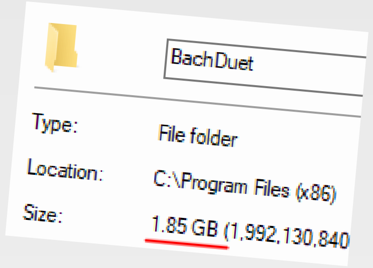
Euterpe: A Web Framework for Interactive Music Systems

Yongyi Zhang*, Christodoulos Benetatos*, Zhiyao Duan

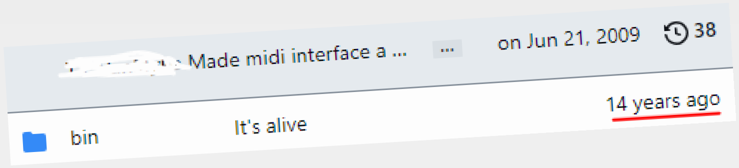
JAES 2023

Problem Statement

- Research **stops** at **open sourcing** the core algorithms
- Prototype systems that are **not easily accessible**
  - Large executable files
  - Unmaintained codebases
  - Platform dependent implementations
  - Complicated installation processes

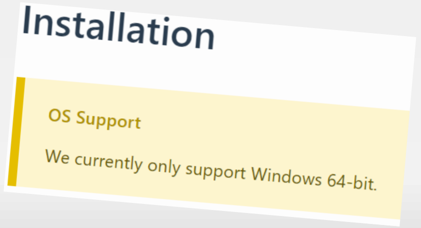


Type:	File folder
Location:	C:\Program Files (x86)
Size:	1.85 GB (1,992,130,840



Made midi interface a ... on Jun 21, 2009 38

bin It's alive 14 years ago



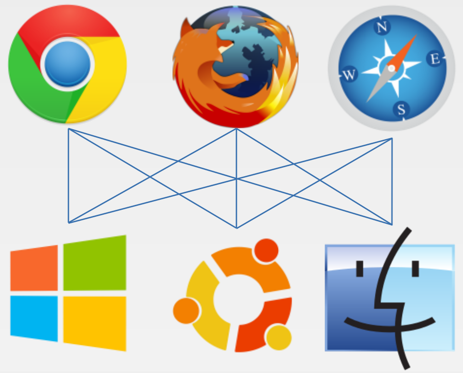
Installation

OS Support

We currently only support Windows 64-bit.

A Solution …

- Promote the development of **web** musical systems
  - Pros
    - Utilize the web’s natural cross-platform compatibility
    - End-users are familiar with the browser environment
    - No installation required



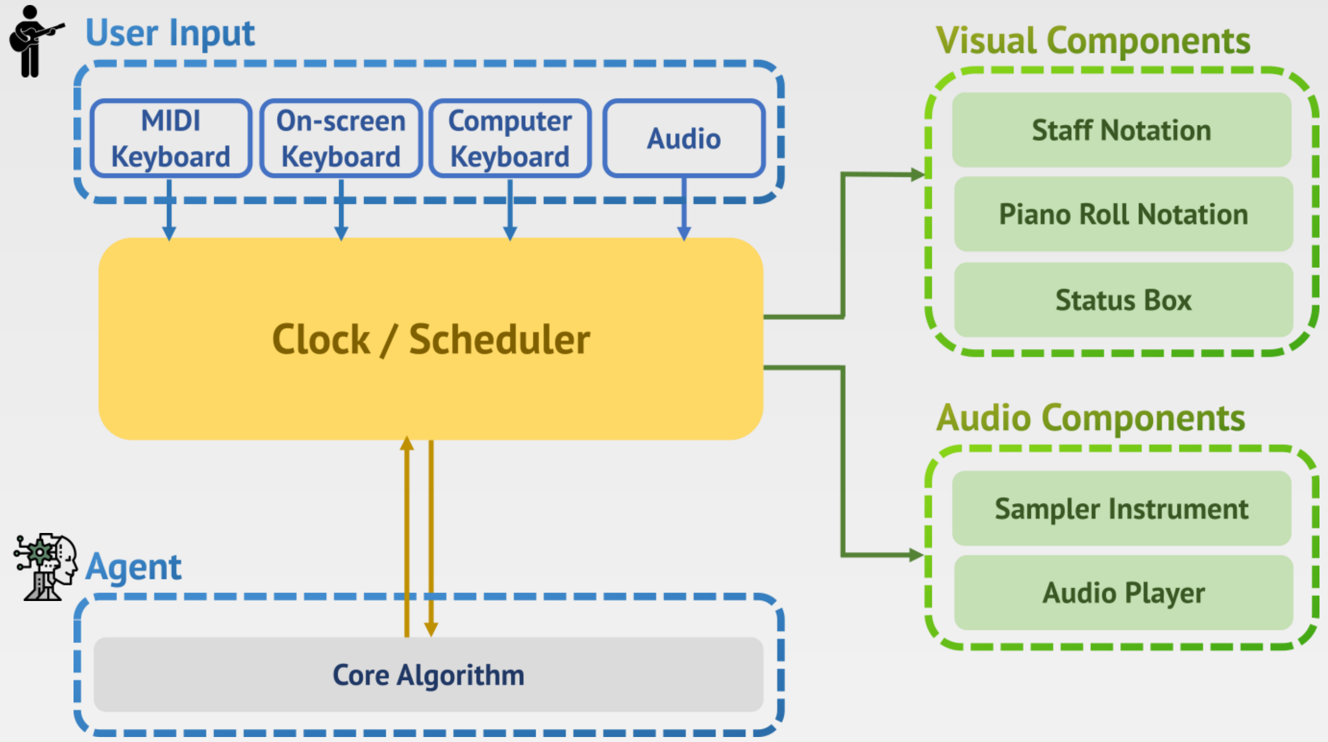
A Solution …

- Promote the development of **web** musical systems
  - Cons
    - Knowledge of web programming is required (JavaScript, CSS, HTML)

