Building Custom Effects with EarSketch
Mr. Michaud
Summer Camp 2013

Summarized From: http://earsketch.gatech.edu/category/learning/creating-your-own-effects

Overview:

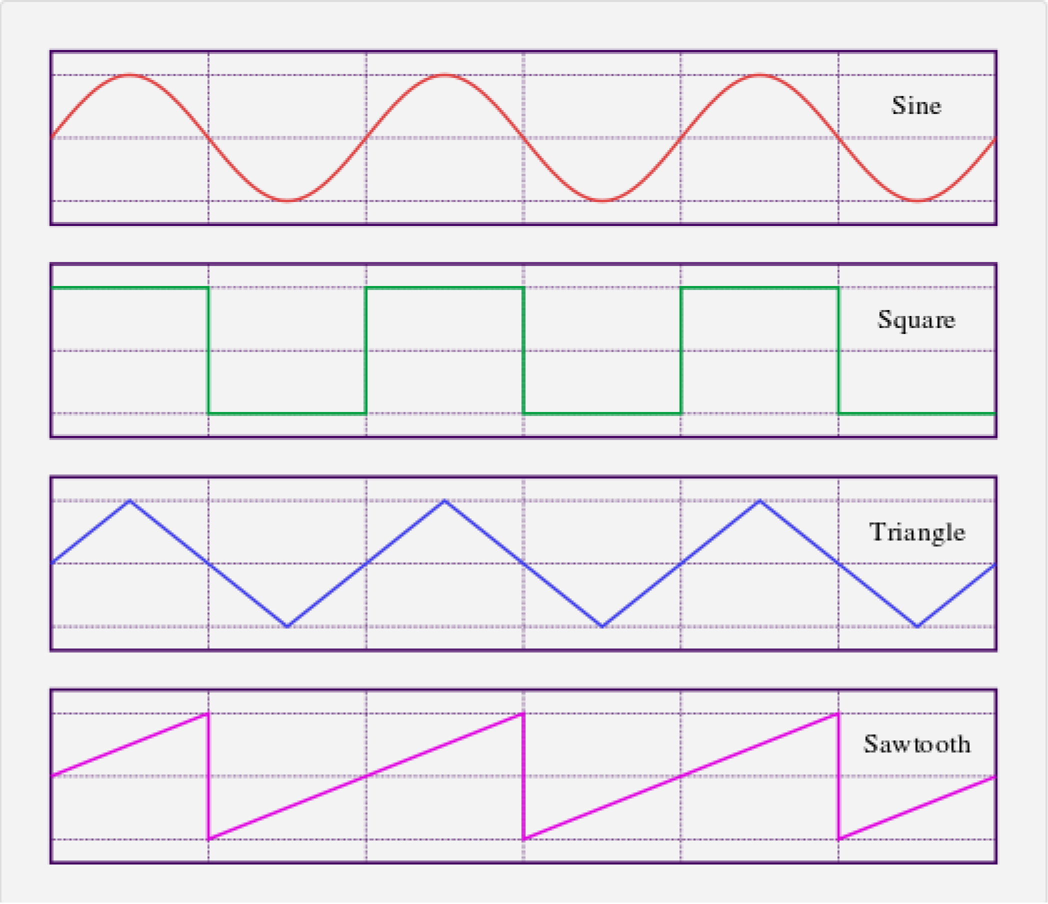
In music production – effects such as distortion, reverb, delay, ring modulation, and EQ are generated by linking together components within a sound system. The sound engineer would then wire together these components to direct, split, and combine the audio signal. Each machine 'on the rack' would process and change the audio signal. The signal then passes back to the main mixing board and out to the speaker system.

In python and EarSketch programming, we can simulate this process with code to build our own effects generators. This lesson will outline the steps build effects.

The outline of steps are:

1. Declare an instance of an effect using the initEffect() function.
2. Create the Unit Generators for the Effect with the createUGen() function.
    a. INPUT
    b. TIMES
    c. TRIANGLE, SQUARE, SAW, SINE . . .
    d. OUTPUT
3. Wire the Unit Generators Together with the connect() function.
4. Set the Parameters with the setParamMin(), setParaMax(), and setParam() function
5. Create the Control with the createControl() function
6. Finish the effect with the finishEffect() function.

