



Pygame Workshop

By Jonah Cook and Tim



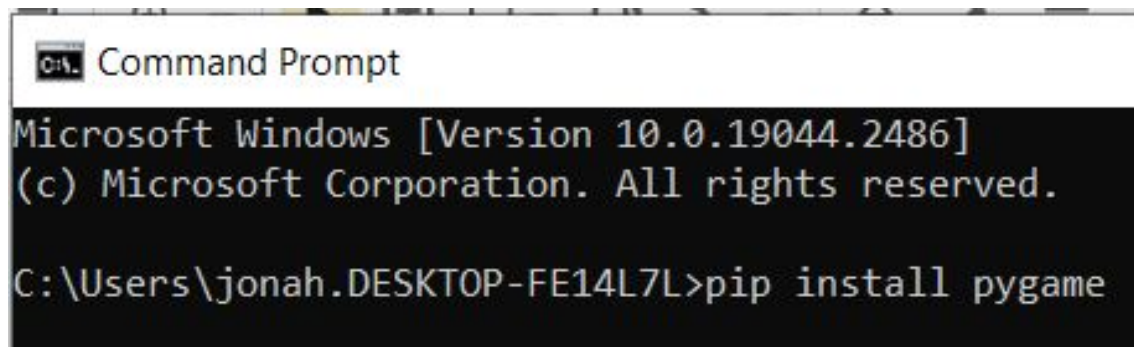
Things you will need:

- Basic knowledge of python as this is an intermediate level workshop
- Having python installed
- An IDE compatible with python, such as Visual Studio Code
- A charged laptop



Installing Pygame

To start, open up your Command Prompt (if you are on Windows), the terminal (for Mac), or the command line (for Linux). Run the command “pip install pygame” (this will require you to have python already installed)



```
Command Prompt
Microsoft Windows [Version 10.0.19044.2486]
(c) Microsoft Corporation. All rights reserved.

C:\Users\jonah.DESKTOP-FE14L7L>pip install pygame
```



Getting started

Put the following code into your IDE:

```
import pygame

pygame.init()
clock = pygame.time.Clock()
window = pygame.display.set_mode((600, 600))
pygame.display.set_caption('my pygame window')

while True:
    clock.tick(60)
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            pygame.quit()
```



Explanation

- “import pygame” just imports the pygame library into your file at runtime to use
- Next we initialize all pygame modules, like the clock
- We then create a clock object to cap the framerate of our game at a steady speed
- We then create our window, give it a set size, and caption it
- Then we create an infinite loop for our game, where every frame it loops through all the input received, and if someone clicked on an X the game closes



Drawing shapes

One way you can draw shapes is with the “pygame.draw.rect” function

This function has three arguments, “pygame.draw.rect(surface, rgb, rect object)”

For us, the surface would just be “window”

The rgb is the color of the rectangle, and its form is (r, g, b) where each number is a value from 0 to 255

The rect object is in the form of “pygame.Rect(left_x, top_y, right_x, bottom_y)”



Other shapes

You can draw a circle with `pygame.draw.circle()`

Its arguments are a bit different, being (surface, color, center, radius) where “center” is in the form of (x, y) and the radius is just the radius of the circle

You can also draw a line with `pygame.draw.line()`

Its arguments are (surface, color, start_pos, end_pos) where the starting and ending points are represented as (x, y)



Updating your game screen

Every frame, you need to call the “`pygame.display.update()`” method to update the game screen and display anything new

Without it, the game screen would never update

You can add it as the last line in your game loop so any new changes take effect immediately after handling input



Try drawing some shapes!

Try making some shapes on your own to learn how it works. You will have five minutes to do this. Some sample code is provided below:

```
pygame.quit()  
pygame.draw.rect(window, (0, 255, 255), pygame.Rect(100, 100, 300, 300))  
pygame.draw.circle(window, (255, 255, 255), (450, 350), 75)  
pygame.draw.line(window, (255, 0, 0), (100, 400), (400, 500))  
pygame.display.update()
```



Handling input

Say we wanted to listen for input from the user. We can do that by adding on to the code inside of “for event in pygame.event.get()” to listen for more types of input than just the screen closing.

```
if event.type == pygame.QUIT:
    pygame.quit()
elif event.type == pygame.KEYDOWN:
    if event.key == pygame.K_UP:
        # can also be K_DOWN, K_LEFT, and K_RIGHT
        # also works with keyboard keys, such as A-Z and SPACE
```



Adding images

You can add images to a pygame program by using the “os” package. Import the os package with “import os” and then add any images you want to use to a nested folder inside the folder your python file is currently in. Below, I am getting the image named “birb.jpg” from the “images” folder in my code folder, shrinking it down to 40x40, and converting it into a form usable by pygame.

```
birb = pygame.transform.scale(pygame.image.load(os.path.join('images', 'birb.jpg')), (40, 40)).convert_alpha()
```

Now it's your turn! Create a nested folder, add some image to it, and make it show up on your pygame window with the “blit” command shown below. Keep in mind that x and y define the upper left corner of where the image will go.

```
window.blit(birb, (x, y))
```



Challenge time!

We will spend the rest of this workshop creating something! For this challenge, you should try to make a two-player game, where both players are represented by some image or colored shape. One player should move using WASD, and the other should move with the arrow keys. The players should move on a grid, and move a set distance any time a movement key is pressed. Player 2 must evade player 1, as if they get too close, player 1 wins! Find documentation at [pygame.org/docs](https://www.pygame.org/docs)

Tips:

- Use variables to keep track of player positions, and use key listeners under “`pygame.event.get()`” to update positions if keys are pressed
- Every frame, check if the x and y positions of both players are very close to each other, and if they are, end the game
- Don’t be afraid to ask for help! This can be a daunting task, but breaking it down into easier, more manageable steps can help a lot