Sonic Pi Composition Curriculum Plan

"My favourite music is the music I haven't yet heard." –John Cage

Unit Overview:

This unit will help the teacher navigate through a class project on making music (composition) with programming (also known as coding) through a platform called Sonic Pi. Don't worry if you've had no experience in either, as this unit is designed for complete beginners for both the student and teacher. However, you will need to become familiar with the materials presented here if you've had no experience with programming and/or music composition.

Sonic Pi is a platform created by Sam Aaron (research associate at Cambridge University) and is used specifically for writing music while learning to code. You can watch a great TEDxNewcastle presentation of Sam demonstrating Sonic Pi where he also explains his rationale behind building this platform. I've also made a shorter introduction to Sonic Pi which you can show to your students here.

This unit of work is project based, where students make music with Sonic Pi for a selection of short videos of about 1-2 minutes in length. Each student will engage in both a group and individual project. The videos have specific themes on current issues facing the world like climate change, plastic in the oceans, and the refugee crisis, where the aim is to create what the students think might be an appropriate musical background for their chosen film. These topics have been chosen so that they engender a rich discussion and engage musically with meaningful contexts, however, the teacher or student could use any video that they would like to. The videos can be found under the resource dropdown in the sidebar (or just click here).

This unit of work also gets students to engage in chance based/algorithmic composition with music listening examples from Mozart, John Cage, Steve Reich, and generative music by Brian Eno. There are two pedagogical reasons for this: (1) to pull students away from the imitation of established styles of music and towards building an intuitive sense of arranging sounds for an introduction to composition, and (2) to get students to engage with what computers can do effectively – executing algorithms.

This unit of work hopes to develop students' computational thinking in an interdisciplinary way with both music and programming. A framework by Brennan and Resnick (2012) has been embedded into the design of this unit of work from their experiences with Scratch at MIT.

Please read pedagogy and teaching notes link before commencing this project.

If you have any suggestions for any part of this unit of work, email me at petriechris@gmail.com and I'll be happy to hear your thoughts (or typo corrections)!

PLEASE NOTE:

→ This unit of work has been designed for six one hour and 40 minute lessons for students of roughly 12/13 years of age for the new Digital Technologies Curriculum introduced in New Zealand in 2018. However, the learning sequence, activities and materials presented in this unit of work are designed to be flexible to the learning needs and capabilities of teachers and students. This means if appropriate: activities, materials, listening examples (everything!) could be swapped or omitted.

→ This curriculum plan has been designed as part of a Master's thesis in Education. A mixed method case study has been conducted on this unit of work. The final thesis and other associated publications will be posted on this website once they've been published.

Download a PDF version of this unit plan

Contents:

Unit plan overview	-
Preparation tasks	-
Lesson #1 Getting comfortable	1 hour 40 minutes
Lesson #2 Scales and samples	1 hour 40 minutes
Lesson #3 Algorithms in music	1 hour 40 minutes
Lesson #4 Synthesisers and effects	1 hour 40 minutes
Lesson #5 Debugging, optimisation, and efficiency	1 hour 40 minutes
Lesson #6 Showcase and final reflections	1 hour 40 minutes
Assessment plan (rubrics)	-

Unit plan overview:

Core subject(s)	Music (Composition); Computer Science/Digital Technologies	
New Zealand Curriculum Level:	**Computer Science:** NZ Digital Technologies progress outcomes level 4: **Music:** Music–Sound arts achievement standards level 4	
Planned for:	Year 8 music class. Individual, group, and class activities/projects	
New Zealand Curriculum links	**Music** Practical Knowledge (PK): -Music theory focus: Timbre, Texture, Tonality Developing ideas (DI): -Composition Techniques (Algorithmic composition) -Individual Composition Communicating and Interpreting (CI): -Group composition project -Writing music for video with a brief (topical issues facing the world today) Understanding Context (UC): -film music, algorithmic music, generative music -Listening from Mozart, John Cage, Steve Reich, and generative music by Brian Eno	**Digital Technologies Hangarau – Matihiko** Computational Thinking: -Programming (sequence, selection, iteration) -Algorithms -Abstraction/decomposition -Pattern recognition -experimenting and exploring -Data representation (audio) -Debugging -Reflection Designing and Developing Digital Outcomes: -Two algorithmic compositions for film (group and individual projects)
Classroom time required:	Six 1 hour and 40 minute lessons	
Resources needed:	For the teacher: ● Projector connected to a computer with high quality speakers loud enough to play music at high volumes ● Many spare headphones ● Internet connection ● Headphone splitters (one between two students) For each student: ● Computer for each student with an internet connection	
Notes	→ All lessons will start with a 5-minute recap of concepts covered in previous lessons	

	→ For lessons 1-5: 20 minutes before each lesson finishing, students will: a) Save progress (not overwriting previous saved files) and take screenshots of progress b) Answer quiz on concepts covered in Programming and Music, complete short written diaries on challenges and successes (5 minutes) c) Participate in class discussion on what they found challenging and successful (5 minutes)

Preparation tasks:

- Confirm that computers are switched on, logged-in, connected to the internet and working with audio (speakers, headphones, headphone splitters) - Ensure students can save their work. *A list of all required equipment for this unit of work is in the unit plan	5 to 10 minutes
IN PREPARATION FOR LESSON #6 ONLY: - It would be easier for this lesson to export each project as an audio file, download each video and use free video editing software to present each project such as movie maker (Windows) or iMovie (Mac). This is less awkward than attempting to sync the video on a projector with the original sound on mute while the students project is pressed at the same time - Ensure each project plays before this class starts. Some students might have left bugs in their code that might be embarrassing for them if error messages appear when presenting their work	Over 1 hour depending on the number of students

Lesson #1 Getting comfortable:

Suggested learning sequence	**Key concepts**	**Sonic Pi syntax to be taught this lesson**	**Interdisciplinary Curriculum Links**			**Learning Outcomes**
			Computational Thinking	Programming	Music (strands)	

1. *Introduction*: What is music composition? What is programming? **15 minutes**
2. *Activity 1*: Explore Sonic Pi basic commands: `play, sleep, run`. **15 minutes**
3. *Activity 2*: Adjusting an algorithm to generate music. **30 minutes**
4. *Activity 3*: Group composition brief. **20 minutes**
5. *Wrap-up Activity*: Quiz and reflection. **10 minutes**
	Music:
	-pitch (high and low)
	-timbre
	-synthesiser
	-repetition
	
	Programming:
	-sequence
	-loop
	-input/output
	-debugging
		Activity #1
		`play, sleep`
		
		Activity #2
		`use_synth`
		`_times.do`
		`loop do`
		`live_loop :foo do → end`
			Debugging, abstraction, experimenting	Iteration, loops, simple functions, sequence, input/output	PK, UC	Music:
						-All students will explore and experiment with at least three different synthesised sounds using Sonic Pi
						-All students will use repetition in a composition
						
						Programming:
						-All students will make a sequence of at-least four steps
						-All students will make and debug a loop to abstract repetition in their composition

Lesson #2 Scales and samples:

Suggested learning sequence	Key concepts	Sonic Pi syntax to be taught this lesson	Interdisciplinary Curriculum Links			Learning Outcomes
			Computational Thinking	Programming	Music (strands)	

1. *Introduction*: What did we cover in the last lesson? **5 minutes**
2. *Activity 1*: Write efficient code to ‘Time’ by Hans Zimmer from the film ‘Inception’. **20 minutes**
3. *Activity 2*: Create your own tonal mood. **40 minutes**
4. *Activity 3*: Listening and reflecting. **20 minutes**
5. *Wrap-up Activity*: Quiz and reflection. **15 minutes**
	Music:
	-tonality
	-scale
	-pentatonic
	-octave
	-melody
	
	Programming:
	-pattern recognition
	-functions
	-lists
	-naming conventions
		Activity #1
		`using note names e.g. :C4`
		
		Activity #2
		`sample`
		`play [:A, :E, :D, :G]`
		`notes = (ring :E4, :Fs4, :B4, :Cs5, :D5, :Fs4, :E4, :Cs5, :B4, :Fs4, :D5, :Cs5)”`
		`using the .tick, .choose and .shuffle methods`
			Debugging, iteration, abstraction, making algorithms, patterns	Functions, loops, arguments,, data types (lists), methods, naming conventions	PK, DI, UC	Music:
						-All students will recognise basic composition repetition/variation in ‘Time’ by Hans Zimmer
						-All students will layer three sounds on top of one another
						-All students will create their own scale to find a tonality that suits their chosen video
						-All students will discuss the characteristics of generative or systems based music
						
						Programming:
						-All students make a musical algorithm that plays the chord sequence of Hans Zimmer’s ‘Time’
						-All students will develop strategies to make an algorithm more efficient through using iteration (loops)
						-All students will create and debug algorithms for a loop of a series of notes (scale) they have created

Lesson #3 Algorithms in music:

Suggested learning sequence	Key concepts	Sonic Pi syntax to be taught this lesson	Interdisciplinary Curriculum Links			Learning Outcomes
			Computational Thinking	Programming	Music (strands)	

1. *Introduction*: What did we cover in the last lesson? **5 minutes**
2. *Activity 1*: Introduce brief for individual project. **30 minutes**
3. *Activity 2*: Introduction to generative music with Sonic Pi and student time on individual projects. **30 minutes**
4. *Activity 3*: Pair/group/class reflection on progress so far. **15 minutes**
5. *Wrap-up Activity*: Quiz and reflection. **15 minutes**
	Music:
	-texture
	-mood
	-sampling
	-stretching, layering, cuttoff/highpass
	
	Programming:
	-random number generator
	-selection
	-arguments
		Activity #1
		`rate`
		`cutoff`
		
		Activity #2
		`rrand()`
		`.choose`
		`use_random_seed`
		`if else`
			Decomposition, conditional logic, making algorithms	Methods, selection, arguments, parameters	PK, DI, UC	Music:
						-All students will identify and experiment with characteristics of algorithmic music and mood/timbre
						-All students will be introduced to sampling and experiment with basic sample manipulation
						
						Programming:
						-All students will use a random number generator (`rrand`) within Sonic Pi
						-All students will experiment conditional logic (`if/else`)
						-All students will use arguments to stretch and sculpt audio samples with `.rate` and `.cutoff`

Lesson #4 Synthesisers and effects:

Suggested learning sequence	Key concepts	Sonic Pi syntax to be taught this lesson	Interdisciplinary Curriculum Links			Learning Outcomes
			Computational Thinking	Programming	Music (strands)	

1. *Introduction*: What did we cover in the last lesson? **5 minutes**
2. *Activity 1*: Data representation of audio. **30 minutes**
3. *Activity 2*: New Sonic Pi commands (additive and subtractive synthesis with sine, sawtooth, triangle waves). **30 minutes**
4. *Activity 3*: Reflection of progress so far and class discussion of selected listening examples. **10 minutes**
5. *Activity 4*: Student time for group projects. **20 minutes**
6. *Wrap-up Activity*: Quiz and reflection. **15 minutes**
	Music:
	-reverb, delay, distortion
	-attack, release sustain, decay
	-sine, sawtooth, triangle waves
	
	Programming:
	-decomposition (making synthesisers)
	-abstraction
	-bits to audio (data representation)
	-methods
		Activity #1
		`(none)`
		
		Activity #2
		`with_fx:`
		`-reverb`
		`-echo`
		`-synth: (sine, square, saw, tri)`
		`-attack -release -sustain -decay`
		
		Activity #3 and #4
		`(none)`
			Abstraction, debugging, decomposition	Methods, parameters	PK, DI	Music:
						-All students will create their own unique sounds using additive and subtractive synthesis
						-All students will use basic effects e.g. `reverb, delay, cutoff`
						
						Programming:
						-All students will create and use their own personalised synthesiser
						-All students will explore `attack, release` and `sustain` of synthesising a unique and personalised electronic instrument

Lesson #5 Debugging, optimisation, and efficiency:

Suggested learning sequence	Key concepts	Sonic Pi syntax to be taught this lesson	Interdisciplinary Curriculum Links			Learning Outcomes
			Computational Thinking	Programming	Music (strands)	
1. *Introduction*: What did we cover in the last lesson? **5 minutes** 2. *Activity 1*: Class discussion on examples of film music about topical issues. **10 minutes** 3. *Activity 2*: Recap of Sonic Pi commands and student time for group compositions. **30 minutes** 4. *Activity 3*: Student time for the development of individual soundscape compositions and final hand in. **30 minutes** 5. *Wrap-up Activity*: Quiz and reflection. **15 minutes**	Music: -using 6 music elements (pitch, texture etc) to give feedback on each other's composition Programming: -recap of any gaps	(none)	Abstraction, making algorithms, debugging, efficiency/optimisation	As per lessons 1-4, debugging	DI	Music: -All students will develop and finalise musical ideas for both individual and group projects -All students will adhere to the projects length constraints of under 2 minutes Programming: -Lessons 1-4 -All students will adhere to the projects length constraints of under 2 minutes

Lesson #6 Showcase and final reflections:

Suggested learning sequence	Key concepts	Sonic Pi syntax to be taught this lesson	Interdisciplinary Curriculum Links			Learning Outcomes
			Computational Thinking	Programming	Music (strands)	
1. *Introduction*: Making constructive comments musically and computationally. **10 minutes** 2. *Activity 1*: Class demonstration and comments of all projects with film. **70 minutes** 3. *Wrap-up Activity*: Quiz and reflection. **15 minutes**	Music: -constructive feedback -arc of composition -variation/repetition ratio Programming: -efficiency	(none)	Evaluation/Reflection	As per lessons 1-4, debugging	DI, CI	Music: -All students will evaluate their projects for their fitness for purpose to their chosen video -All students will have the opportunity to give constructive feedback

	-optimisation -performance constraints (time limit)					Programming: -All students will adhere to the project length constraints of the length of the chosen video -All students will reflect on the efficiency of their own and another student's code: reflect on potential ways to optimise the code

Assessment plan:

Final Projects	link
End of lesson quizzes and reflections	link

Administrative Details

Contact info	petriechris@gmail.com
Credits	Developed by Chris Petrie.
Last updated on	27/06/2018
Copyright info	The content of this page is licensed under the Creative Commons Attribution 3.0 License, and code samples are licensed under the Apache 2.0 License.