In [1]:

```
import pygame as pg
from pygame.locals import *
from time import sleep
from random import randint
import numpy as np
import sys
```

```
pygame 1.9.6
Hello from the pygame community. https://www.pygame.org/contribute.ht
ml (https://www.pygame.org/contribute.html)
```

https://www.youtube.com/watch?v=C6jJg9Zan7w (https://www.youtube.com/watch?v=C6jJg9Zan7w)

Ideias:

- Tela inicial
- Ajeitar sistema gráfico de Scores
- Organizar mudança de Velocidade
- se der:
  - variar os movimentos de acordo de onde bater na pá.
- música -

In [2]:

```
class colors:
    def __init__(self):
        self.white=(255,255,255)
        self.black=(0,0,0)
        self.red=(255,0,0)
        self.green=(0,255,0)
        self.blue=(0,0,255)
cor= colors()
cor.white
class Default:
    window=[1200,600]
class Paddle:
    cor=colors()
    window=Default.window
    larg=15
    jump=window[1]*0.1 #pulo da pá
    compr=window[1]*0.25#comprimento da pá
    p=pg.Surface([larg,compr])
    p.fill(cor.white)
    x=0
    y=4*jump#window[1]/2
    p_pos=(x,y)
    def mov_up(self):
        if self.y >= self.compr/3:
            self.y-=self.jump
            self.p_pos=(self.x,self.y)
    def mov_down(self):
        if self.y <= self.window[1]-self.compr*1.3:
            self.y+=self.jump
            self.p_pos=(self.x,self.y)
######################################################################33
class Ball:
    cor=colors()
    #scor=Score()
    window=Default.window
    window=[window[0]-10,window[1]-10]
    tam_b=13
    ball=pg.Surface([tam_b,tam_b])
    ball.fill(cor.blue)

    b_pos=[window[0]/2,window[1]/2]
    out_left=0
    out_right=0
    out_total=out_left+out_right
    incr_speed=1#(scor.scr+1)#incrementar velocidade
    Inicial_speed=[2,-2] #velocidade inical
    ball_runing=1
    k=5
    speed=[k,-k]#np.array(Inicial_speed)*incr_speed*ball_runing
    def mov_ball(self):
        x=self.b_pos[0]+self.speed[0]
        y=self.b_pos[1]+int(self.speed[1])
        self.b_pos=[x,y]
    def reset_ball(self,loc_pont):#loc_pont: relacionado a x; 0:esquerda; 1: direit
        self.b_pos=[self.window[0]/2,self.window[1]/2]
        vx = self.k * loc_pont
        vy=self.k*self.gerar_2num()
        self.speed=[vx,vy]
    def Colid_ball(self,pos_paddle1,pos_paddle2):
        P=Paddle()
        L=P.larg
        bx=self.b_pos[0]
        by=self.b_pos[1]
        ypu=pos_paddle1[0]
        ypd=pos_paddle1[1]
        ypu2 = pos_paddle2[0]
        ypd2 = pos_paddle2[1]
        #vertical
        if self.b_pos[1] <= 1:
            self.speed[1]=-self.speed[1]
        elif self.b_pos[1]>= self.window[1]-self.tam_b+5:
            self.speed[1]=-self.speed[1]
        #horizontal + scoring
        elif(bx<=L)and(by+self.tam_b>=ypu and by<ypd):
            self.speed[0]=-self.speed[0]
        elif(bx>=self.window[0]-L)and(by+self.tam_b>=ypu2 and by<ypd2):
             self.speed[0]=-self.speed[0]
        elif self.b_pos[0]<= 1: #sair pela esquerda
            self.reset_ball(1)
            self.out_left+=1
        elif self.b_pos[0]>= (self.window[0]-self.tam_b+5): #sair pela direita
            self.reset_ball(-1)
            self.out_right+=1
    def gerar_2num(self):
        V=[-1,1]
        return V[randint(0,1)]
    def Init_ball(self,event,ball_run=False):
        if ball_run != True: #se apertar "Espaço"  velocidade passa a agir
            if event.type == pg.KEYDOWN:
                if event.key == pg.K_BACKSPACE:
                    self.Inicial_speed=[1,1]
                    ball_run=True
        return ball_run
######################################################################
class Score:
    cor=colors()
    window=Default.window
    scr=0
    font_name="Arial"
    size_f=15

    #f=pygame.font.SysFont('Arial', 30)
    #t=f.render('Your score was: '+str(score), True, (0, 0, 0))
    #screen.blit(t, (10, 270))
#####################################################################
class APP:
    cor=colors()
    window=Default.window
    title="Pong  by @HenrickyL"
    #line
    larg_line=2
    color_line=cor.red
    def __init__(self):
        self.BG=None
        self.run=True
        self.ball_run=False
        self.paddle_L=Paddle()
        #self.paddle_L.x=50
        self.paddle_R=Paddle()
        self.paddle_R.x=self.window[0]-15
        self.ball=Ball()
    #self.S_p1=Score()
        #linhas demilimitadoras
        self.L_up=pg.Surface([self.window[0],self.larg_line])
        self.L_up.fill(self.color_line)#cor
        self.L_dw=pg.Surface([self.window[0],self.larg_line])
        self.L_dw.fill(self.color_line)
        self.L_lf=pg.Surface([self.larg_line,self.window[1]])
        self.L_lf.fill(self.color_line)
        self.L_rg=pg.Surface([self.larg_line,self.window[1]])
        self.L_rg.fill(self.color_line)
    def Off_game(self):
        self.BG.fill(cor.white)
        pg.display.update()
        sleep(0.5)
        print("player 1:%d\n Player 2: %d" %(self.ball.out_left,self.ball.out_right
        #colocar funcionalidade de mostrar score e depois fechar usando sleep, e fe
        pg.quit()
    def draw(self):
        self.BG.fill(cor.black)
        self.BG.blit(self.ball.ball,self.ball.b_pos)
        self.BG.blit(self.paddle_R.p,self.paddle_R.p_pos)
        self.BG.blit(self.paddle_L.p,self.paddle_L.p_pos)
        #score
            #self.S_p1.blit(self.S_p1.txt, (10, self.window[1]*0.2))
        #linhas
        self.BG.blit(self.L_up,[0,0])#self.larg_line+5])
        self.BG.blit(self.L_dw,[0,self.window[1]-self.larg_line])
        self.BG.blit(self.L_lf,[0,self.larg_line])
        self.BG.blit(self.L_rg,[self.window[0]-self.larg_line,0])
        pg.display.update()
    def main(self):
        pg.font.init()# init()
        self.BG=pg.display.set_mode(self.window)
        pg.display.set_caption(self.title)
        self.run=True
        self.ball_run=False
        self.setting()
        self.Off_game()

    def Ball_interaction(self):
        self.ball.mov_ball()#mover a bola
        #variaveis
        pyL=self.paddle_L.p_pos[1]
        pyR=self.paddle_R.p_pos[1]
        L=self.paddle_R.compr
        paddle_left=[pyL,pyL+L]
        paddle_right=[pyR,pyR+L]
        #verificar colisão
        self.ball.Colid_ball(paddle_left,paddle_right)
    def setting(self):
        clock=pg.time.Clock()
        while(self.run):
            clock.tick(70)
            if self.ball_run == True:
                self.Ball_interaction()
            self.draw()
            for event in pg.event.get():
                if event.type == QUIT:
                    self.run=False
                else:
                    if self.ball_run == False:
                        if event.type == KEYDOWN:
                            if event.key == pg.K_SPACE:
                                self.ball_run=True
                    if event.type ==KEYDOWN:
                        if event.key == pg.K_UP:
                            self.paddle_R.mov_up()
                        elif event.key == pg.K_DOWN:
                            self.paddle_R.mov_down()
                        elif event.key == pg.K_w:
                            self.paddle_L.mov_up()
                        elif event.key == pg.K_s:
                            self.paddle_L.mov_down()
                self.draw()
theapp=APP()
theapp.main()
```

```
player 1:2
 Player 2: 4
```

In [6]:

```
#teste de plot_font_display
import pygame.font
pygame.init()
pygame.font.init()
BG=pg.display.set_mode([800,600])
pygame.display.set_caption("teste")
score=5
f=pygame.font.SysFont(None,30)
t=f.render('Your score was: '+str(score), True, (255, 0, 0))
BG.blit(t, (10, 270))
pg.display.update()
sleep(5)
pygame.quit()
```

In [ ]:

In [ ]: