Traqueur

User Manual

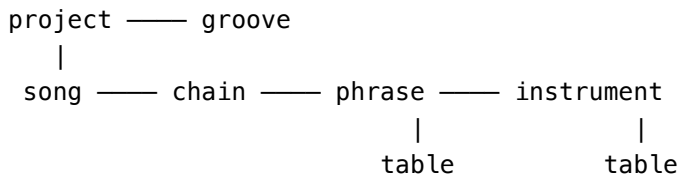
Version 1.3.d328eca May 7, 2026

Contents

Introduction

Traqueur is a music tracker and DAW optimised for portable and low-resource devices. It runs on macOS, Linux, Raspberry Pi, and Steam Deck. The interface is built around a hierarchy of nested objects, each with its own editing screen.

Data Structure



> **Note** The song grid has 8 channels and 256 rows. Up to 128 sample instruments and 16 MIDI instruments are available per project.

Default Key Mapping (Keyboard)

> **Note** These are the default mappings for desktop keyboard use. All keys can be overridden in `config.xml` (see Section 14). For gamepad or Steam Deck use, configure `mapping.xml`.

Button	Default Key
A	A
B	S
START	Space
LT	Right Ctrl
RT	Left Ctrl
LT+RT+SELECT	Esc — Project Selection / Quit

> **Note** LT and RT are intentionally swapped relative to GamePark hardware because the keyboard arrow keys are on the opposite side of the keyboard from the GP controller layout.

Basic Editing & Navigation

Keys	Action
ARROWs	In-screen navigation
A	Insert Chain / Phrase / Note
A, A	Insert next unused Chain / Phrase / Instrument
LT+(B, A)	Clone highlighted item into next unused slot
B+A	Cut highlighted item
A+UP / DOWN	Change value ±`0x10`
A+LEFT / RIGHT	Change value ±1
B+UP / DOWN	Song: page up/down; Chain: next/prev phrase; Instrument/Table: ±`0x10`
B+LEFT / RIGHT	Chain/Phrase: next/prev channel; Instrument/Table: ±1; Song: toggle Song/Live mode
RT+ARROWs	Navigate between screens
LT+UP / DOWN	Jump to next populated row past a blank row

> **Tip** **LT+UP/DOWN** is especially useful in Live Mode: jump directly to the next populated row and queue it without hunting through empty space.

Selections

Keys	Action
LT+B	Start selection (cursor cell only)
LT+B+B	Start selection (entire row)
LT+B+B+B	Start selection (entire screen)
ARROWs	Grow or shrink existing selection
B	Copy selection to buffer
LT+A	Cut selection *or* paste clipboard at cursor

Playback

Switch between Song and Live modes with **B+LEFT/RIGHT** in the Song screen.

Keys	Song Mode
START	Song screen: play/stop from highlighted row. Chain screen: solo play from current step. Phrase screen: solo play from **00**.
RT+START	Play from edited row in all screens.

Keys	Live Mode
START	Queue highlighted step. Queued items blink >.
START×2	Immediate-mode queue (triggers at end of current phrase). Fast-blinks >.
LT+START	Queue all channels on current row.
LT+START×2	Immediate-mode queue for all channels.
RT+START	Queue current channel to stop. Blinks _.
RT+START×2	Immediate-mode stop. Fast-blinks _.

Muting

Keys	Action
RT+B	Toggle mute. Release RT first = stays muted; release B first = reverts.
RT+A	Solo channel. Release RT first = stays solo'd; release A first = all revert.
LT+RT	Restore full playback on all channels.

> **Note** RT+A and RT+B respect active selections — the action applies to all channels present in the selection.

Starting from Scratch

Tables 1: Basics

Tables 2: Instrument Tables

Groove

Live Mode and Muting

Commands

Each phrase row can hold two commands. Commands affecting instruments can run on any step, including the step that triggers the instrument.

Commands marked ∗ may have platform-specific behaviour.

Command	Values	Description
ARPG	abcd	Cycle relative pitches a/b/c/d by semitone in a loop
CRSH∗	aabb	Pre-crush drive aa (00=bypass); bit depth bb (0x0=1 bit, 0xF=16 bit)
DLAY∗	--bb	Delay the note by bb ticks
FBMX	aabb	Approach feedback mix xxbb at speed aaxx
FBTN	aabb	Approach feedback tune xxbb at speed aaxx
FCUT	aabb	Adjust filter cutoff to bb at speed aa
FLTR	aabb	Lowpass: set absolute cutoff aa and resonance bb
FRES	aabb	Adjust filter resonance to bb at speed aa
HOP∗	aabb	Jump to step bb in next phrase; in tables: jump to row bb aa times then continue
IRTG	aabb	Retrigger instrument transposed bb semitones (cumulative)
KILL∗	--bb	Stop instrument after bb ticks
LEGA∗	aabb	Exponential pitch slide to bb at speed aa
LPOF	aaaa	Shift loop start+end by aaaa (values > 0x800 go backward)
MDCC	aabb	MIDI CC: control# aa, value bb
MDPG	--bb	MIDI program change (0000 = program 1)
PAN∗	aabb	Pan to bb at speed aa
PFIN	aabb	Pitch fine-tune: width bb at speed aa (00–7F up, FF–80 down)
PLOF	aabb	Play offset: jump to chunk aa or move ±bb chunks (256 chunks total)
PTCH∗	aabb	Linear pitch change: time aa, semitones bb
RTRG∗	aabb	Retrigger: advance loop aa ticks each cycle, loop length bb ticks
TABL∗	--bb	Trigger table bb
TMPO∗	--bb	Set tempo in hex (003C=60 BPM, 0190=400 BPM)
VOLM∗	aabb	Approach volume bb at speed aa (bb 00=silent; aa 00=instant)

Command Notes

ARPG Speed is fixed and cannot be changed. `ARPG 3000` loops original/+3st; `ARPG 4050` loops original/+4/+0/+5.

FCUT / FLTR `FCUT 0080` sets cutoff to 50% instantly; `FCUT 1000` closes filter at speed `10`. `FLTR 00FF` = unfiltered signal.

HOP Instant — instrument and commands on the same row still execute. No effect on instruments themselves.

IRTG The retriggered instrument is *not* reset: all running state (filter, volume, etc.) is preserved. Transpositions are cumulative. Useful for dubby echoes and non-4/4 timing without switching grooves.

LEGA vs. PTCH LEGA is exponential (equal speed across octaves); PTCH is linear. If pitch offset is 0 on a note row, a LEGA slide occurs automatically from the previous note.

LPOF Resets each time a new note triggers. Cannot itself trigger a note — a sample must already be playing.

RTRG `RTRG 0001` loops 1 tick; `RTRG 0102` loops 2 ticks advancing 1 each cycle; `RTRG 0101` is a no-op (offset equals loop length).

VOLM To create a swell: set the instrument volume to 0 and use VOLM to ramp up.

Configuration

Flags can be passed at the command line or stored permanently in `config.xml` (placed alongside the executable).

Command-Line Usage

```
./traqueur -OPTION=VALUE
# e.g.
./traqueur -FULLSCREEN=YES
```

config.xml Format

```
<CONFIG>
  <FULLSCREEN value='YES'/>
  <AUDIODRIVER value='Real'/>
  <AUDIOBUFFERSIZE value='512'/>
</CONFIG>
```

Config Reference

Key	Example	Notes
BACKGROUND	0F0F0F	Background colour (hex RGB)
FOREGROUND	E5E5B9	Foreground colour

Key	Example	Notes
HICOLOR1	35748B	Row-count colour in Song Screen
HICOLOR2	FF4138	Cursor colour
CURSORCOLOR	D0DC12	Cursor overlay colour
FULLSCREEN	YES	Start in fullscreen
SCREENMULT	2	Screen scale multiplier
KEY_A	j	A button
KEY_B	k	B button
KEY_LEFT	right ctrl	Left (LT) navigation button
KEY_RIGHT	left ctrl	Right (RT) navigation button
KEY_UP	w	Up arrow
KEY_DOWN	a	Down arrow
KEY_LSHOULDER	s	Left shoulder trigger
KEY_RSHOULDER	d	Right shoulder trigger
KEY_START	space	Start button
KEYDELAY	185	ms before key repeat begins
KEYREPEAT	40	ms between key repeats
ROOTFOLDER	/home/user	Project storage location
SAMPLELIB	root:	Sample library location
RENDER	FILERT	FILE, FILESPLIT, FILERT, FILESPLITRT
VOLUME	60	Hardware volume at startup (GP2X/Dingoo only)
DUMPEVENT	YES	Log all SDL events to file
AUDIOAPI	ASIO	W32: ASIO or MMSYSTEM (better MIDI timing)
AUDIODRIVER	Real	W32: first driver whose name starts with this string
AUDIOBUFFERSIZE	512	W32: increase if audio glitches
AUDIOPREBUFFERCOUNT	2	W32: blank buffers pre-queued before sequencer starts
MIDIDELAY	1	W32: MIDI output delay in ms

Key values use SDL_KeySym names. For USB controllers and Steam Deck button mapping, use `mapping.xml`.

> **Tip** For reliable MIDI sync on Windows, use `AUDIOAPI=MMSYSTEM`. Latency is slightly higher but timing stability is significantly better.

Rendering to File

Set `RENDER` to one of: `FILE` (render without playing back), `FILESPLIT` (render each channel to a separate file), `FILERT` (render while playing), or `FILESPLITRT` (render each channel while playing).

Custom Font

Create a `.bmp` font using the original font as a template (`sources/Resources/original.bmp`), then run `sources/Resources/mkfont.py` to generate an array to paste into the source.