

Forum:[Hack Forums](#)

Thread: [Wave Generator](#)

DudeONym - January 24, 2017, 8:37 am

Keeping it far more functional than the last part, in my mind it helps aid in abstracting away the sound output.

```
Code from __future__ import division
import math
from Graph import *
pi=math.pi
tau=pi*2
hp=pi/2
blend=lambda left,right,control:(left*(control+1)+right*(-control+1))/2 #Slider, when control is 1 only
left is played, when control is -1 only right, control of 0 is equal parts of both waves.
blendwave=lambda left,right,control,rate=16:lambda x:blend(left(x),right(x),control(x/rate))#returns a
wave that is a combination of left and right oscilating by a third wave 'control' at 1/rate the wave
frequency.
tune=lambda wave,freq,samplerate=44100:lambda x:wave(x/samplerate*tau*freq) #Tune a wave to
a given frequency and divide by the sample rate in HZ
sin=math.sin
saw=lambda x:((x-pi)%tau-pi)/pi
tri=lambda x: abs((x-hp)%tau-pi)/hp-1
sqr=lambda x:1 if int(x/pi)%2==0 else -1
add=lambda *w:lambda x:sum(map(lambda y:y(x),w))/len(w) #Constuct a wave from the mean of n
waves
overtone=lambda wave,octaves=2,dec=2:lambda x:sum(map(lambda
y:wave(x*(2**y))/dec**y,range(octaves)))/sum(map(lambda y:1/dec**y,range(octaves)))##Construct
a wave in multiple octaves, dividing by a volume of dec each successive octave
GraphWave(blendwave(overtone(sin),overtone(add(tri,saw,sqr)),sin),16) #An example bringing
every function together into one not to pleasent sounding, but educational, wave.Donate money to
my paypal for me to pay for some of my $8USD VPS bill and I can help out with some python
work/tuition ?
More to come.
```

```
Code def PitchFunc(Frequency,Magnitude,Value):
    return Frequency*(Magnitude**(Value+1)/Magnitude)
```

```
class Envelope:
    pass
```

```
class Flat(Envelope):
    def __init__(self,n):
        self.n=n
        self.length=1
    def make(self,Length,SampleRate=44100):
        return [self.n]*Length
```

```
def Pos(start,end):
    return lambda x:(x-start)/(end-start)
```

```

def Mid(a,b,r):
    return a+(b-a)*r

class Envelope:
    pass
class ADSR_Norm(Envelope):
    def __init__(self,Attack,Decay,Sustain,Release):
        self.Attack=Attack
        self.Decay=Decay
        self.Sustain=Sustain
        self.Release=Release
    def make(self,Length,SampleRate=44100):
        Env=[]

        points=[self.Attack/1000*SampleRate,(self.Attack+self.Decay)/1000*SampleRate,(self.Attack+s
elf.Decay+self.Release)/1000*SampleRate,(Length+self.Release)/1000*SampleRate]
        AtkFunc=Pos(0,points[0])
        DecFunc=Pos(points[0],points[1])
        RelFunc=Pos(0,self.Release/1000*SampleRate)
        for i in range(Length):
            if i <= points[0]:
                val=AtkFunc(i)
            elif i <=points[1]:
                val=Mid(1,self.Sustain,DecFunc(i))
            else:
                val=self.Sustain
            Env.append(val)
        for i in range(int(self.Release/1000*SampleRate)):
            Env.append(Mid(val,0,RelFunc(i)))
        return Env

```

Some object oriented stuff for creating enveloped sounds, tuned to seconds to be sample rate independent.

Codestroy - January 28, 2017, 7:50 am

Thanks good job ?

TheWarOwl - January 28, 2017, 9:39 am

Just a heads up because I got a warning for this myself. It's against hf rules to ask for donations, you should probably take that out.

DudeONym - March 3, 2017, 11:04 pm

(Jan 28, 2017 10:39 AM)Piero-Bird Wrote: [\[CITE\]](#) Just a heads up because I got a warning for this myself. It's against hf rules to ask for donations, you should probably take that out.

Thank you!

I actually did read the rules once or twice, but I spaced on it.

It's too late to edit it, if a mod sees this, please do such for me. Thanks and sorry :(