Computational Music Mixing with EarSketch: Engaging Students With Python and Music

ISTE 2015: Session B318
Tuesday, June 30th
Mr. Christopher Michaud
michaudc@marist.com
www.nebomusic.net
Marist School / Georgia Institute of Technology
Who am I?

- Temple University
- Georgia Tech
- George Mason University
- Marist University
Websites for Presentation

nebomusic.net

earsketch.gatech.edu/earsketch2
Why Music and Programming?

• *A Philosophy of Music Education*: Bennett Reimer

Music allows humans to encode, refine, reflect, and share non-verbal thought in a dynamic moving system that changes over time.
What is Programming?

• Allows humans to encode, refine, reflect, and share logical thought in a dynamic moving system that changes over time.

• Music and Programming (Computational Thinking) are the same Human Skill expressed in two different domains: non-verbal feelings in art/music and logical / motion thought in Programming or Computational Thinking
Another Book . . .

Learning Sequences in Music: Edwin Gordon

Concept of ‘Audiation’ - The inner thought of music.

Table 2. Types of Audiation (Gordon, 2007b, p. 15).

<table>
<thead>
<tr>
<th>Type</th>
<th>Activity</th>
<th>Music type</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Listening to</td>
<td>familiar or unfamiliar music</td>
</tr>
<tr>
<td>2</td>
<td>Reading</td>
<td>familiar or unfamiliar music</td>
</tr>
<tr>
<td>3</td>
<td>Writing</td>
<td>familiar or unfamiliar music from dictation</td>
</tr>
<tr>
<td>4</td>
<td>Recalling and performing</td>
<td>familiar music from memory</td>
</tr>
<tr>
<td>5</td>
<td>Recalling and writing</td>
<td>familiar music from memory</td>
</tr>
<tr>
<td>6</td>
<td>Creating and improvising</td>
<td>unfamiliar music while performing or in silence</td>
</tr>
<tr>
<td>7</td>
<td>Creating and improvising</td>
<td>unfamiliar music while reading</td>
</tr>
<tr>
<td>8</td>
<td>Creating and improvising</td>
<td>unfamiliar music while writing</td>
</tr>
</tbody>
</table>

Table 3. Stages of Audiation (Gordon, 2007b, p. 20).

<table>
<thead>
<tr>
<th>Stage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Momentary retention</td>
</tr>
<tr>
<td>2</td>
<td>Imitating and audiating tonal patterns and rhythm patterns and recognizing and identifying a tonal center and macrobeats</td>
</tr>
<tr>
<td>3</td>
<td>Establishing objective or subjective tonality and meter</td>
</tr>
<tr>
<td>4</td>
<td>Retaining in audiation tonal patterns and rhythm patterns that have been organized</td>
</tr>
<tr>
<td>5</td>
<td>Recalling tonal patterns and rhythm patterns organized and audiated in other pieces of music</td>
</tr>
<tr>
<td>6</td>
<td>Anticipating and predicting tonal patterns and rhythm patterns</td>
</tr>
</tbody>
</table>
Audiation is to music as Computational Thinking is to Computer Science

Non-Verbal Feelings: Music

Thought: Programming / Computational

Motion / Time / Flow Thinking
To put this another way . . .

Eine Kleine

Kalman Filter

Time Update (prediction)

1. Compute the Kalman Gain
   \[ K_k = P^-_k H^T (HP^-_k H^T + R)^{-1} \]
2. Update the estimate via \( z_k \)
   \[ \hat{x}_k = \hat{x}^-_k + K_k (z_k - H\hat{x}^-_k) \]
3. Update the error covariance
   \[ P_k = (I - K_k H)P^-_k \]

Measurement Update (correction)

Initial estimates at \( K = 0 \)
The outputs at \( K \) will be the input for \( K+1 \)

Encoding the Non-Verbal

 Encoding the Logic
Motivation

• Large population of students through which music is the primary vehicle for personal expression.

• In CS – we tend to focus on:
  • Graphics / Games / Visual
  • Text Processing / Database
  • Logic / Algorithmic / Mathematical

• EarSketch seeks to engage students in CS through the art of music.
Why Learn to Program?

What’s wrong with this picture? Share these stats.

1,000,000 more jobs than students by 2020

1.4 million computing jobs

$500 billion opportunity

400,000 computer science students


Computer science is a top paying college degree and computer programming jobs are growing at 2X the national average.
Why Learn to Program?

Choose a State

Georgia

20,371 open computing jobs (growing at 4.8x the state average)
1,836 computer science graduates
133 schools teach computer science

- CS counts as math or science credit
- Clear certification pathways for CS teachers
- CS curriculum standards

Georgia fact-sheet | More advocacy tools

Georgia is one of 25 states where students can count computer science for credit towards high school graduation!

Take Action

https://code.org/promote

What is EarSketch?

• Online Programming and Music Mixing Workstation

- Used at Georgia Tech to teach Digital Music Mixing.
- Tool for Programming
- Free!
- Web based
Description of EarSketch

- Programming Environment
- Python Based
- Web Based App or Installed System of Software
- API built in Python for Music Mixing
- NSF Funded project to encourage computational interest through the mixing and sharing of music.
- Curriculum and Social Media Site
Authentic

Relevant

Expressive
EarSketch Website: Web based IDE and DAW

- Sound Library
- Music View: Display and Playback
- Documentation and Curriculum
- Toggle Views
- Coding Window: Programming

Getting Started with EarSketch

Why Learn Programming for Music?

There are many ways to get involved in making music, including playing an instrument, writing music, designing sound for film, producing beats, and so on. Computers have greatly expanded these possibilities. The musician's toolbox has grown, and new skills are needed to use these tools.

In EarSketch, you will write code that the computer understands as a set of instructions, or an algorithm, to make music with. Not only does this make traditional styles of music-making more efficient, it also opens many new possibilities for music that could never have existed before computers.

The practice of creating music by programming is called algorithmic composition. Here are some reasons you might want to program to create music:

- You can automate tedious tasks. Imagine that you want to combine hundreds or even thousands of snippets of sound taken from dozens of audio files. You can do this through a graphical user interface (GUI).
What is Python?

• General purpose Computer Programming Language
• Web development: Google and Yahoo
• Game Development, Science, Graphics
• In Business: National Weather Service, NASA, IBM, Disney, and Nokia
• Used in Development of the Google Car and other Robotic Systems
Sample EarSketch Program

```python
# python code
#
# script_name: sample.py
#
# author: Christopher Michaud
#
# description: Demo of EarSketch Sections

from earsketch import *

init()
setTempo(120)

# A Section - Three Tracks
fitMedia(TECHNO_LOOP_PART_003, 1, 1, 5)
fitMedia(TECHNO_CLUBLEAD_001, 2, 1, 5)
fitMedia(TECHNO_CLUBSFX_001, 3, 1, 5)

finish()
```
Essential Elements we will use in Python:

- **Comments**
  
  # This is a comment – meant for Humans

- **Includes** – loading preset methods or data
  
  from earsketch import *

- **Functions** – telling the computer “what to do”
  
  fitMedia(drums, 1, 1, 5)

- **Variables and data types** – Names for information stored by program
  
  Beat1 = “0+++0+++0+0+0+++”

- **Tabs**: Enclose code in sections

- **Lists**: Groups variables into one data structure
# EarSketch Python Functions

<table>
<thead>
<tr>
<th>EarSketch Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>init()</code></td>
<td>Start new EarSketch Mix and clear the DAW</td>
</tr>
<tr>
<td><code>setTempo(120)</code></td>
<td>Sets the beats per minute (speed) of the Mix</td>
</tr>
<tr>
<td><code>println(“Hello”)</code></td>
<td>Prints message to console</td>
</tr>
</tbody>
</table>
## EarSketch Python Functions

### Music Mixing Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>fitMedia(file, track, start, end)</code></td>
<td>Sets the start and end time for a media file.</td>
</tr>
<tr>
<td><code>makeBeat(file, track, measure, beatString)</code></td>
<td>Creates a beat with a specific measure and beat string.</td>
</tr>
<tr>
<td><code>setEffect(track, effect, parameter, sV, sM, eV, eM)</code></td>
<td>Applies an effect to a track with specified parameters.</td>
</tr>
<tr>
<td><code>rhythmEffects(track, effect, parameter, list, measure)</code></td>
<td>Applies rhythm effects to a track with a list of effects and measures.</td>
</tr>
</tbody>
</table>
“fitMedia” Function

\[
\text{fitMedia(file, track, start, end)}
\]

- Location of Media Sound
- Which Track in Reaper
- Start measure.
- End Measure

Example:

\[
\text{fitMedia(HIP_HOP_DRUMS1_2M, 1, 1, 9)}
\]
Setting Volume Effects

setEffect(track, VOLUME, GAIN, level, start, level2, end)

• Example

setEffect(1, VOLUME, GAIN, -40, 1, 10, 5)
## Selected List of Effects and Parameters

<table>
<thead>
<tr>
<th>Effect</th>
<th>Parameter</th>
<th>Min to Max Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>VOLUME</td>
<td>GAIN</td>
<td>-60 to 12</td>
</tr>
<tr>
<td>DELAY</td>
<td>DELAY_TIME</td>
<td>0 to 300.0</td>
</tr>
<tr>
<td>CHORUS</td>
<td>CHORUS_LENGTH</td>
<td>1.0 to 15.0</td>
</tr>
<tr>
<td>CHORUS</td>
<td>CHORUS_NUM_VOICES</td>
<td>1.0 to 8.0</td>
</tr>
<tr>
<td>DISTORTION</td>
<td>DISTO_GAIN</td>
<td>0.0 to 50.0</td>
</tr>
<tr>
<td>FILTER</td>
<td>FILTER_FREQ</td>
<td>20.0 to 20000.0</td>
</tr>
<tr>
<td>PAN</td>
<td>LEFT_RIGHT</td>
<td>-100 to 100 (Left to Right)</td>
</tr>
</tbody>
</table>

Complete Effect list at: [http://earsketch.gatech.edu/category/learning/reference/every-effect-explained](http://earsketch.gatech.edu/category/learning/reference/every-effect-explained)
“makeBeat” Method

makeBeat(file, track, measure, BeatString)

Location of Media Sound
Which Track in Reaper
What measure.

Example: “0+++0+++0+0+0+++”

Example:

makeBeat(drums, 1, 1, “0+0+0+++00-00+++”)
Beat String notation

“0, 1, 2, 3 . . . “ = Which Media Sound you want for the segment of beat. Correspond to placement in a List that is one based.

Note: 0 will refer to a sound if it is the only media file in the argument.

“+” Means extend or loop the Media sound 1/16th of a measure.

“-” Means 1/16th measure of rest.
makeBeat(ELEKTRO_HOUSE_DRUMS3_2M, 1, 1, "0+++0+++0+0+0+++")
makeBeat(ELEKTRO_HOUSE_DRUMS3_2M, 1, 1, "0+++0++00+0+0-00")
"0+++0+++0+0+0+++"

makeBeat(ELEKTRO_HOUSE_DRUMS3_2M, 1, 1, "0+++0+++0+0+0+++")
makeBeat(ELEKTRO_HOUSE_DRUMS3_2M, 1, 1, "0-0-0-0-0-0-0-0-")
For Loops: Skip Counting

```python
fillDrum = HIP_HOP_SYNTHDRUMS2_2M
beat = "0+++0+++0-000+00"

for measure in range(1, 9, 2):
    makeBeat(fillDrum, 1, measure, beat)
```

*measure* is the “index variable” = assigned values from the `range()` function

*(1, 9, 2)* means start counting at 1, end before 9 [meaning 8] and skip count by 2:

*(1, 3, 5, 7)*
Functions: Recycle and Reuse!

```python
def sectionA(start, end):
    stompDrums = HIPHOP_STOMP_BEAT_002
    bongoDrums = HIPHOP_DUSTYPERCUSSION_002
    keys = HIPHOP_SOLOMOOGLEAD_001
    scratch = ELECTRO_SFX_WHITENOISE_SCATTER_002
    fitMedia(stompDrums, 1, start, end)
    fitMedia(bongoDrums, 2, start, end)
    fitMedia(keys, 3, start, end)
    for measure in range(start, end):
        if measure % 2 == 0:
            fitMedia(scratch, 4, measure, measure+1)
```

Now I can use this section anywhere!

```
sectionA(1, 9)
sectionA(17, 25)
```
Creating a Function

1. Definition:

2. Decide on Variables for Music

3. Write fitMedia() calls

4. Write any For Loops

5. Set Effects

```python
def sectionA(start, end):
    stompDrums = HIPHOP_STOMP_BEAT_002
    bongoDrums = HIPHOP_DUSTYPERCUSION_002
    keys = HIPHOP_SOLOMOOGLEAD_001
    scratch = ELECTRO_SFX_WHITENOISE_SCATTER_002
    fitMedia(stompDrums, 1, start, end)
    fitMedia(bongoDrums, 2, start, end)
    fitMedia(keys, 3, start, end)

    for measure in range(start, end):
        if measure % 2 == 0:
            fitMedia(scratch, 4, measure, measure+1)
```
Rhythm Effects

# Define List of Values for Effects
valueList = [1000, 20000]
panList = [-100, 100]

# Define BeatString for Effects
filterString = "0011001100110011"

# Music for Track
fitMedia(EIGHT_BIT_ANALOG_DRUM_LOOP_003, 1, 1, 9)

# For Loop to call Effects
for m in range(1, 9):
    rhythmEffects(1, FILTER, FILTER_FREQ, valueList, m, filterString)
    rhythmEffects(1, PAN, LEFT_RIGHT, panList, m, filterString)
Exercises for Classroom

• Create EarSketch Account

• Mix1: (AB Section Exercise)
  • Use fitMedia and setEffect
  • Music in sections

• Mix2: (makeBeat and For Loop Exercise)
  • makeBeat
  • For Loop Structure

• Mix3: (Defining Functions Exercise)

• Final Mix
Final Mix Project Goal

- Define Three Functions
  - sectionA(start, end)
  - sectionB(start, end)
  - sectionC(start, end)

- Each function will have at least 3 musical clips
- At least one function will use a for loop and makeBeat
- At least one function will use effects

- Call your functions to create a music mix
  - ABABCBB
  - At least 64 Measures
ISTE Standard S.1.a: Create original works as a means of personal or group expression

• EarSketch is a tool that equips students generate musical mixes and create functions programmatically to manipulate sound recordings. The result is an artifact (sound file and textual code) that can be shared with others or modified in group work setting.
ISTE Standard T.1.d: Model collaborative knowledge construction

• The EarSketch online tool and Social media site allows student and teachers to reflect on each other's work and collaborate and reflect on student approach and process in creating music mix projects. The EarSketch curriculum fosters remixing of computational and musical ideas through the models of pair programming.
ISTE Standard T.2

• Section a: EarSketch features a library of culturally relevant musical sounds that allow students and instructors entrance into the world of mixing without needing extensive musical training.
Tell us what you think!

There are two ways to provide feedback on this session:

• ISTE 2015 mobile app
• isteconference.org/feedback

My Email:
michaudc@marist.com