sq - MIDI Sequencer Manual

sq Project

Generated on 2025-12-03

Contents

1 Introduction 3

2 sq 3
- 2.1 Overview . . . 3
- 2.2 Key Features . . . 3
- 2.3 Install from package . . . 4
- 2.4 Install with mise . . . 4
- 2.5 Quick Start . . . 4
- 2.6 Configuration . . . 4

3 Core Concepts 4
- 3.1 Pattern Creation . . . 4
- 3.2 Note Attributes . . . 5
- 3.3 Key Cycles and Arrangements . . . 5
- 3.4 Overlay System . . . 5
- 3.5 Actions . . . 5

4 Key Mappings 6
- 4.1 Key Bindings Quickstart . . . 6
- 4.2 Complete Reference . . . 7
- 4.3 Pattern Mode Mappings . . . 11
- 4.4 Mono Mode Pattern Mappings . . . 13
- 4.5 Chord Mode Mappings . . . 13
- 4.6 Arrangement Mappings . . . 15
- 4.7 Overlay Key Mappings . . . 15
- 4.8 Euclidean Rhythm Generator . . . 16

5 Beat Pattern Creation 16
- 5.1 Example . . . 16
- 5.2 Euclidean Rhythms . . . 17

6 Note Alteration 18
- 6.1 Accent . . . 18
- 6.2 Gate . . . 18
- 6.3 Ratchet . . . 19
- 6.4 Wait . . . 19

7 Actions 19
- 7.1 Line Reset . . . 19
- 7.2 Line Reset All . . . 19
- 7.3 Line Bounce . . . 19
- 7.4 Line Bounce All . . . 19
- 7.5 Line Skip Beat . . . 19

7.6 Line Delay 20
7.7 Line Reverse 20

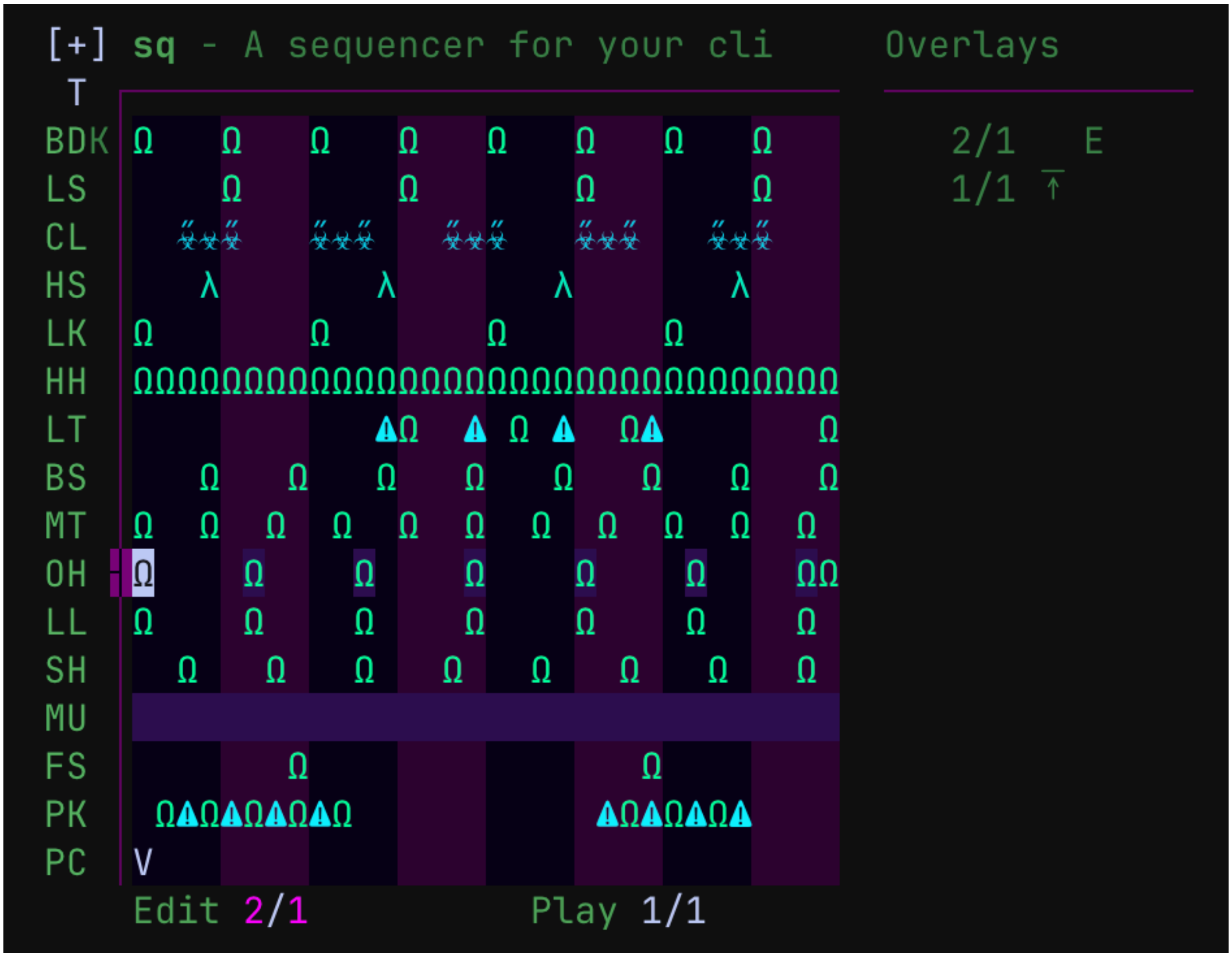
8 Arrangement **20**
8.1 Sections and Parts 20
8.2 Groups 20
8.3 Section Attributes 21

9 Key Cycles **22**

10 Overlay **23**
10.1 Overview 23
10.2 Overlay Key Definition 24
10.3 WIDTH 25
10.4 START 25
10.5 Order And Stacking 26

1 Introduction

2 sq



```
[+] sq - A sequencer for your cli     Overlays
 T
BDK Ω   Ω   Ω   Ω   Ω   Ω   Ω   Ω        2/1   E
LS      Ω       Ω       Ω       Ω        1/1 ⤒
CL    ☣☣☣   ☣☣☣   ☣☣☣   ☣☣☣   ☣☣☣
HS     λ       λ       λ       λ
LK  Ω       Ω       Ω       Ω
HH  ΩΩΩΩΩΩΩΩΩΩΩΩΩΩΩΩΩΩΩΩΩΩΩΩΩΩΩΩΩΩΩΩ
LT             ⚠Ω  ⚠ Ω ⚠  Ω⚠       Ω
BS     Ω   Ω   Ω   Ω   Ω   Ω   Ω   Ω
MT  Ω  Ω  Ω  Ω  Ω  Ω  Ω  Ω  Ω  Ω  Ω
OH  Ω    Ω    Ω    Ω    Ω    Ω    ΩΩ
LL  Ω    Ω    Ω    Ω    Ω    Ω    Ω
SH    Ω   Ω   Ω   Ω   Ω   Ω   Ω   Ω
MU
FS         Ω               Ω
PK   Ω⚠Ω⚠Ω⚠Ω⚠Ω           ⚠Ω⚠Ω⚠Ω⚠
PC  V
    Edit 2/1          Play 1/1
```

Figure 1: Beta

A powerful MIDI sequencer designed for the command line

2.1 Overview

sq is a terminal-based MIDI sequencer that brings midi sequencer capabilities to your CLI. Built with Go and designed for efficiency, sq offers rapid beat creation, complex arrangements, and advanced pattern manipulation through an intuitive keyboard-driven interface.

