Catch Me If You Can! - A Pygame Product

Developed by: Ricky Koppula (21112021) & Teena Davis (21112029)

Pygame is a versatile and popular Python library that empowers game developers to create 2D games, simulations, and interactive applications. Leveraging the power of the Simple DirectMedia Layer (SDL), Pygame provides a user-friendly, cross-platform framework for managing game graphics, sound, and input devices, making it accessible for those looking to dive into game development. With a robust community and extensive documentation, Pygame facilitates the creation of games limited only by one's imagination.

In this game, 'Catch Me If You Can!', players control a character on the screen, navigating it through a field filled with ghosts. The character's goal is to "capture" the other ghost while avoiding the edges of the screen, losing lives for each collision. The game typically features a score system to track the player's success, and in some variants, it may incorporate high scores to motivate players to compete with their best

In [ ]:
```python
# Importing the necessary libraries
import pygame
import random
```

In [ ]:
```python
# Initiating the Pygame environment
pygame.init()

# Set the constraints
WIDTH, HEIGHT = 800, 600
LIFE_COLOR = (0, 255, 0)
MAX_LIVES = 5
HIGH_SCORES_FILE = "highscore.txt"

# Set up the screen
screen = pygame.display.set_mode((WIDTH, HEIGHT))
pygame.display.set_caption("Catch Me If You Can!")
clock = pygame.time.Clock()
```

In [ ]:
```python
# Load background music
pygame.mixer.init()
pygame.mixer.music.load('background_music.mp3')

# Play background music
pygame.mixer.music.play(-1)  # -1 means the music will loop indefinitely

# Load the images for the strobe effect (background)
image_files = ['t1.jpeg']
background_images = [pygame.image.load(filename) for filename in image_fi

# Resize the background images
for i, image in enumerate(background_images):
    background_images[i] = pygame.transform.scale(image, (WIDTH, HEIGHT))

# Load the excorist and ghost images
player_image = pygame.image.load('detective.png')
player_image = pygame.transform.scale(player_image, (80, 80))
ghost_image = pygame.image.load('ghost.png')
ghost_image = pygame.transform.scale(ghost_image, (100, 100))
```

In [ ]:
```python
class Player:
    def __init__(self):
        self.image = player_image
        self.rect = self.image.get_rect()
        self.rect.center = (WIDTH // 2, HEIGHT // 2)
        self.lives = MAX_LIVES
        self.score = 0

    def move(self):
      # defines the direction and speed of the excorist
        keys = pygame.key.get_pressed()
        if keys[pygame.K_LEFT]:
            self.rect.x -= 15
        if keys[pygame.K_RIGHT]:
            self.rect.x += 15
        if keys[pygame.K_UP]:
            self.rect.y -= 15
        if keys[pygame.K_DOWN]:
            self.rect.y += 15

        # Ensure the player stays within the screen
        self.rect.x = max(0, min(WIDTH - self.rect.width, self.rect.x))
        self.rect.y = max(0, min(HEIGHT - self.rect.height, self.rect.y))

    def draw(self):
        screen.blit(self.image, self.rect)
```

In [ ]:
```python
ghost = []
# define function to display ghost appearance on the screen with random l
def create_ghost1():
    x = random.randint(0, WIDTH - ghost_image.get_width())
    y = random.randint(0, HEIGHT - ghost_image.get_height())
    ghost.append(pygame.Rect(x, y, ghost_image.get_width(), ghost_image.g

create_ghost1()
```

In [ ]:
```python
running = True
player = Player()
```

```python
background_image_index = 0
background_change_rate = 30  # Adjust this value to control the backgroun

# Font for scoreboard and high scores
font = pygame.font.Font(None, 36)
high_scores_font = pygame.font.Font(None, 36)

# Load high scores from a file or create an empty list
high_scores = []
try:
    with open(HIGH_SCORES_FILE, 'r') as file:
        high_scores = [int(line.strip()) for line in file.readlines()]
except FileNotFoundError:
    pass

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

    player.move()
    screen.fill((124, 124, 124))

    # Display the background strobe effect
    screen.blit(background_images[background_image_index], (0, 0))

    # Check for collisions with small balls
    for ghost1 in ghost[:]:
        if player.rect.colliderect(ghost1):
            ghost.remove(ghost1)
            create_ghost1()
            player.score += 1

    player.draw()

    for ghost1 in ghost:
        screen.blit(ghost_image, ghost1)

    # Check for collisions with the edges
    if (
        player.rect.left <= 0
        or player.rect.right >= WIDTH
        or player.rect.top <= 0
        or player.rect.bottom >= HEIGHT
    ):
        player.lives -= 1
        if player.lives == 0:
            # Update high scores
            if not high_scores or player.score > min(high_scores):
                high_scores.append(player.score)
                high_scores.sort(reverse=True)
                high_scores = high_scores[:1]  # Keep the top 5 high scor
                with open(HIGH_SCORES_FILE, 'w') as file:
                    for score in high_scores:
                        file.write(str(score) + '\n')
            running = False

    pygame.display.update()
    clock.tick(background_change_rate)  # Adjust the background change sp
    background_image_index = (background_image_index + 1) % len(backgroun

    # Display the scoreboard and high scores
    score_text = font.render(f'Score: {player.score}', True, LIFE_COLOR)
    screen.blit(score_text, (10, 40))
    high_scores_text = high_scores_font.render("High Scores:", True, LIFE
    screen.blit(high_scores_text, (10, 70))
    for i, score in enumerate(high_scores):
        score_text = high_scores_font.render(f"{score}", True, LIFE_COLOR
        screen.blit(score_text, (160, 70))
    pygame.display.update()
    clock.tick(80)

# Game over screen
font = pygame.font.Font(None, 36)
game_over_text = font.render("Game Over", True, LIFE_COLOR)
screen.blit(game_over_text, (WIDTH // 2 - 100, HEIGHT // 2 - 18))
pygame.display.update()

# Stop the background music
pygame.mixer.music.stop()

pygame.time.delay(2000)

pygame.quit()
```