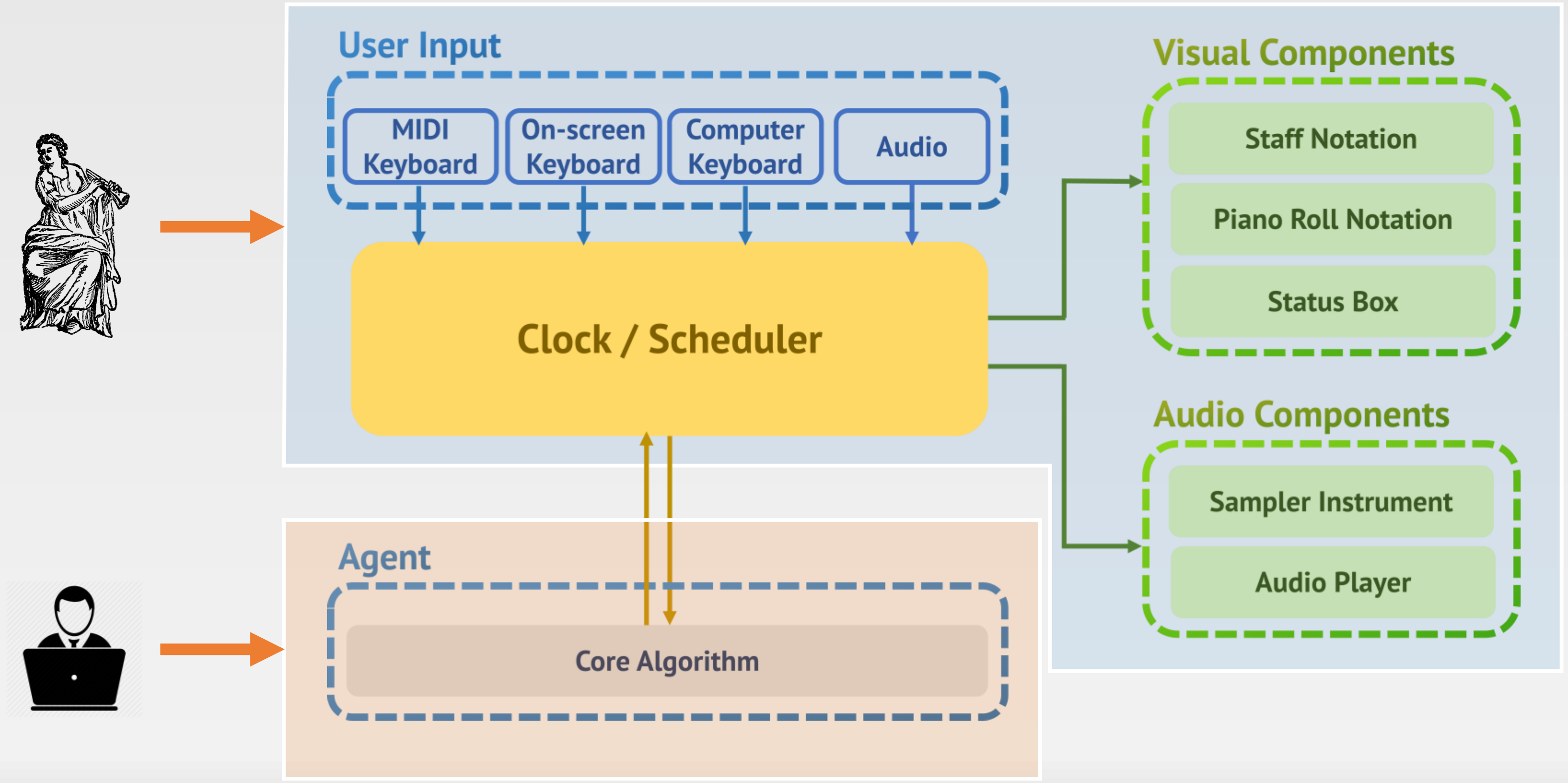
Euterpe’s Goal

- **Alleviate challenges** associated with **web programming**
    - Offer ready-made submodules for common system components
    - Developers focus solely on their system's unique features

Generic IMS Architecture



Generic IMS Architecture

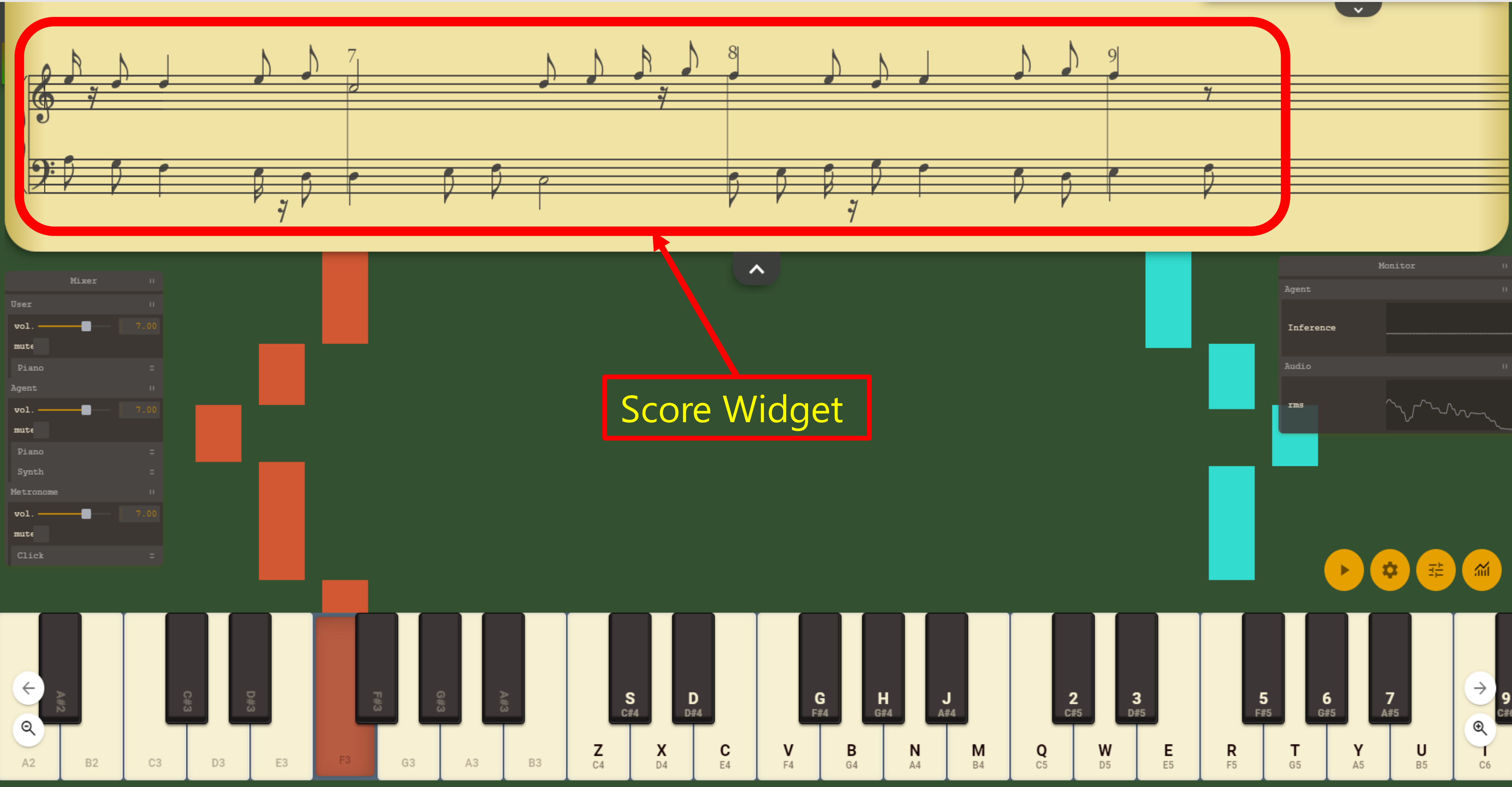


Design

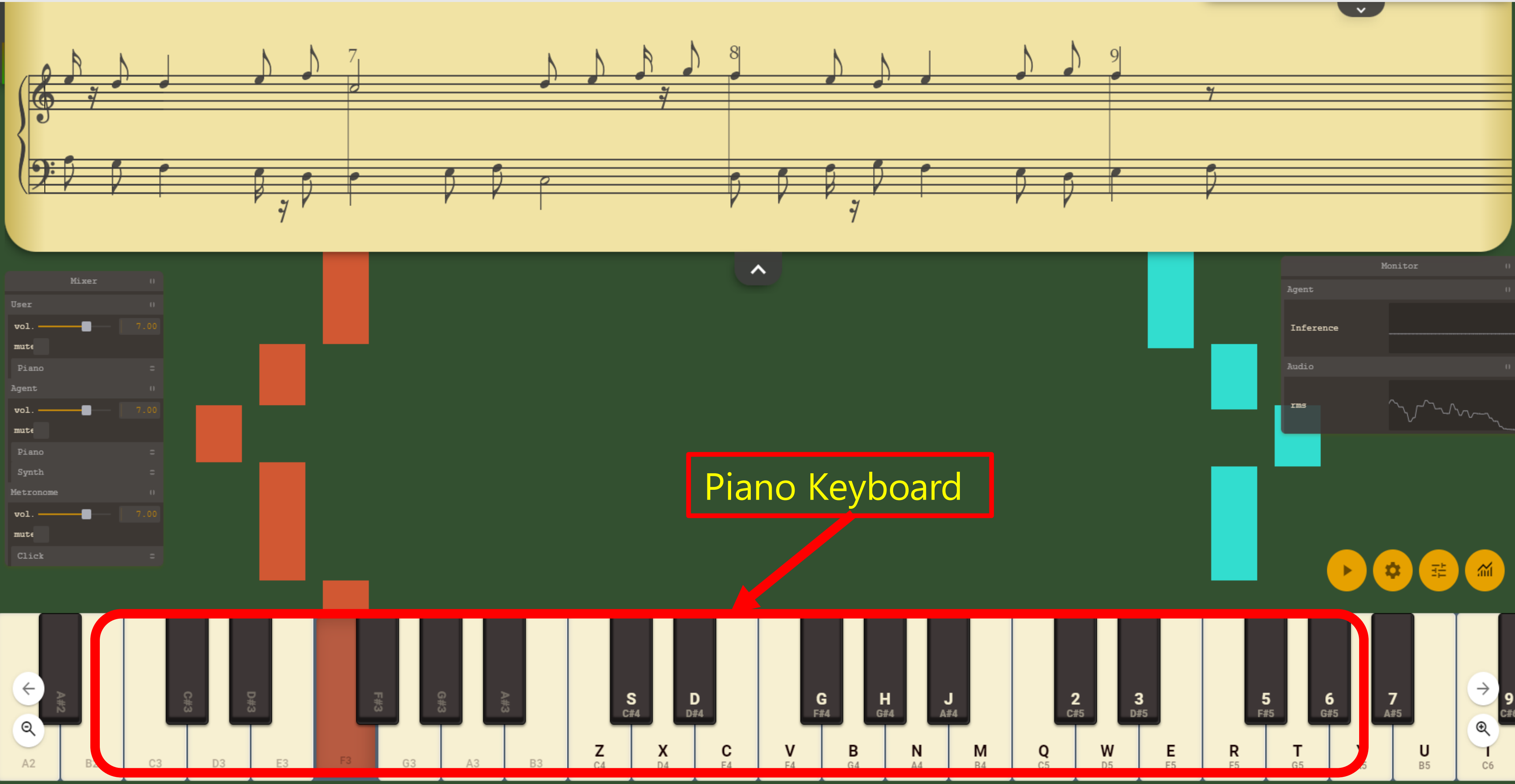
- Modular
  - **Separate** the **Agent code** from the peripheral components

