PyGame Tractor Pong Tutorial - Part 4

Contents

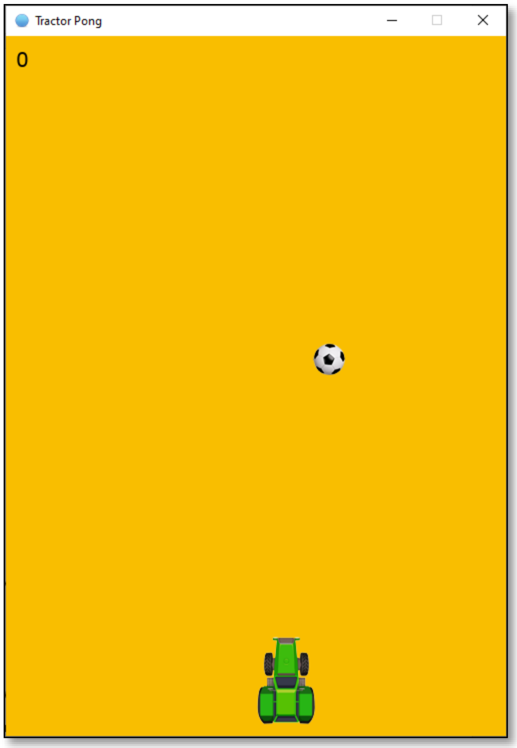
Time required: 30 minutes

Preview of the Game

Atari. - the year: 1973 - the date: - November 29th -

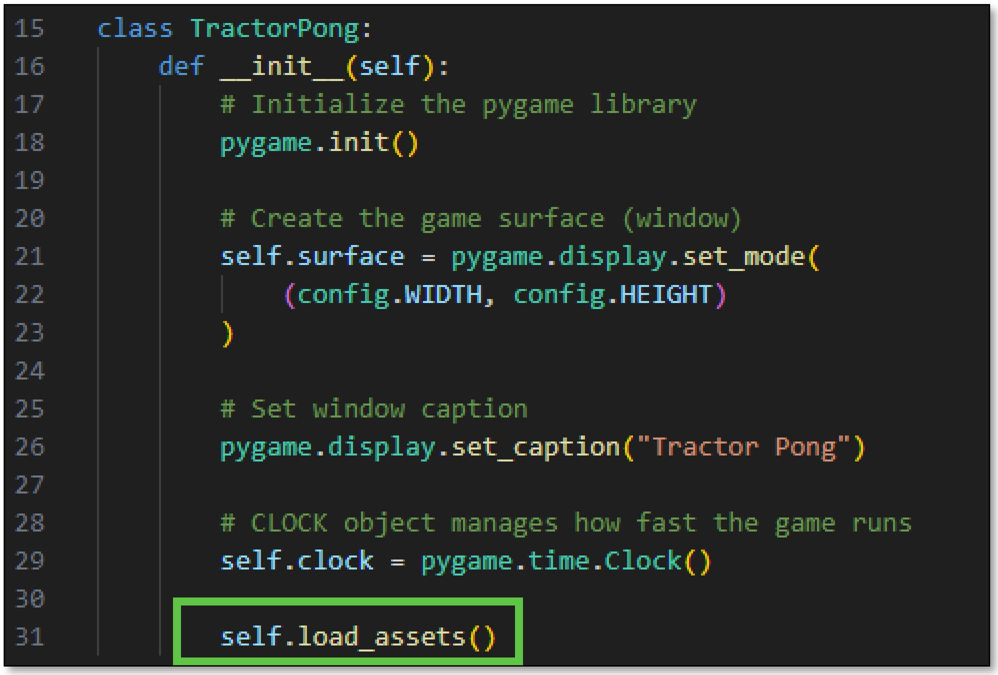
That game is called Pong . . . . Then there was Tractor Pong.

Tractor Pong Demo Video



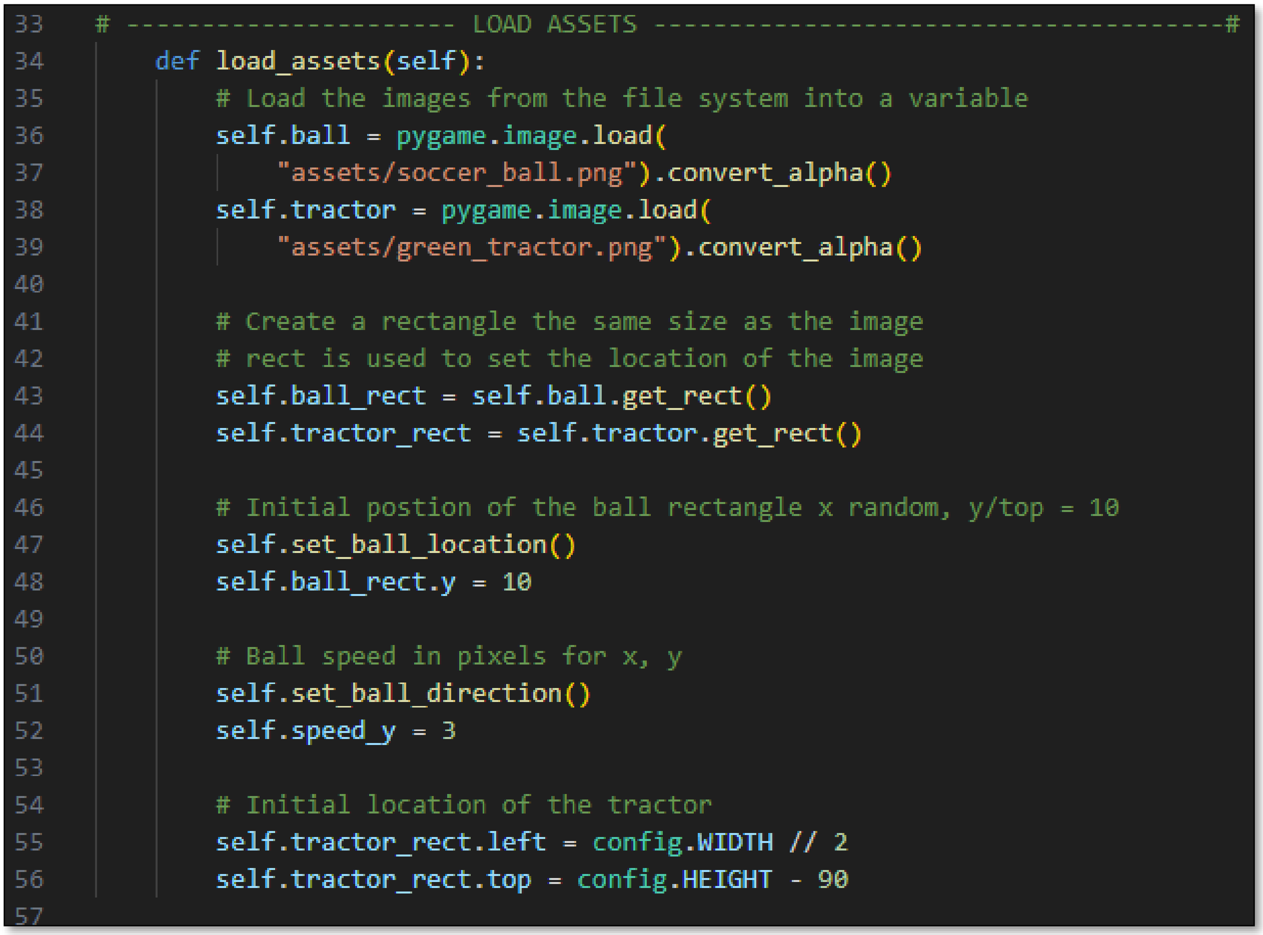
Time to Bounce

1. Save **tractor_pong_3.py** as **tractor_pong_4.py**
2. Let's move things around into different methods. Some of the code that was in here will be moved to different methods.