2.2 Key Features

- **MIDI Integration**: Control hardware or software instruments via MIDI
- **Rapid Beat Creation**: Create drum patterns in just a few keystrokes using pattern mode
- **Euclidean Rhythms**: Generate mathematically-distributed note patterns for complex polyrhythms
- **Advanced Overlays**: Add mathematical variations to sequences with the overlay system
- **Flexible Arrangements**: Structure songs with sections, parts and groups
- **Real-time Manipulation**: Modify patterns, accents, gates and timing while playing
- **Vim-inspired**: Familiar key bindings for efficient workflow

2.3 Install from package

Pre-built packages for macOS and Linux are found on the Releases page.

2.4 Install with mise

```
mise install ubi:chriserin/sq
```

2.5 Quick Start

1. **Launch sq**: `sq`
2. **Create a beat**: Move cursor with `hjkl`, press `1` for notes on every beat
3. **Play**: Press `Space` to play/stop
4. **Save**: `Ctrl+s` to save your sequence

2.5.1 Connecting to hardware or software

Midi inports and outports can be confusing. One software's inport is another software's outport and vice versa. When passing the `--outport` flag to `sq` you create an outport for sq that will be an inport for the other software. Both Logic and GarageBand scan the inports and automatically listen to all inports. Likewise, Both Logic and GarageBand create an inport that sq will automatically connect to if it is the first outport listed by `sq list`. When the outport is not listed first, you can connect to that outport with the `--midiout <partial name>` flag. `--midiout Logic` will connect to Logic's outport.

2.5.2 Basic Beat Creation Example

Create a basic beat with just 6 keystrokes:

- Bass drum: cursor on BD line, press `8` (note every 8 beats)
- Snare: `j` (down), `4` then `8` (note every 8 beats starting at beat 5)
- Hi-hat: `j` (down), `1` (note every beat)

The resulting sequence will look like this:

```
BDK┤▧       ▧       ▧       ▧
SN │    ▧       ▧       ▧       ▧
H1 │▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧
```

2.6 Configuration

sq uses Lua for configuration. Configuration files can be located in 4 different locations:

- `./`
- `./config/`
- `~/.sq/`
- `~/.config/sq/`

If a configuration file is not found, an initial configuration is written to `~/.confing/init.lua`.

3 Core Concepts

3.1 Pattern Creation

Pattern Mode enables rapid beat creation:

- **Numbers (1-9)**: Add/remove notes every N beats
- **Pattern Mode - Value**: `na` (accent), `nw` (wait), `ng` (gate), `nr` (ratchet)

- **Shift+Numbers**: Increase values, Numbers alone decrease values

  Detailed Guide: Beat Creation

3.2 Note Attributes

Each note has four modifiable properties:

- **Accent** (`A/a`): MIDI velocity (8 levels)
- **Gate** (`G/g`): Note duration (8 levels + long gates with `E/e`)
- **Ratchet** (`R/r`): Multiple hits per beat (8 levels)
- **Wait** (`W/w`): Swing timing delay (8 levels)

  Detailed Guide: Note Alteration

3.3 Key Cycles and Arrangements

Key Cycles track sequence iterations and drive both arrangements and overlays:

- Key line (marked with `K`) determines when cycles advance
- Arrangements use key cycles to switch between sections
- Overlays use key cycles to determine when variations apply

Arrangements structure your songs:

- **Sections**: Containers that reference parts with playback attributes
- **Parts**: The actual musical content
- **Groups**: Collections of sections that can repeat

  Detailed Guides: Key Cycles | Arrangement System

3.4 Overlay System

Overlays add mathematical variations to patterns:

- `1/1`: Root overlay
- `2/1`: Every 2nd cycle
- `3/1`: Every 3rd cycle
- `1/4`, `2/4`, `3/4`, `4/4`: First, second, third, fourth of every 4 cycles

Complex overlay keys support width, start delays, and stacking behaviors.

  Detailed Guide: Overlay System

3.5 Actions

Actions manipulate playback cursors:

- **Line Reset** (`ss`): Reset cursor to first beat
- **Line Bounce** (`sb`): Bounce between first beat and action
- **Line Skip** (`sk`): Skip current beat
- **Line Reverse** (`sr`): Reverse playback direction
- **Line Delay** (`sz`): Repeat current beat

Some actions have “All” variants (`sS`, `sB`, `sK`) that affect all playback cursors.

  Detailed Guide: Actions

4 Key Mappings

4.1 Key Bindings Quickstart

4.1.1 Navigation

- `hjkl`: Move cursor (left/down/up/right)
- `<>`: Jump to line start/end
- `bf/bl`: Jump to first/last line

4.1.2 Playback

- `Space`: Play/stop arrangement once
- `Alt+Space`: Loop arrangement
- `'+Space`: Loop current overlay
- `Ctrl+Space`: Loop current part
- `nl`: Toggle bounded loop (loop between bounds)
- `<>`: Expand left/right loop bound
- `, .`: Contract left/right loop bound

4.1.3 Pattern Creation

- `f`: Add single note
- `d`: Remove note
- `1-9`: Add/remove notes every N beats
- `shift+1-9`: Add note every N empty spaces
- `c`: Clear line from cursor to end
- `bu`: Apply Euclidean rhythm pattern
- `nv`: Reverse notes in selection or from cursor

4.1.4 Note Modification

- `A/a`: Increase/decrease accent
- `R/r`: Increase/decrease ratchet
- `G/g`: Increase/decrease gate by eighth of beat
- `E/e`: Increase/decrease gate by whole beat
- `W/w`: Increase/decrease wait

4.1.5 Arrangement Controls

- `Ctrl+a`: Toggle arrangement view
- `Ctrl+]`: New section after current
- `Ctrl+p`: New section before current
- `]s/[s`: Next/previous section

4.1.6 Advanced Features

- `o`: Toggle chord mode
- `v`: Visual selection mode
- `y/p`: Yank/paste
- `bd`: Duplicate note/chord/selection
- `m/M`: Mute/solo line
- `u/U`: Undo/redo

4.1.7 Input Modes

- `Ctrl+t`: Tempo controls

- `Ctrl+k`: Cycles input controls
- `Ctrl+b`: Beat input controls
- `Ctrl+e`: Accent input controls
- `Ctrl+o`: Overlay key controls
- `Ctrl+d`: MIDI setup controls

Use +/- to increase/decrease values for each control, use numbers 0 -9 to directly input a particular number. In MIDI setup controls use K/J to change increase/decrease values of every line.

