**DOKUMENTASI**

**”PROJECT PYGAME DAN KIVY”**

**PEMROGRAMAN BERORIENTASI OBJEK**



**Disusun Oleh :**

|  |
| --- |
| **NAMA : 1. Nur Aeni (170411100001)**  **2. Ashof Barkhia R. (170411100064)**  **MATA KULIAH : PBO 3A** |

**Teknik Informatika**

**Fakultas Teknik**

**Universitas Trunojoyo Madura**

**PYGAME**

1. **Pengertian Pygame**

Pygame adalah salah satu modul Python seperti tinker, yang dirancang untuk membuat permainan. Pygame menambahkan fungsi sangat baik di SDL perpustakaan. Hal ini memungkinkan 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.

Pygame ini gratis. Dan dirilis di bawah Lisensi GPL , Anda dapat membuat sebuah aplikasi yang open source, gratis, freeware, shareware, dan game komersial dengan pygame ini*.*

1. **Instalasi Pygame**

Sebelum menggunakan modul pygame terlebih dulu kita harus melakukan instalasi pygame, dengan mengikuti cara-cara berikut ini :

* Siapkan file pygame

Download library pygame di https://www.Ifd.uci.edu/~gohlke/pythonlibs/ .

Sesuaikan dengan versi python awal yang terinstall.

* Ganti file pygame.whl menjadi zip

Ganti extension file library pygame lalu extract isi library.

* Copy file pygame yang dibutuhkan
* Masuk ke direktori python

**(C:\User\#username\AppData\local\Programs\Python\Python35-32)**

* Masuk ke dalam folder include dan buat folder bernama **’’pygame’’**

Dalam folder hasil extract file library pygame yang sudah di download, masuk ke **’’pygame-1.9.4data\header’’**, copy semua file dalam folder tersebut dan masukkan pada folder **C:\User\#username\AppData\local\Programs\Python\Python35-32\**

**include\pygame.**

* Kembali ke folder hasil extract file liberary pygame, copy folder **’’pygame’’** dan ’’pygame-1.9.4.dist-info’’ ke dalam :
* **C:\User\#username\AppData\local\Programs\Python\Python35-32\Lib\ site-packages**
* Cek hasil Instalasi

Buka IDLE python, lakukan perintah ’’import pygame’’ jika sudah tidak ada tulisan eror maka pygame berhasil terinstall

1. **Penjelasan dari OOP Pygame ”Galaxy Wars”**

|  |
| --- |
| boss |
| * rect * rect.top * rect.left * cepat * tembak * gerak * pelatuk * nyawa * bulletinformation * bulletspeed * spreecount * spree * isautopilot * shot * reloadtime |
| + checkbounds()  + update()  + drawplayer()  + bedil() |

|  |
| --- |
| Player |
| * rect.top * rect.left * cepat * tembak * gerak * pelatuk * nyawa * kills * skore * tundatembak * isautopilot * shot * won |
| + checkbounds()  + update()  + drawplayer()  + bedil()  +autopilot() |

|  |
| --- |
| bintang |
| * radius * warna * cepat * nofstar * posbintang |
| + gambarbintang()  + pindahbintang() |

|  |
| --- |
| Musuh |
| * rect * image * cepat * tembak * gerak * pelatuk * nyawa * isautopilot * ledakan\_sound * shot |
| + checkbounds()  + update()  + drawplayer()  + bedil()  +autopilot() |

|  |
| --- |
| Enemydrone |
| * rect.left * rect.top * image * cepat * tembak * gerak * nyawa * waittime * ledakan\_sound * shot |
| + checkbounds()  + update()  + drawplayer()  + bedil()  +autopilot() |

|  |
| --- |
| Enemysaucer |
| * rect * rect.center * image * index * tembak * gerak * nyawa * waittime * ledakan\_sound * shot * haltpos |
| + checkbounds()  + update()  + drawplayer()  + bedil()  +autopilot() |

|  |
| --- |
| Kotaksehat |
| * rect.left * rect.top * image * gerak * maxleft * maxright |
| + checkbounds()  + update()  + drawplayer()  + bedil()  +autopilot() |

|  |
| --- |
| Peluru |
| * rect.center * direction * image |
| + update() |

|  |
| --- |
| Pelurumusuh |
| * rect.center * rect * direction * image * cepat |
| + update() |

|  |
| --- |
| Ledakan |
| * rect * rect.center * index * image |
| + update() |

Penjelasan Class

Semua Class di program game Galaxy Wars berada di Public tidak ada yang private

Class Bintang termasuk class : Class biasa tidak terikat inheritance dengan class manapun

Class Player termasuk class : inheritance overriding

Class Boss termasuk class : inheritance overriding

Class Musuh termasuk class : inheritance overriding

Class Enemydrone termasuk class : inheritance overriding

Class Kotaksehat termasuk class : inheritance overriding

Class Enemysaucer termasuk class : inheritance overriding

Class Peluru termasuk class : inheritance overriding

Class Pelurumusuh termasuk class : inheritance overriding

Class Ledakan termasuk class : inheritance overriding

Global Variable :

+ black, white, sky, red, yellow, green

+ size = (width, height) = (1024, 600) untuk menentukan ukuran permainan pada desktop

+ clock = pygame.time.Clock() variabel untuk pemanggilan FPS

+ FPS = 21 frame per second sebuah tangkapan frame yang didapat setiap detiknya(Perdetiknya)

+ maxspeed = 15 untuk menentukan kecepatan pada permainan

+ screen = pygame.display.set\_mode(size) variabel pemanggilan untuk size permainan

Method and Class from pygame :

+ pygame.init()

+ time.Clock()

+ display.set\_mode()

+ display.set\_caption()

+ event.get()

+ event.type()

+ mixer.Sound()

+ image.load()

+ display.update()

+ Clock.tick(FPS)

+ quit()

Class dan Method Galaxy Wars

Class bintang:

|  |
| --- |
| Bintang |
| * Radius * warna * cepat * nofstar * posbintang |
| + gambarbintang()  + pindahbintang() |

Memiliki property :

* radius untuk menentukan letak bintang
* warna untuk menentukan warna bintang
* cepat untuk menentukan kecepatan bintang
* nofstars untuk bintang yang lain
* posbintang untuk menentukan posisi bintang

Method :

+ gambarbintang() untuk menggambar bintang dari class bintang

+ pindahbintang() untuk memindahkan posisi bintang

Class player :

|  |
| --- |
| Player |
| * rect.top * rect.left * cepat * tembak * gerak * pelatuk * nyawa * kills * skore * tundatembak * isautopilot * shot * won |
| + checkbounds()  + update()  + drawplayer()  + bedil()  +autopilot() |

Memiliki property :

* rect.top untuk menentukan posisi pesawat di atas
* rect.left untuk menentukan posisi pesawat di bawah
* cepat untuk menentukan kecepatan
* tembak untuk menentukan kecepatan peluru
* gerak untuk menentukan gerakan pesawat
* pelatuk untuk menentukan jumlah peluru pesawat
* nyawa untuk menenukan nyawa player
* kills untuk menghitung jumlah yang telah dibunuh
* skore untuk menghitung skore
* tundatembak untuk menentukan jeda dari tembakan peluru
* isautopilot untuk menentukan gerak pesawat otomatis
* shot untuk menentukan tembakan
* won untuk menentukan kemenangan

Method :

+ checkbounds() untuk memeriksa batas player

+ update() untuk memperbarui player

+ drawplayer() untuk pemanggilan load gambar pesawat pada player

+ bedil() untuk menentukan tembakan dan jumlah peluru

+ autopilot() untuk menentukan gerak otomatis

Class boss:

|  |
| --- |
| Boss |
| * rect * rect.top * rect.left * cepat * tembak * gerak * pelatuk * nyawa * bulletinformation * bulletspeed * spreecount * spree * isautopilot * shot * reloadtime |
| + checkbounds()  + update()  + drawplayer()  + bedil() |

Memiliki property :

* rect.top untuk menentukan posisi boss di atas
* rect.left untuk menentukan posisi boss di bawah
* cepat untuk menentukan kecepatan
* tembak untuk menentukan kecepatan peluru
* gerak untuk menentukan gerakan boss
* pelatuk untuk menentukan jumlah peluru boss
* nyawa untuk menenukan nyawa boss
* isautopilot untuk menentukan gerak boss otomatis
* bulletformation adalah informasi peluru
* bulletspeed untuk menentukan kecepatan peluru
* spreecount untuk menghitung kesenangan (peluru)
* spree untuk kesenangan (peluru)

Method :

+ checkbounds() untuk memeriksa batas boss

+ update() untuk memperbarui boss

+ drawplayer() untuk pemanggilan load gambar pesawat pada boss

+ bedil() untuk menentukan tembakan dan jumlah peluru boss

Class musuh :

|  |
| --- |
| Musuh |
| * rect * image * cepat * tembak * gerak * pelatuk * nyawa * isautopilot * ledakan\_sound * shot |
| + checkbounds()  + update()  + drawplayer()  + bedil()  +autopilot() |

Memiliki property :

* rect untuk menentukan posisi musuh
* image untuk mengambil gambar sekaligus rect pada musuh
* image.blit untuk menentukan koordinat musuh
* ledakan\_sound untuk mengambil file suara dalam folder

Method :

+ checkbounds() untuk memeriksa batas musuh

+ update() untuk memperbarui musuh

+ drawplayer() untuk pemanggilan load gambar pesawat pada musuh

+ bedil() untuk menentukan tembakan dan jumlah peluru musuh

+ autopilot() untuk menentukan gerak otomatis

|  |
| --- |
| Enemydrone |
| * rect.left * rect.top * image * cepat * tembak * gerak * nyawa * waittime * ledakan\_sound * shot |
| + checkbounds()  + update()  + drawplayer()  + bedil()  +autopilot() |

Class enemydrone:

Memiliki property :

* rect untuk menentukan posisi saucer
* rect.center untuk menentukan posisi saucer di tengah
* cepat untuk menentukan kecepatan
* tembak untuk menentukan kecepatan peluru
* gerak untuk menentukan gerakan saucer
* waitTime untuk menentukan waktu tunggu
* nyawa untuk menenukan nyawa saucer
* ledakan\_sound untuk mengambil file suara dalam folder

Method :

+ checkbounds() untuk memeriksa batas saucer

+ update() untuk memperbarui saucer

+ drawplayer() untuk pemanggilan load gambar pesawat pada saucer

+ bedil() untuk menentukan tembakan dan jumlah peluru saucer

+ autopilot() untuk menentukan gerak otomatis pada saucer

Class enemysaucer:

|  |
| --- |
| Enemysaucer |
| * rect * rect.center * image * index * tembak * gerak * nyawa * waittime * ledakan\_sound * shot * haltpos |
| + checkbounds()  + update()  + drawplayer()  + bedil()  +autopilot() |

Memiliki property :

* rect untuk menentukan posisi saucer
* rect.left untuk menentukan posisi saucer di kiri
* index untuk menentukan index
* image untuk menentukan gambar
* tembak untuk menentukan kecepatan peluru
* gerak untuk menentukan gerakan saucer
* waitTime untuk menentukan waktu tunggu
* nyawa untuk menenukan nyawa saucer
* ledakan\_sound untuk mengambil file suara dalam folder
* haltpos untuk menentukan posisi
* shot untuk menentukan tembakan

Method :

+ checkbounds() untuk memeriksa batas saucer

+ update() untuk memperbarui saucer

+ drawplayer() untuk pemanggilan load gambar pesawat pada saucer

+ bedil() untuk menentukan tembakan dan jumlah peluru saucer

+ autopilot() untuk menentukan gerak otomatis pada saucer

Class kotaksehat:

|  |
| --- |
| Kotaksehat |
| * rect.left * rect.top * image * gerak * maxleft * maxright |
| + checkbounds()  + update()  + drawplayer()  + bedil()  +autopilot() |

Memiliki property :

* self.rect.left untuk menentukan posisi kotak sehat di kiri
* self.rect.right untuk menentukan posisi kotak sehat di kanan
* self.gerak untuk menentukan posisi kotak sehat
* self.maxleft untuk menentukan maksimal kiri
* self.maxright untuk menentukan maksimal kanan
* image untuk menentukan gambar

Method :

+ checkbounds() untuk memeriksa batas kotak sehat

+ update() untuk memperbarui kotak sehat

+ drawplayer() untuk pemanggilan load gambar kotak sehat

+ autopilot() untuk gerak otomatis kotak sehat

Class peluru :

|  |
| --- |
| Peluru |
| * rect.center * direction * image |
| + update() |

Memiliki property :

* rect.center untuk menentukan posisi peluru di tengah
* direction untuk direksi
* image untuk menentukan gambar

Memiliki method :

+ update() untuk memperbarui peluru

Class pelurumusuh :

|  |
| --- |
| Pelurumusuh |
| * rect.center * rect * direction * image * cepat |
| + update() |

Memiliki Property :

* rect untuk menentukan posisi peluru musuh
* rect.center untuk menentukan posisi tengah peluru musuh
* direction untuk direksi
* self.cepat untuk menentukan kecepatan peluru musuh
* image untuk menentukan gambar

Memiliki Method :

+ update() untuk memperbarui peluru musuh

|  |
| --- |
| Ledakan |
| * rect * rect.center * index * image |
| + update() |

Class ledakan:

Memiliki property :

* self.image untuk menentukan gambar ledakan
* self.index untuk menentukan indeks
* self.rect untuk menentukan posisi ledakan
* self.rect.center untuk menentukan posisi ledakan saat di tengah

Memiliki method:

+ update() untuk memperbarui ledakan

1. **Program ”Galaxy Wars”**

import os

import pygame

import sys

import time

import math

import random

from pygame.locals import \*

pygame.init()

size = (width, height) = (1024, 600)

black = (0, 0, 0)

white = (255, 255, 255)

green = (0, 155, 0)

red = (155, 0, 0)

sky = (0, 0, 0)

clock = pygame.time.Clock()

FPS = 21

maxspeed = 15

screen = pygame.display.set\_mode(size)

def cpumove(cpu, target):

if target.rect.left < cpu.rect.left:

cpu.pelatuk = 1

cpu.cepat = -2

elif target.rect.left > cpu.rect.left:

cpu.pelatuk = 1

cpu.cepat = 2

if random.randrange(0, 30) == 1:

cpu.tembak = 1

else:

cpu.tembak = 0

def bossmove(cpu, target):

if target.rect.left < cpu.rect.left and cpu.spree == False:

cpu.pelatuk = 1

cpu.cepat = -2

elif target.rect.left > cpu.rect.left and cpu.spree == False:

cpu.pelatuk = 1

cpu.cepat = 2

if random.randrange(0, 3) == 1 and cpu.spree == False:

cpu.bulletformation = 0

cpu.bulletspeed = 20

cpu.tembak = 1

else:

cpu.tembak = 0

if cpu.spree == False and random.randrange(0, 250) == 71:

cpu.spree = True

else:

pass

def load\_image(

name,

sizex=-1,

sizey=-1,

colorkey=None,

):

fullname = os.path.join('Sprites', name)

image = pygame.image.load(fullname)

image = image.convert()

if colorkey is not None:

if colorkey is -1:

colorkey = image.get\_at((0, 0))

image.set\_colorkey(colorkey, RLEACCEL)

if sizex != -1 or sizey != -1:

image = pygame.transform.scale(image, (sizex, sizey))

return (image, image.get\_rect())

def showhealthbar(

nyawa,

barcolor,

pos,

unit,

):

healthbar = pygame.Surface((nyawa \* unit, 10), pygame.SRCALPHA, 32)

healthbar = healthbar.convert\_alpha()

pygame.draw.rect(screen, barcolor, pos)

def displaytext(

text,

fontsize,

x,

y,

color,

):

font = pygame.font.SysFont('sawasdee', fontsize, True)

text = font.render(text, 1, color)

textpos = text.get\_rect(centerx=x, centery=y)

screen.blit(text, textpos)

def moveplayer(Player):

if Player.isautopilot == False:

if Player.rect.left >= 0 and Player.rect.right <= width:

if Player.pelatuk == 1:

Player.gerak[0] = Player.gerak[0] + Player.cepat

if Player.gerak[0] < -maxspeed:

Player.gerak[0] = -maxspeed

elif Player.gerak[0] > maxspeed:

Player.gerak[0] = maxspeed

elif Player.gerak[0] >= -maxspeed and Player.gerak[0] \

< 0 and Player.pelatuk == 2:

Player.gerak[0] += math.fabs(Player.gerak[0] / 20)

if Player.gerak[0] > 0:

Player.gerak[0] = 0

elif Player.gerak[0] <= maxspeed and Player.gerak[0] \

> 0 and Player.pelatuk == 2:

Player.gerak[0] -= math.fabs(Player.gerak[0] / 20)

if Player.gerak[0] < 0:

Player.gerak[0] = 0

else:

Player.autopilot()

def alurcerita(wavecounter):

if wavecounter >= 0 and wavecounter <= 700: # musuh

return 0

elif wavecounter > 700 and wavecounter <= 1100: # saucer

return 1

elif wavecounter > 1100 and wavecounter <= 1500: # drone

return 2

elif wavecounter > 1500 and wavecounter <= 1900: # musuh and saucer

return 3

elif wavecounter > 1900 and wavecounter <= 2300: # drone and saucer

return 4

elif wavecounter > 2300 and wavecounter <= 2700: # musuh and drones

return 5

elif wavecounter > 2700: # boss

return 6

class bintang:

def \_\_init\_\_(self,radius,warna,nofstars,cepat=5):

self.radius = radius

self.warna = warna

self.cepat = cepat

self.nofstars = nofstars

self.posbintang = [[0 for j in range(2)] for i in range(self.nofstars)]

for x in range(self.nofstars):

self.posbintang[x][0] = random.randrange(0, width)

self.posbintang[x][1] = random.randrange(0, height)

def gambarbintang(self):

for x in range(self.nofstars):

pygame.draw.circle(screen,self.warna,(self.posbintang[x][0],self.posbintang[x][1]),self.radius)

self.pindahbintang()

def pindahbintang(self):

for x in range(self.nofstars):

self.posbintang[x][1] += self.cepat

if self.posbintang[x][1] > height:

self.posbintang[x][1] = 0

class player(pygame.sprite.Sprite):

def \_\_init\_\_(self):

pygame.sprite.Sprite.\_\_init\_\_(self)

(self.image, self.rect) = load\_image('pesawat.png', 72,

72, -1)

#posisi\_pesawat

self.rect.top = size[1] - 100

self.rect.left = size[0]/2

self.cepat = 0

self.tembak = 0

self.gerak = [0, 0]

self.pelatuk = 0

self.nyawa = 200

self.kills = 0

self.skore = 0

self.tundatembak = 0

self.isautopilot = False

self.shot = False

self.won = False

def checkbounds(self):

if self.rect.left < 0:

self.rect.left = 0

self.gerak[0] = 0

self.cepat = 0

if self.rect.right > width:

self.rect.right = width

self.gerak[0] = 0

self.cepat = 0

def update(self):

self.rect = self.rect.move(self.gerak)

self.tundatembak += 1

if self.tembak == 1 and self.tundatembak%3 == 1:

self.bedil()

if self.nyawa > 200:

self.nyawa = 200

def drawplayer(self):

screen.blit(self.image, self.rect)

def bedil(self):

(x, y) = self.rect.center

self.shot = peluru(x - 14, y, (0, 255, 0), 1)

self.shot = peluru(x + 14, y, (0, 255, 0), 1)

def autopilot(self):

if self.rect.centerx < width / 2:

self.gerak[0] = 5

else:

self.gerak[0] = -5

if self.rect.centerx - width / 2 < 5 and self.rect.centerx \

- width / 2 > -5:

self.gerak[0] = 0

self.gerak[1] = -10

class boss(pygame.sprite.Sprite):

def \_\_init\_\_(self):

pygame.sprite.Sprite.\_\_init\_\_(self)

(self.image, self.rect) = load\_image('bossx.png', 200, 400, -1)

self.rect = self.image.get\_rect()

self.rect.top = 100

self.rect.left = random.randrange(0, width - 72)

self.cepat = 0

self.tembak = 0

self.gerak = [0, 0]

self.pelatuk = 0

self.nyawa = 600

self.bulletformation = 0

self.bulletspeed = 20

self.spreecount = 0

self.spree = False

self.shot = False

self.isautopilot = False

self.reloadtime = 0

def checkbounds(self):

if self.rect.left < 0:

self.rect.left = 0

self.gerak[0] = 0

self.cepat = 0

if self.rect.right > width:

self.rect.right = width

self.gerak[0] = 0

self.cepat = 0

def update(self):

self.checkbounds()

moveplayer(self)

self.rect = self.rect.move(self.gerak)

if self.tembak == 1 and self.reloadtime == 0:

self.bedil(self.bulletformation, self.bulletspeed)

if self.reloadtime > 0:

self.reloadtime -= 1

if self.nyawa <= 0:

self.kill()

if self.spree == True and self.spreecount <= 70:

self.spreecount += 1

if self.spreecount % 5 == 1:

self.gerak[0] = 0

self.cepat = 0

self.bedil(1, 10)

else:

pass

else:

self.spree = False

self.spreecount = 0

def drawplayer(self):

screen.blit(self.image, self.rect)

def bedil(self, bulletformation=0, bulletspeed=20):

(x, y) = self.rect.center

if bulletformation == 0:

self.shot = pelurumusuh(x, y + self.rect.height / 2, (255,

0, 255), [0, 1], bulletspeed)

self.shot = pelurumusuh(x - self.rect.width / 2 + 30, y

- self.rect.height / 2 + 50, (255,

0, 255), [0, 1], bulletspeed)

self.shot = pelurumusuh(x + self.rect.width / 2 - 30, y

- self.rect.height / 2 + 50, (255,

0, 255), [0, 1], bulletspeed)

elif bulletformation == 1:

self.shot = pelurumusuh(x, y, (255, 0, 255), [1.5, 1],

bulletspeed)

self.shot = pelurumusuh(x, y, (255, 0, 255), [-1.5, 1],

bulletspeed)

self.shot = pelurumusuh(x, y, (255, 0, 255), [1.2, 1],

bulletspeed)

self.shot = pelurumusuh(x, y, (255, 0, 255), [-1.2, 1],

bulletspeed)

self.shot = pelurumusuh(x, y, (255, 0, 255), [0, 1],

bulletspeed)

self.shot = pelurumusuh(x, y, (255, 0, 255), [0.9, 1],

bulletspeed)

self.shot = pelurumusuh(x, y, (255, 0, 255), [-0.9, 1],

bulletspeed)

self.shot = pelurumusuh(x, y, (255, 0, 255), [0.6, 1],

bulletspeed)

self.shot = pelurumusuh(x, y, (255, 0, 255), [-0.6, 1],

bulletspeed)

self.shot = pelurumusuh(x, y, (255, 0, 255), [0.3, 1],

bulletspeed)

self.shot = pelurumusuh(x, y, (255, 0, 255), [-0.3, 1],

bulletspeed)

if random.randrange(0, 10) == 4:

musuh(random.randrange(0, 4))

if random.randrange(0, 50) == 41:

enemysaucer(random.randrange(0, width - 50))

if random.randrange(0, 200) == 121:

enemydrone(random.randrange(0, width - 50))

class musuh(pygame.sprite.Sprite):

def \_\_init\_\_(self, n=0):

pygame.sprite.Sprite.\_\_init\_\_(self, self.containers)

sheet = pygame.image.load('Sprites/enemy\_musuh.png')

self.images = []

rect = pygame.Rect((0, 0, 85, 92))

image = pygame.Surface(rect.size)

image.blit(sheet, (0, 0), rect)

self.images.append(image)

rect = pygame.Rect((86, 0, 71, 92))

image = pygame.Surface(rect.size)

image.blit(sheet, (0, 0), rect)

self.images.append(image)

rect = pygame.Rect((158, 0, 68, 92))

image = pygame.Surface(rect.size)

image.blit(sheet, (0, 0), rect)

self.images.append(image)

rect = pygame.Rect((227, 0, 65, 92))

image = pygame.Surface(rect.size)

image.blit(sheet, (0, 0), rect)

self.images.append(image)

self.image = self.images[n]

self.image = self.image.convert()

colorkey = -1

colorkey = self.image.get\_at((10, 10))

self.image.set\_colorkey(colorkey, RLEACCEL)

self.image = pygame.transform.scale(self.image, (36, 36))

self.rect = self.image.get\_rect()

self.image = pygame.transform.rotate(self.image, 180)

self.rect.top = 0

self.rect.left = random.randrange(0, width - 72)

self.cepat = 0

self.tembak = 0

self.gerak = [0, 0]

self.pelatuk = 0

self.nyawa = 2

self.isautopilot = False

self.ledakan\_sound = \

pygame.mixer.Sound('Sprites/explosion.wav')

self.ledakan\_sound.set\_volume(0.1)

self.shot = False

def checkbounds(self):

if self.rect.left < 0:

self.rect.left = 0

self.gerak[0] = 0

self.cepat = 0

if self.rect.right > width:

self.rect.right = width

self.gerak[0] = 0

self.cepat = 0

def update(self):

self.checkbounds()

moveplayer(self)

self.autopilot()

self.rect = self.rect.move(self.gerak)

if self.tembak == 1:

self.bedil()

if self.nyawa <= 0:

(x, y) = self.rect.center

if pygame.mixer.get\_init():

self.ledakan\_sound.play(maxtime=1000)

ledakan(x, y)

self.kill()

def drawplayer(self):

screen.blit(self.image, self.rect)

def bedil(self):

(x, y) = self.rect.center

self.shot = pelurumusuh(x, y, (255, 255, 0), [0, 1], 12)

def autopilot(self):

if self.rect.top < height:

self.gerak[1] = 5

else:

self.kill()

class enemydrone(pygame.sprite.Sprite):

def \_\_init\_\_(self, x):

pygame.sprite.Sprite.\_\_init\_\_(self, self.containers)

(self.image, self.rect) = load\_image('drone.png', 50,

102, -1)

self.rect.top = -self.rect.height

self.rect.left = x

self.cepat = 0

self.tembak = 1

self.gerak = [0, 0]

self.nyawa = 20

self.shot = False

self.waitTime = 0

self.ledakan\_sound = \

pygame.mixer.Sound('Sprites/explosion.wav')

self.ledakan\_sound.set\_volume(0.1)

def checkbounds(self):

if self.rect.left < 0:

self.rect.left = 0

self.gerak[0] = 0

self.cepat = 0

if self.rect.right > width:

self.rect.right = width

self.gerak[0] = 0

self.cepat = 0

def update(self):

self.checkbounds()

self.autopilot()

self.rect = self.rect.move(self.gerak)

if self.tembak == 1 and self.waitTime % 10 == 1:

self.bedil()

if self.nyawa <= 0:

(x, y) = self.rect.center

if pygame.mixer.get\_init():

self.ledakan\_sound.play(maxtime=1000)

ledakan(x, y,100)

self.kill()

def drawplayer(self):

screen.blit(self.image, self.rect)

def bedil(self):

(x, y) = self.rect.center

self.shot = pelurumusuh(x, y + self.rect.height / 2, (255, 0,

0), [0, 1], 10)

self.shot = pelurumusuh(x, y + self.rect.height / 2, (255, 0,

0), [-0.5, 1], 10)

self.shot = pelurumusuh(x, y + self.rect.height / 2, (255, 0,

0), [0.5, 1], 10)

self.shot = pelurumusuh(x, y + self.rect.height / 2, (255, 0,

0), [-1, 1], 10)

self.shot = pelurumusuh(x, y + self.rect.height / 2, (255, 0,

0), [1, 1], 10)

def autopilot(self):

if self.rect.top < height - 500:

self.gerak[1] = 3

elif self.rect.top > height - 500 and self.waitTime < 1000:

self.gerak[1] = 0

self.waitTime += 1

if self.waitTime >= 150:

self.gerak[1] = 5

if self.rect.top > height:

self.kill()

class enemysaucer(pygame.sprite.Sprite):

def \_\_init\_\_(self, x):

pygame.sprite.Sprite.\_\_init\_\_(self, self.containers)

sheet = pygame.image.load('Sprites/saucer.png')

self.images = []

for i in range(0, 672, 96):

rect = pygame.Rect((i, 0, 96, 96))

image = pygame.Surface(rect.size)

image = image.convert()

colorkey = -1

colorkey = image.get\_at((10, 10))

image.set\_colorkey(colorkey, RLEACCEL)

image.blit(sheet, (0, 0), rect)

image = pygame.transform.scale(image, (48, 48))

self.images.append(image)

self.image = self.images[0]

self.index = 0

self.rect = self.image.get\_rect()

self.rect.center = (x, -self.rect.height)

self.nyawa = 10

self.waitTime = 0

self.tembak = 1

self.gerak = [0, 0]

self.haltpos = random.randrange(300, 510)

self.shot = False

self.ledakan\_sound = \

pygame.mixer.Sound('Sprites/explosion.wav')

self.ledakan\_sound.set\_volume(0.1)

def checkbounds(self):

if self.rect.left < 0:

self.rect.left = 0

self.gerak[0] = 0

self.cepat = 0

if self.rect.right > width:

self.rect.right = width

self.gerak[0] = 0

self.cepat = 0

def update(self):

self.checkbounds()

self.autopilot()

self.rect = self.rect.move(self.gerak)

if self.tembak == 1 and self.waitTime % 10 == 1:

self.bedil()

if self.nyawa <= 0:

(x, y) = self.rect.center

if pygame.mixer.get\_init():

self.ledakan\_sound.play(maxtime=1000)

ledakan(x, y,75)

self.kill()

self.index += 1

self.index = self.index % 7

self.image = self.images[self.index]

self.image = pygame.transform.rotate(self.image, 90)

self.images[self.index] = self.image

def drawplayer(self):

screen.blit(self.image, self.rect)

def bedil(self):

(x, y) = self.rect.center

self.shot = pelurumusuh(x, y, (0, 0, 255), [0, 1], 18)

def autopilot(self):

if self.rect.top < height - self.haltpos:

self.gerak[1] = 3

elif self.rect.top > height - self.haltpos and self.waitTime \

< 1000:

self.gerak[1] = 0

self.waitTime += 1

if self.waitTime >= 150:

self.gerak[1] = 5

if self.rect.top > height:

self.kill()

class kotaksehat(pygame.sprite.Sprite):

def \_\_init\_\_(

self,

x,

y,

nyawa,

):

pygame.sprite.Sprite.\_\_init\_\_(self, self.containers)

self.nyawa = nyawa

(self.image, self.rect) = load\_image('healthpack1.png', 40, 40,

-1)

self.rect.left = x

self.rect.top = y

self.gerak = [3, 0]

self.maxleft = self.rect.left - 20

self.maxright = self.rect.right + 20

def checkbounds(self):

if self.rect.left < 0:

self.rect.left = 0

self.gerak[0] = 0

self.cepat = 0

if self.rect.right > width:

self.rect.right = width

self.gerak[0] = 0

self.cepat = 0

def update(self):

self.checkbounds()

self.autopilot()

self.rect = self.rect.move(self.gerak)

if self.nyawa <= 0 or self.rect.top > height:

self.kill()

def drawplayer(self):

screen.blit(self.image, self.rect)

def autopilot(self):

if self.rect.right > self.maxright:

self.gerak[0] = -3

elif self.rect.left < self.maxleft:

self.gerak[0] = 3

self.gerak[1] = 5

class peluru(pygame.sprite.Sprite):

def \_\_init\_\_(

self,

x,

y,

color,

direction=1,

):

pygame.sprite.Sprite.\_\_init\_\_(self, self.containers)

self.image,self.rect = load\_image('lazer1.png',5,25,-1)

self.rect.center = (x, y - direction \* 20)

self.direction = direction

def update(self):

(x, y) = self.rect.center

y -= self.direction \* 20

self.rect.center = (x, y)

if y <= 0 or y >= height:

self.kill()

class pelurumusuh(pygame.sprite.Sprite):

def \_\_init\_\_(

self,

x,

y,

color,

direction,

cepat,

):

pygame.sprite.Sprite.\_\_init\_\_(self, self.containers)

self.image = pygame.Surface((10, 10), pygame.SRCALPHA, 32)

self.image = self.image.convert\_alpha()

self.col = list(color)

for i in range(5, 0, -1):

self.col[0] = color[0] \* float(i) / 5

self.col[1] = color[1] \* float(i) / 5

self.col[2] = color[2] \* float(i) / 5

pygame.draw.circle(self.image, tuple(self.col), (5, 5), i,

0)

self.rect = self.image.get\_rect()

self.rect.center = (x, y)

self.direction = direction

self.cepat = cepat

def update(self):

(x, y) = self.rect.center

y += self.direction[1] \* self.cepat

x += self.direction[0] \* self.cepat

self.rect.center = (x, y)

if y <= 0 or y >= height or x <= 0 or x >= width:

self.kill()

class ledakan(pygame.sprite.Sprite):

def \_\_init\_\_(self, x, y,radius=-1):

pygame.sprite.Sprite.\_\_init\_\_(self, self.containers)

sheet = pygame.image.load('Sprites/ledakan.png')

self.images = []

for i in range(0, 768, 48):

rect = pygame.Rect((i, 0, 48, 48))

image = pygame.Surface(rect.size)

image = image.convert()

colorkey = -1

colorkey = image.get\_at((10, 10))

image.set\_colorkey(colorkey, RLEACCEL)

image.blit(sheet, (0, 0), rect)

if radius != -1:

image = pygame.transform.scale(image,(radius,radius))

self.images.append(image)

self.image = self.images[0]

self.index = 0

self.rect = self.image.get\_rect()

self.rect.center = (x, y)

def update(self):

self.image = self.images[self.index]

self.index += 1

if self.index >= len(self.images):

self.kill()

def main():

gameOver = False

menuExit = False

stageStart = False

bossStage = False

gameOverScreen = False

menuselect = -1

menuhighlight = 0

wavecounter = 0

wave = 0

bntgjth1 = bintang(1,white,50,5)

bullets = pygame.sprite.Group()

enemybullets = pygame.sprite.Group()

enemies = pygame.sprite.Group()

explosions = pygame.sprite.Group()

drones = pygame.sprite.Group()

saucers = pygame.sprite.Group()

healthpacks = pygame.sprite.Group()

peluru.containers = bullets

pelurumusuh.containers = enemybullets

musuh.containers = enemies

ledakan.containers = explosions

enemydrone.containers = drones

enemysaucer.containers = saucers

kotaksehat.containers = healthpacks

user = player()

pygame.display.set\_caption('GalaxyWars')

bg\_music = pygame.mixer.Sound('Sprites/bg\_music1.ogg')

boss\_music = pygame.mixer.Sound('Sprites/boss\_music.ogg')

(logoimage, logorect) = load\_image('gamelogo1.png', -1, -1, -1)

logorect.left = width / 2 - logorect.width / 2

logorect.top = height / 2 - logorect.height \* 5 / 4

bg,bgrect = load\_image('bg5.png')

while not gameOver:

while not menuExit:

for event in pygame.event.get():

if event.type == pygame.QUIT:

menuExit = True

gameOver = True

if event.type == pygame.KEYDOWN:

if event.key == pygame.K\_DOWN or event.key \

== pygame.K\_UP:

menuhighlight += 1

elif event.key == pygame.K\_RETURN:

menuselect = menuhighlight % 2

if menuselect == 0:

stageStart = True

menuExit = True

bg\_music.play(-1)

elif menuselect == 1:

pygame.quit()

quit()

else:

pass

screen.blit(bg,bgrect)

bntgjth1.gambarbintang()

user.drawplayer()

screen.blit(logoimage, logorect)

displaytext('Play', 32, width / 2 - 20, height \* 3 / 4

- 40, white)

displaytext('Exit', 32, width / 2 - 20, height \* 3 / 4,

white)

if menuhighlight % 2 == 0:

screen.blit(pygame.transform.scale(user.image, (25,

25)), [width / 2 - 100, height \* 3 / 4

- 55, 15, 15])

elif menuhighlight % 2 == 1:

screen.blit(pygame.transform.scale(user.image, (25,

25)), [width / 2 - 100, height \* 3 / 4

- 15, 15, 15])

pygame.display.update()

clock.tick(FPS)

while stageStart:

for event in pygame.event.get():

if event.type == pygame.QUIT:

stageStart = False

gameOver = True

if event.type == pygame.KEYDOWN:

user.pelatuk = 1

if event.key == pygame.K\_LEFT:

user.cepat = -2

elif event.key == pygame.K\_RIGHT:

user.cepat = 2

elif event.key == pygame.K\_UP:

user.tembak = 1

elif event.key == pygame.K\_ESCAPE:

quit()

if event.type == pygame.KEYUP:

if event.key == pygame.K\_LEFT or event.key \

== pygame.K\_RIGHT:

user.pelatuk = 2

user.cepat = 0

if event.key == pygame.K\_UP:

user.tembak = 0

if wavecounter % 500 == 499 and random.randrange(0, 2) == 1 \

and len(healthpacks) < 1:

kotaksehat(random.randrange(0, width - 50), 0, 10)

if random.randrange(0, 8) == 1 and len(enemies) < 10 \

and (wave == 0 or wave == 3 or wave == 5 or wave == 6):

musuh(random.randrange(0, 4))

if random.randrange(0, 20) == 1 and len(saucers) < 3 \

and (wave == 1 or wave == 3 or wave == 4 or wave == 5):

enemysaucer(random.randrange(0, width - 50))

if random.randrange(0, 30) == 21 and len(drones) < 2 \

and (wave == 2 or wave == 3 or wave == 4):

if len(drones) > 0:

for drone in drones:

if drone.rect.left < width / 2:

enemydrone(random.randrange(width / 2 + 60,

width - 60))

else:

enemydrone(random.randrange(0, width / 2

- 60))

else:

enemydrone(random.randrange(0, width - 60))

if wave == 6:

bossStage = True

stageStart = False

finalboss = boss()

user.nyawa += 80

user.rect.left = width / 2

user.rect.top = size[1] - 100

user.isautopilot = False

user.gerak = [0, 0]

boss\_music.play(-1)

for ship in enemies:

cpumove(ship, user)

for musuhhit in pygame.sprite.groupcollide(enemies,

bullets, 0, 1):

musuhhit.nyawa -= 1

if musuhhit.nyawa <= 0:

user.kills += 1

user.skore += 1

for dronehit in pygame.sprite.groupcollide(drones, bullets,

0, 1):

dronehit.nyawa -= 1

if dronehit.nyawa <= 0:

user.kills += 1

user.skore += 10

for saucerhit in pygame.sprite.groupcollide(saucers,

bullets, 0, 1):

saucerhit.nyawa -= 1

if saucerhit.nyawa <= 0:

user.kills += 1

user.skore += 5

for firedbullet in pygame.sprite.spritecollide(user,

enemybullets, 1):

user.nyawa -= 1

for enemycollided in enemies:

if pygame.sprite.collide\_mask(user, enemycollided):

user.nyawa -= 2

enemycollided.nyawa -= enemycollided.nyawa

for dronecollided in drones:

if pygame.sprite.collide\_mask(user, dronecollided):

user.nyawa -= 10

dronecollided.nyawa -= dronecollided.nyawa

for saucercollided in saucers:

if pygame.sprite.collide\_mask(user, saucercollided):

user.nyawa -= 4

saucercollided.nyawa -= saucercollided.nyawa

for health\_pack in healthpacks:

if pygame.sprite.collide\_mask(user, health\_pack):

user.nyawa += health\_pack.nyawa

health\_pack.nyawa -= health\_pack.nyawa

if user.nyawa <= 0:

gameOverScreen = True

stageStart = False

user.update()

user.checkbounds()

screen.blit(bg,bgrect)

bntgjth1.gambarbintang()

if user.nyawa > 0:

showhealthbar(user.nyawa, green, [100, height - 20,

user.nyawa \* 4, 10], 4)

displaytext('HEALTH', 22, 50, height - 15, white)

displaytext('Score:', 22, width - 100, 15, white)

displaytext(str(user.skore), 22, width - 35, 15, white)

user.drawplayer()

enemies.update()

bullets.update()

enemybullets.update()

explosions.update()

drones.update()

saucers.update()

healthpacks.update()

bullets.draw(screen)

enemybullets.draw(screen)

enemies.draw(screen)

explosions.draw(screen)

drones.draw(screen)

saucers.draw(screen)

healthpacks.draw(screen)

wave = alurcerita(wavecounter)

wavecounter += 1

pygame.display.update()

clock.tick(FPS)

moveplayer(user)

print (

wavecounter,

wave,

user.kills,

user.nyawa,

user.rect.left,

user.gerak[0],

user.rect.right,

)

while bossStage:

for event in pygame.event.get():

if event.type == pygame.QUIT:

gameOver = True

bossStage = False

if event.type == pygame.KEYDOWN:

user.pelatuk = 1

if event.key == pygame.K\_LEFT:

user.cepat = -2

elif event.key == pygame.K\_RIGHT:

user.cepat = 2

elif event.key == pygame.K\_UP:

user.tembak = 1

elif event.key == pygame.K\_ESCAPE:

quit()

if event.type == pygame.KEYUP:

if event.key == pygame.K\_LEFT or event.key \

== pygame.K\_RIGHT:

user.pelatuk = 2

user.cepat = 0

if event.key == pygame.K\_UP:

user.tembak = 0

bossmove(finalboss, user)

for ship in enemies:

cpumove(ship, user)

for userbullet in bullets:

if pygame.sprite.collide\_mask(finalboss, userbullet):

if finalboss.nyawa > 2:

finalboss.nyawa -= 1

else:

bossStage = False

gameOverScreen = True

user.skore += 200

user.won = True

userbullet.kill()

for musuhhit in pygame.sprite.groupcollide(enemies,

bullets, 0, 1):

musuhhit.nyawa -= 1

if musuhhit.nyawa <= 0:

user.kills += 1

user.skore += 1

for dronehit in pygame.sprite.groupcollide(drones, bullets,

0, 1):

dronehit.nyawa -= 1

if dronehit.nyawa <= 0:

user.kills += 1

user.skore += 10

for saucerhit in pygame.sprite.groupcollide(saucers,

bullets, 0, 1):

saucerhit.nyawa -= 1

if saucerhit.nyawa <= 0:

user.kills += 1

user.skore += 5

for firedbullet in pygame.sprite.spritecollide(user,

enemybullets, 1):

user.nyawa -= 1

for musuhcollided in enemies:

if pygame.sprite.collide\_mask(user, musuhcollided):

user.nyawa -= 2

musuhcollided.nyawa -= musuhcollided.nyawa

for dronecollided in drones:

if pygame.sprite.collide\_mask(user, dronecollided):

user.nyawa -= 10

dronecollided.nyawa -= dronecollided.nyawa

for saucercollided in saucers:

if pygame.sprite.collide\_mask(user, saucercollided):

user.nyawa -= 4

saucercollided.nyawa -= saucercollided.nyawa

if user.nyawa <= 0:

gameOverScreen = True

bossStage = False

user.update()

user.checkbounds()

screen.blit(bg,bgrect)

bntgjth1.gambarbintang()

if user.nyawa > 0:

showhealthbar(user.nyawa, green, [100, height - 20,

user.nyawa \* 4, 10], 4)

displaytext('HEALTH', 22, 50, height - 15, white)

if finalboss.nyawa > 0:

showhealthbar(finalboss.nyawa, red, [100, 20,

finalboss.nyawa \* 0.8, 10], 0.8)

displaytext('BOSS', 22, 50, 25, white)

displaytext('Score:', 22, width - 100, 15, white)

displaytext(str(user.skore), 22, width - 35, 15, white)

user.drawplayer()

enemies.update()

bullets.update()

enemybullets.update()

drones.update()

saucers.update()

explosions.update()

finalboss.update()

bullets.draw(screen)

enemybullets.draw(screen)

enemies.draw(screen)

drones.draw(screen)

saucers.draw(screen)

explosions.draw(screen)

finalboss.drawplayer()

pygame.display.update()

clock.tick(FPS)

moveplayer(user)

while gameOverScreen:

for event in pygame.event.get():

if event.type == pygame.QUIT:

gameOverScreen = False

gameOver = True

if event.type == pygame.KEYDOWN:

if event.key == pygame.K\_RETURN:

gameOverScreen = False

gameOver = True

screen.fill(sky)

bntgjth1.gambarbintang()

if user.won == False:

displaytext('Game Over', 26, width / 2 - 30, height

/ 2, white)

else:

displaytext('Congratulations! You Won!', 26, width / 2

- 30, height / 2, white)

displaytext('Your score: ', 26, width / 2 - 40, height / 2

+ 40, white)

displaytext(str(user.skore), 26, width / 2 + 50, height / 2

+ 43, white)

displaytext('Press Enter to exit...', 14, width / 2 - 30,

height / 2 + 90, white)

pygame.display.update()

clock.tick(FPS)

pygame.quit()

quit()

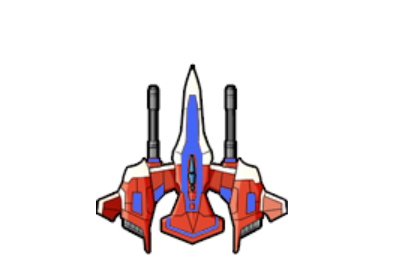
main()

1. **Aset yang digunakan**
2. Gambar

* Ini adalah background yang digunakan pada pygame



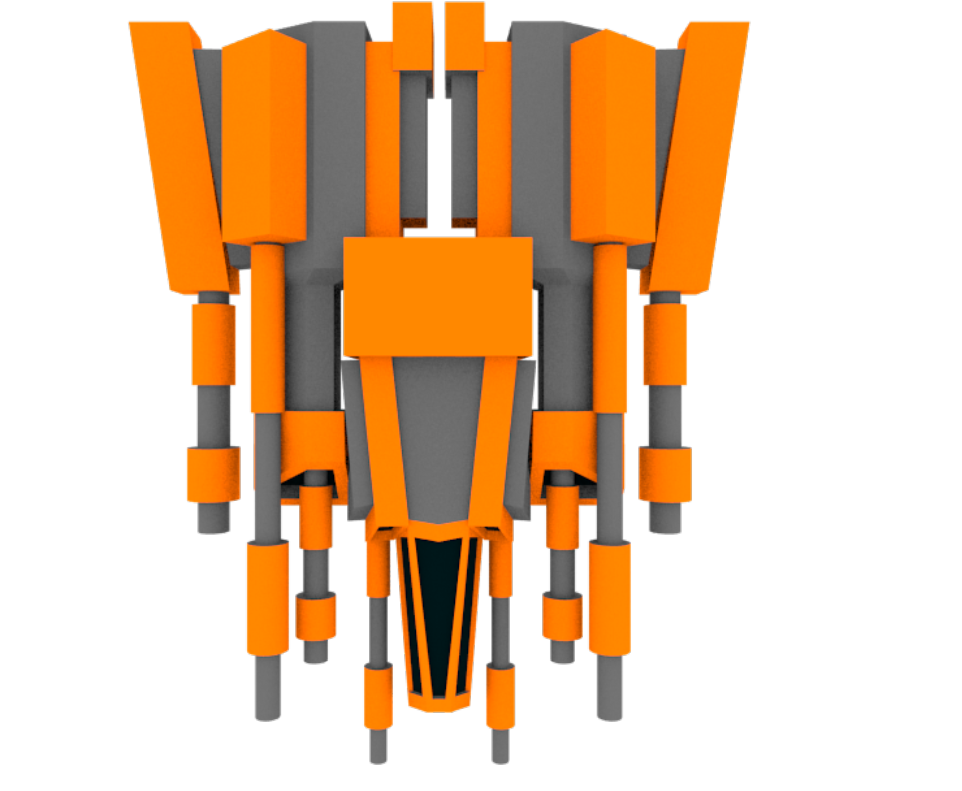
* Ini adalah gambar Hero yang melawan para musuh



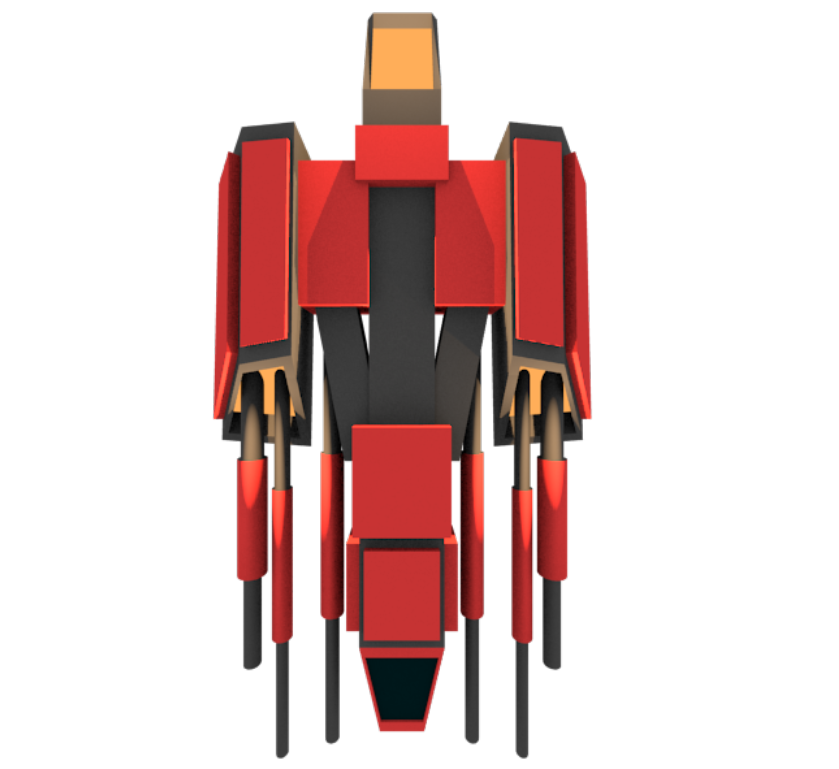
* Ini adalah logo pygame



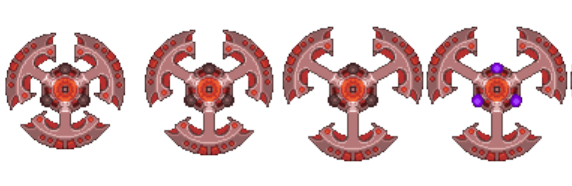
* Ini adalah gambar Boss atau musuh yang utama pada pygame



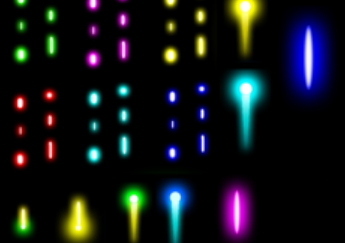
* Ini adalah gambar musuh yang lainnya







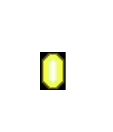
* Ini adalah gambar yang seolah-olah akan menjadi bintang jatuh



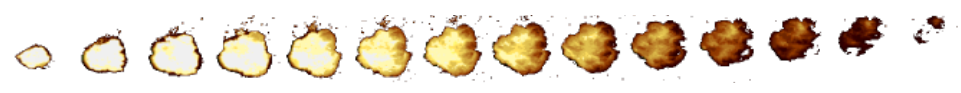
* Ini adalah gambar healthpack



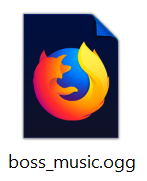
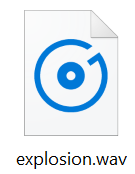
* Ini adalah gambar lazer



* Ini adalah gambar ledakan ketika berhasil membunuh musuh



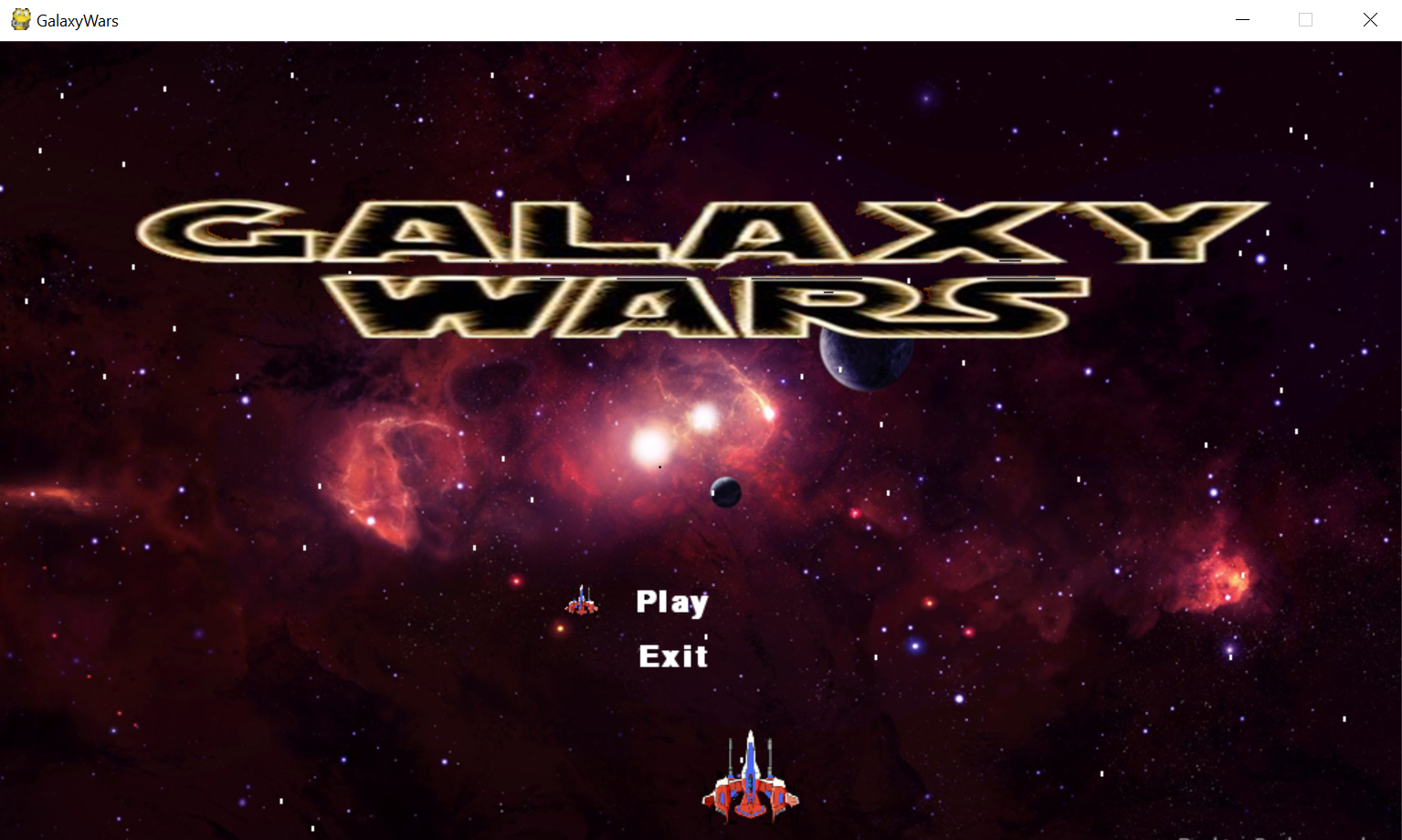
1. Suara

Yang pertama adalah suara background pygame, yang kedua adalah suara ketika musuh yang utama muncul, yang ketiga adalah suara ledakan ketika berhasil membunuh musuh.

**f. Screen tampilan Pygame ”Galaxy Wars”**

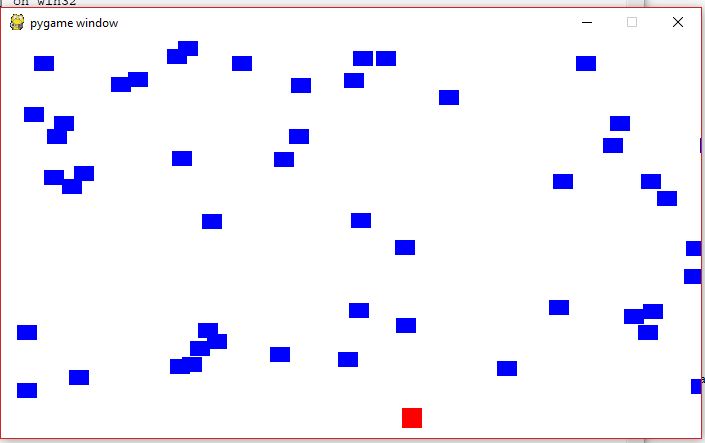
Untuk project pygame kali ini kami membuat sebuah game shotter yang kami berinama **”Galaxy Wars”** yang terinspirasi dari game Bullets yang ada pada web kivy. Berikut ini contoh tampilan dari gamenya.



**g.** **Tinjauan Pustaka**

Terinspirasi dari :

Link : <http://programarcadegames.com/python_examples/show_file.php?file=bullets.py>

****

**KIVY**

1. **Pegertian Kivy**

Kivy merupakan framework yang dibangun menggunakan library dari bahasa pemrograman python yang bersifat open source. Tujuan dikembangkannya framework ini agar dapat membantu developer secara cepat dalam mengembangkan aplikasi yang memiliki tampilan antarmuka inovatif seperti aplikasi yang mendukung multitouch.

Framework ini dapat dijalankan pada system operasi Windows, Mac OS, Linux, Android, hingga Raspberry Pi. Selain itu, kivy mendukung untuk input beragam perangkat seperti WM\_Touch, WM\_Pen, Mac OS X Trackpad, Magi Mouse, Mtdev, Linux kernel HID, dan TUIO.

1. **Instalasi Kivy**

Sebelum menggunakan modul kivy terlebih dulu kita harus melakukan instalasi, dengan mengikuti cara-cara berikut ini :

* Install PIP

PIP adalah sebuah aplikasi system manajemen package yang digunakan untuk menginstall dan mengelola package yang ditulis dengan python. Untuk install ketikkan perintah berikut :

**python –m pip install –upgrade pip wheel setuptools** tunggu hingga proses selesai.

* Install Dependencies

Untuk menginstall ketikkan perintah ini :

**python –m pip install docutils pygments pypwin32 kivy.deps.sdl2 kivy.deps.glew** dan selanjutnya ketikkan **python –m pip install kivy.deps.gstremer** tunggu hingga proses selesai. **Note :** Jika menemukan Error saat menginstal, setelah menginstal pip tambahkan **opsi –no-cache-dir**.

* Install Kivy

Ketikkan perintah berikut :

**Python –m pip install kivy** tunggu hingga selesai.

1. **Penjelasan OOP pada kivy ”Stopwatch”**

StackLayout

StackLayout

|  |
| --- |
| Timer |
| - size\_hint\_y  - timer\_name  - visible\_time  - start\_btn  - reset\_timer\_btn  - remove\_timer\_btn  - add\_widget  - total\_seconds  - stop\_time  - running |
| + update ()  + clk\_start\_btn()  + clk\_reset\_timer\_btn()  + clk\_remove\_timer\_btn() |

|  |
| --- |
| MainScreen |
| - size\_hint\_y  - bind  - add\_timer\_btn  - save\_and\_quit\_btn  - add\_widget |
| + clk\_add\_timer\_btn()  + create\_timer ()  + clk\_save\_and\_quit ()  + save () |

TextInput

|  |
| --- |
| LimitedInput |
| - |
| + insert\_text() |

|  |
| --- |
| MainScroll |
| - size\_hint  - size  - scroll\_timeout  - add\_widget |
| + |

App

ScrollView

|  |
| --- |
| TimeYourselfApp |
| - |
| + build() |

Class dan Method :

|  |
| --- |
| MainScreen |
| - size\_hint\_y  - bind  - add\_timer\_btn  - save\_and\_quit\_btn  - add\_widget |
| + clk\_add\_timer\_btn()  + create\_timer ()  + clk\_save\_and\_quit ()  + save () |

Penjelasan Class

Semua Class di program game Kivy Stopwatch berada di Public tidak ada yang private.  
  
Class MainScreen termasuk class : Inheritance Super  
  
Class LimitedInput termasuk class : inheritance overriding  
  
Class Timer termasuk class : inheritance overriding  
  
Class MainScroll termasuk class : inheritance overriding

Class TimeYourselfApp termasuk class : inheritance overriding  
Memiliki property :

* Size\_hint\_y untuk pemberian kondisi petunjuk ukuran
* bind untuk mengatur tinggi min
* add\_timer\_btn untuk membuat button agar menambahkan stopwatch
* save\_and\_quit\_btn untuk membuat button agar dapat menyimpan data dan keluar dari program
* add\_widget untuk membuat widget

Method :

+ clk\_add\_timer\_btn() untuk membuat timer button

+ create\_timer () untuk mengatur jalan dan tampilan timer

+ clk\_save\_and\_quit () untuk membuat kondisi ketika mengklik button save n quit

+ save () untuk menyimpan data timer

|  |
| --- |
| Timer |
| - size\_hint\_y  - timer\_name  - visible\_time  - start\_btn  - reset\_timer\_btn  - remove\_timer\_btn  - add\_widget  - total\_seconds  - stop\_time  - running |
| + update ()  + clk\_start\_btn()  + clk\_reset\_timer\_btn()  + clk\_remove\_timer\_btn() |

Class dan Method :

Memiliki property :

* Size\_hint\_y untuk pemberian kondisi petunjuk ukuran
* Timer\_name untuk membuat kolom isian nama timer
* visible\_time untuk memperlihatkan waktu
* start\_btn untuk membuat button start
* reset\_timer\_btn untuk membuat button reset
* remove\_timer\_btn untuk membuat button remove
* add\_widget untuk menambahkan widget
* total\_seconds untuk pemberian kondisi total waktu
* stop\_time untuk pemberian kondisi saat waktu berhenti
* running untuk pemberian kondisi saat app berjalan

Method :

+ update () untuk mengupdate waktu timer

+ clk\_start\_btn() untuk pemberian kondisi saat ingin memulai start dan berhenti atau stop

+ clk\_reset\_timer\_btn() untuk pemberian kondisi ketika ingin mereset timer

+ clk\_remove\_timer\_btn() untuk pemberian kondisi ketika ingin remove

Class dan Method :

|  |
| --- |
| LimitedInput |
| - |
| + insert\_text() |

Memiliki property :

* Tidak memiliki property

Method :

+ insert\_text() untuk pemberian kondisi dan pengisian kolom nama timer

Class dan Method :

|  |
| --- |
| MainScroll |
| - size\_hint  - size  - scroll\_timeout  - add\_widget |
| + |

Memiliki property :

* size\_hint untuk pemberian kondisi petunjuk ukuran
* size untuk mengatur lebar dan tinggi window
* scroll\_timeout untuk mengatur timeout timer
* add\_widget untuk menambahkan widget

Method :

+ Tidak memiliki method

Class dan Method :

|  |
| --- |
| TimeYourselfApp |
| - |
| + build() |

Memiliki property :

* Tidak memiliki property

Method :

+ build() untuk membuat objeck dari class dan mempunyai nilai pengembalian dari class

MainScroll

1. **Program ”Stopwatch”**

import time

import pickle

import kivy

from kivy.app import App

from kivy.uix.label import Label

from kivy.uix.textinput import TextInput

from kivy.uix.button import Button

from kivy.uix.stacklayout import StackLayout

from kivy.uix.scrollview import ScrollView

from kivy.clock import Clock

from kivy.config import Config

Config.set('graphics', 'resizable', False)

from kivy.core.window import Window

Window.size = (Window.width//2, Window.height//1.1)

kivy.require('1.9.1')

FONT\_SIZE = 11

class MainScreen(StackLayout):

timers = []

def \_\_init\_\_(self, \*\*kwargs):

super(MainScreen, self).\_\_init\_\_(\*\*kwargs) #inheritance super

self.size\_hint\_y = None

self.bind(minimum\_height=self.setter('height'))

self.add\_timer\_btn = Button(text="Tambahkan Baru", size\_hint=(.5, None), height=Window.height//12,

font\_size=FONT\_SIZE)

self.add\_timer\_btn.bind(on\_release=self.clk\_add\_timer\_btn)

self.save\_and\_quit\_btn = Button(text='Simpan dan Keluar', size\_hint=(.5, None), height=Window.height//12,

font\_size=FONT\_SIZE)

self.save\_and\_quit\_btn.bind(on\_release=self.clk\_save\_and\_quit)

self.add\_widget(self.add\_timer\_btn)

self.add\_widget(self.save\_and\_quit\_btn)

# load saved data

for timer\_data in load\_data().values():

self.create\_timer(total\_seconds=timer\_data['total\_seconds'], timer\_name=timer\_data['timer\_name'])

# auto save

Clock.schedule\_interval(self.save, 1)

def clk\_add\_timer\_btn(self, obj):

self.create\_timer()

def create\_timer(self, total\_seconds=0, timer\_name=''):

timer = Timer(height=Window.height//6.5)

timer.total\_seconds = total\_seconds

timer.timer\_name.text = timer\_name

timer.visible\_time.text = convert\_seconds\_to\_text(total\_seconds)

self.add\_widget(timer)

self.timers.append(timer)

def clk\_save\_and\_quit(self, \*args):

if self.save():

TimeYourselfApp().stop()

def save(self, \*args):

timers\_dic = {}

for t in self.timers:

timers\_dic[self.timers.index(t)] = {"total\_seconds": t.total\_seconds, "timer\_name": t.timer\_name.text}

try:

with open('saved\_data', 'wb') as fp: #save data pada sebuah file

pickle.dump(timers\_dic, fp)

except Exception as e:

print("Exception when saving data: {}".format(e))

else:

# People looked trough my code till the end

return True

def convert\_seconds\_to\_text(total\_seconds=0):

days = int(total\_seconds // 86400)

if days == 1:

word\_days = ' day '

else:

word\_days = ' days '

days = str(days) + word\_days

hours = int(total\_seconds // 3600 % 24)

if hours == 1:

word\_hours = ' hour '

else:

word\_hours = ' hours '

hours = str(hours) + word\_hours

minutes = int(total\_seconds // 60 % 60)

if minutes == 1:

word\_minutes = ' minute '

else:

word\_minutes = ' minutes '

minutes = str(minutes) + word\_minutes

seconds = total\_seconds % 60

return '{}{}{}{:.1f} s'.format(days, hours, minutes, seconds)

def load\_data():

try:

with open('saved\_data', 'rb') as fp:

data = pickle.load(fp)

except Exception as e:

print("Exception when loading saved data: {}".format(e))

return {}

return data

class LimitedInput(TextInput):

def insert\_text(self, substring, from\_undo=False):

if len(self.text) > 25:

substring = ''

return super(LimitedInput, self).insert\_text(substring, from\_undo=from\_undo)

class Timer(StackLayout):

def \_\_init\_\_(self, \*\*kwargs):

super(Timer, self).\_\_init\_\_(\*\*kwargs)

self.size\_hint\_y = None

self.timer\_name = LimitedInput(hint\_text='Tulis sesuatu di sini', size\_hint=(.25, .5),

font\_size=FONT\_SIZE, allow\_copy=False)

self.visible\_time = Label(text=convert\_seconds\_to\_text(), size\_hint=(.75, .5), font\_size=FONT\_SIZE)

self.start\_btn = Button(text='Start', size\_hint=(.8, .5), font\_size=FONT\_SIZE)

self.start\_btn.bind(on\_release=self.clk\_start\_btn)

self.reset\_timer\_btn = Button(text='Reset', size\_hint=(.15, .5), font\_size=FONT\_SIZE)

self.reset\_timer\_btn.bind(on\_release=self.clk\_reset\_timer\_btn)

self.remove\_timer\_btn = Button(text='X', size\_hint=(.05, .5), font\_size=FONT\_SIZE, background\_color=[0.9, 0, 0, 1])

self.remove\_timer\_btn.bind(on\_release=self.clk\_remove\_timer\_btn)

self.add\_widget(self.timer\_name)

self.add\_widget(self.visible\_time)

self.add\_widget(self.start\_btn)

self.add\_widget(self.reset\_timer\_btn)

self.add\_widget(self.remove\_timer\_btn)

self.total\_seconds = 0

self.stop\_time = 0

self.running = False

def update(self, \*args):

self.total\_seconds = time.time() - self.stop\_time

self.visible\_time.text = convert\_seconds\_to\_text(self.total\_seconds)

def clk\_start\_btn(self, obj):

if self.running:

self.running = False

Clock.unschedule(self.update)

self.start\_btn.text = "Start"

else:

self.running = True

self.stop\_time = time.time() - self.total\_seconds

Clock.schedule\_interval(self.update, 0.1)

self.start\_btn.text = "Stop"

def clk\_reset\_timer\_btn(self, obj):

if self.running:

self.running = False

Clock.unschedule(self.update)

self.start\_btn.text = "Start"

self.total\_seconds = 0

self.visible\_time.text = convert\_seconds\_to\_text(self.total\_seconds)

def clk\_remove\_timer\_btn(self, obj):

self.parent.remove\_widget(self)

MainScreen.timers.remove(self)

class MainScroll(ScrollView):

def \_\_init\_\_(self, \*\*kwargs):

super(MainScroll, self).\_\_init\_\_(\*\*kwargs)

self.size\_hint = (1, None)

self.size = (Window.width, Window.height)

self.scroll\_timeout = 60

self.add\_widget(MainScreen())

class TimeYourselfApp(App):

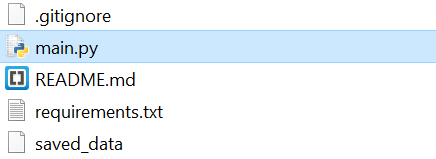
def build(self):

self.title = 'Stopwatch'

return MainScroll() #membuat object class yg inherit dari class mainscroll

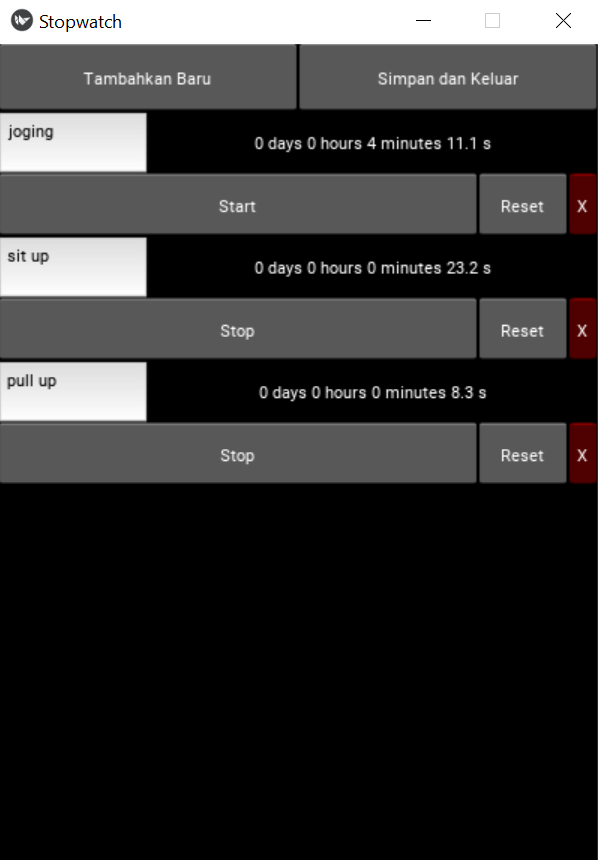
TimeYourselfApp().run()

1. **Aset yang digunakan pada ”Stopwatch”**



1. **Screen tampil pada kivy ”Stopwatch”**

Untuk project kivy kali ini kami membuat sebuah aplikasi stopwatch yang yang dapat mengitung waktu yang ingin kalian tahu saat melakukan sesuatu hal. Berikut ini contoh tampilannya.



1. **Tinjauan Pustaka**

Terinspirasi dari :

Link : - https://kivy.org/doc/stable/examples/gen\_\_canvas\_\_tesselate\_\_py.html

- <https://gist.github.com/Laspimon/740185c161c386ddaf2a>