- Configuration files
  - Allow app setup and **customization without** writing **JavaScript**

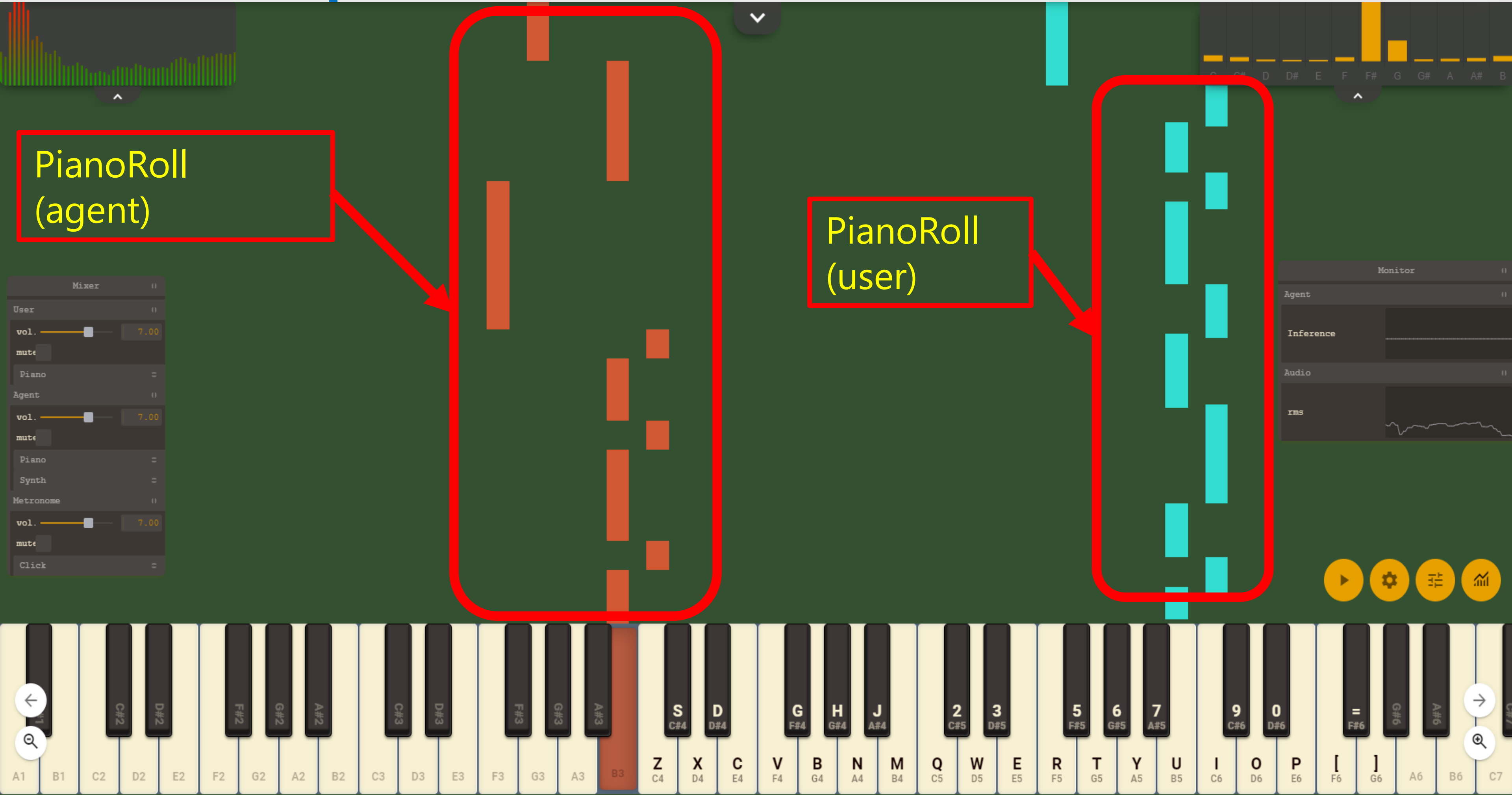
Visual Components



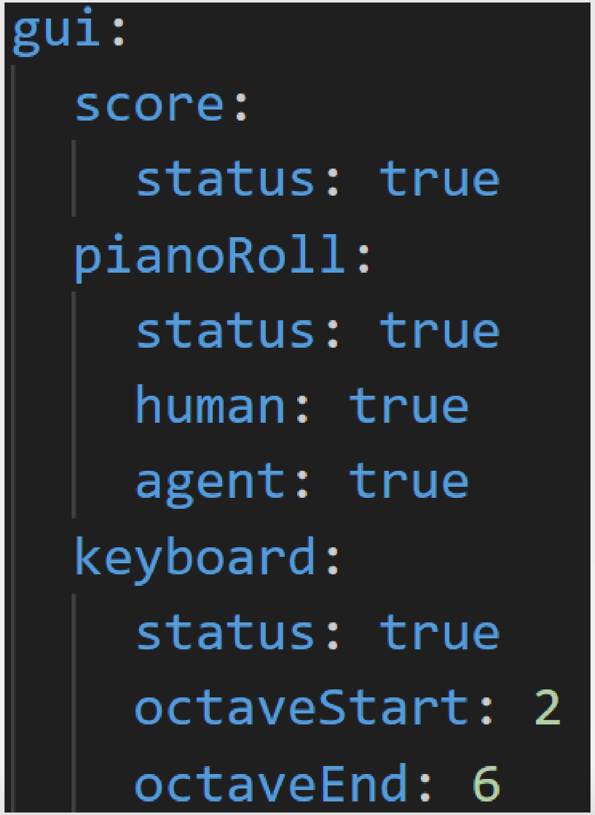
Visual Components



Visual Components

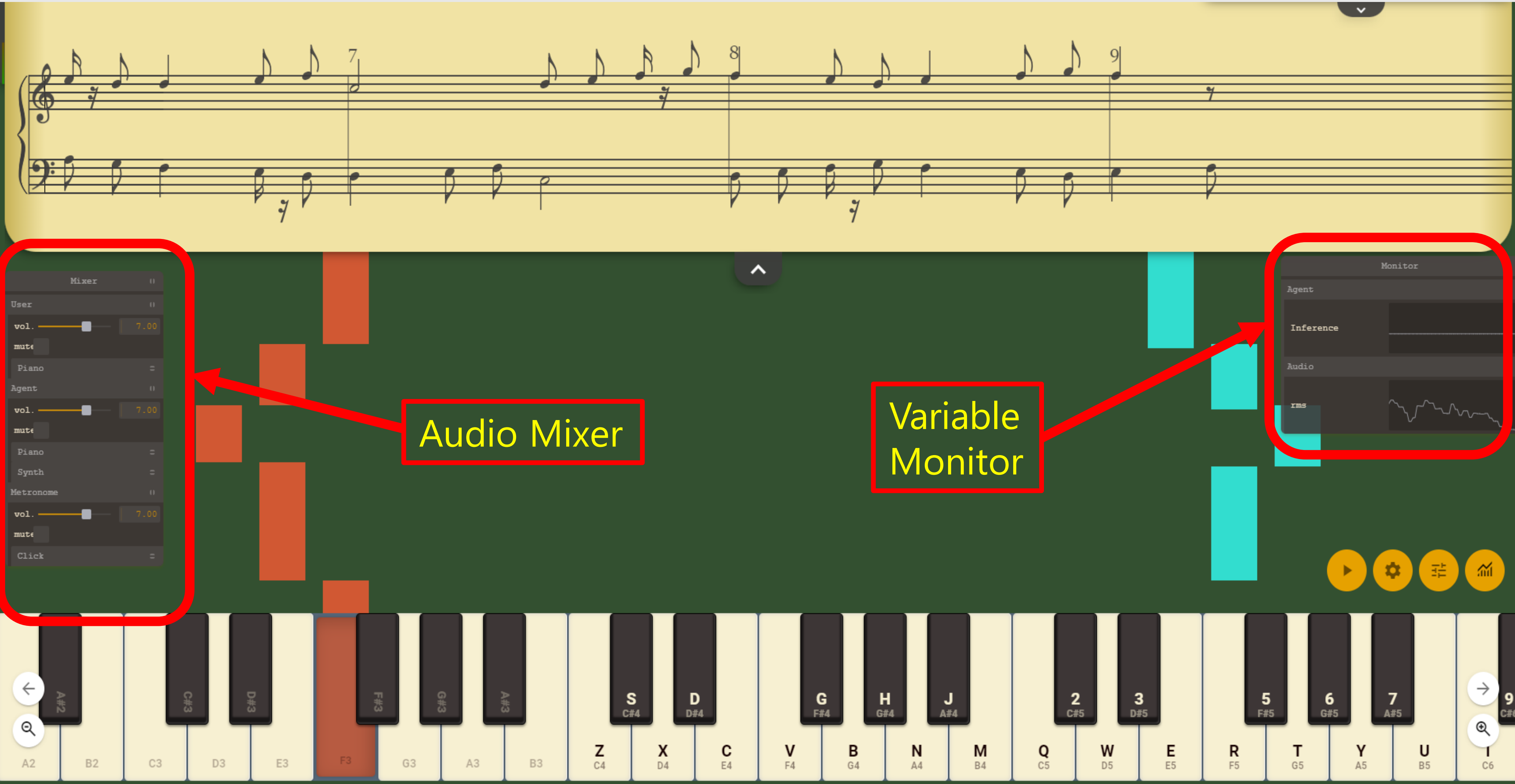


Configuration

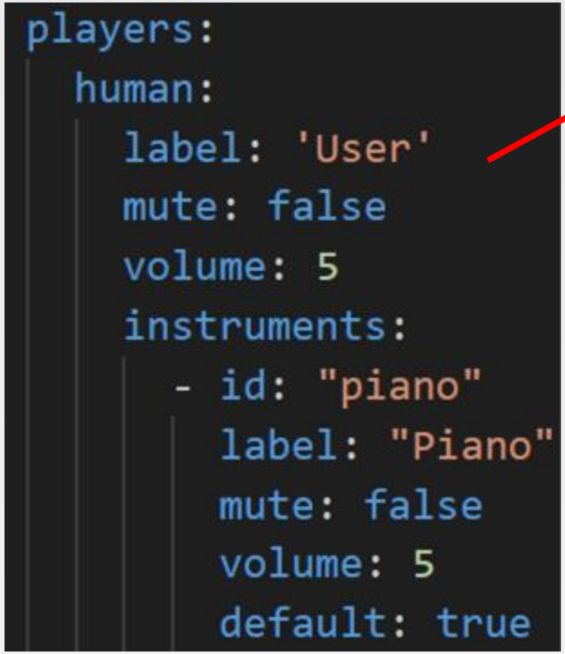


```yaml
gui:
  score:
    status: true
  pianoRoll:
    status: true
