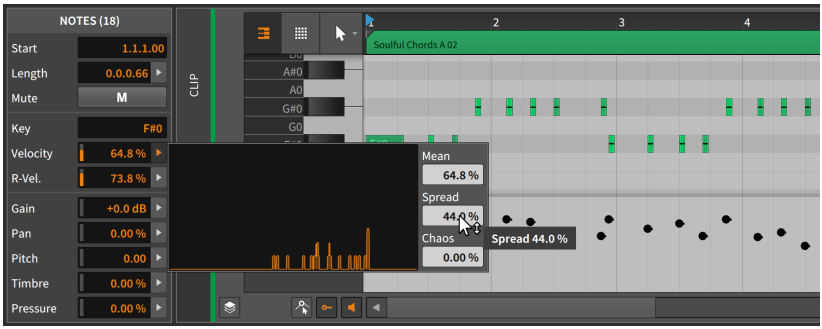
11.2.3. Working with Multiple Note Events

As it was with audio events, the **Histogram** becomes available when multiple note events are selected (see section 10.2.2.2).



In this example image, the **Inspector Panel** has labeled its bottom section as *NOTES (18)*, indicating that 18 notes are currently selected. And with this selection of multiple note events, the *Velocity*, *R-Velocity*, *Gain*, *Pan*, *Pitch*, and *Timbre* parameters all can now use the **Histogram** interface for editing.

The **Histogram** works exactly the same as it did in the audio event context (again, see section 10.2.2.2). The **Histogram** can be useful in the note context, for example, when notes were programmed without much diversity in their velocities.



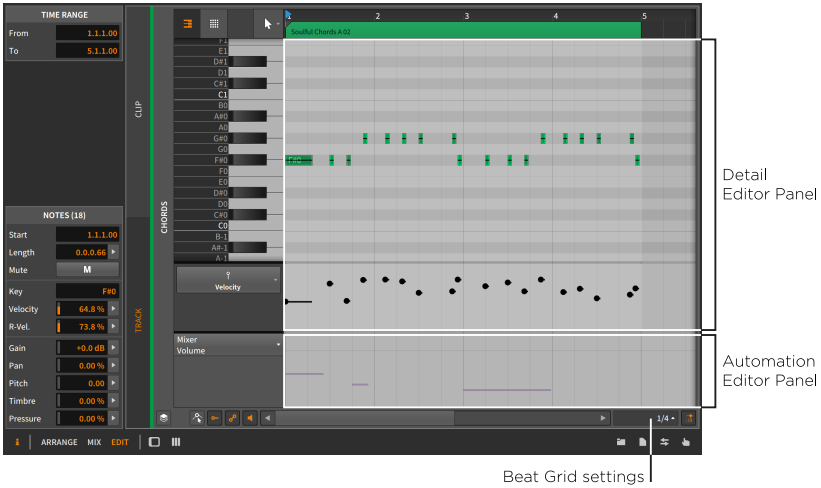
It doesn't take much to add subtle — or less subtle — variety with the **Histogram**. If you look, you will find places where it can aid your workflow.

11.3. The Edit View

Now that we have exhaustively covered both the **Automation Editor Panel** (in chapter 9) and the **Detail Editor Panel** (in both this chapter and chapter 10), we can now take a look at the **Edit View**, the last of Bitwig Studio's three views.

As we've discussed before, each view is a curated layout of Bitwig Studio's panels that is meant to serve a particular musical task. The **Arrange View** is purposed for assembling music, placing the important **Arranger Timeline Panel** centrally and giving you access to all panels around it. The **Mix View** centers around the **Mixer Panel**, focusing on the mixing board capabilities of each track while also streamlining the **Clip Launcher Panel** to facilitate improvisation.

Both of these views are oriented to show your project's tracks side by side, letting you craft a balance between them. But the **Edit View** is focused on the details of single tracks and clips.



The description just given and the image above should both be familiar at this point. The **Edit View** has two central panels: the **Detail Editor Panel** with an optional **Automation Editor Panel** fused beneath it. Aside from their positioning and the *Automation Editor Panel view toggle*, these panels work exactly as we have already learned them.

This combination allows you to focus on either the track or clip level so that you can work with the note/audio events contained there, the attached expressions, and the automation all beside each other. And putting the **Detail Editor Panel** front and center gives you much more display space for seeing more notes at a time — or, in the case of layered editing mode, more tracks. These are all welcome additions to the toolbox.

As a final point, the **Edit View** also strengthens the utility of *display profiles*. Since these profiles are meant to enable you at various stages of music production, you can probably imagine situations for having the full project on one screen (the "big picture") so that you can select a single clip or track and have its contents presented on the second screen (the "close up"). Again, once you scratch the surface, you will find uses for these functions in your workflow.

12. Operators, for Animating Musical Sequences

Music is normally conceived in a rough, fuzzy way and then programmed into the computer as an expression of stark certainty. If only some of the composer's thought process could be entrusted to the computer, so that changing circumstances might yield different, symmetric results. And this is why Bitwig has *Operators*.

Operators change when or how notes and audio events are triggered. In other words, **Operators** allow you to take sequenced events and animate them with randomness, cycle-aware logic, performance controls, and other interrelationships that expand what a clip is capable of.

Let's spend a moment with this pile of various and sundry ideas.

› *Randomness*, weighting any event to be more or less likely each time the play head arrives.

› *Cycle-aware logic*, considering how many times a clip has looped. So events can be triggered only on (or after) a clip's initial loop, or an event can think in a number of cycles (say, six loops per cycle) and then trigger on the first, second, and fourth loops each time.

› *Performance controls*, mapping events to play (or not) when the mappable *Fill* button is on.

› *Interrelationships between events*, so events play only when the previous event did (or didn't).

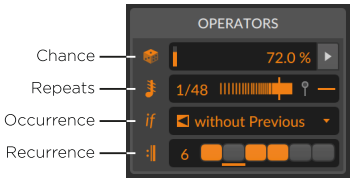
› Even the simplest idea — turning a single event into countless, ramping *retriggers* — multiplies your sound design possibilities while keeping the mayhem manageable.

These are strong ideas individually (and they do make appearances all thru Bitwig Studio). But they are together within **Operators**, which can shift the compositional process a bit, allowing you to program conditional relationships between events and much more.

We'll start by examining each of the four Operator modes. Then we'll look at a few functions that relate to **Operators** in one way or another. And after you've played with the **Operators** one at a time, do try them together (maybe *Chance* on one event, followed by *without Previous* on the next). Because while each mode has its charms, simple combinations can yield exquisitely musical results.

12.1. Operator Modes

When note or audio events are selected, an *Operators* section is visible in the **Inspector Panel**. Each line represents one of the four Operator modes, with most parameters available right there.



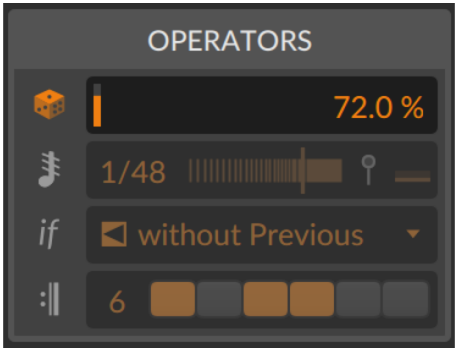
For any new event — whether by drawing notes, or splitting an audio clip, or recording either notes or audio, etc. — all **Operators** are set to neutral states, which have no effect other than to play back a normal event each and every time the playhead passes it. These default values will be noted below.

But note that mode icons also double as toggles. So if you try something new with *Occurrence* settings, you can always click the *if* button to temporarily bypass just the *Occurrence* behaviors for the selected events. By default, all of these modes are enabled, which means that trying out any Operator is as simple as selecting some notes or audio events, and setting a value in the **Inspector Panel**.

As each mode is unique, let's spend a moment with each of them.

12.1.1. Chance

Chance sets the likelihood that any event will occur, adding a mercurial element to your events.



Chance only has a single parameter, which represents the probability that this event will play. So if an event's *Chance* value is set to *50 %* (half of the time) and the clip plays four times, the event is most likely to play two of the times and to not play the other two times.

Chance is visualized on each event like the face of a die (dice). The number of dots or spots shown represents the current setting:

› 5 dots - *80 %* to almost *100 %*

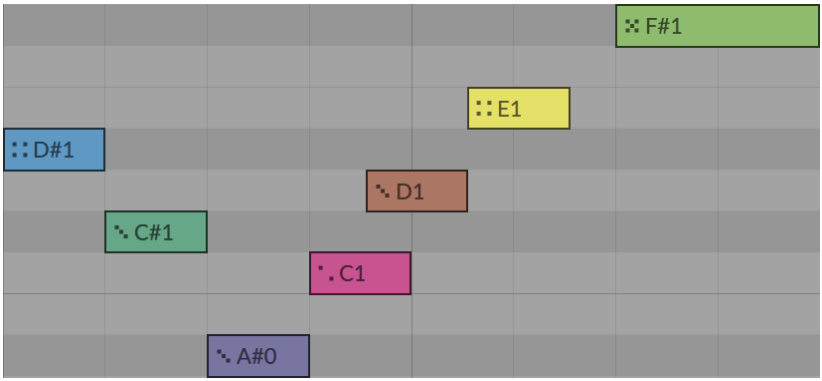
› 4 dots - *60 %* to almost *80 %*

› 3 dots - *40 %* to almost *60 %*

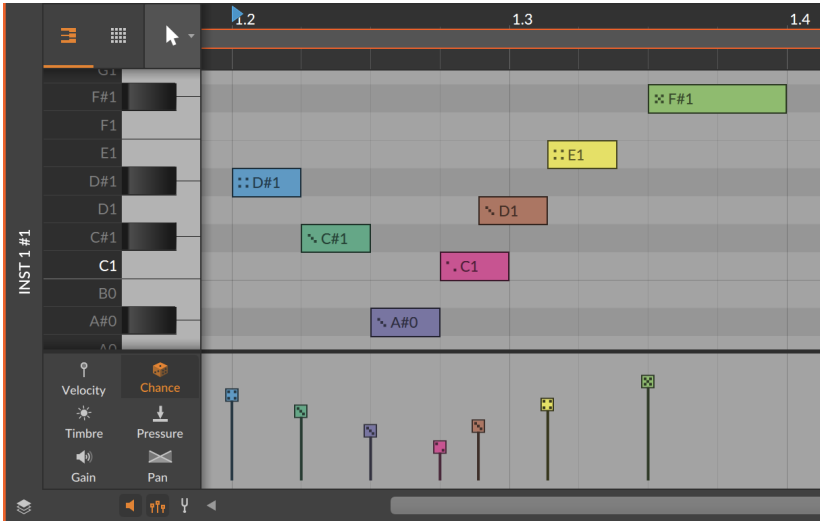
› 2 dots - *20 %* to almost *40 %*

› 1 dot - *0 %* to almost *20 %*

For example, the following series of notes goes from high to low probability, and then back up again.



And when working with notes, the *Chance* expressions have their own editor, appearing right after the velocity expressions (see section 11.1.2.2).



All this talk of "most likely" and what is "probable" reminds us that *Chance* is random. So other than its default value (a neutral *100 %*, meaning *always*) or a setting of *0 %* (read: *never*), every other value is perfectly unpredictable for any single moment.

As the one randomized Operator, *Chance* is determined by the clip's *Seed* parameter (see section 5.1.10.7). This makes its behavior identical to expression *Spread* (see section 10.1.3).

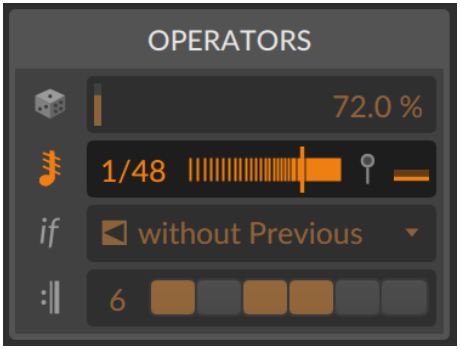
And if only using the *Chance* Operator on an event, you will see at the start of each clip cycle whether that event will play or not. For notes, a full stroke around the note shows that it will play this time. For audio events, a normal, bright color stripe in the audio event header indicates it will trigger.

Note

Not all **Operators** produce playback visualization. So when multiple **Operators** are in use, you may see the visualization for a positive *Chance* outcome, but the event might not trigger for other reasons.

12.1.2. Repeats

Repeats causes retriggers within the original event, letting any single event create (and control) myriad more.

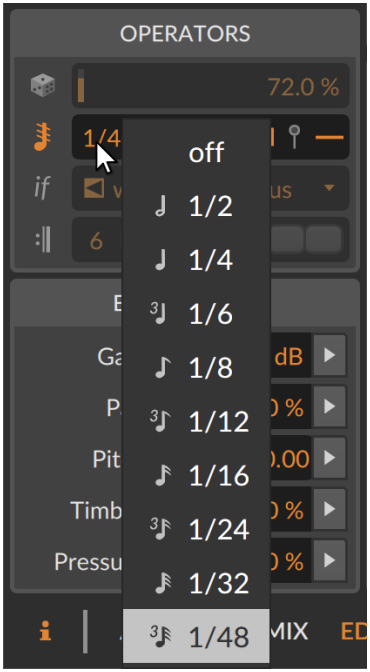


Repeats has at least two parameters.

› *Repeat Rate* determines when the retriggers will happen. It defaults to *Off* (no effect; the same as typing in *1* or *0* [zero]), and this parameter actually has two modes.

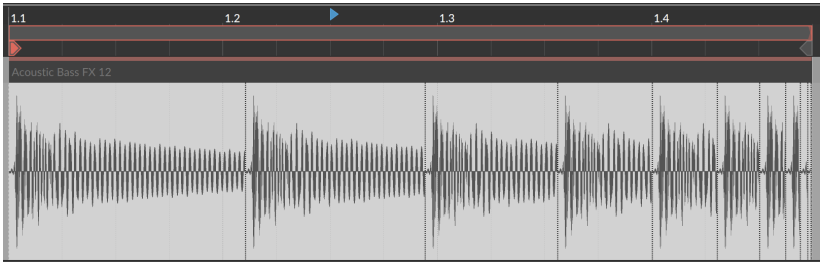
When dragging upward, you scroll thru positive numbers (*2*, *3*, up to *128*). This sets the number of pieces that the event is divided into. This also means that changing the length of the event will change the placement of its repeats.

If you drag downward, you go thru fractions (*1/2*, *1/3*, up to *1/128*). This sets the rate at which repeats will occur in beat time, which is not affected by the length of the event. And while any value is available, some standard musical intervals are available in a pop-up menu by right-clicking on the *Repeat Rate* parameter itself.



› *Repeat Curve* is represented by the horizontal slider beside *Repeat Rate*. The default value is centered (*0 %*), which keeps all repeats in their original position. Negative values (to the left of center) set the repeats to be closer together at the start of the event, and positive values (to the right of center) bunch the repeats closer together toward the event's end.

These two parameters determine the placement and timing of the repeats. Each event retrigger acts like a restart of the note or audio, with a visualization that clearly shows the placement and effect. For audio events, the waveform is shown restarting to match the playback behavior.

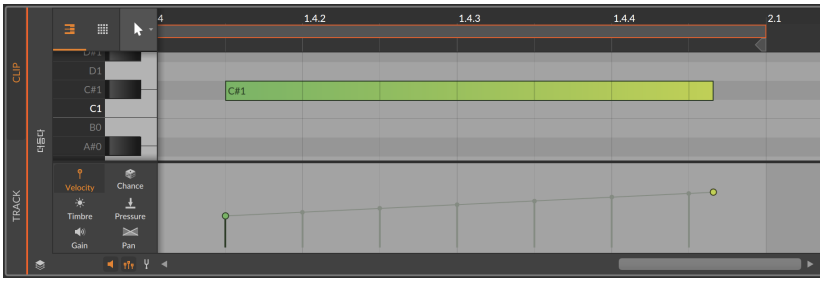


For notes, *Repeats* has two additional parameters related to velocity.

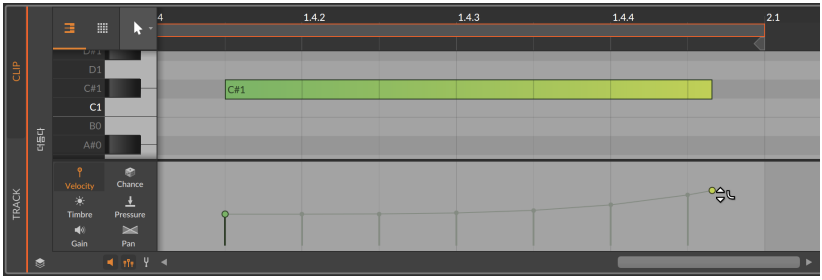
› *Repeat Velocity End* sets the target velocity for the end of the repeats. Since velocity is only used at the beginning of each note, this level may never be reached, but the curve will be maintained if the rate or timing of repeats is changed. The parameter range is a bipolar percentage, mapping the end point relatively across the full velocity range.

So let's assume a note with a velocity of *40 %*. A *Repeat Velocity End* setting of *0 %* would represent no change, starting every repeat with the original note velocity. A *Repeat Velocity End* value of *50 %* would initially trigger at velocity *40 %* with the successive repeats ramping up to a velocity of *70 %*. And a *Repeat Velocity End* value of *-75 %* would initially trigger at velocity *40 %* with the successive repeats tapering down toward a velocity of *10 %*.

› *Repeat Velocity End* is found at the right end of the *Repeats* line in the **Inspector Panel**, beside the vertical velocity pin icon (when notes are selected). It is also visualized in the velocity expressions lane as a draggable handle at the end of the note.



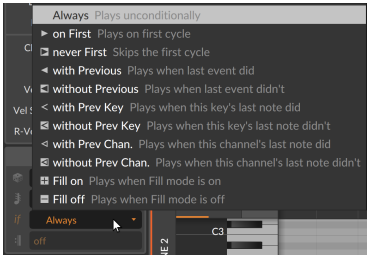
› *Repeat Velocity Curve* is available inside the velocity expression lane. By holding [ALT] and dragging the velocity end handle up or down, the curve of any repeated event can be bent to move toward the target velocity sooner or later.



One final note. Any note or audio event using the *Repeats* Operator is still a single event — at least until you choose to *Slice At Repeats* (see section 12.2.1). And as a single event, expressions can be draw across the length of each event, including across repeats.

12.1.3. Occurrence

Occurrence sets conditions for each event. The choice of *Condition* is presented in a single menu.



For any event using *Occurrence*, the icon for the selected *Condition* is shown on the event. And as we go thru each *Condition*, keep in mind that they are each self-contained with no additional parameters.

› *Always* - The event will play every time. This is the default, neutral state.

› *on First* - Plays on the first pass (including retriggers) of the clip

› *never First* - Plays every time *except* on the first pass (including retriggers) of the clip

› *with Previous* - Plays if the immediately previous event did

› *without Previous* - Plays if the immediately previous event didn't

› *with Prev Key* [note events only] - Plays if the immediately previous note on this key did

› *without Prev Key* [note events only] - Plays if the immediately previous note on this key didn't

› *with Prev Chan.* [note events only] - Plays if the immediately previous note on this channel did

› *without Prev Chan.* [note events only] - Plays if the immediately previous note on this channel didn't

› *Fill on* - Plays when *Fill* mode is on, in the global transport (see section 2.3.2)

› *Fill off* - Plays when *Fill* mode is off, in the global transport (see section 2.3.2)

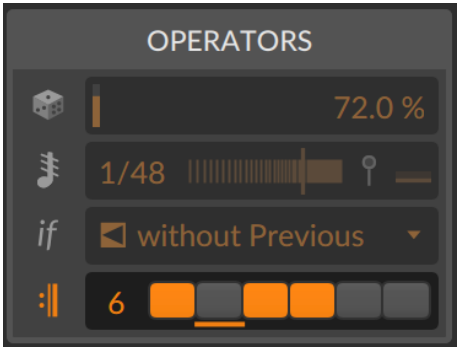
So thinking of the list from the start of this chapter, *Occurrence* includes cycle-aware options (the two *First* modes), interrelationships (all modes using the *Previous* idea), and performance controls (*Fill on* and *Fill off*).

Note

Of the *Occurrence* modes, only the two *First* modes provide playback visualization.

12.1.4. Recurrence

Recurrence gives each event its own looping timeline.



As shown above in the **Inspector Panel**, there are two parameters along with one visualization element that make this work.

› *Recurrence Length* sets the number of loops per cycle for this event. This can be set between *1* (the default, neutral value, shown as *Off*) and *8*.

› Following the length value is a matching number of toggle boxes. Each *Recurrence Step* is clickable to toggle whether the event will trigger on that particular loop of the cycle.

› You'll also notice an underline beneath one of the step toggles. This little indicator tells you which loop of the cycle is currently playing back.

This pattern is also shown on the right edge of events themselves, with a series of shaded (on) and empty (off) rectangles.

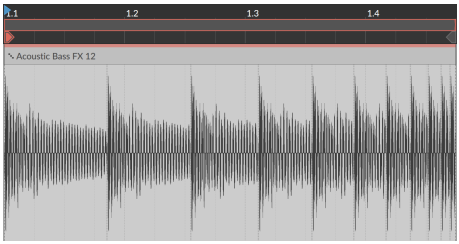
And finally, *Recurrence* does provide playback visualization on each event when a loop cycle is started.

12.2. Operator-related Functions

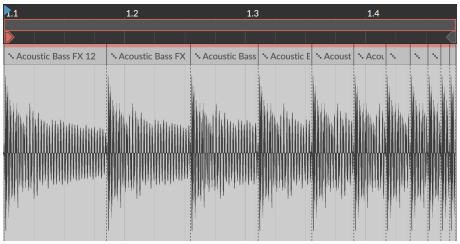
Operators make a great deal possible, but sometimes you want to take things a step further. Let's examine a couple features for unspooling your events (*Slice At Repeats*) or clips (*Expand*), and then check in on an old friend (*Consolidate*) to see how they are handling these new data.

12.2.1. Slice At Repeats

The charm of *Repeats* is that the original event houses all of the repeats, with the parameters remaining adjustable. This is usually better than slicing the original event into new ones, but for certain cases, that might be exactly what you want to do. While it works for notes too, we can visualize it with audio event.



To turn an event's Repeats into individual events: select the event and then choose *Event* › *Slice At Repeats* .



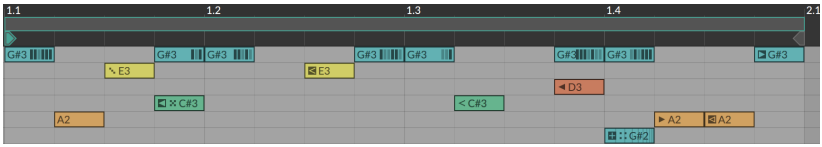
This is the same as if you had taken the knife tool and manually sliced the event at each repeat — except each new region has *Repeats* turned off. But all other expressions and **Operators** will be preserved in each new event, potentially changing the playback (for example, *Chance* in the above image).

12.2.2. Expand, from the Clip Launcher

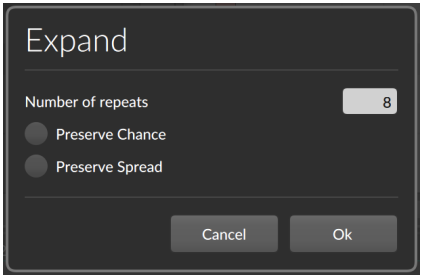
With **Operators**, the number of bars you see in a clip is much smaller than the unique output you can get when the clip is looping. Depending on the modes used, each loop could be unpredictable and unlikely to repeat their pattern, or spread out so that repeats only happen every dozen (or few hundred) loops, or both. This is a lot to keep in mind.

The *Expand* function takes any Launcher clip and supports printing out two, twenty, or however many cycles of the original as a new clip (with all possible **Operators** removed and printed as permanent events). And unlike "bounce" functions, *Expand* outputs the same kind of clip you started with — note clips remain note clips, audio clips remain event-based audio. This lets you see all the nested patterns and relationships that **Operators** can bring to a "simple" loop, or even start a precise edit without the randomness.

For this example, here is a one-bar note clip as our source material.



*To print several repetitions of a Launcher clip with **Operators** removed:* select the original Launcher clip and then choose *Clip* › *Expand...*. A dialog will appear with three settings.

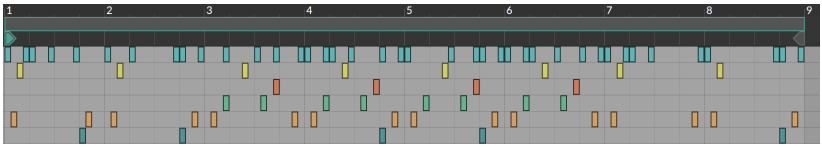


› *Number of repeats* represents the number of cycles of the clip that will be printed. So if our example is one-bar long, setting a value of *8* will create a new eight-bar clip on the next available clip slot.

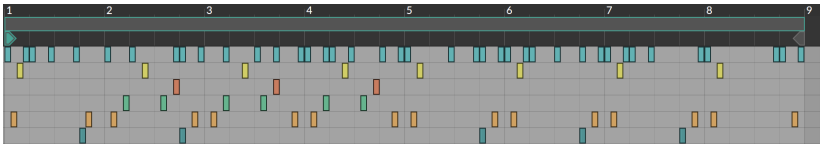
› *Preserve Chance* keeps all events' *Chance* Operator settings. This will maintain that random element on playback. This is off by default, flattening all *Chance* values to events that either happen or do not.

› *Preserve Spread* keeps all events' expression *Spread* values (see section 10.1.3). This will maintain that random element on playback. This is off by default, flattening all *Spread* occurrences to discrete values.

By taking our example source above and choosing to *Expand* with *8* repetitions, I got this clip.



And if I try expand with exactly the same settings, the presence of *Chance* may take things in a different direction.



Only a few things remain unchanged (for example, events without any **Operators**, and events that can't be known ahead of time, such as events that rely directly or indirectly on a *Fill* mode state). Some things come and go depending on the loop count (like the blue-ish notes on the top row, which only use *Recurrence* settings). Other events are purely randomized (like the yellow note within beat one of each bar). And other

events are chained together (notice that the early beat-one yellow note OR a green note appear, due to an *Occurrence* of *without Previous* on that first green note; never both).

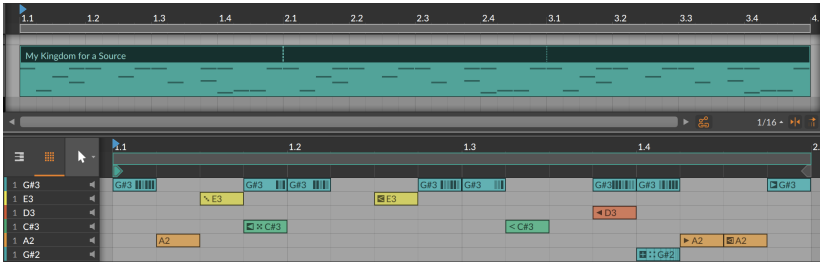
Whether *Expand* is just providing a visual readout or the plan is to lock-in the randomness of **Operators** and expression *Spread* as printed notes, that is up to you.

12.2.3. Consolidate

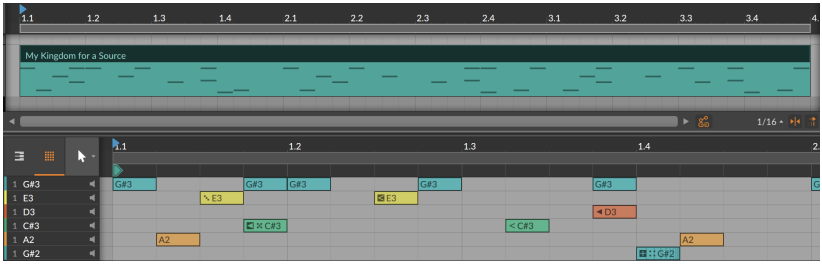
The clip *Consolidate* function has come up in previous chapters as a way to lock in or solidify a clip. So *Consolidate* is also an option to flatten settings. Where *Expand* is oriented toward taking a Launcher clip and making something much bigger, *Consolidate* can flatten any clip at its current size, which makes it particularly useful on the Arranger where looping clips of defined length might live.

And just as *Expand* had options for how to handle the randomized *Chance* and expression *Spread* elements, *Consolidate* can either flatten or preserve these as well. It just depends on whether the clip in question is behaving randomly on each pass, or if it has a consistent *Seed* value (see section 5.1.10.7).

For this example, I've taken the same clip we were using to look at *Expand* in the previous section. I've just dragged it to the Arranger and set it to loop three times.

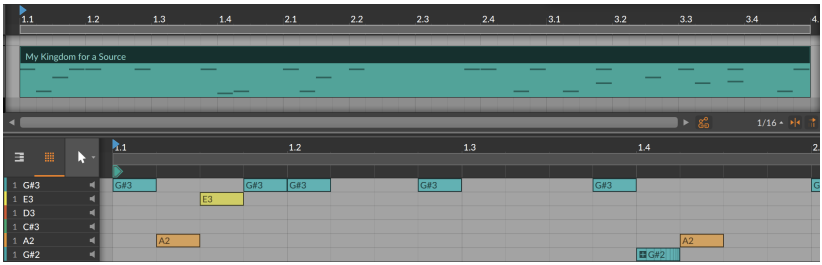


This clip's *Seed* value is currently set to *Random*, which means *Chance* and expression *Spread* elements will be freshly randomized at the moment this clip starts playing. And this is preserved if *Consolidate* is used on this clip, as shown below.



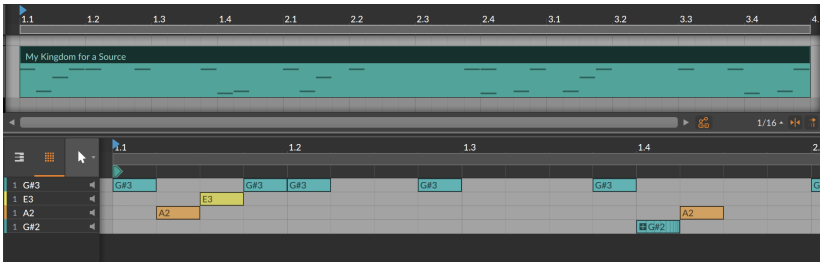
So the looping regions have become real copies, as is always the case with *Consolidate*. And **Operators** that are unpredictable remain on their events. Only settings expressing certainty (like cycle-aware *Recurrence* settings or *First*-related modes used in isolation) will be flattened to regular events with the original **Operators** removed.

If the clip has a replayable *Seed* value, then that seed will be used to permanently print all randomized *Chance* and expression *Spread* values.



So fewer notes exist now because of the direct and indirect random relationships, and the only **Operators** that now remain are those considering the *Fill* mode state (and any following events tied to those with *Previous* logic), and events using *Repeats*, which are never touched by either *Expand* or *Consolidate*.

And if a new *Seed* value is requested for the original clip, using *Consolidate* again might yield a different result.



And it did. This time.

13. Going Between Notes and Audio

The previous two chapters dealt extensively with audio events and note events, which you could think of as our primary states of musical matter. And those last two chapters are the longest in this document because there is quite a lot to do in Bitwig Studio with audio and note events.

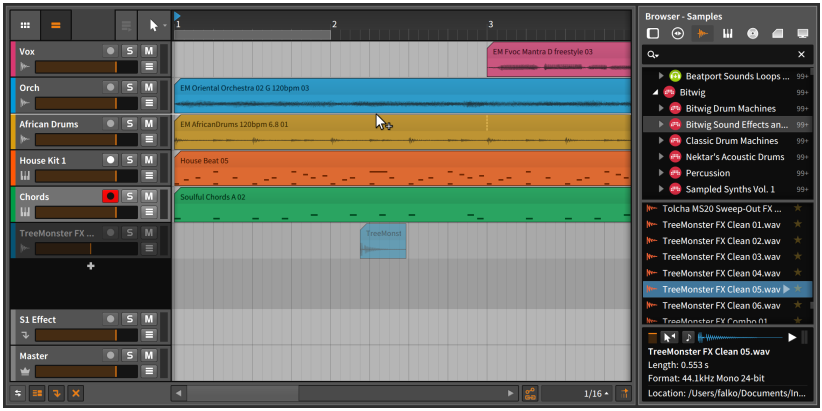
Working with notes has inherent advantages, as does working with audio. Without trying to tell you how either of these types "should" be used, it is fair to say that notes allow for a greater level of flexibility and control, and audio is both highly portable and can be wonderfully mangled.

But occasionally, matter changes state. This happens in the physical world when water freezes, and it also happens when you use Bitwig Studio to bounce a note clip in place. And just as ice sometimes melts, even audio can be sliced into note events.

This chapter will explore ways of taking audio materials into the note domain, ways of transforming note events into audio ones, and places where both coexist. We may not be altering nature here, but these options will only afford you more opportunities to customize your workflow and sound.

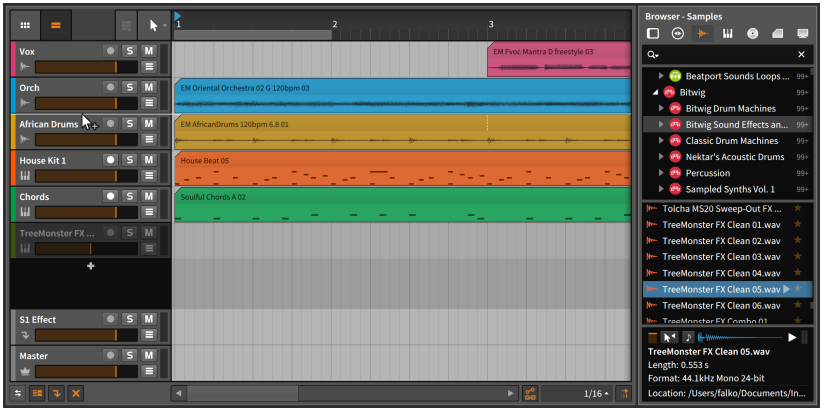
13.1. Loading Audio into a New Sampler

We learned early on how to import media files from the **Browser Panel** as clips. We saw how to bring clips of any kind into the **Arranger Timeline Panel** (see section 5.1.1) and into the **Clip Launcher Panel** (see section 6.2.1). In both cases, we also saw how to create a new track for that clip by dragging it to the space between any two existing tracks.

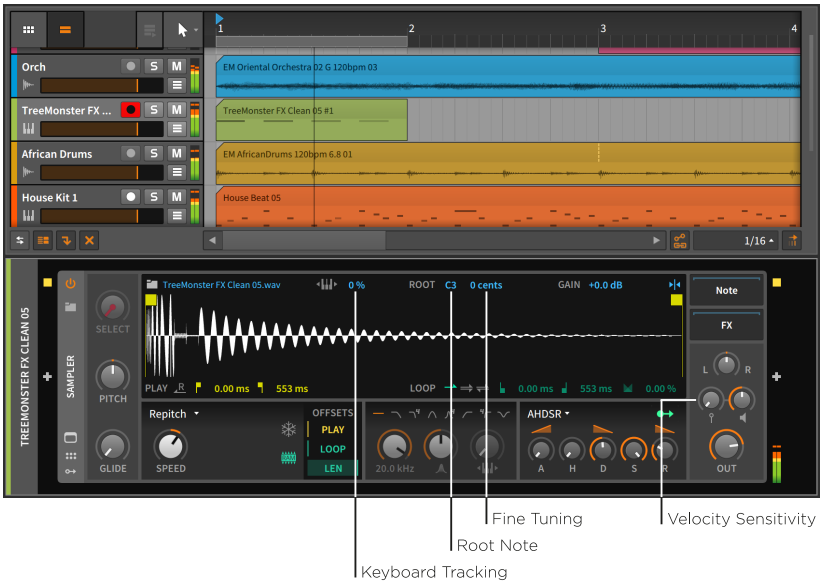


When importing an audio file, Bitwig Studio provides one additional option worth mentioning here.

To load an audio file into a new Sampler device on a new instrument track: click and drag the clip from the **Browser Panel** to the space between two existing track headers.



Once the mouse is released, a new instrument track will be inserted in that place, and the track will be selected.



With the track record enabled, you can now use notes to trigger the audio that was just loaded.

Rather than exploring the **Sampler** in any great detail, we'll look at just a few parameters that affect how the notes you play are interpreted by **Sampler**.

› *Keyboard Tracking*: When disabled, any note triggers the sample at its original pitch. When enabled, each note's pitch setting will change the playback speed and pitch of the sample.

› *Root Note*: The note which will play the sample at its original pitch. This setting takes effect only when *Keyboard Tracking* is enabled.

› *Fine Tuning*: A small interval adjustment for the *Root Note* setting, in units of cents (hundredths of a semitone). This setting takes effect only when *Keyboard Tracking* is enabled.

› *Velocity Sensitivity*: The amount that each note's velocity affects the loudness of the sample. At the lowest setting (*+0.00 dB*), velocity is ignored.

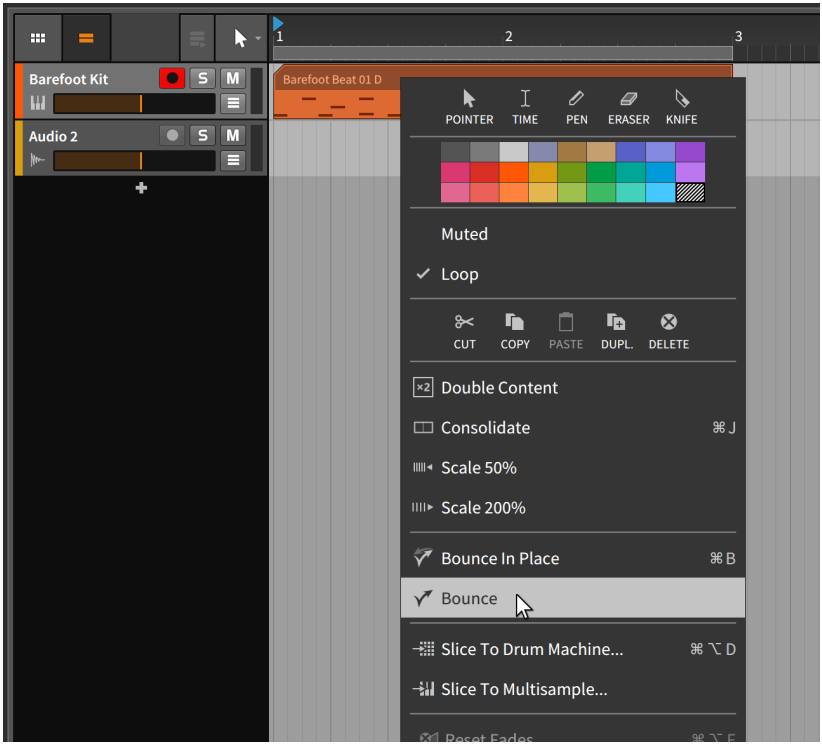
Note

For detailed information about **Sampler**, see section 19.23.5. (Descriptions for all Bitwig devices can be found in chapter 19.)

13.2. Bouncing to Audio

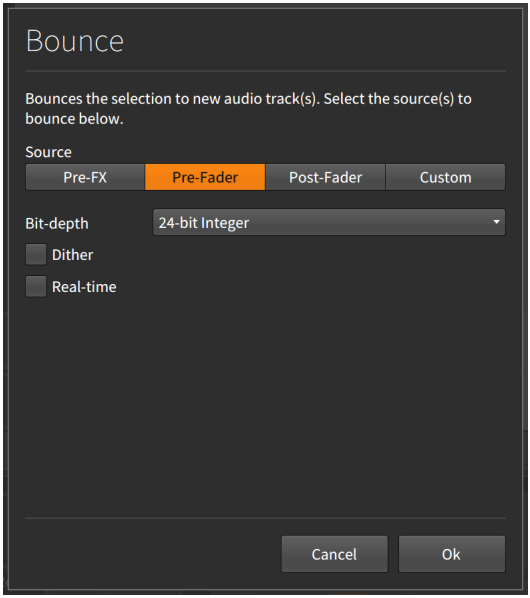
The concept of an audio *bounce* may be familiar. Sometimes called an *export* or *render* in other contexts, a bounce is a consolidated audio version of some part of your project. In this case, we want to investigate bouncing a note clip.

By right-clicking a note clip, a couple of bounce options are listed in the context menu. (These same options also appear in the *Edit* menu.)



13.2.1. The Bounce Function

The simple *Bounce* function presents a dialog box.



The *Source* choices refer to different places in the track's signal flow, and you get to select which point you would like the audio to come from.

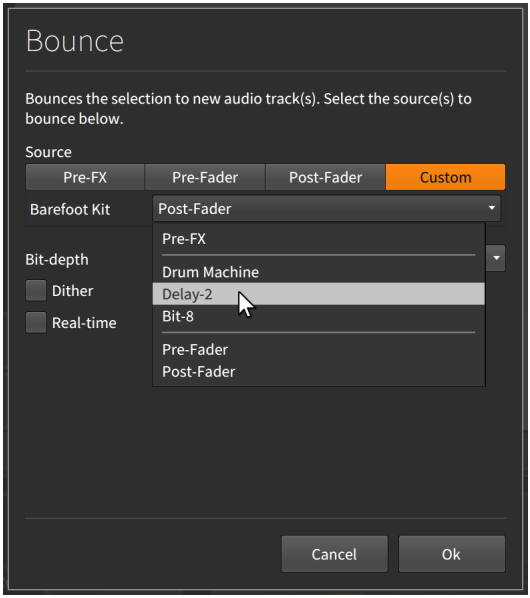
The options include:

› *Pre-FX*: The raw audio signal from the primary instrument's output.

› *Pre-Fader*: The audio signal after the track's device chain but before the track's volume setting is applied.

› *Post-Fader*: The audio signal after the track's device chain and volume setting.

› *Custom*: A special menu of options that includes every top-level signal junction in the track, including from within the device chain.



In this example, the instrument track in question has three top-level devices: **Drum Machine**, **Delay-2**, and **Bit-8**. Selecting one of these options chooses the audio output of that device for the bounce.

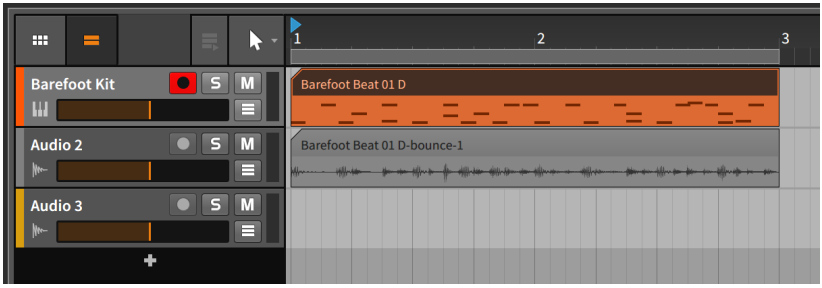
Three additional parameters follow:

› *Bit depth*: The resolution of the bounced audio file.

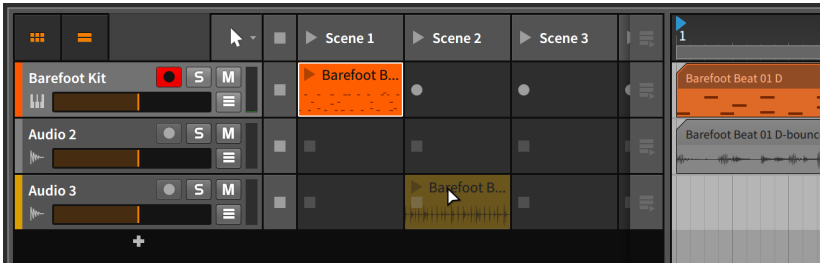
› *Dither*: A toggle for whether shaping is applied for the selected bit depth.

› *Real-time*: A toggle to bounce at the actual speed (and time duration) of the selection. This setting is necessary if you are bouncing external hardware, etc.

After making your selections, click *Ok* to bounce the audio onto a new track.



If you want a standard pre-fader bounce, you can also click and drag a clip while holding [ALT] ([SHIFT]+[CTRL] on Mac).



13.2.2. The Bounce in Place Function and Hybrid Tracks

The *Bounce in place* function is similar to the *Bounce* function with two key differences.

First, it presents no dialog box, taking the audio output from the primary instrument (*Pre-FX*).

Second, it replaces the clip you are bouncing with the bounce itself.

Note

Since *Bounce in place* deletes your source clip, it is a good practice to copy the clip (perhaps to the Clip Launcher) before using this function.

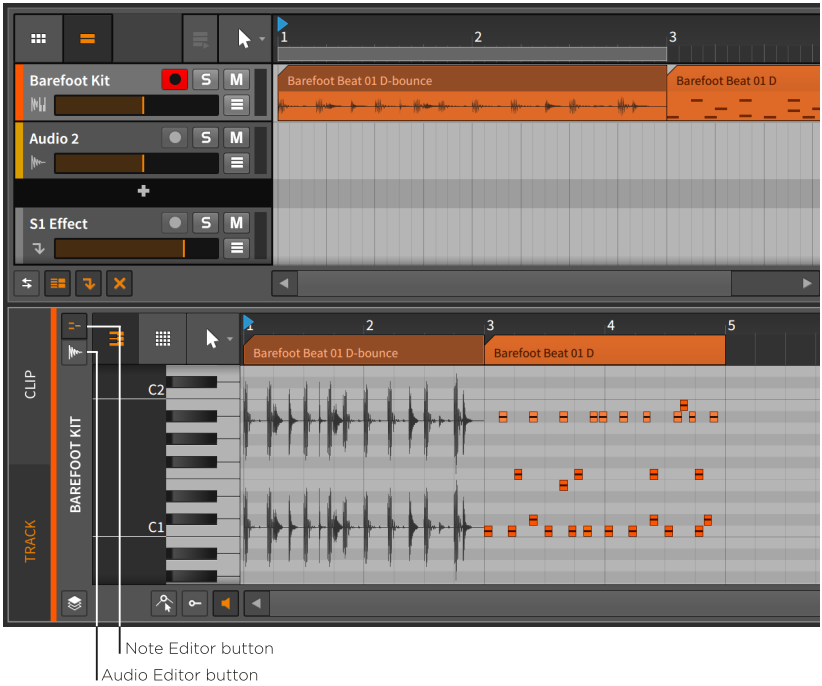
Note

When using *Bounce in place* on a metaclip within a group track (see section 5.1.9), the newly bounced clip is placed on the group track's

internal master track instead of replacing the source clip. Accordingly, the group track will now ignore its component tracks for that section, outputting only the audio of that bounced clip.

Since this was the only note clip on the track, Bitwig Studio has converted it from an instrument track to an audio track while preserving the entire device chain.

If there were other note clips on the track, it would have been converted from an instrument track to a hybrid track.



Since hybrid tracks allow both audio and note clips to be present, the **Detail Editor Panel** now has its *Audio Editor*and *Note Editor* buttons to keep things straight. These buttons (and the panel) work as they did when we first saw them in layered editing mode (see section 11.1.4). Otherwise, hybrid tracks work the same as instrument and audio tracks.

Note

To enable this workflow of hybrid tracks, virtually every Bitwig device passes thru signals that are not its focus. For example, normal note effect and instrument devices pass thru audio signals that reach them. And instrument and audio effect devices send on the note signals they receive, as following audio devices or modulators may take advantage of them.

The primary exception to this rule are devices using **The Grid**, which have some routing parameters for defining their "thru" behaviors (see section 17.2.1).

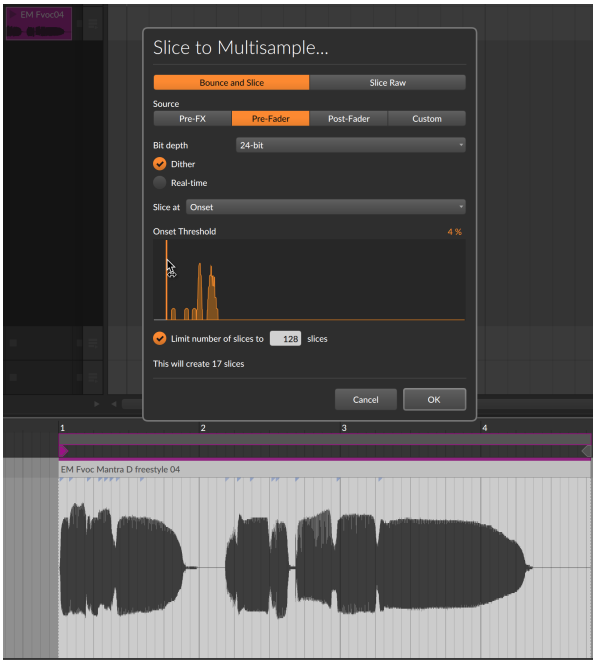
13.3. Slicing to Notes

The concept of a musical *slicing* operation may be familiar. The idea is to take an audio waveform and cut it into logical pieces that can be played with note messages.

By right-clicking an audio clip, a couple of slicing options are listing in the context menu. (These same options also appear in the *Edit* menu.)

13.3.1. The Slice to Multisample Function

The *Slice to Multisample...* function presents a dialog box.



The dialog begins with two options regarding the source to be sliced:

› *Bounce and Slice*: Executes a bounce function of the clip before slicing it. If this is selected, the signal flow options from the *Bounce* dialog are shown below (see section 13.2.1).

› *Slice Raw*: Simply slices the raw source event.

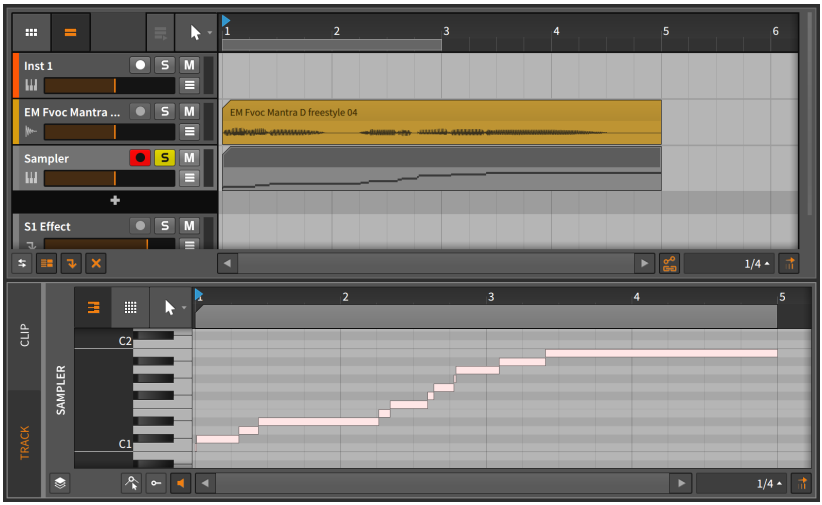
After these choices comes the critical *Slice at* setting, which determines at what interval slices will be made. The choices are self-explanatory, including event-based intervals (*Beat Marker*, *Onset*, and *Audio Event*) and time-based intervals (*Bar*, *1/2 note*, *1/4 note*, *1/8 note*, *1/16 note*, and *1/32 note*).

When *Onset* is selected, an additional histogram view appears in the dialog, showing the current selection's onsets based on their strength. The *Onset Threshold* parameter lets you use only use the strongest onsets. If an *Onset Intensity Threshold* for set for this event (see section 10.2.1.2), then that value will be used. A setting of *0 %* will use all onsets.

Additionally, if the **Detail Editor Panel** is onscreen, it will stay bright even when the dialog is open to visualize which onsets will be used. If the dialog's *Onset Threshold* is changed — by adjusting the numeric control or dragging the vertical slider within the histogram representation — the shown/dimmed onsets in the **Detail Editor Panel** will update.

The final option in the dialog is to *Limit number of slices* or not. This does not alter the *Slice at* setting, but simply stops slicing if the maximum slice count has been reached.

Choosing to *Slice Raw* at each *Onset* and clicking *OK* would lead to a new instrument track with a new note clip.



On this new instrument track, a **Sampler** device has also been created with the corresponding slice of audio assigned to each note seen in the note clip.

The original audio clip could now be rearranged by editing the note events, or it could be reinterpreted on the fly by playing any of these notes in real time.

13.3.2. The Slice to Drum Machine Function

The *Slice to Drum Machine...* function leads to the exact same dialog as *Slice to Multisample...* and produces a new instrument track with a new note clip in the same way, but the instrument track is given a **Drum Machine** device with each slice loaded into its own separate **Sampler**.

The choice between **Sampler** and **Drum Machine** is really one of workflow. While **Sampler** places all slices in the same signal chain, the **Drum Machine** gives you independent chains (and a unique **Sampler**) for each slice. If you want to process individual slices in different ways, you might favor the **Drum Machine**.

In the end — like so many things in Bitwig Studio — the choice is up to you and your personal preference.

14. Working with Projects and Exporting

The title of this chapter isn't meant to cause confusion. Yes, we have been working with projects for the majority of this document, but there are a few details about projects that we haven't covered yet, including a few details about how Bitwig Studio manages project files.

Each Bitwig Studio *project file* uses the BWPROJECT extension. When you save a Bitwig Project file, the project file itself is placed in a new *project folder*. Whenever new content files are generated in a project, the program will automatically place them in the project folder within new sub-folders (such as *samples*, *plugin-states*, *recordings*, *bounce*, etc.).

While Bitwig Studio has its own preferences and settings, there are also project-based parameters that are stored within each project. And while preferences do apply across the entire program, these settings have to be reconciled with the content of your actual project file and folder.

Note

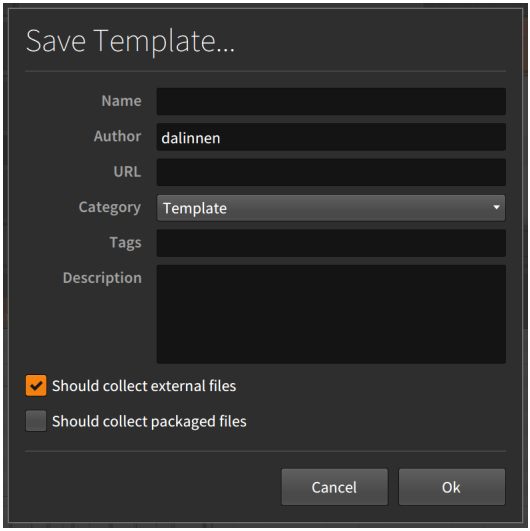
For information about storing project-specific mappings for computer keyboards and MIDI controllers, use the **Mappings Browser Panel** (see section 15.4).

In this chapter, we will see how to save project templates, either for your own use or for sharing with the wider world. We will look at the **Project Panel**, which manages your project's metadata and the status of files and plug-ins being used. We will talk a bit about the global groove settings and how they impact your project. We will show ways to share content between projects. And finally, we will examine exporting audio, MIDI, and even your entire project content from Bitwig Studio.

14.1. Saving a Project Template

Establishing a good workflow is key to working efficiently on new projects. Having *project templates* that are preloaded with common track setups, device arrangements, monitoring assignments, etc., may be a great boon to you and your productions.

Directly beside the *Save as...* function in the *File* menu is the *Save as Template...* option. Selecting this option pulls up a dialog.



There are six fields that you can set for your template.

› *Name*: The title of the template.

› *Author*: The name of the template's author (by default, your Bitwig user name is used).

› *URL*: A web address for the author.

› *Category*: Whether you would consider this project a *Template*, *Demo*, or *Tutorial*.

› *Tags*: Metadata pieces that can be used for choosing or sorting templates later. To finalize a tag, press [ENTER] after typing it. Use as many tags as you like.

› *Description*: A fuller explanation of the template.

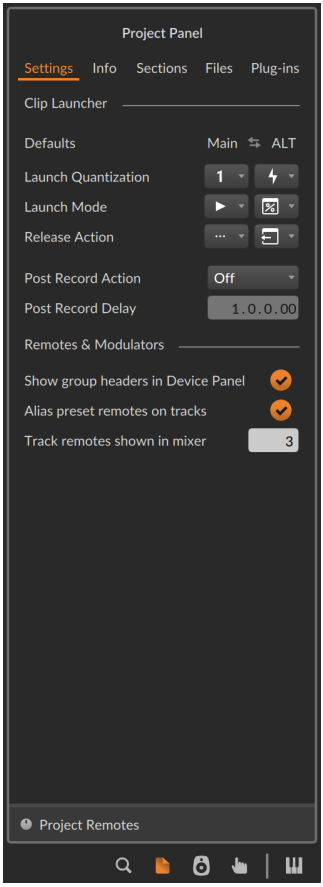
Three basic options can be toggled as well: whether to *Show welcome dialog* when the template is opened, whether the save process *Should collect external files* to the project, and whether the save process should *Should collect packaged files* that are used by the project.

To create a new file from a template: go to the *File* menu and select *New From Template...* (directly beside the *New...* option).

To set a template as the default for any new project: in the *General* tab of the *Preferences* window, find and enable the *Use a template for new Projects* setting under the *Template* heading. Then click on the ellipsis (...) button and select the template file to be used.

14.2. The Project Panel

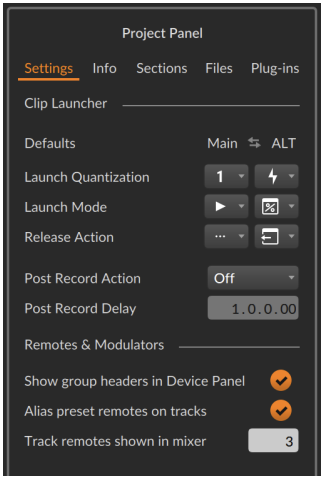
The **Project Panel**, one of the "access panels" in Bitwig Studio, can be shown by clicking the file icon in the window footer.



The purposes it serves are divided over five tabs and a special pane.

14.2.1. Settings Tab

The *Settings* tab contains project-specific settings.



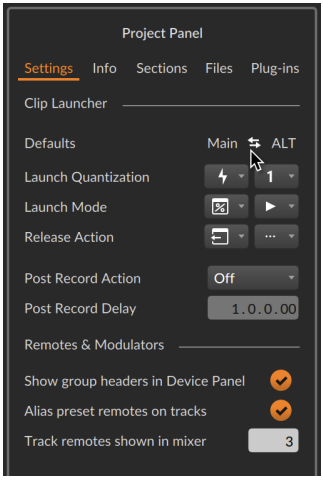
This starts with the *Defaults* for the *Clip Launcher*. These settings represent this project's configurations for both the *Main* and *ALT* trigger actions. All clips are initially set to trust these project-wide settings, representing the same functions described earlier (see section 6.2.5.2 and section 6.3.2).

The default *Main* settings represent a traditional trigger behavior: the *Launch Quantization* waits until the next bar line (*1 bar*), the *Launch Mode* will cleanly *Trigger from Start* of the clip, and when the trigger gesture is released, playback will simply *Continue* whatever it was doing (essentially the null *Release Action*).

In contrast, the default *ALT* settings offer a legato jump into the new clip: the *Launch Quantization* is set to *Off* so the change happens immediately, with the new clip taking over from the previous clip's relative position (as the *Launch Mode* is set to *Legato from Clip (or Project)*), and when the trigger gesture is released, the *Release Action* causes playback to *Return* to whatever was happening before this clip was triggered.

While these two behaviors can be quite expressive, you can of course change these project settings and quickly redefine performance of this song, even while it is playing back.

To swap the project's Main and ALT trigger behaviors: click the stacked arrows icon between the *Main* and *ALT* labels.



A final *Post Record Action* options are available as well, and how long to *Delay* the selected action after a Launcher clip recording has stopped.

Then in the *Remotes & Modulators* section are a few settings related to track-level modulators and remotes, including:

› Whether to *Show group headers in Device Panel* or not (which would hide the headers for group and project levels in the **Device Panel**).

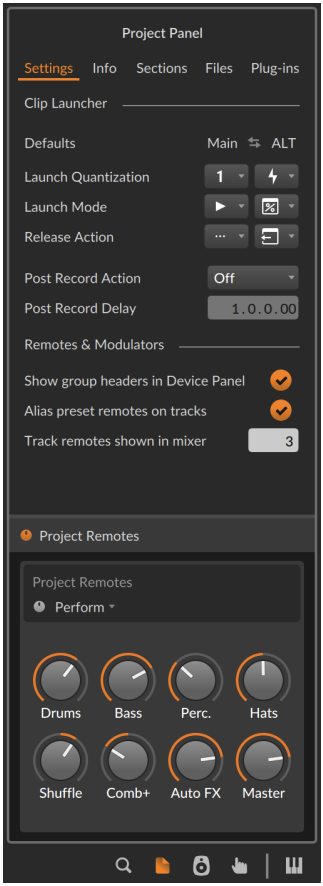
› Whether to *Alias preset remotes on tracks*, which would show certain device remotes in the place of unique track remotes (when none are present).

› The number of *Track remotes shown in mixer*, in case you'd like to limit that section's space or pick a number that matches your controller, etc.

14.2.2. Project Remotes Pane

A special *Project Remotes pane* is available at the bottom of the **Project Panel**.

To show the Project Remotes pane: click the *Project Remotes* icon at the bottom of the **Project Panel**.



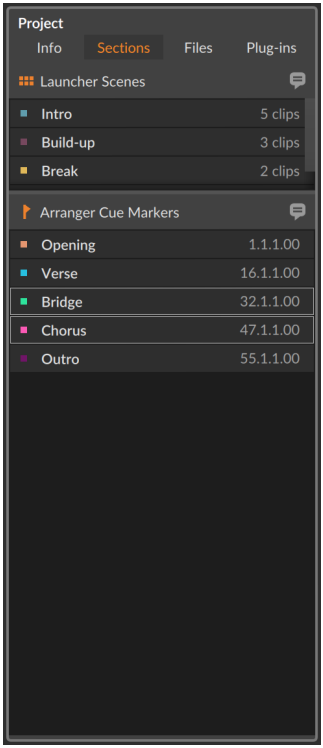
This special *Project Remotes* pane can also be used for mapping or editing your project remotes, and it will stay visible regardless of which tab you are on in the **Project Panel**.

14.2.3. Info Tab

The *Info* tab allows you to fill in several metadata fields to describe your project. While you can use each of these fields as you see fit, their purpose is to help you store information and notes about each project file. Depending on which audio format you choose, some or all of these values might be included as tags in your output files.

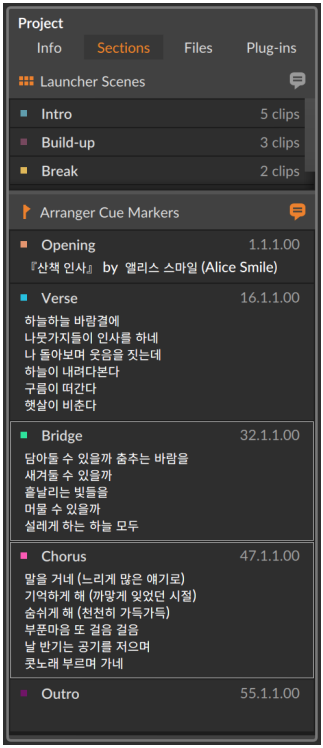
14.2.4. Sections Tab

The *Sections* page shows all Arranger cue markers and Launcher scenes in one list, displaying the name and color of each, as well as the position (for cue markers) and the number of contained clips (for scenes).

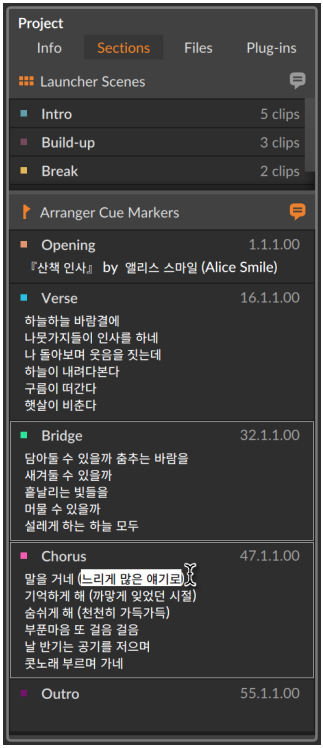


Selections made in this tab act the same as when actual Arranger cue markers or Launcher scenes are selected. For example, pressing [RETURN] triggers whatever is selected. Or right-clicking selected Arranger cue marker(s) allows you to *Loop Selected Region*, setting the Arranger Loop Selector to match the selected marker range.

Additionally, the Launcher and cue marker icons on the left edge can be clicked to toggle their visibility in the panel. And the 'text bubble' icons on the right of each section expand the entries to include space for showing their comments.

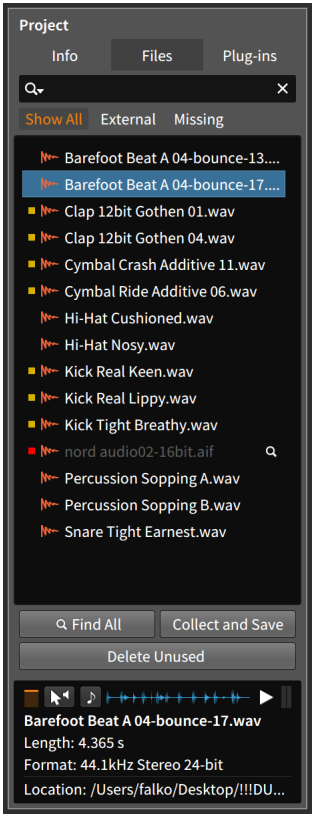


When comments are shown, clicking in each entry's header line selects that scene/cue marker. Clicking in the space below allows text entry of new comments, or even dragging to make text selections and edits.



14.2.5. Files Tab

The *Files* tab lets you view and manage the audio files that are used by the current project.



The central focus of this tab is the list of audio files. At the top of the tab is a search field for narrowing the files being shown based on their name. And when one of the audio files is selected, an *info pane* will appear at bottom. This pane displays information about your file selection and offers a few options for auditioning files, similar to the **browsers** (see section 4.2.4.1).

To the left of each audio file listed is either a yellow square, a red square, or a blank space. This indicates the file's status.

› A file with a blank space to its left is stored within the project's folder.

› A yellow square indicates that the file being used is *external*, or located outside of the project folder.

› A red square indicates that the file is currently *missing* and cannot been found. At the right edge of each missing file is a magnifying glass

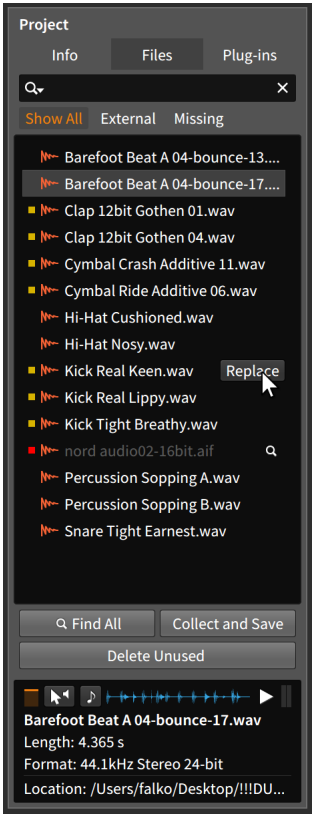
icon. Whenever a project has a missing file, its icon in the project tab section will include an exclamation point (*!*).

Files of all statuses will be shown when the *Show All* view button is enabled (it is the default). The other view buttons, *External* and *Missing*, show only files of those statuses when selected.

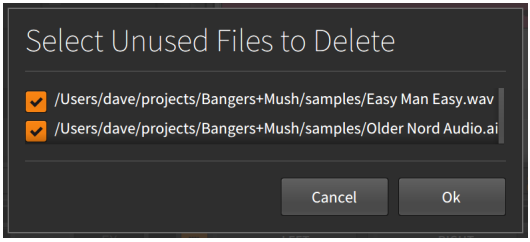
To search for a missing audio file: click the magnifying glass icon to the right of the file's listing. In the open file dialog that appears, navigate to the folder you would like searched, and then click *Open*.

To search for all missing audio files: click the *Find All* button at the bottom of the audio file list. In the open file dialog that appears, navigate to the folder you would like searched, and then click *Open*.

To replace one audio file with another: mouse over the file listing to be replaced, and click on the *Replace* button that appears on the right. In the open file dialog that appears, select the file you wish to replace it with, and then click *Open*.

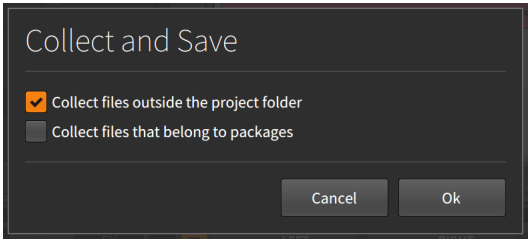


To delete unused files from the project folder: click the *Delete Unused* button at the bottom of the audio file list. In the dialog that appears, uncheck any files that you want to keep, and then click *Ok*.

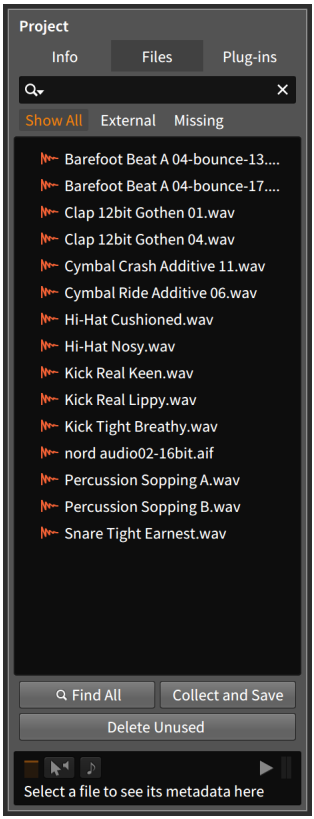


To move external audio files into the project folder: click the *Collect and Save* button at the bottom of the audio file list. In the dialog that appears, select whether regular external files should be collected, and

whether files within Bitwig Studio packages should be collected. Then click *Ok*.

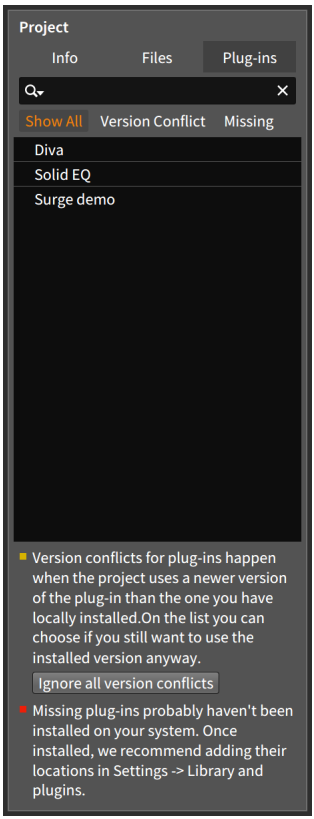


The *Collect and Save* function, found under *File* › *Collect and Save*. Depending on the options you have chosen, you can use this to quickly move all used audio files into the project folder.



14.2.6. Plug-ins Tab

The *Plug-ins* tab lets you view and manage plug-ins that are used by the current project.



This tab is laid out very similarly to the *Files* tab. In this case, the central focus of the tab is the list of plug-ins. There is still a search field above the list. And to the left of each plug-in listed is either a yellow square, a red square, or a blank space.

› A plug-in with a blank space to its left is operating normally.

› A yellow square indicates that the plug-in has a *version conflict*. This means that the plug-in found on your system is an older version of the one that was saved in the project. When this happens, you can try to resolve it yourself, or you can ask Bitwig Studio to ignore the conflict.

To tell Bitwig Studio to ignore all plug-in version conflicts: click the *Ignore all version conflicts* button at the bottom of the plug-in list.

› A red square indicates that the plug-in used in your project is currently *missing* and cannot be found. When this happens, you can manually install the plug-in in question and make sure that the plug-in's location is known to Bitwig Studio (see section 0.2.2.5 for information on the *Locations* page of the **Dashboard**).

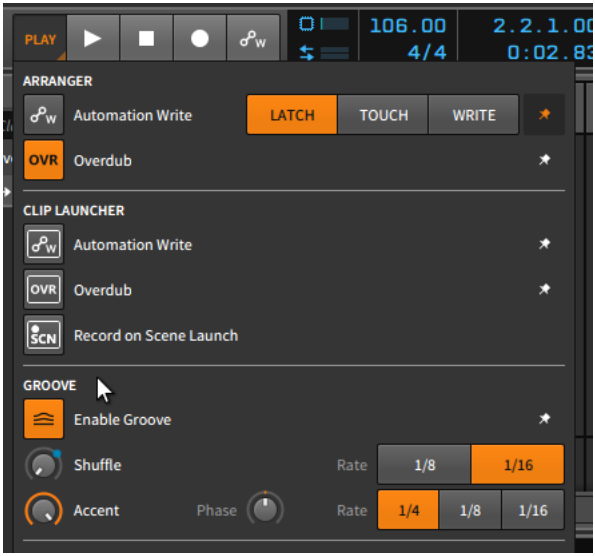
Plug-ins of all statuses will be shown when the *Show All* view button is enabled (it is the default). The other view buttons, *Version Conflict* and *Missing*, show only plug-ins of that respective status when selected.

14.3. The Global Groove

The musical idea of *shuffle* is to take a balanced (or "straight") rhythmic pattern and make every second note of the pattern a little late (or "swung"). The groove function in Bitwig Studio allows you to apply this idea so that notes which were programmed straight can be swung by a variable amount on playback. This function is nondestructive and can be adjusted or disabled at any time.

While each clip has local *Shuffle* and *Accent* settings (see section 5.1.10.6), the groove settings themselves are set at the project level.

The global *Groove* controls are found in the *Play* menu.

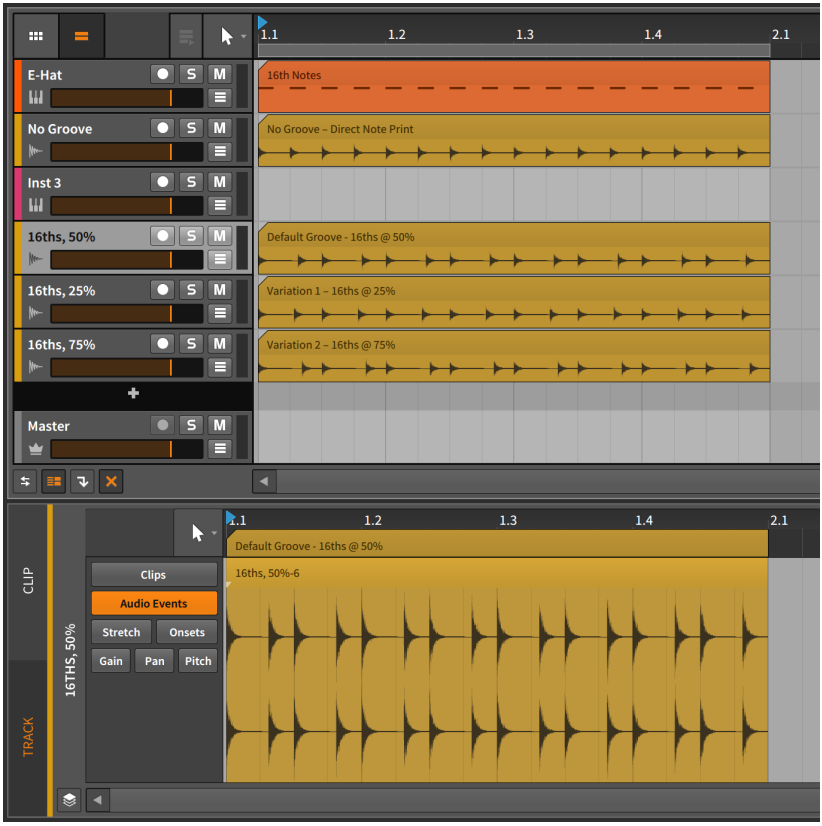


When the *Enable Groove* button is toggled on, the Global Groove settings will be applied to any clip requesting them.

The *Shuffle* category has two settings:

› *Rate* determines whether groove will be applied at the *1/8* note or *1/16* note level.

› The *Shuffle* control itself sets the amount. More specifically, this is the distance (from *0.00%* to *100%*) that even-numbered beats are delayed to the next lower beat division. So if the *Rate* is set to *1/16* notes, the *Shuffle* setting determines how far each second 1/16 note is pushed toward the following 1/32 note.



In the above example, the source track is completely straight 1/16 notes (the *E-Hat* track). The three bottom audio tracks represent that source track printed with various amounts of *1/16* note groove applied.

The **Detail Editor Panel** focuses on the *50% Shuffle* example. Here, you can clearly see that each second 1/16 note is shifted halfway to the following 1/32 note.

The *Accent* category has three settings:

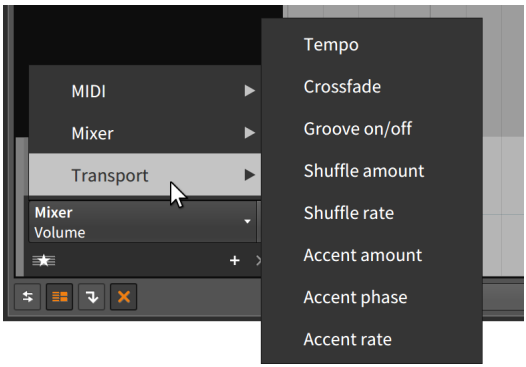
› The *Rate* determines whether a slight emphasis is applied to every *1/4* note, *1/8* note, or *1/16* note.

› The *Accent* itself sets the relative emphasis applied at the set interval. This is set between *0.00%* and *100%*.

› *Phase* sets an offset amount that the accent interval is shifted by. This is set between *-50.0%* and *50.0%*.

Note

All of these groove parameters can be automated in your project's master track under the *Transport* category. You can also automate the project's tempo from here.

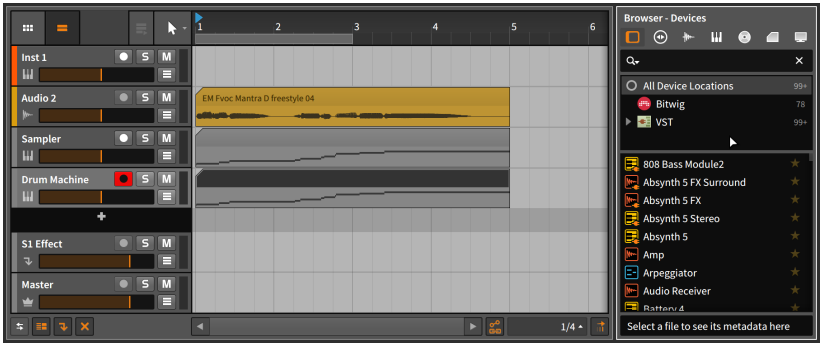


14.4. Working with Multiple Projects

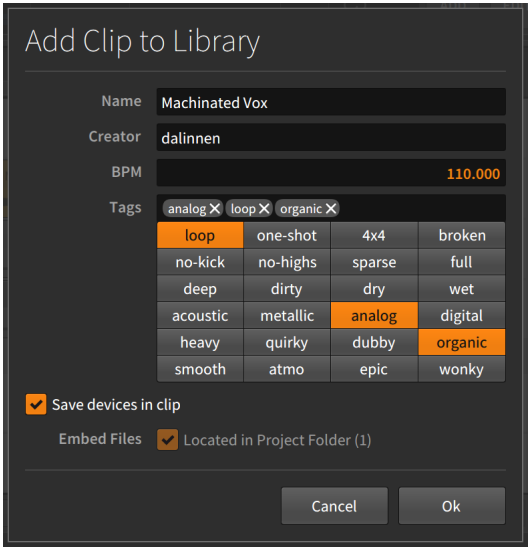
Bitwig Studio makes it quite easy to get your work from one project into another. This can be done either by storing your own library content via the **Browser Panel**, or by directly transferring data between open projects.

14.4.1. Adding Clips to the Browser Panel

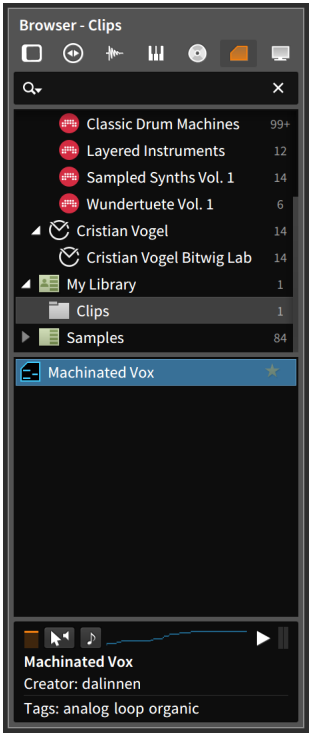
To add a clip to the library: click and drag the clip into the **Browser Panel**. In the dialog that appears, edit the clip *Name* if desired, enable any appropriate *Tags*, and then click *Ok*.



It doesn't matter which tab of the **Browser Panel** is showing when you drag your clip over. In fact, the **Browser Panel** doesn't even have to be called up before you begin dragging as you can call up any panel while using the mouse. In the case of the **Browser Panel**, you can press [B] any time to call it up.



Once you have stored your clip, it can be found and managed from the *Clips* tab of the **Browser Panel**.



Any clip stored in this fashion also contains its own parameters, the track's device chain, and any automation data.

14.4.2. Going Directly between Projects

Bitwig Studio allows you to have multiple projects open at the same time, with each open project represented in the project tab section of the window's header (see section 2.1.1). In addition to making it easy to quickly switch between projects, this also allows you to copy data between them.

To transfer a clip(s) from one project to another: select and copy the clip(s) in the original project. Switch to the destination project, move the playhead to the desired insertion point (this can be done by clicking on either a Clip Launcher slot or at the position within the Arranger Timeline), and then paste.

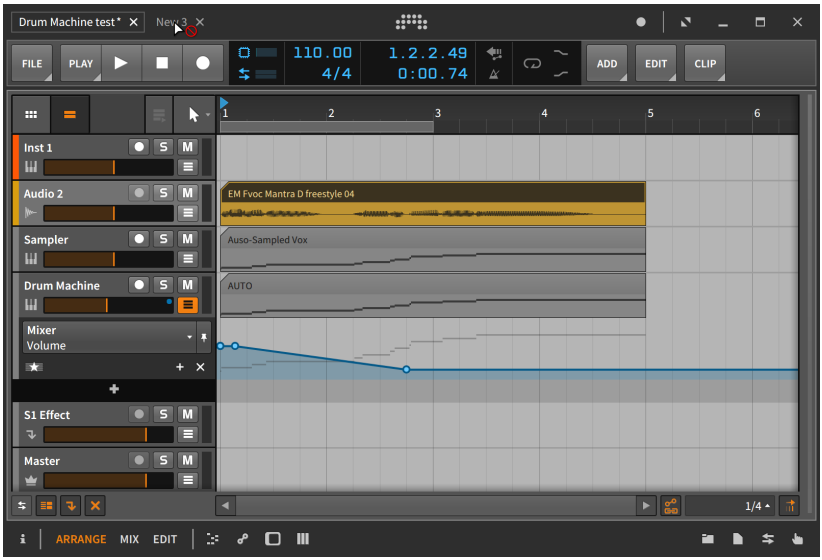
Note

Copying and pasting clips within a project will maintain the original clip's automation but not its device chain. Copying and pasting clips between projects will maintain neither.

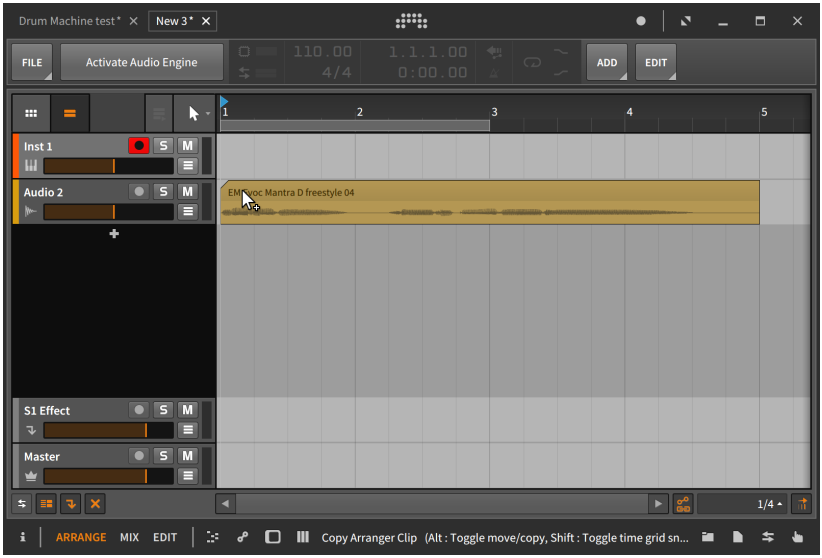
To transfer a device(s) from one project to another: select and copy the device(s) in the original project. Switch to the destination project, select the target track, and then paste.

The other option is to drag items directly from one open project to another.

To transfer an item(s) between two open projects: click and drag the item(s) from the original project to the target project's tab. While still holding the mouse, wait for the target project to load, and then drag and release the item in the appropriate location.



The cursor that includes a circle with a diagonal line thru it indicates that releasing your item(s) on the project tab itself would do no good. Very quickly, the target project will load.



Note

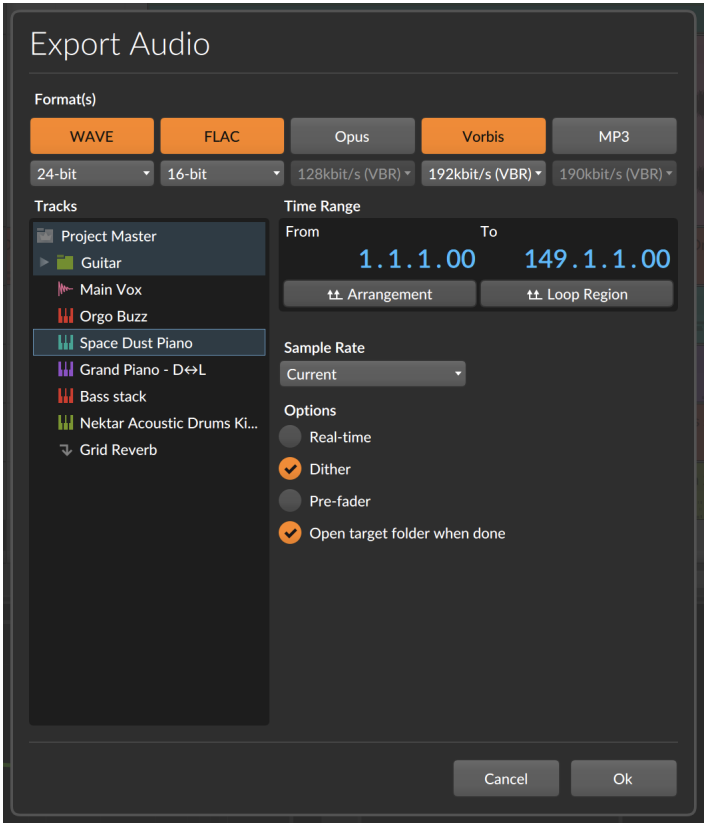
Dragging clips between projects will maintain only the clip, not the automation or device chain. Device(s) can be transferred separately using the same method.

Note

If you want to copy multiple tracks from one project to another, you can use the method above with multiple tracks selected. You could also encapsulate all desired tracks into one group track, transfer that one group track to the second project, and then ungroup the track (see section 3.2.2).

14.5. Exporting Audio

To export audio from Bitwig Studio, use the *File › Export Audio...* function. When this option is selected, a dialog box with several sections appears.



› The *Format(s)* section selects which audio formats will be output for each selected track. If multiple formats are selected, than each exported track will be saved multiple times. Beneath each audio format is a menu of export profiles to choose from.

› The *Tracks* section lists all activated tracks in the project. Select each track that you want to separately exported. Note that group tracks are shown as folders that can be select, and/or they can be unfolded to access individual child tracks. And if you want to export the entire project (a completed song, for instance), check the *Project Master* from the top of the list.

› The *Time Range* section determines what portion of the project will be exported. Both the *From* and *To* parameters are set using song positions. If a time selection is present in the project, that period is used by default. You can also click to select the full *Arrangement*, or the Arranger *Loop Region*.

› The *Sample Rate* setting determines if any conversion happens on export. The default setting of *Current* keeps the audio engine's current sample rate with no conversion.

› The *Options / Format* section gives you four more settings.

  › *Real-time* overrides the standard offline bounce to execute it in realtime. This is useful when you have external audio paths being used live, etc.

  › Select *Dither* to add a very small amount of noise to your exports. This can help lower-resolution export files best match the high-resolution internal signals of Bitwig Studio. This noise is generally inaudible.

  › To ignore all mixer volume automation, select the *Pre-fader* option. This can be especially helpful when exporting stems.

  › After bouncing, *Open target folder when done* will point your file manager application to the folder where files were written.

Once the *Ok* button is pressed, the files will be created.

Note

Only Arranger Timeline selections (not Launcher clips) can be exported in this way.

The *Export Audio* function dialog uses a current selection for its default settings. So if you want to export only a single clip from one track, first select that clip and then choose *File* › *Export Audio…* .

14.6. Exporting MIDI

To export MIDI from Bitwig Studio, choose *File* › *Export MIDI…* . In the save file dialog that appears, set the desired name and location for your MIDI file. This file will include all notes present in your project's Arranger Timeline, organized by track.

14.7. Exporting Projects

To export a DAWPROJECT file from Bitwig Studio, choose *File* › *Export DAWproject…* . In the save file dialog that appears, set the desired name and location for your DAWPROJECT file. This will save all of your

generic project data into a file, which can be opened by any other music software that supports the format (more information here [https://www.bitwig.com/support/technical_support/dawproject-file-format-faqs-62/]).

15. MIDI Controllers

MIDI controllers — or simply *controllers* — can be a critical part of any production environment or performance setup. Bitwig Studio supports MIDI controllers in general, whether you are playing in notes or you are mapping physical knobs and sliders to the program's parameters.

Bitwig Studio comes with various *controller scripts*. Each script is programmed for a specific MIDI controller, with a few scripts for *Generic* controllers of any make.

For the generic controllers, functionality is basic. If the controller has keys, you can send note messages. And if it has assignable knobs, you can map those knobs to any mappable control in Bitwig Studio.

For the controllers that are specifically supported, more functions are allowed. This can include control of track mixer functions, device remote controls and parameters, the transport, clip launching, and more. As each controller can vary greatly in size, shape, and functionality, the built-in mappings supported by Bitwig Studio also vary from controller to controller.

Note

Anyone with knowledge of Java or JavaScript and the MIDI specification can customize any of the included controller scripts or even write their own. For full details on Bitwig Studio's controller API, go to the **Dashboard**, click on the *Help* tab, and then click on the *Documentation* page. Various *Developer Resources* can be found here.

This chapter covers how to use both the default mappings for your controller (if supported), and how to manually assign and manage MIDI mappings. It also shows how to achieve simple parameter and controller (or computer keyboard) pairings via the **Mappings Browser Panel**.

15.1. Soft Control Assignments

For any controller used with Bitwig Studio, certain default behaviors are available. We will start by revisiting the **Dashboard** for additional controller settings and documentation. Then we will get to know the Remote Controls pane, which is available on every device.

15.1.1. The Remote Controls Pane

As we discussed in chapter 8, all actual device control elements are found within the **Device Panel**. In this section, we will revisit the **Device Panel** to see how it facilitates soft control assignments.

"Soft control assignments" refers to controller assignments that can dynamically shift, following your focus on different tracks and devices within a project. By default, this functionality only targets the currently selected device.

In the example above, the **Delay-2** device is currently selected, as its slightly brightened device header indicates. By clicking on the **E-CLAP** device, it will become the currently selected device and receive focus. If a recognized MIDI controller is connected and set up in Bitwig Studio, the device's interface might even get a splash of color.

The colored interface items represent the eight current soft control assignments. The details of these mappings are available in the device's Device Mappings pane, which is shown when the *Remote Controls button* (which looks like a group of six controller items) is clicked.

The *Remote Controls pane*shows the *soft control assignments* that come with being the currently selected device. Each assignment is represented here by a color-accented controller. And since your controller's eight hardware controls will be used over and over again, they are always colored in rainbow order (red, orange, yellow, green, cyan, blue, indigo, and violet) to help you mentally connect each particular hardware control with its ever-changing software assignment.

Note

Depending on the type of parameter under control, either a knob, button, or chooser (a drop-down menu, indicated with a downward-facing arrow on its far right) will be used.

Remote controls work similarly on the track level, regardless of whether you are on a "low-level" audio or note track, somewhere in a nested group track, or on the project level with the master track.

If a track has no remote control page, preset or user-created remotes for the best matching device can be aliased to the track level (see section 14.2.1). But you are always free to create new remote control pages at any of these levels. Otherwise, all the rules for device remote controls apply on the track level as well.

To rename a soft control: double-click the soft control's name. If no name is provided, the name of the parameter under control will be used.

Clicking on the *Remote Control Pages menu* exposes all current mapping pages.

There are three different types of mapping pages:

› *Preset pages* are sets of remote controls tied to this particular device instance or preset.

› *Device pages* are sets of remote controls linked to every device of this kind across your installation of Bitwig Studio. So any changes made to this particular **E-Clap** device's device pages would be read by all **E-Clap** devices.

› *Modulator pages* represent remote controls for any modulators loaded in this preset. They are tied to the particular modulator in use and are not editable.

To create a new preset page: click on the Remote Control Pages menu, and then choose the *Add New Page* option.

The Wi-Fi icons represent controls which are not yet assigned.

To make a soft control assignment: first click on an available controller's Wi-Fi icon, and then click on the device parameter you wish to assign to it.

We can now switch to other remote control pages (via the Remote Control Pages menu) and then return to this preset page. This new preset page was named *Perform* by default, but similarly to the soft controls, you can click on the preset page's name in order to rename it.

Clicking the Remote Controls Editor button causes the *Remote Controls Editor* to appear in the central panel area.

You will notice that the Remote Controls pane itself is now showing an unassign button (as an *x* icon) in the top right of each assigned controller.

To remove a soft control assignment: click the assigned controller's unassign button, either in the Remote Controls pane or in the Remote Controls Editor.

Note

A soft control assignment can also be removed directly from the **Device Panel** either by:

› Right-clicking in the soft control's area, and then selecting *Delete Remote Control* from the context menu.

› Holding [ALT] and mousing over the Remote Controls pane will cause an unassign button (the little *x* icon) to appear in the top right of each assigned soft control. Continuing holding [ALT] and click on any of these buttons to remove that assignment.

The Remote Controls Editor is scrollable. Changes made to pages in the *Preset Pages* section are stored immediately. Changes to pages in the *Device Pages* section must be saved, either by clicking the *Save and Apply* button or by using the save dialog that appears when you close the Remote Controls Editor.

To reorder remote control pages: click and drag the pages within their section.

To duplicate a remote control page: hold [ALT], and then click and drag the page you wish to duplicate within its section.

Note

Pages cannot be moved or copied between sections.

To rename a page: double-click the current name of the page.

To add tags to a remote control page: click in the bottom row of the page area, beneath the bottom four soft controls.

To add a ninth slot to any remote controls page: right-click on the title bar of the remote controls page in the Remote Controls Editor, and then select *Allow 9 Slots* from the context menu.

This can be especially handy if you are using a MIDI controller with nine faders.

To create a new remote control page: click the *Add Page button* (the large plus *+*) icon at the end of the *Device Pages* section. (The plus sign at the end of the *Preset Pages* section can also be used to create a new preset page.)

To delete a remote control page: click the *Delete Page button* (the *x* icon) to the far right of the page name.

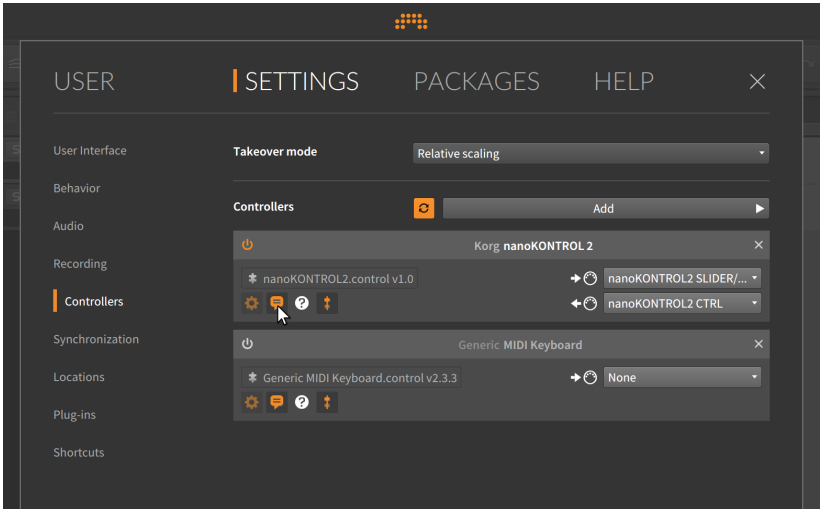
Before we move on, let's consider a use of the rainbow order in another context. Most controllers that support soft control assignments can also support a "mixer mode."

The following images demonstrate a project in **Mix View** both without and then with mixer mode engaged:

Notice how the volume and pan knobs for the first eight tracks are using the first eight soft controls, as shown in rainbow order again.

15.2. Controller Visualizations, Takeover Behavior, and Documentation

Earlier, we saw how to recognize our MIDI controllers under *Settings* > *Controllers* in the **Dashboard** (see section 0.2.2.2). Let's now go back to the **Dashboard**, the *Settings* tab, and the *Controllers* page to examine the individual controller options.



In addition to port mappings and script information, each controller entry also has a set of icons in the bottom left that relates to its use.

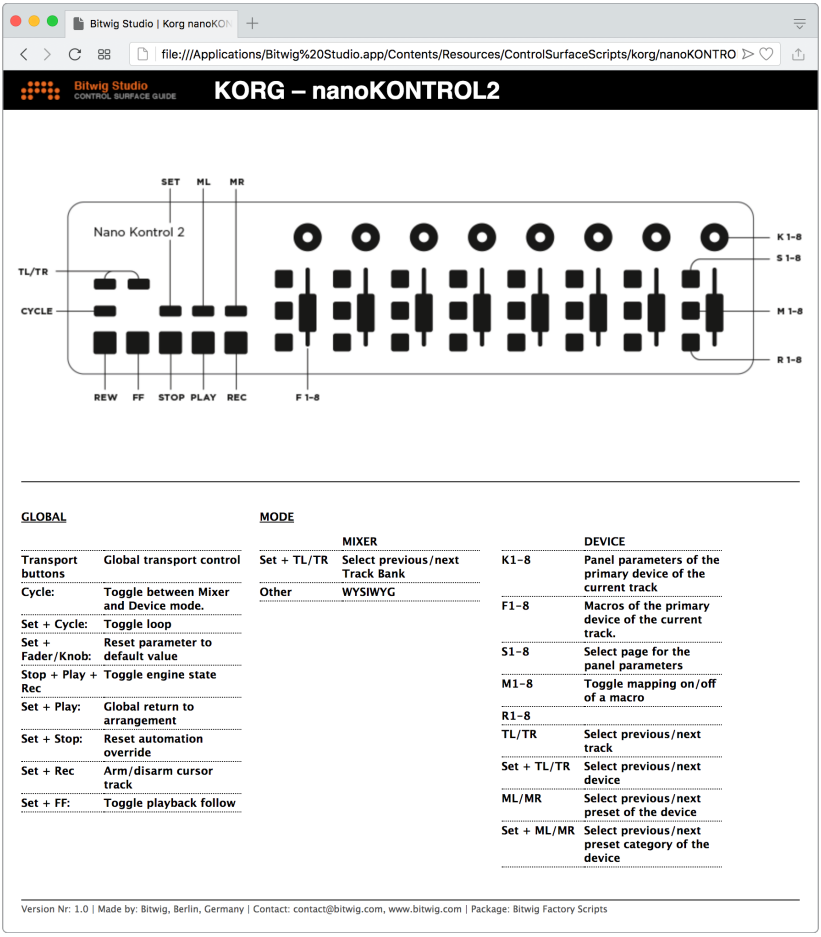
› The gear button toggles the visibility of additional settings, if such settings are specified by the controller script in use.

› The "speech balloon" button toggles on-screen controller visualizations (see section 2.2.5).

› The fader button toggles whether the global *Takeover mode* (set at the top of the page) is applied the controller in question or not. Takeover modes set the behavior of how incoming messages from individual controls are used by their associated software parameters. Modes include:

  When a controller has disabled use of the global takeover mode, it is the same as being set to *Immediate*.

› The question mark (?) button provides a link to documentation for the particular controller script in use.

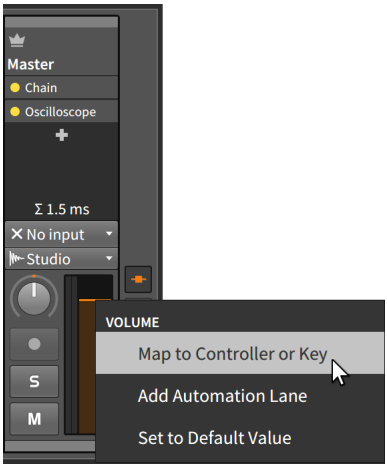


15.3. Manual Controller Assignment

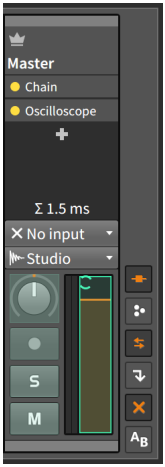
Any device with assignable hardware knobs/faders allows manual assignment of these controls to project parameters, such as device parameters or track mixer elements.

To make a manual controller assignment: right-click on the parameter you wish to assign and select *Learn Controller Assignment...* from the context menu. The targeted parameter will now be framed in bright

green with an animated circle icon, indicating that you should "turn a knob." Then move the hardware control you wish to assign.



In this example, we've right-clicked on the master track's volume control.



After you move the hardware controller in question, the software parameter will return to its normal appearance, but the on-screen control will be moving as you move the physical control.

To remove a manual controller assignment: right-click on the parameter you wish to unassign, and click the *x* icon to the right of the mapping you wish to delete.

Finally, if you are using soft control assignments, you can still make manual controller assignments. In this situation, any new assignments will override soft assignments that usually work in the current mode.

As an example, let's start from the "mixer mode" case from the end of the last section.

By manually assigning the master track's volume and pan controls to my hardware controller's eighth fader and knob, both of those master track controls will be colored violet, and the track that was previously using those controllers (*FX Storm*) will lose them.

In this example, my last fader and knob will always control the master track while my controller is in mixer mode.

15.4. The Mappings Browser Panel

The **Mappings Browser Panel** is another one of the "access panels" in Bitwig Studio. When the panel's view toggle is pressed, the panel itself slides into (or out of) view, exposing all preexisting mappings and allowing you to either edit or delete them. But unlike when other panels are visible, this one also changes the appearance of the project itself.

Before getting to functionality, the first parameter in the panel is worth noting. The *Map source priority* setting decides whether any *Controller script* in use should get the first chance to process incoming MIDI messages (potentially overriding mapping established here) or if incoming *MIDI* should be handed over raw to the any active mappings in this panel.

While the **Mappings Browser Panel** is visible on screen, any parameter of your project that can be mapped appears with a green overlay. Clicking on one of these parameters causes a pair of arcs to spin circularly backwards and forwards, indicating that this parameter is ready to be mapped.

The next computer key that is pressed or MIDI control that is touched will now be assigned to the selected parameter. In this example, we have selected the volume fader of the master track. If we now move a MIDI controller that is sending continuous controller 7 messages, the volume fader will display this as long as the **Mappings Browser Panel** remains visible.

16. Modulators, Device Nesting, and More

We have talked about and dealt with devices all thru this document. As we have seen, it's quite possible to operate devices in all the normal ways without delving into their advanced functionality. In this chapter, however, we'll explore device capabilities that are deeply powerful and generally unique to Bitwig Studio.

The aim of this chapter is not to educate you on any particular device or its parameters. While we will examine a few devices here in detail, our purpose is primarily to investigate concepts that are relevant to many devices. A separate reference section on the Bitwig devices themselves can be found in chapter 19.

In this chapter, we will investigate nested device chains, we will examine Bitwig Studio's unique **Unified Modulation System** (and the modulators that it supports), and we will take note of some of the advanced plug-in options provided.

Congratulations; we've made it to the deep end of the pool. Now take a big breath.

16.1. Nested Device Chains

We discussed long ago how each track has its own device chain. Since then, there have been references to "top-level devices," meaning the devices that are directly in a track's device chain.

Most of the Bitwig devices actually possess one or more device chains of their own. These lower-level device chains, or *nested device chains*, solve several problems inherent to software-based music production.

For one thing, a single preset can contain vast configurations of devices, from a standard single device to something far more ornate. For another, the idea of nesting devices allows for unique signal routings that aren't usually possible in software, such as blending serial and parallel structures across a single device chain.

But we will return to device chains in a moment. Since the idea of parallel signal structures has already been mentioned, we should start this discussion with the humble, crucial *Mix* knob.

16.1.1. The Mix Parameter

For many audio effect processes, it is critical that the original, unprocessed sound is blended together with the affected sound. A good

example is a simple delay effect. Hearing the original sound provides context for the delayed copy that follows. (A simple delay effect with no original sound mixed in could be better described as "late.")

To facilitate this blending, the idea of a *wet/dry* control is common in audio effects. This is usually implemented as a single knob that cross-fades between purely "dry," unprocessed signal at the minimum value, and purely "wet," post-processed signal at the maximum value, with every value in between representing a gradual blend of the two.

In Bitwig devices, this function is found on many devices via a parameter called *Mix*.

In the above example, we are using the **Freq Shifter** audio FX device, which is a frequency shifter. With the *Mix* parameter set to *33 %*, a third of the device's output is the result of the frequency shifting process. This means that the signal received by the device (before any effect is applied) makes up the remaining two-thirds of the output, for a 2:1 blend of dry to wet signal. If *Mix* was set to *66.6 %*, the balance would be reversed, with wet signal predominating at a 2:1 ratio.

So when you find a *Mix* parameter knob in the bottom right corner of a Bitwig device, it is providing this same wet/dry, parallel processing structure. In any of these cases, a *Mix* setting of *100 %* would produce an output with no truly dry signal, and a setting of *0.00 %* would effectively bypass the device by outputting only dry signal.

Note

If you find a *Mix* parameter knob that isn't in the bottom right corner of the device, it is carrying out a different function that is specific to that particular device.

Finally, *Mix* is not exclusive to audio FX devices and can be found on some devices in nearly every category. In the categories that don't use this *Mix* parameter (note FX and instruments), any incoming audio is generally passed directly to the audio outputs.

16.1.2. Container Devices

After starting with a simple in-line routing control, we will move on to nested device chains. And we will start with devices that are made to provide parallel device chains.

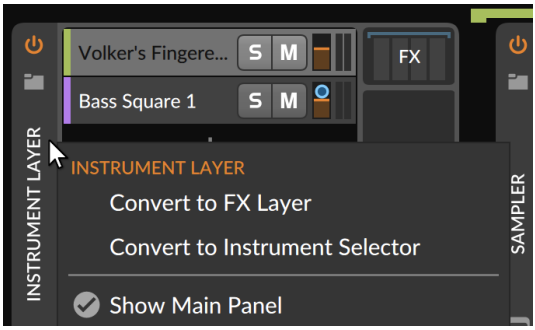
Container devices are utility devices whose primary function is to host other devices. So while most devices contain some type of nested device chain, container devices couldn't exist without them.

Three particular container devices — (**Drum Machine**, **Instrument Layer**, and **FX Layer**) came up in passing when we first saw the mixer's track fold button (see section 7.1.1), and the two "layer" devices reappeared indirectly when we discussed dragging devices to layer them (see section 8.3). Each of these devices allows for a large number of device chains within them.

Note

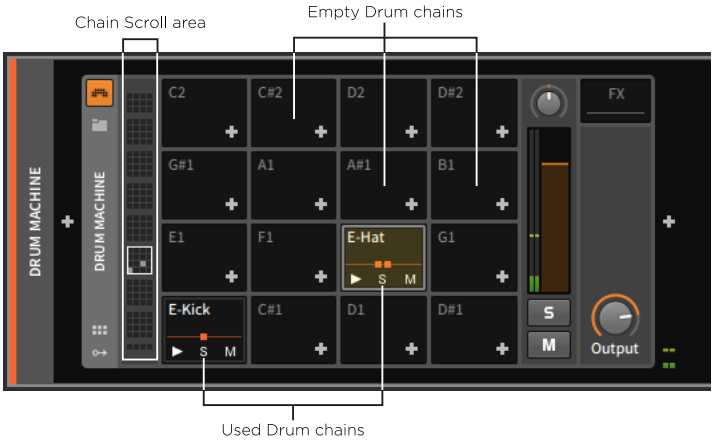
Among the container devices, the layer family exists for sending signal to multiple note effects (**Note FX Layer**), instruments (**Instrument Layer**), or audio effects (**FX Layer**). There is also a family of *selector* devices (**Note FX Selector**, **Instrument Selector**, and **FX Selector**) for sending signal to only one device at a time in a controllable fashion (see section 19.4.5).

When using devices in the layer and selector families, you can always right-click on the device header for various *convert* options, as shown here on the header of **Instrument Layer**.



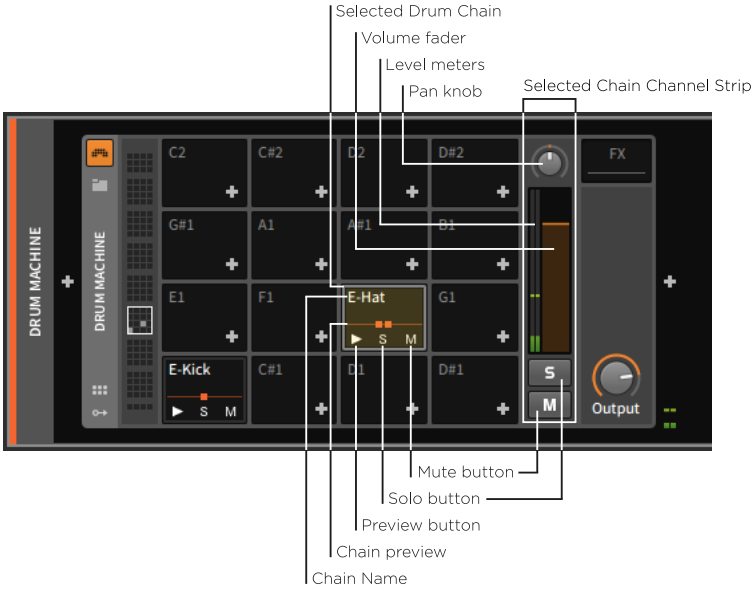
16.1.2.1. Drum Machine

Drum Machine is made to house multiple instruments, each of which will be triggered by a specific note message (for example, C1 for a kick drum, F#1 for closed hi-hat, etc.).



Corresponding with the 128 possible MIDI notes, **Drum Machine** offers up to 128 device chains, each called a *drum chain*. 16 drum chains are displayed at a time, and the *chain scroll area* on the left allows you to click or scroll the focus to a different set of chains.

An empty drum chain simply displays the note that it responds to and an *Add Device button* (+) for loading a device directly into that chain.



Used drum chains each have their *chain name* listed at top, and at bottom are a *preview button*, a *solo button*, and a *mute button*.

To the right of the displayed drum chains is the *selected chain channel strip*. Whichever drum chain is selected is surrounded by a blue-green border, and this area of the device provides a small channel strip for that chain, including larger solo and mute buttons, a *volume fader*, a *pan knob*, and *level meters*.

Every used drum chain also has a small *chain preview* displayed across its middle. This central line with squares placed along it is a silhouette of the drum chain, with the squares representing the number of devices currently at the top-level of the drum chain.

Note

While only so many squares fit within this small chain preview area, additional devices may be added to the drum chain.

To view an individual chain: click the chain.

What can now be seen is the drum chain itself, which is, again, a device chain. The two squares from the chain preview were representing these **E-Hat** and **Delay-1** devices, which have the exact same interfaces we are accustomed to.

With the drum chain fully expanded, note that the selected chain is now ringed by a dusty blue frame. The devices within this chain also have a downward-facing bracket above them, both showing the boundaries of the chain contents and connecting these contents to their source by using the same highlight color in both places.

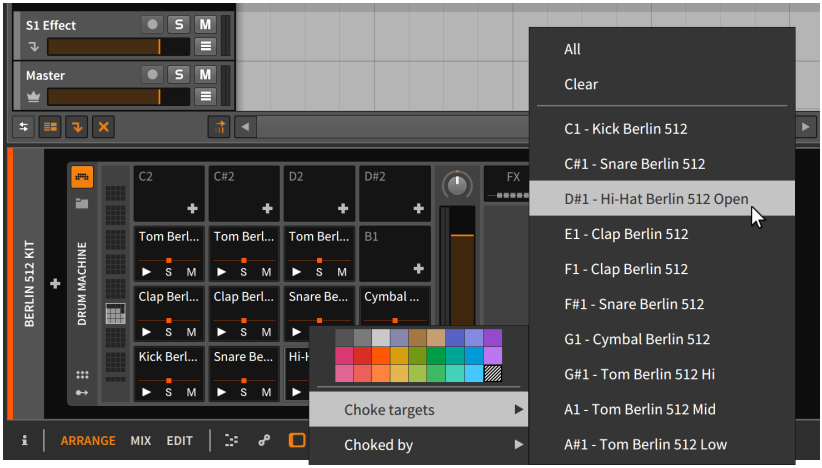
To reiterate this idea, the **Delay-1** device is currently within this drum chain. This means that only this particular instrument (triggered by F#1) will have this device applied to it.

If I were to move this device to the right and out of the drum chain, it will now be in the track's device chain just after the **Drum Machine**.

Accordingly, all audio coming out of **Drum Machine** is now being affected by **Delay-1**.

One other function unique to the **Drum Machine** container device is its ability to have certain triggered notes cut off, or "choke," other notes. This allows you to associate related elements into a single *choke group*, allowing only one of those elements to sound at a time. A classic choke group example are hi-hat elements of a drum kit, where triggering a closed hi-hat sample should silence an open hi-hat sample that was playing. But many other uses can be imagined.

To assign a chain's choke target: right-click the chain in question, and then from the *Choke targets* submenu, select the chain you want to be choked when the current chain is triggered.



To assign a chain as a choke target: right-click the chain in question, and then from the *Choked by* submenu, select the chain you want to cause the current chain to be choked.

These two equivalent options allow you to create a choke group relationship either from a source or destination perspective. But also note what this unique interface implies: that chain A could choke chain B, but chain B could allow chain A to continue playing.

16.1.2.2. Instrument Layer

Instrument Layer is made to house multiple instruments, all of which will be triggered by any incoming note message. The general effect of this device is to make layered sounds or "stacks."

The chains in this device can be called *instrument chains* or layers. Each is still representing a full device chain, but unlike **Drum Machine**, there is no set number of chains. Because of this, there is only one *Add Device button* in the main interface of **Instrument Layer**, with each added device being placed on a newly created instrument chain. If enough layers are added, the chain list itself can be scrolled vertically.

Each layer has its own built-in channel strip, quite similar to each track header in the **Arranger Timeline Panel**. Also as in the Arranger, the selected layer is given a silvery tint.

Note

Also similar to instrument tracks, each layer has settings in the **Inspector Panel** to control the *Channel* that messages are being heard *From* (see section 5.3.2.2). This allows you to set up multitimbral **Instrument Layer** devices, where a single track could trigger different layers by placing notes and other messages on different channels.

16.1.2.3. FX Layer

FX Layer is virtually identical to **Instrument Layer** except it is made to house a layer of FX layers.

16.1.3. Other Common Device Chain Types

There are several other types of nested device chains within Bitwig Studio. Some appear rarely or only once, but a few are reused multiple times.

Some of the most common types of nested device chains include:

› *FX* (or *Post FX*): A nested device chain for processing the device's entire audio output. The only difference between placing effects in this device chain instead of after the device is that this chain is fully stored with this device, which makes moving the device along with its modifiers (or saving presets) much easier. This chain type is mostly possessed by instruments and containers for instruments.

Post FX chains work in exactly the same way, but tend to show up on devices where other chains occurred first.

› *Pre FX*: A nested device chain for processing signal immediately before it enters the device.

› *Wet FX*: A nested device chain that processes only the wet portion of the device's output. The dry signal skips this chain and is mixed back in afterward. All devices with this chain also have *Mix* parameter knobs.

› *FB FX*: A nested device chain that is placed within the device's feedback loop. This is common on delay devices.

Note

Just like Bitwig devices, plug-ins can be used in any device chain at any level.

16.2. The Unified Modulation System

In sound synthesis, *modulation* is the idea that one component can influence another in a controlled way. For a simple musical example, think of vibrato (the subtle bending of pitch back and forth). To achieve this with synthesis, we often connect the output of a low-frequency oscillator (LFO) to a pitch input of an oscillator. The frequency of the LFO determines the rate of the vibrato, and the level of the LFO's signal determines the depth of the modulation.

Modulation can lead to elements that automatically change over time, based on assigned parameters and preexisting control sources. Some would say that modulation leads to more interesting and efficient results in sound programming. These are both good points.

In the days of modular hardware synthesizers, each modulation was highly visible as it was achieved by a patch cord properly connecting two modules. But in our era of computer-based music production, we see knobs on screen far more often than patch cords, and assigning (or even showing) modulations has become a real challenge. Many different interface models have been attempted, but no standard has been found.

Bitwig Studio has its own unique, program-wide method for dealing with modulations. This *Unified Modulation System* allows you to easily assign and edit modulations (so you don't get stuck with fixed modulation routings). It also preserves parameter control as often as possible (so the modulated parameter's knob can still be used, allowing you to easily shift the modulation range). Even the current value of a modulated parameter is visible with this **Unified Modulation System**.

In this section, we will explore the **Unified Modulation System** by learning to work with Bitwig Studio's unique modulator devices. We will then see the same principles used to assign modulations within an instrument.

16.2.1. Modulator Devices

While most systems require us to work around a fixed number of modulation sources — say, two LFOs, three envelope generators,

perhaps some control over keytracking, and maybe a sidechain source for using external audio or note messages — these choices tend to be arbitrary and formulaic from the user perspective. Some sounds require no LFOs, and some require ten. In Bitwig Studio, these options are left completely open for the user.

Modulator devices are special-purpose modules that are made to be loaded into any device. Their purpose is to allow a particular method of control over device parameters. Types of modulators include:

› Mappable interface controls, such as **Button**, **Buttons**, **Macro-4**, and **Macro**.

› Standard sources of modulation signals, such as **4-Stage**, **ADSR**, **AHDSR**, **Beat LFO**, **Classic LFO**, **LFO**, and **Steps**.

› Methods for using incoming MIDI and note messages, such as **Keytrack**, **MIDI**, and **Note Sidechain**.

› Use of external signals for modulation, such as **Audio Sidechain**, **Envelope Follower**, and **HW CV In**.

› Options for using one signal to be split across multiple destinations, such as **Select-4**, **Vector-4**, **Vector-8**, and **XY**.

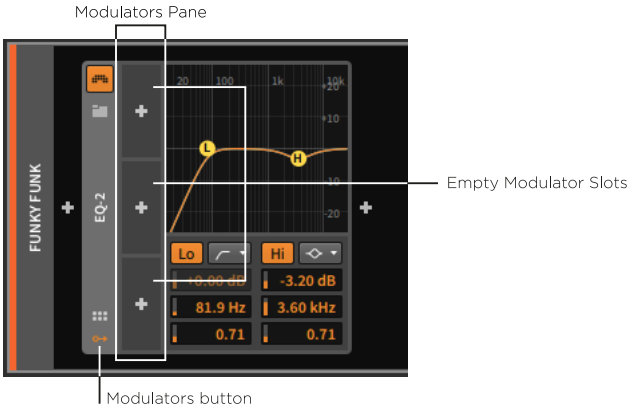
› Interesting ways of blending control signals to create a signal modulation source, such as **Mix** (for crossfading two levels or signals) and **Math** (for creating more complex relationships).

› A little bit of chaos, with **Random**.

Note

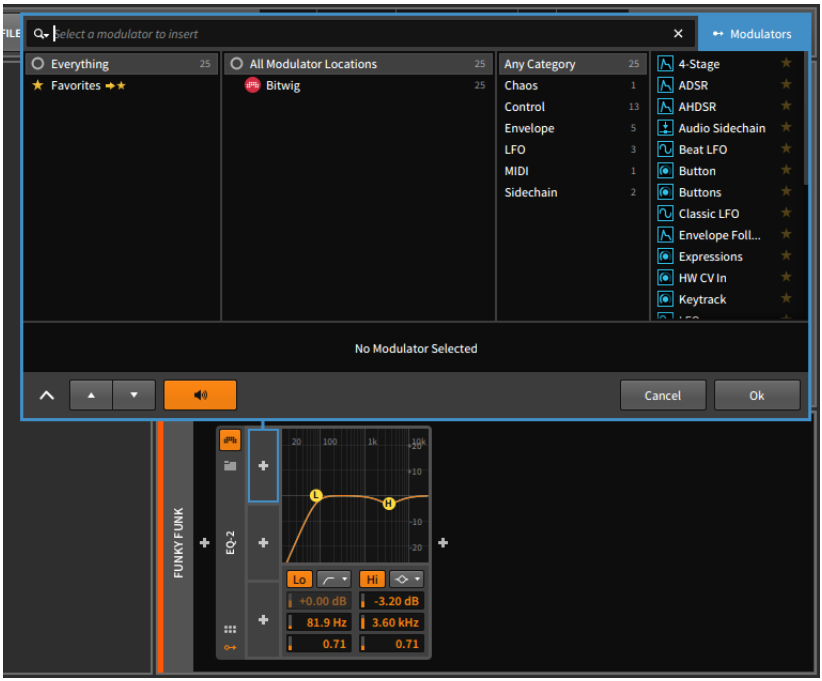
For descriptions of the modulator devices, see section 19.27.

If you click the modulators button, the *Modulators pane* is toggled to be visible.

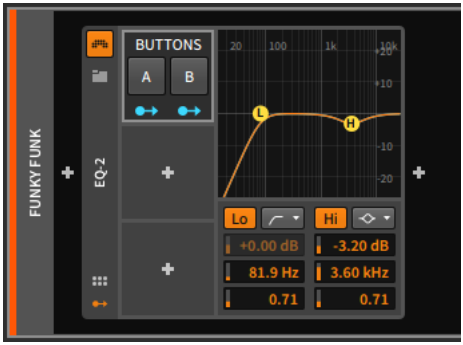


An empty modulator pane loads with three available modulator slots. If all modulator devices are loaded into all three slots, another three slots will appear, and they will keep appearing as often as you run out.

In the center of each modulator slot is an *Add Modulator button*. Clicking this button calls up a special version of the **Pop-up Browser**.



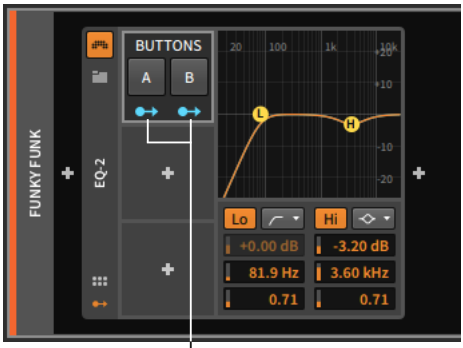
As we discussed before, the **Pop-up Browser** is context-sensitive, providing the most relevant options for the place we have invoked it. Calling it up from the modulator pane provides only modulator devices. Otherwise, the **Pop-up Browser** is working as we would expect it to, providing categories for the available devices and previewing any selected device in the **Device Panel**. And again, clicking *Ok* places the selected device onto the track.



To cut, copy, paste, duplicate, or delete a modulator: right-click the background of the modulator slot, such as the modulator's name.

Also note that the option to toggle modulators as *Active* or not is also present in this context menu. This is a good way to "bypass" a modulator for a moment.

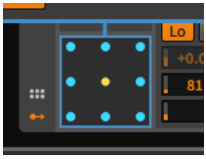
In this example, we have selected the **Buttons** modulator. This device provides two controls which can be toggled to manipulate any assigned parameters. As each button is separately assignable, this device has two *modulation routing buttons*.



The modulation routing button resembles an output port with a patch cord coming out of it, awaiting connection. Clicking a modulation routing button switches to a mode where you can select as many destinations as you like, each with its own modulation amount. When enabled, the button itself begins flashing, all currently assigned destinations become brightly colored, and all potential destinations are shaded.

Note

When a modulator device possesses multiple modulation routing buttons, each button is sometimes represented by only the initial circle of the icon. An example of this is the **Vector-8** device, which has eight modulation routing buttons spread along the sides and corners of a square.

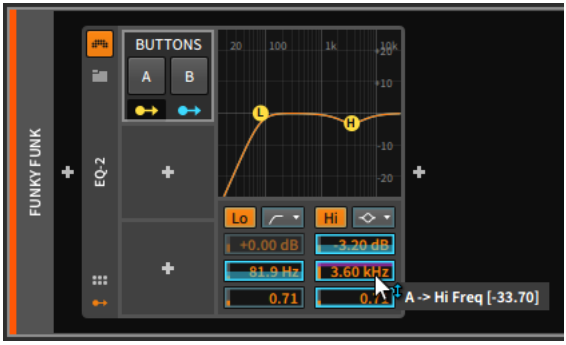


To create a modulation routing: enable the modulation source's modulation routing button. Then click the target parameter and drag its value to set the point of maximum modulation.

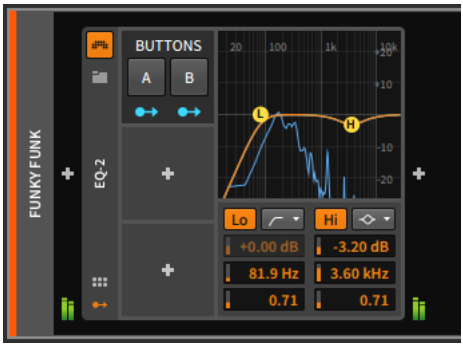
Note

Because the modulation range is set relatively, the range displayed is also relative and does not directly correspond to the parameter's values. So you can twist the modulation range past the parameter's normal range, and this is correct. See the example below, targeting the **Filter** device's *Resonance* parameter with an **LFO** modulator.

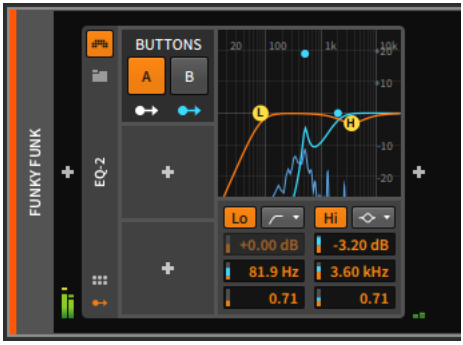
You can assign additional parameters in the same fashion.



So in this **EQ-2** example, we have a high-pass filter set around 80 Hz and a bell filter bringing level down about 3 dB around 3.6 kHz. While button *A* of our **Buttons** modulator is off, those default values remain in place.



But when button *A* is switched on, its modulations kick in. This shifts the high-pass filter up, putting its cutoff frequency around 2 kHz. The bell filter has its cutoff lowered a bit, its gain increased a great deal, and its Q increased slightly. The parameters and the frequency graphic all indicate these adjustments with cyan markers showing the current state of things. (In an auditory sense, these parameter adjustments narrow and focus the frequencies being passed thru.)



Buttons is a relatively simple modulator, offering a binary transition from one set of parameter values to another. Two points worth noting.

First, unlike the discrete behavior of **Buttons**, many modulators work in a continuous way, either transitioning smoothly between states or responding proportionately.

Second, many modulator devices have additional controls that don't fit within their allotted modulator slot. These modulators have a right-facing triangle on the middle of their right edge. When this button is clicked, a pane of additional modulator parameters is exposed.

It is also worth noting that all modulator parameters — both those present atop the modulator slot and those within the additional parameters pane — can themselves be targets of modulations.

Once modulation mappings are present, they can be manipulated and duplicated in myriad ways, either from the **Inspector Panel** (see section 16.2.4.3) or from the **Device Panel**.

To clear a modulation routing from the modulation source: right-click the source's modulation routing button, and then click the *x* icon to the left of the desired parameter.

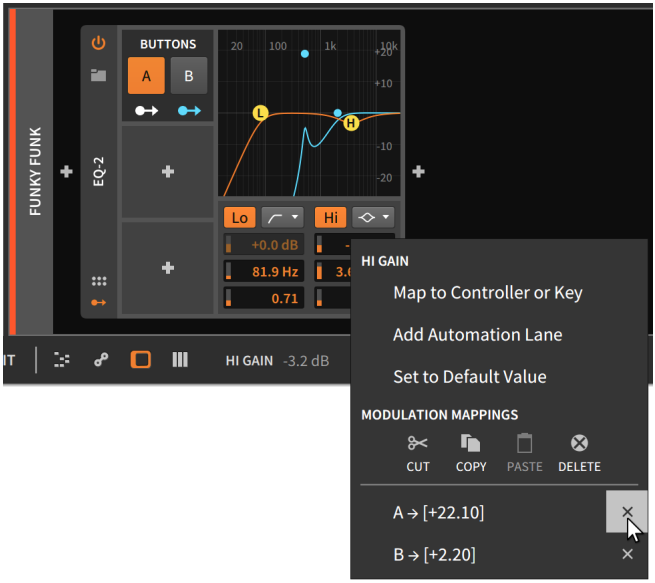
To clear all modulation routings from a modulation source: right-click the source's modulation routing button, and then select *Delete*.

Note

The cut, copy, and paste options from the context menu above also apply to all modulation routings, allowing you to move all listed modulation routings between different modulators. Additional options for dragging and copying modulation routings are available from the **Inspector Panel** (see section 16.2.4.3).

Modulation routings can also be removed from the parameter being targeted.

To clear a modulation routing from the parameter under control: right-click the parameter. Under the *MODULATION ROUTINGS* section of the context menu, find the desired modulation source and click the *x* icon at the far right.



16.2.1.1. The Curve Editor & Pop-out Editors

Some modulators contain editable playback data. And this data can be as important as your automation, notes, or audio. For this case, certain modulators (and some modules) have their own resizeable **Pop-out Editor**.

A special case is the **Curve Editor** for the various curve-based devices that read and write BWCURVE files. All manner of drawing and editing is supported here.

Clicking any curve display opens the **Curve Editor** in a resizeable window.

Note

Key commands mentioned in this section reflect Bitwig's *Default keyboard mappings*. If you are working with your own key commands, most functions can be found and mapped as you like (see section 0.2.2.4).

Seven tools are available:

› *Pointer* [1] - For selecting and adjusting points and their curvature, etc.

› *Pencil* [2] - For freehand drawing of shapes.

› *Step* [3] - A shape for creating flat lines within each grid step.

› *Half Step* [4] - A shape for creating flat lines that spend the first 50% of each step at the level set (the second 50% at *0* [zero]).

› *Saw Up* [5] - A shape for creating a ramp within each step from *0* (zero) to the level set.

› *Saw Down* [6] - A shape for creating a ramp within each step from the level set to *0* (zero).

› *Triangle* [7] - A shape for creating a ramp within each step from *0* (zero) to the level set, and back to *0* (zero).

› The numbers shown above represent the key command for switching to that tool while the **Curve Editor** is open.

Set the grid for drawing, with the *4* x *4* type control in the bottom left.

› The first number controls the number of horizontal, x (↔) divisions on the grid.

› The second number controls the number of vertical, y (↕) divisions on the grid.

› The *Larger Beat Grid* command (default mapping: [.]) also applies to the curve editor with the nearest duplet values.

With a value of *8*, this moves the beat grid to *16*, then to *32*, etc.

› The *Smaller Beat Grid* command (default mapping: [,]) also applies to the curve editor with the nearest duplet values.

With a value of *8*, this moves the beat grid to *4*, then to *2*, etc.

› The *Next Beat Grid Subdivision* command (default mapping: [ALT]+[.]) also applies to the curve editor with the nearest triplet values.

With a value of *8*, this moves the beat grid to *12*, then to *24*, etc.

› The *Previous Beat Grid Subdivision* command (default mapping: [ALT]+[,]) also applies to the curve editor with the nearest triplet values.

With a value of *8*, this moves the beat grid to *6*, then to *3*, etc.

› All of these key commands can be used even while the mouse is held down, adjusting the grid while drawing, for example.

SNAP toggle visually hides the grid lines and disables all snapping with the *Pointer* tool.

› When *SNAP* is on, holding [SHIFT] temporarily disables it.

› Even when *SNAP* is off, all shape drawing tools continue to use the horizontal division to determine their drawing size).

› The *Toggle Snap* command (default mapping: [S]) also applies to the curve editor, and can be used even while the mouse is held.

Several interactions are available within the curve editor:

› Selecting any point shows both that point's *Value* and its *Curvature* (to the next point) in the **Inspector Panel**.

› *To move a segment between two points:* hold [CTRL] ([CMD] on Mac) and drag up or down on a segment.

› *To bend a segment:* hold [ALT] and drag the area between two points up or down.

› *To draw inverse curves (like an S-curve) around a point:* hold [ALT] and drag up or down on the point.

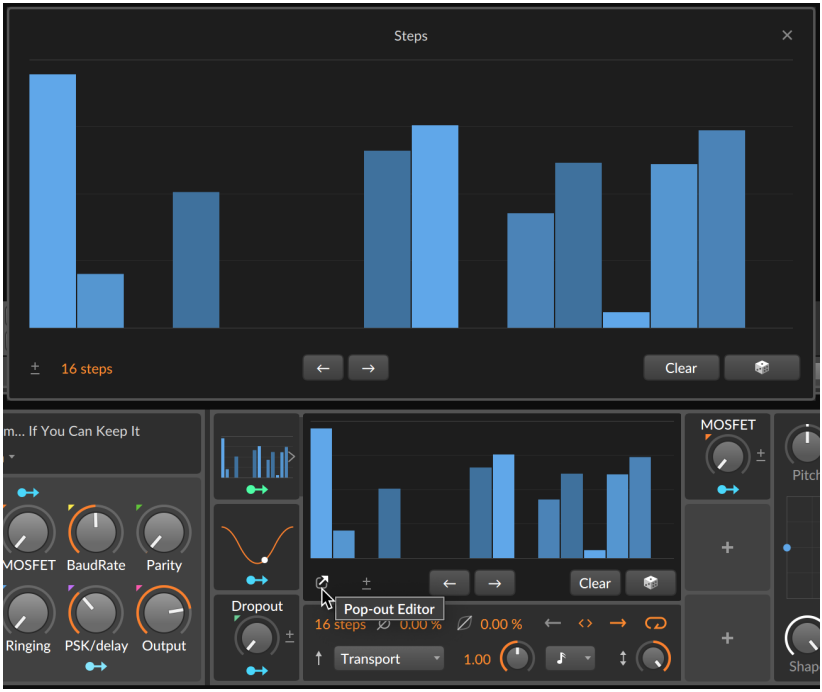
› *To draw identical curves around a point:* hold [ALT]+[SHIFT] and drag up or down on the point.

› *To move a point as well as all points follow it:* hold [CTRL] ([CMD] on Mac) and drag the point.

› Right-clicking within the curve editor offers a menu of *Transform* options for adjusting the entire curve.

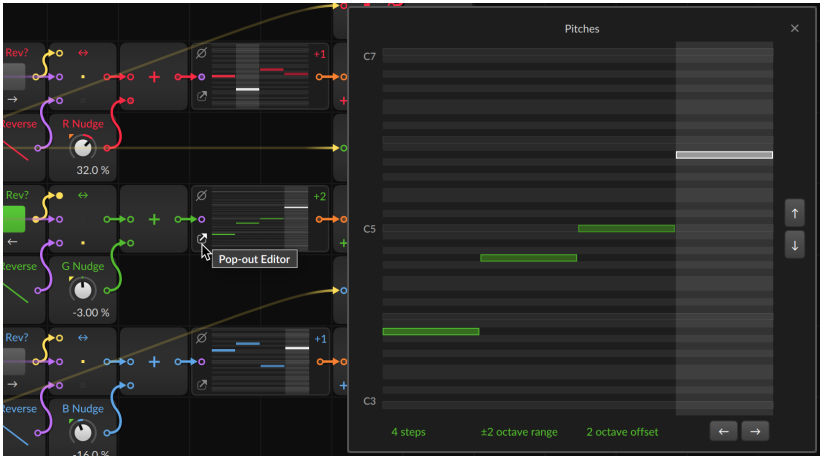
The folder icon at the top left of the window switches to the curve browser for loading other content. The save icon beside it allows saving your current curve by giving it a name, description, category, and any tags you care to use.

Aside from curve-based devices, some other data sequencer modules and modulators have their own custom **Pop-out Editor**, accessible by clicking the little "pop-out" icon in their panel. The **Steps** (Sequence) modulator is one of these.



Each custom **Pop-out Editor** offers a resizeable interface along with some editing functions, such as the nudge backward and forward buttons seen below the sequence data.

This is especially nice in the *The Grid*, giving a clearer distinction between patching mode and data editing mode, as seen here with the **Pitches** (Data) sequencer module popped out.



16.2.2. Track- and Project-level Modulations

Modulators can be used directly on any track, including group tracks and the master track itself. For each level you go up, the list of available destinations grows.

Available modulation targets for tracks include:

› All device parameters on that track.

› All available mixer controls (*Volume*, *Pan*, *Mute*, FX send levels, and *Crossfade Mode*), which can all be mapped in the **Mixer Panel** (and mostly in the track's **Inspector Panel** as well).

Available modulation targets for group tracks include:

› All device parameters on that track, and those within any child track.

› All available mixer controls on that track, and those within any child track

Available modulation targets at the project level (via the master track) include:

› All device parameters on any track.

› All available mixer controls on any track.

› All transport controls that are automatable on the master track (*Tempo*, global *Crossfade*, the *Fill* button, and all groove parameters [*Groove on/off*, *Shuffle amount*, *Shuffle rate*, *Accent amount*, *Accent*

phase, *Accent rate*), which can either be mapped either in the **Mixer Panel**, inside the *Play* menu, or from the transport area.

The modulator panels for tracks are available in the **Device Panel** via the track headers there (see section 8.1.3), with the destinations available in the various places listed above.

16.2.3. Modulations within a Device

Several devices have their own built-in modulation sources. Instrument devices are the best examples so let's look at Bitwig Studio's two-oscillator workhorse, **Polysynth**.

Two on-board control sources are available here, both representing control modules within **Polysynth**. *FEG* (filter envelope generator) is hardwired to the filter's cutoff frequency, just as *AEG* (amplitude envelope generator) controls the volume amplifier of the instrument.

The presence of a modulation routing button on each of these envelope generators suggests that they can also be used for other modulations. Clicking on one of these buttons enables a modulation routing mode, similar to how it worked with modulator devices.

In fact, all behaviors related to connecting, editing, and clearing modulations work exactly the same in all contexts. (This is the virtue of a unified system.)

One difference in the example above is that the modulation routing buttons (for *FEG* and *AEG*) as well as all available target parameters are tinted light green. In the previous section's modulator examples, everything was tinted blue. These subtle shadings do indicate a small but critical difference.

The blue color here indicates a *monophonic modulation*. In the context of modulation sources, a monophonic source generates only one control signal that is then applied to all targets identically (musically speaking, *unison*).

But a green color indicates a *polyphonic modulation*. Polyphonic sources produce multiple control signals, potentially providing a unique signal for each note event (musically speaking, *divisi*). This is the same idea we experienced with expressions before, where each note contained its own, concurrent curve.

Note

CLAP plug-ins can support polyphonic modulation. Of course individual plug-ins vary in what they support, so consult their manufacturer for specifics.

As with musical instruments, polyphony and monophony represent two distinct palettes. This is to say that there is no "winner" between the two; in different cases, each paradigm is preferable. And in Bitwig Studio, we sometimes get a choice between the two.

By adding an **LFO** modulator device to **Polysynth**, it and its potential destinations will appear in blue.

Because of Bitwig Studio's per-note modulation capabilities, modulators such as this **LFO** can be switched to a polyphonic mode.

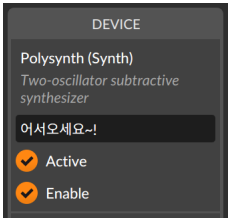
To toggle a modulator between monophonic and polyphonic mode: right-click anywhere within an occupied modulator slot, and then toggle the *Per-Voice* option.

Once the *Per-Voice* option is enabled, this modulator will begin working in polyphonic mode.

Finally, while nested devices can be modulated from the top-level device, polyphonic modulation sources are usually made available as summed monophonic signals. An example can be seen here, with the *FEG* module monophonically targeting a nested **De-Esser** device.

16.2.4. Devices in the Inspector Panel

When a device is selected and the **Inspector Panel** is visible, modulation sources and active modulation routings are the primary parameters being displayed, but a few other parameters are uniquely available in the top of the **Inspector Panel**.



Displayed at the top are the device's name (and category) along with a short description. After these read-only entries are three standard parameters:

› First is a text field for the name of the device. By default, the official name of the device is shown in italicized silver. This can be overridden by typing a name into the field. Deleting an entered name restores the device's official name.

› The *Active* option toggles whether the device is currently active or deactivated.

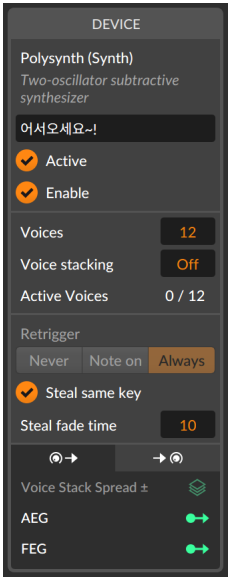
> **Note**
>
> For more information on activating and deactivating project elements, see section 3.2.6.

› The *Enable* toggle is a functional mirror of the device's device enable button.

Beyond these standard parameters, each device has its own parameters in the **Inspector Panel**. To make sense of the range of possibilities, we will look at various examples now, starting with voice parameters for Bitwig Studio instruments and MPE options for plug-ins. Then we will examine the two modulation-related tabs for general devices and a modulator device that has inspector parameters.

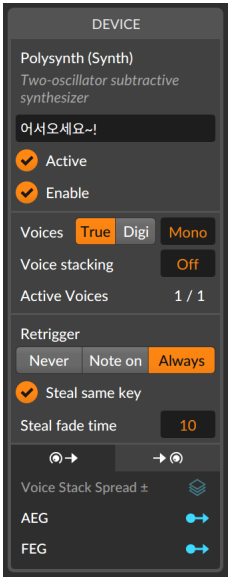
16.2.4.1. Voice Parameters for Instruments

Several of Bitwig Studio's instruments feature a variety of voice-related parameters in their inspectors, just below the standard device parameters.

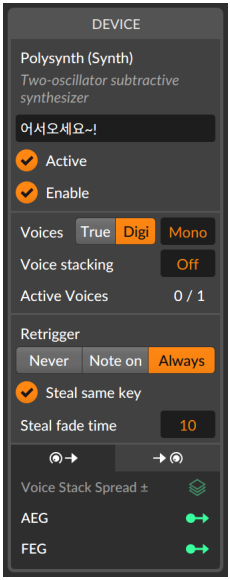


The *Voices* setting determines the number of triggerable voices. When this is set to two or more, the instrument is capable of *polyphony*. This also engages *voice management*, which means that each voice is only active from the time a note-on signal triggers it until the voice is considered finished. So the *Active Voices* readout above is showing that zero out of twelve (*0 / 12*) potential voices are currently engaged, and it will remain so until a new note successfully triggers the instrument.

When *Voices* is set to one, the instrument is now in one of two *monophonic* modes, which the display is indicating by showing *Mono*.



True Mono mode acts similarly to a monophonic analog synthesizer, including the behavior of trigger-based envelopes and the fact that the voice in use is always on, reflected in the *Active Voices* of *1 / 1* in the image above.



On the other hand, *Digi Mono* mode acts like a modern monophonic recreation. All envelopes start from the beginning of the attack stage because two voices are actually alternated here to create a slightly overlapping version of mono. And since using two voices is technically polyphonic, voice management is engaged, just as the *Active Voices* of *0 / 1* in the image above indicates.

Note

Any of the three voice modes can work with Bitwig's **Voice Stacking**, which simply multiplies each engaged voice with additional voices (see section 16.2.5).

Additionally, either *Mono* mode offers *Retrigger* options for legato playing (either starting or releasing a note while another note is being held). These options dictate whether envelope generators will retrigger *Never*, only on *Note on* signals, or on both note-on and note-off signals (*Always*).

Finally, the *Steal same key* option allows each note played to kill any voice previously triggered by the same note over the set *Steal fade time*.

16.2.4.2. Plug-in Inspector Parameters

When a plug-in is selected, the **Inspector Panel** shows a few options.

The *Suspend* option sets how Bitwig Studio determines when the plug-in is not needed and can be safely suspended for the time being. (When this occurs, the plug-in's "power" button icon turns into a moon, indicating that it is resting and saving CPU cycles.) There are three options for this setting:

› *Never* - The plug-in remains active perpetually.

› *When silent* - Bitwig Studio determines when the plug-in isn't needed, based on whether audio is going into and out of the plug-in.

› *Trust plug-in* (default) - Bitwig Studio uses background notifications from the plug-in to determine when it is not active.

When the plug-in is enabled to *Use MPE* (multidimensional polyphonic expression), the functional pitch-bend range (*PB range*) can be set in semitones. So the example above of *± 48* semitones represents a range of four octave up and down. For more information on plug-ins and MPE, see section 16.3.

16.2.4.3. The Modulation Sources Tab, Modulation Transfer Functions, and Modulation Scaling

The *modulation sources tab* is the first tab. Its icon suggests a parameter that is continuing onward in modulation.

Each modulation source of the selected device is listed here, along with a functional modulation routing button.

Beneath each modulation source entry is a list of all active modulations coming from that source. The amount of modulation is shown in orange at the far right and can be adjusted here. The silver *x* that precedes the parameter name on the far left can be clicked to terminate the modulation. (And shown before each modulation amount is a faint graph, which we will discuss at the end of this section.)

In addition, you can also deactivate modulations (like bypass) as well as move modulations from one source to another.

To toggle whether a modulation is active or bypassed: [SHIFT]-click on the text of any modulation routing listed in the **Inspector Panel**. (You can also right-click on the modulator routing button, and then [SHIFT]-click on a modulation listed in the context menu.)

To toggle the active/bypassed state of all modulations from one modulator source: [SHIFT]-click on title of the modulator source in the **Inspector Panel**.

To move a modulation routing from one modulation source to another: click on its silver name, then drag it onto a different modulation source and release.

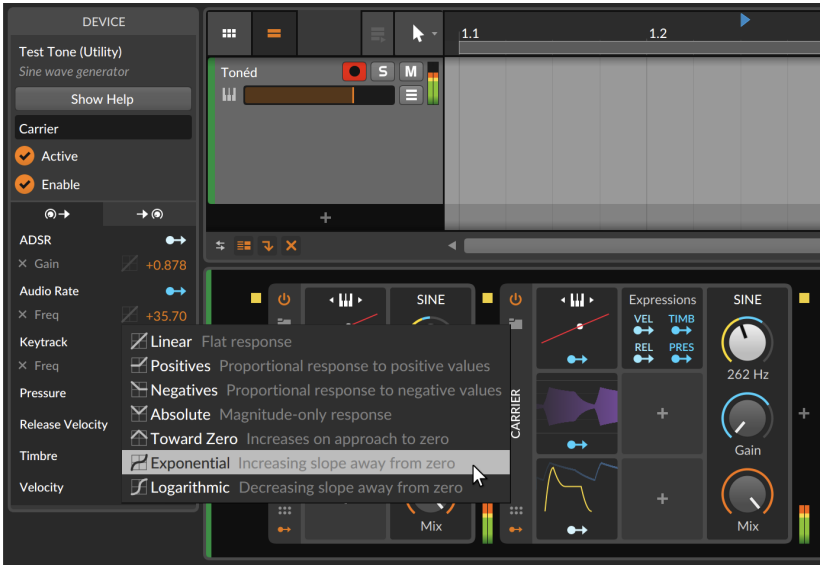
In the image above, for example, you could drag the *Gain* entry listed under *ADSR* to *Keytrack* instead. Now incoming note pitches would be manipulating the *Gain* value.

To copy a modulation routing from one modulation source to another: clicking on its silver name, and then drag it onto a different modulation source and release while holding [ALT].

And to move or copy all modulation routings from a particular source, click on the source's title in white (in the above example, *ADSR* would work) or its modulation routing button. Note that the available actions listed in the window footer (see section 2.2.3) provide helpful reminders of what drag actions are available. Similar actions are also available in the **Device Panel** when [ALT]-dragging either modulation routing buttons or target parameters themselves. Again, the available actions serve as a very helpful reference in that case as well.

Now back to those faint graphs, which are shown for each modulation and do become a bit brighter when hovered over.

This little graph is actually a drop-down menu of *modulation transfer functions* (or curve options) that can be set for each individual modulation connection.



Let's expand a little on each of these modes:

› *Linear* (bipolar) - Flat response, applying each modulation directly as it is received from the source. As this has no effect, mappings set to *Linear* (the default) appear slightly dimmed in the **Inspector Panel**.

› *Positives* (unipolar) - Proportional response to positive values. All positive incoming modulator values are sent out in the positive domain; all negative values coming in are zeroed out.

› *Negatives* (unipolar) - Proportional response to negative values. All negative incoming modulator values are sent out in the positive domain; all positive values coming in are zeroed out.

› *Absolute* (unipolar) - Magnitude-only response. All positive incoming modulator values are sent out in the positive domain; and all negative incoming values are also sent out in the positive domain.

› *Toward Zero* (unipolar) - Increases on approach to zero, mapping both the extreme values (*-1* and *+1*) to zero and incoming *0* values to *+1*.

› *Exponential* (bipolar) - Increasing slope away from zero, creating a more gradual curve between zero and either extreme. This curves the signal to reach the positive and negative maximums less often.

› *Logarithmic* (bipolar) - Decreasing slope away from zero, creating a quicker curve between zero and either extreme. This curves the signal to reach the positive and negative maximums more quickly.

Also note that when in a modulation routing mode, modulations coming from other source are shaded in the **Inspector Panel**, indicating that these can be modulated as well.

In the above image, the *Vel*(ocity) source of the note **Expressions** modulator is currently in mapping mode. Since the three modulations active in this **Test Tone** device (with the user name *Carrier*) are coming from other sources, they are all potential modulation targets. This *modulation scaling* feature allows one modulator source to scale the output of any individual modulation connection. So by clicking on the modulation of the *Gain* parameter from the **ADSR** modulator, each note velocity will now scale the depth of modulation.

The two new listing in the **Inspector Panel** indicate that a modulation scaling is taking place. And if the **ADSR** was then used to modulate a different parameter, that new connection would not be scaled by *Vel*(ocity) as these scalings can be done per-modulation.

Note

In case one modulation connection is using both a transfer function and modulation scaling, the transfer function is applied first, followed by the modulation scaling.

16.2.4.4. The Modulation Destinations Tab

The *modulation destinations tab* is the second tab. Its icon suggests a parameter that is being modulated.

Each parameter being actively modulated in the selected device is listed here.

Beneath each parameter is a list of all active modulations reaching that parameter. The amount of modulation is shown in orange at the far right and can be adjusted here. And the silver *x* that precedes the parameter name on the far left can be clicked to terminate the modulation.

Similar options are also available in this tab for moving or copying modulation routings from one destination to another (see section 16.2.4.3).

16.2.4.5. Modulator Inspector Example

Each modulator has its own **Inspector Panel** as well.

Each modulator shows its modulation sources and mappings, but **Expressions** happens to have a couple of parameters as well.

16.2.5. Voice Stacking

Before diving into **Voice Stacking**, a little context regarding polyphony is in order.

Synthesizers that can play more than one note at a time generally use one *voice* for each note being triggered. So the number of voices available to the synthesizer limits how many notes can be played at a time.

Unison is a classic synthesizer technique for creating thicker sounds. It works by layering multiple voices for each note played (and slightly adjusting some settings for each voice in an imperfect, analog fashion). So if a synthesizer has a two-voice unison mode, each note played will sound two detuned voices. This would help thicken the sound while also reducing the number of voices (or *polyphony*) available.

Bitwig's *Voice Stacking* starts from the same principle, allowing you to layer up to 16 voices for each note played (or, in the case of audio effects, for each channel activated). Each stack can then have *any* parameter varied per voice, either individually or in a distributed fashion.

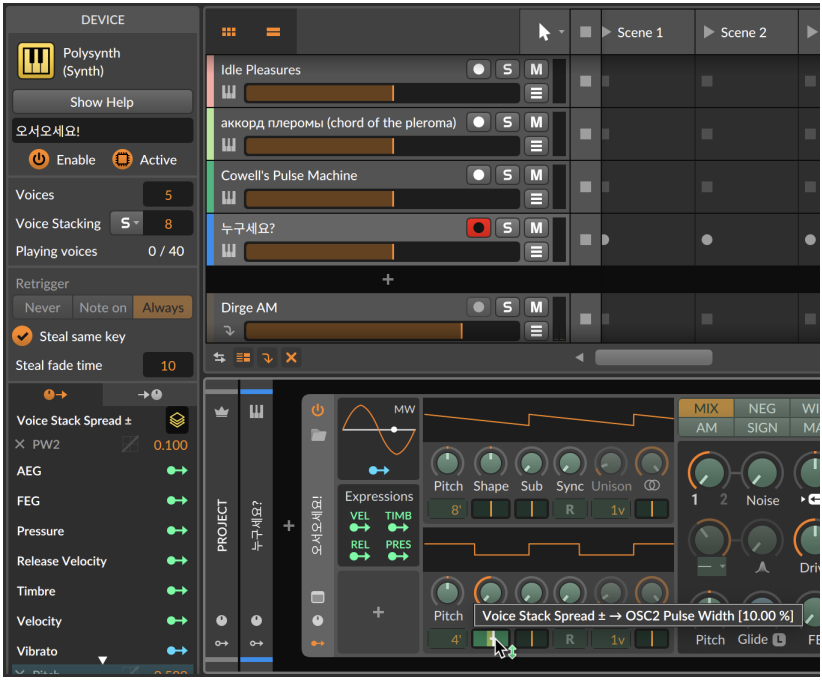
Note

Devices that support **Voice Stacking** include Bitwig Studio's **FM-4**, **Organ**, **Phase-4**, **Polysynth**, and **Sampler**; all of **The Grid** devices (**Poly Grid**, the **FX Grid** audio effect, and the **Note Grid** note effect); and all Grid-powered devices (the **Polymer** synthesizer, as well as the **Filter +** and **Sweep** audio effects). Additionally, CLAP plug-ins can support voice stacking; check with any plug-in manufacturer for details.

As multiple voices are being used for each single note played, voice stacking can steeply increase the load on your processor.

In the **Inspector Panel** of the selected device, *Voice Stacking* is right beneath the *Voices* setting. So if *Voices* is set to *5* and *Voice Stacking* is set to *8*, then you can play up to five notes, and each will trigger eight unique voices sounding together. As the device shown above currently has no engaged voices, the *Playing voices* are listed as *0 / 40*.

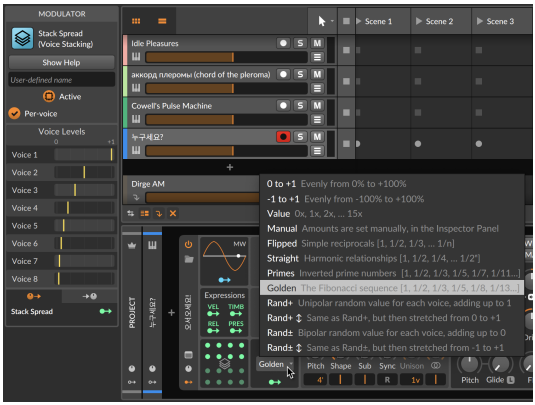
The first modulator available below in the Inspector is *Voice Stack Spread ±*. Assigning this modulation source to a parameter spreads each voice in the stack in an even, bipolar fashion across the modulation range. Let's take an example case.



In the image above, the built-in *Voice Stack Spread ±* modulator is targeting the pulse width of oscillator 2. The modulation amount is currently *+0.10*. So with *Voice Stacking* set to *8*, the range of modulation is spread out so that voice 1 is modulated by *-0.10*, voice 8 is modulated at the full *+0.10*, and all the other voices are spaced equally in between. If the *Voice Stacking* count is changed, the full range of *-0.10* to *+0.10* will be maintained, and the spacing between the voices will uniformly fit that range.

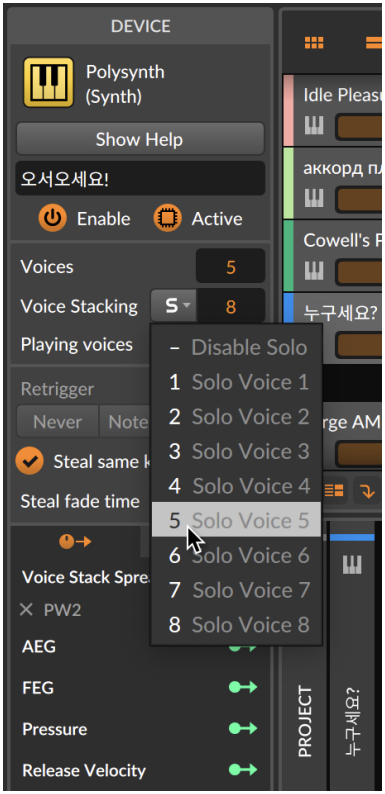
Note

Modulators in the *Voice Stacking* category extend this functionality. Individual control of each voice in a stack comes via the **Voice Control** modulator, and 12 spread modes are available from the **Stack Spread** modulator (see section 19.27.8).

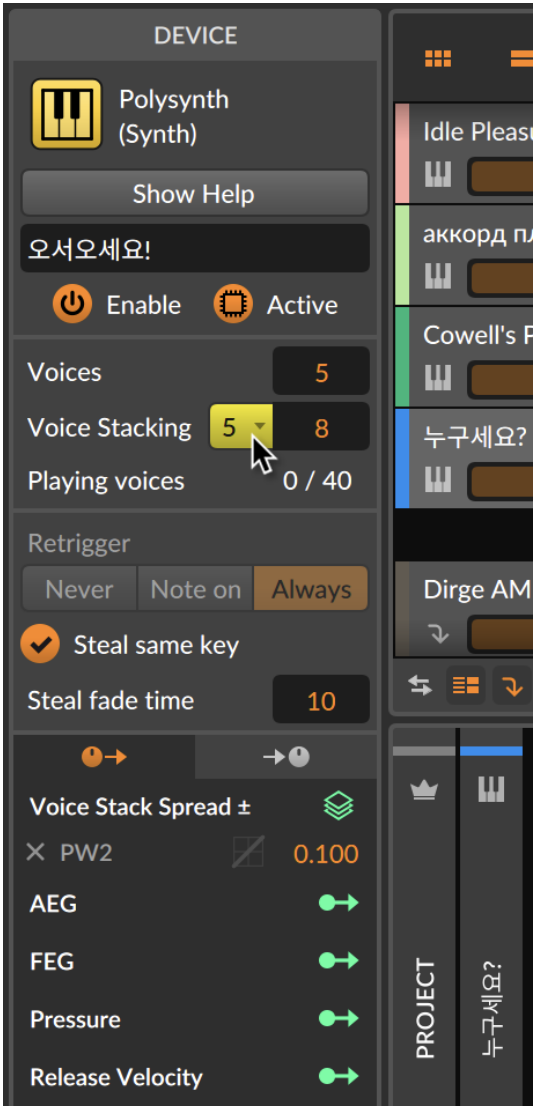


While programming sounds with **Voice Stacking**, it can be useful to temporarily solo one voice at a time. So when *Voice Stacking* setting is enabled, a solo-style *S* button appears beside it.

To solo an individual voice within an active voice stack: click the voice solo menu (*S*), and then select which voice to temporarily solo.



When a voice is being soloed, only audio for that individual voice will sound, and the voice solo menu will show yellow with the voice number that is currently heard.

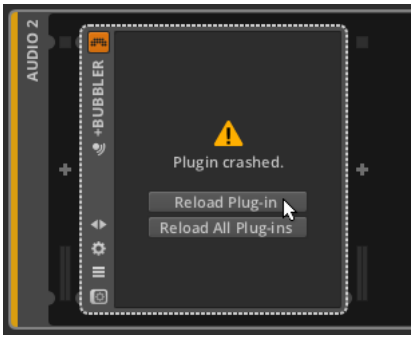


To disable voice solo within an active voice stack: click the voice solo menu (*S*), and then select the *Disable Solo* option at the top of the list. All voices will immediately be heard again.

16.3. Plug-in Handling and Options

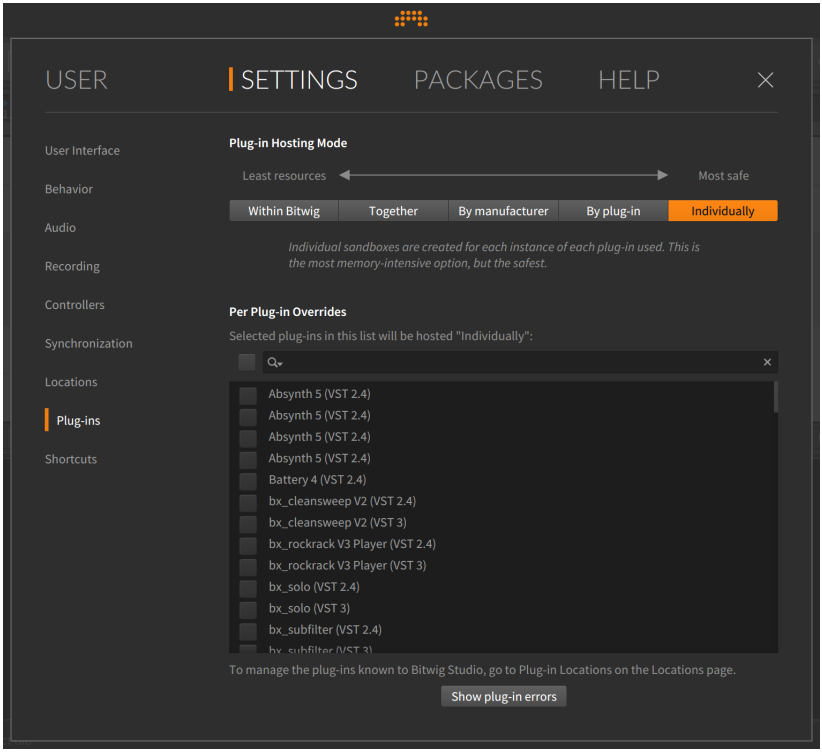
Importantly, Bitwig Studio can handle plug-ins separately from the program itself. By segregating plug-ins into separate sandboxes or processes, the chance of a plug-in crashing other parts of the program is greatly reduced. In many cases, a plug-in crash will happen discreetly, allowing audio to continue playback seamlessly.

If a plug-in does crash, its interface in the **Device Panel** will be replaced with a notification.



By clicking *Reload Plug-in*, the plug-in will be freshly called up again. Clicking *Reload All Plug-ins* will reload every crashed plug-in and leave those that haven't crashed alone.

In the *Settings* tab of the **Dashboard** is a page of settings for *Plug-ins*.



The primary setting here is the *Plug-in Hosting Mode*, which determines how isolated each plug-in process is. As the left-to-right spectrum of options indicates, the settings are progressive with those on the left potentially using less RAM and those toward the right offering greater safety. The options are:

› *Within Bitwig* hosts plug-ins along with Bitwig Studio's audio engine. This keeps the required computer resources to a minimum, but this also means that one plug-in crashing would also crash the audio engine.

› *Together* still hosts all plug-ins, well, together but does it separately from the audio engine. So a crashing plug-in would take the other plug-ins with it, but Bitwig Studio's audio engine should continue running.

› *By manufacturer* hosts all plug-ins into groups based on their manufacturer. This can be particularly useful when a software creator intends for their various plug-ins to communicate with one another.

› *By plug-in* hosts each instance of the same plug-in together. So if you use a particular plug-in on multiple tracks, loading those plug-ins together may save a significant amount of computing resources while also ensuring that a plug-in should only crash when a copy of the same plug-in does. (In other words, no plug-in's stability should be compromised by another plug-in.)

› *Individually* hosts every plug-in instance by itself. This ensures full isolation for each plug-in process, meaning a plug-in crash should not affect anything beyond itself. This will require more computing resources, but that is the trade-off.

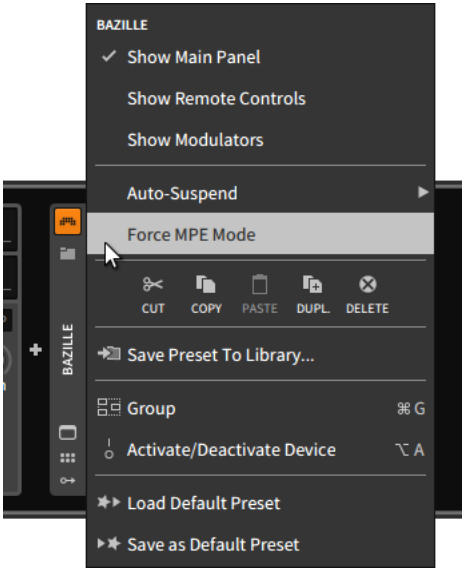
Note

A project currently loaded on the audio engine is not automatically reloaded when the *Plug-in Hosting Mode* changes. In that case, only new plug-ins added will follow the updated setting.

To force the entire project to use a new plug-in hosting mode, either reopen the project or reload the audio engine.

And the list of plug-ins below allows you to select any plug-ins that should run *Individually*, effectively overriding the global setting above. The search box just above the list allows you to quickly find plug-ins from the list.

Finally, if you are using a multitimbral plug-in, its performance may be improved by forcing it to use MPE (multidimensional polyphonic expression) mode. This option is available by right-clicking on a plug-in's device header.



This modern MIDI specification interfaces well with Bitwig Studio's per-note modulation capabilities. Many plug-ins (and probably more of them in the future) opt for this mode on their own, but during this early-adoption phase, enabling *Force MPE Mode* may help get the most out of your plug-ins and any fully-equipped hardware controllers. This option and additional settings are also available from the device's **Inspector Panel** (see section 16.2.4.2).

17. Welcome to The Grid

We have discussed plenty of places where Bitwig Studio uses modular thinking. Most often, this is in the sense of reusable or contained blocks — whether that's audio clips being broken into audio events, controller scripting that can address different tracks/devices in identical ways, or even the dragging-and-dropping of items across any project, or even into another. Sometimes, these modular ideas have had a sound synthesis connotation, most obviously in the expressions of the **Unified Modulation System**. But the idea of a truly modular sound-design environment within Bitwig was always, well, a good idea.

This idea has been made real with **The Grid**. Between the library of 180+ modules (see chapter 19 for short descriptions), the intuitive editing gestures (spread across this chapter), and the twin supports of interactive help (see section 17.1.2.1) and direct module scopes (see section 17.1.2.2), **The Grid** offers our take on modular patching.

Many rules of patching systems are perpetuated: Out ports are connected to in ports via patch cords. Parameters are directly accessible from the face of each module. In ports often have attenuators for scaling the signal on the way in. Control busses sum, and unconnected ports use a value of zero…

And the rules of Bitwig Studio are still applied: The parameters of any module used are the parameters of that device. Parameters can be automated or mapped, modulat(or)ed or accessed by controller scripts. MPE note signals are directly supported. CV signals can come in or out with simple 1x1 modules. Any signal can become a modulator that is used elsewhere…

And yet, there is something new here. Swapping modules and preserving their related settings just makes sense. Having sound never stop — even as modules are added and deleted — is joyful. Stereo control signals are logical but literally sideways. A module version of Bitwig's **Sampler** is like happening across an old friend with a new face. Seeing the sonic effect of a change before you hear it makes things so much faster, and somehow more natural. And how about a way to generate streams of note signals…

But before we start dancing about architecture, let's talk about the patching framework that is **The Grid**.

17.1. Using the Grid Editor

As with any device, Grid presets can be loaded and immediately auditioned. Factory content will always have remote control mappings

for adjusting the sound, and **Poly Grid** patches (generally) respond to notes, **FX Grid** patches (usually) respond to incoming audio, and **Note Grid** patches present a mixed bag of some note processors, some note generators, etc. etc. So at minimum, **The Grid** provides additional sources of sound content.

Note

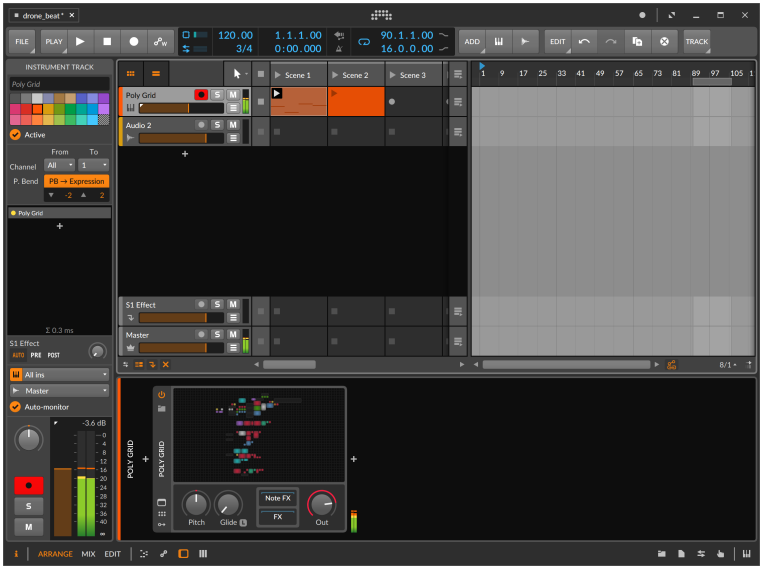
Whatever your balance of using provided content and creating your own, another reason to have three different devices for clear browsing.

When searching for *Instrument* presets (such as clicking *+* on an empty instrument track), **Poly Grid** presets will be shown beside those for **Phase-4** and other recognized VST instrument presets.

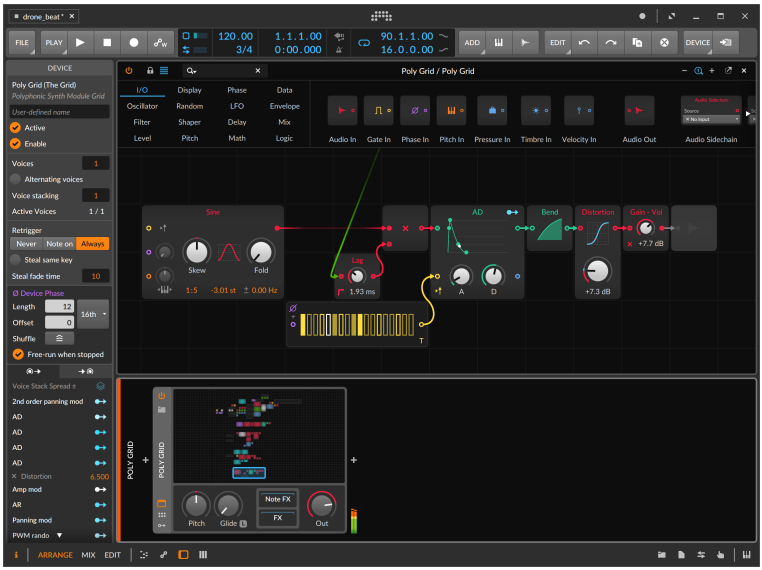
When asking for content in an audio effect context (such as clicking *+* on an audio track or within an instrument's *FX* chain), **FX Grid** presets will be offered beside those for **EQ+** and containers doing audio processing, like **Multiband FX-3** presets.

And when clicking *+* before an instrument, *Note FX* presets will be offered, including **Note Grid** patches, **Strum** presets, and whatever else you've got.

The act of tweaking a patch or making one from scratch — *patching*, for short — means getting comfortable with the Grid editor. The **Expanded Device View** is used for the Grid editor window so all the normal rules apply (see section 8.1.4). You can also interact with the overview display at the center of each Grid device.



To open the Grid editor: click on the Grid device's overview display within the **Device Panel**.



To scroll within a Grid patch: click (or click and drag) within the device's overview display to move the display box.

Within the Grid editor, you can also scroll by:

› Using the scroll wheel of your mouse.

› Hovering over an empty area of the patch, and either [SHIFT]-clicking or middle-click (with your mouse's scroll wheel button) and dragging on the background.

› On a touchscreen, tap on an empty area of the patch with two fingers and drag.

And if you would like more room for the Grid editor, you can hide the **Device Panel** and others by clicking their panel icons (see section 2.2.1), or you can undock the editor by clicking the undocking button (see section 8.1.4).

To close the Grid editor: double-click on the Grid device's overview display within the **Device Panel**, or click the *x* in the top right of the **Expanded Device View** window.

Now that we can open the editor, let's take a look around.

17.1.1. The Module Palette

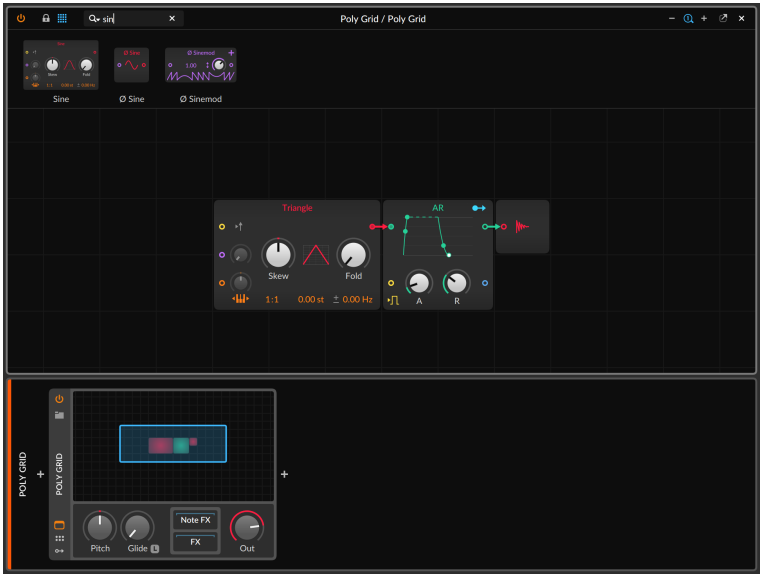
The top of the Grid editor window is home to the *module palette*, which serves as our general browser for Grid modules.

The left side of the palette displays the 16 categories of module. Clicking on any category visually previews all of its modules to the right of the categories, as all modules in the selected *Filter* category are shown in the above picture. In case the modules don't fit the available space, the preview area can be scrolled horizontally, or even with a vertical scroll wheel on a regular mouse.

Note

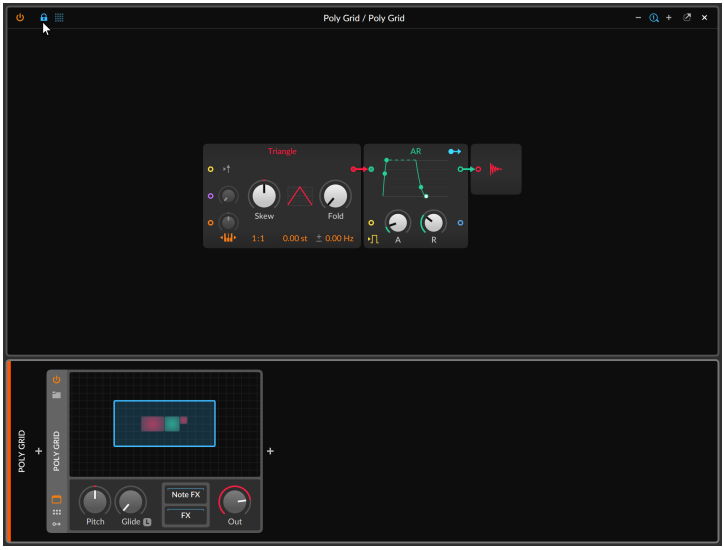
For a short description of each module by category, see section 19.28.

To search for modules: click the search field in the top left of the Grid editor window, and start typing. The module categories are then hidden, using that entire area to display modules that best match your search.

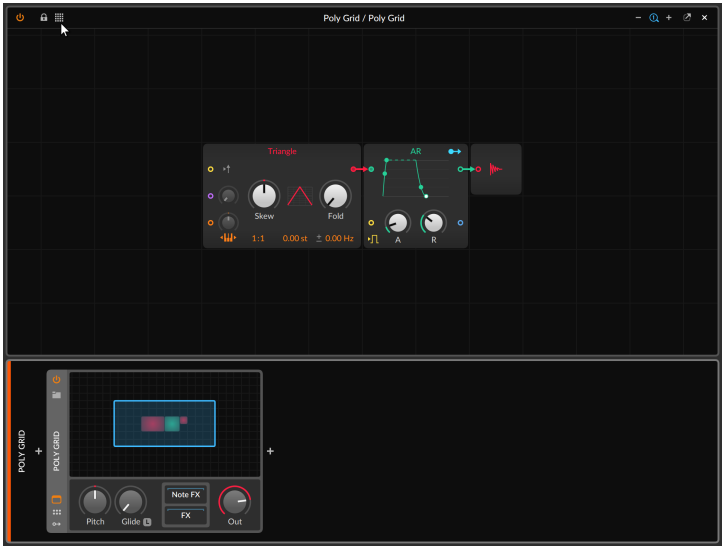


In addition to the standard **Expanded Device View** buttons in the top left corner (to enable/disable the device) and in the top right (to undock and close the window), a few additional buttons are present in the Grid editor:

› The padlock icon enables a locked mode, where parameters can be adjusted but modules cannot be added or cables changed. To differentiate locked mode, both the module palette is hidden from view and the background lines in the editor are removed, eliminating the blueprint-feeling for performances.



› The icon to the right of the padlock looks like a four-by-four table, suggesting the category portion of the module palette. Clicking this icon toggles the visibility of the module palette (and its search field), which can provide more editing space when you don't need the palette.



› On the right side of the window header is a trio of zoom buttons. They allow you to zoom out (-), restore to 100% (the magnifying glass with a *1* inside it), and zoom in (*+*) on the patch within the Grid editor.

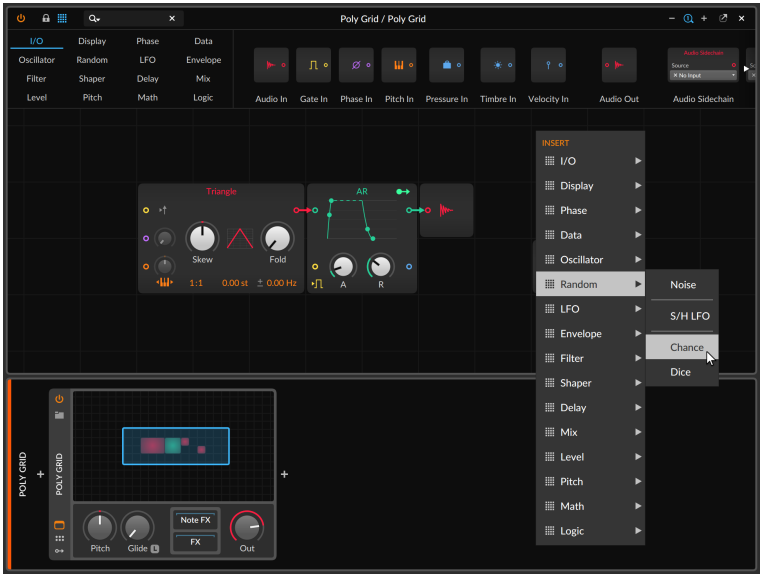
> **Note**
>
> This zooming is independent of the program's scaling level that can be set for each monitor in use (see section 0.2.2.5).

The rest of the window displays your patch for manipulation and editing.

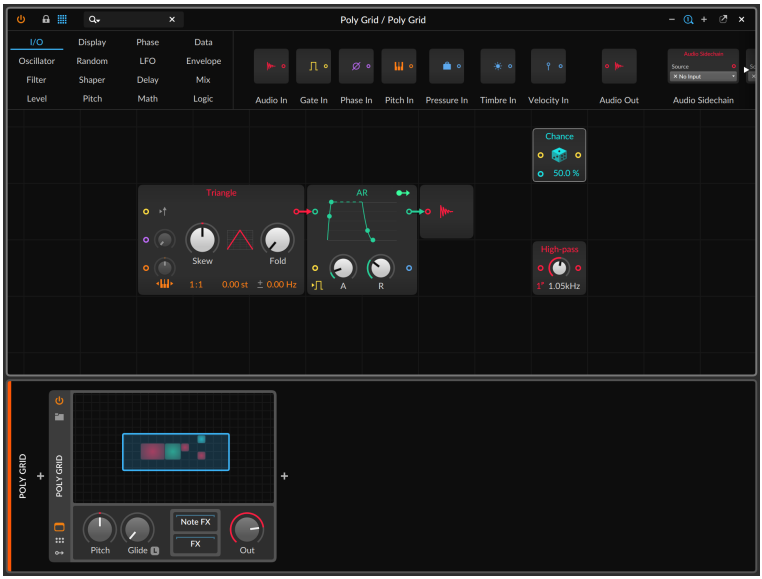
17.1.2. Working with Modules

To add a module to the patch: drag any module from the module palette into an unoccupied area of the patch.

You can also right-click an unoccupied area of the patch to get a text-menu version of the categories and their modules.



Clicking on a module will then insert it into the patch at the location of your original right-click.

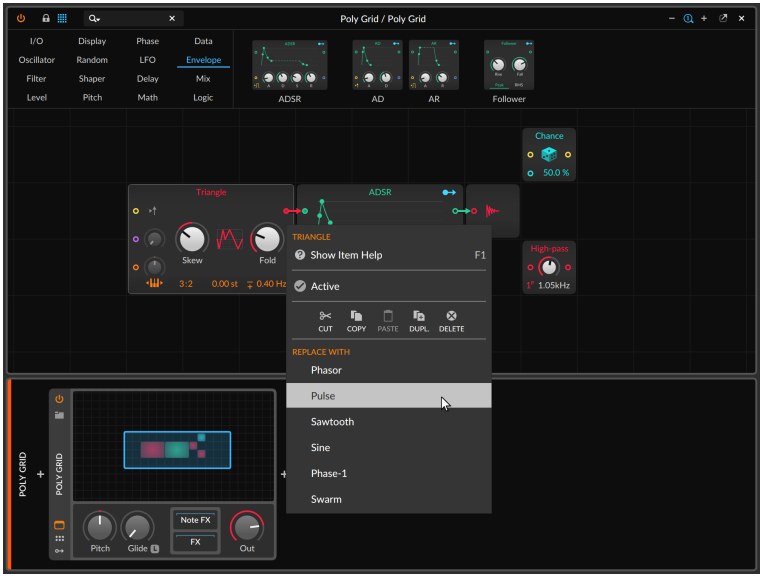


To replace one module in your patch with another: drag the new module from the palette onto the center of module you wish to replace.

In this example, we are dragging **ADSR** from the module palette onto the center of the **AR** module in the current patch. The highlight around the **AR** module shows that it is currently targeted.

The result, shown above, is that **AR** has indeed been swapped out with **ADSR**. This includes any compatible parameters being maintained, all relevant patch cords being recreated, and all modulator paths being remapped to/from the new module.

To replace one module in your patch with a related module: right-click on the module you wish to replace, and then select the new module from the *Replace with* section of the context menu.



Both methods of replace would produce the exact same outcome — in this case, a **Pulse** oscillator in place of the **Triangle** oscillator, using any corresponding settings and the connections that the original had.

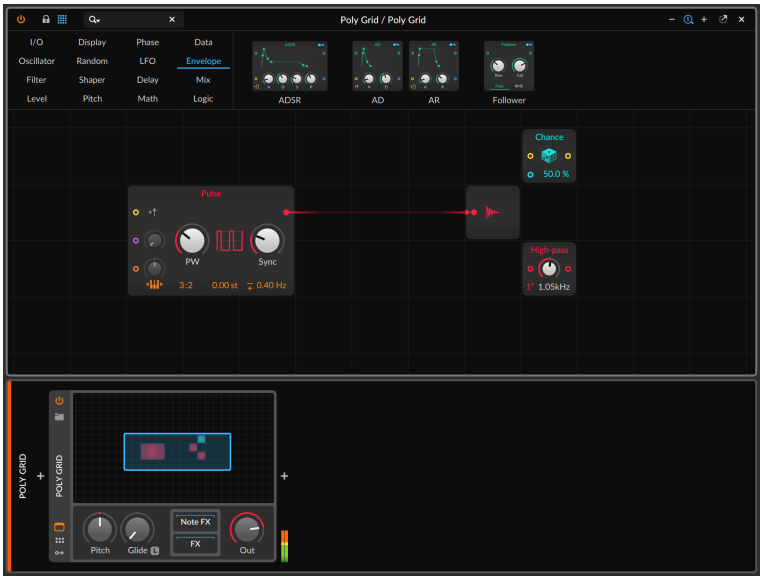
Note

In Bitwig Studio, **Polymer** is a hybrid modular synthesizer based around slots for selecting an oscillator, filter, and envelope generator module. And **Filter+** and **Sweep** are filter effects, each with slots for waveshapers and one or two filters.

In all three of these Grid-powered devices, a clickable menu in each slot is the equivalent of the *Replace with* function, preserving related settings and modulations when swapping devices.

Furthermore, each of these devices can be converted to fully editable Grid devices. Right-click on the device header of **Polymer** for the *Convert to Poly Grid* function, or if using **Filter+** or **Sweep**, right-clicking their device headers will offer *Convert to FX Grid*.

To delete a module: select the module in the Grid editor, and then press [DELETE] or [BACKSPACE].



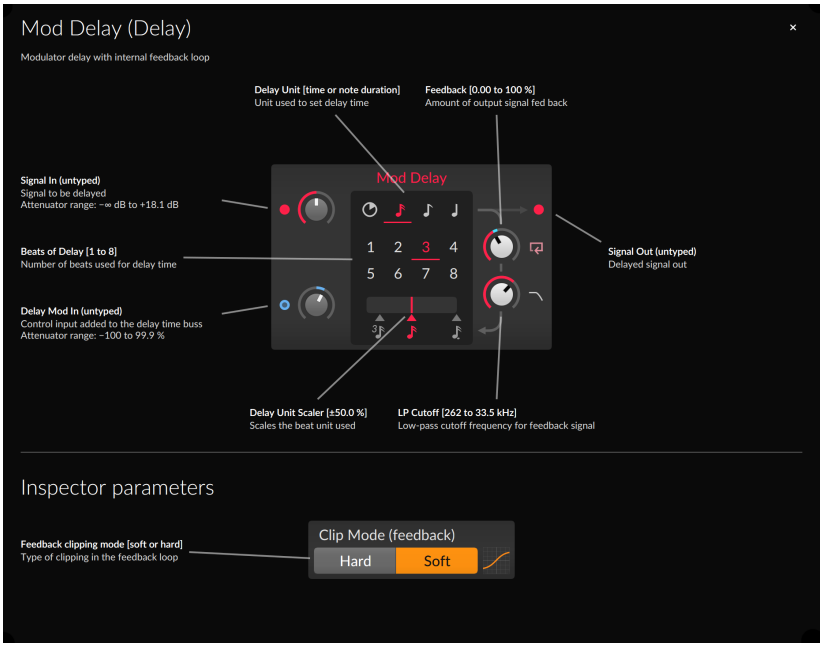
In the example above, we deleted the **ADSR** module in the patch. Instead of just deleting the module and all attached cords, **The Grid** saw a signal running thru **ADSR** and replaced the cord, straight from the **Triangle** oscillator to the **Audio Out** buss. Drones away.

17.1.2.1. Interactive Module Help

One feature of **The Grid** is that documentation of each module is built into the program. While we are happy to have you reading this manual, details local to each module are more useful when deploying the modules themselves.

To view a module's documentation: select the module in the current patch, and then select *Show Help Item* from the *Module* menu.

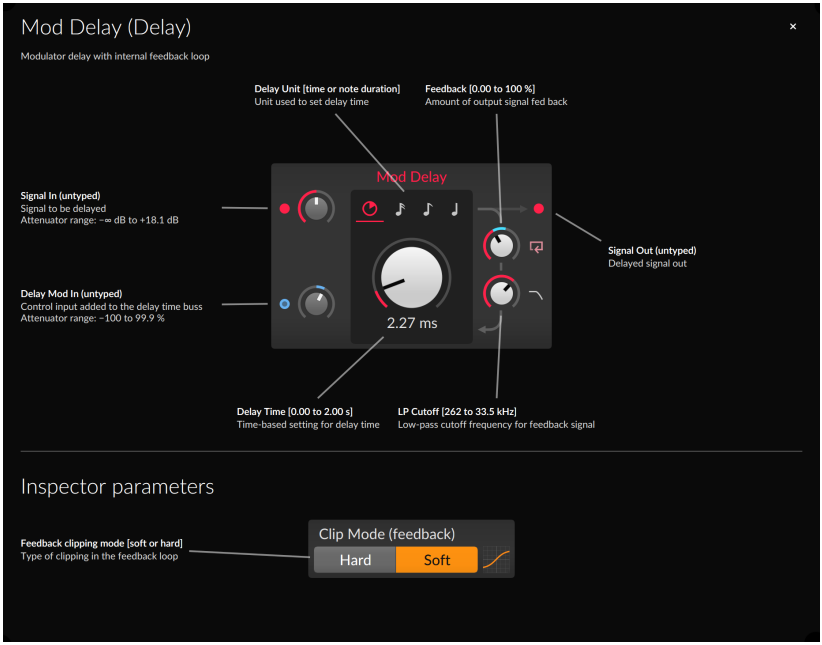
You can also access this feature by selecting the module and then either clicking the *Show Help* button in the **Inspector Panel** or pressing [F1] (the default mapping for *Show Help Item*). The special show help window will appear.



On the surface level, the help view displays all relevant parameter information for this type of module. In addition to the module's regular interface, any *Inspector parameters* are also shown below. This can be especially helpful as these parameters are often out of sight, out of mind.

Beyond the text on screen, this help view is indeed showing the same module that is in your patch. This means that port signal indicators and modulator rings are reflecting the current state of this particular

module, and parameters can be freely adjusted while this view is open. And if a mode setting changes the available parameters on this module, the help view will follow. Using the **Mod Delay** example shown above, switching the delay unit from 16th notes to free time will change both the parameters available and the descriptions present, as seen below.



17.1.2.2. Module Scopes in the Inspector Panel

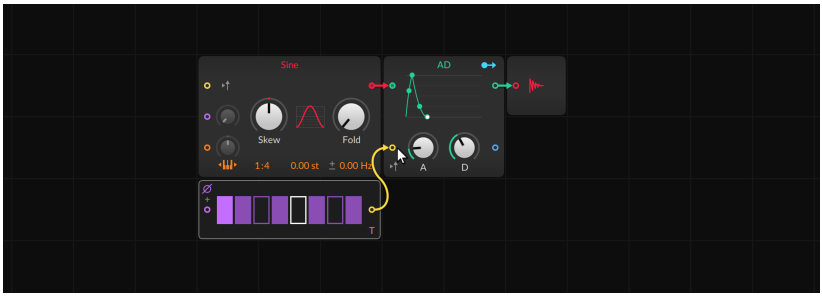
When a module is selected in the Grid editor, the **Inspector Panel** shows more than its available parameters. It also displays an oscilloscope view of the signals at each in and out port.

In the example above, the **AR** module is selected in the Grid editor. So the **Inspector Panel** automatically displays three in port scopes (two for the ports on the device, plus one for the pre-cord that is currently enabled) and two out port scopes (for signal at the two out ports). Note that the *Gate In* scope is dimmed and folded away because no cord is currently connected to this in port.

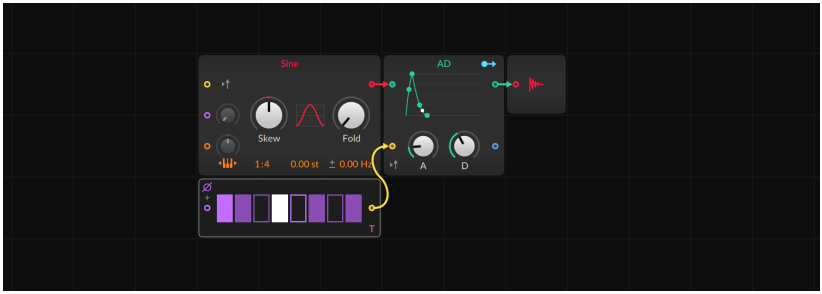
17.1.3. Working with Patch Cords

Now that we can add modules, we have but to connect them. That means we need to put our (virtual) patch cords to work.

To create a patch cord: click on either an in or out port, and then drag to a port of the opposite kind.



Cables will snap to nearby ports as you drag them around. Once you release the mouse button (or let go of your finger), the cable will be connected, and signal will begin flowing.

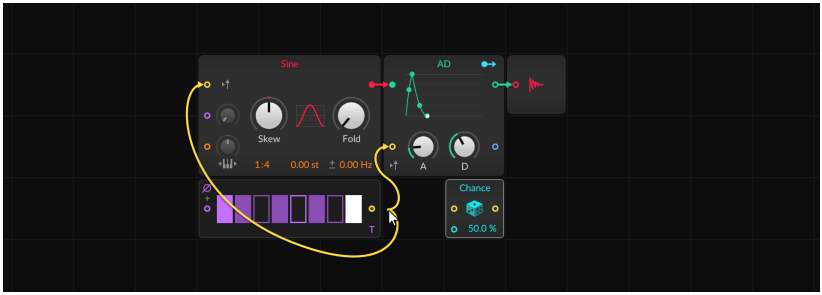


Note

The Grid allows an out port to be connected to multiple in ports, but in ports can only receive one cable. You can, of course, merge multiple signals and connect the result to an in port. **The Grid** even allows you to do it with modifier keys (see section 17.1.4).

To delete a patch cord: double-click the in port or out port where the cord is connected.

To move a patch cord: double-click and drag either end of the cord to another port and release. This will move all cables at that port so if you are dragging from an out port that has several connected cables, they will all be moved together.



Note

If you double-click and move a patch cord(s) to an unoccupied area, the connection(s) will be deleted.

17.1.4. Inserting Modules with Cords, and Vice Versa

We've already covered the necessities for any virtual modular environment: adding and removing modules, and then connecting them with patch cords. But **The Grid** goes beyond these baseline requirements, placing a premium on clear gestures plus some patching intelligence from Bitwig Studio. We already looked at replacing modules (see section 17.1.3), but inserting modules and patch cords together is another way to prioritize sound design over patch management.

To insert a module with patch cords: drag the new module from the palette to the port where you want it connected, and then release.

Dragging over an empty in or out port will connect a corresponding port to it.

Instead of dragging to a particular port, you can also drag to the left or right edge of a module.

Bitwig Studio will then connect the new module to the in or out port that seems most appropriate.

You can also drag a new module onto a port where a patch cord is already present.

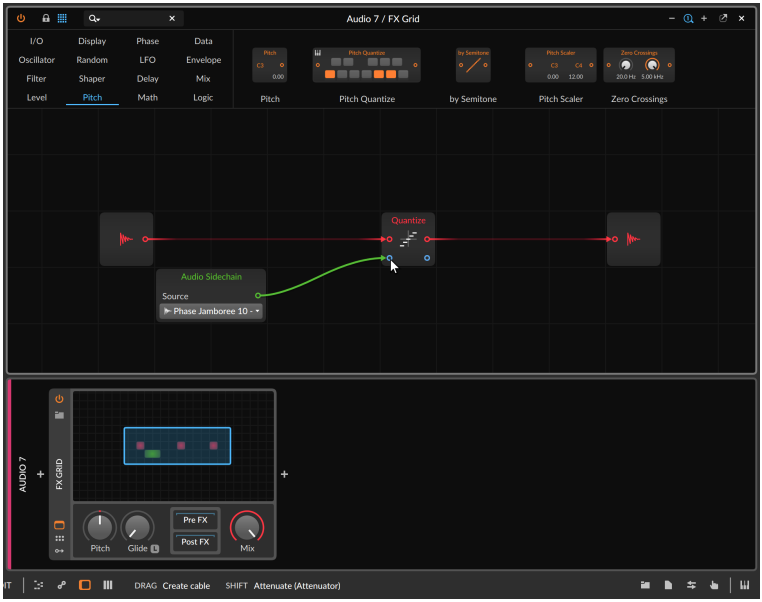
This previous signal path will be connected thru the new module.

Note

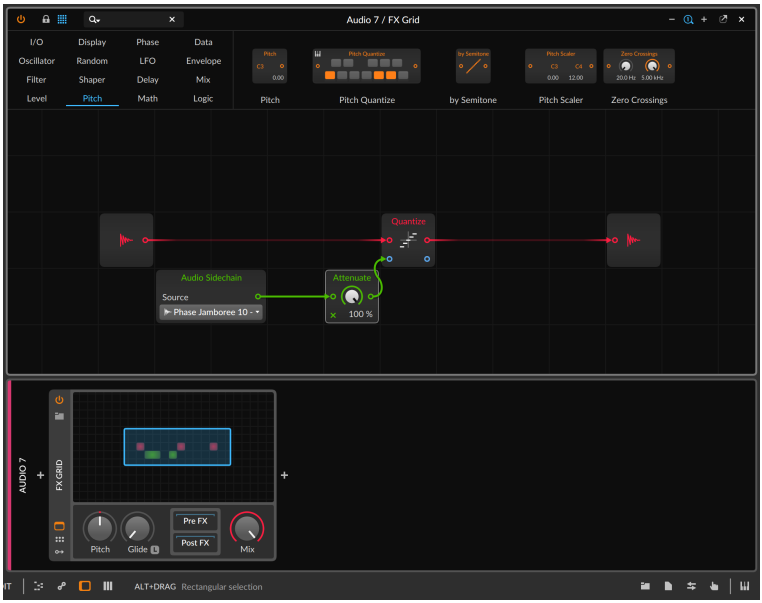
If you drag the module onto a connected in port, that one cable will be rerouted thru the module (if possible). If you drag onto a connected out port, all cables present will be routed thru the new module.

There are also gestures for adding common processor and merge modules when drawing new patch cords.

To add a processor module when creating a patch cord: draw the cord from the desired out port to the in port, and then hold one of the available modifiers listed in the window footer.

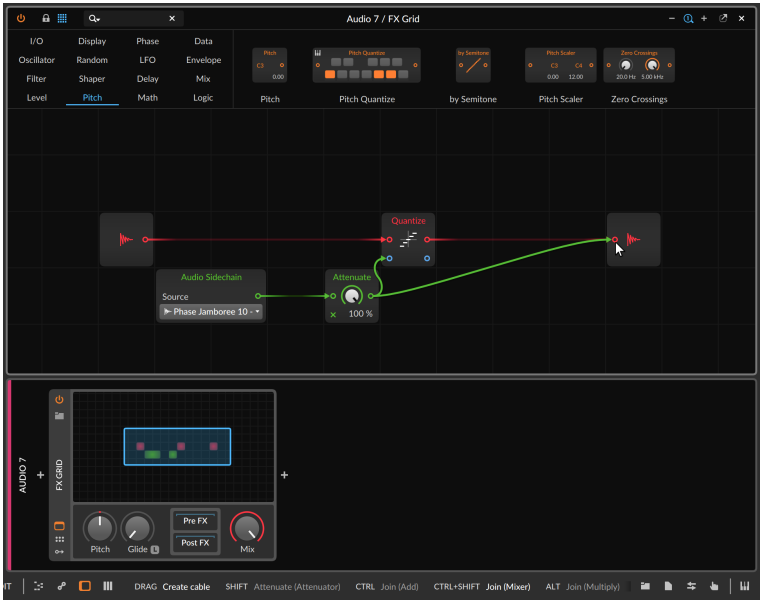


In the case above, [SHIFT] is being held so when the mouse or touch is released, an **Attenuate** module will be added in line.

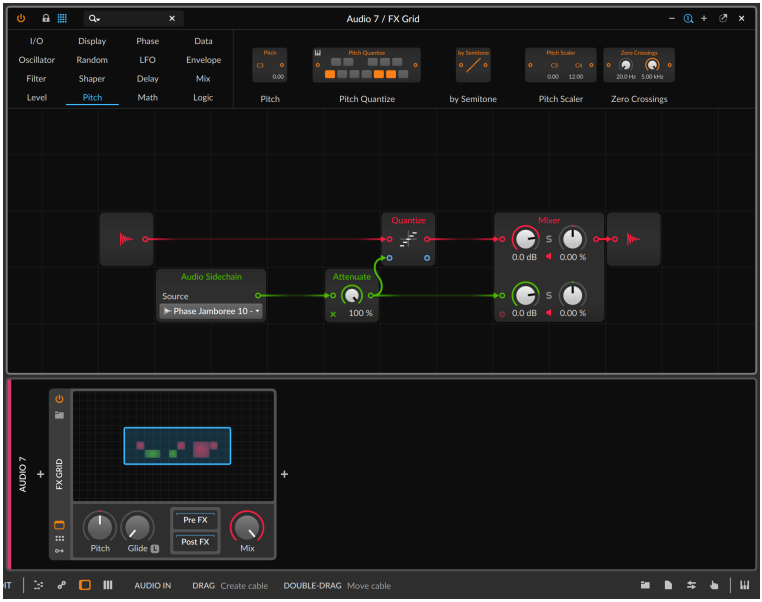


And since in ports can only receive one cable, there are also gestures for creating an additional in port by means of various "merge" modules.

To merge a preexisting signal with a newly created patch cord: draw the cord from the desired out port to the occupied in port, and then hold one of the available modifiers listed in the window footer.



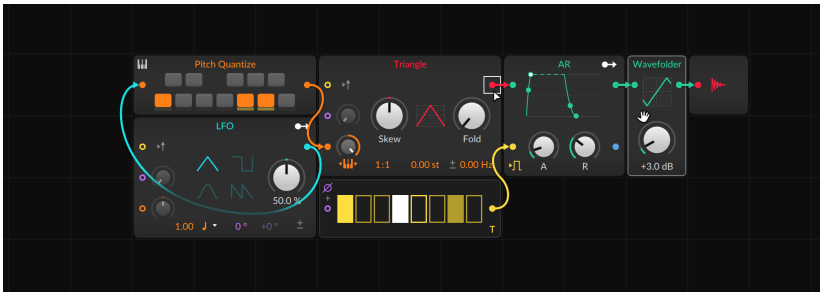
In the case above, the modifier for a **Mixer** module is being held, so both the original cord and the new one being drawn will be merged via a **Mixer** and connected to the original in port.



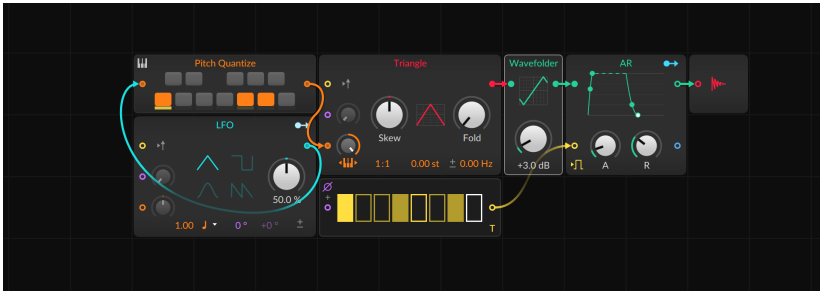
17.1.5. Reordering Modules

Modules can also be reordered with similar behavior as the workflows for inserting modules with patch cords (see section 17.1.4).

To reorder a module within your patch: drag the module from its current position onto the port where you want it connected, and then release.



Once the click/touch is released, the module will be rerouted within the patch.



17.2. Special Connections

A few special cases exist around and within **The Grid** that are worth looking at.

17.2.1. Grid Devices and Thru Signals

Virtually every Bitwig device passes thru signals that are not its focus. For example, normal note effect and instrument devices pass thru audio signals that reach them, which helps facilitate workflows such as *Bounce In Place* (see section 13.2.2). And instrument and audio effect devices send on the note signals they receive, as following audio devices or modulators may take advantage of them.

Grid devices are a bit unique here, as we expect you may use these devices in ways we do not expect. Accordingly, **Note Grid**, **Poly Grid**, and **FX Grid** all have Inspector parameters for whether received note signals (*Note Thru*) and non-note MIDI messages (*Control Thru*) should be passed on to the output, in addition with any signals that might be created by the device via **Note Out** and **CC Out** modules.

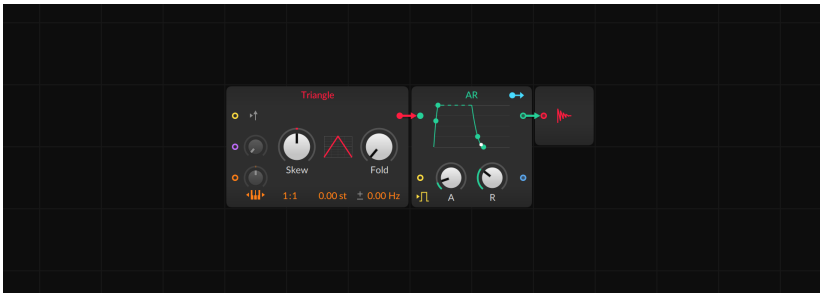
Note Grid has an additional option for *Audio Thru* as well, but it is slightly different since merging audio doesn't always end well. When enabled, audio reaching the device is simply passed thru — and any **Audio Out** modules used by the patch are suspended. When *Audio Thru* is disabled, audio routed thru or generated inside the Grid patch will be passed on, but audio reaching the **Note Grid** device is not automatically passed thru. (**Poly Grid** automatically passes audio thru, where as the audio effect-oriented **FX Grid** relies on its *Mix* parameter [and any manual configurations you may patch] to blend between dry and wet signals.)

All of these parameters are on by default, except for **Note Grid** which has *Note Thru* disabled (since the default preset of **Note Grid** is already passing all notes thru with any processing you may add). But defaults are meant to be broken, especially when a Grid patch takes you in a new direction.

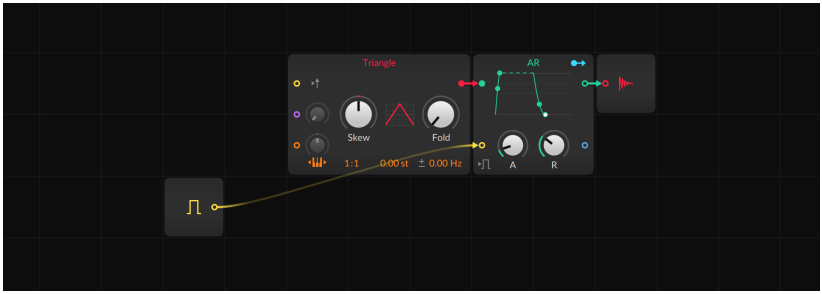
17.2.2. Module Pre-cords

Within Grid patches, there are some common connections that are made more often than not. *Pre-cords* provide wireless connections for some of the most common connections, generally taking the form of a toggle near an in port that connects to the same module buss. This both allows modules to be preconfigured with regular connections and saves the clutter of cords running from single *I/O* modules to various destinations across each patch.

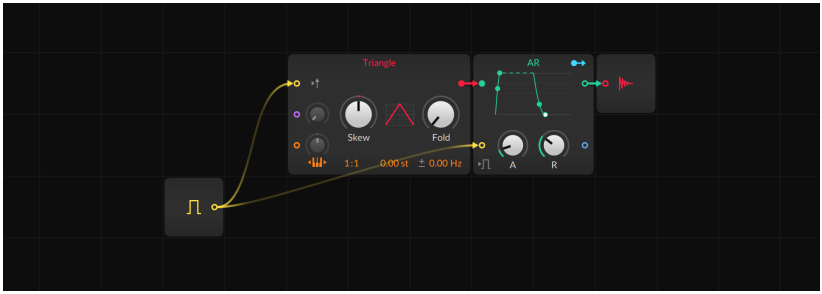
For example, the default **Poly Grid** patch contains three pre-cords.



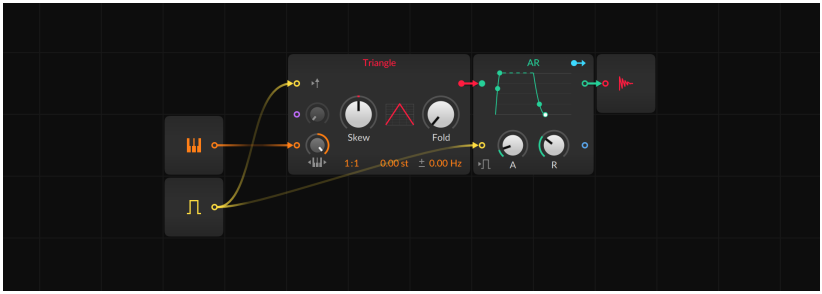
› On the **AR** module, there is an icon in the bottom left corner representing a two-state logic signal. This is a pre-cord for bringing all note gate signals reaching the device to the envelope generator's gate in port. This toggle is enabled by default as envelope generators are most often gated by note input. Manually creating this connection would require the **Gate In** module.



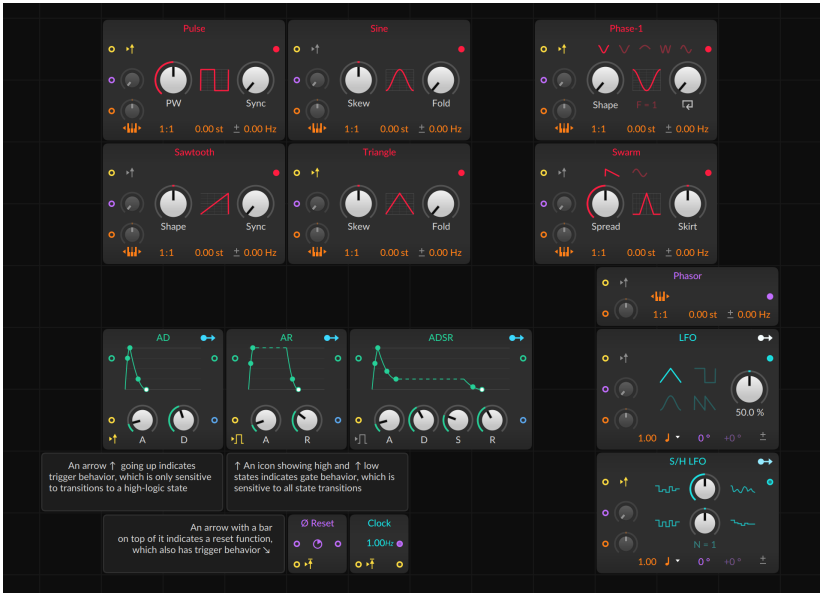
› On the **Triangle** module, there is an arrow icon to the right of the yellow retrigger in port. This is a pre-cord for bringing all note gate signals reaching the device to the oscillator's retrigger port (for restarting the oscillator's phase). This toggle is disabled by default. Manually creating this connection would also require the **Gate In** module.



› On the **Triangle** module, there is an icon in the bottom left corner showing a piano keyboard with arrows in both directions, representing keyboard tracking. This is a pre-cord for bringing all note pitch signals reaching the device to the oscillator's pitch buss. This toggle is enabled by default as oscillators usually incorporate the pitch of incoming notes. Manually creating this connection would also require the **Pitch In** module — and to open the module's pitch in port attenuator all the way so that notes land in the proper place.

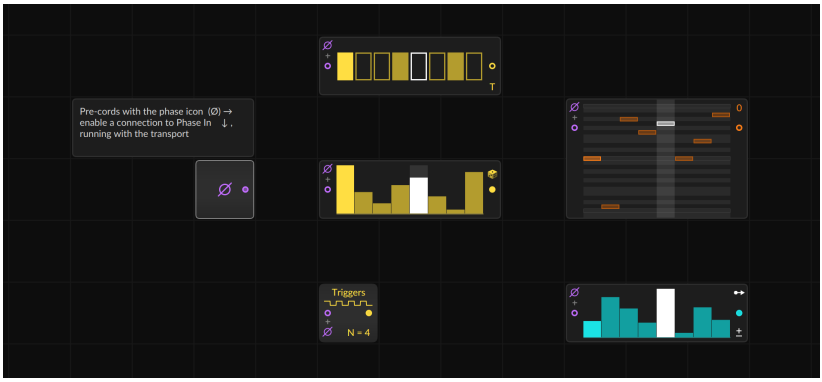


Multiple modules have pre-cords from note gate signals, with various icons to help illustrate the buss's behavior.



Several modules have pre-cords from note pitch signals. In the case of oscillators, these are toggles. For filters, the pre-cord takes the form of an attenuator.

And the data-sequencer modules have pre-cords from the device's phase signal (which is configured in the **Inspector Panel** for the device). These connections could be made manually from the **Phase In** module.

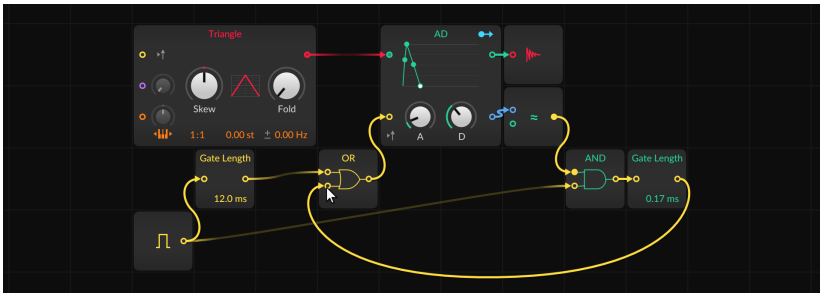


Some other modules (such as **Sampler** and **Pitch Quantize**) use pre-cords in ways specific only to those modules, which make their module-specific help views especially useful.

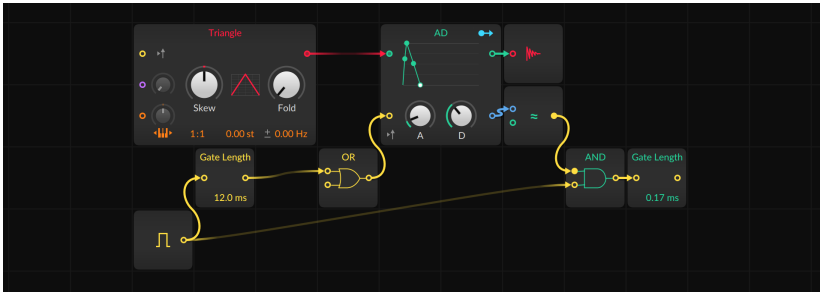
17.2.3. Making Feedback with "Long Delay"

Feedback loops are possible but prevented when made directly in **The Grid**.

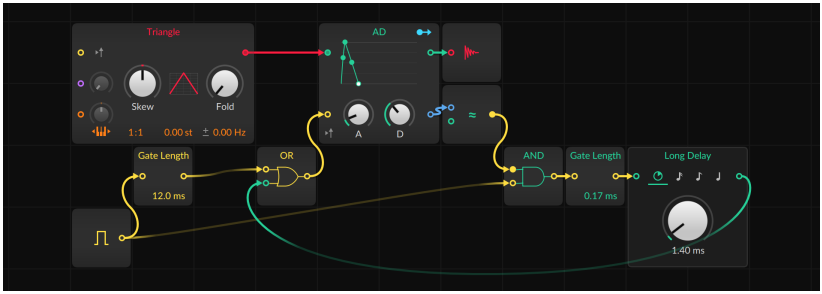
For example, the patch below is attempting to create a looping envelope. (The logic is that the envelope should trigger both when a new note is played [via **Gate In**] **OR** if the envelope signal [second output of **AD**] is equal to zero **AND** the note is still being held down [via **Gate In**].) In the screenshot, I am adding a cable to complete the feedback loop, passing a signal coming out of the envelope generator and back to its gate input.



But once the mouse/touch is released, the cable disappears because this kind of direct feedback is not allowed.



To make a feedback connection: insert a **Long Delay** (*Delay/FX*) module into the feedback path.



Long Delay is specially configured to allow feedback and has a minimum delay time of one block size (see section 0.2.2.1).

17.3. On Grid Signals

In addition to being exposed with literal, virtual patch cords, signals in **The Grid** are different from other signals in Bitwig Studio.

17.3.1. Signal Types

While any signal can be connected anywhere, there are certain signal types within **The Grid**, often indicated by port color but always identified in each module's help view.

› *Logic* (yellow). A bistate signal, often for triggering an event or setting a state. For in ports, any signal level at or above *+0.5* is treated as high logic, and anything below is assessed as low logic. Logic in ports are only sensitive to these state changes so jumping from *0* to *+0.5* will register, but a slow ramp from *+0.5* to *+1* means nothing. For out ports, a high-logic state is expressed as *+1*, and a low-logic state is transmitted as *0*.

For short, we may sometimes refer to a *trigger*, which is the transition from a low-logic state to a high-logic state. This signal is often used to start a function.

› *Phase* (purple). A unipolar signal from *0* to just below 1, often for driving data lookup. For in ports, signals are wrapped into the range. For example, a value of *+1.02* would be used as *+0.02*, and a value of *-0.3* would be treated as *+0.7*.

› *Pitch* (orange). A bipolar signal used by Bitwig for specifying pitch. *0* represents "middle C" (C3) with each change of ±0.1 representing an octave, so a typical signal range of *-1* to *+1* represents twenty octaves.

> **Note**
>
> While pitch signals in **The Grid** are generally used as is, outputting notes via the **Note Out** module only allows the permissible MIDI note range (see section 19.28.1.14).

› *Untyped signal* (often red). The most common signal type, of unspecified range and function. Inputs of mixers or filters or math modules, virtually all outputs that aren't implementing either logic, phase, or pitch signal characteristics — they are all most often untyped and thereby follow the color set for the module they are on.

> **Note**
>
> Generic signal modules are normally red, with typical control modules defaulting to turquoise. So ports of either color are untyped signal ports. And when a module has multiple untyped in ports, those ports will adopt the color of an incoming patch cord.

› *Secondary untyped signal* (blue). When a module has two kinds of untyped signal in or out ports, the secondary port is shown in blue. For example, a merge module might have multiple primary in ports (using the module color) for the various signals to route and one control input (colored blue) for selecting which input is passed thru.

17.3.2. Stereo By Nature, and 4x Faster

Every signal in **The Grid** is stereo. This means that whenever you see one patch cord, you are actually seeing a stereo pair. So yes, every audio cable is stereo, but so are all pitch, phase, and trigger signals as well. Altering any of these various control or timing values will affect the corresponding audio.

In addition to common stereo placement functions (found in **Mixer**, **Pan**, and **Stereo Width** modules of the *Mix* category), a number of modules are configured to make working in stereo easy and interesting:

› Most *Oscillator* modules (**Pulse**, **Sawtooth**, **Sine**, **Triangle**, **Wavetable**, **Sub**, **Phase-1**, and **Swarm**) have frequency offset values, set in Hertz (Hz). When this value's polarity signal (± when the value is positive, ∓ when it's negative) is clicked, the frequency offset is inverted for the right channel. In the *Phase* category, the **Phasor** module (a good starting point for building your own oscillators with other *Phase* and *Data* modules) also has this same option.

› The **LFO** module (under *LFO*) and **S/H LFO** (under *Random*) both have a purple phase parameter, which defaults to *0 º*. And to the right of that phase control is an offset control for the right channel, which starts at *+0 º* and is thus grayed out by default. Both parameters are visualized on the **LFO** module.

› In the *Mix* category, **Stereo Split** and **Stereo Merge** allow you to separate and reconstruct a signal as left–right and/or mid–side pairs.

› Also in the *Mix* category, **LR Gain** provides independent level controls (±200%) for the left and right channels of any signal passing thru.

› In the *Random* category, **Noise** also has a stereo option (via the clickable on-panel stereo icon). This creates independent signals for the left and right channels.

› Several modules in the *Level* category (**Value**, **Attenuate**, **Bias**, and **Bend**) and in the *Phase* category (**Ø Bend**, **Ø Pitch**, **Ø Shift**, and **Ø Skew**) have an Inspector parameter called *Stereo-ize*, which inverts the value used for operation on the right channel. The same is true for the **Pitch** constant module (in the *Pitch* category).

› **Flanger+** and **Phaser+** (*Delay/FX*) both have a specialized *Stereo-ize* option, which inverts the right modulation signal. This works whether that classic modulation signal is coming from the default internal LFOs, or if you have connected a signal to the *Mod In* port.

› The **Ø Reverse** (*Phase*) module as well as **Invert** and **Reciprocal** (Math) and a few additive processors in the *Pitch* category (**Octaver**, **Ratio**, and **Transpose**) have a *Stereo-ness* parameter, which sets whether the processing is applied to the entire signal (*Mono*) or if it is just done on one channel (*Left* or *Right*).

In addition to being stereo, all signals within **The Grid** also operate at four-times (400%) your configured sample rate. This is to ensure maximum fidelity not only for the final output, but also for any audio-rate modulation or other synthesis techniques that might be employed.

Note

A few modules have in ports that flatten any incoming stereo signal to mono. This is often because the result has to be mono (such as **CC Out**, **Note Out**, and **Modulator Out** [*I/O*]), or because stereo operation would be unnecessarily complicated (**Sampler** [*Oscillator*] and **Recorder** [*Delay/FX*]). Specifics can be found within Bitwig in each module's in-app help entry (see section 17.1.2.1).

17.3.3. Working with Modulators

Modulators are already a way to control parameters within Bitwig Studio (see section 16.2.1). Just as nearly all device and plug-in parameters are accessible with modulator devices, all Grid device and module parameters can be controlled in exactly the same way.

In addition to their signal out ports, some Grid modules can also act as modulators. Many typical "control" devices — LFOs, envelopes, the **Steps** *Data* sequencer — have an on-board modulation routing button. And the **Modulator Out** module (in the *I/O* category) can take any Grid signal and use it as a modulator.

In addition to being usable outside of **The Grid**, modulators also have a place within Grid patches. Grid modules often have more parameters than in ports. To control parameters that don't have in ports, you can use modulators.

The only thing to know is that modulator signals operate differently from Grid signals. While Grid signals run at four-times the current sample rate and are stereo (see section 17.3.2), all modulators are mono and operate at your current sample rate. This is true for all modulators, whether they are dedicated modulator devices or Grid modules, no matter what their target is.

17.3.4. Voicing Management in The Grid

Instrument voicing modes and related topics were covered in a previous section (see section 16.2.4.1). Before spending a few words on how these settings affect **FX Grid**, it is worth taking a look at how voice management is generally handled in **The Grid**.

Various Grid modules have a parameter called *Affect Voice Lifetime*. When this parameter is enabled, the module is included in the calculation for whether each voice is still sounding and should be kept alive. Modules that have this parameter include:

› **AR**, **AD**, **ADSR**, and **Pluck** (*Envelope)*. For each of these envelope generators, a voice will stay active as long as the envelope has not reached the end of its release (for **AR** and **ADSR**) or decay (for **AD**) stage — or, in the case of **Pluck**, whichever gets to zero first. *Affect Voice Lifetime* is enabled for these envelopes by default, making them the first determinant of how long to keep voices alive.

› **Note In** (*I/O*). When this module's *Affect Voice Lifetime* is enabled, a voice will be kept alive for as long as its note gate signal is on (in a high-logic state). *Affect Voice Lifetime* is enabled by default.

› **Gate In** (*I/O*). Identical to the behavior of **Note In**, *Affect Voice Lifetime* will keep any voice on while its note gate signal is on. For **Gate In**, this parameter is disabled by default.

› **Audio Out** (*I/O*). When this module's *Affect Voice Lifetime* is enabled, a voice will be kept alive until it has fallen below the *Silence Threshold* setting for the designated *Hold Time*. The *Affect Voice Lifetime* parameter is disabled by default.

Only when all conditions being considered have finished is a voice extinguished. For example, only one envelope needs to be active to keep a voice alive. And enabling an additional *Affect Voice Lifetime* parameter can only keep notes to the same length or allow them to go longer; it will never shorten them.

17.3.4.1. Voicing "FX Grid"

FX Grid is a special device. While it is an audio effect, it is also fully responsive to note messages, allowing for the creation of effects that trigger an independent voice with each note that is played. It does this by including the voicing options of Bitwig Studio's polyphonic instruments (see section 16.2.4.1). All of the same voicing modes are available; they just act a little differently in this different context.

› *True Mono* is the default mode for **FX Grid**. In an instrument like **Poly Grid**, this mode always keeps the voice on, which can create a droning instrument (when envelopes aren't employed). Similarly with **FX Grid**, this mode always keeps the voice on, which is perfect for a traditional effect processor that should respond whenever audio of any level enters.

› *Polyphony* (whenever *Voices* is set to *2* or more) requires an incoming note signal to trigger each voice. Otherwise, the effect will not sound. This also means that voice management will be used to determine when each voice should be ended.

› *Digi Mono* is also available. It works as previously described (again, see section 16.2.4.1) and also requires note signals to produce any sound.

Since note signals are required to articulate sound in both *Digi Mono* mode and when using polyphony, notes can be received at the input of the device. This default behavior is useful on an instrument track that is already being driven by notes, but this doesn't help on an audio track.

*To change the default note source on an **FX Grid** device:* go to the **Inspector Panel** for the device and change the *Note Source* setting. This will reroute not only the various *I/O* modules that receive device input but also all pre-cords (see section 17.2.2).

Things are greatly simplified by a special *Auto-gate* option, which is in the **Inspector Panel** beneath the *Note Source* chooser. With this option enabled, a simple envelope is invisibly applied to any **FX Grid** patch (including the **Filter+** and **Sweep** devices) so that an incoming note on signal will immediately trigger and enable that voice for the length of the note. And at note off, the voice will fade out for the set *Auto-gate Release Time*.

Note

The *Auto-gate* option has no effect on an **FX Grid** set to *True Mono*, so the setting is safely on by default. This makes it possible to try an audio effect patch either in *Digi Mono* or with full polyphony — without editing the underlying patch.

Finally, the *Voice stacking* setting works as it does with instruments and can be used in any voice mode. So a direct audio processor in *True Mono* mode could be stacked — and use the *Voice Stack Spread ±* modulator to distribute different settings to each voice in the stack — without the need to use note signals at all.

17.3.4.2. Voicing "Note Grid"

As a note effect, **Note Grid** is also unique in a few ways. Like **FX Grid** among Audio FX devices, **Note Grid** is the only Note FX device that can work polyphonically. Said another way, all Note FX devices handle notes individually, but only **Note Grid** allows modulators to work in a per-note (or polyphonic) fashion. And with **Note Grid**, you can also make polyphonic Grid patches where you act on each note individually.

As a polyphonic device, all of the same voicing options are available (see section 16.2.4.1), and they are worth revisiting in a few potential contexts of **Note Grid**.

› *Polyphony* (whenever *Voices* is set to *2* or more) is the default mode of **Note Grid**. This requires an incoming note signal to trigger each voice, which is a perfect match for the device's "note processor" default preset. Within a simple note-processor context, the number of *Voices* determines the maximum number of notes that can be output at a time, so set this as high (or low) as you'd like.

› *True Mono*, on the other hand, does not require note input to stay alive. This makes it ideal for "note generator" patches, where internal triggers generate notes via one (or more!) **Note Out** modules. This mode is also ideal for systems driven by control change messages (**CC In**) without notes.

› *Digi Mono* is also available. As before, it is technically polyphonic and requires note signal to produce its overlapping mono output.

Finally, the *Voice stacking* setting is applicable here as well, allowing multiple notes to be created for each trigger, in any voicing mode.

18. Working on a Tablet Computer

Bitwig Studio supports certain models of tablet computer. Features have been built in to Bitwig Studio to create a more seamless experience on tablets. These unique software options are primarily expressed thru a special display profile.

In other areas, features of Bitwig Studio in general have unique utility in a touch-screen context. A good example is Bitwig Studio's menu system, which allows you to create shortcut buttons for any menu functions that you would like one-touch access to (see section 2.3.1).

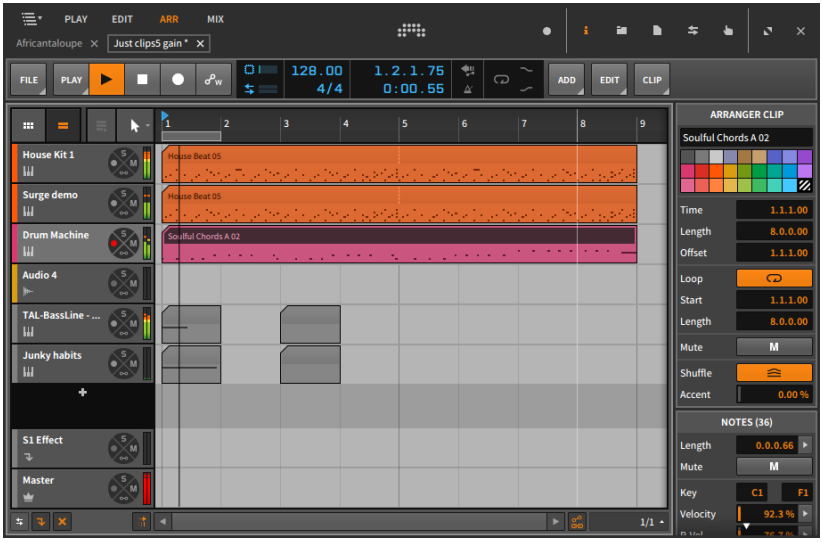
In additional, it seemed important to imagine a new way of working with this new type of hardware. So we will also get introduced to the **Radial Gesture Menu**, which magically — and in a context-sensitive fashion — appears as a halo around your finger. Dragging has never been so intuitively designed.

Note

The features described in this section may not be available if you are not on a supported operating system and computer.

18.1. The Tablet Display Profile

The *Tablet* display profile is specially designed for touch- and stylus-based interfaces. To achieve this, the layout of the window has been rearranged and a few custom solutions have been integrated.



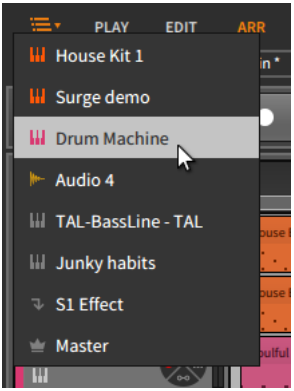
The most obvious changes from other display profiles are probably in the enlarged window header at the top of the screen, where some old friends have moved about and a new friend has appeared. Some items of note:

› *View words.* The capitalized, bold words that appear in the top left of the window represent the currently available views (with *ARR* being the currently engaged option in this example). The available views will be discussed in the next section (see section 18.1.1).

› *Panel icons.* Mingling with the window controls (see section 2.1.3) at the top right of the window is this set of icons, each representing one of the available panels (see section 2.2.1). Depending on the view selected, the available panel icons (and their corresponding panels) will change.

› *Project tabs.* These tabs represent all currently open Bitwig Studio projects (see section 2.1.1). In this display profile, the project tabs are found below the view words.

› *Track selector menu.* Located in the top left corner of the window, the track selector menu is a new item. This menu allows us to focus on any track within the current project.



The track selector menu is our only means of switching tracks in views that display only one track at a time, but it remains available in all views.

18.1.1. Tablet Views

Four views are available within the *Tablet* display profile, three of which are familiar and one of which is brand new:

› *PLAY*. The **Play View** is only available within the *Tablet* display profile. Its primary purpose is to allow note entry via your tablet computer's touch screen.

At the top of this view is a rather modest version of the Arranger Timeline, displaying a single track at a time. This makes the track selector menu and buttons necessary for switching out the single track being displayed.

You also must choose between the **Arranger Timeline Panel** or the **Clip Launcher Panel** as only one can be shown at a time.

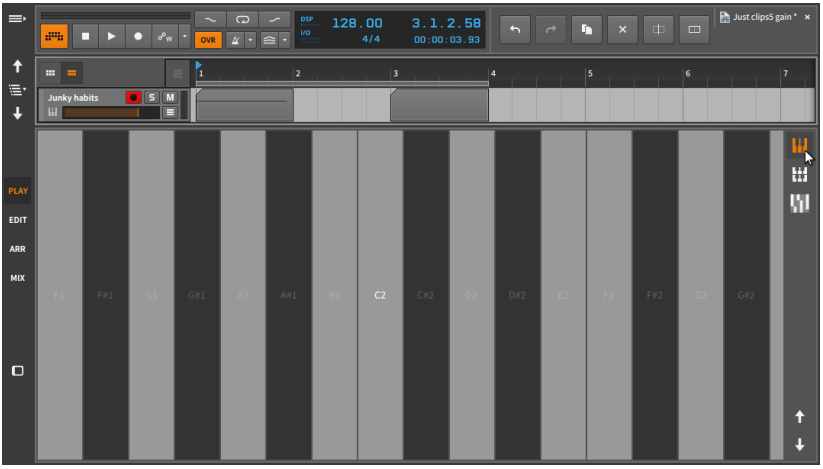
Note

You can still drag clips between the Arranger and Launcher by dragging your source clip from one panel onto the view toggle (found just above the single track header here) of the other. This is similar to dragging a clip from one project tab to another (see section 14.4.2).

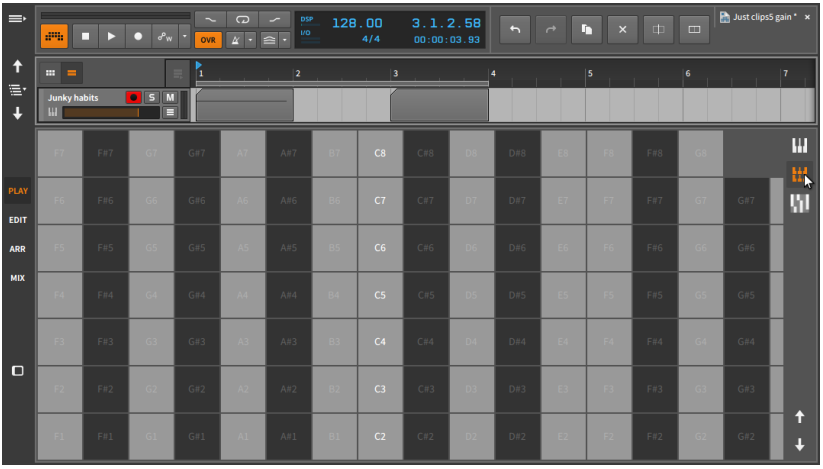
The **Device Panel** may be displayed in the center of the window. Neither the access panels nor the **Inspector Panel** are available in this view.

At the bottom of this view is the **On-Screen Keyboard Panel**, which is where note entry and monitoring is possible. There are three keyboard modes available here:

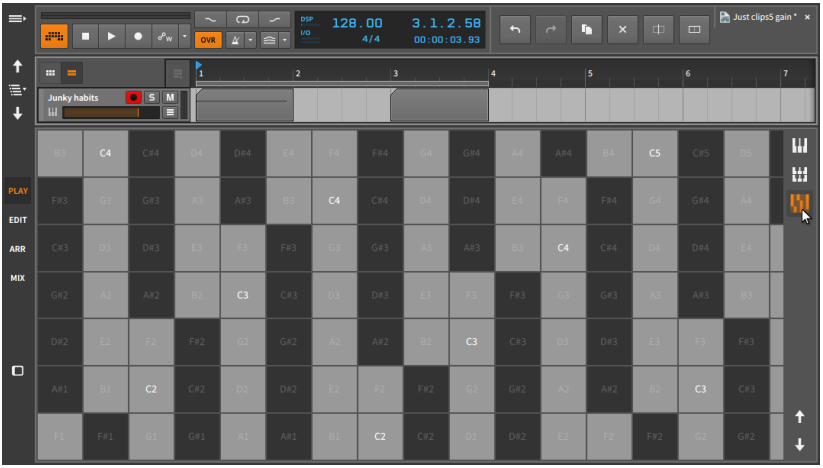
The *Piano* keyboard provides a single row of equally-sized vertical bars for playing and creating notes.



The *Octaves* keyboard shrinks the *Piano* bars into squares and stacks them in octaves, filling the available screen space with keys.



The *Fourths* keyboard is similar to the *Octaves* keyboard but stacked in fourths.

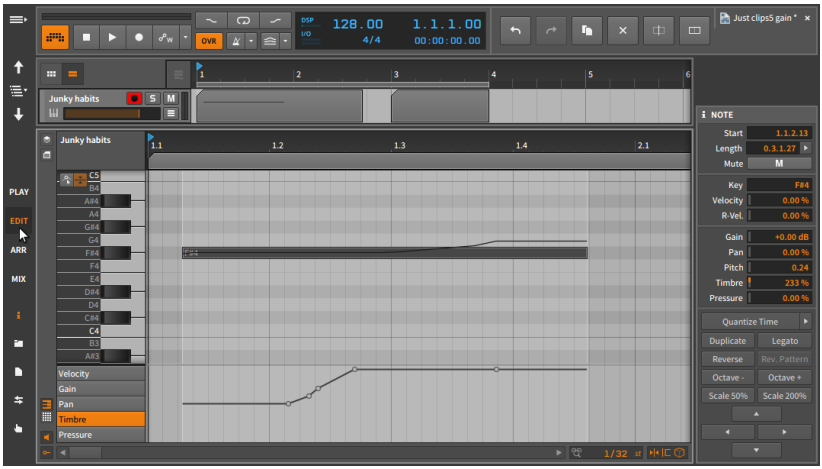


Each of these keyboard modes supports multitouch input so that multiple notes can be played at one time. While playing notes with your finger(s) or stylus, each mode also allows you to input Micro-pitch expressions (see section 11.1.3) by dragging from side to side, to input timbre expressions (see section 11.1.2.5) by dragging up and down, and

to input pressure expressions (see section 11.1.2.6) by adding and easing pressure.

Finally, the up and down arrow buttons in the bottom right of the **On-Screen Keyboard Panel** shift all available keyboard notes up or down by an octave.

› *EDIT*. This specialized **Edit View** is similar to the standard version (see section 11.3).

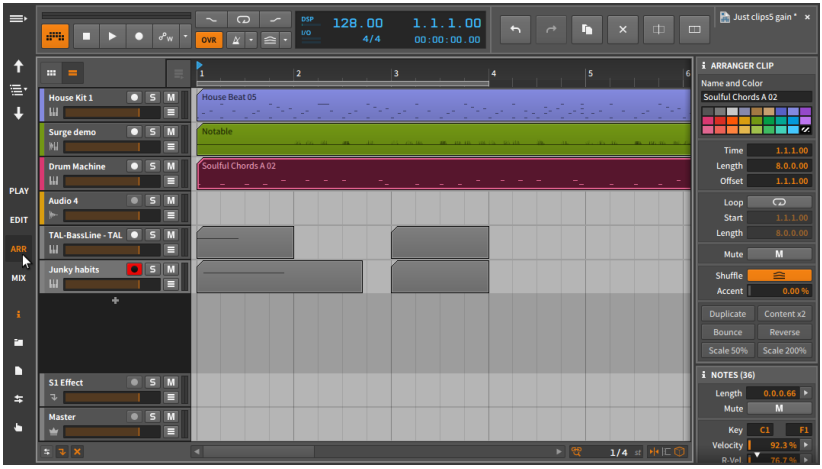


As with the **Play View**, the Arranger Timeline at top can display only one track at a time, and you must choose between viewing the **Arranger Timeline Panel** or the **Clip Launcher Panel**.

Featured beneath the Arranger Timeline is the familiar **Detail Editor Panel**.

Finally, the **Inspector Panel** and all of the access panels are available in this view, with only one being visible at a time on the right of screen. You may also notice in the image above four directional arrows in the bottom of the **Inspector Panel**. Pressing the up or down arrow will nudge any selected notes by one semitone, and pressing the left or right arrow will shift any selected notes by the current beat grid resolution (see section 3.1.2).

› *ARR*. This specialized **Arrange View** is quite similar to the standard version (see chapter 3).



Again, only the **Arranger Timeline Panel** or the **Clip Launcher Panel** can viewed at one time (not both). And the **Inspector Panel** and all of the access panels are available in this view, with only one being visible at a time on the right of screen.

› *MIX*. This specialized **Mix View** is quite similar to the standard version (see chapter 7).

The main difference here is that the optional **Device Panel** is shown above the **Mix Panel** instead of below it.

Note

The *Dual Display (Studio/Touch)* also provides a similar window for a touch-screen or tablet interface, along with a second window for a standard monitor.

18.2. The Radial Gesture Menu

To both create a quicker touch-screen workflow and preserve screen real estate (by only showing information and interfaces when they are needed), Bitwig Studio has created the unique and starkly intuitive *Radial Gesture Menu*.

When pressing various locations of the Bitwig Studio interface, a ring of options will appear around your finger (or stylus). By presenting shortcut buttons above, below, to the left, and to the right of your finger, you can then simply drag thru any of those action icons to engage that particular function as you continue dragging your finger.

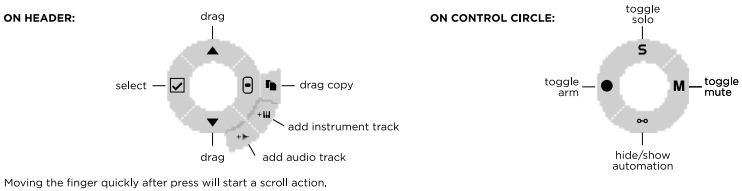
As you will find, there is a level of consistency between the **Radial Gesture Menu** configurations. For example, pressing and then dragging right tends to create an object, while pressing and then dragging left often engages an eraser mode for deleting objects.

On top of the cardinal directions, an additional, partial outer ring of functions will appear when appropriate. This comes in handy particularly when you have clicked on an object which you will be acting upon. These outer ring items are meant to be pressed by a finger that you haven't used yet.

The beauty of this system is that once you become familiar with the swipe patterns, you may start moving your fingers before the **Radial Gesture Menu** even appears on-screen. And if you do so, everything will work totally fine and even quicker.

The context here is important so the following images serve as good "shortcut charts" while you are getting comfortable with tablet computing.

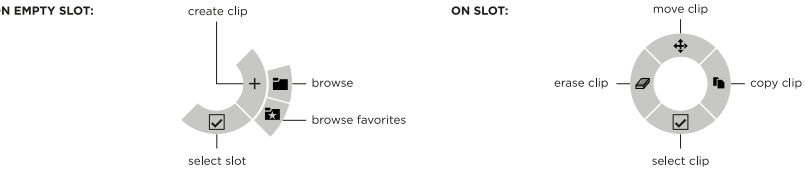
When pressing on a track's header, these options become available (including the option to move your finger slightly to the right to access a vertical track volume fader):



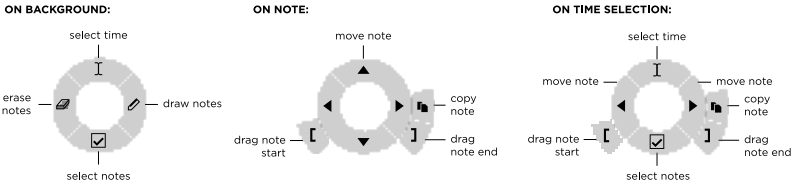
Moving the finger quickly after press will start a scroll action.

When pressing in the **Arranger Timeline Panel** either on blank space or on a clip, the following options become available:

When pressing in the **Clip Launcher Panel** either on an empty slot or on a launcher clip, the following options become available:

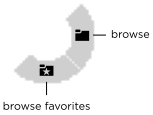


Working with notes in the **Detail Editor Panel** is particularly flexible, providing different menu configurations when you press down in a blank space, when you have notes selected, or when you have made a time selection. In these cases, the following options become available:

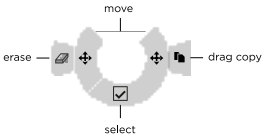


When pressing in a track's device chain within the **Device Panel**, you can press down either on empty space or on a device, making the following options available:

ON BACKGROUND:

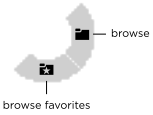


ON DEVICE:

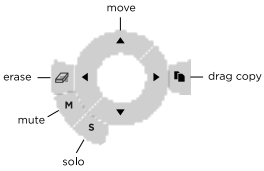


Within the *Tablet* display profile's **Play View** is a drum interface that provides *drum pads* on tracks using the **Drum Machine** device. On those drum pads, you can press down either on empty pads or on loaded ones, making the following options available:

ON EMPTY PAD:



ON PAD:



19. Device Descriptions

This appendix provides a short description of each device that comes with Bitwig Studio. The devices are organized by category. Information on using devices can be found in chapter 8, and chapter 16 provides an explanation of more advanced device concepts.

Parameter information is available in the window footer as you mouse over device parameters (see section 2.2.4). And **Interactive Help** is built in to every device, modulator, and Grid module within Bitwig Studio.

*To access **Interactive Help** for any device, modulator, or Grid module:* select the object, and then either press [F1], click *Show Help* in the **Inspector Panel**, or right-click on the object and select *Show Item Help* from the context menu. The Interactive Help window will then open with a live, editable copy of the device, and links to any relevant online videos.

