Plugging VSTs into Python

Tim Thompson
tjt@nosuch.com

Outline

- nosuch.vstutil – a Python module for manipulating:
  - audio devices
  - audio snippets
  - VST plugins
- Example code
- Example art – Radio Free Quasar
- Example instrument – w/multitouch and graphics

Classes in nosuch.vstutil

- AudioSnippet
  - init parameters: filename or audiodata
  - methods: duration, stretch
- AudioLoop
  - init parameters: snippet, offset, length, loops
  - methods: restart, setlength, setoffset
- PaAudioDevice
  - methods: abort, attach, close, is_active, open, reset_reduction, start, stop

Classes in nosuch.vstutil

- VstPlugin
  - init parameters: dll
  - methods: can_receive_midi, is_synth, is_vst2, name, num_inputs, num_outputs, num_params, num_programs, param_label, param_display, param_name, program_name, send_midi, get_enable, set_enable, set_param, set_program, set_input

Implementation details

- Portaudio
  - provides audio device interface
- libsndfile.dll
  - provides sound file reading
- pyrex
  - generates python/C interface code

Basic use of VstPlugin

- Instantiate it:

```
v = VstPlugin(dll="ringmod.dll")
```

- Connect its input to the output of other plugins

```
v.setinput(v2)
```

- Randomize its parameters

```
n = v.num_params()
for i in range(n):
    v.set_param(i,random())
```

Looping and playing sound

```
#!/usr/bin/env python
import time
import nosuch.vstutil
import sys

a = PaAudioDevice()
sound = AudioSnippet(filename="winsound1.wav")
loop1 = AudioLoop(sound1,loops=-1)  # loop forever
a.open()
a.start()
a.attach(loop1)
time.sleep(10)      # sound is heard
a.stop()
a.close()
sys.exit(0)
```

Connecting a VST synth and effect

```
effect = VstPlugin(dll="BJ Ringmodulator.dll")
synth = VstPlugin(dll="StrataVar.dll")

# Connect output of synth to effec
effect.setinput(synth)

# Open audio and connect effect's output to it
a.open()
a.start()
a.attach(effect)

# Send random MIDI notes to synth
# while randomizing parameters of both VSTs
for i in range(100):
   time.sleep(2.0)
   pitch = int(random() * 128) % 128
   vstrandparam(synth)
   vstrandparam(effect)   # see next slide
   synth.send_midi(1,pitch,8000,80)
```

Utility functions

```
def vstrandparam(v):
   for i in range(v.numParams()):
       v.set_param(i,random())

def vstinfo(v):
   print "Is VST2 = ",v.is_vst2()
   print "Is synth = ",v.is_synth()
   print "numParams = ",v.numParams()
   print "numInputs = ",v.numInputs()
   print "numOutputs = ",v.numOutputs()
   print "can_receive_midi = ",v.can_receive_midi()
   print "can_send_midi = ",v.can_send_midi()
```

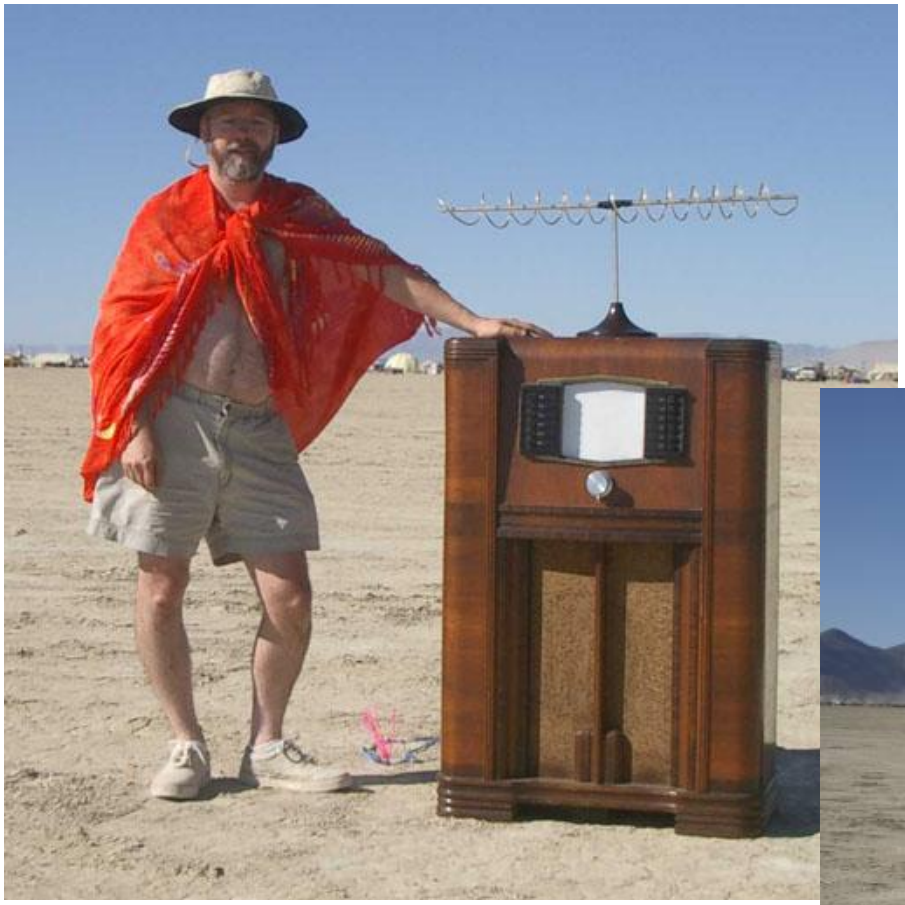
Demos

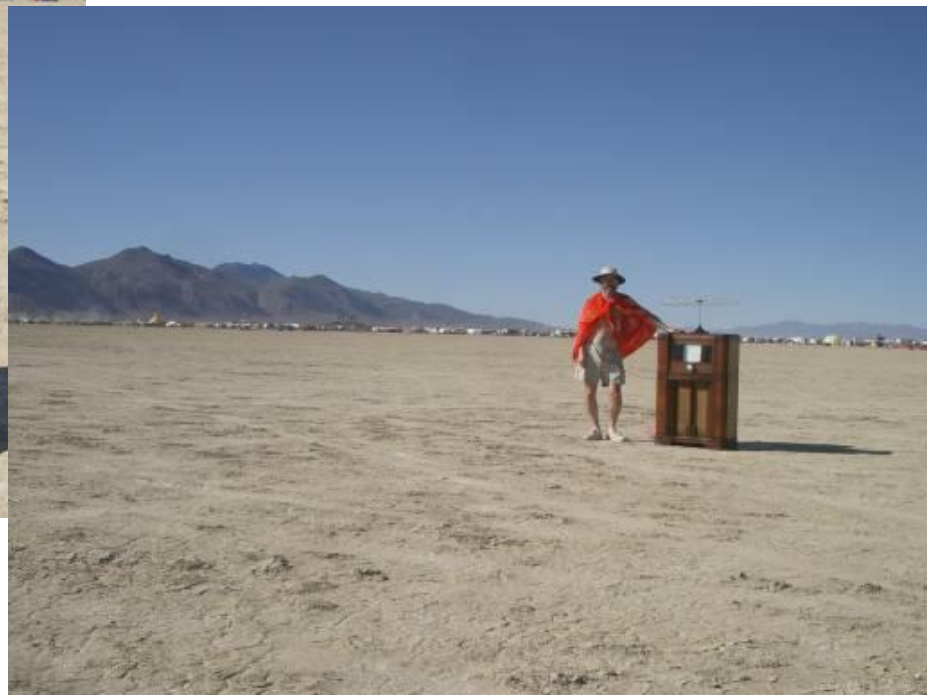
- demo1
  - play VST synth through VST effect and randomize parameters
- demo2
  - play wave file through VST effect and randomize parameters
- demo3
  - play VST synth through VST effect with LFOs attached to parameters (and periodically randomize)

Radio Free Quasar

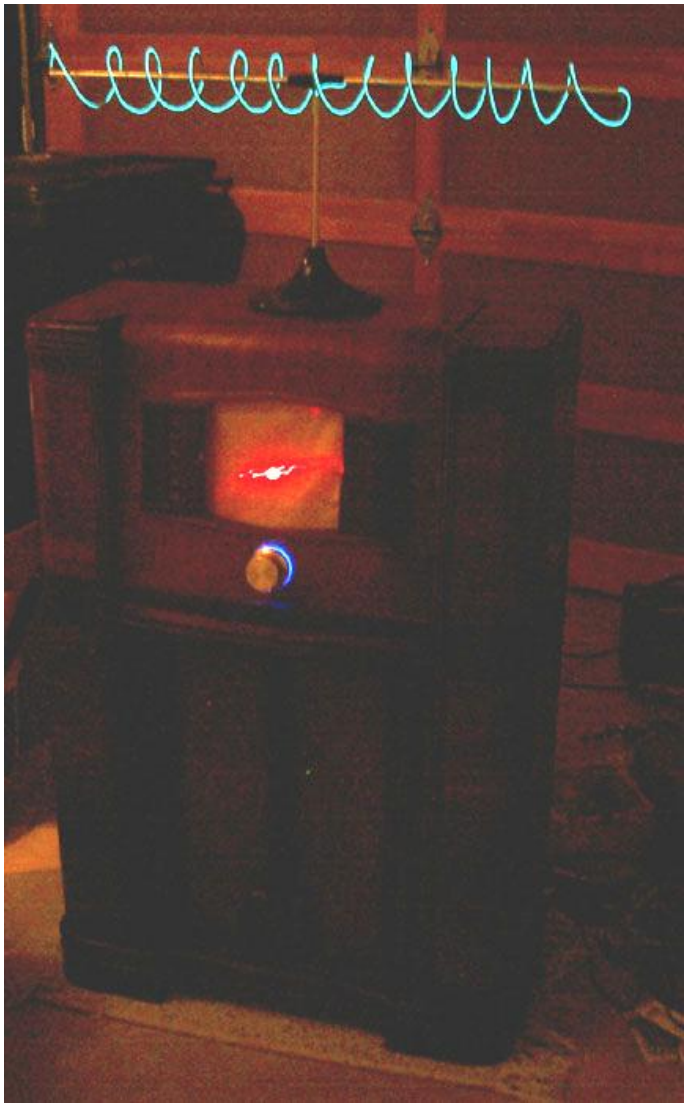
- Art installation for Burning Man 2004
- Antique radio
- Computer generating audio
- Laser generating graphics
- Big knob for control

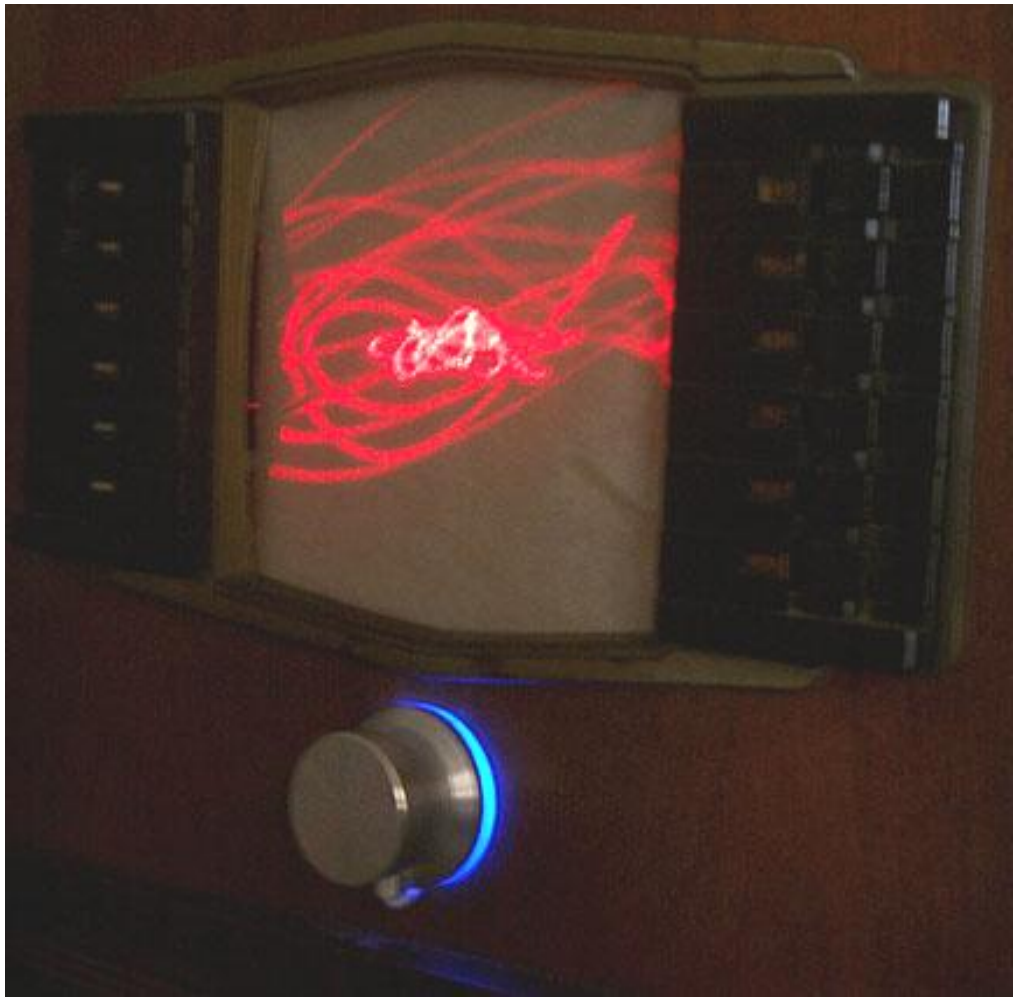
Radio Free Quasar at Burning Man



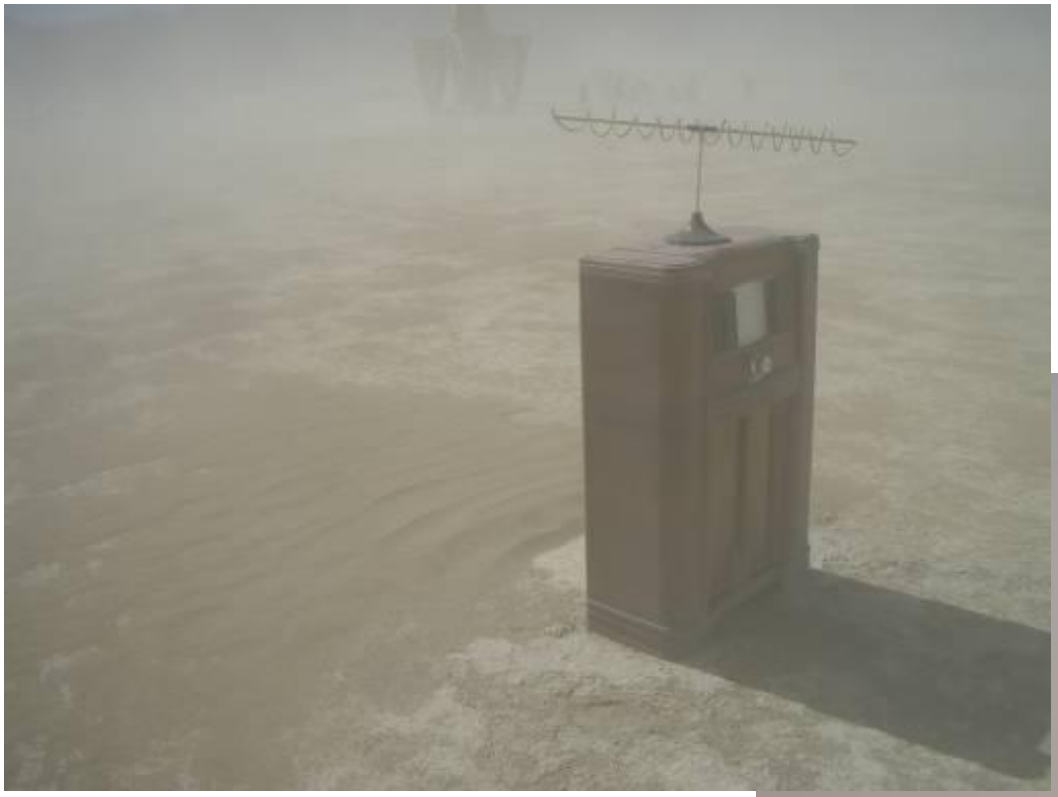


Radio Free Quasar

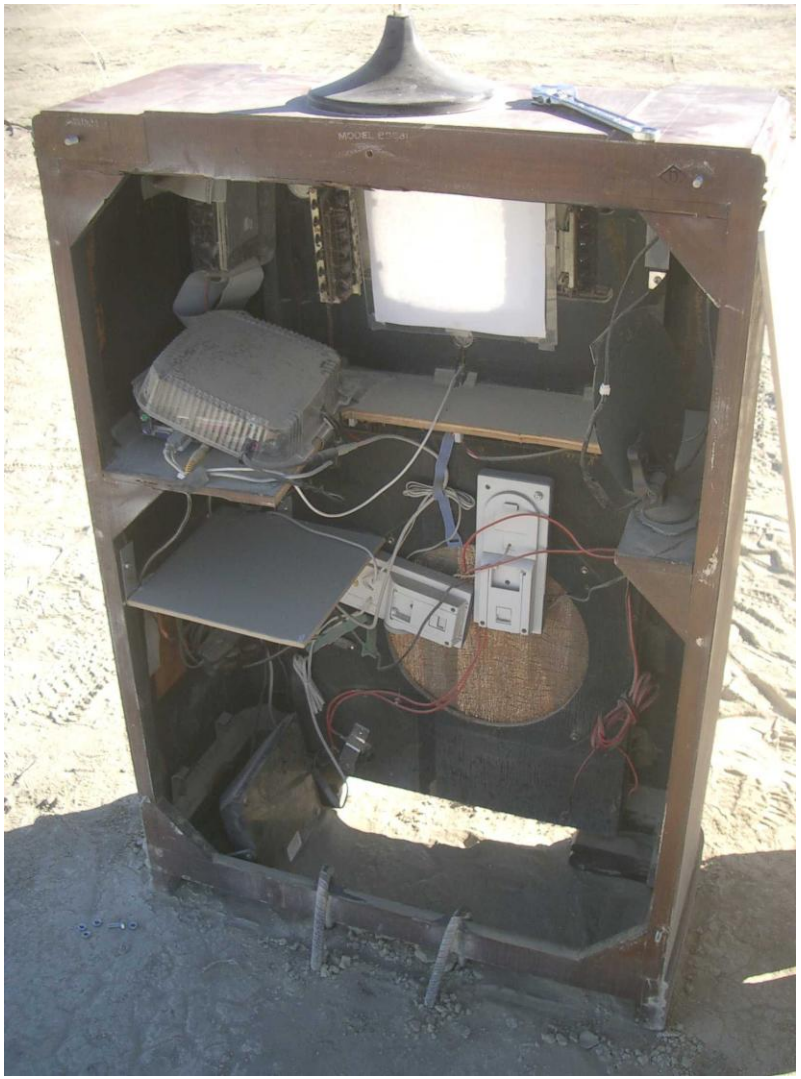


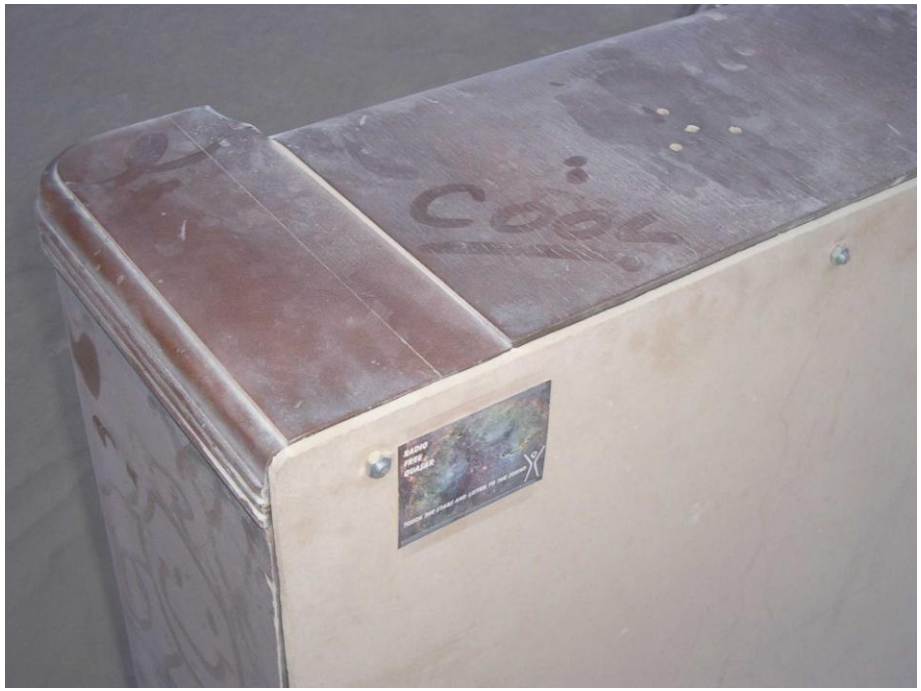


Radio Free Quasar at Burning Man

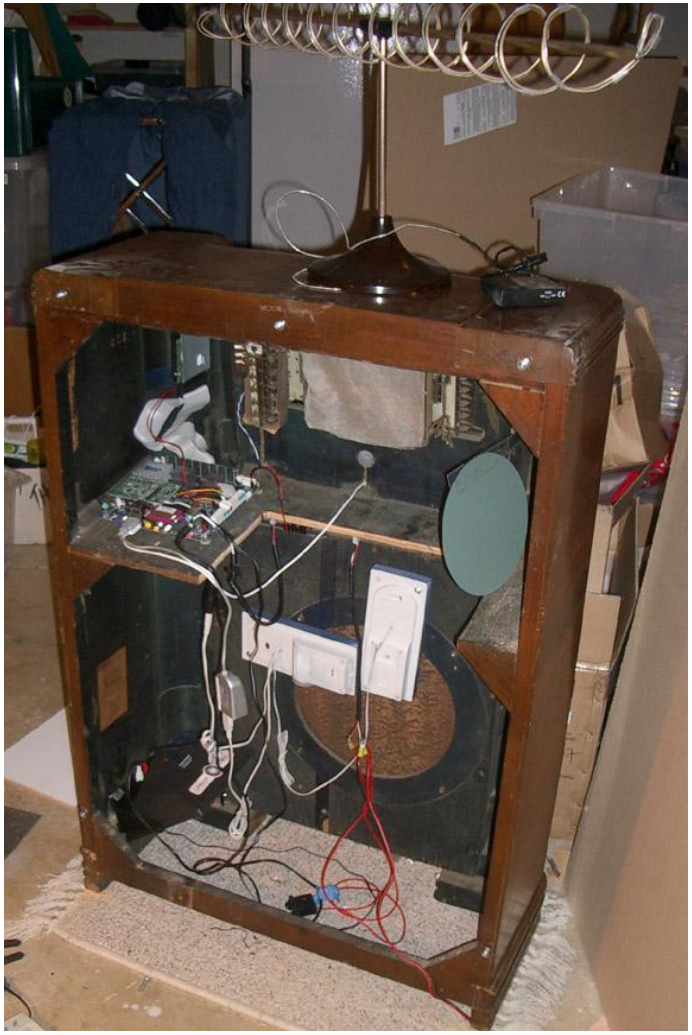


Radio Free Quasar at Burning Man





Radio Free Quasar



Radio Free Quasar – the pieces

- 10 robust VST plugins chained serially
- Collection of WAV files
- Python program:
  - selects wave files
  - enables/disables/randomizes VST plugins
  - allows interactive control from keyboard
- Big knob on radio sends keypresses
- Automatic randomization if no user input

Radio Free Quasar – hardware

- Windows XP
- Mini-ITX motherboard (fanless, low power)
- DC power supply
- Power sequencer
- USB-powered speakers
- USB knob (Griffin Powermate)
- Laservibe laser (driven by computer’s audio output)
- EL-wire antenna

Radio Free Quasar – interactive control

r	: randomize wave file and plugin parameters
w	: change to a different wave file
0-9	: randomize parameters of plugin N
d	: disable all plugins
e	: enable all plugins
s	: save current parameters
p	: select previously saved parameters

Throwing together VSTs, graphics, and multitouch

- ergo (events routed to graphical objects)
  - Python-based visual performance tool
  - Multitouch (iGesture) pads used to draw graphics
  - Events from MIDI devices also trigger graphics
  - MIDI controller used to control graphic parameters
- vstutil library creates VST synth and two effects
  - Multitouch pad controls notes and effects parameters
  - Buttons on MIDI controller can randomize parameters

Other libraries of interest

- nosuch.midiutil
  - MIDI device and data manipulation
  - Uses updated pyportmidi (pypm) module, which provides an interface to the portmidi library.
- nosuch.fingerutil
  - Fingerworks iGesture (multitouch) interface

Status and Availability

- nosuch.vstutil
  - Simple packaging, Windows-only
  - Public domain
  - Downloadable at **`http://nosuch.com/tjt`**
- nosuch.midiutil and nosuch.fingerutil
  - contact **`tjt@nosuch.com`**

Plugging VSTs into Python

Tim Thompson
tjt@nosuch.com