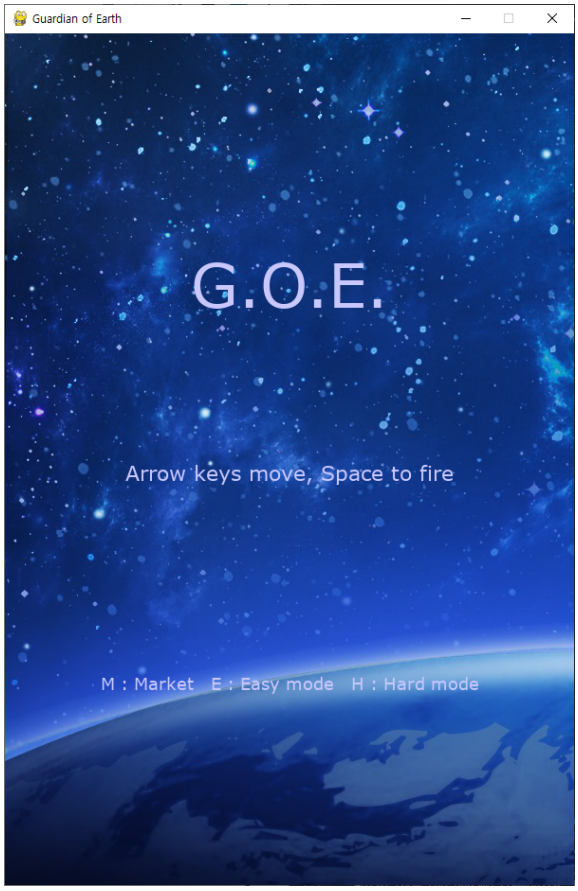
GOE

Genre : scroll shooting game

Library : pygame

Initial screen



There is 3 choices, go to market, play easy mode, and play hard mode.

Market(M) : Pay coin to upgrade your plane

Easy mode(E) : Increase 1 mob per 9 second

Hard mode(H) : Increase 1 mob per 3 second, powerups come less than easy mode.

