How to add a scrolling camera to PyGame

Learning how to model a camera that rotates or scrolls while gaming will add realism and make the game more interesting. Here's how to add a scrolling camera to PyGame .

Scrolling is one of the common features of scrolling in the game world. This element adds depth and realism to the game and, at the same time, improves the overall gameplay experience.

You have different ways to implement a scrolling camera in PyGame, so make sure you understand the difference between them.

Create a simple game

Before starting, install pip on the device and use the command below to install the PyGame module:

```
pip install pygame
```

Now you can create a simple game with a rectangle for the player and two static platforms. The player can move left and right with the arrow keys.

Start by importing the pygame module. Then re-initialize it and create a game window using the **pygame.display.set_mode()** function . Then set the window caption and create a clock object to manage the frame rate.

```
import pygame pygame.init() WINDOW_WIDTH = 800 WINDOW_HEIGHT = 600 screen = pygar
```

Next, set up players and static platforms. Determine the size of the player and its initial position.

```
PLAYER_WIDTH = 50 PLAYER_HEIGHT = 50 player_x = WINDOW_WIDTH // 2 - PLAYER_WIDTH
```

Then create a game loop that handles events and updates the screen. In this loop, check for events such as exiting the game or moving the player with the corresponding arrow keys.

```
while True: ????# X? lý s? ki?n ????for event in pygame.event.get():
????????if event.type == pygame.QUIT: ???????????pygame.quit()
???????????quit() ????# V? hình n?n ????screen.fill(BACKGROUND_COLOR) ????
# V? hình ch? nh?t t?nh ????
pygame.draw.rect(screen, RECTANGLE_COLOR_1, rectangle_1) ????
pygame.draw.rect(screen, RECTANGLE_COLOR_2, rectangle_2) ????# V? ng??i ch?
i ????player_rect = pygame.Rect(player_x, player_y, PLAYER_WIDTH, ????????
PLAYER_HEIGHT) ????pygame.draw.rect(screen, (0, 0, 0), player_rect) ????
# Update màn hình ????pygame.display.update() ????# Gi?i h?n t? l?
```

```
khung hình ????clock.tick(30)
```

Camera settings

Now that you have a simple game with rectangular players and two static platforms, you can start working on the camera. In PyGame, the camera is basically just an offset that works for any object you draw at the screen. This means that if you move the camera to the left, everything on the screen will appear to move to the right.

Going to the camera setup step, you first need to define a variable to hold the **x** offset of the camera. Call this variable **camera_offset_x** and initialize it to 0.

```
# ??t ?? chênh l?ch camera camera_offset_x = 0
```

Next, update the position of all the objects that you draw on the screen, to account for the camera offset. You can do this by adding the value **camera_offset_x to the X** position of each object. For example, you can update a player's location like this:

```
# V? ng??i ch?
i player_rect = pygame.Rect(player_x + camera_offset_x, player_y, PLAYER_WIDTH,
????PLAYER_HEIGHT) pygame.draw.rect(screen, (0, 0, 0), player_rect)
```

Similarly, you can update the position of static platforms as follows:

```
# V? hình ch? nh?t t?
nh rectangle_1_draw_pos = rectangle_1.move(camera_offset_x, 0) pygame.draw.rect(s
????
 rectangle_2_draw_pos = rectangle_2.move(camera_offset_x, 0) pygame.draw.rect(sc
```

Move the camera with the keyboard

Now that you've successfully set up the camera, you can move it around. A simple way to do this is to use the keyboard. For example, you can move the camera to the left when the player presses the left arrow key.

To do this, add the following code inside the event loop that handles keystrokes:

```
if event.type == pygame.KEYDOWN: ????if event.key == pygame.K_LEFT:
????????camera_offset_x -= PLAYER_SPEED ????
elif event.key == pygame.K_RIGHT: ????????camera_offset_x += PLAYER_SPEED
```

The alternative is to change the player's x coordinate when the key is pressed, then update the camera offset. You can do this as follows:

```
# X? lý s? ki?n for event in pygame.event.get(): ????
if event.type == pygame.QUIT: ????????pygame.quit() ????????quit() ????
if event.type == pygame.KEYDOWN: ??????????if event.key == pygame.K_LEFT:
????????????????player_x -= PLAYER_SPEED ?????????
elif event.key == pygame.K_RIGHT: ????????????????player_x += PLAYER_SPEED
```

Then, update the camera offset corresponding to the player's x coordinate as follows:

```
camera_offset_x = WINDOW_WIDTH // 2 - player_x - PLAYER_WIDTH // 2
```

Move the camera with the mouse

Another way to move the camera is to use the mouse. You can allow the player to drag the screen around by clicking & dragging the mouse.

To do this, track the position of the mouse when the player presses the left mouse button. When moving the mouse, update the player's x coordinate. It will change according to the difference between the current mouse position and the original position you tracked, **mouse_start_pos** .

```
# X? lý s? ki?n for event in pygame.event.get(): ????
if event.type == pygame.QUIT: ????????pygame.quit() ????????quit() ????
if event.type == pygame.MOUSEBUTTONDOWN: ????????if event.button == 1:
???????????mouse_start_pos = pygame.mouse.get_pos() ????
if event.type == pygame.MOUSEMOTION: ????????
if pygame.mouse.get_pressed()[0]: ???????????
mouse_current_pos = pygame.mouse.get_pos() ???????????
mouse_offset_x = mouse_current_pos[0] - mouse_start_pos[0] ???????????
player_x -= mouse_offset_x ???????????mouse_start_pos = mouse_current_pos
```

Above is **how to add camera roll to PyGame** . Hope the article is useful to you.

You finished reading the article "**How to add a scrolling camera to PyGame**" edited by the **TipsMake** team. We hope this article has provided you with many useful tech tips and tricks. You can search for similar articles on tips and guides. Thank you for reading and for following us regularly.