    human: true
    agent: true
  keyboard:
    status: true
    octaveStart: 2
    octaveEnd: 6
```

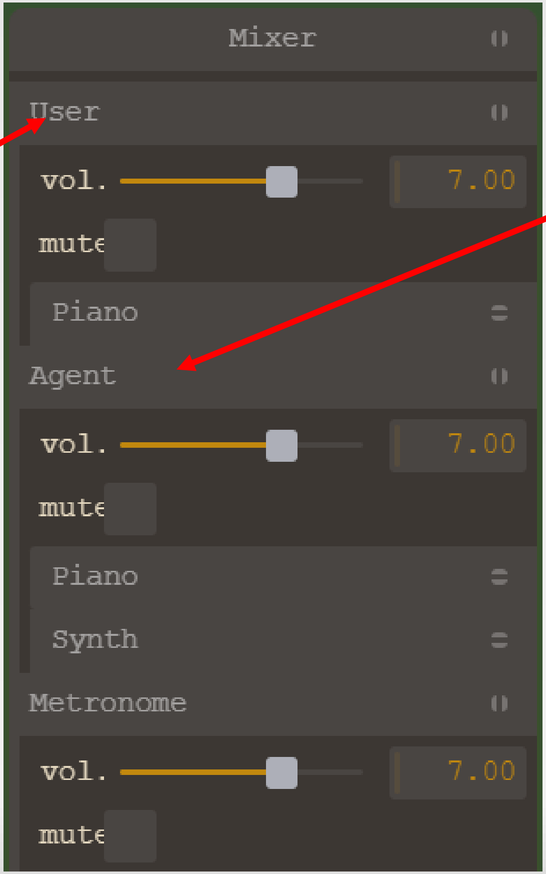
Visual Components



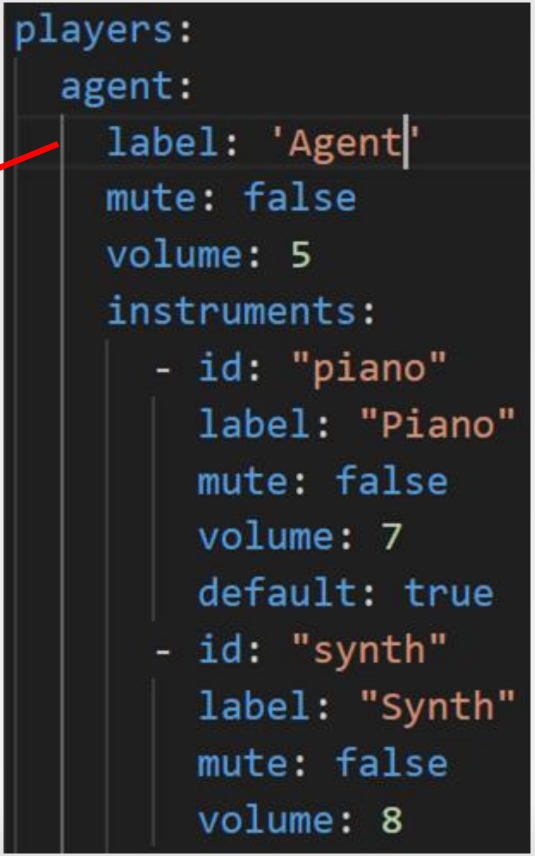
Configuration



```yaml
players:
  human:
    label: 'User'
    mute: false
    volume: 5
    instruments:
      - id: "piano"
        label: "Piano"
        mute: false
        volume: 5
        default: true
```