Market

PowerUP Time(T) : Increase duration of powerup

SHIELD(H) : Increase max shield

SPEED(S) : Increase speed

SHOOT SPEED(F) : Decrease delay of shoot

P : Go back to initial screen

Play screen

1 : Number of life

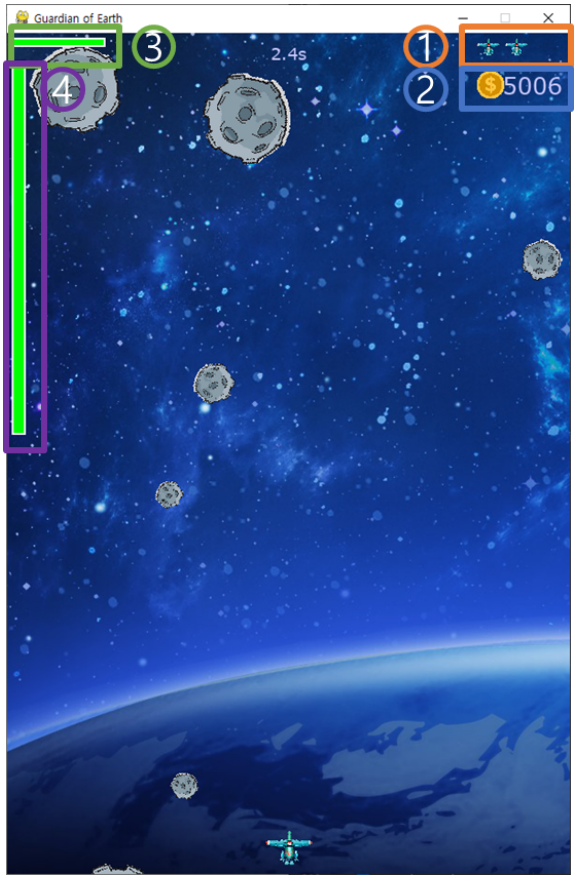
2 : Coin

3 : Shield bar

4 : Power bar

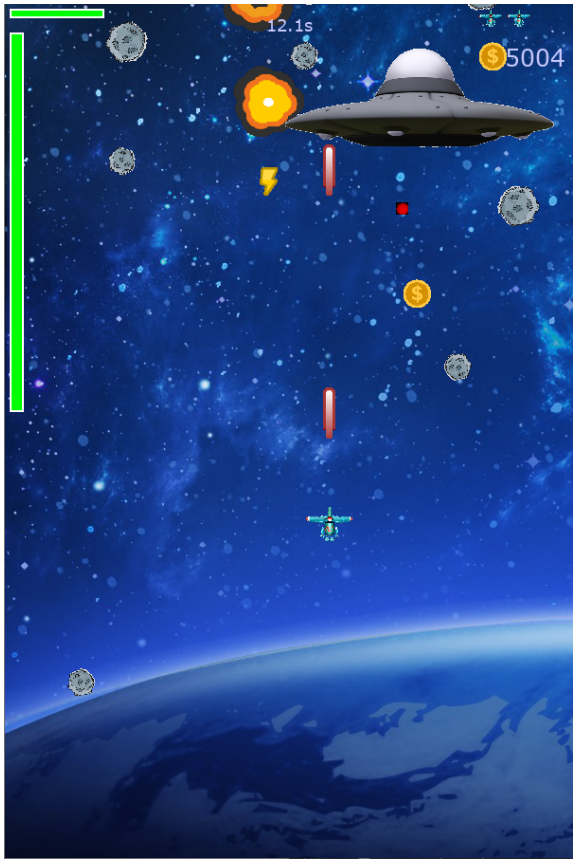
It turns orange when power lv2 and red

when power lv3



Boss

Every 12second boss appears. It attacks player. Bullets of boss are unbreakable. You should dodge by moving plane.



Class

▼ Player

```
class Player(pygame.sprite.Sprite):
    def __init__(self):
        pygame.sprite.Sprite.__init__(self)
        self.image = pygame.transform.scale(player_img, (50, 38))
        self.image.set_colorkey(BLACK)
        self.rect = self.image.get_rect()
        self.radius = 20
        # pygame.draw.circle(self.image, RED, self.rect.center, self.radius)
        self.rect.centerx = WIDTH / 2
        self.rect.bottom = HEIGHT - 10
        self.speed = 6
        self.speedx = 0
        self.speedy = 0
        self.maxshield = 100
        self.shield = self.maxshield
        self.shoot_delay = 250
        self.last_shot = pygame.time.get_ticks()
        self.lives = 2
        self.hidden = False
        self.hide_timer = pygame.time.get_ticks()
        self.power = 1
```

```
        self.power_time = pygame.time.get_ticks()
        self.coin = 5000

    def update(self):
        # timeout for powerups
        if self.power >= 2:
            if self.power >= 4 or pygame.time.get_ticks() - self.power_time > POWE
RUP_TIME:
                self.power -= 1
                self.power_time = pygame.time.get_ticks()

        # unhide if hidden
        if self.hidden and pygame.time.get_ticks() - self.hide_timer > 1000:
            self.hidden = False
            self.rect.centerx = WIDTH / 2
            self.rect.bottom = HEIGHT - 10

        self.speedx = 0
        self.speedy = 0
        keystate = pygame.key.get_pressed()
        if keystate[pygame.K_LEFT]:
            self.speedx = -self.speed
        if keystate[pygame.K_RIGHT]:
            self.speedx = self.speed
        if keystate[pygame.K_UP]:
            self.speedy = -self.speed
        if keystate[pygame.K_DOWN]:
            self.speedy = self.speed
        if keystate[pygame.K_SPACE]:
            self.shoot()
        self.rect.x += self.speedx
        self.rect.y += self.speedy
        if self.rect.right > WIDTH:
            self.rect.right = WIDTH
        if self.rect.left < 0:
            self.rect.left = 0
        if self.rect.bottom > HEIGHT:
            self.rect.bottom = HEIGHT
        if self.rect.top < HEIGHT/5:
            self.rect.top = HEIGHT/5

    def powerup(self):
        self.power += 1
        self.power_time = pygame.time.get_ticks()

    def shoot(self):
        now = pygame.time.get_ticks()
        if now - self.last_shot > self.shoot_delay:
            self.last_shot = now
            if self.power == 1:
                bullet = Bullet(self.rect.centerx, self.rect.top)
                all_sprites.add(bullet)
                bullets.add(bullet)
                shoot_sound.play()
            elif self.power == 2:
                bulletl = [Bullet(self.rect.left, self.rect.centery),
                           Bullet(self.rect.right, self.rect.centery)]
                all_sprites.add(*bulletl)
```

```
                bullets.add(*bulletl)
                shoot_sound.play()
            elif self.power >= 3:
                bulletl = [Bullet(self.rect.left, self.rect.centery),
                           Bullet(self.rect.right, self.rect.centery),
                           Bullet(self.rect.centerx, self.rect.top)]
                all_sprites.add(*bulletl)
                bullets.add(*bulletl)
                shoot_sound.play()

    def hide(self):
        # hide the player temporarily
        self.hidden = True
        self.hide_timer = pygame.time.get_ticks()
        self.rect.center = (WIDTH / 2, HEIGHT + 200)
```

▼ Mob

```
class Mob(pygame.sprite.Sprite):
    def __init__(self):
        pygame.sprite.Sprite.__init__(self)
        self.image_orig = random.choice(meteor_images)
        self.image_orig.set_colorkey(BLACK)
        self.image = self.image_orig.copy()
        self.rect = self.image.get_rect()
        self.radius = int(self.rect.width * .85 / 2)
        self.rect.x = random.randrange(WIDTH - self.rect.width)
        self.rect.bottom = random.randrange(-80, -20)
        self.speedy = random.randrange(1, 8)
        self.speedx = random.randrange(-3, 3)
        self.rot = 0
        self.rot_speed = random.randrange(-8, 8)
        self.last_update = pygame.time.get_ticks()

    def rotate(self):
        now = pygame.time.get_ticks()
        if now - self.last_update > 50:
            self.last_update = now
            self.rot = (self.rot + self.rot_speed) % 360
            new_image = pygame.transform.rotate(self.image_orig, self.rot)
            old_center = self.rect.center
            self.image = new_image
            self.rect = self.image.get_rect()
            self.rect.center = old_center

    def update(self):
        self.rotate()
        self.speedx *= np.random.choice([1,-1],1,p=[.995,.005])[0]
        self.rect.x += self.speedx
        self.rect.y += self.speedy
        if self.rect.top > HEIGHT + 10 or self.rect.left < -100 or self.rect.right
> WIDTH + 100:
            self.rect.x = random.randrange(WIDTH - self.rect.width)
```

```
            self.rect.y = random.randrange(-100, -40)
            self.speedy = random.randrange(1, 8)
```

▼ Boss

```
class Boss(pygame.sprite.Sprite):
    def __init__(self):
        pygame.sprite.Sprite.__init__(self)
        img = pygame.image.load(path.join(img_dir, "BOSS.png")).convert()
        img = pygame.transform.scale(img, (288, 187))
        self.image = img
        self.image.set_colorkey(BLACK)
        self.rect = self.image.get_rect()
        self.rect.x = random.randrange(WIDTH - self.rect.width)
        self.rect.top = 5
        self.lives = 50
        self.speedx = 3
        self.shoot_delay = 300
        self.last_shot = 0
        self.last_update = pygame.time.get_ticks()

    def shoot(self):
        now = pygame.time.get_ticks()
        if now - self.last_shot > self.shoot_delay:
            self.last_shot = now
            bullet = BossBullet(self.rect.centerx, self.rect.bottom)
            all_sprites.add(bullet)
            bossbullets.add(bullet)
        # shoot_sound.play()

    def hide(self):
        self.kill()

    def update(self):
        self.rect.x += self.speedx
        if self.rect.centerx > WIDTH or self.rect.centerx < 0:
            self.speedx *= -1
        self.shoot()
```

▼ Bullet

```
class Bullet(pygame.sprite.Sprite):
    def __init__(self, x, y):
        pygame.sprite.Sprite.__init__(self)
        self.image = bullet_img
        self.image.set_colorkey(BLACK)
        self.rect = self.image.get_rect()
        self.rect.bottom = y
        self.rect.centerx = x
        self.speedy = -10

    def update(self):
```

```
        self.rect.y += self.speedy
        if self.rect.bottom < 0:
            self.kill()
```

▼ BossBullet

```
class BossBullet(Bullet, pygame.sprite.Sprite):
    def __init__(self, x, y):
        pygame.sprite.Sprite.__init__(self)
        super().__init__(x, y)
        self.speedy = 5
        self.image = boss_bullet_img
```

▼ Pow

```
class Pow(pygame.sprite.Sprite):
    def __init__(self, center, percent):
        pygame.sprite.Sprite.__init__(self)
        self.type = np.random.choice(['shield', 'gun', 'coin'], p=percent)
        self.image = powerup_images[self.type]
        self.image.set_colorkey(BLACK)
        self.rect = self.image.get_rect()
        self.rect.center = center
        self.speedy = 5

    def update(self):
        self.rect.y += self.speedy
        # kill if it moves off the top of the screen
        if self.rect.top > HEIGHT:
            self.kill()
```

▼ Explosion

```
class Explosion(pygame.sprite.Sprite):
    def __init__(self, center, size):
        pygame.sprite.Sprite.__init__(self)
        self.size = size
        self.image = explosion_anim[self.size][0]
        self.rect = self.image.get_rect()
        self.rect.center = center
        self.frame = 0
        self.last_update = pygame.time.get_ticks()
        self.frame_rate = 75

    def update(self):
        now = pygame.time.get_ticks()
        if now - self.last_update > self.frame_rate:
            self.last_update = now
            self.frame += 1
```

```
            if self.frame == len(explosion_anim[self.size]):
                self.kill()
            else:
                center = self.rect.center
                self.image = explosion_anim[self.size][self.frame]
                self.rect = self.image.get_rect()
                self.rect.center = center
```

▼ Warning_message

```
class Warning_message():
    def __init__(self):
        self.image = warning_img
        self.rect = self.image.get_rect()
        self.rect.center = [WIDTH/2, HEIGHT/3]
        self.time = 30

    def draw(self):
        if self.time>=0:
            self.image.set_alpha(abs(10 - self.time%20)*10)
            screen.blit(self.image, self.rect)
            self.time -= .2
            return True
        else:
            return False
```

Function

▼ show_go_screen

```
def show_go_screen():
    waiting = True
    while waiting:
        screen.blit(background, background_rect)
        draw_text(screen, "G.O.E.", 64, WIDTH / 2, HEIGHT / 4)
        draw_text(screen, "Arrow keys move, Space to fire", 22,
                WIDTH / 2, HEIGHT / 2)
        draw_text(screen, "M : Market   E : Easy mode   H : Hard mode", 18, WIDTH
 / 2, HEIGHT * 3 / 4)
        pygame.display.flip()
        clock.tick(FPS)
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                pygame.quit()
            if event.type == pygame.KEYUP:
                if event.key == pygame.K_m:
                    show_market()
                elif event.key == pygame.K_e:
                    respawn = 3000
                    percent = [.2,.2,.6]
```

```
                    waiting = False
                elif event.key == pygame.K_h:
                    respawn = 300
                    percent = [.1,.1,.8]
                    waiting = False
                else:
                    waiting = False
```

▼ show_market

```
def show_market():
    def item(name, x, y, cost, key):
        draw_text(screen, f"{name} ({key})", 20, x+100, y)
        img = pygame.image.load(path.join(img_dir, f"{name}.png")).convert()
        img = pygame.transform.scale(img, (200, 200))
        img_rect = img.get_rect()
        img_rect = [x,y+23]
        screen.blit(img, img_rect)
        draw_text(screen, f"COST : {cost}", 20, x+100, y+223)
    waiting = True
    while waiting:
        screen.blit(background, background_rect)
        draw_text(screen, "MARKET", 64, WIDTH / 2, HEIGHT / 12)
        # draw_text(screen, "Arrow keys move, Space to fire", 22,
        #           WIDTH / 2, HEIGHT / 2)
        draw_text(screen, "Press P to play", 18, WIDTH / 2, HEIGHT * 5/6)
        item('PowerUP Time',30,HEIGHT / 4 - 50, 50, 'T')
        item('SHIELD',WIDTH - 230 ,HEIGHT / 4 - 50, 50, 'H')
        item('SPEED',30,HEIGHT / 2, 50, 'S')
        item('SHOOT SPEED',WIDTH - 230,HEIGHT / 2, 50, 'F')
        clock.tick(FPS)
        draw_coin(screen, WIDTH - 100, 5, coin_img, player.coin)
        pygame.display.flip()
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                pygame.quit()
            if event.type == pygame.KEYUP:
                if event.key == pygame.K_t:
                    if player.coin >= 50:
                        player.coin -= 50
                        player.power_time += 200
                elif event.key == pygame.K_h:
                    if player.coin >= 50:
                        player.coin -= 50
                        player.maxshield += 10
                elif event.key == pygame.K_s:
                    if player.coin >= 50:
                        player.coin -= 50
                        player.speed += .5
                elif event.key == pygame.K_f and player.shoot_delay>5:
                    if player.coin >= 80:
                        player.coin -= 80
                        player.shoot_delay = max(player.shoot_delay-5,0)
                elif event.key == pygame.K_p:
                    waiting = False
```

Init state and loop

▼ init

```
# Load all game graphics
background = pygame.image.load(path.join(img_dir, "background_earth.jpg")).convert
()
background_rect = background.get_rect()
player_img = pygame.image.load(path.join(img_dir, "playerShip1_blue.png")).convert
()
# player_img = pygame.image.load(path.join(img_dir, "playerShip1_orange.png")).con
vert()
player_mini_img = pygame.transform.scale(player_img, (25, 19))
player_mini_img.set_colorkey(BLACK)
bullet_img = pygame.image.load(path.join(img_dir, "laserRed16.png")).convert()
boss_bullet_img = pygame.image.load(path.join(img_dir, "boss_bullet.png")).convert
()
boss_bullet_img = pygame.transform.scale(boss_bullet_img, (13, 13))
meteor_img = pygame.image.load(path.join(img_dir, "meteor.png")).convert()
meteor_images = [pygame.transform.scale(meteor_img, (i, i)) for i in (100,45,30)]
explosion_anim = {}
explosion_anim['lg'] = []
explosion_anim['sm'] = []
explosion_anim['player'] = []
for i in range(9):
    filename = 'regularExplosion0{}.png'.format(i)
    img = pygame.image.load(path.join(img_dir, filename)).convert()
    img.set_colorkey(BLACK)
    img_lg = pygame.transform.scale(img, (75, 75))
    explosion_anim['lg'].append(img_lg)
    img_sm = pygame.transform.scale(img, (32, 32))
    explosion_anim['sm'].append(img_sm)
    filename = 'sonicExplosion0{}.png'.format(i)
    img = pygame.image.load(path.join(img_dir, filename)).convert()
    img.set_colorkey(BLACK)
    explosion_anim['player'].append(img)
powerup_images = {}
coin_img = pygame.image.load(path.join(img_dir, "coin.png")).convert()
coin_img = pygame.transform.scale(coin_img, (30, 30))
powerup_images['coin'] = coin_img
powerup_images['shield'] = pygame.image.load(path.join(img_dir, 'shield_gold.pn
g')).convert()
powerup_images['gun'] = pygame.image.load(path.join(img_dir, 'bolt_gold.png')).con
vert()
warning_img = pygame.image.load(path.join(img_dir, "warning.png")).convert()
warning_img = pygame.transform.scale(warning_img, (600, 200))

# Load all game sounds
shoot_sound = pygame.mixer.Sound(path.join(snd_dir, 'pew.wav'))
shield_sound = pygame.mixer.Sound(path.join(snd_dir, 'pow4.wav'))
power_sound = pygame.mixer.Sound(path.join(snd_dir, 'pow5.wav'))
expl_sounds = [pygame.mixer.Sound(path.join(snd_dir, snd)) for snd in ['expl3.wa
```

```
v', 'expl6.wav']]
player_die_sound = pygame.mixer.Sound(path.join(snd_dir, 'rumble1.ogg'))
pygame.mixer.music.load(path.join(snd_dir, 'tgfcoder-FrozenJam-SeamlessLoop.ogg'))
pygame.mixer.music.set_volume(0.4)

pygame.mixer.music.play(loops=-1)
# Game loop
game_over = True
running = True
player = Player()
flag = False
mobflag = True
b = True
respawn = 3000
percent = [.2,.2,.6]
```

