**PYGAME TUTORIAL (https://www.geeksforgeeks.org/pygame-tutorial/)**

**Python PyGame** library is used to create video games. This library includes several modules for playing sound, drawing graphics, handling mouse inputs, etc.

**\*\*\*ABOUT PYGAME**

1. Pygame is truly portable, it’s code can be run on a different operating system like Linux, Windows, MacOS.
2. Pygame is free and It is possible to create open-source, free and commercial games with it.
3. Pygame code is written in C language, which is 10-20 times faster than python code and Assembly Language

**SPRITE, SURF, and RECT:**

**Sprite:** Sprite is just a 2d object that we draw on the screen. We can use them by extending the sprite class.

**Surf:** Surfaces are like blank sheets of paper on which we draw. Our screen object is also a Surface. They can hold images as well.

**Rects:** Rectangular area that we define on a surface.

**Eg:-**

class Square(pygame.sprite.Sprite):

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

super(Square, self).\_\_init\_\_()

self.surf = pygame.Surface((25, 25))

self.surf.fill((0, 200, 255))

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

**Eg:-**

pygame.init()

screen = pygame.display.set\_mode((800, 600))

square1 = Square()

square2 = Square()

square3 = Square()

The above lines of code initialize pygame using the command pygame.init() which is necessary to use the pygame module commands.

After that, we define our screen object and its dimensions in pixels. Then in the next lines, we initialize our three squares.

**BLIT and FLIP –**

**Blit:** Blit keyword is used to draw a surface on another surface. In simple words when we draw a surface we just blit it onto some other surface.

**Flip:** It is used to update the entire screen after everything is drawn. Remember that the flip only works after drawing all the necessary surfaces otherwise,it will update nothing.

**Eg:-**

screen.blit(square1.surf, (40, 40))

screen.blit(square2.surf, (40, 530))

screen.blit(square3.surf, (730, 40))

screen.blit(square4.surf, (730, 530))

pygame.display.flip()

In the above lines of code first, we blit four squares on the screen, and then we flip to make them appear.

**\*\*\*pygame.init():** This command is used to initiate the pygame module.

**\*\*\*pygame.display.set\_mode((500,500)):** This command is used to make a window of desired size, (width, height).

**\*\*\*pygame.display.set\_caption(title = “”):** This command is used to set the title of the window/ board.

**\*\*\*pygame.event.get():** This is used to empty the event queue. If we do not call this, the window messages will start to pile up and,

the game will become unresponsive in the opinion of the operating system.

**\*\*\*pygame.QUIT:** This is used to terminate the event when we click on the close button at the corner of the window

**Eg:-**

import pygame

pygame.init()

**# CREATING CANVAS**

canvas = pygame.display.set\_mode((500, 500))

**# TITLE OF CANVAS**

pygame.display.set\_caption("My Board")

exit = False

while not exit:

for event in pygame.event.get():

if event.type == pygame.QUIT:

exit = True

pygame.display.update()

**\*\*\* IMPORTING IMAGE**

**Eg:-**

import pygame

pygame.init()

color = (255,255,255)

position = (0,0)

**# CREATING CANVAS**

canvas = pygame.display.set\_mode((500,500))

**# TITLE OF CANVAS**

pygame.display.set\_caption("Show Image")

**# IMPORTING IMAGE**

image = pygame.image.load("Screenshot.png")

exit = False

while not exit:

canvas.fill(color)

canvas.blit(image, dest = position)

for event in pygame.event.get():

if event.type == pygame.QUIT:

exit = True

pygame.display.update()

**\*\*\***In Pygame we use **rect()** method to draw rectangle boxes on the window. Pygame uses Rect objects to store and manipulate rectangular areas. It can be formed by the combination of left, top, width, and height values.

**Eg:-**

import pygame

pygame.init()

color = (255,255,255)

rect\_color = (255,0,0)

**# CREATING CANVAS**

canvas = pygame.display.set\_mode((500,500))

**# TITLE OF CANVAS**

pygame.display.set\_caption("Show Rectangle")

exit = False

while not exit:

canvas.fill(color)

for event in pygame.event.get():

if event.type == pygame.QUIT:

exit = True

pygame.draw.rect(canvas, rect\_color,

pygame.Rect(30,30,60,60))

pygame.display.update()

**\*\*\***Get size of formed screen by using **screen.get\_size()** method.

**Eg:-**

**# import package pygame**

import pygame

**# initialize pygame**

pygame.init()

**# Form screen**

screen = pygame.display.set\_mode((500, 500))

**# get the size**

x, y = screen.get\_size()

**# quit pygame**

pygame.display.quit()

**# view size (width x height)**

print(x, y)

**\*\*\*Resizable PyGame Window**

**Eg:-**

**# import package pygame**

import pygame

**# Form screen with 400x400 size and with resizable**

screen = pygame.display.set\_mode((400, 400),pygame.RESIZABLE)

**# set title**

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

**# run window**

running = True

while running:

for event in pygame.event.get():

if event.type == pygame.QUIT:

running = False

**# quit pygame after closing window**

pygame.quit()

**\*\*\*fill():** This method is used to fill the display with the color specified.

**Eg:-**

**# Importing the library**

import pygame

**# Initializing Pygame**

pygame.init()

**# Initializing surface**

surface = pygame.display.set\_mode((400, 300))

**# Initializing RGB Color**

color = (255, 0, 0)

**# Changing surface color**

surface.fill(color)

pygame.display.flip()

**\*\*\*To change the name of pygame window:**

**Syntax:** pygame.display.set\_caption('Title of window')

**\*\*\*To change the icon of pygame window:**

**Syntax:** pygame.display.set\_icon(Icon\_name)

**Eg:-**

**# import pygame module**

import Pygame

**# initializing imported module**

pygame.init()

**# Displaying a window of height 500 and width 400**

pygame.display.set\_mode((400, 500))

**# Here we set name or title of our pygame window**

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

**# Here we load the image we want to use**

Icon = pygame.image.load('gfglogo.png')

**# We use set\_icon to set new icon**

pygame.display.set\_icon(Icon)

**# Creating a bool value which checks if game is running**

running = True

**# Keep game running till running is true**

while running:

**# Check for event if user has pushed**  **any event in queue**

for event in pygame.event.get():

**# If event is of type quit then set running bool to false**

if event.type == pygame.QUIT:

running = False

**\*\*\*Putting our image on display surface**

Eg:-

**# Importing the library**

import Pygame

**# Initializing Pygame**

pygame.init()

**# creating the display surface**

display\_surface = pygame.display.set\_mode((500, 500 ))

**# Creating the image surface**

image = pygame.image.load('gfg\_logo.png')

**# putting our image surface on display surface**

display\_surface.blit(image,(100,100))

**# updating the display**

pygame.display.flip()

**\*\*\*pygame.time.wait**

This function is used to pause the running of the program for few seconds. it takes time in milliseconds as parameter.

**\*\*\*pygame.time.delay**

This function works the same as pygame.time.wait function the difference is that this function will use the processor (rather than sleeping) in order to make the delay more accurate.

**Eg:-**

**# importing pygame module**

import pygame

**# importing sys module**

import sys

**# initialising pygame**

pygame.init()

**# creating display**

display = pygame.display.set\_mode((500, 500))

**# Creating the image surface**

image = pygame.image.load('gfg\_logo.png')

**# putting our image surface on display surface**

display.blit(image,(100,100))

**# making the script wait for 5000 seconds**

pygame.time.wait(5000)/pygame.time.wait(5000)

**# creating a running loop**

while True:

**# creating a loop to check events that are occurring**

for event in pygame.event.get():

if event.type == pygame.QUIT:

pygame.quit()

sys.exit()

**# updating the display**

pygame.display.flip()

**\*\*\*pygame.time.get\_ticks**

This function gives the time which has been running in milliseconds.

**Eg:-**

**# importing pygame module**

import pygame

**# initialising pygame**

pygame.init()

**# creating a variable**

i=0

while i<5:

**# storing the time in ticks variable**

ticks=pygame.time.get\_ticks()

**# printing the variable ticks**

print(ticks)

**# increasing the value of i by 1**

i=i+1

**# pausing the script for 1 second**

pygame.time.wait(1000)

**\*\*\*pygame.time.Clock**

This function is used to create a clock object which can be used to keep track of time.