Pygame Car Crash Tutorial - Part 6

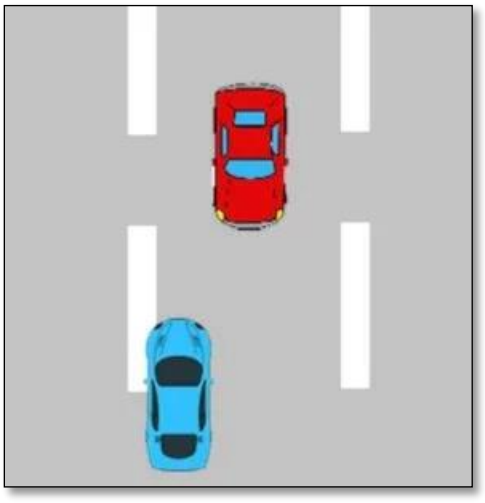
Contents

Time required: 30 minutes

Preview of the Game

Here's a sneak peak of the game that we are going to work on.

CarCrashDemo Video



Car Crash is simple arcade type game. The object is to move your blue car back and forth to avoid the oncoming red cars.

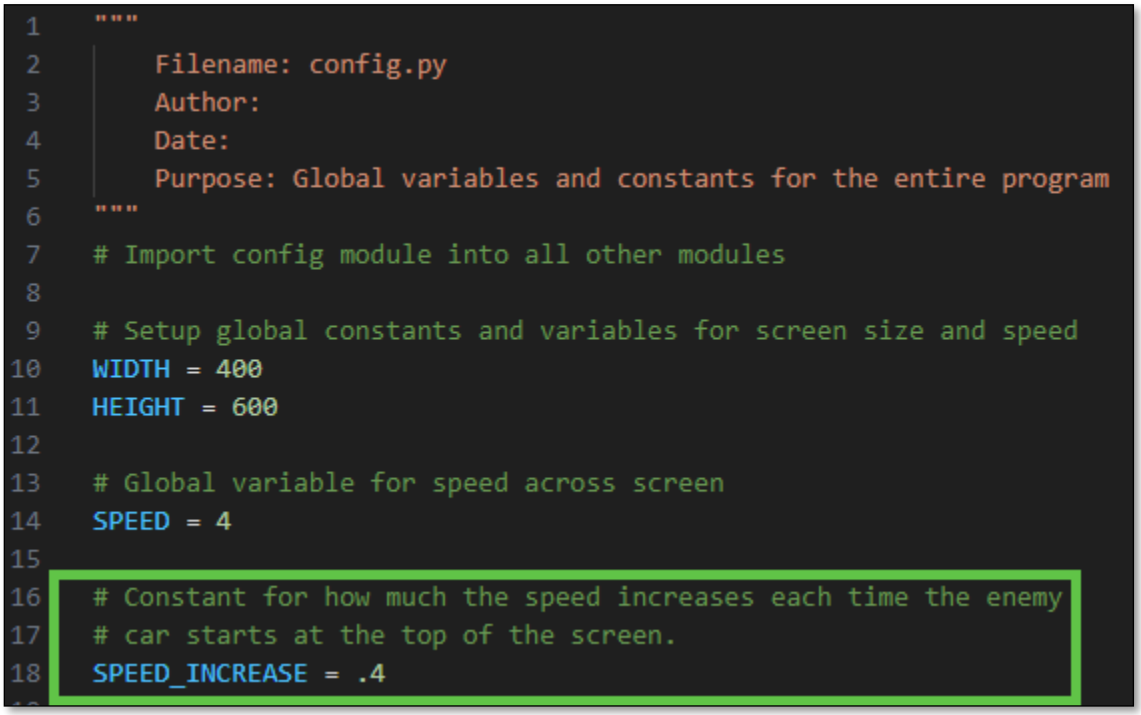
Our game is still incomplete. There's no fun in playing a game with the same thing happening over and over again. There is no end point, no variation in the game difficulty and most importantly, there are no consequences of colliding with the enemy.

In this section we're going to cover Collision Detection, user events and some other minor features.

The player and enemy classes will not change.

config.py

Add a speed increase constant to the config.py file. This will be how much the speed of the enemy car increase with each pass. You can increase or decrease this if you wish.



```
"""
    Filename: config.py
    Author:
    Date:
    Purpose: Global variables and constants for the entire program
"""
# Import config module into all other modules

# Setup global constants and variables for screen size and speed
WIDTH = 400
HEIGHT = 600

# Global variable for speed across screen
SPEED = 4

# Constant for how much the speed increases each time the enemy
# car starts at the top of the screen.
SPEED_INCREASE = .4
```

enemy.py

The speed of the enemy car will increase each times it starts at the top of the screen.

1. Define the initial speed.

```
13  class Enemy(pygame.sprite.Sprite):
14      """Define the enemy class and methods"""
15  # ------------------------ INITIALIZE ENEMY OBJECT ----------------------#
16
17      def __init__(self):
18          """Construct an enemy object from Sprite class"""
19
20          # Call the constructor of the superclass (pygame.sprite.Sprite)
21          super().__init__()
22
23          # Set initial speed of enemy car
24          self.speed = config.SPEED
25
```

2. Increase the speed each time the enemy car starts at the top of the window. The purpose of this code is to make the game more challenging as time passes.

```
42      # ------------------------ DRAW AND MOVE-----------------------------#
43      def update(self):
44          # Move the sprite down speed pixels at a time
45          self.rect.move_ip(0, self.speed)
46
47          # When the top of the sprite reaches the botton of the surface
48          if (self.rect.top > config.HEIGHT):
49              # Get a random location 40 pixels away from left and right.
50              x = randint(40, config.WIDTH - 40)
51
52              # Move car above the program window
53              y = -120
54
55              # Move car to beginning position
56              self.rect.center = (x, y)
57
58              # Increase speed each time the enemy car starts at the top
59              self.speed += config.SPEED_INCREASE
```

car_crash_6.py

We are going to use the sleep function to have a pause between when the game ends, and the program exits.

```
"""
    Filename: car_crash_6.py
    Author:
    Date:
    Purpose: Add collisions
"""
# pip install pygame-ce
# Import modules
import pygame
from sys import exit
from time import sleep
# Import our game classes
import config
import player
import enemy
```

Collision Detection

Add this code to the car_crash_6.py program.

```
# -------------------- CHECK COLLISION ----------------------------------#
    def check_collision(self):
        # If a collision occurs between Player and Enemy
        if pygame.sprite.spritecollideany(
            self.player_sprite,
            self.enemies
        ):
            sleep(1)
            # Quit Pygame
            pygame.quit()
            # Exit Python
            exit()
```

This section of code is related to collision detection in Python PyGame. Remember how we created groups earlier? You're about to see a massive benefit that we get from having meaningful groups.

The **spritecollideany()** function takes two parameters, the first must be a regular Sprite, like **player** or **enemy**. The second must be a Sprite group, such as **enemies** or **allSprites**. This function compares the sprite passed in the first parameter, to see if it's touching any of the sprites in the group passed in parameter two.

In our case, it checks to see whether our Player has collided with any of the sprites in the **enemies** group. The benefit of this function is that even if there are 1000 enemy sprites, we don't have to check collisions with them individually, and instead just use this function.

When the collision is True, we kill all the sprites using the **kill()** function, fill the screen with red, wait two seconds and close the entire program.

The last part to add is to have the game_loop call the check_collision() method

```
88  # ---------------------- GAME LOOP ---------------------------------------#
89      def game_loop(self):
90          """Start the infinite Game Loop"""
91          while True:
92              self.check_events()
93              self.check_collision()
94
95              # --------------- DRAW ON BACKBUFFER -------------------------#
96              # Draw everything on the backbuffer first
97              # Fill the surface with the background image loaded earlier
98              self.surface.blit(self.background, (0, 0))
```

When the enemy car collides with the player car, the game exits.

Assignment Submission

Zip up the program files folder and submit in Blackboard.