Part 2 – Pygame Car Crash Tutorial

Contents

Time required: 30 minutes

The Code

Open **car_crash.py** Change the code to the following. The minor changes are marked in green.

```
'''
    Name: car_crash.py
    Author:
    Date:
    Purpose: Draw both cars
'''

# Import modules
import pygame, sys
from pygame.locals import *
import player
import enemy

# Create a Player and Enemy object
player = player.Player()
enemy = enemy.Enemy()

class CarCrash:
    ''' Setup the object data fields '''
    # Setup color and screen size constants
    WHITE = (255, 255, 255)
    WIDTH = 400
    HEIGHT = 600
    # Constant for Frames Per Second (FPS)
    FPS = 60
    # Setup a computer clock object
    FramePerSec = pygame.time.Clock()
```

```
    def __init__(self):
        ''' Initialize the object '''
        # Initialize pygame for action
        pygame.init()

        # Create the game window, color and caption
        self.surface = pygame.display.set_mode((self.WIDTH, self.HEIGHT))
        self.surface.fill(self.WHITE)
        pygame.display.set_caption("Car Crash")

    def run_game(self):
        ''' Start the infinite Game Loop '''
        while True:
            # Closing the program by clicking the X
            # causes the QUIT event to be fired
            for event in pygame.event.get():

                # Exit game if window is closed
                if event.type == QUIT:
                    # Quit Pygame
                    pygame.quit()
                    # Exit Python
                    sys.exit()

            # Fill the surface with white to clear the screen
            self.surface.fill(self.WHITE)

            # Draw the sprites on the surface
            player.draw(self.surface)
            enemy.draw(self.surface)

            # Redraw the surface
            pygame.display.update()

            # How often our game loop executes
            self.FramePerSec.tick(self.FPS)

# Call the main function
if __name__ == '__main__':
    # Create game instance
    car_crash = CarCrash()
    # Start the game
    car_crash.run_game()
```

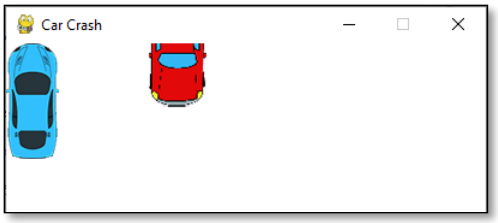
Enemy Class

The player class stayed the same. The enemy class is almost the same.

Open the **player.py** file and save it as **enemy.py**

```
'''
    Name: enemy.py
    Author:
    Date:
    Purpose: All logic for the player's car is in this class
'''
# Import modules
import pygame
from pygame.locals import *
import random

class Enemy(pygame.sprite.Sprite):
    ''' Define the enemy class and methods '''
    # Construct an Enemy object
    def __init__(self):

        # Construct an enemy object from Sprite class
        super().__init__()

        # Load an image from file
        self.image = pygame.image.load("enemy.png")

        # Create a surface rectangle the same size as the image
        self.surf = pygame.Surface((50, 80))

        # Create a rectangle with a random X location
        # Y is 0, the car starts at the top of the surface
        self.rect = self.surf.get_rect(center=(random.randint(40, 360), 0))

    # Draw the Enemy object on the surface
    def draw(self, surface):
        surface.blit(self.image, self.rect)

```

Example run:



This is how the game will look at this stage. The blue player car and the red enemy car are drawn on the screen. The enemy car will appear randomly on the X axis for each program run. Movement will be added later.

Classes and Objects

We added an Enemy class for to our program. The only difference between the Enemy and the Player class is that the Enemy class starts at a random x location.

```
# Define the enemy class and methods
class Enemy(pygame.sprite.Sprite):

    # Construct an Enemy object
    def __init__(self):

        # Construct an enemy object from Sprite class
        super().__init__()

        # Load an image from file
        self.image = pygame.image.load("enemy.png")

        # Create a surface rectangle the same size as the image
        self.surf = pygame.Surface((50, 80))

        # Create a rectangle with a random X location
        # Y is 0, the car starts at the top of the surface
        self.rect = self.surf.get_rect(center=(random.randint(40, 360), 0))

    # Draw the Enemy object on the surface
    def draw(self, surface):
        surface.blit(self.image, self.rect)
```

Modules, Variables, and the Surface

```
1  # Import modules
2  import pygame, sys
3  from pygame.locals import *
4  import random
```

At the top of the code the standard Pygame modules `pygame`, `sys`, and `from pygame.locals import *` are imported. `random` is imported as we want the enemy car to appear at random locations.

```
11 # Initialize pygame
12 pygame.init()
13
14 # Assign FPS (Frames per Second) a value
15 FPS = 60
16
17 # Color constants
18 WHITE = (255, 255, 255)
19
20 # Constants for screen size
21 SCREEN_WIDTH = 400
22 SCREEN_HEIGHT = 600
23
24 # Setup a computer clock object
25 FramePerSec = pygame.time.Clock()
```

We fsetup the `FPS` (frames per second) constant as 60. `FramPerSec` is setup as a computer clock object to ensure we get 60 frames per second according to the computer clock.

Color constants are setup using standard RGB (Red, Green, Blue) values for use later in the program. `SCREEN_WIDTH` and `SCREEN_HEIGHT` allow us to easily change the screen size if we wish.

```
# Create the game window, color and caption
surface = pygame.display.set_mode((SCREEN_WIDTH, SCREEN_HEIGHT))
surface.fill(WHITE)
pygame.display.set_caption("Car Crash")
```

This code sets up the display surface, clears the window by painting it with `WHITE`, and sets the caption.

Objects and the Game Loop

```
# Create game objects
player = Player()
enemy = Enemy()
```

This code creates a Player and an Enemy object from our classes. We could create as many unique objects as we wish. For this game, we only need on Player and one Enemy.

```
# Create game objects
player = Player()
enemy = Enemy()

# Game Loop
while True:

    # Exit the game
    for event in pygame.event.get():
        if event.type == QUIT:
            pygame.quit()
            sys.exit()

    # Fill the surface with white to clear the screen
    surface.fill(WHITE)

    # Draw game sprites on the surface
    player.draw(surface)
    enemy.draw(surface)

    # Update the screen
    pygame.display.update()

    # Game Loop 60 times a second
    FramePerSec.tick(FPS)
```

The commands shown above are all in the game loop, they repeat continuously.

1. Test for any Pygame events. If we close the program window, the program exits.
2. Refresh and clear the surface using the `surface.fill(WHITE)` function.
3. Call the Enemy and the player draw functions for both the Player and Enemy objects, drawing them to the surface.
4. The `pygame.display.update()` command updates the screen with all the commands that have occurred up-till this point.
5. The `tick()` method makes sure the Game Loop repeats only 60 times per second based on the computer clock.

Assignment Submission

Zip up the program files folder and submit in Blackboard.