iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 9 **Issue:** VI **Month of publication:** June 2021

DOI: https://doi.org/10.22214/ijraset.2021.35735

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 9 Issue VI June 2021- Available at www.ijraset.com

Pygame: Develop Games using Python

Piyush N Shinde[1], Yash J Chavan[2], Shubham G Chilka[3], Gaurav N Patil[4], Manoj Raghunath Kharde[5]
[1, 2, 3, 4,5]*8th semester, Department of Computer Engineering, Pravara Rural Engineering College, Loni*

***Abstract:** In this paper, we will present the process of creating a simple game(Dracula) made in Python. For the purposes of this work, we created the game Dracula using the Pygame package. The aim of this paper is to show the process of creating a game, i.e the gradual construction of each element of the game. The game was created in several phases, and we will describe and explain each in detail. We will start from the construction of the basic skeleton of the game, that is. a graphical representation of the background image window. We will gradually introduce and add new elements (heroes, enemies, bullets, levels ...) all the way to a fully built and functional game. We will point out the possibilities and limitations that arise when programming the dracula game.*

***Keywords:** Python, Pygame, Dracula game, programming*

I. INTRODUCTION

Python programming language is a senior programming language for this purpose. It was developed in 1991 by the Dutchman Guido van Rossum, and was named after the cult British series "Monty Python's Flying Circus". In the Python community, Rossum is nicknamed BDFL, which stands for Benevolent Dictator for Life (benevolent lifelong dictator).

Python programming is an integrated language as it executes code line by line and makes it faster also it is object oriented and high level language with dynamic semantics. Its a high-level built language in data structures, which is combined with dynamic typing as well as dynamic binding, which makes it a very attractive language for Rapid Application Development, and also for use as a scripting language to connect components together. Python is not very complex, as well as easy to learn syntax improves readability and therefore it reduces the cost of program maintenance. Python supports various modules and packages, which helps in program modularity and code reuse. The Pygame package is a set of Python modules for building simpler games with a graphical interface. It is built on a set of SDL libraries (Simple DirectMedia Layer) 7 through which we access the multimedia hardware of the computer. Pygame is compatible with most operating systems and platforms (Linux, Windows, MacOS, ...). Parts of Pygame are written in the C programming language, which is several times faster than Python. Pygame is a highly portable module that runs on nearly every platform and operating system.

II. PYGAME MODULES

Modules are pieces of software, specific functionality, within which are defined definitions of related classes, functions or constants. File modules with the * .py extension, and the file name is also the module name. By installing Python, we also get a number of standard modules, such as random (module for generating pseudo-random numbers) or math (module for many mathematical function.

Some important pygame modules are:

Module Name	Purpose
pygame.display	control the display of windows or screens
pygame.draw	drawing simple shapes (rectangles, lines, polygons ...)
pygame.event	interaction with events (keyboard, mouse, joystick ...)
pygame.font	loading and rendering fonts
pygame.image	uploading and capturing images
pygame.key	uploading and capturing images
pygame.mixer	uploading and capturing images
pygame.mouse	uploading and capturing images
pygame.sprite	a set of classes for game objects (Sprite, Group ...), collision functions
pygame.time	monitoring, time control
pygame.transform	surface manipulation (resizing, rotating, rotating, filtering images)

Table 1: List of important pygame modules

III. EASE OF USE

Multi core CPUs can be used easily. With the common dual core CPUs as well as 8 core CPUs that are cheaply available on desktop systems, making use of multi core CPUs gives you an advantage to do more in your game.

Some selected functions in pygame release the dreaded python GIL, which you can do from C language.

Pygame uses the optimized C language and Assembly code for core functions. C language code is 10 to 20 times faster than python code because it's procedural, and assembly code is 100 times or more times faster than python code.

Comes with many operating systems. It Just requires apt-get, emerge and pkg_add, or yast install to get installed on your system. You don't need to mess with installing it outside of your operating system's package manager as it is some commands away . It comes with binary pos system installers and uninstallers for Windows and MacOSX. Pygame does not require any kind of setup tools with even the ctypes to install.

Truly portable. It supports Linux, 64-bit Windows, Windows (95, 98, ME, 2000, XP, Vista etc), Windows CE, BeOS, MacOS, Mac OS X, FreeBSD, OpenBSD, BSD/OS, Solaris, IRIX, and QNX. The code supports AmigaOS, Dreamcast, Atari, AIX, OSF/Tru64, RISC OS, SymbianOS and OS/2, but these are not officially supported by this module. You can also use it on the on hand devices, game consoles and One Laptop Per Child (OLPC) computer.

It's Simple and easy to use. Kids and adults make shooter games with pygame. Pygame module is used in the One Laptop Per Child (OLPC) computer project and has been taught in the essay courses to small kids and college students and also used by the people who first programmed in z80 assembler or c64 basic.

Many games have been published. Including Australian Game festival finalists, Indie Game Festival finalists, multimedia projects, popular shareware, and open source games. 660 plus projects have been published until now on the pygame websites such as: list needed. Many games have been released with the SDL (which pygame is based on), so you should be sure that it has been tested well by millions of users. You control your main loop. You have to call the pygame functions because they don't call your functions. This gives you better control over other libraries, and for different types of programs.

Pygame does not require a GUI to use all the functions. You can also use pygame from a command line if you want to use it to process images, get joystick input and play sounds.

Fast response to reported bugs. Some of the bugs are patched within an hour of being reported. Please have a look on the mailing list for BUG... you'll see for yourself. Sometimes it suck at bug fixes, but mostly it has pretty good bug fixers. Bug reports are pretty rare these days as a lot of them have been fixed already.

Small amount of code. It doesn't contain hundreds of thousands of lines of code for things that you won't use anyway. The core is always kept simple, and extra things that are like GUI libraries, and effects are developed separately outside of the pygame.

Modular. You can always use pieces of pygame separately. Do you want to use a different sound library? Then that's fine. Many core modules can be initialized and used separately in pygame.

IV. BASIC SYNTAX

The basic syntax of Pygame is:

```
import pygame
pygame.init()
screen = pygame.display.set_mode((400,500))
done = False
while not done:
for event in pygame.event.get():
if event.type == pygame.QUIT:
done = True
pygame.display.flip()
```

Now let's understand basic syntax of the above program line by line:

1) *import pygame:-* This provides us access to the pygame framework and imports all the functions of pygame.
2) *pygame. init()* - This is used to initialize the required module of the pygame.
3) *pygame.display.set_mode(width, height)* - This function is used to display a window of desired size. Return value is a surface object which is the object where we can perform graphical operations.
4) *pygame.event.get()*- This function is used to empty the event queue and if we do not call this, then the window messages will start to pile up and soon the game will become unresponsive in the opinion of operating system.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 9 Issue VI June 2021- Available at www.ijraset.com

5) *pygame.QUIT* – This instruction is used to terminate the event when we click on the close button at the corner of the window.
6) *pygame.display.flip()* – As we know that Pygame is double-buffered, this shifts the buffers. It is essential to call this function in order to make any kind of updates that you make on the game screen to make it visible.

V. CREATING a PYGAME WINDOW

For creating a pygame display window we require variables for width and height of the window and these variables are accessed by the pygame function Set Mode and then a window is created of specific resolution.

```
import pygame
pygame.init()
#RESOLUTION
display_width = (required width)
display_height = (required height)
win = pygame.display.set_mode((display_width, display_height))
while not done:
for event in pygame.event.get():
if event.type == pygame.QUIT:
done = True
pygame.display.flip()
```

By running the above code, we created a pygame window as shown in the figure below(fig.1.) where we can display our game content.