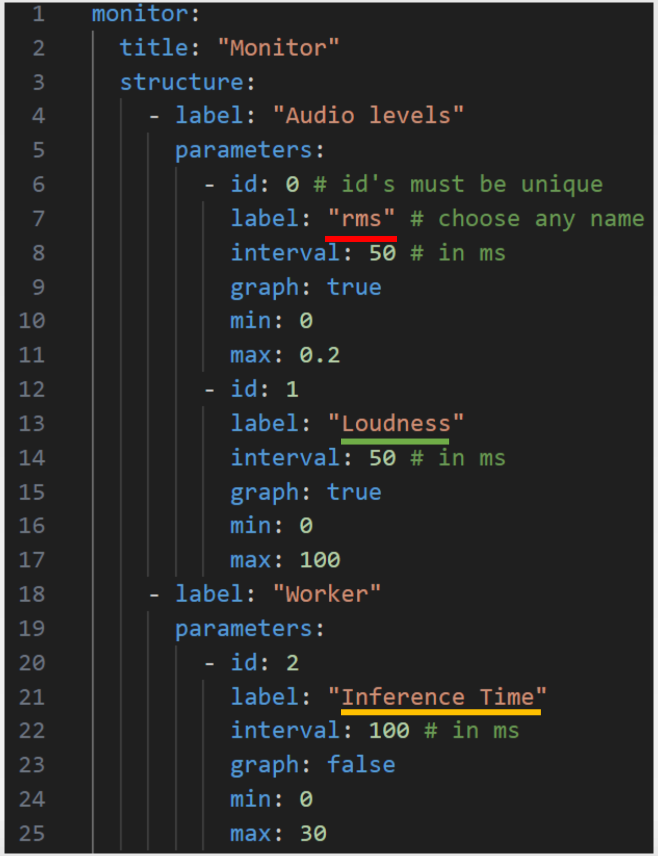


```
Mixer
User
vol.      7.00
mute
Piano
Agent
vol.      7.00
mute
Piano
Synth
Metronome
vol.      7.00
mute
```

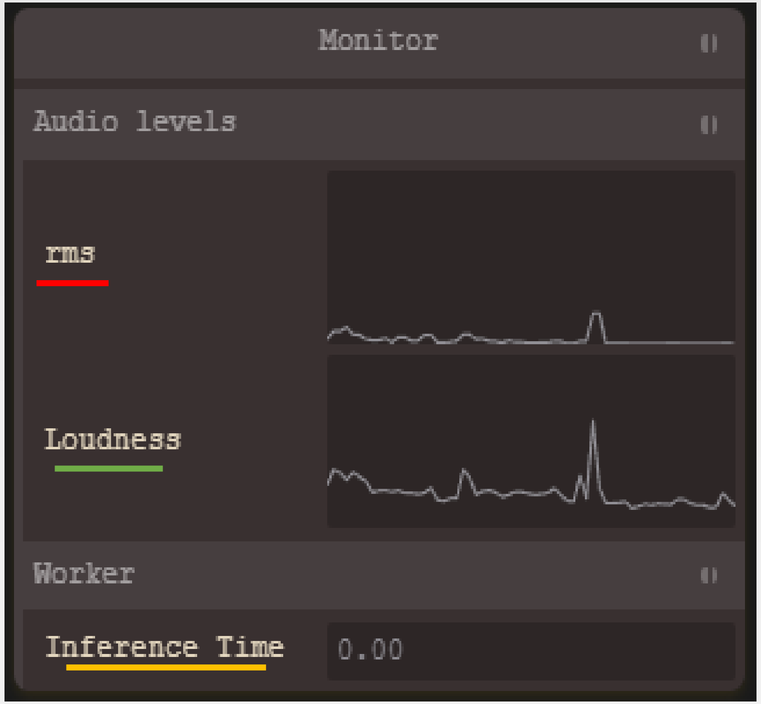


```yaml
players:
  agent:
    label: 'Agent'
    mute: false
    volume: 5
    instruments:
      - id: "piano"
        label: "Piano"
        mute: false
        volume: 7
        default: true
      - id: "synth"
        label: "Synth"
        mute: false
        volume: 8
```

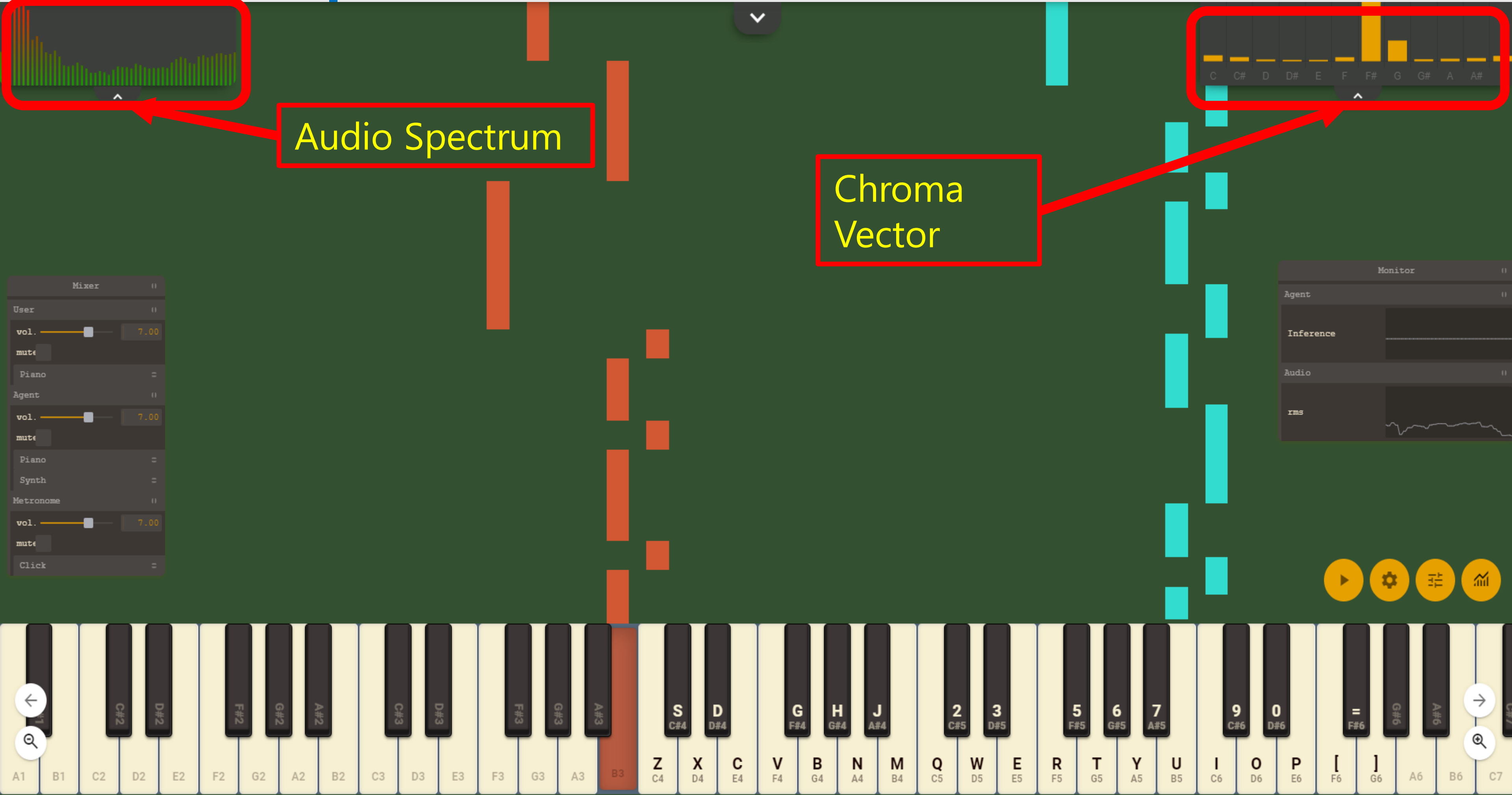
Configuration



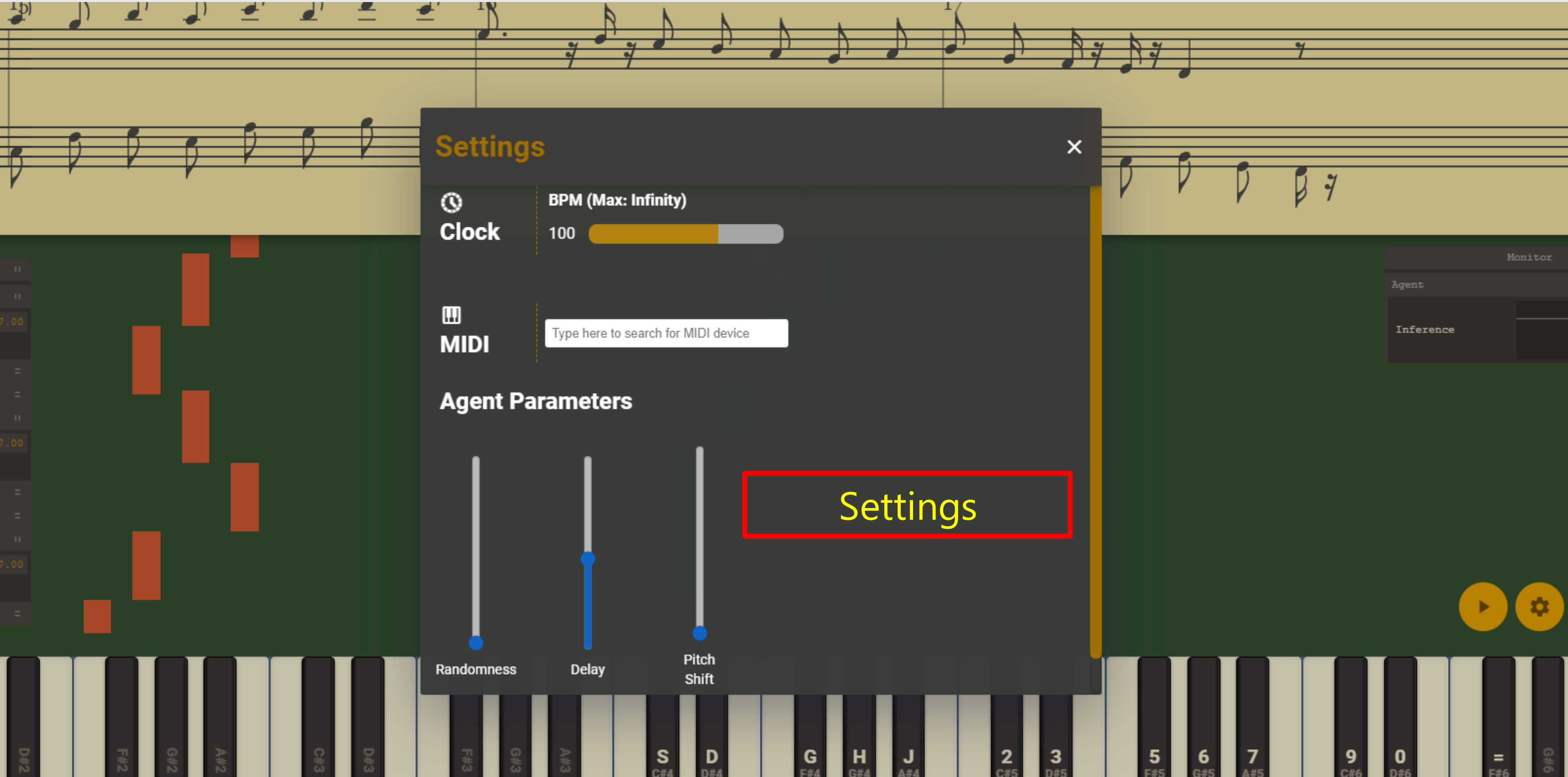
```yaml
monitor:
  title: "Monitor"
  structure:
    - label: "Audio levels"
      parameters:
        - id: 0 # id's must be unique
          label: "rms" # choose any name
          interval: 50 # in ms
          graph: true
          min: 0
          max: 0.2
        - id: 1
          label: "Loudness"
          interval: 50 # in ms
          graph: true
          min: 0
          max: 100
    - label: "Worker"
      parameters:
        - id: 2
          label: "Inference Time"
          interval: 100 # in ms
          graph: false
          min: 0
          max: 30
```



Visual Components


Audio Spectrum
Chroma Vector
C C# D D# E F F# G G# A A#
Mixer
User
vol. 7.00
mute
Piano
Agent
vol. 7.00
mute
Piano
Synth
Metronome
vol. 7.00
mute
Click
Monitor
Agent
Inference
Audio
rms
A1 B1 C2 D2 E2 F2 G2 A2 B2 C3 D3 E3 F3 G3 A3 B3
Z C4 X D4 C E4 V F4 B G4 N A4 M B4 Q C5 W D5 E E5 R F5 T G5 Y A5 U B5 I C6 O D6 P E6 [F6] G6 A6 B6 C7
S C#4 D D#4 G F#4 H G#4 J A#4 2 C#5 3 D#5 5 F#5 6 G#5 7 A#5 9 C#6 0 D#6 = F#6

Visual Components



Settings

Clock

BPM (Max: Infinity)

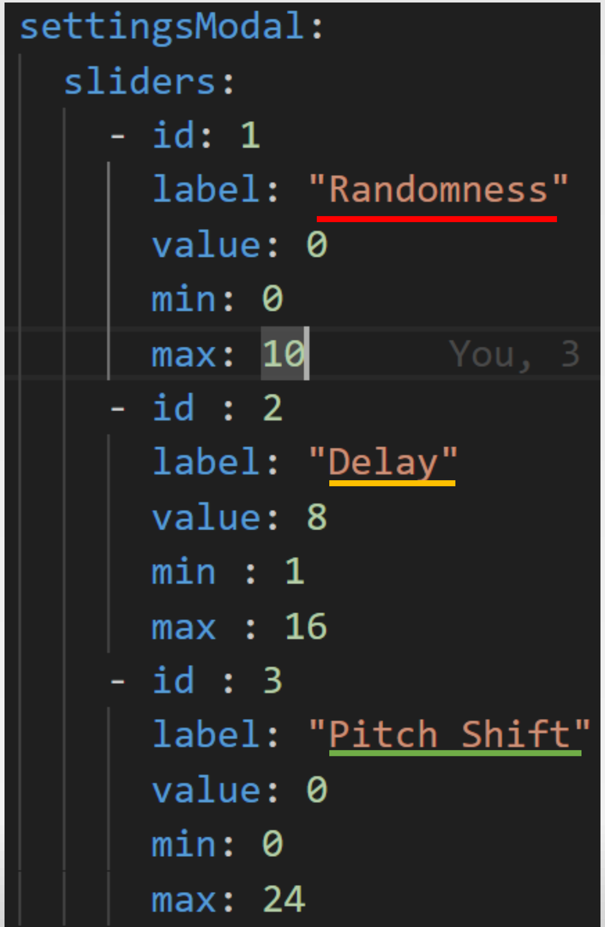
100

MIDI

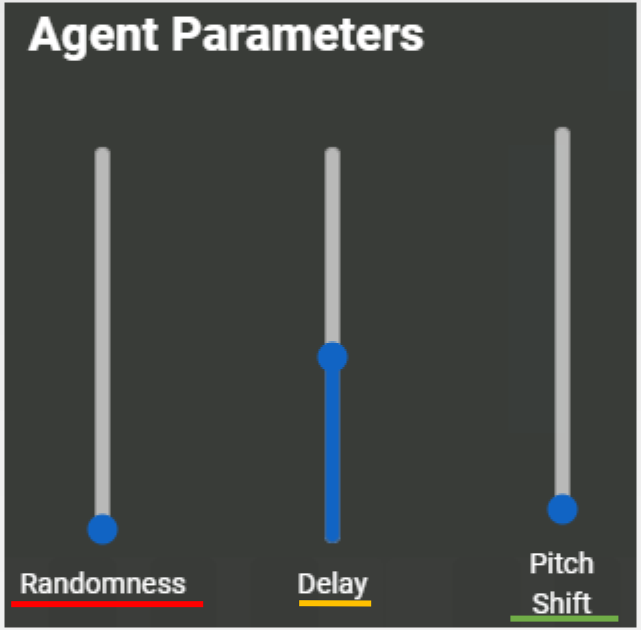
Type here to search for MIDI device

Agent Parameters

Configuration

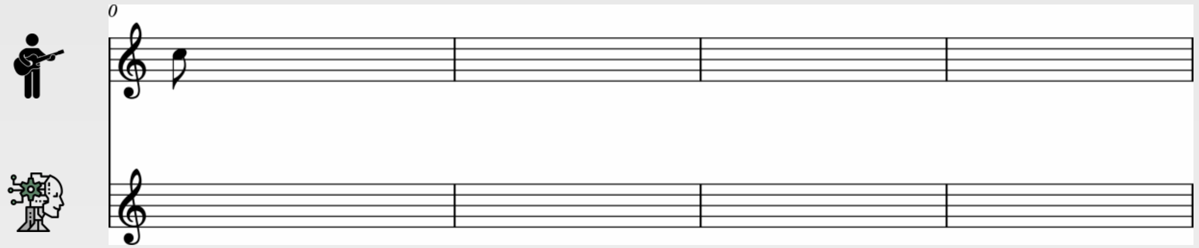


```
settingsModal:
  sliders:
    - id: 1
      label: "Randomness"
      value: 0
      min: 0
      max: 10          You, 3
    - id : 2
      label: "Delay"
      value: 8
      min : 1
      max : 16
    - id : 3
      label: "Pitch Shift"
      value: 0
      min: 0
      max: 24
```



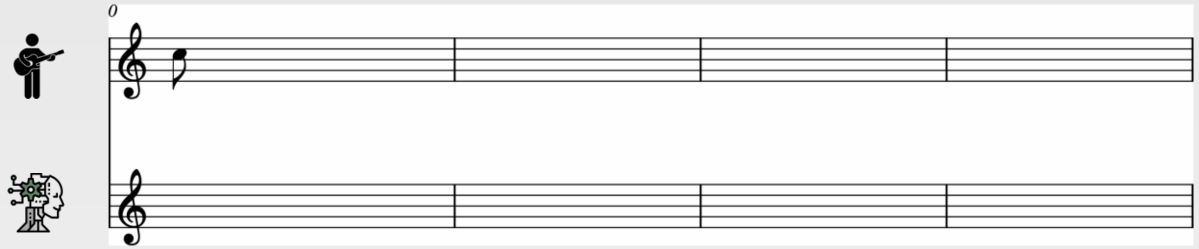
Paradigms of Music Interaction

- Call & Response

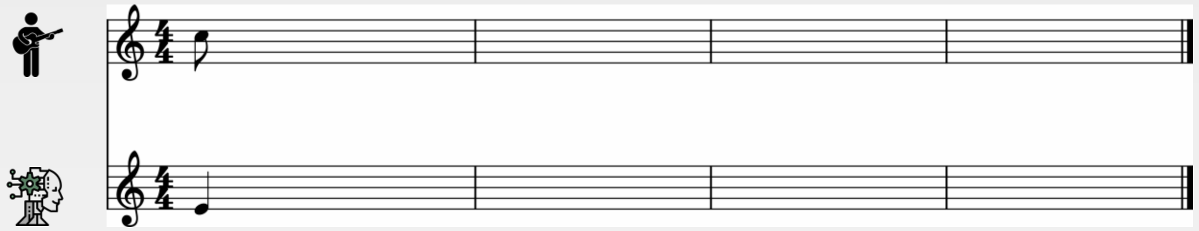


Paradigms of Music Interaction

- Call & Response

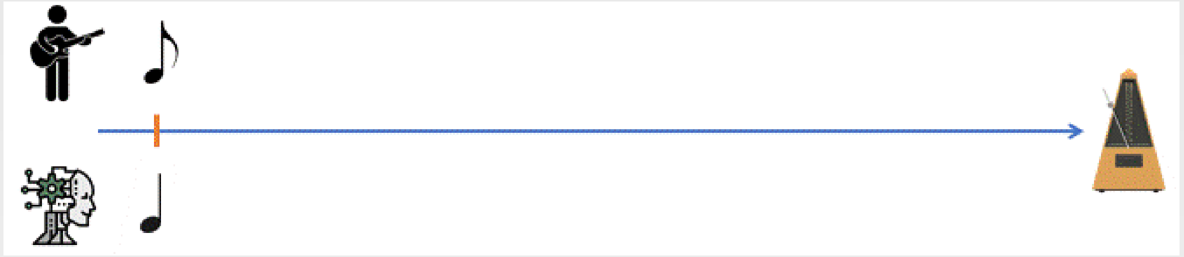


- Simultaneous



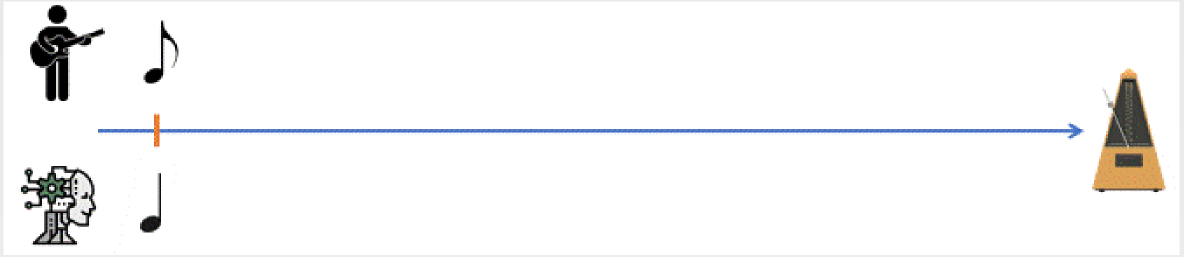
Paradigms of Music Interaction

- Grid-based

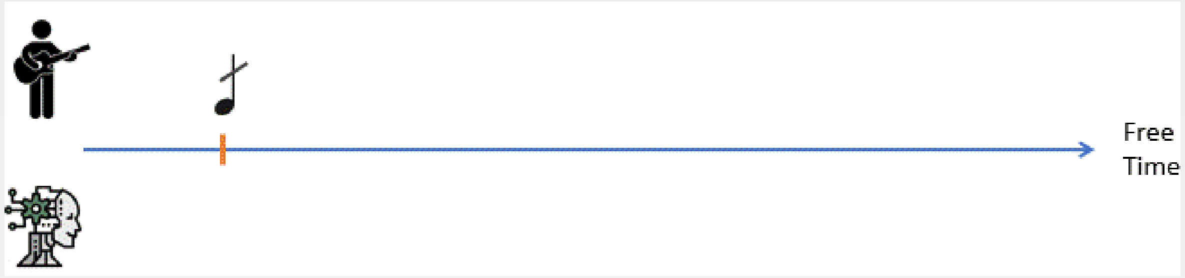


Paradigms of Music Interaction

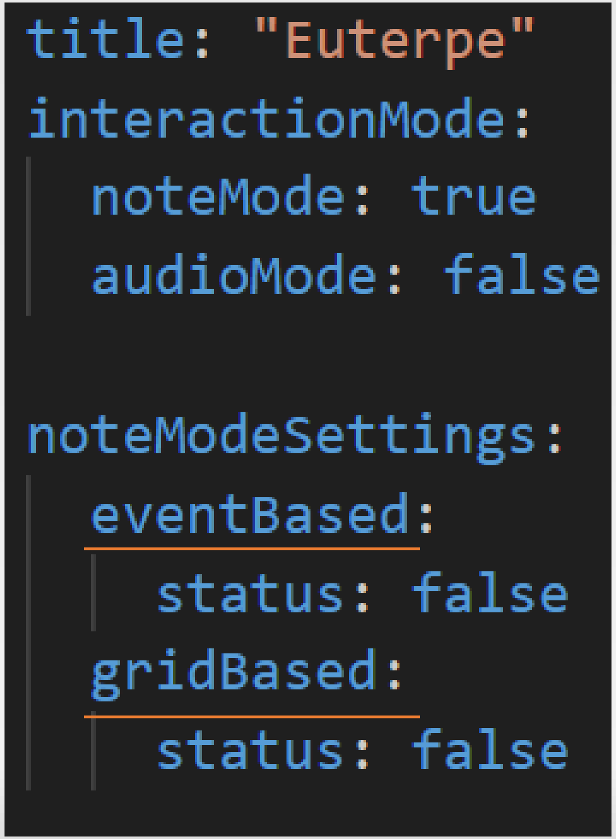
- Grid-based



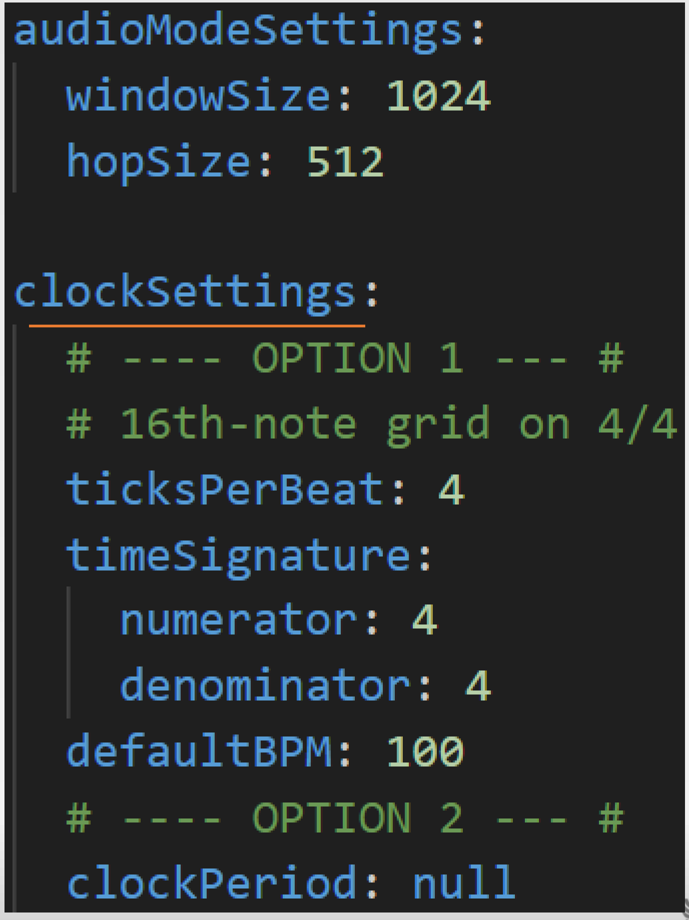
- Event-based



Configuration



```yaml
title: "Euterpe"
interactionMode:
  noteMode: true
  audioMode: false

noteModeSettings:
  eventBased:
    status: false
  gridBased:
    status: false
```



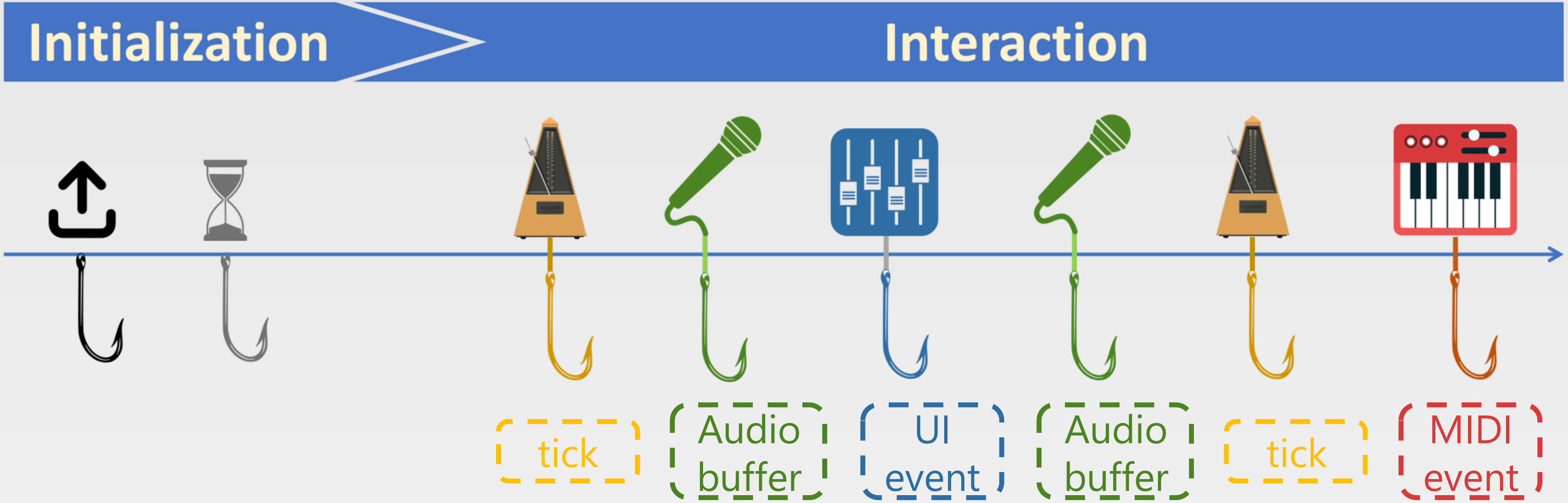
```yaml
audioModeSettings:
  windowSize: 1024
  hopSize: 512

clockSettings:
  # ---- OPTION 1 --- #
  # 16th-note grid on 4/4
  ticksPerBeat: 4
  timeSignature:
    numerator: 4
    denominator: 4
  defaultBPM: 100
  # ---- OPTION 2 --- #
  clockPeriod: null
```

Agent

- Provides 6 hook functions
  - **Empty** functions to be filled.
    - Invoked **automatically** at specific events or stages within the interaction
    - Can be activated/deactivated from the configuration file

```
function hook(event){
    // your code
}
```

Euterpe Lifecycle



Agent - Hooks

- loadExternalFiles()
    - Load external resources useful for the Agent

- loadAlgorithm()
    - Core algorithm initialization
    - Checkpoint fetching
    - NN model loading
    - warmup NN

Agent - Hooks

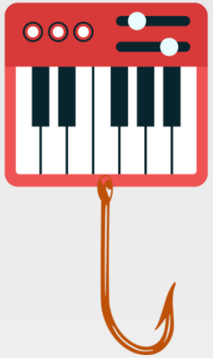
- updateParameter(id, value)
  - Invoked when the user interacts with the GUI (buttons, sliders etc.)
  - The Agent’s hyper-parameters are updated

Agent - Hooks

- processClockEvent(tick)
  - Invoked periodically based on the Clock’s “tick”
  - Used on a time-grid based interaction

Agent - Hooks

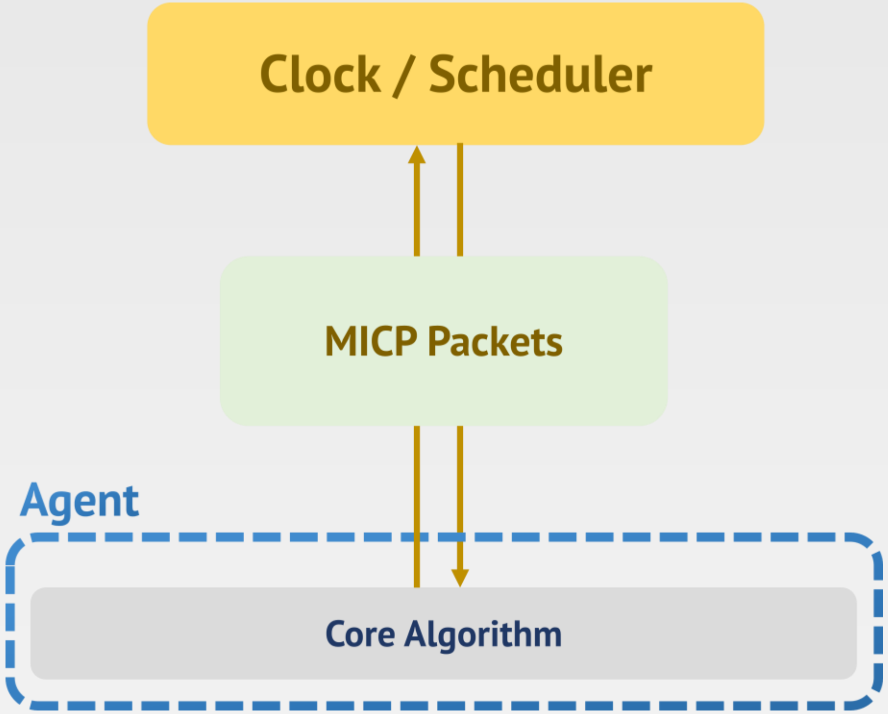
- processNoteEvent(event)
  - Invoked when a MIDI note is received
  - Used in “event-based” mode



Agent - Hooks

- processAudioBuffer(buffer)
  - Invoked when a new audio buffer is available
  - Every hopSize samples

Music Interaction Communication Protocol - MICP



MICP – NoteEvent

• **player**	: Agent or User
• **instrument**	: Which sampler instrument to use for playback
• **device**	: The user's input device (i.e MIDI keyboard)
• **type**	: Note_On, Note_Off or Note_Hold
• **name/midi/chroma**	: Info about the note (i.e C4, 60, 0)
• **channel/velocity**	: Midi specific info
• **createdAt** (tick, seconds)	: When was this note created/generated (timestamp)
• **playAfter** (tick, seconds)	: Play the note with a delay
• **duration**	: The duration of the note (optional)

Coding Session

Online Guide :

https://xribene.github.io/