Pygame Summary

"Don’t let the snake hit you."

Achraf Groiez

Before

Import

1. Import pygame.

2. Import os(in some case you can need it. For example, remove the line in the terminal(os.system("cls"))

3. The color are in rgb like in CSS.

```
import pygame
import os
RED = (255, 0, 0)
```

Screen

Making the screen

1. First set the width and the height.

2. Make a variable for the screen and put the width and the height inside.

3. Make a while loop.

```python
WIDTH, HEIGHT = 900, 500
WIN = pygame.display.set_mode((WIDTH, HEIGHT))
run = True
while run:
    #Event handler
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            run = False
pygame.quit()
```

Background image/color

1. Import the image and set the background image with the window height and width.
2. Put the image in the screen.
3. Declare the color.
4. Put the color on the screen.

```
SPACE = pygame.transform.scale(
    pygame.image.load(
        os.path.join('Assets', 'space.png')), (WIDTH, HEIGHT)
)
         #Image (x, y)
WIN.blit(SPACE, (0, 0))
#The color section:
RED = (255, 0, 0)
WIN.fill(RED)
```

Give the screen a name(In the top left)

1. Put the text you want between the cotes.

```
pygame.display.set_caption("Youtube Game")
```

Change the default icon(In the top left)

1. Declare/import your icon.
2. Set the new icon.

```
new_icon = pygame.image.load("logo1.jpg")
pygame.display.set_icon(new_icon)
```

FPS

1. Declare the FPS.

2. Declare the clock(before the while).

3. In the while make this.

```
FPS = 60
clock = pygame.time.Clock()
clock.tick(FPS)
```

Infinite background

1. Make the screen.

2. Make a variable for the while loop.

3. Load the image.

4. Get de width of this image

5. Make a variable to change the position of the background image.

6. Make a variable to know how much picture you need.

7. Make a while loop.

8. The for is here to print three image with a different x because the index(i) is increasing by one and every time the loop is running the scroll change, that why the image is changing position(because of the scroll).

9. The scroll change to make the effect that the image is moving.

10. This if need to by here because without him the image is going to have a bug.

11. You need to update the screen.

```
screen = pygame.display.set_mode((1000, 500))
run = True
bg = pygame.image.load("bg.png").convert()
bg_width = bg.get_width()
scroll = 0
tiles = math.ceil(1000 / bg_width) + 1
while run:
    #The fps isn’t here but it is better to put it.
    for i in range(0, tiles):
        screen.blit(bg, (i * bg_width + scroll, 0))
    scroll -= 5
    if abs(scroll) > bg_width:
        scroll = 0
    #Quit event isn’t here check the Condition section subtitle=Leaving the game.
    pygame.display.update()
```

Element spawn

Text in pygame screen

1. Initialize the pygame font library.

2. Declare the variable, the first parameter is the font, the second is the size.

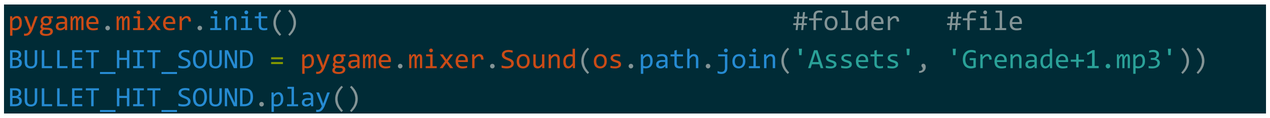
3. Declare a variable, you need to put the variable you create before with .render() and this .render() need parameter. The first parameter is the text you want to print on the screen(before the first coma), the number 1(I don't know), the last is the color.

4. The last step is to make WIN.blit() this is to put the element in the screen, this function takes 2 parameter first the element and in second the (x, y) that represent is position.

```python
pygame.font.init()
HEALTH_FONT = pygame.font.SysFont('commicsans', 40)
red_health_text = HEALTH_FONT.render("Health: "+str(red_health), 1, WHITE)
WIN.blit(red_health_text, (WIDTH - red_health_text.get_width() - 10, 10))
```

Sound effect

1. Initialize the pygame mixer library.
2. Declare/import the sound.
3. Put the name of the variable(step 2) and put the .play() and it will launch the sound.



```
pygame.mixer.init()                                             #folder    #file
BULLET_HIT_SOUND = pygame.mixer.Sound(os.path.join('Assets', 'Grenade+1.mp3'))
BULLET_HIT_SOUND.play()
```

CREATE RECTANGLE

1. Declare the rectangle.
2. Draw it in the window.

```python
#                      POSITION    | WIDTH, HEIGHT
BORDER = pygame.Rect(WIDTH/2 - 5, 0, 10, HEIGHT)
#                Window, Color, Rectangle of before
pygame.draw.rect(WIN, BLACK, BORDER)
```

MAKE EVENT

1. Create an event.
2. Post the event.
3. Make an if for the event.

```python
RED_HIT = pygame.USEREVENT + 2
pygame.event.post(pygame.event.Event(RED_HIT))
if event.type == YELLOW_HIT:
    yellow_health -= 1
    BULLET_HIT_SOUND.play()
```

Put an image in the screen

1. Import the image.
2. Declare the height and the width.
3. Make another variable where you rotate and scale the image(optional).
4. Put the image on the screen.

```python
YELLOW_SPACESHIP_IMAGE = pygame.image.load(
      os.path.join('Assets', 'spaceship_yellow.png')
)
SPACESHIP_WIDTH, SPACESHIP_HEIGHT = 50, 50
YELLOW_SPACESHIP =  pygame.transform.rotate(
    pygame.transform.scale(
        YELLOW_SPACESHIP_IMAGE, (SPACESHIP_WIDTH, SPACESHIP_HEIGHT)
    ), 90
)
WIN.blit(YELLOW_SPACESHIP, (10, 10))
```

Input

Keyboard

1. The keys_pressed must be a list and inside you put the keyboard key.
2. You need an if that contain a condition with key_pressed and the element.
3. Declare the key_pressed.

```
def yellow_handle_movement(keys_pressed, theObject):
    if keys_pressed[pygame.K_a] and theObject.x - VEL > 0: #LEFT
        theObject.x -= VEL

    if keys_pressed[pygame.K_d] and theObject.x + VEL + theObject.width <
BORDER.x : #RIGHT
        theObject.x += VEL

    if keys_pressed[pygame.K_s] and theObject.y + VEL + theObject.height < HEIGHT
- 15: #DOWN
        theObject.y += VEL

    if keys_pressed[pygame.K_w] and theObject.y - VEL > 0: #UP
        theObject.y -= VEL
keys_pressed = pygame.key.get_pressed()
yellow_handle_movement(keys_pressed, yellow)
#You can do it this way too:
if event.type == pygame.KEYDOWN: #Si l'événement est KEYDOWN
                if event.key  == pygame.K_LCTRL and len(yellow_bullets) <
MAX_BULLET:
                    bullet = pygame.Rect(yellow.x + yellow.width, yellow.y +
yellow.height//2 - 2, 10, 5)
                    yellow_bullets.append(bullet)
                    BULLET_FIRE_SOUND.play()

                if event.key  == pygame.K_RCTRL and len(red_bullets) <
MAX_BULLET:
                    bullet = pygame.Rect(red.x, red.y + red.height//2 - 2, 10, 5)
                    red_bullets.append(bullet)
                    BULLET_FIRE_SOUND.play()
```

Condition

Leaving the game

1. Declare a boolean variable.
2. Make a while loop.
3. Make a for of pygame.event.get().
4. Make the if.
5. Set the boolean variable that you set before at false.
6. Put pygame.quit(), to leave.

```
run = True
while run:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            run = False
            pygame.quit()
```

Essential

Button

1. The __init__ is the constructeur to make instance of the button(self.clicked is not included)

2. The action is to tell if the button was clicked and this value is return to know if we do the action or not.

3. The pos is to get position.

4. The if(self.rect.collidepoint(pos)) is to know if the mouse is on the button like a hover in html/css.

5. The if(pygame.mouse.get_pressed()[0] == 1 and self.clicked == False) is to know if the is clicked.

6. The if just after the 5 is for limiting the number of the button send information.

7. It is for button the button on the screen.

8. We return the action value at 2.

9. It is for the event we do in the place the buttons are called.

```python
#button class
class Button():
    def __init__(self, x, y, image, scale) -> None:
        widthImage = image.get_width()
        heigthImage = image.get_height()
        #To scale the image I think it is width, height
        self.image = pygame.transform.scale(image, (int(widthImage * scale), 
int(heigthImage * scale)))
        self.rect = self.image.get_rect()
        self.rect.topleft = (x, y)

        #This is to limit the number of click
        self.clicked = False
    def draw(self, WIN):
        action = False
        #get mouse position
        pos = pygame.mouse.get_pos()

        #check mouseover and clicked conditions
        if self.rect.collidepoint(pos):
            #This is for limit the click
            #0 = left mouse button, 1 = middle mouse button, 3 = right mouse
button
            if pygame.mouse.get_pressed()[0] == 1 and self.clicked == False:
                self.clicked = True
                action = True

        #This is for limit the click
        if pygame.mouse.get_pressed()[0] == 0:
            self.clicked = False

        WIN.blit(self.image, (self.rect.x, self.rect.y))

        return action

    #Where we call the button we need this.
    if start_button.draw(WIN):
        print('start')

    if exit_button.draw(WIN):
        run = False
```

End

Update

1. Update the screen after you had something in it.

```
pygame.display.update()
```

if __name__ == “__main__”:

1. Make this, I think, like that if you import the file it is not going to launch.

```
if __name__ == "__main__":
    main()
```

Bibliography

Youtube links

1. Coding With Russ, *YouTube*, “How To Change Default Window Icon In Pygame #shorts” April 19, 2023. https://www.youtube.com/shorts/sCoQ9kXPi98

2. Tech With Tim, *YouTube*, “Pygame in 90 Minutes - For Beginners”, January 8, 2021. https://www.youtube.com/watch?v=jO6qQDNa2UY

3. Coding With Russ, *YouTube*, “PyGame Beginner Tutorial in Python - Infinite Scrolling Background”, March 31, 2022. https://www.youtube.com/watch?v=ARt6DLP38-Y

4. Coding With Russ, *YouTube*, “PyGame Beginner Tutorial in Python - Adding Buttons”, Mai 26, 2021. https://www.youtube.com/watch?v=G8MYGDf_9ho