PyGame Pong Tutorial - Part 7

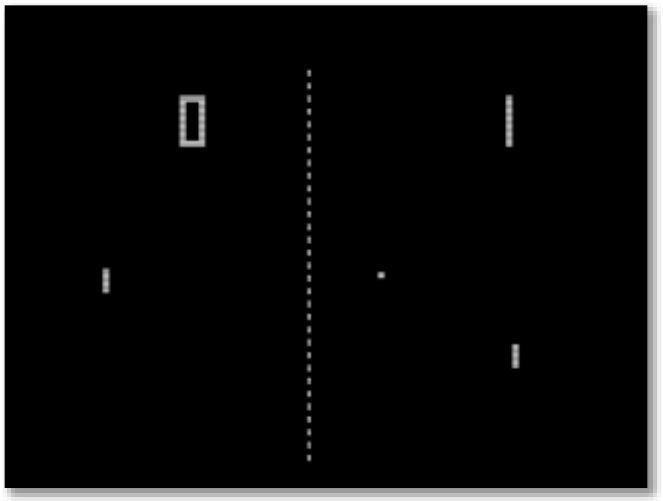
Contents

Time required: 30 minutes

Preview of the Game

Atari. - the year: 1973 - the date: - November 29th - The game is Pong.

Pong Demo Video



Sounds

You can use the sounds in the asset file, or create your own.

- https://www.beepbox.co (Create 8 bit songs.)

- https://sfxr.me/ (Create sound effects.)
- https://elevenlabs.io/sound-effects
- https://www.leshylabs.com/apps/sfMaker

Time for music, sound effects, a game over menu, and a real ping pong game.

pong_assets.zip is attached to this assignment. Unzip it into a folder underneath your game folder called **assets**

1. Save **pong_6.py** as **pong_7.py**
2. Add the following code.

```
"""
Name: pong_7.py
Author:
Date:
Purpose: Add sound and game over
"""

# pip install pygame-ce
import pygame

# pip install pygame-menu-ce
import pygame_menu as pm

# Import sys.exit to cleanly exit program
from sys import exit
from random import randint
from time import sleep
from config import BALL_COLOR, BG_COLOR, WIDTH, HEIGHT, BALL_RADIUS
from paddle import Paddle
```

```
19  class Pong:
20
21      def __init__(self):
22          # pre initalize mixer with larger buffer size for better performance
23          pygame.mixer.pre_init(
24              44100,  # frequency (Hz)
25              16,     # bit depth
26              2,      # number of channels, 1 mono, 2 stereo
27              4096    # buffer size, larger to optimize music playback.
28          )
29
30          # Initialize pygame
31          pygame.init()
```

```
62          self.score_font = pygame.font.SysFont("freesansbold", 18)
63          self.player_score = 0
64          self.computer_score = 0
65
66          # Load background music file into memory
67          pygame.mixer.music.load(
68              "./assets/inspiring-and-uplifting-indie-rock.mp3"
69          )
70
71          # Set volume to 30%, range from 0.0 (mute) to 1.0 (full volume)
72          pygame.mixer.music.set_volume(0.2)
73
74          # Play in a loop until stopped
75          pygame.mixer.music.play(-1)
```

Game Over

Add the following game_over method.

```
97      # ------------------------ DISPLAY GAME OVER ------------------------- #
98      def game_over(self):
99          """Display game over menu using the Pygame Menu library"""
100         # Stop background sound
101         pygame.mixer.music.stop()
102
103         # Play crash sound
104         crash = pygame.mixer.Sound("./assets/game_over.wav")
105         crash.play()
106         crash.set_volume(0.3)
107
108         # Wait 2 second while crash plays
109         sleep(2)
110
111         # Define a menu object for the game over screen
112         game_over = pm.Menu(
113             title="Game over",  # Set title menu to "Game over"
114             width=WIDTH,  # Set to width of game surface
115             height=HEIGHT,  # Set to height of game surface
116             # Set the theme of the menu to an orange color scheme
117             theme=pm.themes.THEME_SOLARIZED,
118         )
```

There are different themes you can choose for the game_over object. This example uses THEME_SOLARIZED. You can use any of the following to customize your menu.

```
THEME_BLUE
THEME_DARK
THEME_DEFAULT
THEME_GREEN
THEME_ORANGE
THEME_SOLARIZED
```

Game Over continues

```
104         # Display final score
105         game_over.add.label(f"Player Score: {self.player_score}")
106         game_over.add.label(f"Computer Score: {self.computer_score}")
107
108         # Add label to provide space between buttons
109         game_over.add.label("")
110
111         # Add a button to the game over menu for exiting the game
112         game_over.add.button(
113             title="Play Again?",     # Button text
114             action=main              # Call main() to start over
115         )
116
117         # Add label to provide space between buttons
118         game_over.add.label("")
119
120         # Add a button to the game over menu for exiting the game
121         game_over.add.button(
122             title="Exit",            # Button text
123             action=pm.events.EXIT    # Exit the game when clicked
124         )
125
126         # Run the main loop of the game over menu on the specified surface
127         game_over.mainloop(self.surface)
```

Check Collision

Modify the check collision method.

```
188     # -------------------- CHECK COLLISION ------------------------------ #
189     def check_collision(self):
190         """Check for all collisions"""
191         # Check for collision with left or right wall
192         # Subtract ball radius to bounce off the edge of the ball
193         if self.ball.left < 0 or self.ball.right >= WIDTH:
194 
195             # Ball goes off the table
196             self.game_over()
197 
198         # Check for collision with top or bottom wall
199         if self.ball.top < 0 or self.ball.bottom >= HEIGHT:
200 
201             # Reverse y direction multiply by -1
202             self.ball_speed_y = self.ball_speed_y * -1
203 
204         # Ball collision with paddles
205         if self.ball.colliderect(self.player):
206             # Reverse ball direction
207             self.ball_speed_x *= -1
208             self.player_score += 1
209 
210             # Play ball bounce sound
211             crash = pygame.mixer.Sound("./assets/hit.wav")
212             crash.play()
213             crash.set_volume(0.3)
214 
215         elif self.ball.colliderect(self.computer):
216             # Reverse ball direction
217             self.ball_speed_x *= -1
218             self.computer_score += 1
219 
220             # Play ball bounce sound
221             crash = pygame.mixer.Sound("./assets/hit.wav")
222             crash.play()
223             crash.set_volume(0.3)
```

Put this code at the end of the program.

```
337  # ---------------------------- MAIN PROGRAM ------------------------------- #
338  def main():
339      # Create game instance/object
340      pong = Pong()
341      # Start the game with the run_game method
342      pong.game_loop()
343
344
345  main()
```

Example run:

Tada, a real game!

There is always room for improvement.

What's Next?

- Change the colors to different RGB color.
- Change the size or shape of the ball or paddles.
- Add more difficulty levels.
- Keep track of the highest score between games.
- Add more music, change the music
- Change the game to make it your own.

Assignment Submission

1. Attach a screenshot showing the operation of the program.

2. Zip up the program files folder and submit in Blackboard.