py4kids (https://github.com/wgong/py4kids)

PyGame (http://www.pygame.org/) - Play to learn & Learn to play

- In Lesson 9 - Class and Object (https://github.com/wgong/py4kids/blob/master/lesson-09-class-objects/lesson-09.ipynb): we learn Fist and Chimp in a simple game
- In Lesson 10 - Python Review (https://github.com/wgong/py4kids/blob/master/lesson-10-python-review/lesson-10.ipynb): we review the complete program chimp.py as a case study and learn how to run pygame in Jupyter Notebook
- In this lesson, we continue learning PyGame by going over a PyGame tutorial (https://eli.thegreenplace.net/2008/12/13/writing-a-game-in-python-with-pygame-part-i/).

In [1]:
```
from jyquickhelper import add_notebook_menu
add_notebook_menu()
```

Out[1]:

PyGame Examples

Chimp to Dog

Trivia changes to make a new game

In [4]:
```
#!/usr/bin/env python
"""
This simple example is used for the line-by-line tutorial
that comes with pygame. It is based on a 'popular' web banner.
Note there are comments here, but for the full explanation,
follow along in the tutorial.
"""


#Import Modules
import os, pygame
from pygame.locals import *
from pygame.compat import geterror

if not pygame.font: print ('Warning, fonts disabled')
if not pygame.mixer: print ('Warning, sound disabled')

FRAMES_PER_SEC = 100

# main_dir = os.path.split(os.path.abspath(__file__))[0]
# data_dir = os.path.join(main_dir, 'data')
data_dir = "../data"

#functions to create our resources
def load_image(name, colorkey=None):
    fullname = os.path.join(data_dir, name)
    try:
        image = pygame.image.load(fullname)
    except pygame.error:
        print ('Cannot load image:', fullname)
        raise SystemExit(str(geterror()))
    image = image.convert()
    if colorkey is not None:
        if colorkey is -1:
            colorkey = image.get_at((0,0))
        image.set_colorkey(colorkey, RLEACCEL)
    return image, image.get_rect()

def load_sound(name):
    class NoneSound:
        def play(self): pass
    if not pygame.mixer or not pygame.mixer.get_init():
```

```
        return NoneSound()
    fullname = os.path.join(data_dir, name)
    try:
        sound = pygame.mixer.Sound(fullname)
    except pygame.error:
        print ('Cannot load sound: %s' % fullname)
        raise SystemExit(str(geterror()))
    return sound


#classes for our game objects
class Fist(pygame.sprite.Sprite):
    """moves a clenched fist on the screen, following the mouse"""
    def __init__(self):
        pygame.sprite.Sprite.__init__(self) #call Sprite initializer
        #self.image, self.rect = load_image('fist.bmp', -1)
        self.image, self.rect = load_image('bone.jpg', -1)
        self.punching = 0

    def update(self):
        "move the fist based on the mouse position"
        pos = pygame.mouse.get_pos()
        self.rect.midtop = pos
        if self.punching:
            self.rect.move_ip(5, 10)

    def punch(self, target):
        "returns true if the fist collides with the target"
        if not self.punching:
            self.punching = 1
            hitbox = self.rect.inflate(-5, -5)
            return hitbox.colliderect(target.rect)

    def unpunch(self):
        "called to pull the fist back"
        self.punching = 0


class Chimp(pygame.sprite.Sprite):
    """moves a monkey critter across the screen. it can spin the
       monkey when it is punched."""
    def __init__(self):
        pygame.sprite.Sprite.__init__(self) #call Sprite intializer
```

```
        #self.image, self.rect = load_image('chimp.bmp', -1)
        self.image, self.rect = load_image('dog.PNG', -1)
        screen = pygame.display.get_surface()
        self.area = screen.get_rect()
        self.rect.topleft = 10, 50
        self.move = 9
        self.dizzy = 0

    def update(self):
        "walk or spin, depending on the monkeys state"
        if self.dizzy:
            self._spin()
        else:
            self._walk()

    def _walk(self):
        "move the monkey across the screen, and turn at the ends"
        newpos = self.rect.move((self.move, 0))
        if self.rect.left < self.area.left or \
            self.rect.right > self.area.right:
            self.move = -self.move
            newpos = self.rect.move((self.move, 0))
            self.image = pygame.transform.flip(self.image, 1, 0)
        self.rect = newpos

    def _spin(self):
        "spin the monkey image"
        center = self.rect.center
        self.dizzy = self.dizzy + 12
        if self.dizzy >= 360:
            self.dizzy = 0
            self.image = self.original
        else:
            rotate = pygame.transform.rotate
            self.image = rotate(self.original, self.dizzy)
        self.rect = self.image.get_rect(center=center)

    def punched(self):
        "this will cause the monkey to start spinning"
        if not self.dizzy:
            self.dizzy = 1
            self.original = self.image
```

```
def main():
    """this function is called when the program starts.
       it initializes everything it needs, then runs in
       a loop until the function returns."""
#Initialize Everything
    pygame.init()
    screen = pygame.display.set_mode((468, 160))
    pygame.display.set_caption('Dog bark')
    pygame.mouse.set_visible(0)

#Create The Backgound
    background = pygame.Surface(screen.get_size())
    background = background.convert()
    background.fill((250, 250, 250))

#Put Text On The Background, Centered
    if pygame.font:
        font = pygame.font.Font(None, 36)
        text = font.render("Dog bark", 1, (210, 10, 10)) # red caption
        textpos = text.get_rect(centerx=background.get_width()/2)
        background.blit(text, textpos)

#Display The Background
    screen.blit(background, (0, 0))
    pygame.display.flip()

#Prepare Game Objects
    clock = pygame.time.Clock()
    #whiff_sound = load_sound('whiff.wav')
    whiff_sound = load_sound('bark-1.wav')
    #punch_sound = load_sound('punch.wav')
    punch_sound = load_sound('bark-1.wav')
    chimp = Chimp()
    fist = Fist()
    allsprites = pygame.sprite.RenderPlain((fist, chimp))


#Main Loop
    going = True
    while going:
        clock.tick(FRAMES_PER_SEC)
```

```
        #Handle Input Events
        for event in pygame.event.get():
            if event.type == QUIT:
                going = False
            elif event.type == KEYDOWN and event.key == K_ESCAPE:
                going = False
            elif event.type == MOUSEBUTTONDOWN:
                if fist.punch(chimp):
                    punch_sound.play() #punch
                    chimp.punched()
                else:
                    whiff_sound.play() #miss
            elif event.type == MOUSEBUTTONUP:
                fist.unpunch()

        allsprites.update()

        #Draw Everything
        screen.blit(background, (0, 0))
        allsprites.draw(screen)
        pygame.display.flip()

    pygame.quit()

#Game Over


#this calls the 'main' function when this script is executed

if False:
    if __name__ == '__main__':
        main()
```

