Pygame - Dev Notes - Sprites - Relative Objects

- Dr Nick Hayward

A brief intro on adding new sprite objects relative to other sprite objects in a game window with Pygame.

Contents

- Intro
- Add new sprites
- Listen for keypress
- Release new sprites
- References
- Demo

Intro

For a given sprite object in the game window, we may use the location of this object as a starting point for other sprite objects. For example, we may release laser beams from the top of a player's ship, or allow a character to throw a flaming ball across the screen.

Add new sprites

To create some new sprite objects, we can create a new class for this sprite object. e.g. some projectiles that a player may appear to fire from the top of a player object, such as a ship &c.

```
# create a generic projectile sprite - for bullets, lasers &c.
class Projectile(pygame.sprite.Sprite):
    # x, y - add specific location for object relative to player sprite
    def __init__(self, x, y):
        pygame.sprite.Sprite.__init__(self)
        self.image = pygame.Surface((5, 10))
        self.image.fill(RED)
        self.rect = self.image.get_rect()
        # weapon fired from front (top) of player sprite...
        self.rect.bottom = y
        self.rect.centerx = x
        # speed of projectile up the screen
        self.speed_y = -10

    def update(self):
        # update y relative to speed of projectile on y-axis
        self.rect.y += self.speed_y
        # remove from game window - if it goes beyond bounding for y-axis at top...
        if self.rect.bottom < 0:
            # kill() removes specified sprite from group...
```

```
            self.kill()
```

In this class, we're creating another sprite object for a projectile such as a laser beam. This projectile will be shot from the top of another object, in this example a player's ship, so we may set the x and y coordinates relative to the position of this player object. We're setting the speed along the y-axis so it travels up the screen.

As we update each projectile object, we simply update its speed, and then check its position on the screen. If it leaves the top of the game window, we can call the generic kill() method on this sprite. This method is available for any sprite object we create in the game window.

Listen for keypress

Then, we need to add a new listener to the game loop to detect a keypress for the *spacebar*. We'll use this keypress to allow a player to shoot these projectiles, a laser beam for example.

```
# 'processing' inputs (events)
for event in EVENTS.get():
    # check keyboard events - keydown
    if event.type == pygame.KEYDOWN:
        # check for ESCAPE key
        if event.key == pygame.K_ESCAPE:
            gameExit()
        elif event.key == pygame.K_SPACE:
            # fire laser beam...
            player.fire()
```

In this example, we've updated our keypress listeners to include a check for each time a player hits down on the *spacebar*. We can use this keypress event to fire our projectile, a laser beam to hit our enemy mobs.

Release new sprites

As the player hits the *spacebar*, for example, we need to create new sprites. These new sprite objects will then be released from the top of the player's object. In effect, the relative position of one sprite object is determining the start position of another sprite object.

So, we now need to update the class for our primary sprite object, e.g. a player, to include a method for firing the projectiles from the top of this sprite object.

For example,

```
# fire projectile from top of player sprite object
def fire(self):
    # set position of projectile relative to player's object rect for centerx and
top
    projectile = Projectile(self.rect.centerx, self.rect.top)
    # add projectile to game sprites group
    game_sprites.add(projectile)
    # add each projectile to sprite group for all projectiles
    projectiles.add(projectile)
```

This sets the start position for the x and y coordinates of each projectile sprite to the current position of the player's sprite object. Then, we simply add each projectile sprite object to the main game sprite group, and a new sprite group for all of the projectiles. We can add this new sprite group as follows,

```
projectiles = pygame.sprite.Group()
```

Now, when a player presses down on the *spacebar* a projectile, our red laser beam, will be fired from the top of the player's sprite object.

References

- pygame.sprite

Demo

- basicsprites5.py