

Baraniecki Karol Byczko Maciej	Prowadzący: Dr inż. Dominik Żelazny	Numer ćwiczenia laboratoria 10
PT 16:30 TP	Temat ćwiczenia: USB, DirectInput	Ocena:
Grupa: D	Data wykonania: 15 października 2021	

## 1 Zadania do opracowania

Teoretyczny opis elementów używanych do bezpośredniej kontroli (Direct Input)

### 1.1 USB w Windows

#### 1.1.1 Warstwa HAL i HEL

HAL - Hardware Abstraction Layer, "nieprzenośna" część jądra NT.

HEL - Hardware Emulation Library, jak sama nazwa mówi, służy do emulacji urządzeń, problem ze znalezieniem dokumentacji, raczej już nie używana.

#### 1.1.2 Standard USB oraz USB 2.0

USB - Universal Serial Bus, po polsku Uniwersalna Magistrala Szeregowa, uniwersalny interfejs komunikacyjny typu plug and play. Obecnie najnowszą wersją jest USB 3.2 wyprodukowany w 2017 roku.

## 2 Zadania do wykonania

1. Napisać program, odczytujący nazwę zainstalowanego joysticka
2. Napisać program ilustrujący działanie joysticka: stwierdzający naciśnięcie myszy, zmianę położenia drążka (w przestrzeni 2D) oraz suwaka.
3. Napisać program zastępujący działanie myszy. Program ma umożliwiać sterowanie kursorem za pomocą joysticka oraz obsługę przycisków fire jako kliknięć myszy.
4. Napisać program realizujący prosty edytor graficzny - rysowanie przy pomocy Joysticka

### 2.1 Odczytywanie nazwy joysticka

```
1 pygame.joystick.init()
2 pad = pygame.joystick.Joystick(0)
3 pad_name = pad.get_name()
```

### 2.2 Kontrola myszki za pomocą joysticka

Do wykonania tego zadania wykorzystaliśmy biblioteki *Mouse* oraz *PyGame*

```
1 import time
2 import mouse
3 import pygame
4 import sys
5 pygame.init()
6
7 joystick_count = pygame.joystick.get_count()
8 print("Number of joysticks: {}".format(joystick_count) )
9
10 if joystick_count == 0:
11     print("No joysticks found")
12     sys.exit(0)
13
14 joystick = pygame.joystick.Joystick(0)
15
16 is_pressed = mouse.is_pressed()
17 is_pressed_right = mouse.is_pressed(button='right')
18
19 while True:
20     pygame.event.pump()
21     time.sleep(1.0 / 30.0)
22     mult = 3
23     mouse.move((mult * joystick.get_axis(0)) ** 3, (mult * joystick.
24               get_axis(1)) ** 3, absolute=False)
25
26     if joystick.get_button(0) and not is_pressed:
27         is_pressed = True
28         mouse.press()
29
30     if not joystick.get_button(0) and is_pressed:
31         is_pressed = False
32         mouse.release()
33
34     if joystick.get_button(2) and not is_pressed_right:
35         is_pressed_right = True
36         mouse.press(button='right')
37
38     if not joystick.get_button(0) and is_pressed_right:
39         is_pressed_right = False
40         mouse.release(button='right')
```

## 2.3 Paint

## 2.4 odczytanie nazwy zainstalowanego joysticka

```
1 import pydirectinput
2 import pygame
3 import random
4
5 # class for drawing circles
6 class Circle:
7     def __init__(self, x, y, radius, color):
8         self.x = x
9         self.y = y
10        self.radius = radius
11        self.color = color
12
13    def draw(self, surface):
14        pygame.draw.circle(surface, self.color, (self.x, self.y),
15                           self.radius)
16
17
18 pygame.init() # initialize
19 pad = pygame.joystick.Joystick(0) # get joystick
20 width, height = 1980, 900 # set window size
21 screen = pygame.display # display screen
22 surface = screen.set_mode((width, height)) # set surface
23
24 pad_name = pad.get_name() # get pad name
25 pygame.display.set_caption(f"{pad_name} - drawing") # display pad
    name
26
27 # cursor position, default center
28 position_x = width / 2
29 position_y = height / 2
30
31 speed = 0.5 # cursor speed
32 running = True # while false, run program
33 drawing = False # if true, draw
34
35 color = (0, 0, 0) # default color
36
37 radius = 25 # cursor radius
38
39 circles = [] # drawn circles
40
41 #main loop
42 while running:
43     for event in pygame.event.get():
44         if event.type == pygame.QUIT:
45             running = False
46         if event.type == pygame.JOYBUTTONDOWN:
```

```
47         if pad.get_button(0):
48             drawing = True
49         if pad.get_button(1):
50             speed *= 0.5 # set speed to slower
51         if pad.get_button(2):
52             speed *= 2 # set speed to higher
53         if pad.get_button(3): # set random color
54             color = (random.randrange(0, 255),
55                     random.randrange(0, 255),
56                     random.randrange(0, 255))
57         if pad.get_button(5): # change cursor size
58             if radius != 1:
59                 radius -= 1
60         if pad.get_button(6):
61             radius += 1
62     if event.type == pygame.JOYBUTTONDOWN: # if fire clicked then
        draw
63         if not pad.get_button(0):
64             drawing = False
65
66     surface.fill((255, 255, 255)) # fill cursor
67
68     # draw all circles
69     for circle in circles:
70         circle.draw(surface)
71
72     # change cursor position
73     position_x += pad.get_axis(0) * speed
74     position_y += pad.get_axis(1) * speed
75
76     # display cursor
77     pygame.draw.circle(surface, (0, 0, 0), (position_x, position_y),
78                        radius)
79
80     # add new circles
81     if drawing:
82         circles.append(Circle(position_x, position_y, radius, color))
83
84     screen.update()
```

### 2.4.1 Dowód działania programu graficznego

