



# Interpretations: An Album Written in Python

APRIL 2026

4 min read • 810 words

---

I'm working on an album. Each track is a Python script. You run it, it renders a WAV file. That's the whole workflow.

The project is called **Interpretations**, and it's built on [PyTheory](#) — the same synthesis engine I've been writing about [lately](#). No DAW. No MIDI. No samples. Just Python files that describe music and a library that turns them into sound.

uv run play

INTER

↑/↓ navigate ←play r

▶	1. RAGA MIDNIGHT	✓	90
	2. SHRUTI LOFI	✓	75
	3. GHOST PROTOCOL	✓	128
	4. SILK ROAD	✓	95
	5. THE OBSERVATORY	✓	112
	6. ACID REIGN	✓	140
	7. BEAST MODE	✓	135
	8. APEX	✓	140
	9. VOLTAGE	✓	138
	10. AN EXCEPTION OCCURRED	✓	80
	11. VOICES	✓	65
	12. INTRUSIVE	✓	92
	13. GRAVITY	✓	88
	14. THE INTERRUPTION	✓	85
	15. SLEIGHT OF HAND	✓	100
	16. WAVEFORMS	✓	118
	17. EMERGENCE	✓	100
	18. CHAKRA	✓	60

<https://interpretations.kennethreitz.org>

It's very much a work in progress, but the concept is solid enough to talk about.

## The Idea

What if a song was a script? Not a MIDI file, not a project folder full of plugin states and automation lanes — a single `.py` file that contains every musical decision: the key, the scale, the tuning system, the instruments, the notes, the effects, the mix. You read the code and you read the music. You run the code and you hear it.

```
python tracks/raga_midnight.py
```

That's it. Out comes a fully rendered, mixed, stereo WAV file. Every note synthesized from scratch. No prerecorded anything.

I like this because it makes music legible in a way that DAW projects aren't. A DAW session is a sprawl of tracks, plugins, automation curves, and hidden state. A Python script is sequential. You can read it top to bottom and understand what happens and when. It's composition as prose.

## What's on It

Nine tracks so far, all over the map stylistically. That's kind of the point — I wanted to see how far the concept could stretch.

**Raga Midnight** is a proper Indian classical piece in D Phrygian with shruti just intonation — tabla solo with a tihai cadence, sitar, tambura drone. **Silk Road** is a caravan that picks up musicians along the ancient trade route: koto in China, sitar and tabla in India, mandolin and doumbek in Persia, guitar and cajon in the Mediterranean. Nobody leaves. Everyone adds their voice.

**The Interruption** is my favorite conceptually — a baroque harpsichord and string quartet playing something lovely, and then at bar 33 a drum and bass breakbeat slams in with sub bass and a reese. The strings keep playing like nothing happened. The beat eventually dissolves. The quartet wins.

**Ghost Protocol** withholds its kick drum until bar 49. **Acid Reign** has dual 303 acid lines with the resonance cranked to 20. **Deep Time** is seven and a half minutes of ambient drone at 40 BPM in just intonation. **Chakra** modulates tempo upward through seven frequency centers at 432 Hz. **Culture Clash** has — and I'm not making this up — a didgeridoo drone, NES Mario theme, Hotline Bling on steel drum, and a DCI marching cadence. In the same track.

It's a lot. I'm having fun.

## How It Works

Each track imports PyTheory's `Score`, `Key`, `Duration`, `DrumSound`, and other building blocks. You define a key and scale, create a score with a tempo and time signature, add parts for each instrument, and write the notes. Then `play_score()` renders everything to audio.

The instruments are all synthesized — [Karplus-Strong for plucked strings](#), physical modeling for drums, additive synthesis for organs, subtractive for synths. Effects chains handle reverb, delay, distortion, filtering, sidechain compression. It's a [mini DAW in the REPL](#), except here the REPL is just a script you run.

The tuning system stuff is where it gets interesting. Several tracks use shruti just intonation — the Indian classical tuning system with pure harmonic ratios instead of the compromises of equal temperament. When the tambura drone in Raga Midnight hits a perfect fifth, you can hear the difference. The overtones line up in a way that equal temperament can't quite reach.

## Why Python

Honestly? Because it was there. I had [PyTheory](#), I had a bunch of synthesis engines, and I wanted to write music with them. The alternative was building a GUI, and I'd rather write songs.

There's something appealing about the constraint, too. A DAW gives you infinite undo, infinite tracks, infinite plugins. A Python script gives you what you write. The composition is the code. If you want to change the reverb on the sitar, you

change a line and re-render. It's slow compared to tweaking a knob in Ableton, but the tradeoff is total reproducibility — anyone with PyTheory installed can run the script and get the exact same output.

## Work in Progress

This is nowhere near done. The mixes need work. Some tracks need structural editing. I'm still figuring out the right balance between writing every note by hand and using PyTheory's pattern tools to generate parts. But the concept — albums as repositories, songs as scripts, music as code you can read — feels like something worth exploring.

The [official site is live](#), and the repo is [on GitHub](#) if you want to poke around. Run a track. Read the code. Tell me what you think.

---

**Update:** the album is now released! Listen on [Spotify](#) or [Apple Music](#).

---

Generated from [kennethreitz.org](#) • 2026