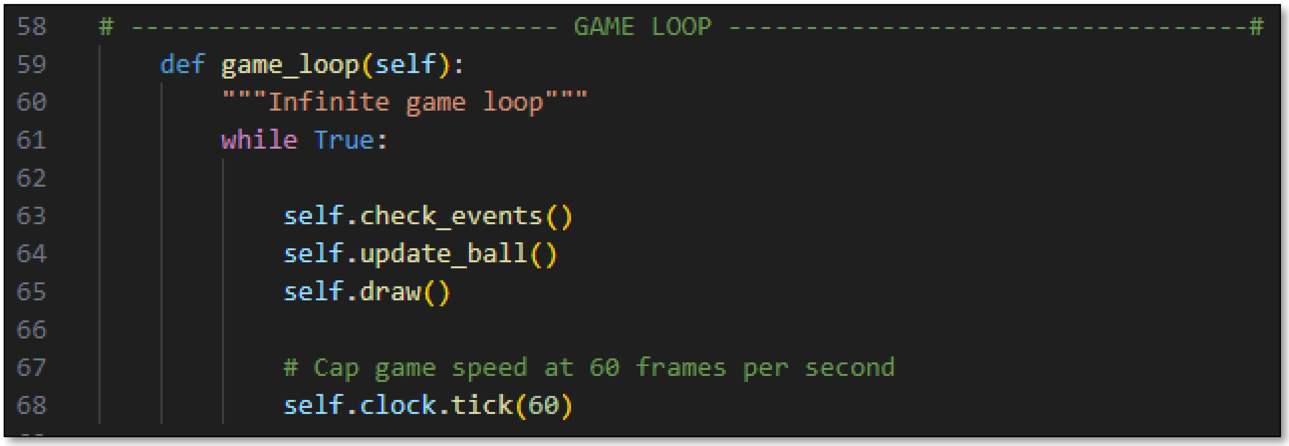
```python
class TractorPong:
    def __init__(self):
        # Initialize the pygame library
        pygame.init()

        # Create the game surface (window)
        self.surface = pygame.display.set_mode(
            (config.WIDTH, config.HEIGHT)
        )

        # Set window caption
        pygame.display.set_caption("Tractor Pong")

        # CLOCK object manages how fast the game runs
        self.clock = pygame.time.Clock()

        self.load_assets()
```

Create a load_assets method. Some of this is code we have already written, copy and paste it into place.



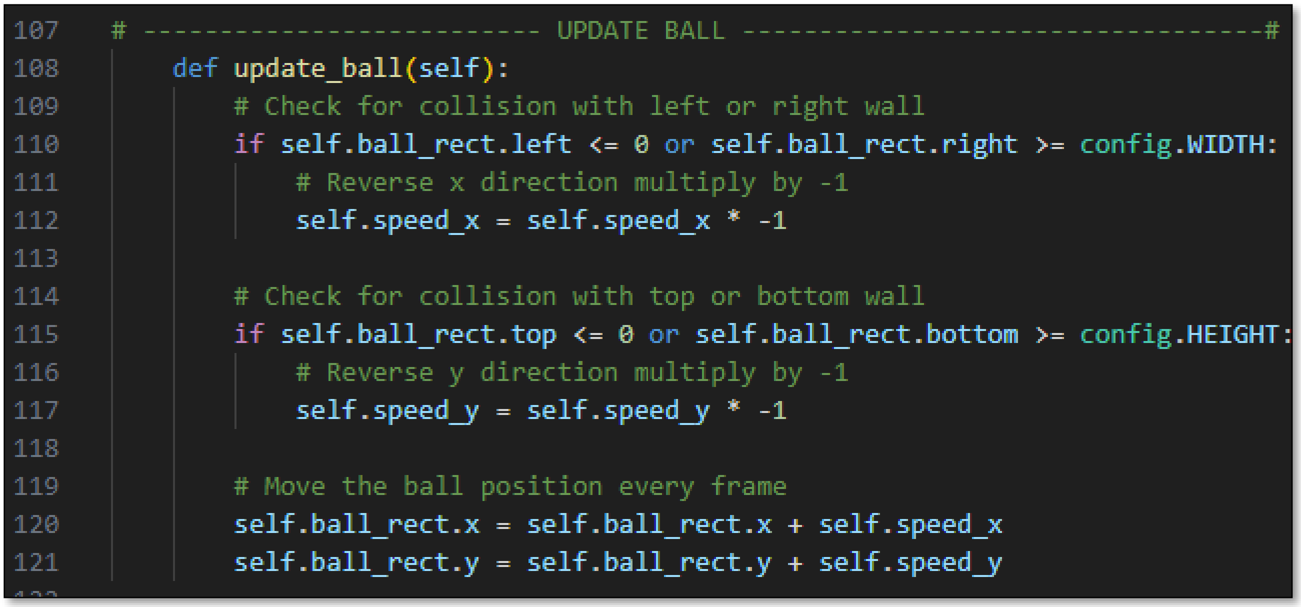
```python
# ---------------------- LOAD ASSETS --------------------------------------#
    def load_assets(self):
        # Load the images from the file system into a variable
        self.ball = pygame.image.load(
            "assets/soccer_ball.png").convert_alpha()
        self.tractor = pygame.image.load(
            "assets/green_tractor.png").convert_alpha()

        # Create a rectangle the same size as the image
        # rect is used to set the location of the image
        self.ball_rect = self.ball.get_rect()
        self.tractor_rect = self.tractor.get_rect()

        # Initial postion of the ball rectangle x random, y/top = 10
        self.set_ball_location()
        self.ball_rect.y = 10

        # Ball speed in pixels for x, y
        self.set_ball_direction()
        self.speed_y = 3

        # Initial location of the tractor
        self.tractor_rect.left = config.WIDTH // 2
        self.tractor_rect.top = config.HEIGHT - 90
```

Our game loop is going to much simpler. We are copying some of the code that used to be here into other methods.



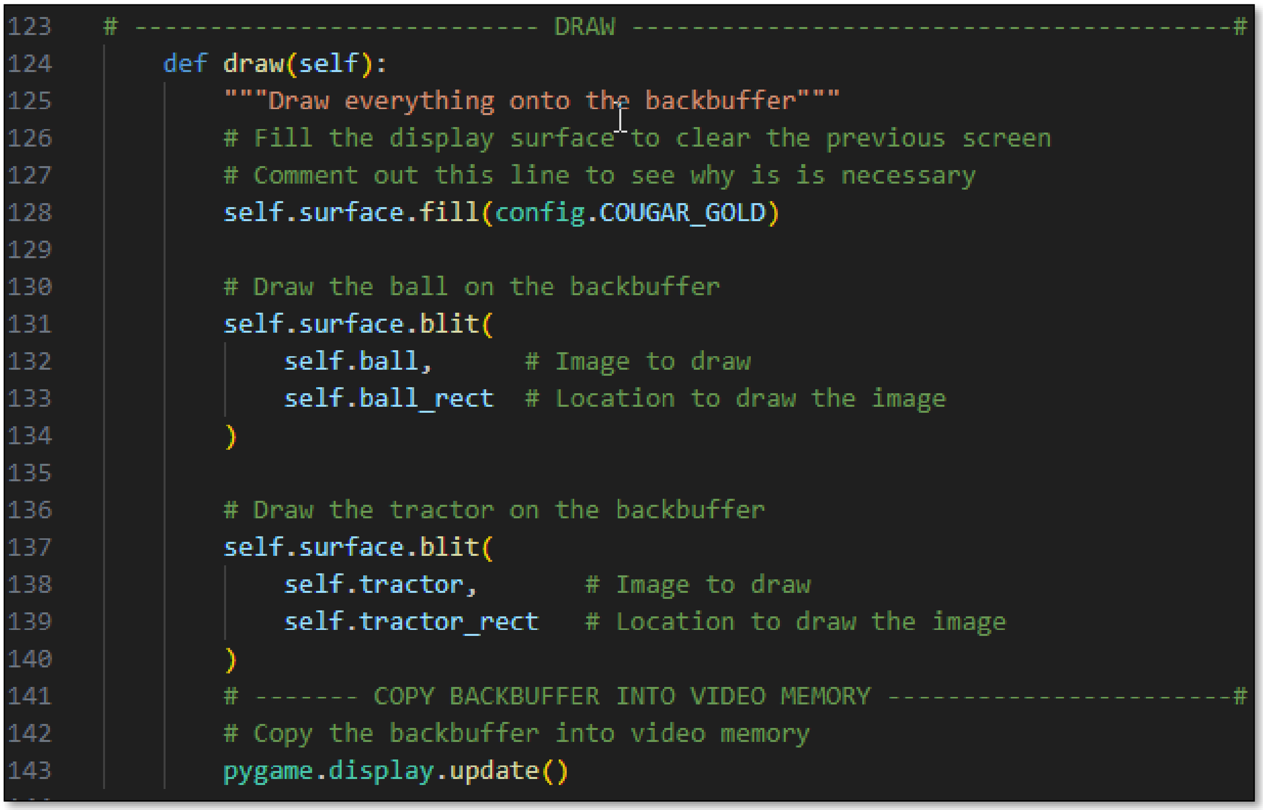
```python
# ---------------------------- GAME LOOP -----------------------------------#
    def game_loop(self):
        """Infinite game loop"""
        while True:

            self.check_events()
            self.update_ball()
            self.draw()

            # Cap game speed at 60 frames per second
            self.clock.tick(60)
```

Everything to do with updating the ball will be in this method.

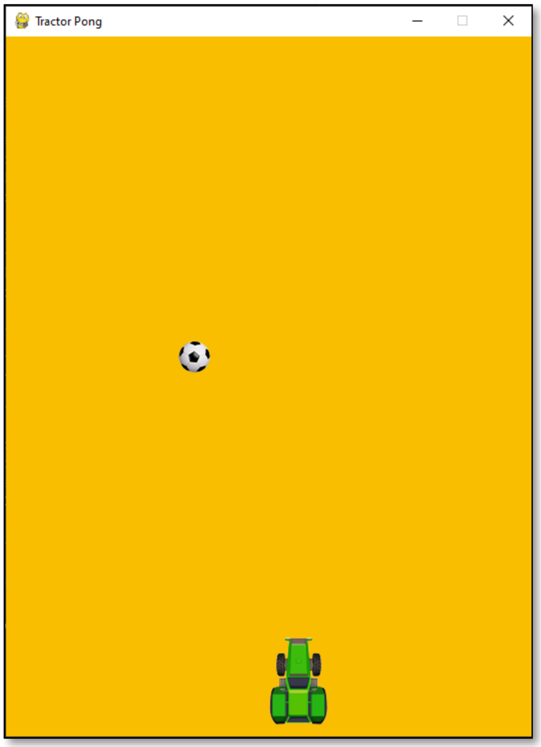


```
# --------------------------- UPDATE BALL -----------------------------------#
    def update_ball(self):
        # Check for collision with left or right wall
        if self.ball_rect.left <= 0 or self.ball_rect.right >= config.WIDTH:
            # Reverse x direction multiply by -1
            self.speed_x = self.speed_x * -1

        # Check for collision with top or bottom wall
        if self.ball_rect.top <= 0 or self.ball_rect.bottom >= config.HEIGHT:
            # Reverse y direction multiply by -1
            self.speed_y = self.speed_y * -1

        # Move the ball position every frame
        self.ball_rect.x = self.ball_rect.x + self.speed_x
        self.ball_rect.y = self.ball_rect.y + self.speed_y
```

All drawing and rendering is in this method. Much of this code was in the game loop.



```
# --------------------------- DRAW ----------------------------------------#
    def draw(self):
        """Draw everything onto the backbuffer"""
        # Fill the display surface to clear the previous screen
        # Comment out this line to see why is is necessary
        self.surface.fill(config.COUGAR_GOLD)

        # Draw the ball on the backbuffer
        self.surface.blit(
            self.ball,      # Image to draw
            self.ball_rect  # Location to draw the image
        )

        # Draw the tractor on the backbuffer
        self.surface.blit(
            self.tractor,       # Image to draw
            self.tractor_rect   # Location to draw the image
        )
        # ------- COPY BACKBUFFER INTO VIDEO MEMORY ----------------------#
        # Copy the backbuffer into video memory
        pygame.display.update()
```



The ball bounces around the screen off the walls.

Assignment Submission

1. Attach all tutorials and assignments.
2. Attach screenshots showing the successful operation of each tutorial program.
3. Submit in Blackboard.