Pygame Project

Display

- Basic Setup

```python
import pygame
pygame.init()
screen = pygame.display.set_mode((800, 600))
pygame.display.set_caption("Pygame Window")
running = True
while running:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False
    screen.fill((255, 255, 255))
    pygame.display.flip()
pygame.quit()
```

- Setting Display

```python
import pygame
pygame.init()
screen = pygame.display.set_mode((1024, 768))
pygame.display.set_caption("New Pygame Window")
```

- Background Color

```python
import pygame
pygame.init()
screen = pygame.display.set_mode((800, 600))
screen.fill((0, 128, 128))  # Fill the screen with a teal color
pygame.display.flip()
```

- Geometry

```
import pygame
pygame.init()
screen = pygame.display.set_mode((800, 600))
rect = pygame.Rect(50, 50, 200, 100)  # Create a rectangle
pygame.draw.rect(screen, (255, 0, 0), rect)  # Draw the rectangle in red
pygame.display.flip()
```

  Shap

Image

- Import File

```
import pygame
pygame.init()
screen = pygame.display.set_mode((800, 600))
image = pygame.image.load('image.png').convert_alpha()
```

Why use covert Convert Alpha

- Setting Image

```
import pygame
pygame.init()
screen = pygame.display.set_mode((800, 600))
image = pygame.image.load('image.png').convert_alpha()
image = pygame.transform.scale(image, (200, 200))  # Resize the image
```

Tranfrom Method

- Display Image

```
import pygame
pygame.init()
screen = pygame.display.set_mode((800, 600))
image = pygame.image.load('image.png').convert_alpha()
screen.blit(image, (100, 100))  # Display the image at coordinates (100, 100)
pygame.display.flip()
```

File

Read File Write File

- Import File

```python
import json
with open('data.json') as f:
    data = json.load(f)
```

- Use JSON File

```python
import json
with open('data.json') as f:
    data = json.load(f)
print(data)
```

- Access JSON Data

```python
import json
with open('data.json') as f:
    data = json.load(f)
# {
# "x": 1,
# "data": {
#   "y": 2,
#   "z": 3
# }
# }
x_value = data['x']
y_value = data['data']['y']
z_value = data['data']['z']
print(f"x: {x_value}, y: {y_value}, z: {z_value}")
```

- Rewrite JSON File

```python
import json
with open('data.json') as f:
    data = json.load(f)
data['new_key'] = 'new_value'
```

```
with open('data.json', 'w') as f:
    json.dump(data, f)
```

Input

- Keyboard

```
import pygame
pygame.init()
screen = pygame.display.set_mode((800, 600))
running = True
while running:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False
        elif event.type == pygame.KEYDOWN:
            if event.key == pygame.K_LEFT:
                print("Left arrow key pressed")
            elif event.key == pygame.K_RIGHT:
                print("Right arrow key pressed")
            elif event.key == pygame.K_UP:
                print("Up arrow key pressed")
            elif event.key == pygame.K_DOWN:
                print("Down arrow key pressed")
        elif event.type == pygame.KEYUP:
            if event.key == pygame.K_LEFT:
                print("Left arrow key released")
            elif event.key == pygame.K_RIGHT:
                print("Right arrow key released")
            elif event.key == pygame.K_UP:
                print("Up arrow key released")
            elif event.key == pygame.K_DOWN:
                print("Down arrow key released")
    keys = pygame.key.get_pressed()
    if keys[pygame.K_LEFT]:
        print("Left arrow key is being held down")
    if keys[pygame.K_RIGHT]:
        print("Right arrow key is being held down")
    if keys[pygame.K_UP]:
        print("Up arrow key is being held down")
    if keys[pygame.K_DOWN]:
        print("Down arrow key is being held down")
pygame.quit()
```

- Mouse

```
import pygame
pygame.init()
screen = pygame.display.set_mode((800, 600))
```

```
running = True
while running:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False
        elif event.type == pygame.MOUSEBUTTONDOWN:
            if event.button == 1:  # Left mouse button
                print("Left mouse button clicked at", event.pos)
            elif event.button == 3:  # Right mouse button
                print("Right mouse button clicked at", event.pos)
        elif event.type == pygame.MOUSEBUTTONUP:
            if event.button == 1:  # Left mouse button
                print("Left mouse button released at", event.pos)
            elif event.button == 3:  # Right mouse button
                print("Right mouse button released at", event.pos)
        elif event.type == pygame.MOUSEMOTION:
            print("Mouse moved to", event.pos)
    mouse = pygame.mouse.get_pressed()
    if mouse[0]:
        print("Left mouse button is being held down")
    if mouse[2]:
        print("Right mouse button is being held down")
pygame.quit()
```

- Joystick

```
import pygame
pygame.init()
screen = pygame.display.set_mode((800, 600))
running = True
while running:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False
        elif event.type == pygame.JOYBUTTONDOWN:
            print("Joystick button pressed")
        elif event.type == pygame.JOYBUTTONUP:
                    print("Joystick button released")
                elif event.type == pygame.JOYAXISMOTION:
                    print("Joystick axis moved")
pygame.quit()
```

- Text Input

```
import pygame
pygame.init()
screen = pygame.display.set_mode((800, 600))
running = True
while running:
        for event in pygame.event.get():
                if event.type == pygame.QUIT:
```

```
                running = False
            elif event.type == pygame.TEXTINPUT:
                print("Text input:", event.text)
pygame.quit()
```

Sprite

Load Sprite Sheet Animate Sprite Sprite and Group

- Sprite Properties

```
import pygame

class Square(pygame.sprite.Sprite):
        def __init__(self, color, width, height):
                super().__init__()
                self.image = pygame.Surface([width, height])
                self.image.fill(color)
                self.rect = self.image.get_rect()
                self.health = 100

        def update(self):
                if self.health <= 0:
                        self.kill()

        def attack(self, damage):
                self.health -= damage
```

- Group of Sprite

```
import pygame

class Square(pygame.sprite.Sprite):
        def __init__(self, color, width, height):
                super().__init__()
                self.image = pygame.Surface([width, height])
                self.image.fill(color)
                self.rect = self.image.get_rect()
                self.health = 100

        def update(self):
                if self.health <= 0:
                        self.kill()

        def attack(self, damage):
                self.health -= damage

pygame.init()
```

```
screen = pygame.display.set_mode((800, 600))
all_sprites = pygame.sprite.Group()
square = Square((255, 0, 0), 50, 50)
all_sprites.add(square)

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

        all_sprites.update()

        screen.fill((255, 255, 255))
        all_sprites.draw(screen)
        pygame.display.flip()

pygame.quit()
```

Time

Framerate

- Start Stop End

```
import pygame
pygame.init()

screen = pygame.display.set_mode((800, 600))
pygame.display.set_caption("Pygame Window")

clock = pygame.time.Clock()
running = True

while running:
        for event in pygame.event.get():
                if event.type == pygame.QUIT:
                        running = False

        screen.fill((255, 255, 255))
        pygame.display.flip()

        clock.tick(60)  # Limit the frame rate to 60 FPS

pygame.quit()
```

5 game in Python