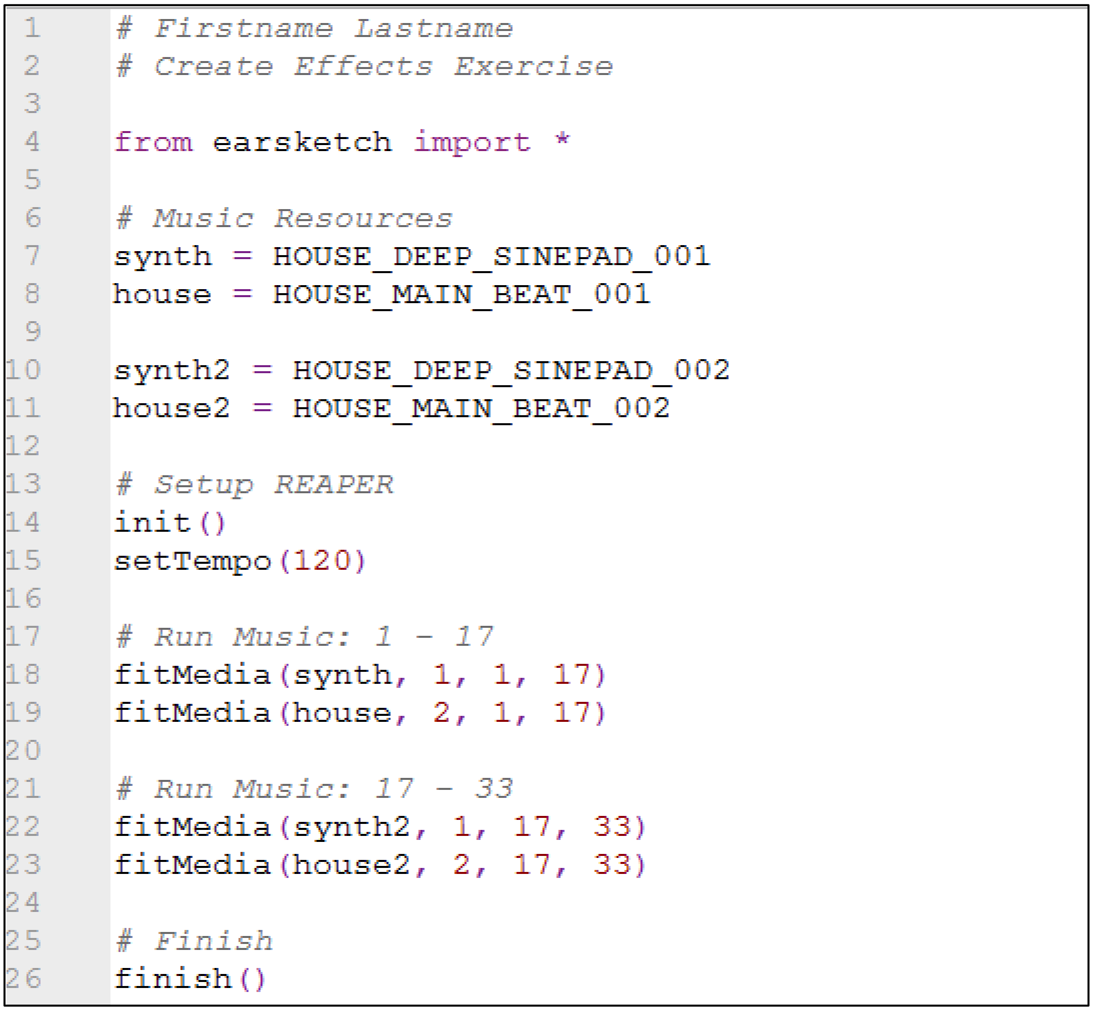
In These exercises we will build RING effects using Triangle and Square waves.


Sine
Square
Triangle
Sawtooth

Exercise 1: Setup EarSketch Mix

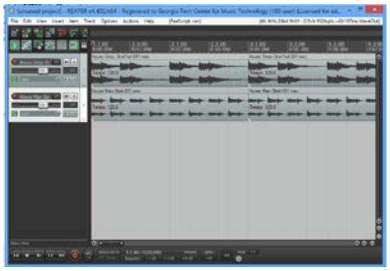
Process:

1. Start Komodo and create a new Python file. Save with the name of ‘lastname_create_effects.py’
2. Type the following code to establish a base mix of drum and melodic sounds:

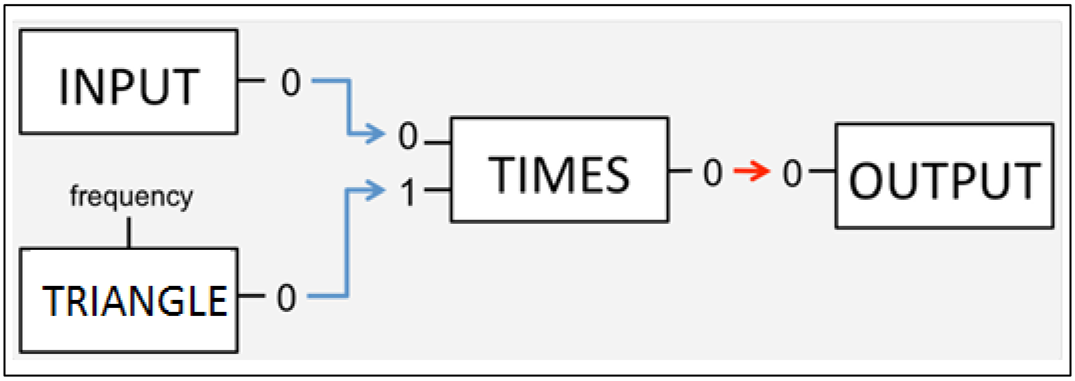


```
# Firstname Lastname
# Create Effects Exercise

from earsketch import *

# Music Resources
synth = HOUSE_DEEP_SINEPAD_001
house = HOUSE_MAIN_BEAT_001

synth2 = HOUSE_DEEP_SINEPAD_002
house2 = HOUSE_MAIN_BEAT_002

# Setup REAPER
init()
setTempo(120)

# Run Music: 1 - 17
fitMedia(synth, 1, 1, 17)
fitMedia(house, 2, 1, 17)

# Run Music: 17 - 33
fitMedia(synth2, 1, 17, 33)
fitMedia(house2, 2, 17, 33)

# Finish
finish()
```

3. Save and Run in REAPER.

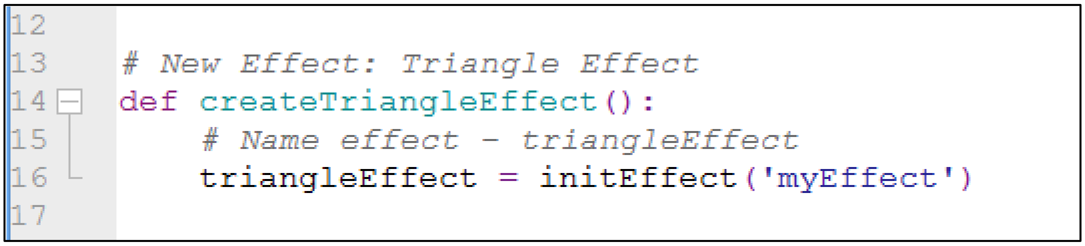


Exercise 2: Write the Function for the Triangle Effect



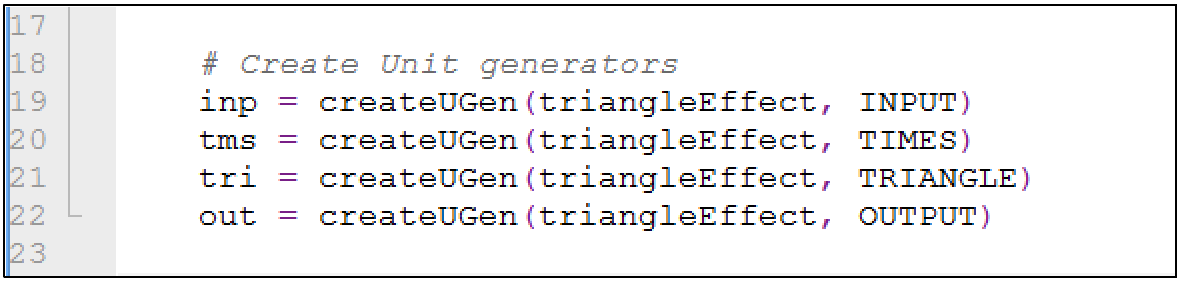
Process:

1. Define the Function and use initEffect() to create a new effect named ‘triangleEffect’ (Lines 13 to 16)



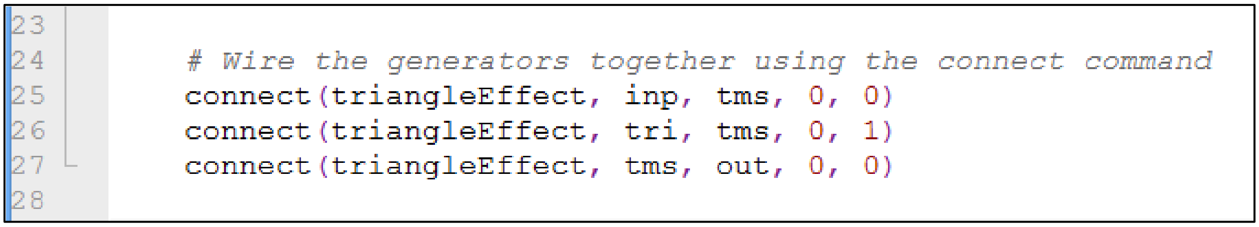
```
12
13  # New Effect: Triangle Effect
14  def createTriangleEffect():
15      # Name effect - triangleEffect
16      triangleEffect = initEffect('myEffect')
17
```

2. Create the Unit Generators for the Effect   (Lines 17 to 23)
   a. inp -> INPUT
   b. tms -> TIMES
   c. tri -> TRIANGLE
   d. out -> OUTPUT



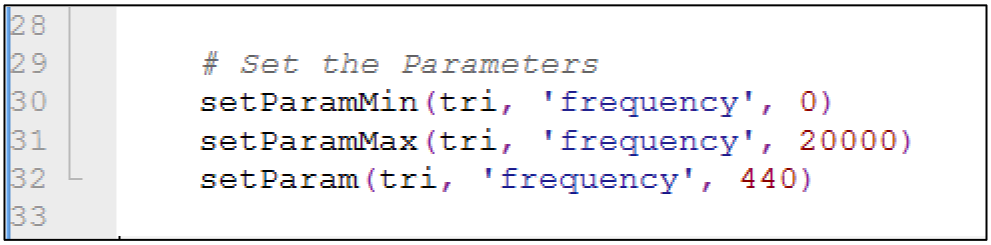
```
17
18      # Create Unit generators
19      inp = createUGen(triangleEffect, INPUT)
20      tms = createUGen(triangleEffect, TIMES)
21      tri = createUGen(triangleEffect, TRIANGLE)
22      out = createUGen(triangleEffect, OUTPUT)
23
```

3. Wire the Unit Generators Together (Lines 23 to 28)



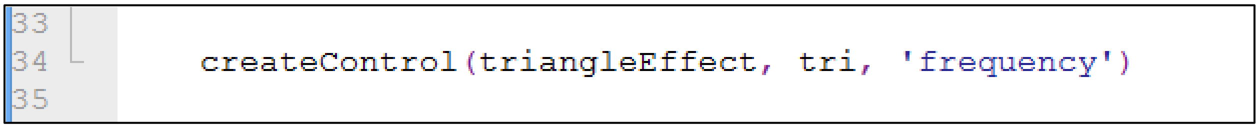
```
23
24      # Wire the generators together using the connect command
25      connect(triangleEffect, inp, tms, 0, 0)
26      connect(triangleEffect, tri, tms, 0, 1)
27      connect(triangleEffect, tms, out, 0, 0)
28
```

4. Set the Parameters: (Min will be 0, Max will be 20,000, Default will be 440) (Lines 28 to 33)



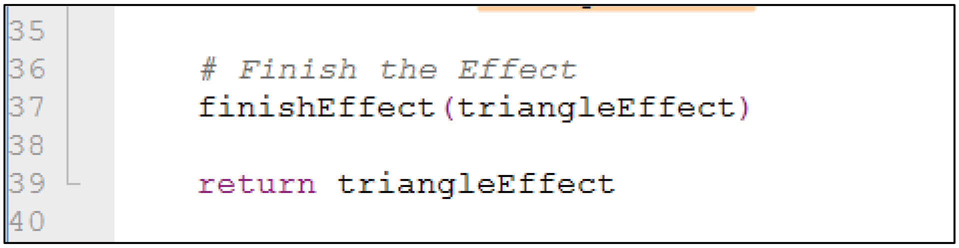
```
28
29      # Set the Parameters
30      setParamMin(tri, 'frequency', 0)
31      setParamMax(tri, 'frequency', 20000)
32      setParam(tri, 'frequency', 440)
33
```

5. Create the Control: (Line 34)



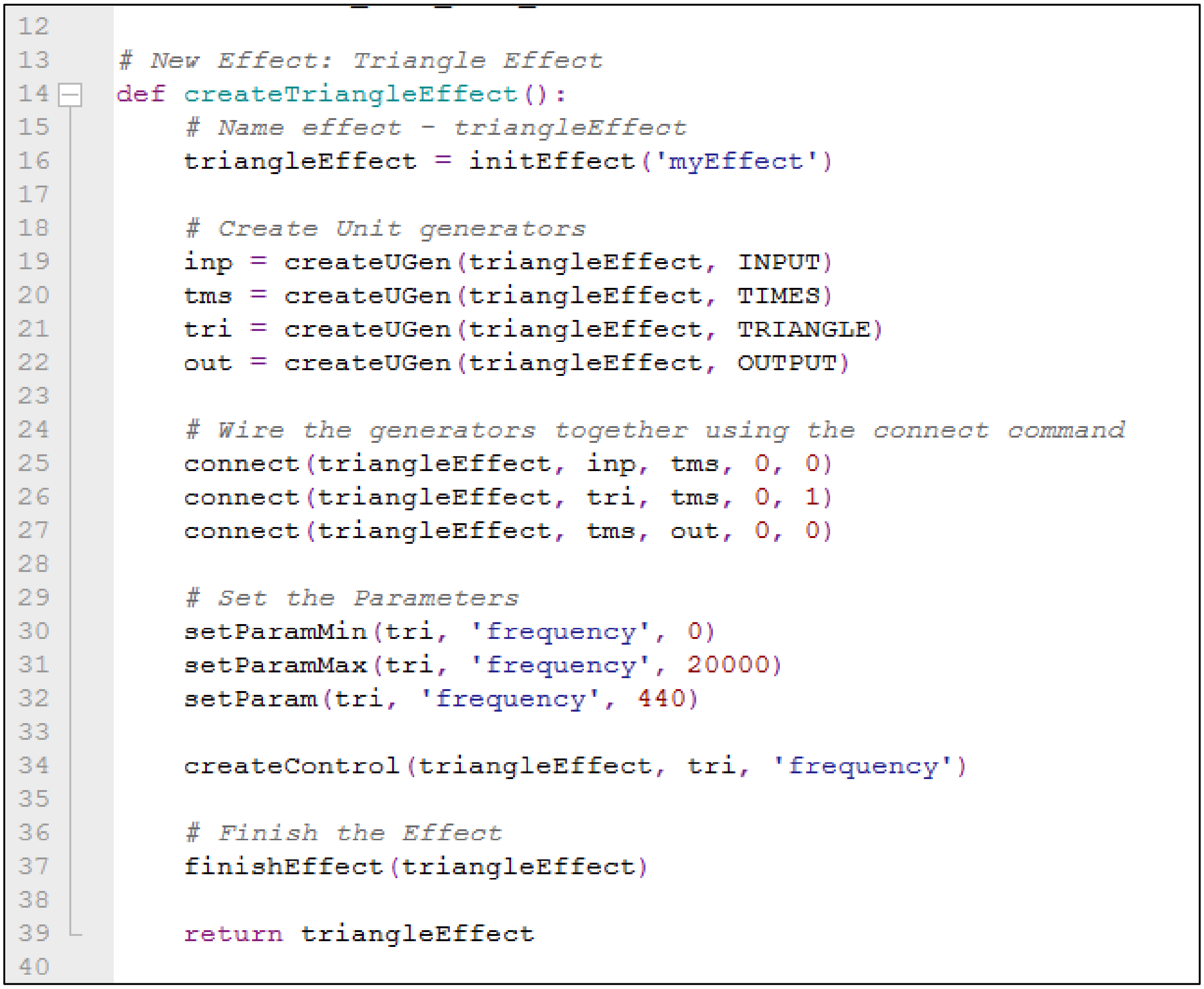
```
33
34      createControl(triangleEffect, tri, 'frequency')
35
```

6. Use the finishEffect() command to finish creating the Effect and then a return to return the Effect when the function is called.



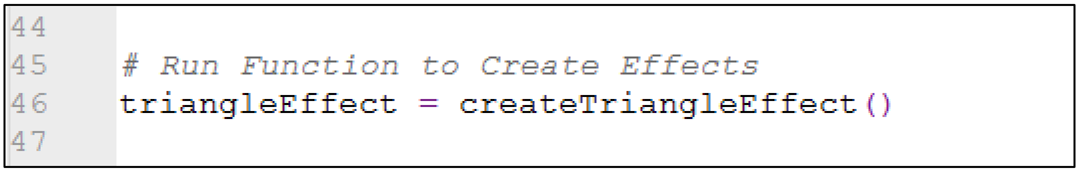
```
35
36      # Finish the Effect
37      finishEffect(triangleEffect)
38
39      return triangleEffect
40
```

7. The Entire Function should look like this:



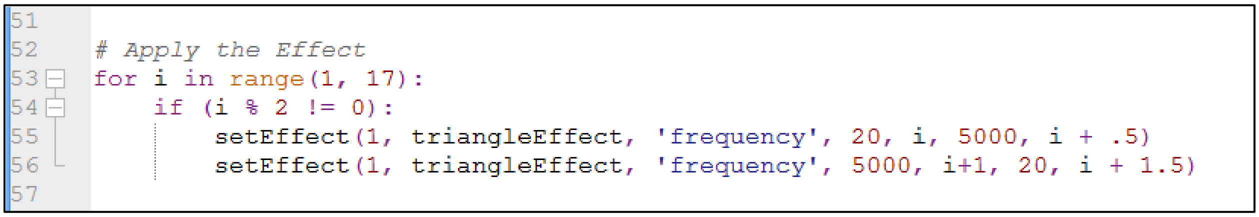
```

# New Effect: Triangle Effect
def createTriangleEffect():
    # Name effect - triangleEffect
    triangleEffect = initEffect('myEffect')

    # Create Unit generators
    inp = createUGen(triangleEffect, INPUT)
    tms = createUGen(triangleEffect, TIMES)
    tri = createUGen(triangleEffect, TRIANGLE)
    out = createUGen(triangleEffect, OUTPUT)

    # Wire the generators together using the connect command
    connect(triangleEffect, inp, tms, 0, 0)
    connect(triangleEffect, tri, tms, 0, 1)
    connect(triangleEffect, tms, out, 0, 0)

    # Set the Parameters
    setParamMin(tri, 'frequency', 0)
    setParamMax(tri, 'frequency', 20000)
    setParam(tri, 'frequency', 440)

    createControl(triangleEffect, tri, 'frequency')

    # Finish the Effect
    finishEffect(triangleEffect)

    return triangleEffect

```

8. Now we need to call this function to create the effect and apply it with the code. Type the following to create the effect: (Lines 43 and 44)



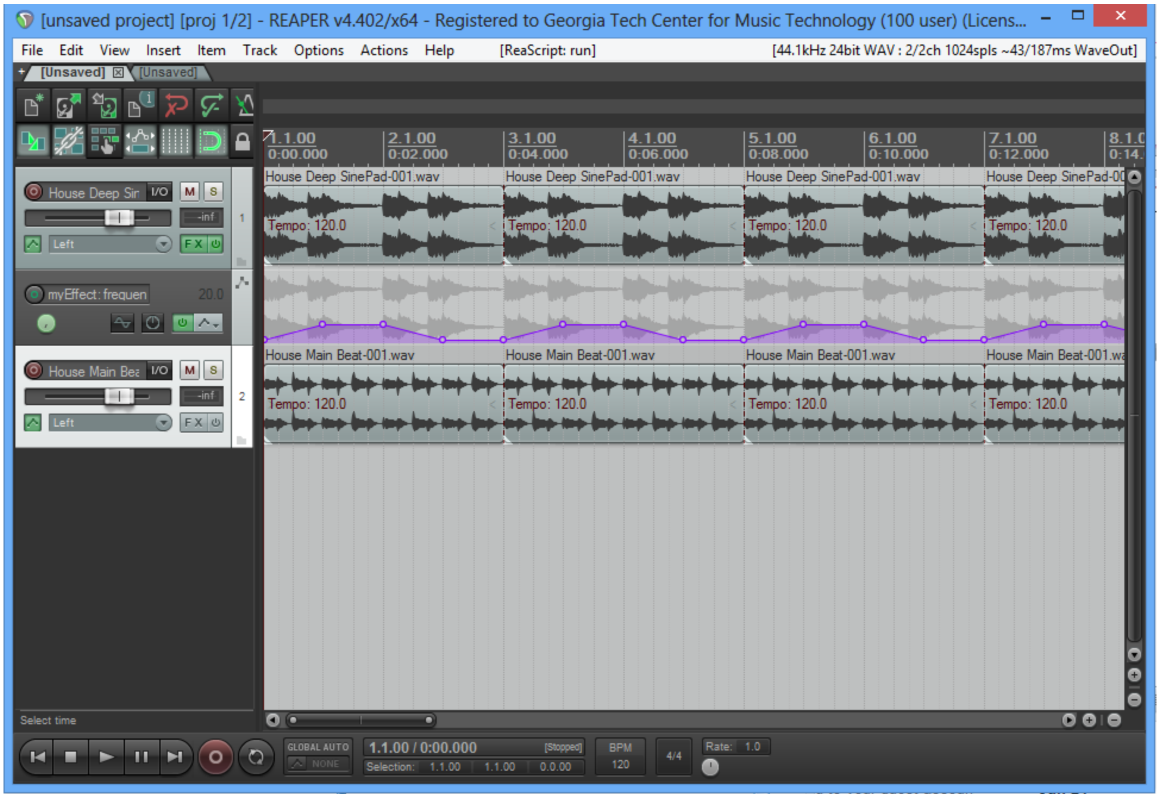
```
44
45  # Run Function to Create Effects
46  triangleEffect = createTriangleEffect()
47
```

9. To call the effect, use the setEffect() function. Here is an example with a for loop to increase and decrease the effect: (Lines 49 to 55)

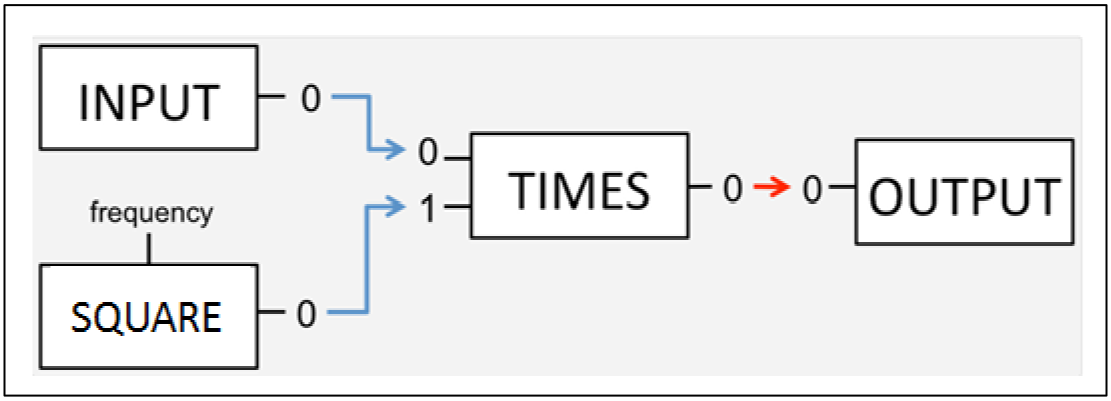


```
51
52  # Apply the Effect
53  for i in range(1, 17):
54      if (i % 2 != 0):
55          setEffect(1, triangleEffect, 'frequency', 20, i, 5000, i + .5)
56          setEffect(1, triangleEffect, 'frequency', 5000, i+1, 20, i + 1.5)
57
```

10. Save and Run the Program. REAPER should look like this:

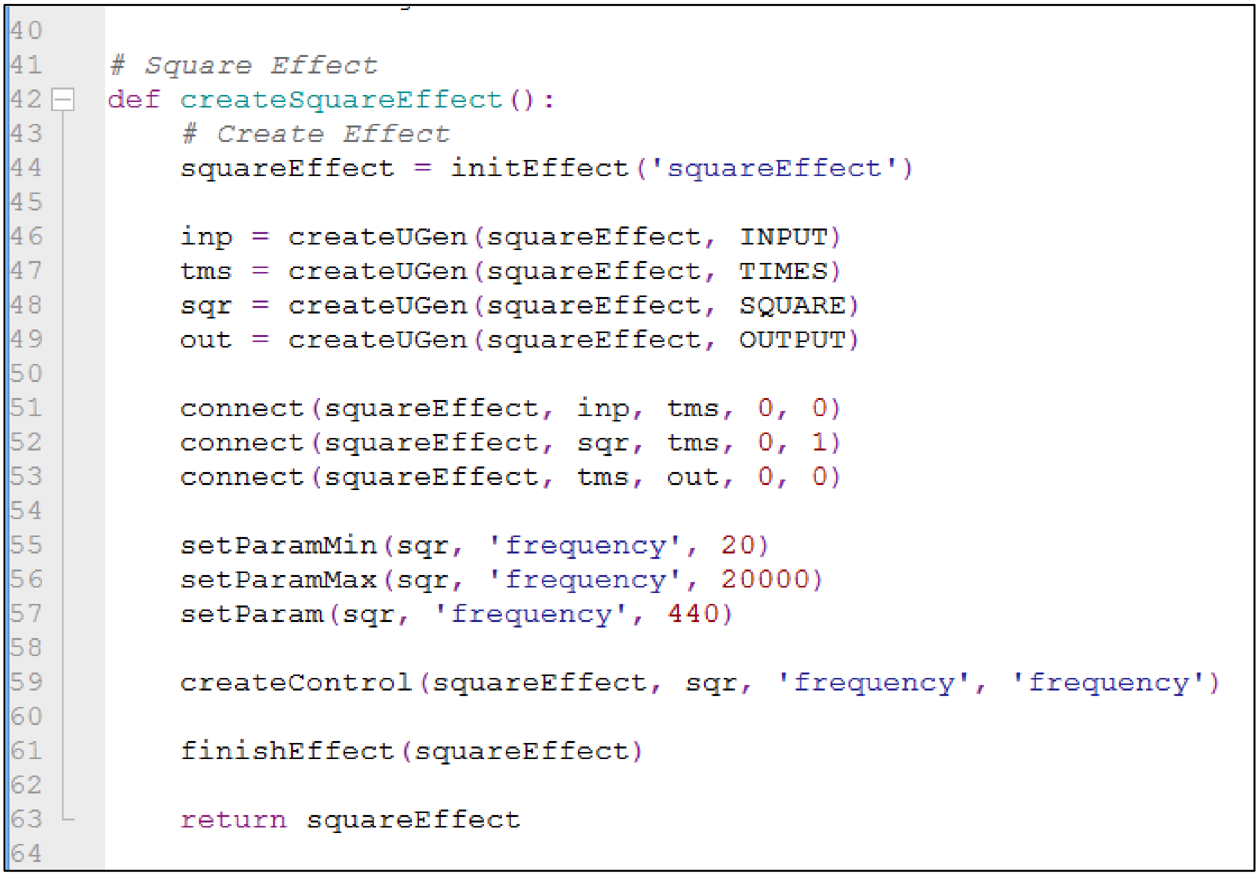


Exercise 3: Writing a Square Effect Function



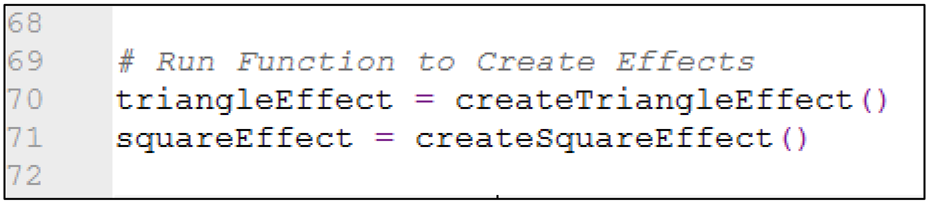
Process:

1. Type the following function to create the Square Effect. Notice how it follows the same format as the Triangle Effect Function



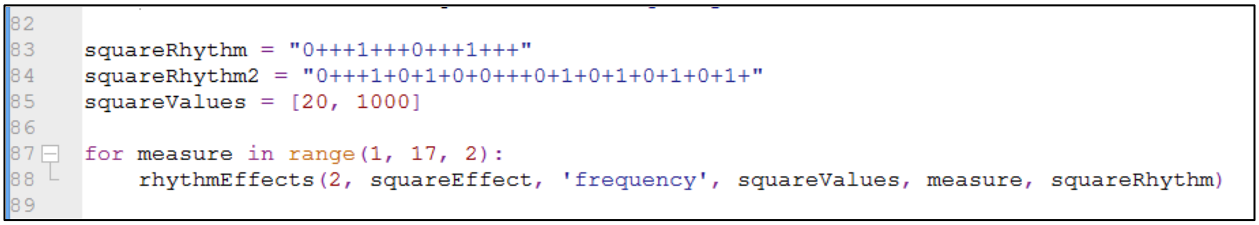
```
# Square Effect
def createSquareEffect():
    # Create Effect
    squareEffect = initEffect('squareEffect')

    inp = createUGen(squareEffect, INPUT)
    tms = createUGen(squareEffect, TIMES)
    sqr = createUGen(squareEffect, SQUARE)
    out = createUGen(squareEffect, OUTPUT)

    connect(squareEffect, inp, tms, 0, 0)
    connect(squareEffect, sqr, tms, 0, 1)
    connect(squareEffect, tms, out, 0, 0)

    setParamMin(sqr, 'frequency', 20)
    setParamMax(sqr, 'frequency', 20000)
    setParam(sqr, 'frequency', 440)

    createControl(squareEffect, sqr, 'frequency', 'frequency')

    finishEffect(squareEffect)

    return squareEffect
```

2. Call the function to create the ‘squareEffect’ effect. (Line 71)



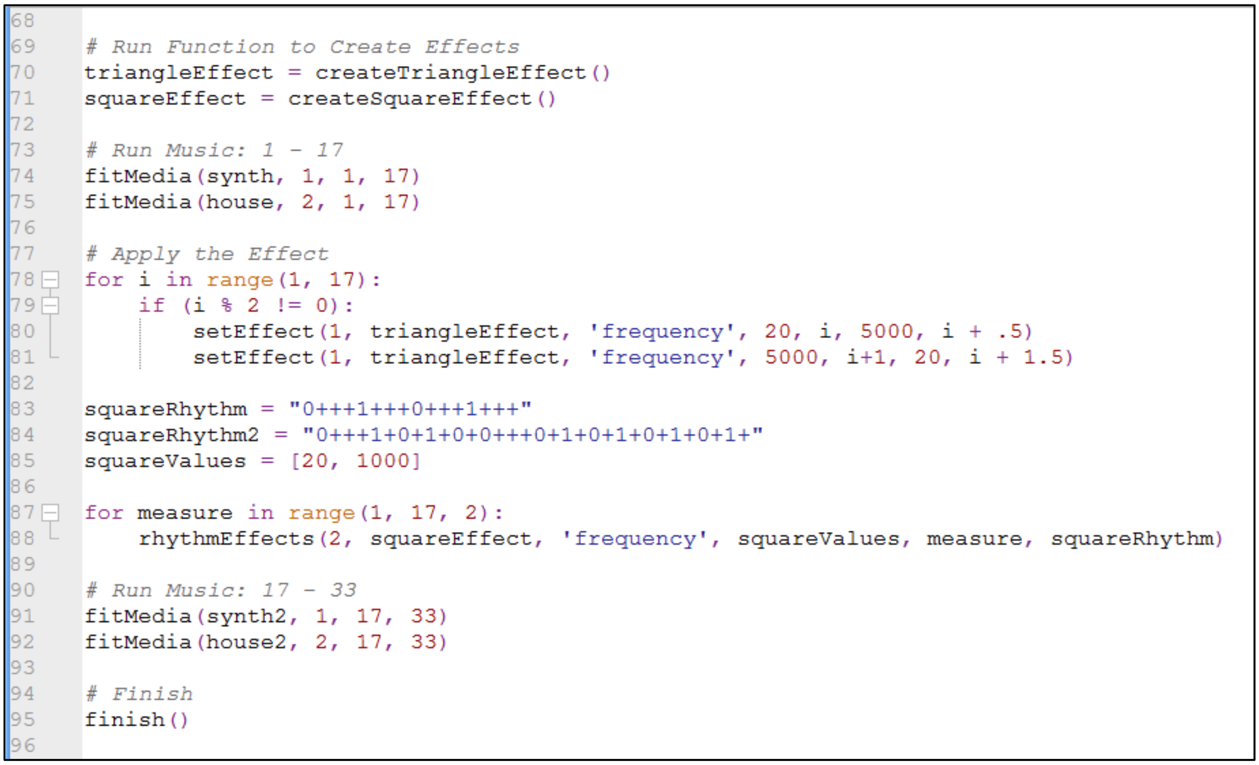
```
68
69    # Run Function to Create Effects
70    triangleEffect = createTriangleEffect()
71    squareEffect = createSquareEffect()
72
```

3. Here an example code running the squareEffect function on track #2 (Lines 82 to 89).



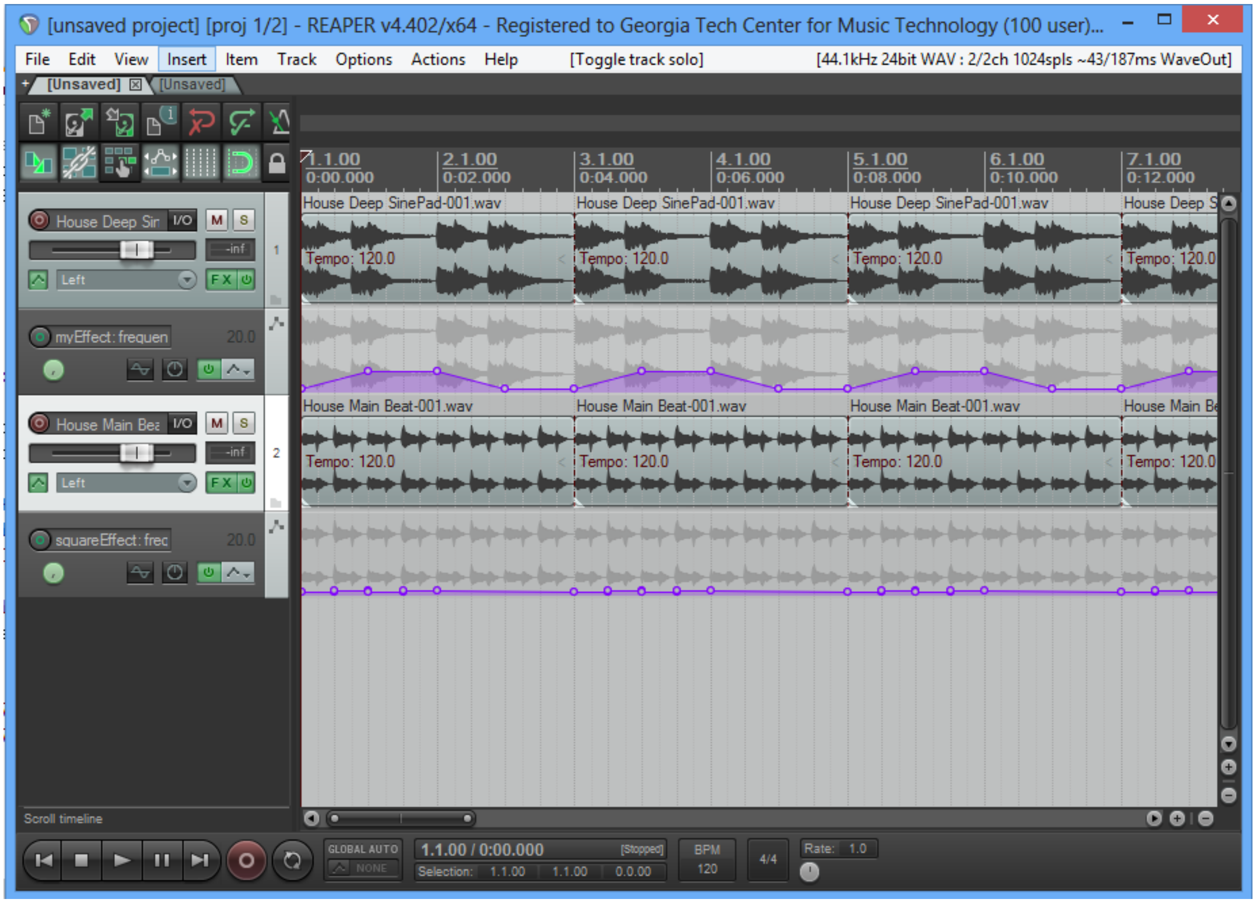
```
82
83    squareRhythm = "0+++1+++0+++1+++"
84    squareRhythm2 = "0+++1+0+1+0+0+++0+1+0+1+0+1+0+1+"
85    squareValues = [20, 1000]
86
87    for measure in range(1, 17, 2):
88        rhythmEffects(2, squareEffect, 'frequency', squareValues, measure, squareRhythm)
89
```

4. The entire music section should look like this:



```
68
69    # Run Function to Create Effects
70    triangleEffect = createTriangleEffect()
71    squareEffect = createSquareEffect()
72
73    # Run Music: 1 - 17
74    fitMedia(synth, 1, 1, 17)
75    fitMedia(house, 2, 1, 17)
76
77    # Apply the Effect
78    for i in range(1, 17):
79        if (i % 2 != 0):
80            setEffect(1, triangleEffect, 'frequency', 20, i, 5000, i + .5)
81            setEffect(1, triangleEffect, 'frequency', 5000, i+1, 20, i + 1.5)
82
83    squareRhythm = "0+++1+++0+++1+++"
84    squareRhythm2 = "0+++1+0+1+0+0+++0+1+0+1+0+1+0+1+"
85    squareValues = [20, 1000]
86
87    for measure in range(1, 17, 2):
88        rhythmEffects(2, squareEffect, 'frequency', squareValues, measure, squareRhythm)
89
90    # Run Music: 17 - 33
91    fitMedia(synth2, 1, 17, 33)
92    fitMedia(house2, 2, 17, 33)
93
94    # Finish
95    finish()
96
```

5. Save and Run in REAPER. Should look like this:


[unsaved project] [proj 1/2] - REAPER v4.402/x64 - Registered to Georgia Tech Center for Music Technology (100 user)...
File
Edit
View
Insert
Item
Track
Options
Actions
Help
[Toggle track solo]
[44.1kHz 24bit WAV : 2/2ch 1024spls ~43/187ms WaveOut]
[Unsaved]
[Unsaved]
1.1.00 0:00.000
2.1.00 0:02.000
3.1.00 0:04.000
4.1.00 0:06.000
5.1.00 0:08.000
6.1.00 0:10.000
7.1.00 0:12.000
House Deep Sir
I/O
M
S
-inf
Left
FX
1
House Deep SinePad-001.wav
Tempo: 120.0
myEffect: frequen
20.0
House Main Bea
2
House Main Beat-001.wav
squareEffect: frec
Scroll timeline
GLOBAL AUTO
NONE
1.1.00 / 0:00.000
[Stopped]
Selection: 1.1.00 1.1.00 0.0.00
BPM
120
4/4
Rate: 1.0