```
# Example engaging class projects
# Brygg Ullmer, Clemson University
# Begun 2025-02-20
import sys, os, traceback
import spectra,  pygame
from enoMidiController import *
from projBase          import *
from projPgzBase       import *

################### coursesPgz Accordion + MIDI ###################

class ProjMidi(ProjectsPgzBase):
  emc               = None #enodia midi controller
  ################## constructor, error ##################
  def __init__(self, **kwargs):
    self.__dict__.update(kwargs) #allow class fields to be passed in constructor
    super().__init__()

    self.initSliders()
    self.midiClearLights()
    self.midiLightInit()

  def err(self, msg): print("ProjMidi error:", msg); traceback.print_exc()
  def msg(self, msg): print("ProjMidi msg:",   msg)
  def midiClearLights(self):
    for i in range(64): self.emc.midiOut.note_on(i, 0, 0)

  def midiLightInit(self):
    colors = [3,4,5,13,21,29,37,45,53,61]
    for i in colors: self.emc.midiOut.note_on(i, i, 6)
    #for i in range(12): self.emc.midiOut.note_on(i, i, 6)

  def midiCB(self, control, arg):
    print("hello class")
    try:
      if self.verbose: print("cpgzm midicb: ", str(control), str(arg))

      if control[0] == 's': #slider
        whichSlider = int(control[-1]) - 1 #control is "s1", "s2", etc.
        whichVal    = int(arg)
        mappedVal   = self.mapSliderVal(whichSlider, whichVal)
        print("slider", str(whichSlider), str(whichVal))

        self.sliderValDict[whichSlider] = self.sliderFullrangeV - whichVal
        #self.sliderValDict[whichSlider] = mappedVal
        #self.assignColumnIdx(whichSlider, mappedVal)
      else:
        print(control, arg)
        if control=="a8" and arg == 127: print("dance")
        if control=="a7" and arg == 127: print("p1 dances")
        if control=="a6" and arg == 127: print("p0 dances")

    except: self.err("midiCb " + str([control, arg]))
  ################## excised ##################
  def initSliders(self):
  def drawSliders(self):
  def drawSlider(self, whichSlider):  ...
  def setAkaiColorIdxCoord(self, colorIdx, x, y, colorBright=6): ...
  def mapAkaiCoord(self, x, y): ...
  def mapSliderVal(self, whichCol, whichSliderVal): ...
  def updateMatrixColors(self): ...

  def draw(self, screen):          self.drawSliders()
  def on_mouse_down(self, pos):    pass #cpgz.on_mouse_down(pos)
  def update(self):                self.emc.pollMidi()

### end ###
```

```
        print("slider", str(whichSlider), str(whichVal))     #H

        self.sliderValDict[whichSlider] = self.sliderFullrangeV - whichVal
        #self.sliderValDict[whichSlider] = mappedVal
        #self.assignColumnIdx(whichSlider, mappedVal)
      else:
        print(control, arg)
        if control=="a8" and arg == 127: print("dance")      #I
        if control=="a7" and arg == 127: print("p1 dances")
        if control=="a6" and arg == 127: print("p0 dances")

    except: self.err("midiCb " + str([control, arg]))

  ################## excised ##################

  def initSliders(self):
  def drawSliders(self):
  def drawSlider(self, whichSlider):  ...
  def setAkaiColorIdxCoord(self, colorIdx, x, y, colorBright=6): ...
  def mapAkaiCoord(self, x, y): ...
  def mapSliderVal(self, whichCol, whichSliderVal): ...
  def updateMatrixColors(self): ...

  def draw(self, screen):         self.drawSliders()
  def on_mouse_down(self, pos):   pass #cpgz.on_mouse_down(pos)
  def update(self):               self.emc.pollMidi()

### end ###
```