In [5]:
```
main()
```

PyGame Components

http://www.pygame.org/docs/index.html (http://www.pygame.org/docs/index.html)

Visual (http://www.pygame.org/docs/ref/image.html) : Pixel, Color, Image

- display
- PixelArray
- Color
- image
- Rect
- Surface
- gfxdraw

Text (http://www.pygame.org/docs/ref/font.html) : Font

- font

Sound (http://www.pygame.org/docs/ref/mixer.html): mixer, midi, music

- mixer
- midi
- music

Event (http://www.pygame.org/docs/ref/event.html) : mouse, key, joystick

- Event
- mouse
- key

Math (http://www.pygame.org/docs/ref/math.html) : 2D/3D vector, matrix, transformation

- math
- numpy

Game (http://www.pygame.org/docs/ref/camera.html) : Lights, Camera, Action

- Clock

- camera
- openGL

PyGame Demos

Show event

In [10]:
```
#!/usr/bin/env python

"""Eventlist is a sloppy style of pygame, but is a handy
tool for learning about pygame events and input. At the
top of the screen are the state of several device values,
and a scrolling list of events are displayed on the bottom.

This is not quality 'ui' code at all, but you can see how
to implement very non-interactive status displays, or even
a crude text output control.
"""
import os.path

from pygame import *
import pygame.mixer

ImgOnOff = []
Font = None
LastKey = None
playing_music = 0


def showtext(win, pos, text, color, bgcolor):
    textimg = Font.render(text, 1, color, bgcolor)
    win.blit(textimg, pos)
    return pos[0] + textimg.get_width() + 5, pos[1]


def drawstatus(win):
    bgcolor = 50, 150, 50
    win.fill(bgcolor, (0, 0, 640, 120))
    win.blit(Font.render('Status Area', 1, (155, 155, 155), bgcolor), (2, 2))

    pos = showtext(win, (10, 30), 'Mouse Focus', (255, 255, 255), bgcolor)
    win.blit(ImgOnOff[mouse.get_focused()], pos)

    pos = showtext(win, (330, 30), 'Keyboard Focus', (255, 255, 255), bgcolor)
    win.blit(ImgOnOff[key.get_focused()], pos)

    pos = showtext(win, (10, 60), 'Mouse Position', (255, 255, 255), bgcolor)
    p = '%s, %s' % mouse.get_pos()
    pos = showtext(win, pos, p, bgcolor, (255, 255, 55))
```

```
    pos = showtext(win, (330, 60), 'Last Keypress', (255, 255, 255), bgcolor)
    if LastKey:
        p = '%d, %s' % (LastKey, key.name(LastKey))
    else:
        p = 'None'
    pos = showtext(win, pos, p, bgcolor, (255, 255, 55))

    pos = showtext(win, (10, 90), 'Input Grabbed', (255, 255, 255), bgcolor)
    win.blit(ImgOnOff[event.get_grab()], pos)

    # add hit to control background music
    pos = showtext(win, (330, 90), 'Toggle m-key to pause/play music', (255, 1, 1), bgcolor)
    #win.blit(ImgOnOff[event.get_grab()], pos)


def drawhistory(win, history):
    win.blit(Font.render('Event History Area', 1, (155, 155, 155), (0,0,0)), (2, 132))
    ypos = 450
    h = list(history)
    h.reverse()
    for line in h:
        r = win.blit(line, (10, ypos))
        win.fill(0, (r.right, r.top, 620, r.height))
        ypos -= Font.get_height()


def main():
    init()

    if mixer and not mixer.get_init():
        print ('Warning, no sound')
        pygame.mixer = None

    # play background music
    if mixer:
        music = os.path.join('data', 'fur-elise.mp3')
        mixer.music.load(music)
        mixer.music.play(-1)
        playing_music = 1

    win = display.set_mode((640, 480), RESIZABLE)
    display.set_caption("Mouse Focus Workout")
```

```
    global Font
    Font = font.Font(None, 26)

    global ImgOnOff
    ImgOnOff.append(Font.render("Off", 1, (0, 0, 0), (255, 50, 50)))
    ImgOnOff.append(Font.render("On", 1, (0, 0, 0), (50, 255, 50)))

    history = []

    #let's turn on the joysticks just so we can play with em
    for x in range(joystick.get_count()):
        j = joystick.Joystick(x)
        j.init()
        txt = 'Enabled joystick: ' + j.get_name()
        img = Font.render(txt, 1, (50, 200, 50), (0, 0, 0))
        history.append(img)
    if not joystick.get_count():
        img = Font.render('No Joysticks to Initialize', 1, (50, 200, 50), (0, 0, 0))
        history.append(img)

    going = True
    while going:
        for e in event.get():
            if e.type == QUIT:
                going = False
            if e.type == KEYDOWN:
                if e.key == K_ESCAPE:
                    going = False
                else:
                    global LastKey
                    LastKey = e.key

                # use K_m to toggle music
                if e.key == K_m:
                    if playing_music:
                        mixer.music.pause()
                        playing_music = 0
                    else:
                        mixer.music.unpause()
                        playing_music = 1

            if e.type == MOUSEBUTTONDOWN:
```

```
                event.set_grab(1)
            elif e.type == MOUSEBUTTONUP:
                event.set_grab(0)
            if e.type == VIDEORESIZE:
                win = display.set_mode(e.size, RESIZABLE)

            if e.type != MOUSEMOTION:
                txt = '%s: %s' % (event.event_name(e.type), e.dict)
                img = Font.render(txt, 1, (50, 200, 150), (0, 0, 0))
                history.append(img)
                # keep last 15 events
                history = history[-15:]



        drawstatus(win)
        drawhistory(win, history)

        display.flip()
        time.wait(10)

    quit()

if False:
    if __name__ == '__main__':
        main()
```

In [11]:
```
main()
```

PyGame Tutorials

PyGame Tutorials — Wiki (http://www.pygame.org/wiki/tutorials)

PyGame Books / References

- https://gamedevelopment.tutsplus.com/tutorials/how-to-learn-pygame--cms-24184 (https://gamedevelopment.tutsplus.com/tutorials/how-to-learn-pygame--cms-24184)
- PyGame API Documentation (http://www.pygame.org/docs/ref/examples.html)

In [ ]:

In [ ]: