DOKUMENTASI

MATA KULIAH PEMROGRAMAN BERBASIS OBJEK

PROGRAM PYGAME DAN KIVY

Dosen Pengampu :

Arik Kurniawati, S.Kom., M.T.

Disusun oleh :

Rini Azlinda (170411100031)

Marwa Majidah (170411100067)

JURUSAN TEKNIK INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS TRUNOJOYO MADURA

2018/2019

1. Kivy

Kivy ini sendiri merupakan framework yang dibangun menggunakan library dari bahasa pemrograman Python yang bersifat Open Source.

Cara Instal kivy

1. Masuk ke CMD (*Command Prompt*) dengan cara menekan tombol Windows lalu ketik CMD atau dengan cara klik start, assesoris, pilih commad promt. Pastikan seri python yang akan diinstall dengan kivy benar.
2. Sambungkan koneksi internet.
3. Ketik perintah Python –m pip install kivy.

2. PyGame

PyGame adalah modul Python yang berisi fungsi dan class yang kita butuhkan untuk membuat game.

Pygame merupakan seperangkat modul Python yang dirancang untuk membuat permainan. Pygame menambahkan fungsi di atas dengan sangat baik di SDL perpustakaan. Hal ini memungkinkan Anda untuk membuat sebuah game dengan fitur yang lengkap dan sebuah program multimedia dalam bahasa python. Pygame sangat portabel dan dapat berjalan pada hampir semua platform dan sistem operasi. Pygame sendiri telah didownload jutaan kali, dan telah memiliki jutaan kunjungan ke situsnya.

Install pygame

1. Buka cmd pastikan seri python yang ingin di install pygame benar.
2. Sambungkan koneksi internet.
3. Python –m pip install pygame.

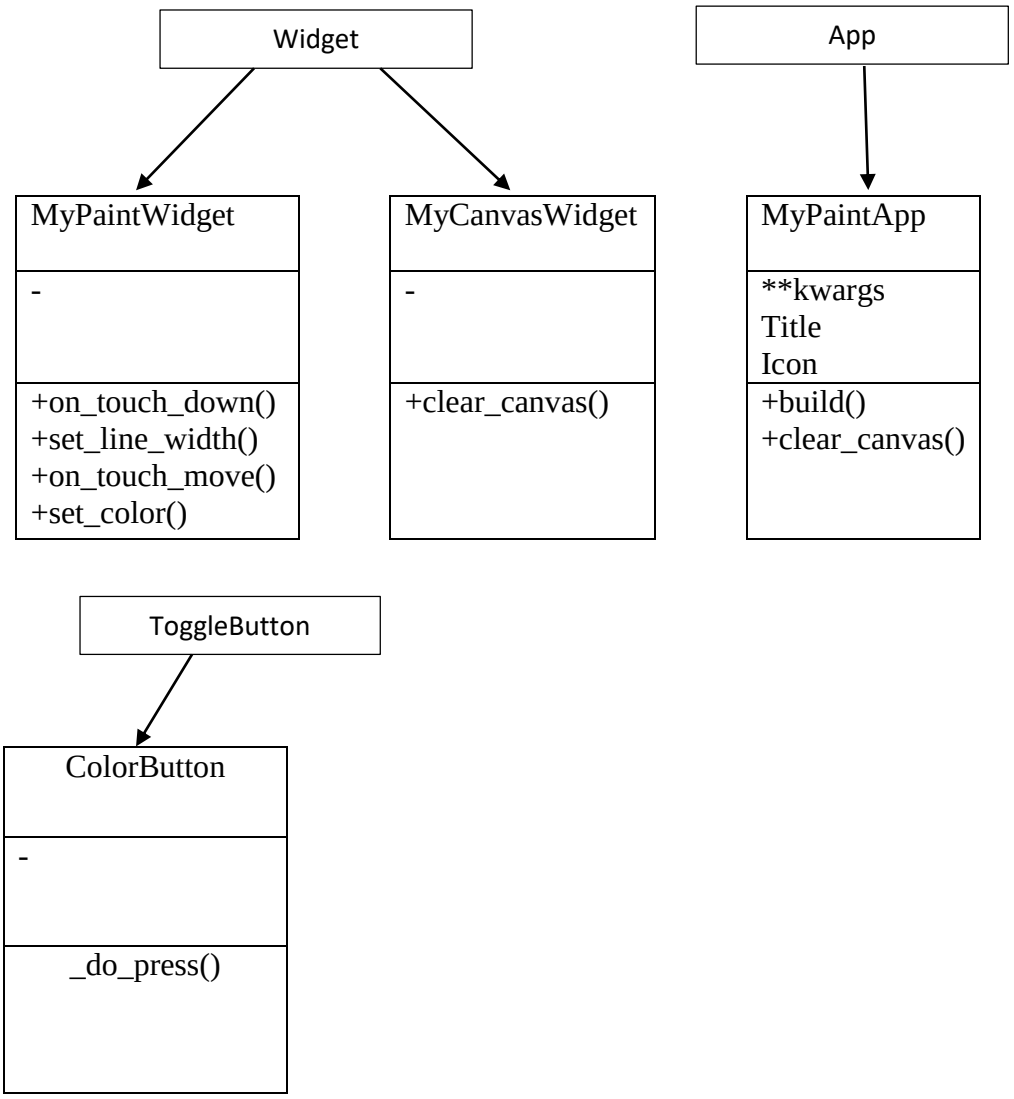
Install pygame dengan cara mendownload file versi wheel

1. Siapkan file pygame.whl
   Download library pygame di https://www.lfd.uci.edu/~gohlke/pythonlibs/ .Sesuaikan versi python yang terinstall dengan versi library pygame yang akan di download.
   Contoh python v3.5 telah terinstall, maka download library pygame dengan cp35m-win32.
2. Rubah file pygame.whl menjadi .zip
3. Rubah extension file library pygame lalu extract isi library tersebut.
4. Copy file pygame yang dibutuhkan
5. masuk ke dalam directory python
   (C:\Users\#username\AppData\Local\Programs\Python\Python35-32)

6. masuk kedalam folder “include” dan buat folder baru bernama “pygame”.
7. Di dalam folder hasil extract file library pygame yang sudah didownload, masuk ke “pygame1.9.4.data\header”,
8. copy semua file di dalam folder tersebut dan masukkan ke dalam folder: C:\Users\#username\AppData\Local\Programs\Python\Python35-32\include\pygame
9. kembali ke folder hasil extract file library pygame tadi, copy folder “pygame” dan “pygame1.9.4.dist-info” kedalam: C:\Users\#username\AppData\Local\Programs\Python\Python35-32\Lib\site-packages
10. Cek hasil installasi
11. Buka IDLE Python lalu lakukan perintah “import pygame”, jika tidak ada tulisan error maka pygame sudah berhasil terinstall.

DOKUMENTASI PROGRAM KIVY

“Paint Editor”



Yang digunakan dari kivy

- Widget
- Button
- Color
- Line
- ToggleButton
- ToggleButtonBehavior
- Config

Class dan Method untuk Aplikasi Paint

MyPaintWidget
-
+on_touch_down() +set_line_width() +on_touch_move() +set_color()

Class MyPaintWidget ➔Inheritance dari Widget

Method :

+ On_touch _down untuk memanggil event saat event diinisiasi

```
def on_touch_down(self, touch):
    if Widget.on_touch_down(self, touch):
        return
    with self.canvas:
        touch.ud['line'] = Line(points=(touch.x, touch.y), width=self.line_width)
```

+ set_line_width untuk mengatur tebal garis saat program awal dijalankan.

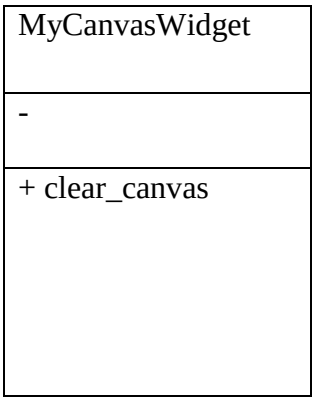
```
def set_line_width(self, line_width=3):
    self.line_width = line_width
```

+ on_touch_move untuk memanggil saat event touch berpindah lokasi

```
def on_touch_move(self, touch):
    if touch.ud:    #supaya slider tidak error
        touch.ud['line'].points += [touch.x, touch.y]
```

+ set_color untuk merubah warna yang akan digunakan

```
def set_color(self, new_color):
    #change paint color
    self.last_color = new_color
    self.canvas.add(Color(*new_color))
```



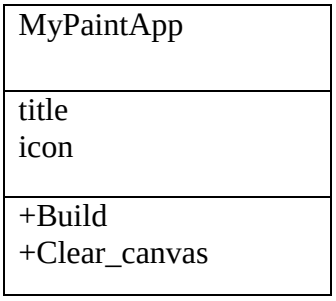
Class MyCanvasWidget ➔ Inheritance dari Widget

Method:

clear_canvas method untuk membersihkan canvas

```
class MyCanvasWidget(Widget):

    def clear_canvas(self):
        MyPaintWidget.clear_canvas(self)
```



Class MyPaintApp ➔ Inheritance dari class App

Property:

- **kwargs untuk mengambil semua keyword arguments yang diberikan sebagai parameter
- Title untuk memberi judul window pada plikasi
- Icon untuk memberi icon pada window aplikasi

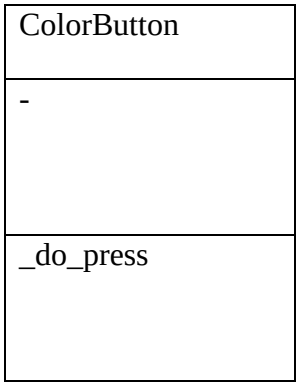
Method

- Build method untuk mengatur widget paint area serta warna pertama yang dapat digunakan saat program awal dijalankan

```
def build(self):
    parent = Widget()
    self.painter = MyCanvasWidget()
    #color settings at the time of start up
    self.painter.ids['paint_area'].set_color(get_color_from_hex('#000000'))
#set to black
    return self.painter
```

- clear_canvas untuk mengatur saat canvas dihapus

```
def clear_canvas(self):
    self.painter.ids['paint_area'].canvas.clear()
    self.painter.ids['paint_area'].set_color(self.painter.ids['paint_area'].
```



Class ColorButton ➔ Inheritance dari ToggleButton

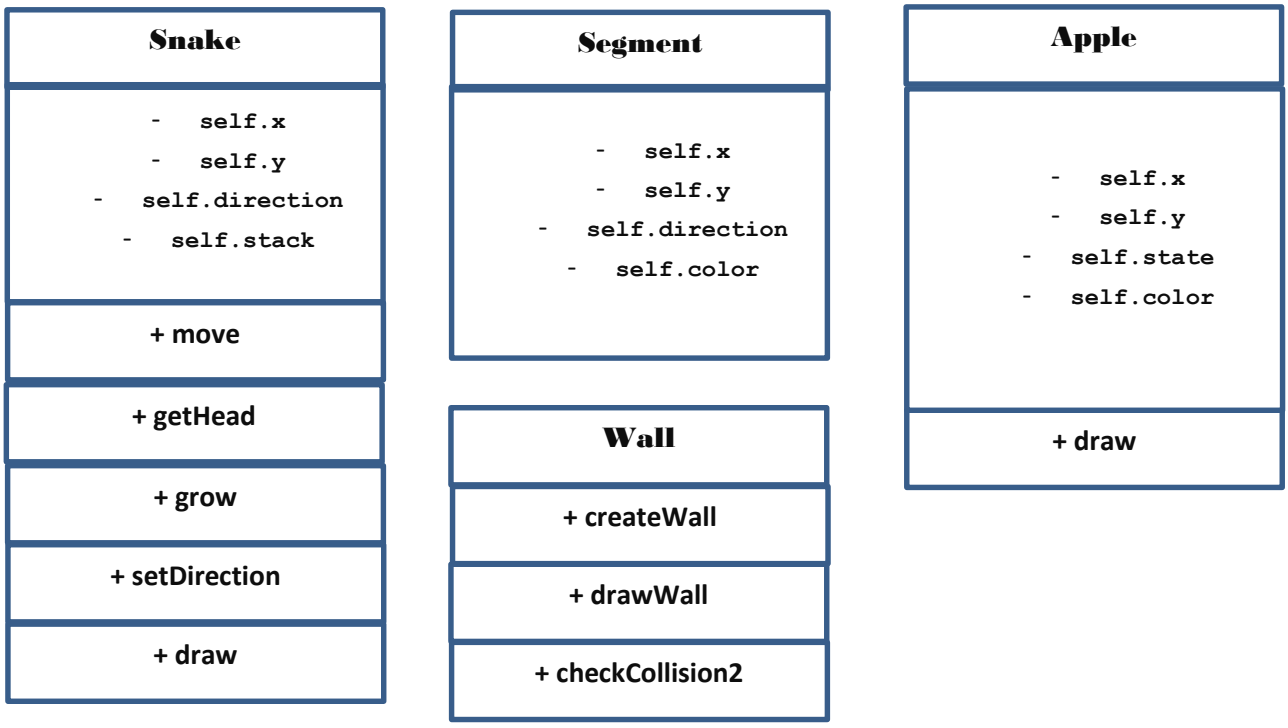
Memiliki Method

- _do_press untuk mengatur toggle behaviour saat button diklik

```
class ColorButton(ToggleButton):
    def _do_press(self):
        if self.state == 'normal':
            ToggleButtonBehavior._do_press(self)
```

DOKUMENTASI PROGRAM PYGAME

“Game Snacke”



`class Snake:` sebagai kelas untuk membuat Snake(ular)

`def __init__(self,x,y):` methode constructor untuk kelas Snake

`self.x = x` atribut untuk menginisialisasi bentuk snake pada sumbu x

`self.y = y` atribut untuk menginisialisasi bentuk snake pada sumbu y

`self.direction = KEY["UP"]` atribut untuk menampung key up

`self.stack = []` atribut sebagai variabel untuk membuat list

`self.stack.append(self)` untuk menambah ke stack

`def move(self):` methode move untuk mengatur perpindahan

`def getHead(self):` methode untuk bentuk kepalanya

`def grow(self):` methode grow dari kelas Snake yang berarti mengatur pertumbuhan atau perkembangan dari snake tersebut. Ia tumbuh dengan cara jika ia ke arah atas dan kiri tubuhnya akan berkurang , ia akan tumbuh ketika ia menuju arah kanan dan bawah.

`def setDirection(self,direction):` methode dari class Snake ini mengatur arah dengan setDirection dan aturannya juga mengikuti program sebelumnya.

`def draw(self,screen):` methode dari class Snake ini mengatur bagaiman cara menggambar atau membuat tubuh snake yang dimulai dari indek ke 0 atau kepalanya berwarna hijau sesuai dengan ukuran snake dan dilanjutkan membuat bentuk tubuh yang tanpa warna(jarak) kemudian membentuk tubuhnyan dengan warna putih.

`class Segment:` sebagai class untuk membuat segment

`def __init__(self,x,y):` methode constructor untuk kelas segment

`self.x = x` atribut untuk menginisialisasi pembentukan segment pada sumbu x

`self.y = y` atribut untuk menginisialisasi pembentukan segment pada sumbu y

`self.direction = KEY["UP"]` atribut untuk menampung key up

`self.color = "green"` atribut untuk menampung string green

`class Apple:` sebagai kelas untuk membuat apple

`def __init__(self,x,y,state):` methode constructor untuk kelas apple

`self.x = x` atribut untuk menginisialisasi bentuk apel pada sumbu x

`self.y = y` atribut untuk menginisialisasi bentuk apel pada sumbu y

`self.state = state` atribut untuk menampung state

`self.color = pygame.color.Color("blue")` atribut untuk menampung inisialisasi yang mengatur warna biru

`def draw(self,screen):` methode draw untuk menggambar dengan inheritance dari screen

```
pygame.draw.rect(screen,self.color,(self.x,self.y,APPLE_SIZE,APPLE_SIZE),0)
```

`class Wall:` sebagai kelas untuk membuat dinding

`def createWall(self):` methode untuk membuat gambar dinding

```
global boundry
self.boundry=[]
for i in range(0,SCREEN_WIDTH):
    self.boundry.append((i,0))
    self.boundry.append((i,SCREEN_HEIGHT-7))
for i in range(0,SCREEN_HEIGHT):
```

```
            self.boundry.append((0,i))
            self.boundry.append((SCREEN_WIDTH-7,i))
    def drawWall(self):
        for each in self.boundry:
            self.wallRect = pygame.Rect(each[0],each[1], WALL_SIZE-2, WALL_SIZE-2)
            pygame.draw.rect(screen, yellow, self.wallRect)
```

methode untuk menggambar dinding

```
    def checkCollision2(self,snake):
```

methode untuk mengecek tabrakan antara ular dan dindingnya

```
        self.bumper_x = SCREEN_WIDTH - WALL_SIZE
        self.bumper_y = SCREEN_HEIGHT - WALL_SIZE
        if(
            snake.x < WALL_SIZE or
            snake.y < WALL_SIZE or
            snake.x + SNAKE_SIZE > self.bumper_x or
            snake.y + SNAKE_SIZE > self.bumper_y):
            endGame()
            return True
        else:
            return False
```

```
#collisiontoapple
def checkCollision(posA,As,posB,Bs):
    #As size of a | Bs size of b
    if(posA.x   < posB.x+Bs and posA.x+As > posB.x and posA.y < posB.y + Bs and posA.y+As > posB.y):
        return True
    return False
```

Program ini untuk mengecek atau mengatur tabrakan yang akan dilakukan snake.

```
def getKey():
    for event in pygame.event.get():
        if event.type == pygame.KEYDOWN:
            if event.key == pygame.K_UP:
                return KEY["UP"]
            elif event.key == pygame.K_DOWN:
                return KEY["DOWN"]
            elif event.key == pygame.K_LEFT:
                return KEY["LEFT"]
            elif event.key == pygame.K_RIGHT:
                return KEY["RIGHT"]
            elif event.key == pygame.K_ESCAPE:
                return "exit"
            elif event.key == pygame.K_y:
                return "yes"
            elif event.key == pygame.K_n:
                return "no"
        if event.type == pygame.QUIT:
            done = True
            pygame.quit()
            sys.exit()
```

Program ini dengan methode getKey untuk menentukan key-key yang akan digunakan pada saat game snake. Jika ingin ke arah atas bisa menggunakan K_UP, jika ingin ke kiri bisa menggunakan K_LEFT, jika ingin ke kanan bisa menggunakan K_RIGHT, jika ingin exit bisa menggunakan K_ESCAPE, jika ingin melanjutkan permainan bisa menggunakan K_Y, jika ingin menghentikan permainan bisa menggunakan K_N, jika ingin kembali bisa menggunakan QUIT.

```
def respawnApple(apples,index,sx,sy):
    radius = math.sqrt((SCREEN_WIDTH/2*SCREEN_WIDTH/2  + SCREEN_HEIGHT/2*SCREEN_HEIGHT/2))/2
    angle = 999
    while(angle > radius):
        angle = random.uniform(0,800)*math.pi*2
        x = SCREEN_WIDTH/2 + radius * math.cos(angle)
        y = SCREEN_HEIGHT/2 + radius * math.sin(angle)
        if(x == sx and y == sy):
            continue
    newApple = Apple(x,y,1)
    apples[index] = newApple

def respawnApples(apples,quantity,sx,sy):
    counter = 0
    del apples[:]
    radius = math.sqrt((SCREEN_WIDTH/2*SCREEN_WIDTH/2  + SCREEN_HEIGHT/2*SCREEN_HEIGHT/2))/2
    angle = 999
    while(counter < quantity):
        while(angle > radius):
            angle = random.uniform(0,800)*math.pi*2
            x = SCREEN_WIDTH/2 + radius * math.cos(angle)
            y = SCREEN_HEIGHT/2 + radius * math.sin(angle)
            if( (x-APPLE_SIZE == sx or x+APPLE_SIZE == sx)
                and (y-APPLE_SIZE == sy or y+APPLE_SIZE == sy) or radius - angle <= 10):
                continue
        apples.append(Apple(x,y,1))
        angle = 999
        counter+=1
```

Program ini yang mengatur bagaimana apple itu akan muncul.

```
def endGame():
    message = game_over_font.render("Game Over",1,pygame.Color("white"))
    message_play_again = play_again_font.render("Play Again? Y/N",1,pygame.Color("green"))
    screen.blit(message,(320,240))
    screen.blit(message_play_again,(320+12,240+40))

    pygame.display.flip()
    pygame.display.update()

    myKey = getKey()
    while(myKey != "exit"):
        if(myKey == "yes"):
            main()
        elif(myKey == "no"):
            break
        myKey = getKey()
    pygame.quit()
    sys.exit()
```

Program ini akan mengatur jika permainan akan berakhir. Jika permainan berakhir akan muncul pesan Play Again?Y/N berawarna hijau, jika anda menekan Y maka permainan kn lanjut namun jika anda menekan huruf N makan permainan akan diakhiri.

```
def createWall():
    global boundry
    boundry=[]
    for i in range(0,SCREEN_WIDTH):
        boundry.append((i,0))
        boundry.append((i,SCREEN_HEIGHT-7))
    for i in range(0,SCREEN_HEIGHT):
        boundry.append((0,i))
        boundry.append((SCREEN_WIDTH-7,i))

def drawWall():
    for each in boundry:
        wallRect = pygame.Rect(each[0],each[1], WALL_SIZE-2, WALL_SIZE-2)
        pygame.draw.rect(screen, yellow, wallRect)
```

Program ini akan mengatur bagaiman cara membuat dinding hingga membuat dindingnya sendiri. Dinding itu dibentuk mengikuti lebar dan tinggi screen sebagai pembatas dengan lebar dan tinggi seukuran dengan ukuran snake-2 yang berwarna kuning.

```
def checkCollision2(snake):
    bumper_x = SCREEN_WIDTH - WALL_SIZE
    bumper_y = SCREEN_HEIGHT - WALL_SIZE
    if(
        snake.x < WALL_SIZE or
        snake.y < WALL_SIZE or
        snake.x + SNAKE_SIZE > bumper_x or
        snake.y + SNAKE_SIZE > bumper_y):
        endGame()
        return True
    else:
        return False
```

Program ini untuk mengecek atau mengatur tabrakan snake dengan dinding dengan cara jika snake tidak melebihi bumper di sumbu x dan y maka snake true dan jika sebaliknya jika snake menyentuh bumper pada sumbu x dan y makan snake akan false(game over).

```
def drawGameTime(gameTime):
    game_time = score_font.render("Time:",1,pygame.Color("green"))
    game_time_numb = score_numb_font.render(str(gameTime/1000),1,pygame.Color("red"))
    screen.blit(game_time,(30,10))
    screen.blit(game_time_numb,(105,14))
```

Program dengan methode drawGameTime ini berfungsi untuk mengatur waktu pada permainan.

```
def main():
    score = 0

    #Snake initialization
    mySnake = Snake(SCREEN_WIDTH/2,SCREEN_HEIGHT/2)
    mySnake.setDirection(KEY["UP"])
    mySnake.move()
    start_segments=3
    while(start_segments>0):
        mySnake.grow()
        mySnake.move()
        start_segments-=1

    #Apples
    max_apples = 1
    eaten_apple = False
    apples = [Apple(random.randint(60,SCREEN_WIDTH),random.randint(60,SCREEN_HEIGHT),1)]
    respawnApples(apples,max_apples,mySnake.x,mySnake.y)
    endgame = 0

    while(endgame!=1):
        gameClock.tick(FPS)

        #Input
        keyPress = getKey()
        if keyPress == "exit":
            endgame = 1

        #Collision to wall
        checkCollision2(mySnake)

        for myApple in apples:
            if(myApple.state == 1):
                if(checkCollision(mySnake.getHead(),SNAKE_SIZE,myApple,APPLE_SIZE)==True):
                    mySnake.grow()
                    myApple.state = 0
                    eaten_apple=True
```

```
          #Position Update
          if(keyPress):
              mySnake.setDirection(keyPress)
          mySnake.move()

          #Respawning apples
          if(eaten_apple == True):
              eaten_apple = False
              respawnApple(apples,0,mySnake.getHead().x,mySnake.getHead().y)

          #Drawing
          screen.fill(background_color)
          for myApple in apples:
              if(myApple.state == 1):
                  myApple.draw(screen)

          createWall()
          drawWall()
          mySnake.draw(screen)

          #Flip screen
          pygame.display.flip()
          pygame.display.update()

main()
```

Pada program ini dengan methode main yang berfungsi untuk mengekskusi semua program yang pertama menginisialisai snake yang akan berjalan dengan posisi ditengah ke arah atas kemudian membentuk apple, kemudian mengatur dinding dan tampilan screen.