

# PyGame Car Crash Tutorial - Part 6

## Contents

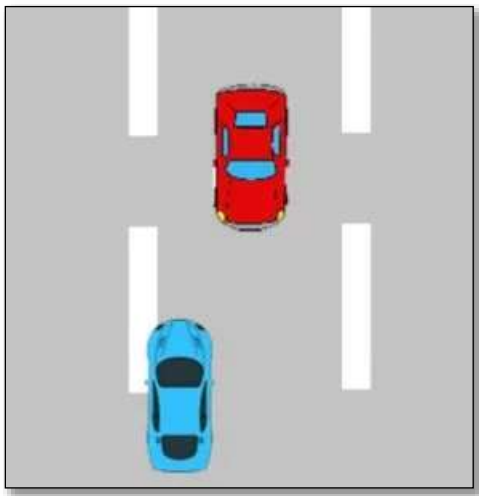
PyGame Car Crash Tutorial - Part 6 .....	1
Preview of the Game .....	1
config.py .....	2
enemy.py .....	2
car_crash_6.py .....	3
Collision Detection.....	4
Assignment Submission.....	5

Time required: 30 minutes

## Preview of the Game

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

[Car Crash Demo 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.

```
1  """
2      Filename: config.py
3      Author:
4      Date:
5      Purpose: Global variables and constants for the entire program
6  """
7  # Import config module into all other modules
8
9  # Setup global constants and variables for screen size and speed
10 WIDTH = 400
11 HEIGHT = 600
12
13 # Global variable for speed across screen
14 SPEED = 4
15
16 # Constant for how much the speed increases each time the enemy
17 # car starts at the top of the screen.
18 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 bottom 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.

```

1  """
2      Filename: car_crash_6.py
3      Author:
4      Date:
5      Purpose: Add collisions
6  """
7  # pip install pygame-ce
8  # Import modules
9  import pygame
10 from sys import exit
11 from time import sleep
12 # Import our game classes
13 import config
14 import player
15 import enemy

```

## Collision Detection

Add this code to the car\_crash\_6.py program.

```

75  # ----- CHECK COLLISION -----#
76  def check_collision(self):
77      # If a collision occurs between Player and Enemy
78      if pygame.sprite.spritecollideany(
79          self.player_sprite,
80          self.enemies
81      ):
82          sleep(1)
83          # Quit Pygame
84          pygame.quit()
85          # Exit Python
86          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

```
93      # ----- GAME LOOP ----- #
94      def game_loop(self):
95          """Start the infinite Game Loop"""
96          while True:
97              self.check_events()
98              self.check_collision()
99
100         # ----- DRAW ON SURFACE ----- #
101         # Draw everything on the surface first
102         # Fill the surface with the background image loaded earlier
103         self.surface.blit(self.background, (0, 0))
104
```

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

---

## Assignment Submission

1. Attach a screenshot showing the operation of the program.
2. Zip up the program files folder and submit in Blackboard.