Note

For more information on **Interactive Help**, see section 17.1.2.1.

19.1. Analysis

Each *analysis* device merely visualizes the signals that reach it. It makes no effect on the audio chain it is a part of.

19.1.1. Oscilloscope

A dual-trace oscilloscope, providing a time-domain representation of incoming and/or external audio signals. Each signal is given its own gain control (for visual purposes only). It can be triggered either continuously, based on a threshold level of one of the two displayed signals, or based on note messages that reach the device.

19.1.2. Spectrum

A dual-trace spectroscope, providing a frequency-domain representation of incoming and/or external audio signals. Various visualization controls are available for the *Frequency Scale* and *Range*, the *Minimum* and *Maximum Amplitude*, and the painting *Style* of each trace. And the persistence of all signals can be adjusted by the *Fall Rate*.

19.2. Audio FX

Each *audio FX* (or audio effect) device manipulates incoming audio signals before passing them onward. Incoming note signals, etc. may be used as triggers but are passed thru without change.

19.2.1. Blur

A comb-filter diffusion effect where each stereo channel has two comb filters, each with two feedback controls.

19.2.2. Freq Shifter

A frequency shifter with an adjustable frequency range. This device can also distribute the upward and downward frequency shift across the stereo field.

19.2.3. Pitch Shifter

A pitch shifter (like a musical signal transposer) with a high-resolution frequency control, a *Grain* setting for adjusting how the processing is done, and a *Mix* control, allowing harmonization.

19.2.4. Ring-Mod

A ring modulator with a definable frequency and a *Mix* control for blending the source material with the resultant sum and difference tones. The device also has *Pre*- and *Post*-processing device chains.

19.2.5. Treemonster

A ring modulator that utilizes the incoming audio signal and a sine wave whose tuning is based on that incoming signal. Pitch detection is sampled only above a set *Threshold* amplitude, can be limited with low-pass and high-pass filters, can be offset (*Pitch*) for shifting the sine tone's frequency, and can be slewed (*Speed*) to respond more quickly or ponderously. For processing, the amount of *Ring* modulation goes anywhere between a simple sine wave (at *0%*) to more harmonically complex results.

19.3. Clap

Clap drum element instruments that use incoming note signals to synthesize audio.

19.3.1. E-Clap

A monophonic electronic clap instrument made from noise, a low-pass filter, and repetitions.

The *NOISE* section comprises the instrument's sound generation parameters. The amplitude for the instrument is controlled by an AD envelope that has a short, fixed attack time and an exponential, adjustable *Decay* time.

Each incoming note message immediately triggers the amplitude envelope. And for the *Duration* time following the beginning of each note, the envelope is retriggered at every *Repeat* time interval.

For example, if *Duration* is set to *45 ms* and Repeat is set to *10 ms*, each note will trigger the amplitude envelope five times: zero milliseconds (the instant the note is received), 10 ms, 20 ms, 30 ms, and 40 ms.

Width sets the amount of stereo flutter added to each noise burst.

The *COLOR* section provides controls for the instrument's low-pass filter. *Freq* sets the cutoff frequency, and *Q* sets the amount of resonance.

The final section offers a control for the instrument's *Vel Sens.*(itivity) and a level control for its *Output*.

Nested Device Chains:

› *FX* - A chain for processing the device's entire audio output.

19.4. Container

Each *container* is a device whose primarily function is hosting other devices.

As each container has a different purpose, the primary signal I/O is listed for each device. (For more information, see section 16.1.2.)

19.4.1. Chain

(Audio in, Audio Out) A container that houses a serial audio device chain. A *Mix* control is provided for blending the dry (original signals reaching the device) and wet (processed signals exiting the device) components together, and a *Gain* control offsets the level of the dry signal only (it is not applied before processing). Any note messages that reach this device are passed out "dry," without adjustment.

19.4.2. FX Layer

(Audio in, Audio out) A container that houses parallel audio chains. Each chain has its own internal mixer controls. (For more information, see section 16.1.2.3.)

19.4.3. FX Selector

(Audio in, Audio out) A container that houses multiple audio chains. Only one audio chain at a time receives the incoming audio, but any chain that was previously receiving audio remains active until its output is silent. Also has a *Note/MIDI Source* chooser, for selecting a track to receive notes from.

See **section 19.4.5** for details on the voice *Mode* options.

19.4.4. Instrument Layer

(Notes in, Audio out) A container that houses multiple instruments in parallel. Each chain has its own internal mixer controls. (For more information, see section 16.1.2.2.)

19.4.5. Instrument Selector

(Notes in, Audio out) A container that houses multiple instruments and their associated device chains. Only one instrument chain at a time receives new notes, but each sounding note continues until its output is silent. A variety of voice *Modes* are available from the **Inspector Panel**:

› *Manual* - Target layer is set by user, controller, modulator, and/or automation

› *Round-robin* - New note triggers the next layer (for notes in series, or individual notes within a chord)

› *Free-robin* - Round-robin, but skips used voices when possible

› *Free Voice* - New note uses the first free layer. Always starts with the first layer for more predictable results. (Also ideal for loading multiple layers with HW CV Instrument to create polyphony with Eurorack hardware.)

› *Random* - New note randomly selects a layer (nice with a pile of different audio FX)

› *Random Other* - New note randomly selects a different layer (to guarantee a change each time)

› *Keyswitches* - Designated notes set the target layer (you define the lowest keyswitch; so if set to note *C2* and the Selector has 3 layers, C2 switches to layer 1, C#2 switches to layer two, and D2 switches to layer 3). Good for film scoring with different sounds and articulations.

› *CC* - Designated continuous controller sets target layer (you define a continuous controller [default CC1 - mod wheel], whose full range will morph evenly thru all layers). For example, using mod wheel to cycle thru various note FX.

› *Program Change* - PC messages set the target layer (program change messages map directly to each layer). Common output from pedal controllers, etc.

Other than *Manual*, all other modes are aware of the layer count. So adding or removing layers will just work without additional configuration.

Note

Any automation of the *Index* parameter is dynamically updated when the device chains are rearranged. And any mode other than *Manual* will ignore any automation or modulation of the *Index*.

19.4.6. Mid-Side Split

(Audio in, Audio out) A container that takes a normal stereo signal and splits it into its mid (centered) and side (panned) components, each of which is provided with an independent chain.

19.4.7. Multiband FX-2

(Audio in, Audio out) A container that splits the incoming audio at a definable frequency and provides independent chains for the audio below and above that frequency.

19.4.8. Multiband FX-3

(Audio in, Audio out) A container that splits the incoming audio at two definable frequencies and provides independent chains for the audio below the first frequency, the audio between the two frequencies, and the audio above the second frequency.

19.4.9. Note FX Layer

(Notes in, Notes out) A container that houses parallel note chains.

19.4.10. Note FX Selector

(Notes in, Notes out) A container that houses multiple note chains. Only one note chain at a time receives the incoming notes, but any chain that was previously outputting notes remains active until its output has ceased.

See **section 19.4.5** for details on the voice *Mode* options.

19.4.11. Replacer

(Audio in, Audio out) A container that filters and analyzes the level of the incoming audio signal, and when the signal rises above a set threshold, notes are generated at a set pitch and velocity. These notes

and the original (dry) audio signal are then passed to the internal *Generator* device chain.

19.4.12. Stereo Split

(Audio in, Audio out) A container that takes a normal stereo signal and splits it into its left and right channels, each of which is provided with an independent chain.

19.4.13. XY FX

(Audio in, Audio out) A container that loads up to four audio effects in parallel and allows you to crossfade their outputs.

19.4.14. XY Instrument

(Notes in, Audio out) A container that loads up to four instruments in parallel and allows you to crossfade their outputs.

19.5. Delay

Each *delay* device is a time-based processor that operates on its incoming audio signals. Each device blends one or more delayed copies of its sound with the undelayed original.

19.5.1. Delay+

Delay+ is an all-purpose delay with a fluid structure and some choice character options, making it good for most any delay situation.

The icons along the left side of the device define the available *Pattern* options:

› *Mono* (one centered circle) - Flattens the incoming signal for processing, and offers a *Pan* control for direction within the effect

› *Stereo* (two overlapping circles) - With a *Width* control and optional *Cross Feedback* (for left → right channel feedback, and vice versa)

› *Ping L* (two separate circles, larger one on the left) - Ping-pong, starting on the left side, and with *Width* control

› *Ping R* (two separate circles, larger one on the right) - Ping-pong, starting on the right side, and with *Width* control

Standard delay options are available for delay time (either in seconds, or beats plus offset for triplet, dotted, or things in between), *Feedback* amount, low- and high-pass filters for controlling feedback, and a dry/wet *Mix* control.

For delay time changes/modulations, a *Time Update Rate* parameter is available, as well as two *Time Update Model* choices:

› *Repitch* - Maintains audio output during delay time changes, making pitch effects audible

› *Fade* - Hides pitch artifacts during delay time changes

And similar to the idea of oscillator detuning, a *Detune* parameter is available in milliseconds, along with a *Stereo Detune* toggle to invert the right channel's detuning amount for instant stereo motion.

The *Feedback* parameter controls the level that output signal is scaled before it is sent back into the delay line. This setting goes from no feedback (*0.00 %*) up thru attenuated values, to full unity gain (*100 %*), even up to amplification (maxing out at *122 %*) to increase signal on each feedback iteration. Around the feedback stage are several controls and effects:

› *Level Control* keeps signal in the feedback loop from exploding, offer both a *Level Control Threshold* for when level control starts, and three *Level Control Modes*:

Soft Clip - A saturation model

Hard Clip - A clipping model

Comp. - A compressor model

› *Width affects Feedback* factors in the *Width* parameter (when available) before the feedback chain.

› A *Blur* effect is available within the initial delay process. As the feedback section returns its output to the delay input, each feedback cycle goes back thru the *Blur* function. Various *Blur Character* options are available:

No Blur - Bypass option

Soft - Short diffusion network

Wide - Short diffusion network, with broader modulation and spread

Still - Long diffusion network

Space - Long diffusion network, with broader modulation and spread

Reverse - Time-offset diffusion system

› The *Forever* mode (shown as a snowflake toggle, similar to other 'freeze' modes across Bitwig) maintains the current feedback buffer, keeping it at unity gain and not passing in any new signal.

› The *FB FX* chain allows the addition of any other Bitwig devices or plug-ins into the feedback stage, making them part of the churn.

Note

The nested *FB FX* chain uniquely provides delay compensation (when inserting devices that require it) by offsetting the delay time.

Finally, the *Ducking* knob helps incoming sounds to be heard. It does this by using an envelope follower to reduce the *Feedback* amount and the internal wet gain level by the relative *Ducking* amount.

19.5.2. Delay-1

A tempo-syncable delay with uniform delay time, offset, and feedback settings for the left and right channels.

19.5.3. Delay-2

A tempo-syncable delay with discrete delay time, offset, and feedback settings for the left and right channels. This device also has warble (*Detune* and *Rate*) and *Crossfeed*(back) settings.

19.5.4. Delay-4

A delay unit comprising four independent taps. Each tap has its own input level control, a general *FX* chain, a *FB FX* chain for its own feedback section, separate feedback controls for how much signal is fed back locally and to each of the other taps, tempo-syncable delay time, simple high-pass and low-pass filters, and output level and panning

controls. After the taps are summed, there is then a master *FX* chain, a global *Feedback* level, and a *Mix* control.

19.6. Distortion

Each *distortion* device is a shaper or other mangling processor that operates on its incoming audio signals.

19.6.1. Amp

A processor that applies the character and idiosyncrasies of various instrument amplifiers to the incoming signal.

The *PRE*-drive stage provides optional *L*(ow), *M*(id), and *H*(igh) EQ stages. In this model, the *L*(ow) and *H*(igh) bands offer high- and low-pass filters, respectively, each with variable frequency, resonance, and slope settings. The *M*(id) band is a bell filter with gain, frequency, and Q settings. Additionally, the far right of the device's interface has a nested device chain for adding additional processors to this *PRE* section.

Next comes the *DRIVE* stage, whose purpose is to overamplify the incoming signal. In addition to applying gain (via the eponymous *Drive* parameter) of up to *48 dB*, a drop-down menu offers various clipping "models" to use, such as *Class AB*, *Eulic*, *Fold B*, etc. And to further massage the overdrive, there is a *Bias* setting for offsetting the signal and a *Sag* for bending it back down when it flies too high (as indicated by the horizontal LED).

After the drive section is a *POST*-drive stage, which is identical in structure to the *PRE*-drive stage detailed above, along with a corresponding *POST* nested device chain at the far right of the device.

As with any amplifier, the last stage is (a simulation of) the speaker *CABINET*. Parameters include physical parameters of the cabinet's width, height, and depth, as well as the amount of sound reflection around the cabinet (which adds an acoustic phasing). To further shape the tone of the cabinet are a *Color* knob and a set of buttons (labeled *A* thru *H*), which offer eight discrete "hue" variations. Finally, there is a polarity control (*ø*) for the phase of this section as well as a *Mix* control to make a blend of pre-cabinet sound (heard by itself at *0%*) and cabinet processed sound (heard alone at *100%*). As always, extreme settings are useful while programming but less fruitful when used in music.

The global section of the device includes a final *Gain* setting and a global *Mix* control.

19.6.2. Bit-8

An audio degrader with assorted parameters for *CLOCK* manipulation, amplitude *GATE*, *SHAPE* (with drive and various distortion options), and *QUANTIZE* modes with fine-tuning options. The device's final output offers a *Mix* knob for the dry/wet blend, a *Wet FX* chain for inserting devices or plug-ins to process only the wet signal, an *Anti-alias* option for using using alternate processing techniques, and a stereo *Width* control.

19.6.3. Distortion

A distortion effect based on hard clipping, with a peak EQ before the clipping is applied, and high- and low-pass filters after.

19.6.4. Saturator

A logarithmic shaping effect. The top-level panel controls for *Drive*, *Normalize*, low-pass filter (both cutoff and slope/model), and *Makeup Gain* options. The full curve editor panel offers quiet and loud settings for *Threshold*, *Amount*, and *Knee* controls, as well as bipolar *Skew* controls for all three on the loud side, for treating positive and negative excursions differently.

19.7. Drum Kit

Drum kit-oriented devices that work with other instruments.

19.7.1. Drum Machine

(Notes in, Audio out) A container that routes note signals to specific chains based on their pitch. Each chain has its own internal mixer controls. (For more information, see section 16.1.2.1.)

19.8. Dynamics

Each *dynamics* device is a processor that operate on its incoming audio signals, based off of those signals' amplitude levels and trends.

19.8.1. Compressor

A compressor with standard threshold, ratio, gain, and timing settings.

19.8.2. De-Esser

A de-esser with a variable high-pass filter and monitoring option for the detection circuit.

19.8.3. Dynamics

A flexible dynamics processor that allows for either downward or upward compression on both the loud and quiet parts of the sound. The device also has a sidechain input, an *FX* device chain for the control signal, and a graphical interface.

19.8.4. Gate

A noise gate with sidechain input and an *FX* device chain for the control signal.

19.8.5. Peak Limiter

A limiter with peak level, gain, and release controls.

19.8.6. Transient Control

A transient detector that can make onsets and sustain segments relatively louder or softer.

19.9. EQ

Each *EQ* (equalizer) device is a set of parallel frequency-specific processors (for example, like a low band and high band) that operate on its incoming audio signals.

19.9.1. EQ+

A parametric equalizer of up to eight bands, with a unique, rainbow-y graphical interface. There are fourteen available modes for each band, global frequency *Shift* and *Gain* controls, an *Adaptive-Q* option (to proportionately scale Q values as gain increases), an option to display a *Reference* track within the spectrum display, and unique layouts in the **Device Panel**, **Inspector Panel**, and **Expanded Device View**.

There are also a number of mouse gestures for adding a band with a specific mode:

› Peak filters are added by double-clicking at the current mouse cursor position.

› Low-/high-shelf filters are added by dragging the left/right edges of the EQ curve.

› Low-/high-cut filters are added by dragging the left/right edges of the EQ graph (off the curve).

› Notch filters are added by dragging the lower edge of the EQ graph.

Different mouse cursors are shown to identify each interaction's filter mode.

19.9.2. EQ-2

A two-band parametric equalizer with resonant filter modes and a graphical interface.

19.9.3. EQ-5

A five-band parametric equalizer with resonant filter modes and a graphical interface. The device also has global controls to morph the strength (*Amount*) and placement (*Shift*) of the EQ curve.

19.9.4. EQ-DJ

A three-band equalizer with definable crossover frequencies and mute controls for each band.

19.10. Filter

Each *filter* device is a frequency-specific processor that operates on its incoming audio signals.

19.10.1. Comb

A comb filter effect with frequency and bipolar feedback controls.

19.10.2. Filter+

A dead-simple FX box, for deploying any waveshaper and filter from *The Grid* directly onto a track

› Pick one of ten *filters* from three categories:

Structural choices for classic circuits:

Low-pass LD - A ladder filter, with variable slope and nonlinear option

Sallen-Key - 16 various low-, high-, and band-pass configurations

SVF - Highly resonant multimode (high-, low-, band-pass & notch) filter

Comb - A comb filter with timed feedback & dampening

Inspired options that speak:

Low-pass MG - A Moog-style low-pass filter, including drive character

XP - An Oberheim-style multimode filter, with 15 configurations)

Vowels - A morphing vowel filter, with various models, pitch and frequency offsets

Character ideas for something new:

Fizz - A nested filter circuit that can sparkle, shimmer like a phaser, or bump

Rasp - A filter that adds brightness around the cutoff, so it can scream or whimper

Ripple - A hyper-resonant circuit for playful feedback, subharmonics, or even distortion

› Pick one of 14 *waveshapers*, sorted for you:

One Knob classics with a singular control:

Chebyshev - Nonlinear shaper that can target harmonics

Distortion - Gentle distortion

Hard Clip - Simple, hard clipper

Quantizer - Signal resolution reducer

Wavefolder - Reflects each cycle back on itself

Parametric options that offer more control:

Diode - Classic circuit model, used for biasing and clipping

Rectifier - Independent positive and negative attenuators

Saturator - Waveshaper with loud/quiet settings + bipolar skews

Transfer - A freely drawable, segmented waveshaper, with BWCURVE-file support

Character ideas, for unique paths and simple control:

Push - Soft clipper with a detailed curve

Heat - S-shaped clipper that starts soft but can drive hard

Soar - Soft wave folder that makes the quietest parts loud

Howl - Wave folder that puts different parts of the signal into loud focus

Shred - Non-linear wave folder for subtle cancellation or big-time artifacts

› Signal flow is simple: audio input → waveshaper → filter

› *Pre FX* and *Post FX* chains are also available, for nesting other devices or plug-ins

› A modulation section offers two built-in sources:

A stereo **LFO** module gives four waveshapes with sync-able *Rate* and *Timebase* controls

The incoming audio itself provides a second modulation source, with optional low-pass filtering and rectification (to make the modulation go in only one direction)

Both *LFO* and *Audio Mod* sources are normalled to the filter's cutoff buss, with attenuators on the filter

These sources are also available as modulators for free control elsewhere, including shaper *Drive* controls, other filter controls, or control of any nested devices in the *Pre FX* and *Post FX* slots

› Additional Inspector controls for *Stereo Spread* and *Wet Gain*

› Other parameters are available in the device's Expanded Device View, which exposes the embedded Grid patch. These include:

LFO *Skew* (to bend the shape), *Phase*, *Phase Offset (R)* (for the right channel, controlling the stereo effect), *Bipolar*, and *Sync to Global Transport* toggle (on by default)

Pitch Buss toggle (with = icon) to not attenuate the audio mod source, giving it a ±10 octave range

A simple **Pan** module, for placing the signal

› Being a Grid-powered device, polyphony and voice stacking are uniquely available in this audio FX device

› By right-clicking the device header, functions are available to:

Convert to Sweep, for bringing all settings into that device (see section 19.10.6)

Convert to FX Grid, for full patching control

19.10.3. Filter

A multimode filter with pre- and post-gain.

19.10.4. Ladder

A multimode ladder filter with a built-in LFO, envelope, and envelope follower to modulate the filter's frequency.

19.10.5. Resonator Bank

A bank of six resonant filters that have frequency, resonance, and gain controls. The device also has global controls to morph these three controls as well as keyboard tracking to offset the filters' frequencies based on incoming note signals.

19.10.6. Sweep

A performable effect device, combining and blending a waveshaper and two filters from *The Grid*

› Everything said about **Filter+** (see section 19.10.2) is true of **Sweep**, except **Sweep** has a second filter slot and generalized controls for this setup

› *Joint Frequency Control* provides control of both filters in a range ±3 octaves

An *Invert* option flips the direction that the *Joint Frequency Control* applies to filter B, allowing you to move their cutoffs in opposite directions

› The *Routing Blend* control smoothly moves thru various device configurations:

At *0 %*, only filter A → waveshaper is heard

50 % is a parallel routing, with device audio input going straight to both filter A (then out) and to waveshaper → filter B (then out)

100 % is fully serial, with device audio input going to filter A → waveshaper → filter B

Positions in between blend these routings, for a continuous range

› *Stereo Pan* is similar to the *Joint Frequency Control*, except it applies the same stereo adjustment to both filters

Positive (rightward) settings move right channel cutoffs up and left cutoffs down, and negative (leftward) settings move left channel cutoffs up and right cutoffs down

Good, quick stereo-ization control

19.10.7. Vocoder

Imposes the timbre of one sound onto another. Has separate chains for the *Modulator* (sound source) and *Carrier* (affected sound), but the incoming audio signal is also used as the modulator. Allows being *8* and *80* filter bands for each section (optionally stereo), along with *Slope* and *Bandwidth* controls. Also provides *Formant* and *Brightness* controls for the modulator signal; *Attack*, *Release*, and *Freeze* controls for the analysis bands; and *Ceiling* and *Floor* controls, for limiter-/expander-type behaviors.

19.11. Hardware

Each *hardware* device sends signals and/or messages to devices beyond Bitwig Studio (such as hardware synthesizers and effect units). This can include transmitting and/or receiving audio signals, control voltage (CV) signals, and clock messages.

19.11.1. HW Clock Out

Two paths for CV clock signal output, to be sent thru set ports of your audio interface. Each path can transmit *Clock* signals at a set interval, a signal only at transport start (*Play* mode), a signal only at transport *Stop*, or a signal for every *Note* received.

19.11.2. HW CV Instrument

A router that sends the incoming note messages out of the system as CV signals. One path is used for *Pitch CV Out*, and one is used for *Gate Out* triggers. *Audio In* is then returned to the system and output from this device.

19.11.3. HW CV Out

A vehicle for sending a CV signal out a set port of your audio interface via a parameter knob. An *AC* switch is provided, as is a low-pass filter control for applying lag to the outgoing signal.

19.11.4. HW FX

A router that sends the incoming stereo audio signal out of the track and system, and then returns another stereo signal back.

19.11.5. HW Instrument

A router that sends the incoming note signals out of Bitwig Studio as MIDI, and then returns the resultant audio.

For note and MIDI output, settings include the MIDI output port to use as well as whether to send all messages on a single MIDI channel or to preserve *Same Ch.*(annel) set for each per note/event in Bitwig Studio. A special *Use MPE* option can be used instead, converting note expressions (see section 11.2.2.4) to appropriate channel voice messages, dynamically allocating channels as necessary, and providing a pitch-bend range parameter. And a toggle is available in case you want to send *MIDI Clock* messages to this MIDI port.

The audio return section includes the audio input to use, a gain level applied to that signal, and a latency offset amount that is set in samples (negative settings adjust the audio to be earlier).

Like most instruments, nested device chains for *Note FX* and audio *FX* are provided.

19.12. Hi-hat

Hi-hat drum element instruments that use incoming note signals to synthesize audio.

19.12.1. E-Hat

An electronic hi-hat instrument made from a blend of noise with a comb filter, FM synthesis, and a one-band equalizer. An XY grid interface is also provided as an alternate means of controlling several parameters.

The section at the top left contains *Attack* and *Decay* times for the AD envelope, along with a contour control for the shape of the decay segment. This global envelope shapes the output of the entire instrument.

The red *COMB* section governs the comb filter that processes the noise generator's output. Parameters include cutoff *Freq*(uency), a bipolar *Feedback* control, and the wet/dry *Mix*. In the XY grid, dragging the red *C* ball adjusts the *Freq* control with horizontal movements and the *Mix* control vertically.

The blue *FM HIT* section controls the carrier of a classic FM operator pair, which creates the impact sound of the hi-hat. The *Freq* knob at left sets the carrier's frequency. This unit has its own AD envelope, which has a short, fixed attack time and an exponential, adjustable *Decay* setting. (Note that a longer decay setting may be interrupted if the global AD envelope has a shorter overall duration.) Finally, the *Mix* knob controls the balance between the noise and FM portions of the instrument. In the XY grid, dragging the blue *H* ball adjusts the *Freq* control with horizontal movements and the *Mix* control vertically.

The yellow *HIT MOD* section provides controls for the modulator of the FM operator pair. The *Freq.* knob adjust the modulator's frequency, and the *Amount* control is the index (or intensity) of modulation applied to the carrier. In the XY grid, dragging the yellow *M* ball adjusts the *Freq.* control with horizontal movements and the *Amount* control vertically.

The orange EQ section controls a simple high-pass filter that receives the blended noise and FM sounds. The cutoff frequency is set with the top numeric control (in hertz or kilohertz), and the control below with a resonance icon represents the filter's Q. In the XY grid, dragging the orange vertical bar from left to right controls the cutoff frequency.

The final section offers a control for the instrument's *Vel Sens.*(itivity) and a level control for its *Output*, along with a *Width* setting for the amount of stereo flutter added to each noise burst.

Nested Device Chains:

› *FX* - A chain for processing the device's entire audio output.

19.13. Kick

Kick drum element instruments that use incoming note signals to synthesize audio.

19.13.1. E-Kick

An electronic kick drum instrument with optional pitch modulation.

The *GEN* section contains parameters for controlling and processing the instrument's slightly rectified sine oscillator. The frequency of this oscillator is set by the *Tune* knob, and its level is controlled by an AD envelope that has a short, fixed attack time and an exponential, adjustable *Decay* time. The *Click* option adds impact to the sound by doubling portions of it, and the *Tone* control sets the cutoff frequency of a gentle low-pass filter.

The *P. MOD* section concerns a separate AD envelope generator that controls pitch modulation applied to the oscillator. You can adjust the *Amount* of modulation in semitones, the *Decay* time, and the shape of that decay segment with the contour control.

The final section offers a control for the instrument's *Vel Sens.*(itivity) and a level control for its *Output*.

Nested Device Chains:

› *FX* - A chain for processing the device's entire audio output.

19.14. Modulation

Each *modulation* device is a processor that manipulates incoming audio signals with an LFO, etc., influencing its function.

19.14.1. Chorus+

Chorus, with four different *Character* modes, each with its own DSP architecture and different *X* + *Y* controls:

› *CE* - A synth-style friend, with different inspirations in the tone (*Tone* + *Width*)

› *DD* - Subtle, 80s, and coming from all sides (*Time* + *Balance*)

› *8v* - Eight voices swirling thru caverns of feedback (*FB* + *Width*)

› *x2* - Classic voice doubling circuit (*Time* + *Width*)

19.14.2. Chorus

A chorus effect with an adjustable LFO with phase offset for the right channel (*R Phase*).

19.14.3. Flanger+

Flanger, with four different *Character* modes, each with its own DSP architecture:

› *DP* - That digital, scrapy cousin who chews up sound

› *MX* - A firm, pedal-style classic

› *TFX* - Smooth and sparkly, with some edge

› *WA* - Stronger, but subtly delicate

Also has an *Alternate Character* toggle to switch to a subtle variation of the selected mode; *Stereo-ize* option, to invert the right modulation signal; a *Manual Override* mode, which disables the internal LFOs and presents a *Mod* parameter that the user can target with their own modulation signals; and an *Added Dirt* option, to insert a little noise for additional coloring.

19.14.4. Flanger

A flanger effect with an adjustable LFO and feedback parameters for both magnitude (*Feedb.*) and phase (*Neg.*). This device can be set to *Retrig*(ger) on incoming note messages.

19.14.5. Phaser+

Phaser, with four different *Character* modes, each with its own DSP architecture:

› *GS* - Our spikey #1 friend

› *EHx* - Classy and smooth, with silky motion

› *MX* - A raspy devil, but solid

› *MF* - Pleasantly greasy and deep

Various *Modulation Curve* options are available:

› *Phaser* - A more "traditional" modulation curve (the default)

› *Speaking* - A shape that can produce "vowel" sounds

› *Barber* ↑ - Barber pole-esque effect, spinning upward

› *Barber* ↓ - Barber pole-esque effect, spinning downward

Also has an *Alternate Character* toggle to switch to a subtle variation of the selected mode; *Stereo-ize* option, to invert the right modulation signal; and a *Manual Override* mode, which disables the internal LFOs and presents a *Mod* parameter that the user can target with their own modulation signals.

19.14.6. Phaser

A phaser whose *Frequency* setting is controlled by an **LFO** modulator module by default. Separate phase (*ø*) controls exist for the frequency on the *L*(eft) and *R*(ight) channels so that you can keep everything synced but interesting. Also includes a feedback (*FB*) control and a high-pass filter with adjustable cutoff frequency and slope (from anywhere between *2-Pole* and *32-Pole*).

19.14.7. Rotary

A rotary-speaker emulation that modulates the signal's placement in the stereo field.

19.14.8. Tremolo

An amplitude modulator that is controlled by an LFO of various waveshapes. This device can be set to *Retrig*(ger) on incoming note messages.

19.15. MIDI

Each *MIDI* device transmits various MIDI messages or modifies them via the track's device chain. This is useful for sending messages to plug-ins or to external hardware (when used in conjunction with Bitwig's *hardware* devices) or simply modifying the channels in use by a device chain.

19.15.1. Channel Filter

A processor for ignoring incoming note or MIDI messages by channel.

19.15.2. Channel Map

A processor for remapping incoming note or MIDI messages by channel.

19.15.3. MIDI CC

A vehicle for sending any MIDI continuous controller (CC) messages via eight parameter knobs. A global MIDI *Channel* can be set.

19.15.4. MIDI Program Change

A vehicle for sending a MIDI program change message at project load and/or manually via the *Send* trigger button. The MIDI *Channel* can be set, and all or individual components of a two-byte bank select message

can be sent (using CC 0 as the most significant bit [MSB] and/or CC 32 as the least significant bit [LSB]).

Additionally, the device has a nested *Chain* with two special options. The *Scoped* option contains the program change and bank select messages, sending them only to devices in the nested chain. And the *Anti Click* option fades the nested chain's output using an adjustable *Decay* time before transmitting the MIDI messages.

19.15.5. MIDI Song Select

A vehicle for sending a MIDI song select message at project load and/or via a manual *Send* trigger button.

19.16. Note FX

Each *note FX* (or note effect) device manipulates incoming note messages before passing them onward. Incoming audio signals are passed thru without change.

19.16.1. Arpeggiator

An MPE-friendly arpeggiator, which cycles thru the notes being held in a set order. Timing is set rhythmically or as milliseconds. For each step, the specified note(s) is output for a set duration with a scaled velocity and pitch offset amount, as well as a global *Randomize* option for velocity, timing, and duration. 17 note patterns are available, in addition to three different *Octave Behavior* modes:

› *Broad* takes each additional octave in sequence, stacking them up with possible irregular/repeating patterns. (This is the default.)

› *Thin* flattens & sorts all notes into one linear shape. (This was the Arpeggiator behavior in v3.1.x and earlier.)

› *1 by 1* executes the full pattern in each octave before moving to the next octave.

19.16.2. Bend

A Micro-pitch expression generator, bending from a relative *Starting Pitch* to the note's original pitch. *Bend Shape* sets the curve for the pitch

glide. *Duration* of the bend can set it either real time or tempo-relative 16th notes. A *Pre-delay* setting is also available, for postponing the pitch bend (the same as on most of the envelope modulators, etc.).

Useful for:

› Adding glissando to any device

› New sound design possibilities by adding a quick pitch envelope on any instrument

› Many responsive possibilities, such as modulating bend amount or time with velocity, etc. etc.

› Setting a pitch curve once before 'stacked' instruments, such as with the **Instrument Layer** or **Instrument Selector** containers (or their **Note FX** brethren)

› An alternate concept of 'glide,' starting relative to the new note (instead of from the pitch played previously)

19.16.3. Dribble

A note repeater that bounces each note until gravity wins. *First Bounce* time (set in real time or tempo-relative 16th notes) is the time that the initial bounce will last *if* maximum velocity is played. *Damping* controls the speed/height loss for each successive bounce; at *0.00 %*, bounce height remains the same. *Shortest Bounce* is a time threshold for ending bounces before they become too close together — or not. *Hold Last Note* optionally keeps the final bounce note held out (as long as the triggering note is still held).

Useful for:

› Adding some trailing character to single-note lines

› Giving per-note 'delays' to chords, especially when each note has a slightly different velocity

› Creating a decaying, 'organic' note repeat effect

› Modulating *Damping* to keep notes at fix repeat lengths, whenever anything happens (like the global *Fill* button is engaged)

19.16.4. Echo

A tempo-syncable note repeater. The number of *Repetitions* can be set, or an infinite feedback mode can be enabled. Within the feedback/

repetition loop are numerous parameters, including *Time* (to make repeated notes relatively closer together or spaced further apart) with a *Random*(ization) option, *Gate* (to scale the length of repeated notes), *Velocity* scaling, and *Pitch* scaling (that can be filtered to only apply within a defined range).

19.16.5. Harmonize

A note transposer that conforms incoming notes based on the active note messages of a different track (set as the *Harmony Source*). To improve the device's logic, a *Pattern Key* should be defined.

19.16.6. Humanize

Randomizes aspects of notes. *Chance* sets the likelihood that each arriving note will be sent on. *Timing* defines the maximum lateness that can be randomly selected for each note. If *Allow Early Notes* is on (±), then delay compensation is used to make the *Timing* range either late or early. *Velocity* sets a bipolar amount of randomization applied at each note on.

Useful for:

› Giving some life to the timing of any input

› Lightly mutating any sequenced passage, making it is different with each repeat

› Randomizing any triggered note clip, where the *Allow Early Notes* option can feel right

› Loosening any predictable output, so that rigid rhythms or probability can be modulated or automated

› Randomly spreading note timing for FX that care about note order (like **Strum**, or **Arpeggiator** using the *Flow* pattern, etc.)

19.16.7. Key Filter

A note transposer, which can correct or remove notes that do not match a set key and mode. Notes can also be shifted before the transposition is applied.

19.16.8. Latch

A note sustainer that either holds the current note until the next one is received (*Simple* mode), only triggers every other note received (*Toggle* mode), or only triggers every other note around a defined velocity threshold (*Velocity* mode). This logic is applied on a polyphonic, per-pitch basis by default, but it can also be applied in a *Mono* fashion.

19.16.9. Micro-pitch

Micro-/macro-tuning of each note type, and octave. Defines the root note (which is kept in tune), and then tuning values for all other pitch classes, as well as the octave. Also provides an *Amount* control (for moderating all pitch offsets back toward standard equal temperament) and for frequency offset around A3 (traditionally *440 Hz*).

19.16.10. Multi-note

A chord builder, allowing up to eight notes to play for each received note message. Each note unit is defined relative to the incoming note's pitch and velocity, with an additional velocity Spread amount (for randomizing each note unit's velocity output) and *Chance* (to set the likelihood that each note unit triggers). And when *Live Note Updating* is on, modulating each unit's *Enable* or *Pitch* parameters produce immediate updates, even for trigger notes that are already being held.

Realize that if you want the original incoming note to be passed thru unaffected, one of the eight note paths must be used for that purpose (with pitch and velocity offsets of *0*).

19.16.11. Note Delay

A utility for delaying all notes that arrive, with an option to also *Delay* (note) *Offs* or to send them immediately. Good for having layers trigger at different times.

19.16.12. Note Filter

A filter for notes. Range is defined by low and high values for both *Key* and *Velocity* parameters, coupled with a *Mode* switch to either *Keep*

only the notes within that range (inclusive), or to *Remove* notes in that range and pass all others.

19.16.13. Note Length

A device to set incoming notes to a fixed, optionally tempo-syncable *Length*. Note velocity can also be set to a *Fixed* value, and notes can be set to *Trigger* either at the start (*Press*) or *Release* of each note. Additionally, *Release* notes can use either the *Fixed* velocity, the velocity of the original note *On* message (which is sometimes more consistent), or the velocity of the triggering note *Off* message (which not all hardware supports well).

19.16.14. Note Repeats

A note repeater and pattern generator. Each note received is retriggered at a *Timebase* (either seconds or tempo-relative units), multiplied by a *Rate*. *Gate Length* is set as a percentage of the repeat rate, or an option to *Hold until Next Trigger* (the musical fermata icon) is also available. *Velocity Decay* sets the change amount of each successive repeat's velocity, either down or up. *Chance* sets the probability that each individual repeat will occur. And *Disable Repeats* is a mappable 'kill switch' that sustains each note after its next repeat starts (and passes new notes directly thru), allowing the repeat function itself to be disabled or modulated.

Two *Pattern* modes are available for organizing note repeats into larger forms:

› *Burst* lines up all note retriggers in a row

› *Euclid* tries to evenly space the note repeats, which can be rhythmically satisfying

Additionally, *Length* sets a pattern to be between *2* and *32* steps, *Density* is the percentage that the pattern gets filled (how many of the steps will play), and *Rotate* pushes the start of the pattern either forward or backward.

And when a *Pattern* mode is used, 'Accents' can be created by keeping a number of the repeats at their original velocity, and attenuating velocity of the 'non-accented' repeats. *Count / Strong Notes* sets the number of current repeats that will be accented. *Low Velocity (Non-accents)* sets the attenuation applied to the non-accented notes. *Opposite / Flip Accent Pattern* inverts the placement of accented and non-accented

notes. And *Keep Accents / Always Play Accents* guarantees that each accented note will play every time, regarding of the *Chance* setting.

Useful for:

› Repeating each incoming note at a set rate

› A performance-ready note effect, particularly with mappings to *Disable Repeats* (for switching the entire effect on and back off) and *Velocity Decay* (so that retriggers can be ramped quieter and then louder)

› Creating probabilistic repeats with the *Chance* parameter

› Giving life to long chords with a low *Chance* setting but the *Hold until Next Trigger* option on, keeping each note sustained until the eventual retrigger arrives

› All manner of note pattern fun, for drum parts or anything else

19.16.15. Note Transpose

A simple note pitch shifter, which can shift the incoming notes by a set number of octaves and/or semitones. A *Fine* control is also provided for shifting by fractions of a semitone.

19.16.16. Quantize

Shifts notes toward the next *Timing Interval*, with an option to follow the global *Groove* or not. *Amount* sets how far each note is moved from its original position toward the next grid line. *Forgiveness* is a threshold for how late notes can be before they are held until the next beat, setting a percentage of the time range to not be quantized at all.

Useful for:

› A real-time performance quantizer, placing all incoming notes exactly on the next grid line

› With an *Amount* of *100 %* and *Forgiveness* at *0.00 %*, a complete 'robot-izer'

› Aligning incoming notes across a beat range (perhaps followed by **Strum**, etc.)

› Creating new rhythmic patterns, particularly by feeding it a fast **Arpeggiator** or **Note Repeats**, etc. with a middle *Forgiveness* value

19.16.17. Randomize

A randomizer of any/all expressions at the start of each note, including:

› *Pitch*, with additional parameters options for whether pitch is *Quantized* to semitones and whether its randomization is *Bipolar*

› *Velocity*, randomized around the current value (taken from the note source and used wherever velocity is mapped, including from the *Expressions* modulator)

› *Timbre*, randomized around the current value (used wherever mapped from the *Expressions* modulator)

› *Pressure*, randomized around the current value (taken from the note source especially for MPE controllers, and used wherever mapped from the *Expressions* modulator)

› *Pan*, randomized around the current value (mapped to the panning of each individual note, and available from the **Pan In** Grid module)

› *Gain*, randomized around the current value (mapped to the gain of each individual note, and available from the **Gain In** Grid module)

Useful for:

› Turning any note clip into an 'anti-loop,' with different parameters for each note that plays

› Giving individual *Pan* positions to each note of a chord or arpeggio

› Creating tiny pitch instability to the original notes, or on a second **Instrument Layer** for 'analog' drift

› Adding additional *Timbre* and *Pressure* variety to any MPE-friendly sound

› Shifting drum notes to sometimes trigger different drum elements

19.16.18. Ricochet

Treats notes as bouncing balls in a room. When balls collide with each other (or with the room's walls), a new note is triggered at that velocity.

Ball Speed scales the speed of each ball (relative to its velocity). *Ball Radius* sets the size of the balls. *Ball Damping* is the amount of slowdown applied after each collision.

Ball Launch Mode determines the direction in which new balls are fired:

› *Random* picks a random direction each time

› *Bar Sync* uses relative bar position, with bar start and end facing straight up (at 12 o'clock)

› *Manual* gives control to the *Ball Launch Angle* parameter for manual setting or modulation, etc.

Room Sides can be set anywhere between *3.0* and *8.0*, including decimal values for some asymmetry. *Room Orientation* turns the room position or spins it. *Room Spatialization* uses each ball's position to effect that note's panning (↔) and timbre (↕) expressions. And *Sound on Initial Notes* sets whether the initial note being received triggers a note, or not (which can be nice on a second layer, etc.).

Useful for:

› Creating an algorithmic variation of your note clip, which is either reproducible (*Bar Sync*) or new each time (*Random*)

› Generating one-shot timbre/pan envelopes by using a big *Room Spatialization* and maximum *Damping*

› Going 'the full Eno' by setting a slow *Speed*, triggering a non-sustaining sound, and holding down the sustain pedal

› Creating a 90s-style delay, but with... Note FX...

19.16.19. Strum

Fragments your chords, playing them one (or more) note at a time. Speed of strumming is set as a *Timebase* (either seconds or tempo-relative units), multiplied by a *Rate*. Strum direction can be set to *Strum Up* (playing lowest note first, then upward) or down. And *Number of Steps* allows sequencing a pattern of up to four steps, so that the next chord played can change strum direction. *Stride* sets the number of notes that are output at a time, and *Grace Period* is the time window for each chord to be collected before strumming begins.

Useful for:

› Animating played chords at a steady rate

› Slight speed-ups or slow-downs, by slowly modulating the strum rate

› Alternating up–down strum patterns, to borrow some plucked/bowed patterns

› A one-shot arpeggiator, running up or down the played notes once

› 'Smart' moving quantize, taking your playing and spreading each note to this beat or the next

19.16.20. Transpose Map

A note transposer, which can remap each note class (for example, so every D becomes an F#, etc.). Notes can also be shifted before the transposition is applied.

19.16.21. Velocity Curve

A (piecewise) velocity shaper with three definable breakpoints.

19.17. Organ

Each *organ* is an instrument emulator that uses incoming note messages to synthesize audio.

19.17.1. Organ

A tonewheel organ.

The drawbars section contains nine standard gain faders (the vertically higher the fader, the louder the gain), each of which represents the level of the respective drawbar harmonic. In order, these harmonics are:

› Fader 1 - *Sub*, or one octave below the fundamental (in organ notation, 16′ [feet])

› Fader 2 - *5th*, or one fifth above the fundamental (5 1/3′)

› Fader 3 - *Primary*, or the fundamental (8′)

› Fader 4 - *8th*, or one octave above the fundamental (4′)

› Fader 5 - *12th*, or one octave and a fifth above the fundamental (2 2/3′)

› Fader 6 - *15th*, or two octaves above the fundamental (2′)

› Fader 7 - *17th*, or two octaves and a major third above the fundamental (1 3/5′)

› Fader 8 - *19th*, or two octaves and a fifth above the fundamental (1 1/3′)

› Fader 9 - *22nd*, or three octaves above the fundamental (1′)

The top of the drawbars interface also has a drop-down menu for the type of oscillator modeling being used for each harmonic. Choices include:

› *Rich* - closer to a traditional tonewheel oscillator, a bit more complex than a pure sine wave

› *Pure* - closer to a pure sine wave

› *Full* - an even richer waveform

Additionally, the small *R* icon to the right toggles retriggering of oscillator phases for successive notes, which creates a gentler, less clicky sound.

Beneath the drawbars section is a small bank of other instrument controls. The bipolar *Pitch* control adjusts the frequency of all oscillators together. This control is set in semitones, with a range of three octaves in either direction (from *-36.00* to *+36.00*). The *Glide* setting sets the amount of time it takes for a new note to smoothly transition from the previous pitch. And the global amplitude envelope has controls for its *Attack* and *Decay* times.

The final section at bottom offers controls for the instrument's panning, gain (the speaker icon), and final *Output* level.

Nested Device Chains:

› *Note* - A chain for processing incoming note messages before they reach this device.

› *FX* - A chain for processing the device's entire audio output.

19.18. Percussion

Percussion instruments that use incoming note signals to synthesize audio.

19.18.1. E-Cowbell

An electronic cowbell instrument with optional pitch control.

The *GENERATOR* section contains parameters for controlling the instrument's two oscillators. The *Pitch* of the first oscillator is directly controllable, and the pitch of the second oscillator is set as an *Offset* of the first. Alternatively, the graphical keyboard toggle button allows you to have the first oscillator track incoming note messages (with the second oscillator's still being set as a relative *Offset*). To the right of the keyboard toggle is a crossfader that sets the balance between the two oscillators, and the *Shape* control lets you determine the oscillators' waveshape.

The low-pass *FILTER* section provides standard *Freq*(uency) cutoff and *Reso.*(nance) controls.

The *RING* modulation section allows you to set the *Freq*(uency) of a modulating sine wave and the dry/wet *Mix* of the ring modulation effect. If *Mix* is set to the far left, the ring modulator will not be heard.

The *AEG* section provides *Attack* and *Decay* times for the AD-style amplitude envelope generator.

The final section offers a control for the instrument's *Vel Sens.*(itivity) and a level control for its *Output*.

Nested Device Chains:

› *FX* - A chain for processing the device's entire audio output.

19.19. Reverb

Each *reverb* device is a time-based processor that tends to elongate the incoming signal, producing distinct room effects or imbuing other tones and sounds.

19.19.1. Convolution

Conceptually, real-time *convolution* is a continuous DSP method for imprinting one sound onto another, running every sample of the incoming signal 'thru' the loaded *impulse* (or *impulse response*) in its entirety. This merges the two sound, effectively multiplying their spectra together so that only frequencies existing in both signals make it to the output — and in relative proportion. This can work for a real, captured space, the tone of any equipment (like a particular amp or mix buss), generated impulses (some of which are in the factory library under *Synthetic*), OR with any audio (a long held piano note? a rhythmic pattern that continues getting louder?) serving as impulse.

As a device in Bitwig, **Convolution** is straightforward with quick adjustment controls for reverb, coloring, or anything else convolution can do. The required *impulse* can be 1-channel (mono) or 2-channel (stereo), and 4-channel ("true stereo") impulses are also supported. Dragging any audio file from one of Bitwig's browsers or the OS's file manager onto **Convolution** will load the first 45 seconds of it as an impulse. Or drop a Bitwig clip from your project or browser onto the device to bounce it directly to an audio impulse.

Note

If a file conversion is necessary, the impulse will be saved in the current project folder's *Impulses* folder as a BWIMPULSE file.

Clicking the folder icon or impulse name in the top of the device loads the *impulse browser*, which visualizes all factory impulses and those from your library, making it easy to see the character of any file beside its length, category, and channel count. The *Import...* button in the bottom of the impulse browser allows the bulk import of audio as impulses, converting them and placing them in your Bitwig user library's *Impulses* folder.

The *Start* and *End Time* positions within the impulse can be adjusted visually (similar to **Sampler**), or with numeric controls in the Inspector. Toggling to the *Volume Envelope* mode switches the central graphic section of the panel to controlling start and end gain values, as well as a

midpoint's timing and gain (again, all shown on numeric controls in the Inspector). A red dot is shown on the *Volume Envelope* toggle when any gain changes are occurring, similar to how the presence of automation marks parameters.

The *Tune* parameter resamples the impulse, changing its pitch and length by the set semitone amount. *Brightness* offers a tilt EQ, which favors the high end when turned to the right, or the low end on the left. *Pre-delay* time, *Wet Gain* amount, and dry/wet *Mix* parameters are also available, as well as a *Wet FX* chain for adding devices and plug-ins to process only the wet output portion.

19.19.2. Reverb

An feedback-based algorithmic reverb effect with distinct controls for *EARLY* reflections and for the later dense reflections (*TANK*). The *TANK* is split into three assignable bands with relative delay times for the low and high bands. This device also has a graphical interface and uniquely offers a *Tank FX* chain for inserting any Bitwig device or plug-in into the feedback cycle of the effect, as well as a *Wet FX* chain for using devices to process only the processed signal.

19.20. Routing

Each *routing* device allows the redirecting of a track's signal path. To achieve this, a router often contains audio and/or note chooser menus for addressing an incoming or outgoing signal to the appropriate destination, including destinations outside of Bitwig Studio.

As each routing device has a different purpose, the primary signal I/O is listed for each device.

19.20.1. Audio Receiver

(Audio in, Audio out) A router that imports audio signal from any designated project source.

19.20.2. Note Receiver

(Notes in, Notes out) A router that imports note signals from any designated project source.

19.21. Snare

Snare drum element instruments that use incoming note signals to synthesize audio.

19.21.1. E-Snare

An electronic snare drum instrument made from two tunable oscillators, a noise generator, and resonant high- and low-pass filters.

The *OSC 1* section houses the primary sine oscillator, whose frequency and decay time can be set directly with the *Tuning* and *Decay* knobs, respectively.

The *OSC 2* section contains a secondary sine oscillator whose settings are relative to oscillator 1. Accordingly, the frequency of oscillator 2 is set as an *Offset* from oscillator 1 in semitones, and oscillator 2's decay time is set with the *Decay X* parameter as a percentage of oscillator 1's decay time.

The *NOISE* section contains parameters related to the noise generator. This includes *Attack* and *Decay* times for the AD envelope that controls level, along with a contour control for the shape of the decay segment. And the *Width* knob sets the amount of stereo flutter added to each noise burst.

The *MIX* section is for controlling the balance between the three generator elements. *Osc* controls the balance between oscillator 1 and oscillator 2, and then *Noise* controls the balance between both oscillators and the noise generator.

Next comes the *FILTER* section, which has a high cut (or low-pass) filter for processing output from both the oscillators and the noise generator. Any noise generator signal is then passed to a low cut (or high-pass)

filter. Individual cutoff frequency controls are available for both the *High Cut* and the *Low Cut* filter, and a single *Q* parameter controls resonance for both filters.

The final section offers a control for the instrument's *Vel Sens.*(itivity) and a level control for its *Output*.

Nested Device Chains:

› *FX* - A chain for processing the device's entire audio output.

19.22. Spectral

These *spectral* devices operate in the frequency domain, working with hundreds of individual frequency bands. The current devices are all audio processors that analyze the incoming signal to group them, then putting the groups onto channels that work in the regular amplitude domain.

19.22.1. Freq Split

Divides the signal into adjacent frequency groups and then sorts them into one of the four channels, for individual mixing and processing.

The *Frequency Split* parameter sets the number of splits across the frequency spectrum. *Split Insertion Direction* sets whether additional splits are added from the right/high edge of the spectrum (←), from the left/low edge of the spectrum (→), or in the middle (↔). *Crossfade Amount* determines the overlap between splits. These three parameters are the primary split parameters. So with the device's default settings, a *Frequency Split* number of *16* with a *Split Insertion Direction* putting new splits on the right/high end (←) means:

› The 1st, 5th, 9th, and 13th splits land in channel *1* (red)

› The 2nd, 6th, 10th, and 14th splits land in channel *2* (blue)

› The 3rd, 7th, 11th, and 15th splits land in channel *3* (yellow)

› The 4th, 8th, 12th, and 16th splits land in channel *4* (magenta)

› And if the *Crossfade Amount* was increased from *0.00 %* (full isolation between bands) to *50.0 %*, then each split would spend its first 25 % crossfading with the previous split, and its last quarter crossfading with the next split.

Split Nudge slides the frequency splits by an offset value, so a setting of *+2.00* pushes the contents of channel *1* into channel *3*, channel *4* into channel *2*, and so on. *Split Spin* also slides the frequency splits but relative to the entire spectrum; so a setting of *-10.0 %* slides all the splits a tenth lower in the entire spectrum. Whether being subtle (*Split Nudge*) or extreme (*Split Spin*), modulating either of these parameters shows the "filter bank" quality of this device, creating new phasers and more when each channel is loaded with different audio FX.

Split Bend curves the frequency split pattern around a new midpoint, either moving the midpoint downward and putting splits closer together in the lower frequencies (negative values) or moving the midpoint higher and having splits closer together in the upper frequencies (positive values). *Split Pinch* kinks the frequency split pattern, either bunching more splits around the midpoint (positive values) or putting more splits into the sides (negative values). In the device's display, the interactive white dot controls *Split Bend* when dragged left to right, and *Split Pinch* when dragged up and down.

There is also a *Spectral Limiter* option (enabled with the *Limiter* toggle in the output section). When enabled, this caps each individual frequency band at the set *Spectral Limiter Threshold* so louder bands will be capped and any band below this threshold will be unaffected. When the *Spectral Limiter* is enabled, the *Threshold* is shown in the device's display with a horizontal orange bar that can be dragged up or down.

There are two *Spectrum Display* modes: *Pre* shows the analysis data (and no audio processing from the channel controls, unless the *Spectral Limiter* is on); *Post* shows each channel's output audio, post-processing.

19.22.2. Harmonic Split

Tracks the fundamental of the incoming sound, for splitting *Nonharmonics* (gray) to one channel and dividing harmonics between *Harmonics A* (orange) and *Harmonics B* (turquoise) channels, for individual mixing and processing.

The *Harmonics Pattern* parameter decides how harmonics are distributed between the *A* and *B* channels:

› The default setting of *2* places every 2nd harmonic in *A* channel, so the *A* channel will be odd harmonics (1st, 3rd, 5th, etc.), and the *B* channel receives all even harmonics (2nd, 4th, 6th, etc.).

› A setting of *4*, for example, would place the 1st, 5th, 9th, etc. harmonics in channel *A*, and all other harmonics (2nd, 3rd, 4th; 6th, 7th…) into channel *B*.

› Higher values lead to narrower results in the *A* channel — and the potential for more extreme processing without "getting into trouble".

› A setting of *1* is special, routing only the fundamental to channel *A* and all other harmonics to channel *B*.

Nonharmonic Sensitivity is a relative control for how picky the harmonic vs. non-harmonic split is. A higher value allows less audio into the *Nonharmonic* channel — and more audio into the two *Harmonics* channels.

The *Maximum Harmonics* Inspector parameter allows you to limit the number of harmonics being tracked, acting as a "ceiling" when you want fewer frequencies in the *Harmonics A* and *Harmonics B* channels.

Fundamental tracking can be adjusted with several analysis parameters, all colored purple:

› *Tilt* favors the high frequencies (when positive) or low frequencies (when negative), useful when the desired fundamental is filtered in the sound, etc.

› The *Low-cut Frequency* and *High-cut Frequency* parameters narrow the frequency tracking area.

› An amplitude *Detection Threshold* can be set, keeping moments that stay below that level in the *Nonharmonics* channel.

› In the device's display, the *Detection Threshold* and *High-* & *Low-cut Frequencies* are interactive lines that can be dragged.

› The device display also illustrates the currently-detected fundamental with a white dot and moving crosshairs.

There are two *Spectrum Display* modes: *Pre* shows the analysis data (and no audio processing from the channel controls); *Post* shows each channel's output audio, post-processing.

19.22.3. Loud Split

Uses two thresholds to separate the *Quiet* (green), *Mid* (yellow), and *Loud* (red) portions of the incoming sound, for individual mixing and processing.

A *Higher Threshold* (red) sets the level where stronger signals are considered *Loud*. A *Lower Threshold* (green) sets the level where weaker signals are considered *Quiet*. Any signal falling between the two

thresholds is considered *Mid*. And each threshold has its own *Knee* value, for setting the transition (and effective crossfade) between adjacent channels.

In the device's display, both thresholds are visualized as horizontal lines, which can be dragged up and down. When clicking a line's dot handle on the right, only the outer band of that threshold will be heard while the mouse is held. And [ALT]-dragging on either threshold adjusts its *Knee* parameter.

Relative Loudness Mode follows the level of the incoming sound, treating *0.0 dB* as the strongest band at any given moment. (This mode uses its own *Relative Higher Threshold* and *Relative Lower Threshold* parameters.)

Rise Time sets the number of blocks before a softer signal fully transitions up into a louder band, like a "resistance" parameter. *Fall Time* sets the number of blocks before a louder signal fully falls down into a quieter band, like a "decay" parameter. And *Tilt* is an analysis parameter, favoring the high frequencies (when positive) or low frequencies (when negative) before the channel splitting is applied.

There are two *Spectrum Display* modes: *Pre* shows the analysis data (and no audio processing from the channel controls); *Post* shows each channel's output audio, post-processing.

19.22.4. Transient Split

Separates the *Transients* (short, unstable sounds; colored yellow) and *Tones* (periodic or pitched sounds; colored blue), for individual mixing and processing.

The *Transient Type* Inspector parameter switches between two different algorithms for which type of *Transient* is being looked for: *Percussive* mode searches for typical impact transients, good for drums or other things that "click" and "smack"; *Noise* mode looks for noisy smears, or even reverb residue.

Transients Decay sets a time (in blocks) for extending detected *Transients*, allowing them to release. *Tones Smoothing* sets a time (in blocks) for extending detected *Tones*, allowing them to decay. And the *Analysis Bias* slider on the left of the displays skews the detection to favor more *Transients* (positive values) or more *Tones* (negative values).

Tilt Amount is an analysis parameter that is colored yellow as it is oriented to how it affects the Transients channel; its effect is the

opposite for the *Tones* channel. And the *Tilt Mode* Inspector parameter changes the method of *Tilt* applied, between a *Standard* model that favors the *Transient* channel's high frequencies (when positive) or its low frequencies (when negative), or a *Contour* approach that subtly adjusts the mid frequencies vs. the highs & lows.

Transient/Tones Blend is the slider at the end/right of the spectral section, for balancing the audio output before each signal reaches its channel.

There are two *Display Style* modes: *Waveform* shows a split amplitude-domain representation of the two groups; *Sonogram* offers the recent frequency-domain history for each group.

19.23. Synth

Each *synth* device generates its audio either from rudimentary source material, from audio files used as samples, or sometimes from coming in via sidechain. Incoming note messages drives these instruments to produce audio output.

19.23.1. FM-4

A four-oscillator FM synthesizer with frequencies set as ratios with offsets, optional self-modulation, a noise generator with a resonant low-pass filter, and a modulation matrix. Each row of the matrix represents one of the four oscillators as a modulation destination, and each column is labeled with the modulation source it represents.

On the far left are four identically equipped sections, representing the four sine oscillator units of the instrument. Oscillator *1* is at top, oscillators *2* and *3* follow, and oscillator *4* is at bottom.

In each unit, the two central controls help determine the sine oscillator's frequency. Each incoming note message is multiplied by the top, unlabeled numeric control to set the oscillator's base frequency for that voice. For example, playing a note message of A3 (440Hz) with a setting of *1.00* triggers that oscillator at 440Hz. Playing A4 again with a setting of *2.00* would set the oscillator to 880Hz, just as a setting of *0.50* would tune the oscillator to 220Hz in this example. This system also allows you to see the frequency settings of two oscillators as a ratio, a very handy way of thinking in FM synthesis.

The numeric control at bottom is an offset, allowing you to then detune each oscillator by a number of Hertz.

The *Mod* control at the right of each oscillator unit attenuates the output of the oscillator to all frequency modulation connections (this does not affect the audio output of the oscillator). Similarly, the oscillator number in the left of each unit is a button for enabling/disabling that oscillator for modulation purposes (again, the audio output for each oscillator is unaffected by the setting of this toggle).

To the right of oscillator *1* is the *N*(oise) section. This noise generator is configured somewhat similarly to the oscillators, with a global *Mod*(ulation) level control at its far right and a button to enable/disable modulation usage at the far left (shown as *N*).

Between these controls are knobs for the *Cutoff* frequency and *Q* of a low-pass filter that the noise generator is connected to, as well as a *Drive* control that can boost the output signal by up to *+48.0 dB*.

While the matrix section that follows is somewhat cryptic, it is the heart of the instrument's frequency modulation model. This table shows the individual amounts of modulation between the five generators that we have just discussed. The columns represent the *sources* of modulation, and the rows represent the four oscillator units, which are the potential frequency modulation *destinations*. These signal attenuators go from *0* (no signal/modulation) to *999* (the fullest amount of modulation available). In this sense, you could also think of these gain values as percentages of modulation.

Note

Just remember that the settings in each oscillator and noise generator unit impact the matrix values. Each modulation amount in the grid is scaled by the source's global *Mod*(ulation) level and

is completely bypassed if the modulation enable/disable switch is flipped off.

As an example, let's look at the third column, which is labeled *3*. Each of the rows in this column represents one of the respective oscillator units as a destination and the amount of attenuation applied to that particular modulation connection. The first row in this column shows the amount that oscillator *3* modulates the frequency of oscillator *1*. Accordingly, the second, third, and fourth rows control the amount that oscillator *3* modulates the frequencies of oscillators *2*, *3*, and *4*, respectively. In any other column, the destinations would be the same but the source would be either a different oscillator (the numbered columns) or the noise generator unit (column *N*).

And as the example above indicates, oscillator *3* can be set to modulate itself — to effectively "feed back" — by setting the third row attenuator to a value greater than zero. The same is true of all four oscillator units when the output of an oscillator is set to modulate its own matrix input.

The section to the right of the *N*(oise) and matrix sections is the instrument's audio mixer. Each generator unit has an attenuator for setting the amount of signal that will reach the instrument's audio output. Just as the matrix and other modulation controls did not affect the audio level of each unit, these gain controls do not affect modulation levels in any way.

Beneath the matrix section are controls that belong to the amplitude envelope generator unit (*AEG*). This module affects the entire instrument's audio output level and can also be routed to additional modulation destinations. After the modulation routing button at left are standard *A*ttack, *D*ecay, *S*ustain, and *R*elease controls.

To the right of the mixer is a thin vertical output section, containing assorted global parameters. *Pitch* allows the pitch of all oscillators to be offset from an octave down (*-12* semitones) to an octave up (*12*). Targeting this parameter with an LFO is an ideal way to create vibrato on this synth. The *Glide* setting is the amount of time that it takes for each new note to smoothly transition from the previous pitch to the current one. And at bottom are per-voice *Gain* and *Pan*(ning) controls, along with an *Output* level control.

Modulation Sources:

› *AEG* (amplitude envelope generator) [polyphonic] - The signal of this instrument's amplitude envelope generator module. (The routing of this module to the instrument's amplitude is hardwired.)

Nested Device Chains:

› *Note* - A chain for processing incoming note messages before they reach this device.

› *FX* - A chain for processing the device's entire audio output.

19.23.2. Phase-4

A phase-manipulation synthesizer (including phase distortion and phase modulation techniques) with four unique oscillator units, a system of global controls for altering the oscillator units' phase distortion and phase modulation settings together, a unique tuning system for setting frequency relationships, a multimode filter capable of audio-rate modulation, and more.

Each oscillator unit is functionally identical and is distinguished by its color and corresponding letter (*R*ed, *B*lue, *Y*ellow, and *M*agenta). The letter at the top left of each oscillator doubles as a bypass toggle for that oscillator unit. There are three ways to control the frequency of each oscillator, all located in the top row of each oscillator's controls, just above the knobs.

The small keyboard icon with arrows around it toggles keyboard tracking on or off. When keyboard tracking is enabled, an offset in semitones (*st*) can be set just below. And when keyboard tracking is disabled, a fixed frequency can be set, either in hertz (*Hz*) or kilohertz (*kHz*). To the left of this, a *RATIO* is then applied to the frequency, allowing you to set oscillators relative to each other (in the fashion of *1:1*, *3:1*, *1:2*, *0:1*, etc.). Finally, an offset frequency (in *Hz*) can be applied from the control at the far right of this section. Above that numeric control are two icons for switching between uniform monophonic detune (the single circle icon) or stereo detune (the two overlapping circles), which applies the set detune amount to the left channel and its inverse to the right.

Next, each oscillator has controls for *phase distortion*. The primary control is *SHAPE*, which affects the overall amount of phase distortion applied. Above the shape knob is a text menu that can be dragged up or down to change its setting. This is the *algorithm* being used for

phase distortion. Each algorithm determines both the source waveform and the path the waveshape will traverse as the *SHAPE* parameter is increased. Beside the algorithm is a numeric for *formant* control. Settings above *1* insert additional sine cycles into the original waveshape. And just above formant is a phase (º) control. This value sets the offset of the original waveform (in degrees). But beyond adjusting the cycle position of the waveform, this control also affects the phase distortion algorithm, producing unique results.

Of the *phase modulation* parameters, first is the *MOD*(ulation) knob, which sets the maximum amount of phase modulation allowed from any oscillator source. The individual levels of modulation are then set by the four smaller knobs to the right, each colored to represent their oscillator. (Yes, this includes potential feedback from the selected oscillator itself.)

You will notice an arrow between the *SHAPE* and *MOD* parameters. If the arrow is pointing toward *MOD*, then phase distortion is applied before phase modulation. And if the arrow is instead facing *SHAPE*, then phase modulation is being applied before phase distortion. Clicking on the arrow rotates it.

And finally, each oscillator has a knob on the far right with a speaker icon beneath it. This is an output volume control, setting to what degree this oscillator is heard as audio.

To the left of the four oscillator units is the global controls section. At top are a *PITCH* control for adjusting all oscillator frequencies in semitones and a *GLIDE* control for setting all portamento times. At bottom are global *SHAPE* and *MOD* knobs, allowing you to change the maximum phase distortion and phase modulation (respective) amounts for all oscillators together. Additionally, the X–Y pad allows control of these two parameters together by clicking and dragging the *4* ball. And if any individual oscillator has its own *SHAPE* and *MOD* controls set below maximum, you may see a ball of that oscillator's color on the X–Y pad as well.

To the right of the oscillator units is the *FILTER* section. The top row sets the filter's mode, toggling between various filter types: a gentle low-pass filter, a 4-pole low-pass filter, a gentle band-pass filter, a 4-pole band-pass filter, a gentle high-pass filter, a 4-pole high-pass filter, a band-reject filter, and a disabled mode, respectively.

The next row, from left to right, contains drive (*DRV*), resonance, and feedback controls. Centered beneath these controls is the oversized cutoff frequency control.

To the left of the cutoff frequency knob are four more small knobs, each colored to match an oscillator unit. These bipolar attenuators set the amount that each oscillator unit is allowed to modulate the filter cutoff

frequency. And to the right of the large filter frequency control are attenuators for how much keyboard tracking and the filter's envelope generator each affect the cutoff.

Beneath the filter settings are two identical rows of parameters, one for the filter envelope generator (*FEG*) and one for the amplitude envelope generator (*AEG*). Each starts with a green routing button for assigning additional modulation destinations. Each is followed by standard *A*ttack, *D*ecay, *S*ustain, and *R*elease controls. Finally, each envelope has a control for how much note velocity scales its output.

The final parameter section has three controls, representing panning, per-voice gain (with a speaker icon that glows red when drive is being applied), and a master *OUT*(put) level knob.

Modulation Sources:

› *FEG* (filter envelope generator) [polyphonic] - The signal of this instrument's filter envelope generator module.

› *AEG* (amplitude envelope generator) [polyphonic] - The signal of this instrument's amplitude envelope generator module. (The routing of this module to the instrument's amplitude is hardwired.)

Nested Device Chains:

› *Note* - A chain for processing incoming note messages before they reach this device.

› *FX* - A chain for processing the device's entire audio output.

19.23.3. Polymer

A hybrid modular synthesizer with slots for selecting one oscillator, one filter, and one envelope generator. The available modules are also used in **The Grid**, but they are available directly from the **Device Panel** in **Polymer**.

Current available modules include:

› 10 oscillator options: **Sine**, **Triangle**, **Pulse**, **Saw**; **Union** (blending pulse, saw, and triangle waves), **Phase-1** (with five phase distortion algorithms, and phase modulation feedback), **Swarm** (an eight-voice unison saw/sine oscillator), **Bite** (exponential FM, hard sync, PWM, and ring mod from dual oscillator feedback); **Wavetable** (with custom unison modes and processing options), and **Scrawl** (a freely drawable, segmented oscillator)

› 10 filter options: **Low-pass LD** (ladder model), **Sallen-Key** (in 16 configurations for various modes and slopes), **SVF** (state-variable, with low-, high-, and band-pass modes, and an extended resonance range), **Comb** (configured as a filter, with *Feedback* and *Dampening Frequency* controls); **Low-pass MG** (inspired by Mr Moog, including mix buss saturation via the *Drive* control), **XP** (inspired by Mr Oberheim, with 15 filter configurations), **Vowels** (modeling vowel sounds with different datasets and filter configurations); **Fizz** (a nested filter for spreading harmonic nodes around), **Rasp** (a nested filter that can scream or whisper), and **Ripple** (a nested filter with hyper-resonant modes)

› 5 envelope generator options: **ADSR**, **AR**, **AD** (with a looping option), **Pluck** (exponential string-style decay), and **Segments** (a freely drawable, segmented envelope generator with unique looping modes), with a modulator routing option for controlling additional parameters

Other front panel parameters include:

› *Sub* oscillator with waveform, octave, and blend controls

› A ↑*SYNC*↑ mode that hard syncs the primary oscillator to the sub oscillator

› A *Phase Modulation Amount* knob for the primary oscillator, set between zero and *800 %* for phase modulation from the sub oscillator

› *Noise* blend control

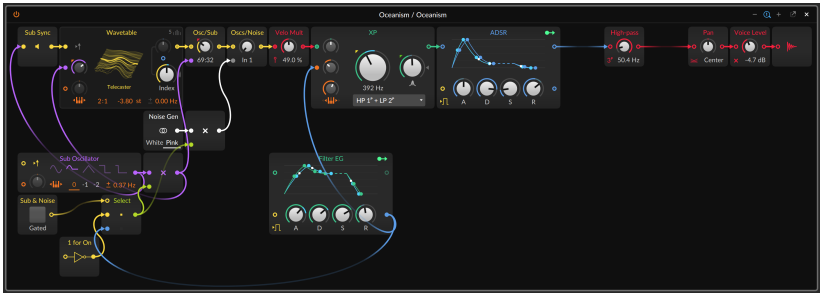
› Filter envelope (*FEG*) generator with ADSR controls, free modulator routing button, and toggle to also envelope the sub oscillator and noise generator outputs

› By right-clicking on the background of most filter modules, a *Resonance Limit* parameter is available, setting the point where clipping (and saturation) begins within the filter's resonance; adjusting this setting together with the filter's *Drive* can greatly change the 'color' of each filter

› High-pass filter cutoff

› Controls for *Pitch*, *Glide*, Velocity Sensitivity, Gain (pre-*FX* chain), Panning, and a summed *Out* level (post-*FX* chain), as well as a nested *Note FX* chain

Detail controls and a schematic view of **Polymer** are available in the Expanded Device View, which is a performance view of the underlying Grid patch. This view also exposes all module panel controls, for both adjusting and modulating them.



*To convert an instance of **Polymer** into **Poly Grid**:* right-click on the device header of **Polymer** (in the **Device Panel**), and then choose the *Convert to Poly Grid* function.

19.23.4. Polysynth

A subtractive synthesizer with two highly dynamic oscillators, an assortment of methods for "blending" those oscillators, a noise generator, a multimode filter, various waveshaping modes, and endless possibilities.

This instrument starts with two substantial oscillator units. Oscillator 1 is found on top, and oscillator 2 is on bottom. As the oscillators are completely identical in structure and parameters, we will only discuss them once.

At the top of each oscillator unit is a dynamic waveshape display. As oscillator parameters are adjusted, this display will reflect the current waveshape generated by this oscillator.

The *Pitch* of an oscillator can be adjusted by a perfect fifth up or down (from *-7.00 st* [semitones] to *7.00 st*). Below this *Pitch* knob is an octave switch in organ foot notation. From the default setting (*8'*) the oscillator can be set from two octaves down (*32'*) to three octaves up (*1'*), or any octave in between.

The *Shape* control allows you to blend three distinct waves. At the center position, you get only a sawtooth wave at the current pitch. Moving from the center position to the left cross-fades into a pulse wave that is one octave up. Moving from the center position to the right cross-fades into a saw that is one octave up. Below this *Shape* knob is a pulse width control that affects both the pulse wave at the left position and the sawtooth at the right position.

A *Sub* pulse wave that is one octave down can also be blended in. Below this *Sub* level knob is a pulse width control for the sub wave.

Each oscillator unit can also be synchronized to a tunable oscillator. The *Sync* knob controls the frequency of the master sync oscillator as an offset from the oscillator unit's pitch (from *0.00* semitones [unison; no effect] to *60.00* semitones [five octaves up]). The reset button (*R*) beneath the *Sync* knob causes the oscillator unit to return to its initial phase for each incoming note.

Next, the lower control determines the number of voices used for each note played by this oscillator unit. Settings range from *1v* (one single voice per note) to *16v* (16 voices per note). When more than one voice is active here, the *Unison* knob above becomes active, allowing you to set the maximum detuning per voice from no detuning (*0 cents*) up to a full semitone (*100 cents*). And beside *Unison* is a control for oscillator width, which is also enabled when the oscillator is using more than one voice. This control adjusts the panoramic spread between the various oscillator voices in use. And beneath that width control is a panning setting for this one oscillator.

The next section starts with various blend operator options at the top of the device. The operator selected determines how oscillators 1 and 2 are mixed together into a composite signal. Options on the top row offer slight variations on the standard mixing/crossfading approach, and the choices on the bottom row are a bit more exotic and surprising. While trying out these unique combinations, don't forget that this parameter too can be a modulation target. A short note on each blend operator:

› *MIX* - A linear mix of oscillators 1 and 2.

› *NEG* - A linear mix of oscillators 1 and a negated version of oscillator 2, potentially creating phase cancellation.

› *WIPE* - A mix of oscillators 1 and 2 but using a slightly nonlinear ramp, resulting in stronger signals at the extremes.

› *AM* - Amplitude modulation of oscillator 1 from oscillator 2. The *1/2* knob is essentially an attenuator for how much modulation is being applied to oscillator 1.

› *SIGN* - A mix of oscillator 1 and a version of oscillator 2 that has oscillator 1's polarity applied to it.

› *MAX* - A mix of oscillator 1 and a hybrid signal reflecting the maximum level of oscillator 1 and 2.

The section below is a grab bag of features that primarily relate to the blend and mixing of the instrument's generator units.

In the first row, the *1/2* knob controls the blend between oscillator 1 and oscillator 2 using the blend operator that was selected above. The *Noise* knob then controls the balance between both oscillators and a white noise generator. And the final knob on this row is actually a control for the filter section. This filter FM parameter allows an audible-rate oscillator of fixed frequency to modulate the filter's cutoff frequency.

The second row of this section starts with an optional high-pass filter that comes after the signal sources are blended. The first parameter contains both a cutoff frequency control and a mode selector via the drop-down menu beneath the knob. The next knob is a resonance control for this high-pass filter. And last is a pre-filter *Drive* control, for either amplifying or attenuating the blended signal at the end of this stage.

The third row starts with global frequency controls. The bipolar *Pitch* control adjusts the frequency of both oscillators. This control is set in semitones, with a range of three octaves in either direction (from *-36.00* to *+36.00*). And the *Glide* setting sets the amount of time it takes for a new note to smoothly transition from the previous pitch. Last is a feedback control (*FB*). By engaging this setting, the spectrum of the sound expands a bit.

The instrument's filter module is found in the next section. The first control sets the filter's mode. This graphical control at top can toggle between seven filter types: a gentle low-pass filter, a 4-pole low-pass filter, a gentle band-pass filter, a 4-pole band-pass filter, a gentle high-pass filter, a 4-pole high-pass filter, and a band-reject filter, respectively.

The following row includes filter controls for the cutoff frequency (with a horizontal arrow icon, suggesting frequency), the amount of resonance

being applied (with a peak-shaped icon), a waveshaping control (more on that in a moment), a keyboard tracking control that determines how much the cutoff frequency is controlled by incoming note pitches, and a control for how much and at what slope the filter envelope generator (*EG*) affects the cutoff frequency. (And don't forget the filter FM control that lives in the previous section and was mentioned there.)

The odd control out in that last row was the waveshaping parameter in the center. This nonlinear distortion offers several modes in the drop-down menu beneath the amount knob. If you want more or less of this effect, try adjusting the *Drive* control from the previous section. Or even modulate *Drive* and/or the shaper amount.

Below the filter section are the instrument's two envelope generators. The filter envelope generator (*FEG*) is normalled to the filter cutoff frequency (via the *EG* attenuator knob in the filter section). The amplitude envelope generator (*AEG*) controls the instrument's main amplifier. Both envelope generators can also be used as modulation signals for other purposes by using their modulation routing buttons. And each envelope generator has standard *A*ttack, *D*ecay, *S*ustain, and *R*elease controls of their own.

The final parameter section contains four knobs: controls for *Vel*(ocity sensitivity), *Gain*, *Pan*(ning), and *Output* level.

Modulation Sources:

› *FEG* (filter envelope generator) [polyphonic] - The signal of this instrument's filter envelope generator module.

› *AEG* (amplitude envelope generator) [polyphonic] - The signal of this instrument's amplitude envelope generator module. (The routing of this module to the instrument's amplitude is hardwired.)

Nested Device Chains:

› *Note* - A chain for processing incoming note messages before they reach this device.

› *FX* - A chain for processing the device's entire audio output.

19.23.5. Sampler

A sampler that can handle single or multiple samples in zones (with resizeable mapping editors) and has multiple play modes, a multimode filter, and numerous modulation opportunities.

This instrument plays back one or more audio files as its source material. The instrument's primary section focuses on the current source material with a waveform display and numerous parameters surrounding it. The options here differ in cases where a single audio file is loaded or when multiple audio files are being used.

When only one audio file is loaded into the instrument, all relevant parameters appear within this section.

Relevant sample parameters appear above and below the waveform editor. Section labels are gray. The top row contains blue clickable icons and numeric controls for general parameters. On the bottom row are *PLAY* parameters in yellow and *LOOP* parameters in green, colors that are also used within the waveform display to visualize these settings.

Starting in the top row, first is a folder icon along with the loaded sample's filename. When the folder button is clicked, the **Pop-up Browser** is called up so you can select a different audio file to load. You can also drag the sample's filename into the **Arranger Timeline** or **Clip Launcher** to create an audio clip.

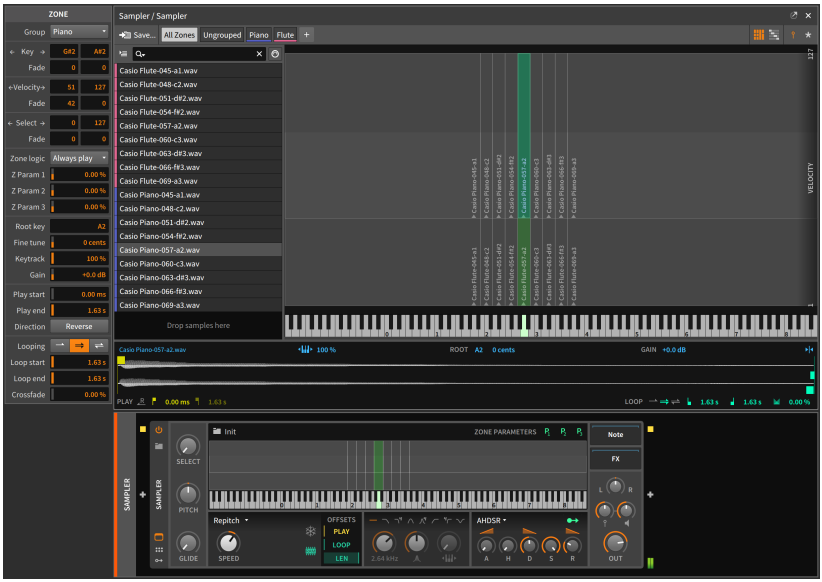
Next is a piano keyboard button followed by a percentage value for the amount of keyboard tracking being used. When set to *0 %*, any note played will trigger the sample at its original pitch. When set to *100 %*, incoming note pitches alter sample playback based on their distance from the *ROOT* settings (of root note and cents offset). Clicking the piano icon toggles between full keyboard tracking (*100 %*) and none (*0 %*), but values in between can be set manually.

The gain control that follows adjusts the sample's level from anywhere between *-12.0 dB* and *+12.0 dB*. And at the end of this top row is a vertical cursor icon with arrows pointing inward. When enabled, edits done on the waveform editor will snap to zero crossings.

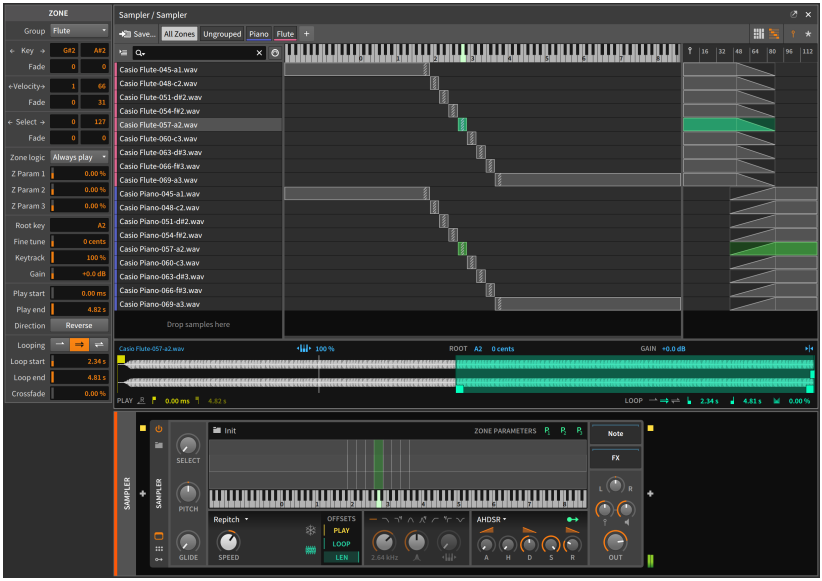
The bottom row starts with *PLAY* controls. The left-facing arrow button capped with an *R* enables reverse mode, causing the sample to play backwards, effectively swapping the play start and play end times (and the loop points as well, when in use). Next are the aforementioned play start and play end times, both set in time units.

Four *LOOP* parameters come next. First are buttons to select which loop mode is in use. The three choices are the single forward arrow (no looping), the stacked forward arrows (single-direction looping), and the stacked forward and reverse arrows (ping-pong looping). Next come the loop start and loop end times, set as exact times just like their *PLAY* counterparts. Finally, a loop crossfade amount is set as a percentage of the available audio material.

When working in a multisample context, two editors are available. Either can be seen in the **Expanded Device View** (see section 8.1.4). We will first look at what is unique about each editor and then cover the vast amount that they have in common.



Shown above is the *grid editor*. This display shows an overlapping view of all current *zones*. These individual sample zones are visualized from low to high note pitches horizontally, and they are shown vertically either by their velocity range (the standard note strike, "pin" icon on the top right) or by their select parameter range (the star icon). Within each zone is the sample's name along with a small triangle pointing down to the root key for that zone. Each zone can be moved by clicking its center and dragging, or each edge of a zone can be adjusted by clicking on that edge and dragging. While the central display stays put, all zones are in a vertically scrollable list on the left side.



In this image is the *list editor*. This view still uses a list of zones on the left side, but only shows those zones that are currently on-screen. The purpose of this view is to display the full details of each zone, including its key range (and root note, shown as a gray, diagonally striped rectangle) in the middle, and either its velocity or select parameter range on the right. All three of these ranges also visualize any crossfading used, allowing gradual transitions at the ends of each range. These ranges can also be interacted with, either by dragging to move an entire range, dragging from an edge to adjust the start or end points, or [ALT]-dragging to add a crossfade to any range.

The editors are identical in many other ways, starting with the top row of either interface.

Starting at the far left is a button to *Save...* the current multisample into the library. After that are filters for viewing either *All Zones* or only the *Ungrouped* zones. Next come any *groups* that have been created within this multisample, either by clicking the *+* icon at the end of this row, or by selecting multiple zones and choosing the *Group* function. In the examples above, groups called *Piano* and *Flute* are present. Clicking any group name will display only its zones and automatically select them all, allowing you to act upon the zones in tandem from the **Inspector Panel**.

At the right edge of this row are two pairs of toggles. This first pair of buttons is for switching between the grid editor and the list editor, and

the second pair chooses whether velocity or the select parameter is being displayed as a secondary axis.

The list of zones on the left displays each zone's group color (in case it is part of a group) and the name of the sample it uses. The top row above the zones provides various filters for how and what to display, including a drop-down menu of sorting options, a search field for filtering zones by part of their sample name, and a toggle button showing a five-pin MIDI port that automatically selects the zone matching the last incoming note message.

The waveform editor will appear at the bottom of the **Expanded Device View** when only one zone is selected. All of the same parameters appear here as when in single sample mode. All of those details and more will also be present in the **Inspector Panel**, even when multiple zones are selected so they can be edited together. Inspector options include:

› *Group* that the zone is a part of, if any.

› *Key* shows the lowest and highest notes that will trigger this zone. Beneath the two note fields are corresponding *Fade* amount fields, showing the length of crossfade (in number of notes) on that side of the range.

› *Velocity* shows the lowest and highest velocities that will trigger this zone. Beneath the two velocity fields are corresponding *Fade* amount fields, showing the length of crossfade (in velocity units) on that side of the range.

› *Select* shows the lowest and highest select parameter values that will trigger this zone. Beneath the two select value fields are corresponding *Fade* amount fields, showing the length of crossfade (in select parameter units) on that side of the range.

› *Zone logic* determines when this zone should sound, particularly when one note is triggering multiple zones. Options include *Always play* (which will always play this zone when triggered) and *Round-robin* (which will only play one matching zone in round-robin mode at a time).

› *Z Param 1*, *Z Param 2*, and *Z Param 3* are amounts of modulation applied via the P_1, P_2, and P_3 modulation sources when this zone is triggered. Those sources appear in the regular **Device Panel** interface of **Sampler** when it is in multisample mode. These modulations are polyphonic, allowing you to offset any parameter by a set amount when a particular zone is triggered.

› *Root key* for this zone, or the note at which no transposition is applied.

› *Fine tune* amount for the root key value.

› *Keytrack* amount as a percentage.

› *Gain* amount applied to the sample.

› *Play start* and *Play end* times, same as the yellow markers in the waveform view, as well as the *Reverse Direction* option, which effectively swaps the start and end times.

› *Looping*, *Loop start*, *Loop end*, and *Crossfade* amounts, all of which correspond to the *FADE* parameters discussed earlier.

All other sections and controls of **Sampler** are the same, regardless of the number of samples being used.

The leftmost section in the device interface contains three parameters. The *Select* parameter decides which multisample is triggered (when select parameter ranges have been defined). *Pitch* can be shifted in semitones, with a range of three octaves in either direction (from *-36.00* to *+36.00*). And *Glide* sets the amount of time that it takes for each new note to smoothly transition from the previous pitch to the current one. In musical terminology, this effect is called *portamento*.

Next is the *play mode* section. It starts with a drop-down menu of the various mode options, which will in turn determine which parameters are available below. Modes include:

› *Repitch* - traditional sampler mode, where the *Speed* parameter changes both playback speed and pitch.

› *Cycles* - a wavetable playback mode that captures periods of the waveform for playback. *Speed* doesn't affect pitch, and broad *Formant* shifting is available as a timbre control. (While this mode excels at reshaping waveforms for pitched playback, disabling keyboard tracking for pitch can produce interesting metallic sounds when different notes are played.)

Note

When a WAV file with a "clm" chunk is imported into **Sampler**, the file will recognized it as wavetable audio, the play mode will be set to *Cycles*, and the *Root key* will use the appropriate value (which determines the size of the wavetables in use).

› *Textures* - a granular playback mode. *Speed* doesn't affect pitch, *Grain* size can be controlled, and randomized *Motion* can be added to the playhead for a less static sound.

Two toggle buttons round out the play mode section. The snowflake icon freezes the sampler's playhead. This gives control of the playhead to you (and any modulators you assign) via the playhead *POS*(ition) control in the following section. Finally, the *RAM* chip icon toggles whether the sample(s) used by this instance of **Sampler** are loaded into memory or not. There is a trade-off, of course: loading samples into RAM consumes memory, but it also allows play and loop points to be modulated.

The following *OFFSETS* section contains playback modulation controls. *PLAY* allows you to modulate the sample start time as a percentage. (When the playhead is frozen, this parameter is renamed playhead *POS*[ition].) *LOOP* modulates the entire loop region's relative position, and *LEN*(gth) modulates the length of the loop to be proportionally shorter. While sample/zone parameters cannot be controlled by modulators, these performance controls can be.

The next section is for the instrument's filter module. Across the top is a row of filter mode options shown with icons (and numeric pole counts to indicate the mode's filter slope, where appropriate). Also included are controls for the filter's cutoff, the amount of resonance being applied (with a peak-shaped icon), and the amount of keyboard tracking (with a keyboard icon bookended by outward facing arrows) that is applied to the cutoff frequency, set relatively from *0 %* to *200 %*.

Then comes the instrument's amplitude envelope section. This envelope generator module affects the entire instrument's audio output level. The envelope generator is switchable between two types of envelopes:

› When the drop-down menu at top is set to *AHDSR*, standard *A*ttack, *D*ecay, *S*ustain, and *R*elease controls are available. Also available is a *H*old control, which sets the time that the envelope pauses at full strength after the completion of the attack segment and before the decay segment begins. Additionally, the timed segments (attack, decay, and release) each have a shape control embedded in their label for changing their curve.

Note

When in *AHDSR* mode, the envelope signal can also be routed to additional modulation destinations via the modulation routing button in the top right of this section. This modulation signal is only generated when the instrument is set to use the *AHDSR* envelope.

› When the mode menu at top is set to *Shot*, the envelope generator acts in a simple one-shot mode with controls for fade in and fade out time. This also disables looping.

Note

Shot mode is incompatible with playhead freeze. When both *Shot* and playhead freeze are enabled, this section will show a red snowflake button. When clicked, freeze mode will be disabled.

The final parameter section contains the nested device chains along with four knobs. Controls for panning (labeled *L* and *R* at the extremes), velocity sensitivity (the standard note strike, "pin" icon), gain (a speaker icon), and the *Out*(put) level are all available.

Modulation Sources:

› *Amplitude EG* (amplitude envelope generator) [polyphonic] - The signal of this instrument's amplitude envelope generator module when it is in *AHDSR* mode. (The routing of this module to the instrument's amplitude is hardwired.)

› P_1 (Zone Parameter 1; only active in multisample mode) [polyphonic] - A modulation whose amount is set by each zone and then applied while the zone is triggered.

› P_2 (Zone Parameter 2; only active in multisample mode) [polyphonic] - A modulation whose amount is set by each zone and then applied while the zone is triggered.

› P_3 (Zone Parameter 3; only active in multisample mode) [polyphonic] - A modulation whose amount is set by each zone and then applied while the zone is triggered.

Nested Device Chains:

› *Note* - A chain for processing incoming note messages before they reach this device.

› *Release* - A chain that receives a note on (of set *Note Length*) when the **Sampler** receives a note off message. The velocity of this trigger can either come from the original note *On* message, or the triggering note *Off*.

› *FX* - A chain for processing the device's entire audio output.

19.24. The Grid

Each Grid device uses **The Grid** for constructing patches (see chapter 17).

19.24.1. FX Grid

A unique modular audio effect, including the option of **Voice Stacking**, and the ability to create polyphonic effects with voice management (see section 17.3.4.1).

19.24.2. Note Grid

A unique modular note processor or generator, including the option of **Voice Stacking**, and the ability to create polyphonic effects with voice management (see section 17.3.4.2).

19.24.3. Poly Grid

For creating instruments, be them mono-/poly-synths, samplers, sequenced patches, cascading drones, or anything else you might imagine.

19.25. Tom

Tom drum element instruments that use incoming note signals to synthesize audio.

19.25.1. E-Tom

An electronic tom instrument with optional pitch modulation.

The *GEN* section contains parameters for controlling and processing the instrument's slightly rectified sine oscillator. The frequency of

this oscillator is set by the *Tune* knob, and its level is controlled by an AD envelope that has a short, fixed attack time and an exponential, adjustable *Decay* time. The *Click* option adds impact to the sound by doubling portions of it, and the *Tone* control sets the cutoff frequency of a gentle low-pass filter.

The *PEG* section concerns a separate AD envelope generator that controls the oscillator's pitch. You can adjust the *Decay* time, the shape of that decay segment with the contour control, and the *Amount* of modulation in semitones.

The final section offers a control for the instrument's *Vel Sens.*(itivity) and a level control for its *Output*.

Nested Device Chains:

› *FX* - A chain for processing the device's entire audio output.

19.26. Utility

Each *utility* device sports various, basic functionality.

19.26.1. DC Offset

A device to add DC offset to the incoming signal. (Yes, add.)

19.26.2. Dual Pan

A device for setting individual panning levels for the incoming left and right channels.

19.26.3. Test Tone

A generator that outputs a waveshape or noise. No input is necessary. Shapes include:

› *Sine* - a single harmonic, respecting the *Frequency* and *Bipolar* settings

› *Triangle* - odd harmonics at inverse squared proportions (with alternating polarities), respecting the *Frequency* and *Bipolar* settings

› *Square* - odd harmonics at inverse proportions, respecting the *Frequency* and *Bipolar* settings

› *Saw Up* - ascending ramp with all harmonics at inverse proportions, respecting the *Frequency* and *Bipolar* settings

› *Saw Down* - descending ramp with all harmonics at inverse proportions, respecting the *Frequency* and *Bipolar* settings

› *Dirac* - A series of one-sample impulses, respecting the *Frequency*

› *White Noise* - Uniform random distribution, producing audio with equal power per frequency

› *Pink Noise* - A 1/f ("one over f") distribution, producing audio with equal power per octave

All modes respect the *Gain* setting for level and dry/wet *Mix* value.

19.26.4. Time Shift

A device for moving incoming audio and/or MIDI signals either forward or backward in time. Whether set to operate in milliseconds (*ms*) or *samples*, positive values represent delay times, and negative values represent times shifted to happen earlier.

19.26.5. Tool

A utility tool for signals that includes amplitude, volume, panning, and width controls as well as channel invert switches and high-resolution output level meters.

19.27. Modulators

Each *modulator* is a special-purpose module that can be added to any Bitwig device or plug-in. The modules output is then assigned to control various parameters of the device.

As in Bitwig Studio, the modulators are categorized below by the type of function they perform. For more information on using modulators, see section 16.2.1.

19.27.1. Audio-driven Category

Devices that convert audio into a modulator signal

19.27.1.1. Audio Rate

An instantaneous (read: non-averaged) sidechain control, routable from any audio signal within the current project. Gain control, an optional low-pass filter with adjustable cutoff frequency, and a *Rectify* switch (to convert the incoming signal to all positive values) are all available.

19.27.1.2. Audio Sidechain

An averaged sidechain control, routable from any audio signal within the current project. Analysis of the incoming signal uses adjustable gain, switchable averaging modes, high- and low-pass filters, and *Attack* and *Release* times.

19.27.1.3. Envelope Follower

A sidechain control that uses the device's incoming audio signal. Analysis of the incoming signal provides adjustable gain, switchable averaging modes, and *Attack* and *Release* times.

19.27.1.4. HW CV In

A sidechain control for control voltage devices that are connected to your audio interface's inputs. Parameters include *Gain*, *Smooth*(ing), and a toggle between alternating current (*AC*) and direct current (*DC*) modes.

19.27.2. Envelope Category

Periodic generators triggered by note ons or offs.

19.27.2.1. ADSR

A standard envelope generator with attack, decay, sustain, and release segments. There is also a *Single Trigger* option, and an option to *Pre-delay* the envelope start in musical or real time.

19.27.2.2. AHD on Release

An attack–hold–decay envelope generator triggered by note off messages, with *Single Trigger* option. The timed segments also have individual curve controls.

19.27.2.3. AHDSR

A standard envelope generator with attack, hold, decay, sustain, and release segments. The timed segments also have individual curve controls. There is also a *Single Trigger* option, and an option to *Pre-delay* the envelope start in musical or real time.

19.27.2.4. Note Sidechain

A standard envelope generator with attack, decay, sustain, and release segments. The gate message driving the envelope generator is routable from any note message source within the current project. There is also an option to *Pre-delay* the envelope start, in musical or real time.

19.27.2.5. Ramp

A simple ramp generator with switchable direction, curve, and optional looping.

19.27.2.6. Segments

A freely drawable, segmented envelope generator, with BWCURVE file support, making use of the **Curve Editor** (see section 16.2.1.1)

› With all familiar envelope modulator controls

› Four *Play Mode* options are available:

One-shot (⇀) - Plays thru the entire shape (while the voice is alive) at note on

Hold (◠) - Uses any one of the curve's points as the hold/sustain level, which is also the release start

Looping (⇛) - Uses any two of the curve's points, and loops forwards between them on sustain; the loop end point is also the release start

Ping Pong (⇌) - Uses any two of the curve's points, and loops forwards and backwards between them on sustain; the loop end point is also the release start

The hold point or loop region & start/end points are shown in blue

You can either drag one of these points to another point, or right-click on any point and choose the appropriate option — either *Set Hold Point* (when in *Hold* mode), or *Set Loop Start / Set Loop End* (when in *Looping* or *Ping Pong* modes)

› A set *Rate* (from *0.2* to *50*) with regard to the set *Timebase* (either *Minutes*, *Seconds*, *Milliseconds*, *Bars* or other beat-time units, *Pitch (of current note)*, or *Hold*)

Both the *Rate* and *Timebase* can be modulated for each note, for example by *Velocity* (from the **Expressions** modulator), or any other source

This *Rate–Timebase* pair defines the *primary interval* of the whole envelope, which defaults to a setting of *1 bar*, and with a shape that ends after one iteration

The **Curve Editor** is scrollable and shows a time ruler in the primary interval (*1*, *2*… n), with the set number of grid units displaying within each primary interval

Clicking and dragging in the ruler area allows for zooming and scrolling, just as with the Arranger

Points can be added or dragged to extend the length of the envelope, so taking the default settings and adding a point at the *4* line would extend the shape to be 4 bars long

› Option to *Enabling Smoothing*, with *Smoothing Time* set in milliseconds/seconds

Both settings can be automated and modulated, for controlling the sharpness vs. smoothness of each voice, for example by *Poly Pressure* (from the **Expressions** modulator), or any other source

On the **Polymer** module version of **Segments**, both parameters are available in a context menu when right-clicking on the module's background

› A *Bipolar* toggle (±) maintains the curve's shape but rescales it, so that the minimum value is *-1* and the middle value is now *0* (zero)

› A *Single Trigger* option to keep the envelope from retriggering when overlapping notes are received

› An *Amount* parameter controlling output scaling of each voice

19.27.3. Interface Category

Providing panel elements for better/unique control, or access to Transport-level interface items.

19.27.3.1. Button

A binary toggle control. Selecting the modulator and viewing the **Inspector Panel** shows an optional smoothing parameter.

19.27.3.2. Buttons

Two independent binary toggle controls. Selecting the modulator and viewing the **Inspector Panel** shows an optional smoothing parameter that applies to both buttons.

19.27.3.3. Globals

Provides modulator signals for three project-wide controls:

› *Fill* - A modulator signal reflecting the current *Fill* mode state (see section 2.3.2)

› *A◆B* - A bipolar modulator signal reflecting the current Global Crossfader value (see section 7.1.9)

› *Play* - A modulator signal reflecting whether the transport is currently playing (*1*) or not (*0*)

The *Fill* and *A◆B* sources can be used as global control sources, routing hardware controllers or automation (from *Master* > *Transport*) to any and all tracks.

19.27.3.4. Macro

One continuous knob control.

19.27.3.5. Macro-4

Four independent, continuous knob controls.

19.27.3.6. Select-4

Four control sources derived from one continuous fader control. The single fader is essentially a crossfader whose position determines which one or two control sources will receive a modulation value.

19.27.3.7. Vector-4

Four control sources derived from one continuous XY control. The single fader is essentially a crossfader whose X and Y positions determine the modulation values received by each control source.

19.27.3.8. Vector-8

Eight control sources derived from one continuous XY control. The single fader is essentially a crossfader whose X and Y positions determine the modulation values received by each control source.

19.27.3.9. XY

Two control sources derived from one continuous XY control. The single fader is essentially a joystick whose X and Y positions are used as the control sources' values.

19.27.4. LFO Category

For regularly repeating patterns or noise.

19.27.4.1. Beat LFO

A tempo-synced (including the option to follow global shuffle) low-frequency oscillator, with shape, phase, and polarity controls.

19.27.4.2. Classic LFO

A tempo-synced low-frequency oscillator, typically used in Bitwig Studio version 1 devices. Provides a *Note Trigger* option and a *Per-Voice* toggle (when applicable).

19.27.4.3. Curves

A freely drawable, segmented LFO, with BWCURVE file support, making use of the **Curve Editor** (see section 16.2.1.1)

› With all familiar LFO modulator controls

› A set *Rate* (from *0.2* to *50*) with regard to the set *Timebase* (either *Hertz*, *Kilohertz*, *Bars* or other beat-time units, *Pitch (of current note)*, or *Hold*)

Both the *Rate* and *Timebase* can be modulated for each note, for example by *Velocity* (from the **Expressions** modulator), or any other source

This *Rate–Timebase* pair defines the speed of the oscillator, which defaults to a setting of *1 Hz*

› *Phase* parameter allows for full control of the envelope's position, small variations, or anything in between

› Option to *Enabling Smoothing*, with *Smoothing Time* set in milliseconds/seconds

Both settings can be automated and modulated, for controlling the sharpness vs. smoothness of each voice, for example by *Poly Pressure* (from the **Expressions** modulator), or any other source

› A *Bipolar* toggle (±) maintains the curve's shape but rescales it, so that the minimum value is *-1* and the middle value is now *0* (zero)

› Five *Trigger Mode* options, similar to other LFOs:

Free - All voices play freely and never reset

Note - Each voice resets to the current *Phase* when it receives a note

Sync - Follows the relative phase (*Phase* + Global transport position) while transport is playing

Grv (Groove) - Follows the groove-relative phase (*Phase* + Global transport position, including groove) while transport is playing

Rnd (Random) - Each voice goes to a random position when it receives a note

› An *Amount* parameter controlling output scaling of each voice

19.27.4.4. LFO

A fully functional low-frequency oscillator, with shape, phase, and polarity controls. It can also be tempo synced, set to fade in, given various reset modes, and be toggled to a polyphonic mode.

19.27.4.5. Random

A tempo-synced random low-frequency oscillator. Output can be discrete or slewed, be unipolar or bipolar, be retriggered by *Note* or *Sync* messages, and be monophonic or polyphonic (when applicable).

19.27.4.6. Vibrato

A musical LFO whose amount can be normalled to *Modwheel* or polyphonic *Pressure* (which will use channel aftertouch if poly pressure isn't present).

19.27.4.7. Wavetable LFO

A morphable LFO, with Bitwig WT file support

19.27.5. Modifier Category

Processor modulators.

19.27.5.1. Math

One control source derived from two continuous knob controls. The output signal is a mathematical relation of the two signals, which is derived either by *MUL*(tiplying), *ADD*(ing), or *SUB*(tracting) the two

signals, or simply taking the *MIN*(imum) or *MAX*(imum) of the two values.

19.27.5.2. Mix

One control source derived from two continuous fader controls. The output signal is determined by the current crossfade position between the two fader values.

19.27.5.3. Polynom

A transfer function for reshaping an incoming modulation signal with a basic polynomial equation. The way to pass a signal into the module for processing is by modulating the *x=* parameter with any other modulator(s).

Each of the four additional parameters represent a term of the cubic function used. x^0 represents the offset applied to the function. x^1 represents the function's slope (or rotation). x^2 represents the parabolic curve applied to the function, and x^3 represents a cubic curve (like an S-curve). The graphical interface helpfully visualizes the transfer function being used, and the processor will clip the output signal to stay within range.

19.27.5.4. Quantize

A processor for reducing the resolution of an incoming modulation signal, often used to transform a fairly continuous signal into one that is more discrete. The way to pass a signal into the module for processing is by modulating the *Input* parameter with any other modulator(s).

The *Quantize* factor sets the resolution of the output signal. A low setting restricts the output to be more like a pulse signal, and a high setting preserves the more continuous elements of the original signal. Additionally, four shape options (*Linear*, *Log*, *Exp*, and *Sinh*) adjust the spacing of the resolution grid used by the device.

19.27.5.5. Sample and Hold

A processor that sustains (or holds) an incoming signal's value at the moment of each clock event. The way to pass a signal into the module

for processing is by modulating the *Input* parameter with any other modulator(s). The *Smooth* parameter represents the transition time between successive samples.

The sampling clock can be set to various metronomic values (such as *4th* for quarter notes, *1/8.* for dotted eighth notes, *bar* for one measure at the project's current tempo, etc.), to free time values (either hertz [*Hz*] or kilohertz [*kHz*]), to the *Pitch* of the latest received note message, or to *Hold*, which keeps the output signal from changing. This base clock rate can then be scaled by the adjacent modulation knob, which at center is *1.00* (100% or no change), at far left is *0.02* (2%), and at far right is *50.00* (5,000% or 50x).

At bottom, this device offers three modes of operation. *Free* allows the sampling clock to run independently, *Gate* restarts the clock whenever a new note message is received, and *Sync* restarts the clock whenever the transport is started.

19.27.6. Note-driven Category

Devices triggered by notes or MIDI.

19.27.6.1. Channel-16

Sixteen control sources, one for each MIDI channel received. With global *Amount* and *Lag* controls, and an option to *Release with Note Offs* or not (for per-voice uses).

19.27.6.2. Expressions

A module to extract incoming *VEL*(ocity), *REL*(ease velocity), *TIMB*(re), and *PRES*(sure) messages. Right-clicking the modulator's title or selecting the modulator itself and viewing the **Inspector Panel** shows parameters for enabling smoothing on all expressions and making the timbre expression relative at the time of each new note. All expressions are polyphonic when applicable.

19.27.6.3. Keytrack+

Uses an embedded **Curve Editor** for drawing, saving, or loading keytracking curves (see section 16.2.1.1)

19.27.6.4. MIDI

A module to extract either continuous controller (*CC*), *PRESSURE*, or pitch *BEND* messages arriving at the device's input.

19.27.6.5. Note Counter

A module whose modulation output is incremented with each new note message received. The number of *Steps* counted and the value *Increment*-ed at each step can be set, as well as the *OUTPUT SCALING* method to be used for the modulation signal.

19.27.6.6. Pitch-12

Twelve control sources, one for each pitch class (C, D, E, etc.) received. With global *Amount* and *Lag* controls.

19.27.6.7. Relative Keytracking

Note pitch modulator, with *Root* note and *Spread* options.

19.27.7. Sequence Category

Providing step-/segment-based modulations.

19.27.7.1. 4-Stage

A looping four-stage envelope generator, with definable times (optionally tempo-synced) and levels (optionally bipolar).

19.27.7.2. ParSeq-8

A special parameter sequencer, with the same global parameters as **Steps** (see section 19.27.7.3). Each step is its own modulation source so

assigned parameters are modulated and then reset when advancing to the next step.

Each step starts with its step number, which can also be clicked to temporarily disable that step's modulations from taking effect. Next is a button with a musical *fermata* icon, which holds any previous modulations when this step begins (instead of resetting them to zero). Finally each step has a bipolar fader for scaling the depth of all modulations on that step.

19.27.7.3. Steps

A tempo-syncable, bipolar step sequencer. Parameters include step count, direction (forward, backward, and/or ping-pong mode that switches direction on each loop), polarity, and phase (⌀) for manual control of the play position. Trigger modes determine when the step sequencer advances:

› *Transport* - Links to the global transport for play–stop status, tempo, and beat position

› *with Groove* - Links to the global transport for play–stop status, tempo, and beat position with groove

› *Free running* - Plays at the set rate, independent of the transport and incoming notes

› *Note / Restart* - Plays at the set rate, with new notes restarting the pattern

› *Note / Random* - Plays at the set rate, with new notes randomizing the position

› *Note / Advance* - Holds the playhead in place, advancing only on incoming new note

Right-clicking on the pattern interface also provides options to copy and paste the pattern, as well as to *Generate* a preset pattern to replace the current one, or to *Transform* the current pattern .

19.27.8. Voice Stacking Category

Devices that work differently on each of the individual voices within an active voice stack. Proper function of these modules requires **Voice Stacking** to be active on the parent device (see section 16.2.5).

19.27.8.1. Stack Spread

Offers 12 spread modes, for varying all voices in a stack with one modulation mapping to any parameter(s). All modes visualize the relative effect on each voice in the **Inspector Panel**.

The first four modes offer simple *splits*, most of which are evenly distributed:

› *0 to 1* spreads voices in a unipolar fashion, from 0% of the set modulation level to 100%.

› *-1 to 1* spreads voices in a bipolar fashion, from -100% of the set modulation level to 100% (just as the built-in *Voice Stack Spread ±* modulator works).

› *Value* spreads voices from 0% in successive increments of 100% for each additional voice, making it easier to set amounts in some cases and to work with enumerated list parameters.

› *Manual* allows you to manually create the distribution of values across voices with faders in the **Inspector Panel** (which can even be automated/modulated themselves).

The middle four modes offer various defined *distributions*, all starting with maximum modulation on voice 1, and then reflecting to smaller and smaller values:

› *Flipped* provides simple reciprocals [1, 1/2, 1/3, ... 1/n].

› *Straight* gives harmonic relationships [1, 1/2, 1/4, ... $1/2^n$], whether for pitch or rhythms.

› *Primes* is the series of prime numbers, inverted [1, 1/2, 1/3, 1/5, 1/7, 1/11...].

› *Golden* provides the Fibonacci sequence [1, 1/2, 1/3, 1/5, 1/8, 1/13...].

Note

If you want to reorient any of these distributions, consider using the modulation transfer functions (see section 16.2.4.3). For example, the *Toward Zero* mode would flip any of the distribution modes, giving the last voice a maximum modulation and the earlier voices getting smaller and smaller.

The bottom four modes offer unique kinds of *randomization*, relating all the voices in any particular stack to each other. In this way, you'll either

get centered values that tend to peak on few voices each time (with the first and third modes), or you'll get more pronounced values that extend to the minimum and maximum each time (with the second and fourth modes):

› *Rand+* creates a unipolar random value for each voice (at note on), with all values adding up to *1*.

› *Rand+* ↕ offers a scaled, stronger version of *Rand+*, tending towards more large values.

› *Rand±* creates a bipolar random value for each voice (at note on), with all values adding up to *0* (zero).

› *Rand±* ↕ offers a scaled, stronger version of *Rand±*, tending towards more large values.

19.27.8.2. Voice Control

Offers individual control of each voice with an active voice stack, with individual modulators for *Stack Voice 1* thru *Stack Voice 16* directly in the modulator square.

19.28. Grid Modules

Each *Grid module* is a building block that can be loaded within any Grid device and interconnected with other modules.

As in Bitwig Studio, the modules are categorized below by the type of function they perform and can be browsed that way. For more information on using **The Grid** and working with Grid devices, see chapter 17. And for full information on the parameters of each module, see the module's help view within Bitwig Studio (see section 17.1.2.1.).

19.28.1. I/O Category

Terminal modules for signals entering or exiting this Grid device

19.28.1.1. Gate In

Supplies note gate signals from the device

19.28.1.2. Phase In

Supplies the device's default phase signal

19.28.1.3. Pitch In

Supplies note pitch signals from the device

19.28.1.4. Velocity In

Supplies note velocity signals from the device

19.28.1.5. Audio In

Supplies audio signals from the device

19.28.1.6. Audio Out

Path to the device's audio output buss. Has an *Output Clipping Mode* option (*Off*, *Hard*, or *Soft*) and an *Output Clipping Level* setting (*0 dB*, *+6 dB*, *+12 dB*, or *+24 dB*) for how to handles overages.

19.28.1.7. Gain In

Supplies note gain expressions

19.28.1.8. Pan In

Supplies note pan expressions

19.28.1.9. Pressure In

Supplies note poly pressure signals from the device

19.28.1.10. Timbre In

Supplies timbre expressions from the device

19.28.1.11. CC In

Supplies select continuous control signals from any/all MIDI channels

19.28.1.12. CC Out

Outputs continuous control signals on any MIDI channel

19.28.1.13. Note In

Provides gate, expressions, and channel of every incoming note. Its eight out ports match the **Note Out** configuration (including the *Enable All Expressions* [...] toggle for unfolding and showing all ports), for easy processor patching.

19.28.1.14. Note Out

Creates output notes, with all expressions available via eight in ports.

› *Gate In* port triggers a note to be created

› *Pitch In*, *Velocity In*, and *Channel In* can either be set with fixed values on the module's face, or provided with signals

Note

The *Pitch In* port requires an input signal between note C-2 (*-0.5*) and G8 (*+0.558*).

The *Velocity In* port requires an input signal above zero.

Only when these conditions are met will a high-logic signal at the *Gate In* port create a new note on.

› When *Enable All Expressions* (... toggle) is on, *Timbre In*, *Pressure In*, *Gain In*, and *Pan In* ports are available for signal control of all note expressions

› When *Enable All Expressions* (… toggle) is off, connections to these additional expression ports are remembered but inactive

› As with any module, multiple **Note Out** modules can be loaded, helpful for sequencer or "groovebox" style patches, or whenever you want to group notes onto different MIDI channels, etc. etc.

19.28.1.15. Audio Sidechain

Supplies audio signals from a selected project path

19.28.1.16. HW In

Supplies audio signals from a selected external path

19.28.1.17. HW Out

Path to a selected external audio output buss

19.28.1.18. CV In

Supplies control voltage (CV) signals from a selected external path

19.28.1.19. CV Out

Path to a selected external CV output buss

19.28.1.20. CV Pitch Out

Path to a selected external CV output pitch buss

19.28.1.21. Key On

Supplies note gate signals from a specified note and channel

19.28.1.22. Keys Held

Number of keys currently held

19.28.1.23. Transport Playing

Supplies the application's playback status

19.28.1.24. Voice Stack Info

Supplies current voice stack index (a polyphonic signal) and the voice stack size

19.28.1.25. Modulator Out

Makes incoming signals available as a modulator signal

19.28.2. Display Category

Visualization and note-taking modules

19.28.2.1. Label

Large text widget

19.28.2.2. Comment

Smaller text widget

19.28.2.3. Oscilloscope

Dual trace oscilloscope, with thru ports and controls for the *Y Maximum* level, whether to paint *Y Bipolar* or not (for unipolar), and whether the *Voice Shown* should be only the *Last voice* played or a sum of *All voices*.

19.28.2.4. Spectrum

Spectrogram for up to four signals

19.28.2.5. VU Meter

Averaging meter

19.28.2.6. XY

Two-dimensional control pad

19.28.2.7. Value Readout

Stereo numeric readout for various domains

19.28.3. Phase Category

Modules that output wrapped phase signals

19.28.3.1. Phasor

Phase signal generator with typical oscillator controls

19.28.3.2. Ø Bend

Imposes a variable curve onto a phase signal

19.28.3.3. Ø Pinch

Imposes an S-curve onto a phase signal

19.28.3.4. Ø Reset

Offsets the incoming phase signal to *0* each time a trigger is received

19.28.3.5. Ø Scaler

Scales an incoming phase signal to be proportionally faster or slower

19.28.3.6. Ø Reverse

Inverts a proper phase signal

19.28.3.7. Ø Wrap

Wraps any signal into the phase signal range

19.28.3.8. Pitch → Ø

Wraps pitch signal's octave as phase signal

19.28.3.9. Ø Counter

Translates successive trigger signals into discrete phase values

19.28.3.10. Ø Formant

Amplifies the incoming signal around *+0.5*

19.28.3.11. Ø Lag

Lag processor that stays within the phase range

19.28.3.12. Ø Mirror

Applies gain to the incoming phase signal and then reflects it

19.28.3.13. Ø Shift

Offsets the incoming phase signal by a set amount

19.28.3.14. Ø Sinemod

Modulates the incoming phase signal with a sine wave

19.28.3.15. Ø Skew

Sets the incoming level to remap to *+0.5*

19.28.3.16. Ø Sync

Amplifies the incoming phase signal before wrapping it

19.28.3.17. Ø Split

Equally distributes phase signal across up to 8 out ports

19.28.4. Data Category

Lookup modules that are read with incoming phase signals

19.28.4.1. Gates

Event sequencer

19.28.4.2. Pitches

Mono pitch sequencer

19.28.4.3. Slopes

A freely drawable, segmented sequencer, with BWCURVE file support, making use of the **Curve Editor** (see section 16.2.1.1)

› With all familiar data sequencer module controls, and their common phase-driven approach

› A stereo *Phase In* port for controlling playback, along with a *Use Device Phase* pre-cord

› A *Bipolar* toggle (±) maintains the curve's shape but rescales it, so that the minimum value is *-1* and the middle value is now *0* (zero)

› *Anti-aliasing* toggle enables smoothed playback of the curve

› *Mute when stopped* option will force *0* (zero) output when the global transport isn't running

19.28.4.4. Steps

Step sequencer

19.28.4.5. Triggers

Generates N triggers evenly across each cycle

19.28.4.6. Probabilities

Probabilistic event sequencer

19.28.4.7. Ø Pulse

Pulse lookup module

19.28.4.8. Ø Saw

Saw lookup module

19.28.4.9. Ø Sine

Sine lookup module

19.28.4.10. Ø Triangle

Triangle lookup module

19.28.4.11. Ø Window

Cosine window module

19.28.4.12. Array

Recordable lookup table

19.28.5. Oscillator Category

Periodic signal generators based on waveforms or samples

19.28.5.1. Pulse

Geometric pulse oscillator

19.28.5.2. Sawtooth

Geometric sawtooth oscillator

19.28.5.3. Sine

Sine wave oscillator

19.28.5.4. Triangle

Geometric triangle oscillator

19.28.5.5. Union

A DC-drifting, analog-inspired oscillator that blends pulse, saw, and triangle waves. Each of these three waves has its own *Level* control, or each waveform visual can be clicked to turn that wave all the way

up (*100 %*) and to set the other two waves to zero. *Pulse Width* can be controlled directly by dragging the slider within the overview display.

19.28.5.6. Wavetable

Wavetable oscillator, with special unison modes & processing options

19.28.5.7. Sub

Sub oscillator, with six *Waveform* options and an *Octave* offset

19.28.5.8. Bite

A *Techniques*-driven oscillator, offering exponential FM, hard sync, PWM, and ring mod from dual oscillator feedback

› Anti-aliasing and internal feedback allow for some very crisp, fresh, and/or modular analog sounds

› Internal Oscillator A & B are identical, each providing seven waveshapes with *Pulse Width* controls, as well as fixed *Saw* and *Sine* options

  Like the **Union** oscillator, the oscillators exhibit some smooth analog drift when *Pulse Width*, for example, is moved

› Oscillator B can pulse-width modulate (*PWM*) Oscillator A

› Oscillator A can do exponential frequency modulation (*xFM*) on Oscillator B

› Oscillator A can also hard *SYNC* Oscillator B:

  SYNC is a useful way to use exponential FM without breaking the pitch of Oscillator B

  Oscillator B also has its own *Pitch Offset* control, for setting (or modulating) more interesting hard sync waveshapes

› A trio of mix controls set the output level for oscillator *A*, oscillator *B*, and a ring-modulated mix (*RM*) of the two

› The Grid module version has a special *Independent Mono Mode* toggle in the Inspector Panel

  This flattens the module to a mono output

This also allows individual oscillator targeting via the in ports, routing left channel inputs only to Oscillator A and right channel inputs to Oscillator B

19.28.5.9. Phase-1

Phase distortion oscillator

19.28.5.10. Scrawl

A freely drawable, segmented oscillator, with BWCURVE file support, making use of the **Curve Editor** (see section 16.2.1.1)

› With anti-aliasing on the oscillator, to help make (almost) any shape work

› With all familiar oscillator module controls

› All the standard module pitch controls:

Key Tracking pre-cord, automatically connecting note pitches to the pitch buss

Numerator and *Denominator* controls, for controlling pitch via ratio

A *Pitch Offset*, for adjustment in semitones

A *Detune* control, for adjustment in Hertz, along with the *Stereo Detune* toggle for an inverse detuning of the right channel

› A *Retrigger on Notes* pre-cord, for reseting the oscillator's phase at note on

› The **Polymer** module version of **Scrawl** also has:

Phase Modulation Amount attenuator (range *0 %* to *800 %*) to allow modulation from the Sub

↑*SYNC*↑ toggle to enable hard sync from the Sub

› The Grid module version of **Scrawl** also has standard oscillator module options:

A stereo *Retrigger In* port

A stereo *Phase In* port, with attenuator (range *0 %* to *800 %*)

A stereo *Pitch In* port and input attenuator

19.28.5.11. Swarm

Unison oscillator

19.28.5.12. Sampler

Module version of the **Sampler** device (see section 19.23.5).

19.28.6. Random Category

Aperiodic and randomized signal generators

19.28.6.1. Noise

White/pink noise generator

19.28.6.2. S/H LFO

Free/beat-synced random oscillator

19.28.6.3. Chance

Weighted random logic signal generator

19.28.6.4. Dice

Uniform random value generator

19.28.7. LFO Category

Periodic low frequency oscillators

19.28.7.1. LFO

Free/beat-synced geometric oscillator

19.28.7.2. Curves

A freely drawable, segmented LFO, with BWCURVE file support, making use of the **Curve Editor** (see section 16.2.1.1)

› With all familiar LFO modulator/module controls

› A set *Rate* (from *0.2* to *50*) with regard to the set *Timebase* (either *Hertz*, *Kilohertz*, *Bars* or other beat-time units, or *Hold*)

  Both the *Rate* and *Timebase* can be modulated for each note, for example by *Velocity* (from the **Expressions** modulator), or any other source

  This *Rate–Timebase* pair defines the speed of the oscillator, which defaults to a setting of *1 Hz*

› *Phase* parameter allows for full control of the envelope's position, small variations, or anything in between

› Option to *Enabling Smoothing*, with *Smoothing Time* set in milliseconds/seconds

  Both settings can be automated and modulated, for controlling the sharpness vs. smoothness of each voice, for example by *Poly Pressure* (from the **Expressions** modulator), or any other source

› A *Bipolar* toggle (*±*) maintains the curve's shape but rescales it, so that the minimum value is *-1* and the middle value is now *0* (zero)

› A stereo *Retrigger In* port, as well as a pre-cord to *Retrigger on Notes*

› A stereo *Phase In* port and input attenuator

› A stereo *Rate In* port and input attenuator

› A *Phase Offset (R)* control, for adjusting the right channel's phase in relation to the general *Phase* value

19.28.7.3. Wavetable LFO

A morphable LFO, with Bitwig WT file support

19.28.7.4. Clock

Phase-signal generator set in Hertz

19.28.7.5. Transport

Synced phase-signal generator

19.28.8. Envelope Category

Modules that produce or extract an envelope, often with a normalled amplifier

19.28.8.1. ADSR

Four-stage gated envelope generator with amplifier. Three *Model* options are available, shown by a clickable letter icon in the top left of the module (*A*, *R*, or *D*):

› *Analog* - Emulating Moog-style fixed curves and nonlinearities

› *Relative* - With adjustable rate-differential curves

› *Digital* - Clean math with adjustable curves, for precise time segments

ADSR has the common *Gate In* port (for controlling the envelope), the *Envelope Out* port (for the created envelope signal), and the *Signal In* and *Out* ports (for attenuating any incoming signal via the envelope).

Additionally, **ADSR** also has a special *Bias Out* port. This port outputs an offset version of the envelope signal that centers around zero in the sustain segment. So if the *Sustain* level is set to *35.0 %*, the *Bias Out* signal will go from *-0.35* to *+0.65* in the attack segment, then coming down to *0* (zero) in the decay segment. After holding at zero for the sustain segment, the release will go from zero back down to *-0.35*. This could be used for a pitch effect that stabilizes in the sustain segment, or anything else you want to try.

19.28.8.2. AD

Two-stage triggered envelope generator with amplifier, looping mode, and three *Model* options (see section 19.28.8.1)

19.28.8.3. AR

Three-stage gated envelope generator with amplifier and three *Model* options (see section 19.28.8.1)

19.28.8.4. Pluck

Plucked string-style envelope generator with amplifier

19.28.8.5. Segments

A freely drawable, segmented envelope generator, with BWCURVE file support, making use of the **Curve Editor** (see section 16.2.1.1)

› A freely drawable, segmented envelope generator

› With all familiar envelope module controls

› Four *Play Mode* options are available:

One-shot (→) - Plays thru the entire shape (while the voice is alive) at note on

Hold (◠) - Uses any one of the curve's points as the hold/sustain level, which is also the release start

Looping (⇒) - Uses any two of the curve's points, and loops forwards between them on sustain; the loop end point is also the release start

Ping Pong (⇌) - Uses any two of the curve's points, and loops forwards and backwards between them on sustain; the loop end point is also the release start

The hold point or loop region & start/end points are shown in the inverse color of the interface

You can either drag one of these points to another point, or right-click on any point and choose the appropriate option — either *Set Hold Point* (when in *Hold* mode), or *Set Loop Start / Set Loop End* (when in *Looping* or *Ping Pong* modes)

› A set *Rate* (from *0.2* to *50*) with regard to the set *Timebase* (either *Minutes*, *Seconds*, *Milliseconds*, *Bars* or other beat-time units, or *Hold*)

Both the *Rate* and *Timebase* can be modulated for each note, for example by *Velocity* (from the **Expressions** modulator), or any other source

This *Rate* –*Timebase* pair defines the *primary interval* of the whole envelope, which defaults to a setting of *1 bar*, and with a shape that ends after one iteration

The **Curve Editor** is scrollable and shows a time ruler in the primary interval (*1*, *2*... n), with the set number of grid units displaying within each primary interval

Clicking and dragging in the ruler area allows for zooming and scrolling, just as with the Arranger

Points can be added or dragged to extend the length of the envelope, so taking the default settings and adding a point at the *4* line would extend the shape to be 4 bars long

› Option to *Enabling Smoothing*, with *Smoothing Time* set in milliseconds/seconds

Both settings can be automated and modulated, for controlling the sharpness vs. smoothness of each voice, for example by *Poly Pressure* (from the **Expressions** modulator), or any other source

On the **Polymer** module version of **Segments**, both parameters are available in a context menu when right-clicking on the module's background

› A *Bipolar* toggle (±) maintains the curve's shape but rescales it, so that the minimum value is *-1* and the middle value is now *0* (zero)

› A stereo *Gate In* port, as well as a pre-cord to *Gate on Notes*

› A stereo in and out port for an internal amplifier, under the control of the envelope signal

› An *Affect voice lifetime* toggle to allow the module to keep voices active

19.28.8.6. Follower-RF

Envelope extractor with segment times

19.28.8.7. Slope ↗

Slope shaper for rising signals

19.28.8.8. Slope ↘

Slope shaper for falling signals

19.28.8.9. Follower

Symmetric envelope extractor

19.28.9. Filter Category

Frequency-dependent amplifiers

19.28.9.1. Low-pass LD

Resonant low-pass ladder filter

19.28.9.2. Low-pass MG

A Moog-inspired low-pass filter, with mix buss saturation via the *Drive* control

19.28.9.3. Sallen-Key

Resonant Sallen-Key filter, with 16 different models of either low-pass, high-pass, or band-pass configurations, with various slopes

19.28.9.4. SVF

Highly resonant multimode filter

19.28.9.5. XP

Friend of **Ladder** device and inspired by Mr Oberheim, with 15 filter configurations

19.28.9.6. Comb

Comb filter with *Feedback* control and *Dampening Frequency* (which is relative to the module's *Cutoff Frequency*)

19.28.9.7. Vowels

An *Inspired* filter that produces vowel sounds

› Can be used several ways, including:

Setting (or hard modulating/automating) a single vowel

Setting and morphing anywhere between two and five vowels

Any combination, all with different configurations and vowel models

› Standard filter controls include:

Drive to affect the incoming signal level

A *Resonance Limit* Inspector Panel (or *Q Limit* via right-click context menu) control, for adjusting when the filter model saturates

› Five *Vowel Position* choosers are available, located around a central, bipolar *Vowel Blend* knob:

Vowel Blend at *-100 %* uses only the nearby *Vowel Position 1*

Vowel Blend at *0 %* uses only the nearby *Vowel Position 3*

Vowel Blend at *+100 %* uses only the nearby *Vowel Position 5*

Vowel Blend corresponds to *Vowel Position 2* at *-50 %* and *Vowel Position 4* at *+50 %*; if set to vowel sounds, only those values will be heard; if set to *None* (the default), the surrounding vowels will be blended perfectly there

Each position offers 27 different vowel sounds to choose from:

i - As in “see” or “eat”

y - With a rounded w-, like “we”

ɪ - As in “sit” or “hit”

ʏ - A medium oo, like “ooze”

ɨ - An exaggerated oo, like “eww” (gross)

ʉ - A slow oo, like "ooh!" (surprise)

ɯ - As in “hook” or “book”

u - As in “pool” or “cool”

e - As in “say” or “rain”

ø - With a closed -l, like “ool”

ɘ - Partly closed, as in “eh”

ɵ - As in “foot” or “would”

ɣ - Partly closed, as in “uh”

o - First sound in “coat” or “bold”

ə - As in “run” or “ton”

ɛ - As in “get” or “rent”

œ - With a round -l, like “ole”

ɜ - Partly closed, as in “ah”

ɞ - Partly closed, as in “aw”

ʌ - As in “fun” or “come”

ɔ - As in “more” or “floor”

æ - As in “cat” or “hat”

ɐ - With a subtle -r, like “are”

a - First sound in “hi” or “fight”

Œ - With an open -l, like “all”

ɑ - As in “far” or “star”

ɒ - As in “want” or “job”

Each *Vowel Position* can be set in two ways:

Clicking on any position opens a pop-up menu of all available vowels sounds and description texts

Clicking and dragging on any position starts moving thru the vowels sounds, for a quick workflow with audible results (if audio is passing)

In The Grid, a stereo in port (*Vowel In*) is available for adding to the *Vowel Blend* value

› *Profile* selects which set of vowel data to use, with choices including:

Women 1 - an older data set, with average values from several women

Women 2 - a modern data set, with average values from several women

Female - values from one individual female

Men 1 - an older data set, with average values from several men

Men 2 - a modern data set, with average values from several men

Male - values from one individual male

Kids - average values from several children

› The *Topology* chooser (on the right edge of the module) sets the structure of the filter, from three choices:

Cascade - Serial low-pass filters; a bit better for traditional text-to-speech sounds

LP/BP - Low-pass and band-pass filters, processed in parallel; a synthier, Autobahn-friendly vibe

LP/BP/HP - Low-pass, band-pass, and high-pass filters in parallel; adds a slight bit more highs

› Several parameters influence the tuning of the internal filters in use:

Cutoff Pitch Offset moves the internal filters by semitones

The *Cutoff In* port and its associated *Cutoff Modulation Amount* attenuator allows stereo manipulation of the *Cutoff Pitch Offset*

Note: While this is like moving the cutoff of a normal filter up and down, the result is different and you might want to start by disabling pitch modulation

The *Cutoff Frequency Offset* (in the Inspector Panel or via right-click context menu) allowing linear frequency manipulation, which can be interesting for formants

Resonance adjusts the relative sharpness of the internal filters

19.28.9.8. Fizz

A modern *Character* filter for spreading harmonic nodes around

› Has a bit of a moving formant sensibility

› Takes place inside a stereo, resonant low-pass filter, with standard options:

Drive to affect the incoming signal level

Main Cutoff Frequency control

Stereo input for cutoff modulation, with *Cutoff Modulation Amount* set in semitones

Key Tracking Amount, for using incoming note pitches to affect the cutoff buss

› For this algorithm, additional controls include:

Feedback Gain, which feeds or chokes the nested filter

Feedback Cutoff Frequency, for tuning the nested filter

A bipolar *Color* control, which shifts the placement and variation of formant peaks

An *Alternate Color* toggle, for a reorienting and different tuning of the circuit

19.28.9.9. Rasp

A modern *Character* filter that can scream or whisper

› Creates resonant peaks on top of the standard filter

› Takes place inside a stereo, resonant filter, with standard options:

Drive to affect the incoming signal level

A *Filter Type* setting, to switch between an outer *Low-pass* filter or a *Band-pass* model

Cutoff/Center Frequency control

Stereo input for cutoff modulation, with *Cutoff Modulation Amount* set in semitones

Key Tracking Amount, for using incoming note pitches to affect the cutoff buss

A *Resonance Limit* Inspector Panel (or *Q Limit* via right-click context menu) control, for adjusting when the filter model saturates

› For this algorithm, additional controls include:

Resonance, which enunciates or chokes the nested filter

A *Brightness Mode* setting, with various options for how resonance peaks move:

Shift - Gently moves past the main cutoff, usually emphasizing a central peak

Double - A tuned mixture of the *Shift* and *Gravity* modes

Gravity - Pulls and pushes toward the main cutoff with a bit of magnetism

The bipolar *Brightness* control applies the set *Brightness Mode*, bending the new resonant nodes thru various harmonic — and inharmonic — positions

19.28.9.10. Ripple

A modern *Character* filter with hyper-resonance

› Three elemental modes provide different levels of fun/wreckage that often anchors to harmonics of the incoming signal

› Takes place inside a stereo, resonant filter, with standard options:

Drive to affect the incoming signal level

Main Cutoff Frequency control

Stereo input for cutoff modulation, with *Cutoff Modulation Amount* set in semitones

Key Tracking Amount, for using incoming note pitches to affect the cutoff buss

› For this algorithm, additional controls include:

Bipolar *Feedback Gain*, which feeds or chokes the nested filter

Feedback Cutoff Frequency, for tuning the nested filter

A *Nature* setting, with various models for the filter:

Earth - Gently moves past the main cutoff, usually emphasizing a central peak

Wind - Focused feedback, ready to blow

Fire - Broad feedback, with some motion

Two additional toggles, *Tweak Feedback* and *Tweak Feedforward*, modify those points in the filter circuit, either dampening or expanding resonance

A *Low Quality* toggle (in the Inspector Panel or via right-click context menu), for adjusting the filter's tuning and reducing the CPU load

19.28.9.11. High-pass

High-pass filter with adjustable slope

19.28.9.12. Low-pass

Low-pass filter with adjustable slope

19.28.10. Shaper Category

Various linear and nonlinear waveshapers

19.28.10.1. Chebyshev

Nonlinear shaper that can target harmonics

19.28.10.2. Distortion

Gentle distortion, with optional *Anti-aliasing* mode

19.28.10.3. Hard Clip

Simple clipper, with optional *Anti-aliasing* mode

19.28.10.4. Quantizer

Reduces signal resolution, with optional *Anti-aliasing* mode

19.28.10.5. Wavefolder

Reflects each cycle back on itself, with optional *Anti-aliasing* mode

19.28.10.6. Diode

A *Parametric* shaper modeling the classic circuit in a modern way

› *Offset* parameter for biasing the signal to be asymmetric

› *Drive* parameter for pushing the signal to bend

› *Low-pass Cutoff Frequency* control for rounding it off and restoring some order

› One *Drive* parameter for going thru the module's unique path, with optional high-order *Anti-aliasing* (*AA*)

19.28.10.7. Rectifier

Scales the positive and negative signal excursions separately, with optional *Anti-aliasing* mode

19.28.10.8. Saturator

Waveshaper with loud/quiet settings + bipolar skews. Module version of the **Saturator** device (see section 19.6.4).

19.28.10.9. Transfer

A freely drawable, segmented waveshaper, with BWCURVE file support, making use of the **Curve Editor** (see section 16.2.1.1)

› A freely drawable, segmented waveshaper

› With familiar shaper module controls and form

› *Anti-aliasing* (*AA*) toggle enables smoothed response of the shaper

› Modulatable *Drive* control that goes in both directions (±24 dB), for pushing the incoming signal to interesting parts of the curve

› A *Bipolar* toggle (±) maintains the curve's shape but rescales it, so that the minimum value is *-1* and the middle value is now *0* (zero)

› The unipolar mode (when *Bipolar* is off) has two options:

Clip - To truncate signals below zero at the value set there

Reflect - To mirror signal below zero negatively, good for processing a bipolar signal symmetrically

19.28.10.10. Push

A *Character* soft clipper with a detailed curve, using one *Drive* parameter for going thru the module's unique path, with optional high-order *Anti-aliasing* (*AA*)

19.28.10.11. Heat

A *Character* S-shaped clipper that starts soft but can drive hard, using one *Drive* parameter for going thru the module's unique path, with optional high-order *Anti-aliasing* (*AA*)

19.28.10.12. Soar

A *Character* soft wave folder that makes the quietest parts loud, using one *Drive* parameter for going thru the module's unique path, with optional high-order *Anti-aliasing* (*AA*)

19.28.10.13. Howl

A *Character* wave folder that puts different parts of the signal into loud focus, using one *Drive* parameter for going thru the module's unique path, with optional high-order *Anti-aliasing* (*AA*)

19.28.10.14. Shred

A *Character* non-linear wave folder for subtle cancellation or big-time artifacts, using one *Drive* parameter for going thru the module's unique path, with optional high-order *Anti-aliasing* (*AA*)

19.28.10.15. Curve

Remaps defined input and output levels

19.28.11. Delay/FX Category

Delay functions and other time-based audio FX

19.28.11.1. Delay

Simple delay

19.28.11.2. Long Delay

Delay set in time or beat units; also allows feedback connections

19.28.11.3. Mod Delay

Modulator delay with internal feedback loop

19.28.11.4. Chorus+

Chorus, with four different *Character* modes. Module version of the **Chorus+** device (see section 19.14.1).

19.28.11.5. Flanger+

Flanger, with four different *Character* modes. Module version of the **Flanger+** device (see section 19.14.3).

19.28.11.6. Phaser+

Phaser, with four different *Character* modes. Module version of the **Phaser+** device (see section 19.14.5).

19.28.11.7. All-pass

All-pass filter configured as a delay

19.28.11.8. Recorder

Signal capture device

19.28.12. Mix Category

Signal routing and mixing modules

19.28.12.1. Blend

Crossfades between two incoming signals

19.28.12.2. Mixer

Stereo mixer for up to six channels

19.28.12.3. Pan

Panning control

19.28.12.4. Stereo Width

Signal width control

19.28.12.5. Toggle In

Switch between two incoming signals, with a button directly on the module

19.28.12.6. Toggle Out

Switch between two outgoing paths, with a button directly on the module

19.28.12.7. Select In

Binary selector between two incoming signals

19.28.12.8. Select Out

Binary selector between two outputs

19.28.12.9. Merge

Router with up to eight in ports, passing one or two adjacent incoming signals out at a time

19.28.12.10. Split

Router with up to eight out ports, sending the incoming signal to one or two adjacent out ports at a time

19.28.12.11. LR Gain

Independent gain controls for a signal's left and right channels

19.28.12.12. Stereo Merge

Constructs a signal from left/right and mid/side components

19.28.12.13. Stereo Split

Separates a signal into its left/right and mid/side components

19.28.12.14. Voice Stack Mix

Modulatable processor with standard mix controls (volume, panning, solo, enable) for each voice in the stack, at any point within a patch

19.28.12.15. Voice Stack Tog

Modulatable processor to toggle the signal for each voice in the stack, at any point within a patch

19.28.13. Level Category

Amplitude-based functions, values, and converters

19.28.13.1. Level

Constant set in decibels

19.28.13.2. Value

Constant set as percentage

19.28.13.3. Amplify

Signal amplifier set in percentage (up to *800 %*)

19.28.13.4. Attenuate

Signal attenuator

19.28.13.5. Bias

Signal offset

19.28.13.6. Gain - dB

Decibel gain control

19.28.13.7. Gain - Vol

Volume gain control

19.28.13.8. Velo Mult

Velocity-controlled scaler

19.28.13.9. Average

Signal averager

19.28.13.10. Lag

Lag processor

19.28.13.11. Bend

Imposes a variable curve onto a signal

19.28.13.12. Clip

Signal clipper

19.28.13.13. Level Scaler

Scales incoming unipolar signal to a defined decibel range

19.28.13.14. Pinch

Imposes an S-curve onto an audio signal, with *Stereo-ize* option

19.28.13.15. Value Scaler

Scales incoming unipolar signal to a defined value range

19.28.13.16. AM/RM

Crossfades between dry carrier, classic amplitude modulation, and ring modulation

19.28.13.17. Hold

Level sustainer

19.28.13.18. Sample / Hold

Level sampler

19.28.13.19. Bi→Uni

Converts a bipolar signal to unipolar

19.28.13.20. Uni→Bi

Converts a unipolar signal to bipolar

19.28.13.21. Poly→Mono

Flattens any signal, making it the same for all voices. With five modes:

› *Last* - Newest voice's signal

› *Sum* - All voices are added together

› *Average* - All voices are averaged

› *Min* - Lowest signal level is used

› *Max* - Highest signal level is used

19.28.14. Pitch Category

Modules that produce pitch values

19.28.14.1. Pitch

Constant set as pitch

19.28.14.2. Octaver

Octave pitch shifter

19.28.14.3. Ratio

Ratio-based pitch shifter

19.28.14.4. Transpose

Semitone pitch shifter

19.28.14.5. Pitch Quantize

Quantizes incoming signal to designated or currently held pitch classes

19.28.14.6. by Semitone

Quantizes incoming signal to exact semitones

19.28.14.7. Pitch Buss

Pitch summing buss with attenuators for up to six inputs

› Attenuators are set in a range of ±36 semitones

› Inputs two to six also have a *Thru (No Attenuation)* option (a clickable = icon) that adds that incoming signal without attenuation, good in the case of actual pitch signals, etc.

19.28.14.8. Pitch Scaler

Scales incoming unipolar signal to a defined pitch range

19.28.14.9. Zero Crossings

Rough pitch estimator

19.28.14.10. Freq → Pitch

Hertz (or kilohertz) to pitch converter, with optional stereo detune

19.28.14.11. Pitch → Freq

Pitch to Hertz (or kilohertz) converter, with optional stereo detune

19.28.15. Math Category

Basic arithmetic operators

19.28.15.1. Constant

Constant for large, precise numbers

19.28.15.2. Invert

Gives a button to reverse polarity (× -1) of the incoming signal, with *Stereo-ness* option

19.28.15.3. Reciprocal

Gives a button to flip (1/x) the incoming signal, with *Stereo-ness* option

19.28.15.4. Add

Add two signals

19.28.15.5. Divide

Divide one signal from another

19.28.15.6. Multiply

Multiply two signals

19.28.15.7. Subtract

Subtract one signal from another

19.28.15.8. Abs

Separates a signal into its magnitude and sign components

19.28.15.9. Ceil

Rounds all decimal values up to the next integer

19.28.15.10. Floor

Rounds all decimal values down to the previous integer

19.28.15.11. MinMax

Provides the current higher and lower values of two signals

19.28.15.12. Quantize

Uses a set step size for the signal

19.28.15.13. Round

Rounds all decimal values below '0.5' increments down and those at or above '0.5' up

19.28.15.14. Product

Multiplies all inputs together

19.28.15.15. Sum

Adds all inputs together

19.28.15.16. Exp

Provides either 2^x, e^x, or 10^x of the incoming signal (x), depending on the *Base* parameter

19.28.15.17. Exponents

Provides a power of the incoming signal, with an integer *Exponent* parameter between *-9* and *+9*

19.28.15.18. Lin $\rightarrow$ dB

Converts linear amplitudes to decibel values

19.28.15.19. Log

Provides either log_2x, log_ex, or $log_{10}x$ of the incoming signal (x), depending on the *Base* parameter

19.28.15.20. Power

Raises one signal to the power of another

19.28.15.21. Roots

Provides a root of the incoming signal, with an integer *Degree* parameter between *1* and *9*

19.28.15.22. dB → Lin

Converts decibel values to linear amplitudes

19.28.16. Logic Category

Comparators and other modules that output logic signals

19.28.16.1. Button

Toggle for sending a logic signal

19.28.16.2. Trigger

Momentary toggle for sending a logic signal

19.28.16.3. Clock Divide

Divides a clock signal to trigger every N pulses

19.28.16.4. Clock Quantize

Holds a trigger signal until the next clock pulse

19.28.16.5. Gate Length

Produces a logic pulse of set duration on trigger

19.28.16.6. Gate Repeat

Produces repeated logic pulses of set duration while input is high

19.28.16.7. Logic Delay

Delays the high- or low-logic states

19.28.16.8. Latch

Allows trigger signals to alternate or set an output state

19.28.16.9. N-Latch

Allows trigger signals to alternate between multiple output states

19.28.16.10. =

Comparator assessing if two signals are roughly equal

19.28.16.11. ≥

Comparator assessing if one signal is either greater than or equal to another

19.28.16.12. >

Comparator assessing if one signal is greater than another

19.28.16.13. ≤

Comparator assessing if one signal is either less than or equal to another

19.28.16.14. <

Comparator assessing if one signal is less than another

19.28.16.15. ≠

Comparator assessing if two signals are unequal

19.28.16.16. NOT

Logic inverter

19.28.16.17. AND

Logic gate seeking all inputs to be true

19.28.16.18. OR

Logic gate seeking any input to be true

19.28.16.19. XOR

Logic gate seeking only one input to be true

19.28.16.20. NAND

Logic gate seeking any inputs to be false

19.28.16.21. NOR

Logic gate seeking all inputs to be false

19.28.16.22. XNOR

Logic gate seeking all inputs to be matching

19.29. Legacy Devices

These Bitwig devices were previously included at the top level of the program. They are still part of Bitwig Studio for compatibility purposes.

19.29.1. Audio MOD

(A Bitwig Studio version 1 *modulator* device; now a *container* device, when shown) A modulator that applies a filter and envelope follower to an incoming audio signal, which is then used as the control signal.

19.29.2. LFO MOD

(A Bitwig Studio version 1 *modulator* device; now a *container* device, when shown) A modulator that provides two low-frequency, tempo-syncable oscillators as independent modulation sources.

19.29.3. Note MOD

(A Bitwig Studio version 1 *modulator* device; now a *container* device, when shown) A modulator that takes incoming or designated note signals and creates summed, monophonic versions of their expressions along with a configurable envelope signal.

19.29.4. Step MOD

(A Bitwig Studio version 1 *modulator* device; now a *container* device, when shown) A step sequencer whose output is used as a modulation source.