Pygame

Dr. Dongchul Kim

What is Pygame?

- Pygame is an open-source library for making video games.
- It provides modules for graphics, sound, and game control.
- It is designed to be used with Python, making it accessible to beginners and versatile for experts.

Key Features

- **Simplicity**: Pygame is easy to start with and has a straightforward set of functions to handle game elements.
- **Flexibility**: It supports various game projects, from simple 2D games to more complex graphical projects.
- **Community**: Pygame has a large and supportive community, offering extensive documentation and a wide range of tutorials and examples.

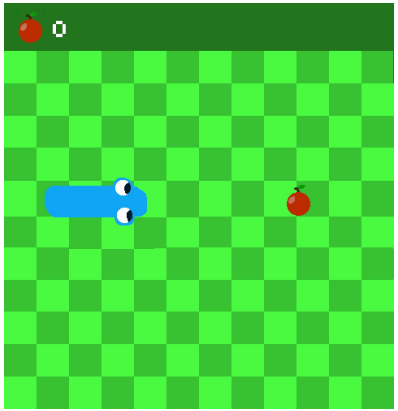
Why Use Pygame for Game Development?

- Ideal for learning basic game development concepts.
- Great tool for prototyping game ideas quickly.
- Allows integration with other Python libraries and tools to enhance game functionality.

Snake Game

The Snake game is one of the oldest and most popular arcade games.

The player controls a long, thin creature, resembling a snake, which moves around the screen, picking up food, or "apples," as it avoids hitting its own tail and the walls.



Snake Game

- Game Objectives
  - Primary Objective: To eat as many apples as possible. Each apple eaten makes the snake longer.
  - Secondary Objective: To avoid colliding with the walls or the snake's own growing body.
- Game Controls
  - Arrow Keys: Up, Down, Left, Right to direct the snake around the game area.

Setting Up Your Development Environment

- Install pygame

```
pip install pygame
```

- Testing the installation

```
import pygame

pygame.init()

print(pygame.ver)
```

```
pygame 2.5.2 (SDL 2.28.2, Python 3.8.10)
Hello from the pygame community. https://www.pygame.org/contribute.html
2.5.2
```

Configuring the Game Environment

Initializing Pygame: Begin by initializing Pygame to set up the necessary resources for game development.

```
import pygame

pygame.init()
```

Creating a game window and setting colors

```
screen_width = 600
screen_height = 400
game_screen = pygame.display.set_mode((screen_width, screen_height))
pygame.display.set_caption('Snake Game')


black = (0, 0, 0)
white = (255, 255, 255)
green = (0, 255, 0)
red = (255, 0, 0)
```

Game Loop!

```
running = True

while running:

    for event in pygame.event.get():

        if event.type == pygame.QUIT: # the close button

            running = False

    # Game logic, drawing code, and screen update will go here

    pygame.display.update()
```

FPS Control

```
clock = pygame.time.Clock()

fps = 15  # frames per second

direction = 'RIGHT' # snake direction

score = 0

running = True

while running:

    for event in pygame.event.get():

        if event.type == pygame.QUIT: # the close button

            running = False

    # Game logic, drawing code, and screen update will go here

    pygame.display.update()

    clock.tick(fps)
```

Let’s create a snake and apple!

```
# Snake Object
snake_segments = []
snake_size = 10  # Size of each snake segment
snake_length = 5  # Initial length of the snake
for i in range(snake_length):
    x = 250 - (snake_size * i)
    y = 200
    segment = pygame.Rect(x, y, snake_size, snake_size)
    snake_segments.append(segment)
```

Apple

```
import random


apple_size = 10
apple_position = (
    random.randrange(0, screen_width // apple_size) * apple_size,
    random.randrange(0, screen_height // apple_size) * apple_size
)
apple = pygame.Rect(apple_position[0], apple_position[1], apple_size, apple_size)
```

Drawing the snake and apple

```
while running:
    for event in pygame.event.get():
        if event.type == pygame.QUIT: # the close button
            running = False
    game_screen.fill(black)  # Clear screen with black background
    for segment in snake_segments:
        pygame.draw.rect(game_screen, green, segment)  # Draw snake segments
    pygame.draw.rect(game_screen, red, apple)  # Draw the apple
    pygame.display.update()
    clock.tick(fps)
```

**Controlling the snake

```
for event in pygame.event.get():
    if event.type == pygame.QUIT:
        running = False
    elif event.type == pygame.KEYDOWN:
        if event.key == pygame.K_UP and direction != 'DOWN':
            direction = 'UP'
        elif event.key == pygame.K_DOWN and direction != 'UP':
            direction = 'DOWN'
        elif event.key == pygame.K_LEFT and direction != 'RIGHT':
            direction = 'LEFT'
        elif event.key == pygame.K_RIGHT and direction != 'LEFT':
            direction = 'RIGHT'
```

Updating Snake’s Position

```
x, y = snake_segments[0].topleft

if direction == 'UP':

    y -= snake_size

elif direction == 'DOWN':

    y += snake_size

elif direction == 'LEFT':

    x -= snake_size

elif direction == 'RIGHT':

    x += snake_size

new_head = pygame.Rect(x, y, snake_size, snake_size)

snake_segments.insert(0, new_head)  # Add new head to the snake
```

```
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_UP:  # and direction != 'DOWN':
                direction = 'UP'
            elif event.key == pygame.K_DOWN:
                direction = 'DOWN'
            elif event.key == pygame.K_LEFT:
                direction = 'LEFT'
            elif event.key == pygame.K_RIGHT:
                direction = 'RIGHT'
    # Move the snake
    x, y = snake_segments[0].topleft # pygame.Rect
    if direction == 'UP':
        y -= snake_size
    elif direction == 'DOWN':
        y += snake_size
    elif direction == 'LEFT':
        x -= snake_size
    elif direction == 'RIGHT':
        x += snake_size
    new_head = pygame.Rect(x, y, snake_size, snake_size)
    snake_segments.insert(0, new_head)
```

Check for collisions

```
# Check for collisions
if (snake_segments[0].left < 0  or snake_segments[0].right > screen_width  or
    snake_segments[0].top < 0  or snake_segments[0].bottom > screen_height  or
    snake_segments[0] in snake_segments[1:] ):
    running = False  # Game over
```

When the snake eats an apple

```
# Check if snake eats apple
if snake_segments[0].colliderect(apple):
    score += 10
    apple_position = (random.randrange(0, screen_width // apple_size) * apple_size,
                      random.randrange(0, screen_height // apple_size) * apple_size)
    apple = pygame.Rect(apple_position[0], apple_position[1], apple_size, apple_size)
else:
    snake_segments.pop()  # Remove the last segment
```

Display Score

```
font = pygame.font.Font(file_path=None, 36)

score_text = font.render('Score: ' + str(score), True, white) # smoothing

game_screen.blit(score_text, (10, 10))
```

Game Over

```
# Game over screen

game_screen.fill(black)

game_over_text = font.render('Game Over', True, red)

game_screen.blit(game_over_text, (screen_width//2 - game_over_text.get_width()//2, screen_height//2))

pygame.display.update()

pygame.time.wait(2000)  # Wait two seconds before closing

pygame.quit()
```