Kyle Forker

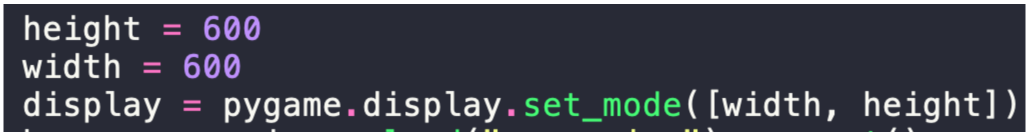
Dr. Wang

CS-489

February 13, 2022

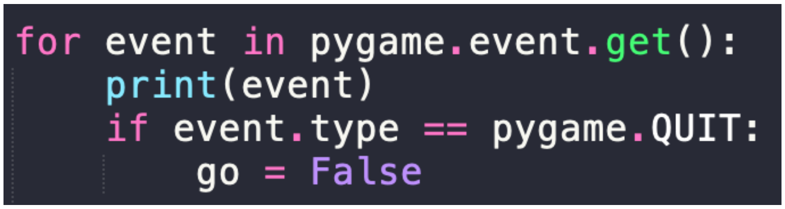
Term 1 Summary

Starting off Term 1 the choice of making the subject of the research on PyGame was chosen. PyGame is a library that can be installed and imported into a Python script. The library is for making games and can even be used for visualization of many different things. A test game has been created to learn about the features and different attributes of PyGame itself. The game in question is snake. This game has benefitted the research in helping the understanding and usage of different aspects of the library. There are so many things that can be done with PyGame. The first step was learning how to setup a display environment. Below is the simplest way to set up a window in which the developer starts off with a project:



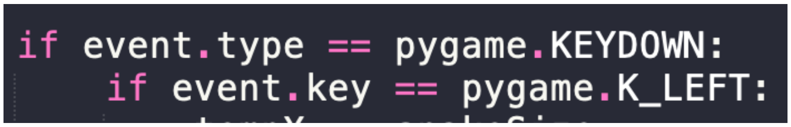
```python
height = 600
width = 600
display = pygame.display.set_mode([width, height])
```

The developer is able to choose any width and height they would want to choose. They then need to use a tuple with the values to set the display width and height. The next step is setting up the game itself. This is usually done with a while loop and then breaking out of it later to end the game. The simplest way to end the game is to:



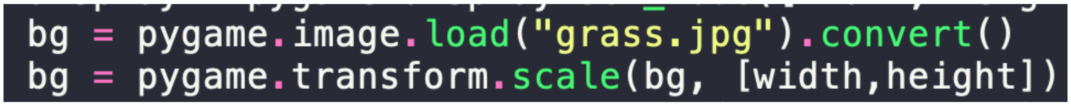
```python
for event in pygame.event.get():
    print(event)
    if event.type == pygame.QUIT:
        go = False
```

This loop takes the event of the game itself and checks to see if the "QUIT" function has been called. If so the developer needs to break out of the while loop to then end the game. The next feature learned was how to deal with keyboard inputs to alter aspects inside the game. This allows any user to move or interact with the game. The below code is the simplest way to check for keyboard inputs and use them in the game:



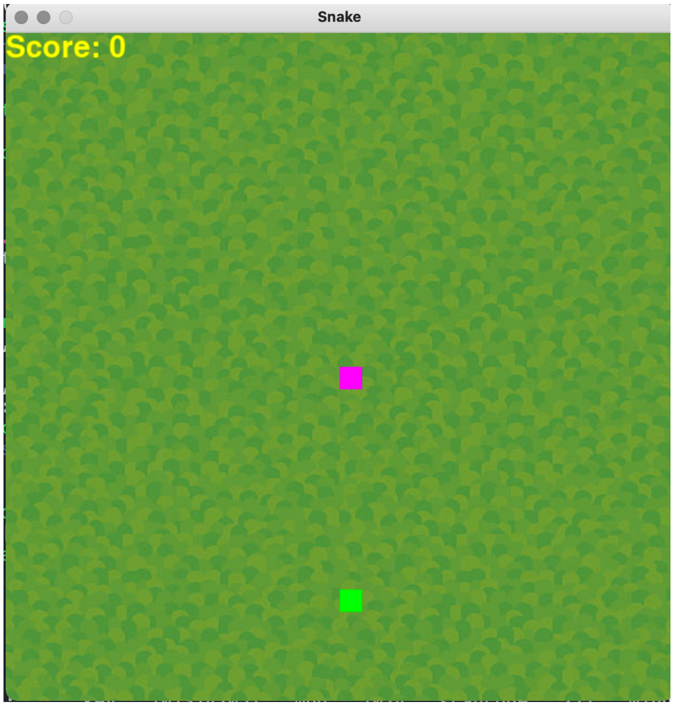
```
if event.type == pygame.KEYDOWN:
    if event.key == pygame.K_LEFT:
```

This checks for an event type. This then checks if the even is a keypress or user input event. This then checks what key the user has pressed and inside the final if statement the developer can create any action they would want inside the game. PyGame also makes it developer friendly to create assets for a game as well. Below is information about how to import and image:

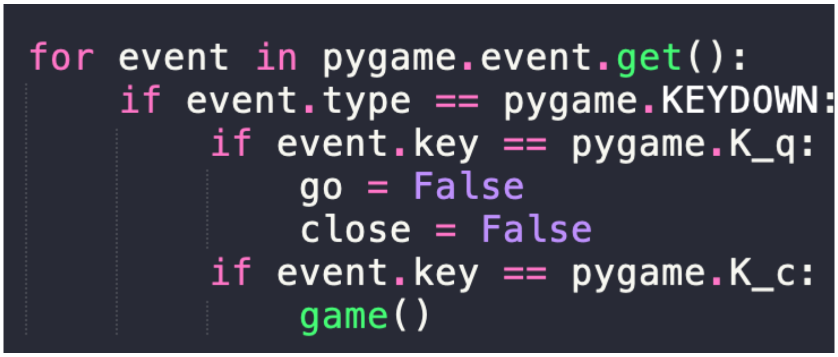


```
bg = pygame.image.load("grass.jpg").convert()
bg = pygame.transform.scale(bg, [width,height])
```

The first line of code loads an image and converts it to a sprite. The second line transforms the image and stretches it to the size of the game window. This allows for the creation of a background. Overall during the first term of this project there has been ample research done in the library to the point that a working game can be created. Below can be found screenshots of the sample game used to begin and understand research:



The game records score. The pink box is the snake in which expands when it goes over the green box known as food in the game. Finally the aspect of replay-ability was researched. This allows a user to play the game repeatedly without having to reopen it every time to play.



```
for event in pygame.event.get():
    if event.type == pygame.KEYDOWN:
        if event.key == pygame.K_q:
            go = False
            close = False
        if event.key == pygame.K_c:
            game()
```

At the end of the game this loop is used to check if a user presses one of the following keys: [q or c]. If the user presses "q" then the game closes entirely and if the user presses "c" then the loop calls the function the game is enclosed in to run it again.

This is the research and progress that has been made during term one of this project.