4.2 Complete Reference

Mapping	Key Binding	Description
PlayStop	Space	Play the full arrangement once. If playing, stop
PlayOverlayLoop	' + Space	Play the current overlay in a loop
PlayLoop	Alt + Space	Play the full arrangement in a loop
PlayPart	Ctrl + Space	Play current part in a loop
PlayRecord	: + Space	Play the full arrangement once and send a record message at the beginning
Increase	+ / =	Increase value of current selection (tempo, beats, cycles, accents, etc.) or tempo by 5 if no specific selection
Decrease	-	Decrease value of current selection (tempo, beats, cycles, accents, etc.) or tempo by 5 if no specific selection
CursorLineStart	<	Move cursor to beginning of current line
CursorLineEnd	>	Move cursor to end of current line
CursorLastLine	b + l	Move cursor to last line
CursorFirstLine	b + f	Move cursor to first line
AccentIncrease	A	Increase accent value for current note.
AccentDecrease	a	Decrease accent value for current note.
ActionAddLineReset	s + s	Add line reset action to current line. When the playback cursor reaches this action, the playback cursor will reset to the first beat.
ActionAddResetAll	s + S	Add reset action all to current line. When the playback cursor reaches this action, all playback cursors will reset to the first beat.
ActionAddLineBounce	s + b	Add line bounce action to current line. When the playback cursor reaches this action it will reverse direction, and reverse again when reaching the line beginning creating a bouncing effect.
ActionAddLineBounceAll	s + B	Add line bounce all action to current line. When the playback cursor reaches this action all playback cursors will reverse direction, and reverse again when reaching the line beginning creating a bouncing effect.
ActionAddSkipBeat	s + k	Add skip beat action to current line. When the playback cursor reaches this action, all the playback cursors will advance an additional beat.
ActionAddLineReverse	s + r	Add line reverse action to current line. When the playback cursor reaches this action, the playback will reverse for this line.
ActionAddLineDelay	s + z	Add line delay action to current line.
ClearOverlay	C	Remove all notes and actions from the current overlay layer

Mapping	Key Binding	Description
GateIncrease	G	Increases the gate value for current note. The gate corresponds to the length of the note.
GateDecrease	g	Decreases the gate value for current note. The gate corresponds to the length of the note.
GateBigIncrease	E	Increase gate value for current note by 8, or 1 full beat. The gate corresponds to the length of the note.
GateBigDecrease	e	Decrease gate value for current note by 8, or 1 full beat. The gate corresponds to the length of the note.
RotateDown	J	Rotate pattern down. In the current column shift all notes down by one line, with a note in the bottom line moving to the top line
RotateUp	K	Rotate pattern up. In the current column shift all notes down by one line, with a note in the bottom line moving to the top line
RotateLeft	H	Rotate pattern left. On the current line shift all notes right of the cursor left by one beat. A note at the cursor's beat will be moved to the last beat of the line.
RotateRight	L	Rotate pattern right. On the current line shift all notes right of the cursor right by one beat. A note at the last beat will be moved to the cursor's beat.
SelectKeyLine	Y	Selects the current line as the key line. The KeyCycle of the part is advanced when the cursor returns to the first beat. See Key Cycles
Mute	m	Mute the current line. Midi messages will not be sent from this line when the line is muted.
Solo	M	Solo the current line. Only midi messages from this line or other soloed lines will be sent.
RatchetDecrease	r	Decrease the number of hits evenly divided within the span of 1 beat
RatchetIncrease	R	Increase the number of hits evenly divided within the span of 1 beat
WaitIncrease	W	Increase the wait value for current note. The wait value is the time between the playback of the note's beat and the sending of the midi message. This is useful for creating a swing effect.
WaitDecrease	w	Decrease the wait value for current note. The wait value is the time between the playback of the note's beat and the sending of the midi message. This is useful for creating a swing effect. The initial value for a note will 0, in which case WaitDecrease will not have an effect.
NextTheme	] + c	Switch to next theme. A theme consists of the set of colors used to draw the sq application and the set of icons used to represent different accent levels.
PrevTheme	[+ c	Move to the previous theme. A theme consists of the set of colors used to draw the sq application and the set of icons used to represent different accent levels.
NextSection	] + s	Move to the next section within the arrangement. If the next section is a group, then this mapping will move to the first section within that group.
PrevSection	[+ s	Move to the previous section. Move to the next section within the arrangement. If the previous section is a group, then this mapping will move to the last section within that group.
NewSectionAfter	Ctrl +]	Create new section after the current section

Mapping	Key Binding	Description
NewSectionBefore	Ctrl + p	Create new section before the current section
BeatInputSwitch	Ctrl + b	This selects the current part's beats which can be increased or decreased with +/-. Using this key combination again will move through selections Beats and Start Beats.
CyclesInputSwitch	Ctrl + k	This selects the current part's cycles which can be increased or decreased with +/-. Using this key combination again will move through selections Cycles and Start Cycles.
AccentInputSwitch	Ctrl + e	This selects the controls that determine the accent values and target. Use +/- to increase and decrease the selections. See Accent Controls
OverlayInputSwitch	Ctrl + o	This selects the inputs that control the overlay period/key. See Overlay Key Controls
SetupInputSwitch	Ctrl + d	Select the inputs that control the midi message for each line. Pressing this key combo repeatedly will move through the channel, target and value inputs.
TempoInputSwitch	Ctrl + t	Select the inputs that control the tempo and subdivision. Press once to select the tempo input, press again to select the subdivisions input.
OverlayStackToggle	Ctrl + u	Toggle the behaviour of the current overlay layer between three options: No association, press up, press down. See Overlays
ChangePart	Ctrl + c	Change the part of the section to either an existing part or a new part
ToggleArrangementView	Ctrl + a	Open the arrangement view when closed. Focus the arrangement view while unfocused and open. Press enter to move focus back to the grid. While open and focused, close the arrangement view. See Arrangement
NewLine	Ctrl + l	Create a new line with a value 1 greater than the previous line
New	Ctrl + n	Create a new sequence using the same template as the current sequence
Save	Ctrl + s	Save the current sequence. If not previously saved, you will be prompted to name the new file. The file will be saved in the directory from which you opened sq
ToggleAccentMode	n + a	Enter Pattern Mode - Accent. Use the facilities of pattern mode to increase or decrease the accent values of the line. See Pattern Mode
ToggleWaitMode	n + w	Enter Pattern Mode - Wait. Use the facilities of pattern mode to increase or decrease the wait values of the line. See Pattern Mode
ToggleGateMode	n + g	Enter Pattern Mode - Gate. Use the facilities of pattern mode to increase or decrease the gate values of the line. See Pattern Mode
ToggleRatchetMode	n + r	Enter Pattern Mode - Ratchet. Use the facilities of pattern mode to increase or decrease the ratchet values of the line. See Pattern Mode
RatchetInputSwitch	Ctrl + y	Select the inputs that control the ratchets for the current note. Press again to select the Span input.
ClearLine	c	Remove all notes from the current line from the current cursor position to the end
NoteRemove	d	Remove note at current position, and remove it from any stacked overlays if the current overlay is higher than the overlay of the current note

Mapping	Key Binding	Description
OverlayNoteRemove	x	Remove note from overlay at current position, allowing notes in lower layers to show through
TogglePlayEdit	b + e	Toggle play edit mode. Press while playing to ensure the current overlay/part does not change while editing. Press again to allow changing.
NoteAdd	f	Add note at current position
ReloadFile	b + r	Reload current file, any changes since the last save will be lost
ActionAddSpecificValue	b + v	Add specific value note to the grid. When cursor is above this note, +/- will affect the specific value of the note
CursorLeft	h	Move cursor left
CursorDown	j	Move cursor down
CursorUp	k	Move cursor up
CursorRight	l	Move cursor right
ToggleChordMode	o	Toggle chord mode. See Chord Mode
Yank	y	Copy current selection to buffer. Copies all values of a visual selection or the value under cursor if no visual selection.
Paste	p	Paste the buffer at the position of the cursor
Duplicate	b + d	Duplicate what is under the cursor. For single notes: duplicates to the next open beat. For chords: duplicates the entire chord to the next beat after the chord is complete. For visual selections: duplicates selection to following area.
Quit	q	Quit the application
Undo	u	Undo last action
Redo	U	Redo last undone action
ToggleVisualMode	v	Toggle visual selection for copying/pasting with `y/p`
NextOverlay	{	Move to next overlay. See Overlays
PrevOverlay	}	Move to previous overlay. See Overlays
Enter	Enter	Confirm current action, move focus to the grid when elsewhere, or escape from visual mode
Escape	Esc	Cancel current action or exit mode, move focus to the grid when elsewhere, or escape from visual mode
ToggleClockPreRoll	b + c	Toggle clock pre-roll (adds pre-roll before start to allow external hardware or software to sync to the Midi Clock)
PlayAlong	; + Space	Play along with external clock
ToggleVisualLineMode	V	Toggle visual line selection
ToggleTransmitting	b + t	Toggle transmitting MIDI messages
MuteAll	b + m	Mute all lines
UnmuteAll	b + M	Unmute all lines
ClearAllOverlays	b + C	Clear all overlays
SaveAs	Ctrl + w	Save the current sequence with a new name

Mapping	Key Binding	Description
ToggleGateNoteMode	n + G	Toggle gate note mode
ToggleWaitNoteMode	n + W	Toggle wait note mode
ToggleAccentNoteMode	n + A	Toggle accent note mode
ToggleRatchetNoteMode	n + R	Toggle ratchet note mode
ModifyKeyInputSwitch	Ctrl + x	Modify the key of the current note
RemoveOverlay	D	Remove the current overlay
IncreaseAllChannels	K	Increase all channels (when channel is selected)
DecreaseAllChannels	J	Decrease all channels (when channel is selected)
IncreaseAllNote	K	Increase all note values (when note value is selected)
DecreaseAllNote	J	Decrease all note values (when note value is selected)
MidiPanic	b + p	Send MIDI panic (all notes off)
ToggleHideLines	b + h	Toggle hiding lines with no notes
ToggleBoundedLoop	n + l	Toggle bounded loop mode. When enabled, overlay playback loops between left and right bounds instead of the full sequence
ExpandLeftLoopBound	<	Expand the left loop bound one beat to the left, increasing the loop region size
ExpandRightLoopBound	>	Expand the right loop bound one beat to the right, increasing the loop region size
ContractLeftLoopBound	,	Contract the left loop bound one beat to the right, decreasing the loop region size
ContractRightLoopBound	.	Contract the right loop bound one beat to the left, decreasing the loop region size
Euclidean	b + u	Apply Euclidean rhythm pattern to selection or pattern. Enter number of hits when prompted, then press Enter to confirm
Reverse	n + v	Reverse notes from cursor to end of line, or reverse notes within visual selection

4.3 Pattern Mode Mappings

The keys of pattern mode have two different behaviours, value and fill. The default mode is Pattern Mode - Fill.

4.3.1 PATTERN MODE - Fill

This is the default mode. When sq starts, it starts in Pattern Mode - Fill.

Numbers will a note every X beats from the cursor to the end of the line. If a note already exists in that location the note will be removed.

EXAMPLE: With the cursor at the start of the line `1` will add a note at every beat. `1` will then remove a note at every beat.

Mapping	Key Binding	Description
NumberPattern	1	Add/remove a note every beat

Mapping	Key Binding	Description
NumberPattern	2	Add/remove a note every 2nd beat
NumberPattern	3	Add/remove a note every 3rd beat
NumberPattern	4	Add/remove a note every 4th beat
NumberPattern	5	Add/remove a note every 5th beat
NumberPattern	6	Add/remove a note every 6th beat
NumberPattern	7	Add/remove a note every 7th beat
NumberPattern	8	Add/remove a note every 8th beat
NumberPattern	9	Add/remove a note every 9th beat
NumberPattern	shift+1 / !	Add a note every empty space
NumberPattern	shift+2 / @	Add a note every empty 2nd space
NumberPattern	shift+3 / #	Add a note every empty 3rd space
NumberPattern	shift+4 / $	Add a note every empty 4th space
NumberPattern	shift+5 / %	Add a note every empty 5th space
NumberPattern	shift+6 / ^	Add a note every empty 6th space
NumberPattern	shift+7 / &	Add a note every empty 7th space
NumberPattern	shift+8 / *	Add a note every empty 8th space
NumberPattern	shift+9 / (	Add a note every empty 9th space

4.3.2 PATTERN MODE - Value (Accent, Gate, Ratchet, Wait)

To enter pattern mode for a value, type `na` for accent, `nw` for wait, `nr` for ratchet or `ng` for gate.

Mapping	Key Binding	Description
NumberPattern	shift+1 / !	Increase value every beat
NumberPattern	shift+2 / @	Increase value every 2nd beat
NumberPattern	shift+3 / #	Increase value every 3rd beat
NumberPattern	shift+4 / $	Increase value every 4th beat
NumberPattern	shift+5 / %	Increase value every 5th beat
NumberPattern	shift+6 / ^	Increase value every 6th beat
NumberPattern	shift+7 / &	Increase value every 7th beat
NumberPattern	shift+8 / *	Increase value every 8th beat
NumberPattern	shift+9 / (	Increase value every 9th beat
NumberPattern	1	Decrease value every beat
NumberPattern	2	Decrease value every 2nd beat
NumberPattern	3	Decrease value every 3rd beat
NumberPattern	4	Decrease value every 4th beat
NumberPattern	5	Decrease value every 5th beat

Mapping	Key Binding	Description
NumberPattern	6	Decrease value every 6th beat
NumberPattern	7	Decrease value every 7th beat
NumberPattern	8	Decrease value every 8th beat
NumberPattern	9	Decrease value every 9th beat

4.3.3 PATTERN MODE NOTE - Value (Accent, Gate, Ratchet, Wait)

To enter pattern mode note for a value, type `nA` for accent, `nW` for wait, `nR` for ratchet or `nG` for gate.

Mapping	Key Binding	Description
NumberPattern	shift+1 / !	Increase value every note
NumberPattern	shift+2 / @	Increase value every 2nd note
NumberPattern	shift+3 / #	Increase value every 3rd note
NumberPattern	shift+4 / $	Increase value every 4th note
NumberPattern	shift+5 / %	Increase value every 5th note
NumberPattern	shift+6 / ^	Increase value every 6th note
NumberPattern	shift+7 / &	Increase value every 7th note
NumberPattern	shift+8 / *	Increase value every 8th note
NumberPattern	shift+9 / (	Increase value every 9th note
NumberPattern	1	Decrease value every note
NumberPattern	2	Decrease value every 2nd note
NumberPattern	3	Decrease value every 3rd note
NumberPattern	4	Decrease value every 4th note
NumberPattern	5	Decrease value every 5th note
NumberPattern	6	Decrease value every 6th note
NumberPattern	7	Decrease value every 7th note
NumberPattern	8	Decrease value every 8th note
NumberPattern	9	Decrease value every 9th note

4.4 Mono Mode Pattern Mappings

Enter mono mode with `O`. Mono mode mappings work the same way as the Pattern mappings listed above, except they work over every line, not just one line.

4.5 Chord Mode Mappings

Chord mode allows users to create and manipulate chords with a set of key mappings.

Example: `tM` create a Major Triad Chord. `7M` will add a major seventh to that chord. `]i` will invert the chord once, placing the root note 12 steps higher at the top of the chord.

Some mappings exist in Pattern Fill mode as well. `L` will move the entire chord to the right one beat. `A` will increase the accent value of every note in the chord.

Pattern mode (value) also works differently in chord mode. Enter pattern mode for accents with `na` and then use `shift+2` to increase the accent value for every 2nd note in the chord.

Chord notes can be doubled. Pressing `]d` once will double the first note of the second. Pressing `]d` again will double the second note of the while the first note remains doubled. If there are no more notes to double, `]d` will remove all doubled notes. `[d` behaves the same way but reversed.

Arpeggiate the notes of the chord with `]p` or `[p`. These mappings will cycle through the available arpeggiated patterns, at the moment there are only two patterns: up and down.

Mapping	Key Binding	Description
MajorTriad	t + M	Add major triad chord
MinorTriad	t + m	Add minor triad chord
DiminishedTriad	t + d	Add diminished triad chord
AugmentedTriad	t + a	Add augmented triad chord
MinorSeventh	7 + m	Add minor seventh
MajorSeventh	7 + M	Add major seventh
AugFifth	5 + a	Add augmented fifth
DimFifth	5 + d	Add diminished fifth
PerfectFifth	5 + p	Add perfect fifth
MinorSecond	2 + m	Add minor second
MajorSecond	2 + M	Add major second
MinorThird	3 + m	Add minor third
MajorThird	3 + M	Add major third
PerfectFourth	4 + p	Add perfect fourth
MajorSixth	6 + M	Add major sixth
Octave	8 + p	Add octave
MinorNinth	9 + m	Add minor ninth
MajorNinth	9 + M	Add major ninth
DecreaseInversions	[+ i	Decrease chord
IncreaseInversions	] + i	Increase chord
OmitRoot	1 + o	Omit root note from chord
OmitSecond	2 + o	Omit second note from chord
OmitThird	3 + o	Omit third note from chord
OmitFourth	4 + o	Omit fourth note from chord
OmitFifth	5 + o	Omit fifth note from chord
OmitSixth	6 + o	Omit sixth note from chord
OmitSeventh	7 + o	Omit seventh note from chord
OmitOctave	8 + o	Omit eighth note from chord
OmitNinth	9 + o	Omit ninth note from chord
RemoveChord	D	Remove chord at current position

Mapping	Key Binding	Description
NextArpeggio	] + p	Next arpeggio pattern
PrevArpeggio	[+ p	Previous arpeggio pattern
NextDouble	] + d	Next double pattern
PrevDouble	[+ d	Previous double pattern
ConvertToNotes	n + n	Convert chord to individual notes
RotateRight	L	Move chord to the right
RotateLeft	H	Move chord to the left
RotateUp	K	Move chord up
RotateUp	K	Move chord down

4.6 Arrangement Mappings

Once having opened and focused the arrangement view with `ctrl+a` a new set of mappings are available. Some mappings are duplicated between the grid and the arrangement view.

Mapping	Key Binding	Description
CursorUp	k	Move the cursor to the previous arrangement section
CursorDown	j	Move the cursor to the next arrangement section
CursorLeft	h	Move the cursor to the section attribute to the left
CursorRight	l	Move the cursor to the section attribute to the right
Increase	+	Increase the value of the currently selected section attribute
Decrease	-	Decrease the value of the currently selected section attribute
GroupNodes	g	Group one or two parts together
DeleteNode	d	Remove the current section attribute
MovePartDown	J	Move the current section below the next section
MovePartUp	K	Move the current section above the next section
RenamePart	R	Rename the current part
Escape	esc / enter	Move focus back to the grid

4.7 Overlay Key Mappings

Once having focused the overlay key inputs with `ctrl+o` a new set of mappings are available.

Mapping	Key Binding	Description
FocusWidth	:	Select the width attribute
FocusInterval	/	Select the interval attribute
FocusShift	^	Select the shift attribute
FocusStart	S	Select the start attribute

Mapping	Key Binding	Description
RemoveStart	s	Remove the start attribute (set to zero)
Increase	+	Increase the selected value
Decrease	-	Decrease the selected value
Escape	esc / enter	Return focus to the grid

4.8 Euclidean Rhythm Generator

The Euclidean rhythm generator creates evenly distributed note patterns based on the mathematical algorithm by Bjorklund. This is useful for creating polyrhythmic patterns, interesting drum patterns, and non-standard subdivisions.

4.8.1 Usage

1. Position your cursor where you want the pattern to start, or create a visual selection
2. Press `b + u` to activate the Euclidean rhythm generator
3. Enter the number of hits you want distributed across the pattern (0-9 for single digits, or type multiple digits like "13")
4. Press `Enter` to confirm and generate the pattern
5. Press `Escape` to cancel without generating

4.8.2 Pattern Scope

- **Without visual selection**: The pattern will be applied from the cursor position to the end of the current line
- **With visual selection**: The pattern will be applied only within the selected region

The algorithm distributes the specified number of hits as evenly as possible across the available beats, creating mathematically optimal rhythmic patterns.

4.8.3 Examples

- **Euclidean(3, 8)**: Three hits over 8 beats creates a tresillo pattern: `X..X..X.`
- **Euclidean(5, 8)**: Five hits over 8 beats creates a Cuban cinquillo: `X.X.X.XX`
- **Euclidean(5, 12)**: Five hits over 12 beats: `X..X.X..X.X.`
- **Euclidean(7, 16)**: Seven hits over 16 beats creates an asymmetric pattern

The Euclidean rhythm generator respects the current overlay and will place notes on the active layer.

5 Beat Pattern Creation

Creating beat patterns is simple and quick in sq with pattern mode. sq begins in pattern mode — fill, where number keys correspond to beat intervals.

5.1 Example

Here is a blank pattern with some named lines that correspond to a bass drum (BD), a snare (SN) and a high hat (H1).

```
BDK┤
SN │
H1 │
```

With the cursor at the first beat of the bass drum, press `8` to create a note every 8 beats.

```
BDK┤▧       ▧       ▧       ▧
SN │
H1 │
```

Move the cursor down to the snare line with `j` and press `4` and `8` to create notes on every 8th beat starting at the 5 beat.

```
BDK│▧       ▧       ▧       ▧
SN ┤    ▧       ▧       ▧       ▧
H1 │
```

For the high hat, move down one line with `j` and press `1` to create a high hat hit on every beat.

```
BDK│▧       ▧       ▧       ▧
SN │    ▧       ▧       ▧       ▧
H1 ┤▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧▧
```

6 keystrokes to create a very simple beat.

To create some accents on the high hats, press `na` to enter Pattern Mode — Accent and `shift+2` to increase the velocity of the high hat hit on every other beat. Press `enter` to escape from this mode.

```
BDK┤▧       ▧       ▧       ▧
SN │    ▧       ▧       ▧       ▧
H1 │▤▧▤▧▤▧▤▧▤▧▤▧▤▧▤▧▤▧▤▧▤▧▤▧▤▧▤▧▤▧▤▧
```

That’s 4 additional key strokes to create that accent pattern.

5.2 Euclidean Rhythms

For more complex and mathematically-distributed patterns, you can use the Euclidean rhythm generator. This creates evenly-spaced note patterns that are useful for polyrhythms and interesting drum patterns.

5.2.1 Creating a Euclidean Pattern

1. Position your cursor at the start of a line
2. Press `bu` to activate the Euclidean rhythm generator
3. Type the number of hits you want (e.g., “5” for 5 hits)
4. Press `Enter` to generate the pattern

For example, to create a 5-hit pattern over 16 beats on a percussion line:

```
PC │
```

Press `bu`, type `5`, and press `Enter`:

```
PC │▧  ▧  ▧  ▧  ▧
```

The hits are distributed as evenly as possible across the available beats.

5.2.2 Using with Visual Selection

You can also apply Euclidean rhythms to a specific range using visual mode:

1. Press `v` to enter visual mode
2. Use `l` to expand your selection to the desired length
3. Press `bu` to activate the Euclidean generator

4. Type the number of hits
5. Press `Enter` to generate

This allows you to create Euclidean patterns of any length within your sequence.

5.2.3 Common Euclidean Patterns

- **Euclidean(3, 8)**: Classic tresillo pattern used in Latin music
- **Euclidean(5, 8)**: Cuban cinquillo rhythm
- **Euclidean(5, 12)**: Common pattern for complex time signatures
- **Euclidean(7, 16)**: Asymmetric pattern for experimental rhythms

Euclidean rhythms work on the current overlay, so you can layer multiple Euclidean patterns with different hit counts to create complex polyrhythmic structures.

6 Note Alteration

When `f` (fill) is pressed a note will be created at the cursors current position. This note will have default properties for accent and gate that are ideal for triggering a drum on either a hardware or software drum machine.

A default note will have an accent value of 60 and a gate value of 20ms. In other words, a midi note on message will be sent with a velocity of 60 and 20ms later, a note off message will be sent.

There are 4 different values that can be changed for each note: Accent, Gate, Ratchet and Wait.

6.1 Accent

Accent by default corresponds to the velocity of a midi note on message. There are by default 8 different accent levels. Pressing `A` will increase the accent level for the note at the cursor position. Pressing `a` will decrease the accent level for this note.

There are different unicode characters for each accent level to help visually recognize patterns being created and to give visual feedback when changing the accent level of a note. These unicode characters will be different for every theme. See Themes (NextTheme/PrevTheme mappings).

6.2 Gate

The gate attribute corresponds to the amount of time between the midi note on message and the midi note off message. By default there are 8 different gate levels. Pressing `G` will increase the gate level for the note at the cursor position. Pressing `g` will decrease the gate level for this note.

The initial gate level will be the shortest possible gate, which is an absolute time of 20ms. Each gate level above this level will create a gate value that is a percentage of the interval created by the tempo and subdivision.

For instance, a gate level of 5 corresponds to a percentage value of 0.5. When the tempo is 120 and the subdivision value is 2, then there are 240 subdivisions per minute and 250ms between each beat. The gate length given these factors is 50% of 250ms or 125ms.

When a gate is increased or decreased a small symbol below the note will change.

6.2.1 Long Gates

Pressing `E` will increase the gate level above the length of the current Beat. Pressing `e` will similarly decrease the gate level. Gates up to 16 beats are currently possible.

When the gate length is longer than the beat interval a bar extending from the note will be drawn to indicate it's length.

6.3 Ratchet

Ratchet corresponds to the number of midi note on messages that will be sent for the note. There are 8 different Ratchet levels and the initial ratchet level for a note will be 1 corresponding to 1 midi note on message (a hit). Pressing `R` will increase the ratchet level for the note at the cursor position. Pressing `r` will decrease the ratchet level for this note.

For each ratchet above level 2 the length of the gate will be 20ms. The ratchet hits will be evenly distributed through the beat interval.

When the ratchet level is increased or decreased a small symbol corresponding to the ratchet level will change.

6.4 Wait

The wait attribute corresponds to the amount of time between the beat and the midi note on message. By default there are 8 different wait levels. Each level corresponds to a percentage of the beat interval, from 0 to 54. Pressing `W` will increase the wait level for the note at the cursor position. Pressing `w` will decrease the wait level for this note.

Combined with pattern mode this can be useful for creating swing effects.

7 Actions

Each line has a playback cursor. There are various actions that can be added to a line that will manipulate the playback cursor in different ways. Each of these actions can be added to a line with a keycombo begining with `s`.

7.1 Line Reset

`ss` — Add a Line Reset Action

When the playback cursor reaches this action, it will reset to the first beat.

7.2 Line Reset All

`sS` — Add a Line Reset All Action

When the playback cursor reaches this action, it will reset all playback cursors to the first beat.

7.3 Line Bounce

`sb` — Add a Line Bounce Action

When the playback cursor reaches this action, it will go backwards until reaching the first beat. When reaching the first beat it will change directions and go forwards until again reaching the bounce action. Unless some other action intervenes it will bounce back and forth between the first beat and the bounce action.

7.4 Line Bounce All

`sB` — Add a Line Bounce All Action

When the playback cursor reaches this action, all playback cursors will go backwards until reaching the first beat. When each playback cursor reaches the first beat they will change directions and go forwards until the line with the playback cursor again reaches the Line Bounce All action. Unless some other action intervenes each playback cursor will bounce back and forth between the first beat and this action.

7.5 Line Skip Beat

`sk` — Add a Line Skip Action

When the playback cursor reaches this action, it will skip this beat. This will place this line's playback cursor ahead of other playback cursors.

7.6 Line Delay

`sz` — Add a Line Delay Action

When the playback cursor reaches this action it will pause on the beat before this action and play the note at that location repeatedly until either interrupted by another action or that part or overlay changes.

7.7 Line Reverse

`sr` — Add a Line Reverse

When the playback cursor reaches this action the playback cursor moves in the opposite direction. When the playback cursor reaches the start of the line it will reset to the location of the action and will continue to move backwards.

8 Arrangement

Arrangements consist of parts, sections and groups. The Arrangement View is shown when pressing `ctrl+a`.

8.1 Sections and Parts

An arrangement consists of sections. These sections are sequenced one after the other as listed in the Arrangement View. Each section contains a part that could be the same or different from other sections.

Initially, there is one section and one part. In the arrangement view that looks like this:

```
Section ────────────────────⟳ Amount─────⟳ Start──Start Beat──────⟳ Keep
  Part 1 ● ────────────────────────1───────────1───────────0───────────-
```

To create another section press `ctrl+]` and accept the “Choose Part” prompt with `enter`. Now there are two sections in the arrangement view:

```
Section ────────────────────⟳ Amount─────⟳ Start──Start Beat──────⟳ Keep
  Part 1                           1           1           0           -
  Part 2 ● ────────────────────────1───────────1───────────0───────────-
```

“Part 1” will play for 1 key cycle and “Part 2” will also play for one key cycle. You can add another section but choose an already existing part by pressing `ctlr+]` and selecting an existing part with + before accepting with `enter`. This will create an arrangement that looks like this:

```
Section ────────────────────⟳ Amount─────⟳ Start──Start Beat──────⟳ Keep
  Part 1                           1           1           0           -
  Part 2                           1           1           0           -
  Part 1 ● ────────────────────────1───────────1───────────0───────────-
```

Sections are just a thin wrapper around parts that determines how many key cycles a part will play along with a set of attributes described below.

Move the section selection down and up with `j` and `k`.

8.2 Groups

A group can be used to iterate over a sequence of parts multiple times. When the Arrangement View is focused, press `g` to create a group that consists of the now focused section and the section after it, if a following section exists.

When “Part 2” of our working example is now selected, pressing `g` will alter an arrangement to look like this:

```
Section ──────────────────────────⟳ Amount─────⟳ Start──Start Beat──────⟳ Keep
   Part 1                              1           1         0          -
  Group ──────────────────────────────── 1──────────────────────────────────────────
   ├─ Part 2  ●                        1           1         0          -
   └─ Part 1                           1           1         0          -
```

The only attribute of the group is the “Cycles Amount”, change this value with `+`/`-`.

Creating a group has the effect of creating an arrangement tree.

8.3 Section Attributes

Each section has four attributes, Cycle Amount, Cycle Start, Start Beat and Cycle Keep. Move left and right between these attributes with `h` and `l`.

8.3.1 Cycles Amount / ⟳ Amount

The number of Key Cycles a section will play is determined by the Cycles Amount. This defaults to 1 as a way to force a more considered choice for the sequence. The Cycles Amount is the left most attribute and is selected by default when the arrangement view is focused initially, allowing quick selections of higher Cycle Amount values with the + key.

8.3.2 Cycles Start / ⟳ Start

The current number of Key Cycles determines the active Overlay. A section can differ from another section playing the same part by using a different Key Cycle at the start. By starting with a key cycle of 2 rather than 1, a part will begin with overlay `2/1`.

8.3.3 Start Beat

A part does not need to start at the first beat. Using the start beat attribute, a section can start at any beat of the part. This can be used to inject oddly timed breaks between sections or to create fills at the starts of sections.

8.3.4 Keep Cycles

If a section is within a group, and the group iterates multiple times, it is possible to maintain the Key Cycles between plays within a group. A “□ ” as a Keep Cycles value indicates that Key Cycles for that section will persist between group plays.

For the following arrangement:

```
Section ──────────────────────────⟳ Amount─────⟳ Start──Start Beat──────⟳ Keep
   Part 1                              1           1         0          -
   Group                                 2
  ├─ Part 2 ● ───────────────────── 1──────────1──────────0──────────□
   └─ Part 1                           1           1         0          -
```

The group will play twice for a sequence of “Part 2”, “Part 1”, “Part 2”, “Part 1”. The first time “Part 2” plays the Key Cycles start with a value of 1. The second time “Part 2” plays the Key Cycles start with a value of 2.

8.3.5 Moving Sections

When there are multiple sections it is possible to move the now selected section up or down within the arrangement tree with `J` or `K`. When a group follows a section that you want to move down then pressing `J` will move the section *into* the group. To arrange the section so that the section follows the group keep press `J` until the section is below and exists at the same tree level.

For instance, given the following arrangement:

```
Section ──────────────────────────⟳ Amount──────⟳ Start──Start Beat──────⟳ Keep
  Part 1 ● ─────────────────────────── 1──────────────1────────────────0───────────────–

   Group                                2
   ├─ Part 2                          1              1                0              ⍰
   └─ Part 1                          1              1                0              –
```

Pressing `J` will alter the arrangement to be:

```
  Group                                2
  ├─ Part 1  ● ───────────────────── 1──────────────1────────────────0───────────────–

   ├─ Part 2                          1              1                0              ⍰
   └─ Part 1                          1              1                0              –
```

Pressing `J` again will alter the arrangement to be:

```
 Group                                2
   ├─ Part 2                          1              1                0              ⍰
  ├─ Part 1  ● ───────────────────── 1──────────────1────────────────0───────────────–

   └─ Part 1                          1              1                0              –
```

Pressing `J` again will alter the arrangement to be:

```
Section ──────────────────────────⟳ Amount──────⟳ Start──Start Beat──────⟳ Keep
   Group                                2
   ├─ Part 2                          1              1                0              ⍰
   ├─ Part 1                          1              1                0              –
  └─ Part 1  ● ───────────────────── 1──────────────1────────────────0───────────────–
```

Pressing `J` a final time will alter the arrangement to be:

```
 Section ──────────────────────────⟳ Amount──────⟳ Start──Start Beat──────⟳ Keep
   Group                                2
   ├─ Part 2                          1              1                0              ⍰
   └─ Part 1                          1              1                0              –
  Part 1 ● ─────────────────────────── 1──────────────1────────────────0───────────────–
```

Our first section has now been moved to the bottom of the arrangement and is now the final section.

Pressing `K` 4 times will move the selected section up four times so that it once again is the beginning section of the sequence.

9 Key Cycles

Every time a playback cursor returns to the start of the Key line, the key cycles for this section are increased. Because sq has playback cursors for each line, one line must be the Key line. This is signified by the `K` next to the line number/name.

Key Cycles are central to both the Arrangement View and the Overlay Key. For Arrangement, we use the count of Key Cycles to determine when to move to the next part. For Overlays, we use the Key Cycles to determine the Overlay to use and when to use it.

Without any actions, a playback cursor will reach the end of the pattern and return to the start of the line, increasing the Key Cycles. If the Arrangement has a setting of just 1 Cycle, which is the default, then this section is done and the arrangement moves to the next section, resetting the Key Cycles. Or if there are no more sections, then the sequence is over and playback stops.

The amount of key cycles for a section can be increased with `ctrl+k` which selects the key cycle inputs.

There are various actions (SEE ACTIONS) that you can place in any line to manipulate the playback cursor for that line. For instance, if you add the line reset action (`ss`) to the line with the Key indicator at beat 5, then when playback occurs only the first 4 beats of that line will play before the playback cursor for that line resets to zero, which will increase the amount of Key Cycles.

10 Overlay

The intent of overlays is to add variation to a sequence at mathematically determined points of the sequence. Overlays may be interesting and useful to some people and over complex and not worth investing in to other people. Either is fine, music is whatever we want it to be.

10.1 Overview

Imagine two pieces of transparent paper one on top of the other. Each has a tic-tac-toe board and each represents only one move. From above we can see both the first move X and the next move O. By layering more transparencies we can fill every section of the tic-tac-toe grid.

The principle is the same here. We want to create new sequences by layering variations on top of the original sequence.

We determine when to apply these different layers with an overlay key. The first overlay key that we see is `1/1`. This is the bottom overlay or root overlay and it is applied on every iteration of the sequence.

An overlay key of `2/1` is applied every other cycle. An overlay key of `3/1` is applied every third cycle.

Overlay keys can be complex, be most of the time they will be used simply. Before describing the complexity, here are two tables that illustrate when each overlay key is applied in a simple system.

10.1.1 Every 1st, second, third and fourth iteration

Key Cycle	`1/1`	`2/1`	`3/1`	`4/1`
1	X			
2	X	X		
3	X		X	
4	X	X		X
5	X			
6	X	X	X	
7	X			
8	X	X		X

10.1.2 The first of four, the second of four, the third of four, the fourth of four

Key Cycle	`1/4`	`2/4`	`3/4`	`4/4`
1	X			
2		X		
3			X	
4				X
5	X			
6		X		
7			X	
8				X

10.2 Overlay Key Definition

The overlay key consists of four parts, two of which are initially visible.

The full key definition is:

```
1:1/1S1
```

The parts of which are:

```
1 - Shift - Segment of the interval to apply the overlay
: - Width Indicator
1 - Width - The duration of key cycles to apply the overlay once applied
/ - Interval Indicator
1 - Interval - The number of key cycles needed to reconsider application
S - Start Indicator
1 - Start - The number of key cycles to wait before allowing application of the o
```

Initially the overlay key is `1/1`. This is the root overlay and will be applied to the first key cycle of every 1st key cycle. More simply, this overlay will be applied every key cycle. If not using more specific overlays, this will always be applied and no understanding of the overlay key is needed.

An overlay key of `1/4` describe an iteration length of 4 and a shift of 1, or in other words, the first of every four. This will apply the overlay to the first of every four key cycles.

If we have four overlay keys for each segment of the interval: `1/4`, `2/4`, `3/4`, `4/4` then they would be applied according to the below table:

Key Cycle	`1/4`	`2/4`	`3/4`	`4/4`
1	X			
2		X		
3			X	
4				X
5	X			
6		X		
7			X	

Key Cycle	`1/4`	`2/4`	`3/4`	`4/4`
8				X

If the interval is only 1, with a shift of 2 as in `2/1`, then it will be applied every other cycle. A key of `4/2` is a fraction that gets reduced to `2/1` and will also play every other cycle. `3/1` is applied every third cycle. `4/1` is applied every fourth cycle and so on.

Key Cycle	`1/1`	`2/1`	`3/1`	`4/1`
1	X			
2	X	X		
3	X		X	
4	X	X		X
5	X			
6	X	X	X	
7	X			
8	X	X		X

For keys that are fractions not evenly divisible like `3/2` then the interval is doubled until it is greater than the shift. `3/2` is the same as `3/4`.

10.3 WIDTH

By default, an overlay key will only be applied for one cycle before we determine if it should be applied again. To apply the overlay for two cycles we can increase the width.

`1:1/1` is the same as `1/1`, the number after the `:` indicates the width. For simple keys this might be redundant. `2:2/1` will be applied every cycle the same as `1/1`. For other keys it has more meaning. `3:2/4` will be applied for the third and fourth out of every four cycles.

Key Cycle	`1:2/4`	`2:2/4`	`3:2/4`	`4:2/4`
1	X			
2	X	X		
3		X	X	
4			X	X
5	X			X
6	X	X		
7		X	X	
8			X	X

10.4 START

By default, an overlay key will be applied at the first cycle that matches. `3/4` will be applied at the third cycle. This is the same as `3/4S1`. The `S` indicates start. We can delay the application of the overlay by providing a start value that is higher than the first matching cycle. `3/4S4` will be applied at the 7th cycle because the program will not check if the key matches until the 4th cycle.

Key Cycle	`1:2/4S4`	`2:2/4S4`	`3:2/4S4`	`4:2/4S4`
1				
2				
3				
4				
5	X			
6	X	X		
7		X	X	
8			X	X

Another useful example is `2:1S8` which applies every other cycle but only after the 8th cycle.

10.5 Order And Stacking

Without any stacking options, only one overlay will be applied at any given time. If two overlays both match, like `1/1` and `2/1` on key cycle 2, then the overlay that matches less often will be on top and the overlay that matches the most will be on the bottom. In this case only `2/1` will be applied for key cycle 2.

The **stack** options of the overlay determine how that overlay relates to other overlays. Overlays can either **press up**, **press down** or **stand alone**. Cycle through stack options with `ctrl+u`

- **press up** - □ □ - Other overlays will be stacked on top of this overlay if they both match.
- **press down** - □ □ - This matching overlay will be stacked on top of the overlay underneath it, even if it doesn't match.
- **stand alone** - This overlay does not affect whether other overlays are applied.

By default the root overlay, `1/1`, always starts with the option **press up**. All other overlays will applied on top of the root overlay.

The **press down** option is useful when we want to aggregate the variations as we move through the cycles.

Key Cycle	`1/4□ □`	`2/4□ □`	`3/4□ □`	`4/4□ □`
1	X			
2	X	X		
3	X	X	X	
4	X	X	X	X
5	X			
6	X	X		
7	X	X	X	
8	X	X	X	X

As you can see, as we move through the cycles the **press down** option has an accumulative effect. With the **press down** option an overlay can be applied by an overlay directly above it if the above overlay has the **press down** option even if the overlay does not match.