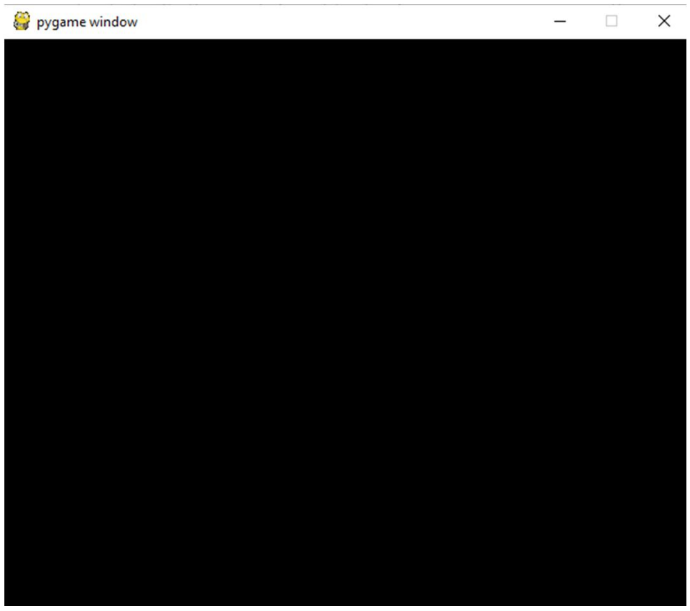


Figure 1: Pygame Window

VI. ADDING THE CHARACTERS

After Creating a display surface object using display.set_mode() method of pygame we will create an Image surface object i.e. the surface object in which the image is drawn on it using image.load() method in pygame. We can copy image surface object to the display surface object using the blit( ) method of the pygame display surface object.
We can show the display surface object on pygame window using the display.update( ) method of pygame.
The following code shows the making of characters using variety of images:

```
walkRight=[pygame.image.load('R1.png'),pygame.image.load('R2.png'),pygame.image.load('R3.png'),pygame.image.load('R4.png')
,pygame.image.load('R5.png'),pygame.image.load('R6.png'),pygame.image.load('R7.png'),pygame.image.load('R8.png'),
pygame.image.load('R9.png')]
walkLeft=[pygame.image.load('L1.png'),pygame.image.load('L2.png'),pygame.image.load('L3.png'),pygame.image.load('L4.png'),p
ygame.image.load('L5.png'),pygame.image.load('L6.png'),pygame.image.load('L7.png'),pygame.image.load('L8.png'),pygame.imag
e.load('L9.png')]
bg = pygame.image.load('bg.jpg')
character = pygame.image.load('standing.png')
```

VII. ADDING SOUND TO SPECIFIC AREAS

To play music or audio files in pygame, pygame.mixer( ) is used (pygame module used for loading and playing sounds). This module contain classes for loading the Sound objects and also controlling playback. There are four steps in order to do so:

Starting the mixer:-

```
mixer.init()
```

Loading the song:-

```
mixer.music.load(“ song.mp3”)
```

Setting the volume:-

```
mixer.music.set_volume(0.7)
```

Start playing the song:-

```
mixer.music.play()
```

For example, we have to add sound to the bullets. The following block of code shows how to add sound to the bullets:

```
bulletSound = pygame.mixer.Sound('shoot.wav')
hitSound = pygame.mixer.Sound('hit.wav')
music = pygame.mixer.music.load('music.ogg')
pygame.mixer.music.play(-1)
keys = pygame.key.get_pressed()
if keys[pygame.K_SPACE] and shootLoop == 0:
        bulletSound.play()
        if man.left:
            facing = -1
        else:
            facing = 1
        if len(bullets) < 5:
            bullets.append(projectile(round(man.x+man.width//2), round(man.y + man.height//2), 6, (0,0,0), facing))
        shootLoop = 1
```

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 9 Issue VI June 2021- Available at www.ijraset.com

VIII. WIN AND LOSE CONDITON

There is always a winning and losing condition in a game to decide the winner. The following code shows block of code representing win and lose situation:

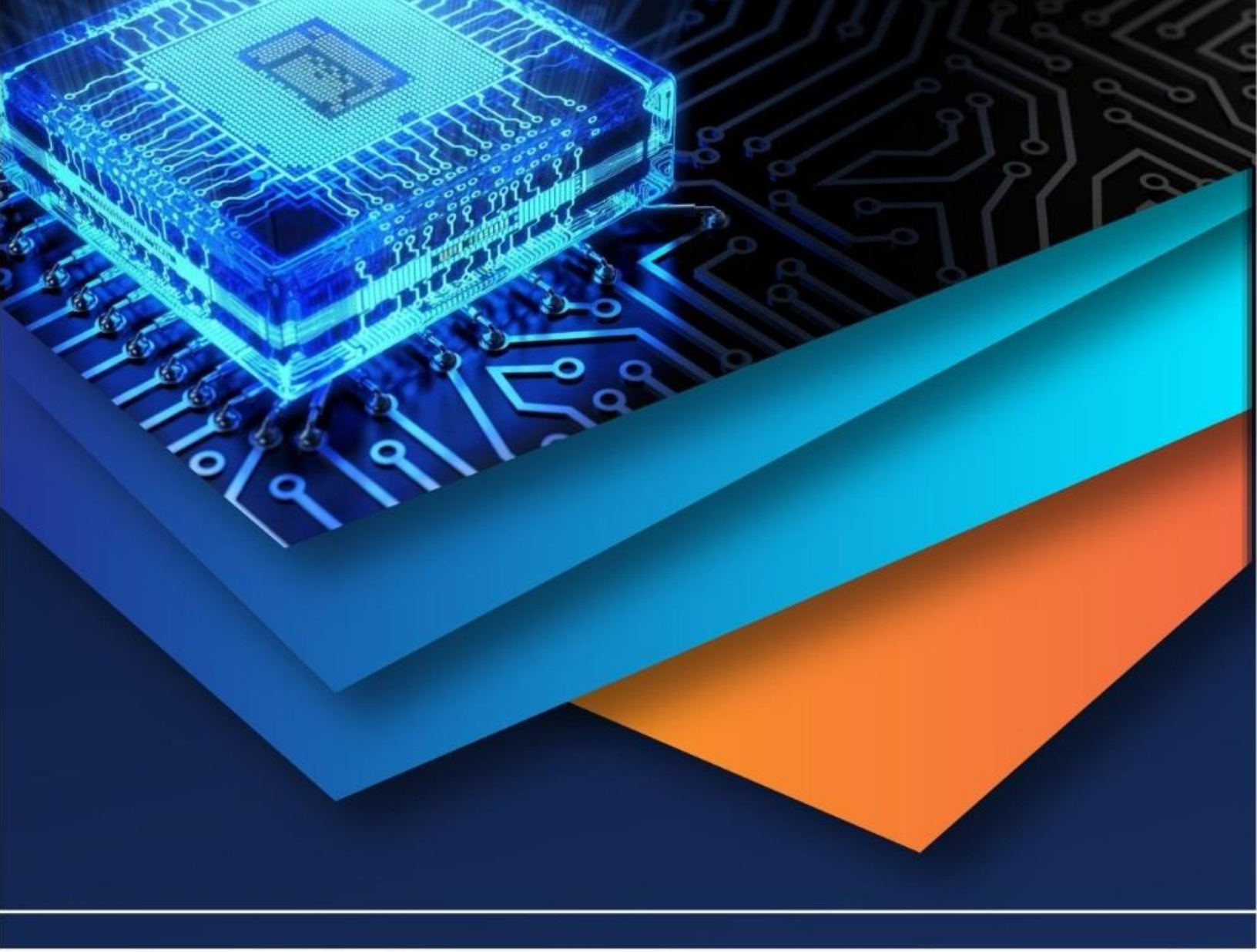
```
def gameOver():
    game_over = True
    while game_over:
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                pygame.quit()
                quit()

        win.fill(white)
        message_to_screen('Game Over', green, -100, size= 'medium')
        message_to_screen('You died', black, -30)
        button ('Play one more time', 50, 400, 120, 50, green, light_green, action='play again')
        button('controls', 200, 400, 100, 50, yellow, light_yellow, action='controls')
        button('quit', 350, 400, 100, 50, red, light_red, action='quit')
        pygame.display.update()
        clock.tick(15)


def You_win():
    winner = True
    while winner:
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                pygame.quit()
                quit()

        win.fill(white)
        message_to_screen('You Won', green, -100, size='large')
        message_to_screen('Congratulations, You win nothing',      black, -30)
button('Play one more time', 150, 500, 150, 50, green, light_green, action='play')
        button('controls', 350, 500, 100, 50, yellow, light_yellow, action='controls')
        button('quit', 550, 500, 100, 50, red, light_red, action='quit')
```

IX. CONCLUSION

This paper describes systematic development process of a 2D game(DRACULA) which is made in entirely made in python. The main purpose of this paper was to show the systematic construction of entire elements of the game. Dracula game was built in several parts, each and every part is displayed and explained in detail. The game seems simple at first glance but more details had to be created in order to make the game interesting and fun to play. So, this paper mentions all the above properties and characteristics. Steps taken in creating the game such as adding initial background image, adding player moves, movement adjustments for hero and enemies, game animation, insertion of the static and dynamic blinking test, also setting the skill levels, various sounds and three states that are game start, paused and game over.

REFERENCE

[1] [What is python? Executive summary].(n.d.).n/a Retrieved from https://www.python.org/doc/essays/blurb/
[2] [Pygame about].(n.d.).n/a Retrieved from https://www.pygame. org/wiki/about
[3] [Python pygame(game development library)].(n.d.).n/a Retrieved from https://www.javatpoint.com/pygame
[4] Zorrilla, G. (n.d.). Pygame 1.5.5 reference manual. Retrived from https://www.pygame.org/ftp/contrib/pygame_docs.pdf
[5] (Crljenko, J.2018).Making a simple arcade game in python.Retreived from https://core.ac.uk/reader/199709841
[6] [Python | displaying images with pygame].(n.d.).n/a Retrieved from https://www.geeksforgeeks.org/python-display-images-with-pygame/
[7] [Python | playing audio file in pygame].(n.d.).n/a Retrieved from https://www.geeksforgeeks.org/python-playing-audio-file-in-pygame/



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY