SonicPi – Introduction

See: https://github.com/CodeClubStrad/CodeClubStrad-Spring-2016/tree/master/2016_02_08_session_5

Play all the notes (version 1)

```
# playAllNotes
# A simple SonicPi for loop used to play a sequence of increasing midi notes
# Try running it...
# When can you start to hear the note?
# When can you no longer hear it?
# You probably can't hear anything outside the range 50 - 120 (depending how old you are
:-)

# MIDI go from 0 to 127 (128 notes)
for note in 0..127
  # print the value of note to the console for feedback (in case we can't hear it)
  puts note
  # play the note
  play note
  # wait 1/2 a second
  sleep 0.5
end
```

Play all the notes (version 2)

```
# playAllNotesLiveLoop
# An improvement on the 'for' loop:
# A simple SonicPi live_loop used to play a sequence of increasing midi notes
# In this case we have to stop (break) the loop at note 127

# start with note 0
note = 0
live_loop :playAllNotes do
  # print the note value to the console for feedback (in case we can't hear it!)
  puts note
  # now play the note
  play note
  # add one to the value of note
  note = note + 1
  # check the value of note - MIDI only goes up to 127 (128 notes)
  if note > 127
      puts "Midi note out of range (> 127)"
    # stop the loop (creates a not very pretty error)
    break
  end
  # wait for 1/2 a second
  sleep 0.5
end
```

Linking Sonic-Pi and MinecraftPi

See: https://github.com/CodeClubStrad/CodeClubStrad-Spring-2016/tree/master/2016_02_29_session_6

SonicPi will automatically connect to MinecraftPi if it is running…

Basic functions

Look at Section 11 of the SonicPi Tutorials:

- `mc_message "text"`
    - -> sends a message to the MinecraftPi chat
- `mc_location`
    - -> returns a list of the current player location (x,y,z)
- `mc_teleport x, y, z`
    - -> moves the player to the location given by x,y,z
- `mc_set_block :type, x, y, z`
    - -> creates a block of type 'type' at x,y,z
- `mc_get_block x, y, z`
    - -> returns the kind of block at that location

'Play' the height of our player in the world

```
# mc_PlayHeight
# © 2016, @dataknut
# A simple SonicPi live loop to play a note that represents
# our Minecraft Pi height (y) value - a higher note will represent being higher up in the
world
# post a message to MinecraftPi to prove the connection is working
mc_message "Hello from Sonic Pi"

live_loop :mc_playHeight do

  # what's our location?
  puts mc_location

  # We could play the note that matches the y location:
  # play mc_location[1]
  # Why does this often not work?
  # Correct - midi notes lie in the range 0 - 127
  # but in MinecraftPi: -128 < y < 128
  # So we need to transform the y value into something more helpful

  # Let's assume 90 is the middle of the range we can hear.
  # MinecraftPi is -128 < y < 128 (a small world :-)
  # So take the value of y and work out it's proportion of 128 (max depth/height)
  # Find the note that is that proportion of half our range (30)
  # Add/subtract that to/from 90
  heightNote = 90 + (mc_location[1]/128)*30
  puts heightNote
  play heightNote, release: 0.5
  sleep 0.5
end
```

‘Play’ the location of our player in the world

```
# mc_playLocation
# Author: @dataknut
# License: https://github.com/CodeClubStrad/CodeClubStrad-Spring-
2016/blob/master/LICENSE.md

# A simple SonicPi live loop to play a note that represents our Minecraft Pi location
(x,y,z) values
# Uses pan (left vs right speaker) for x - see 2.2 Synth Options
# Uses the note for y (height)
# Uses amp (amplitude) for z - see 2.2 Synth Options

# Set this loop running and then move around in MinecraftPi

# post a message to Minecraft to prove the connection is working
mc_message "Hello from Sonic Pi"

live_loop :mc_playLocation do
  # put our location x,y,z values into a variable - saves connecting to minecraft lots of
times
  myLoc = mc_location

  # Turn x into a useful pan value
  # Pan has the range -1 (left) to 1 (right)
  # Use the fact that the MinecraftPi world goes from -128 to 128 on any dimension
  # this should create a value between -1 and +1
  xPan = myLoc[0]/128

  # Turn y into a useful height note (see mc_playHeight.rb)
  yNote = 90 + (myLoc[1]/128)*30

  # Turn z in to a useful amp value - between 0 (no sound) -> 1 (normal) and 2 (loud)
  # See how we have to use .abs to always give a positive value
  zAmp = (myLoc[2].abs/128)*2

  # print location
  puts myLoc

  # the converted values get printed by play to the console anyway

  # Now play the note
  play yNote, pan: xPan, amp: zAmp, release: 0.5
  # Wait 1/2 a second
  sleep 0.5
end
```

More fun:

Look at the MinecraftPi/Sonic-Pi examples in Appendix A6 and A8 in the SonicPi Tutorials.

Another big list of block types

:air :stone :grass :dirt :cobblestone :wood_plank :sapling :bedrock :water_flowing :water :water_stationary :lava_flowing :lava :lava_stationary :sand :gravel :gold_ore :iron_ore :coal_ore :wood :leaves :glass :lapis :lapis_lazuli_block :sandstone :bed :cobweb :grass_tall :flower_yellow :flower_cyan :mushroom_brown :mushroom_red :gold_block :gold :iron_block :iron :stone_slab_double :stone_slab :brick :brick_block :tnt :bookshelf :moss_stone :obsidian :torch :fire :stairs_wood :chest :diamond_ore :diamond_block :diamond :crafting_table :farmland :furnace_inactive :furnace_active :door_wood :ladder :stairs_cobblestone :door_iron :redstone_ore :snow :ice :snow_block :cactus :clay :sugar_cane :fence :glowstone_block :bedrock_invisible :stone_brick :glass_pane :melon :fence_gate :glowing_obsidian :nether_reactor_core