▼ loop

```
while running:
    if not game_over:
        if (pygame.time.get_ticks() - t) % respawn <= 10:
            if mobflag:
                newmob()
                mobflag = False
        else:
            mobflag = True

        if len(mobs)%7 == 2 and b:
            warning = Warning_message()
            flag = warning.draw()
            b = False

        if len(mobs)%7 == 3 and not b:
            boss = Boss()
            boss.lives += 10*boss_count
            all_sprites.add(boss)
            b = True

    if game_over:
        show_go_screen()
        game_over = False
        player.shield = player.maxshield
        player.lives = 2
        all_sprites = pygame.sprite.Group()
        mobs = pygame.sprite.Group()
        bullets = pygame.sprite.Group()
        bossbullets = pygame.sprite.Group()
        powerups = pygame.sprite.Group()
        all_sprites.add(player)
        t = pygame.time.get_ticks()
        boss_count = 0
        for i in range(7):
            newmob()
        score = 0
```

```
    # keep loop running at the right speed
    clock.tick(FPS)
    # Process input (events)
    for event in pygame.event.get():
        # check for closing window
        if event.type == pygame.QUIT:
            running = False

    # Update
    all_sprites.update()

    # check to see if a bullet hit a mob
    hits = pygame.sprite.groupcollide(mobs, bullets, True, True)
    for hit in hits:
        score += 50 - hit.radius
        random.choice(expl_sounds).play()
        expl = Explosion(hit.rect.center, 'lg')
        all_sprites.add(expl)
        if random.random() > 0.5:
            pow = Pow(hit.rect.center, percent)
            all_sprites.add(pow)
            powerups.add(pow)
        newmob()

    # check to see if a mob hit the player
    hits = pygame.sprite.spritecollide(player, mobs, True, pygame.sprite.collide_c
ircle)
    for hit in hits:
        player.shield -= hit.radius * 3
        expl = Explosion(hit.rect.center, 'sm')
        all_sprites.add(expl)
        newmob()
        if player.shield <= 0:
            player_die_sound.play()
            death_explosion = Explosion(player.rect.center, 'player')
            all_sprites.add(death_explosion)
            player.hide()
            player.lives -= 1
            player.shield = player.maxshield

    # check to see if a boss bullet hit the player
    hits = pygame.sprite.spritecollide(player, bossbullets, True, pygame.sprite.co
llide_circle)
    for hit in hits:
        player.shield -= 30
        expl = Explosion(hit.rect.center, 'sm')
        all_sprites.add(expl)
        if player.shield <= 0:
            player_die_sound.play()
            death_explosion = Explosion(player.rect.center, 'player')
            all_sprites.add(death_explosion)
            player.hide()
            player.lives -= 1
            player.shield = player.maxshield

    try:
        # check to see if a bullet hit the boss
        hits = pygame.sprite.spritecollide(boss, bullets, True, pygame.sprite.coll
```

```
ide_mask)
        for hit in hits:
            expl = Explosion(hit.rect.center, 'sm')
            all_sprites.add(expl)
            boss.lives -= 1
            if boss.lives <= 0:
                death_explosion = Explosion(boss.rect.center, 'player')
                # boss.hide()
                boss.rect.right = 0
                boss_count += 1
                all_sprites.remove(boss)
                all_sprites.add(death_explosion)
    except:
        pass

    # check to see if player hit a powerup
    hits = pygame.sprite.spritecollide(player, powerups, True)
    for hit in hits:
        if hit.type == 'shield':
            player.shield += random.randrange(10, 30)
            shield_sound.play()
            if player.shield >= player.maxshield:
                player.shield = player.maxshield
        if hit.type == 'gun':
            player.powerup()
            power_sound.play()
        if hit.type == 'coin':
            player.coin += 1

    # if the player died and the explosion has finished playing
    if player.lives == 0 and not death_explosion.alive():
        game_over = True

    # Draw / render
    screen.fill(BLACK)
    screen.blit(background, background_rect)
    if flag:
        flag = warning.draw()
    all_sprites.draw(screen)
    draw_text(screen, f"{(pygame.time.get_ticks() - t)/1000:.1f}s", 18, WIDTH / 2,
10)
    draw_shield_bar(screen, 5, 5, player.shield/player.maxshield*100)
    draw_power_bar(screen, 5, 30, player.power, player.power_time, pygame.time.get
_ticks())
    draw_lives(screen, WIDTH - 100, 5, player.lives, player_mini_img)
    draw_coin(screen, WIDTH - 100, 40, coin_img, player.coin)
    # *after* drawing everything, flip the display
    pygame.display.flip()
```

Reference

shmup! https://kidscancode.org/blog/2016/11/pygame_shmup_part_14/