

Sonic Pi Generative Music Unit Plan

Lesson #8 - Final Project

Lesson Objectives

Students will be able to will understand requirements and expectations for Generative Music project

Suggested Duration

3 period (45 minutes) + Gallery Walk

NYS Computer Science and Digital Fluency Learning Standards

7-8.CT.4 Write a program using functions or procedures whose names or other documentation convey their purpose within the larger task.

7-8.CT.6 Design, compare and refine algorithms for a specific task or within a program.

7-8.CT.7 Design or remix a program that uses a variable to maintain the current value of a key piece of information.

7-8.CT.8 Develop or remix a program that effectively combines one or more control structures for creative expression or to solve a problem.

Vocabulary

Generative Music: Music that in whole or in part has been created with the use of an autonomous system

Assessments


- Assess _____. Check for the ability to:
 - Incorporate multiple methods of randomization into a program.
 - Explain different types of methods of randomization and how they affect the outcome of their program.

Do Now

Students should read Final Project Assignment Document

Lesson

Part 1 - Intro to Generative Music

1. Introduce concept of Generative Music -
Generative music is a term popularized by Brian Eno to describe music that is ever-different and changing, and that is created by a system.
2. Play example from "Music for Airports" by Brian Eno
 Brian Eno - Ambient 1: Music for Airports [Full Album]
3. Explain that in this case, the system used was a series of tape machines each playing different loops running at different times and speeds, so that the the results would never repeat the same exact sequence.

Read the following quote *"The particular piece I'm referring to was done by using a whole series of very long tape loops, like fifty, sixty, seventy feet long. There were twenty-two loops. One loop had just one piano note on it. Another one would have two piano notes. Another one would have a group of girls singing one note, sustaining it for ten seconds. There are eight loops of girls' voices and about fourteen loops of piano. I just set all of these loops running and let them configure in whichever way they wanted to, and in fact the result is very, very nice. The interesting thing is that it doesn't sound at all mechanical or mathematical as you would imagine. It sounds like some guy is sitting there playing the piano with quite intense feeling. The spacing and dynamics of "his" playing sound very well organized. That was an example of hardly interfering at all."* from

<https://reverbmachine.com/blog/deconstructing-brian-eno-music-for-airports/>

4. Explain that this method was used before computers were widely available and able to easily create musical output. As computers became more efficient and able to process audio more effectively, they were used to create generative music using programming and algorithms.

Part 2 - Assignment Roll out

1. Go over Assignment sheet.
 - Requirements
 - Random methods discussed in class
 - Grading expectations / Rubric
2. Show students an example project and play them the output.

Example Project Code

```
1 live_loop :loop1 do
2   use_random_seed Time.now.to_i
3   if one_in(3)
4     n = rrand_i(0, 4)
5     sample :elec_beep if n == 0
6     sample :ambi_glass_rub if n == 1
7     sample :ambi_soft_buzz if n == 2
8     sample :elec_cymbal if n == 3
9     sample :ambi_piano if n == 4
10  else
11    n = rrand_i(0, 4)
12    sample :glitch_perc1 if n == 0
13    sample :glitch_perc2 if n == 1
14    sample :glitch_perc3 if n == 2
15    sample :glitch_perc4 if n == 3
16    sample :glitch_perc5 if n == 4
17  end
18  if one_in(4)
19    sleep 0.75
20  else
21    sleep 1.25
22  end
23 end
24
25 live_loop :loop2 do
26   use_random_seed Time.now.to_i
27   seq = scale(40, :minor).shuffle
28   attackRoll = dice
29   releaseRoll = dice
30   new_seq = seq.take(dice)
31   tick_reset
32   new_seq.length.times do
33     use_synth :tb303
34     play new_seq.tick, attack: attackRoll, release: releaseRoll, amp: 0.5
35     use_synth :sine
36     play 28, attack: attackRoll, release: releaseRoll, amp: 0.7
37     sleep attackRoll + releaseRoll
38   end
39 end
40
41 rain = "/Users/admin/Downloads/mixkit-thunderstorm-in-the-forest-2396.wav"
42
43 live_loop :loop3 do
44   sample rain
45   sleep sample_duration rain
46 end
```

Part 3 - Work Time

Students will be provided 3 class periods to work on this project.

Assignments will be submitted in a Google Doc on Google Classroom along with comments for written requirement of project

Wrap Up/Assessment

Students will have a gallery walk where they will leave their programs running and they can move from one computer to another and spend 2-3 minutes listening to the piece before moving onto the next one.

Task: You will write a program that creates a piece of generative music using different methods of randomization

Requirements for Generative music

- This piece should be ongoing (no definitive end)
- It should slightly change over time
- Changes should be influenced by some type of random events

Code Requirements

- Minimum of 3 Live Loops
- Use of randomization to create different outputs
 - .choose
 - Data structure manipulation
 - Single line conditional statements
 - Nested Conditional statements
 - One_in probability function
 - Random parameters
 - Coin flips/dice rolls (rand_i, dice)
- Use of play and sample functions
- One loop uses sounds from nature (wind, rain, ocean, forest, jungle etc)











Written Requirements

Code should be copied and pasted into Google Doc.

Use commenting in Google Classroom to identify the following:

- What methods of randomization are used
- How random methods affect the outcome of the program

Rubric

	   	  	 	
Implementation of class concepts / Creativity	Code is significantly different from class examples	Code makes several alterations to the class example	Code copies class example with minimal alteration	Code does not include any attempt to use concepts presented in class
Use of different methods of randomization	Accurately uses 3 or more methods of randomization in code	Accurately uses 2 methods of randomization in code	Accurately uses 1 method of randomization in code	Code does not accurately use methods of randomization
Written Explanation of code	Clearly and logically explain how randomization in code is used	Adequately explain how randomization in code is used	Explanation about how randomization in code is used is unclear/requires more detail	Does not explain how randomization in code is used

