[Coding Mind Academy](https://codingmindsacademy.com)

Coding in Python Pathway:

Advanced Python Programming II

Day 7: Shooting Game Basics

|  |
| --- |
|  |

# Installation and Setup (Students should have done this)

1. Highly Recommend Installing PyCharm Community Version for Free:

<https://www.jetbrains.com/pycharm/download>

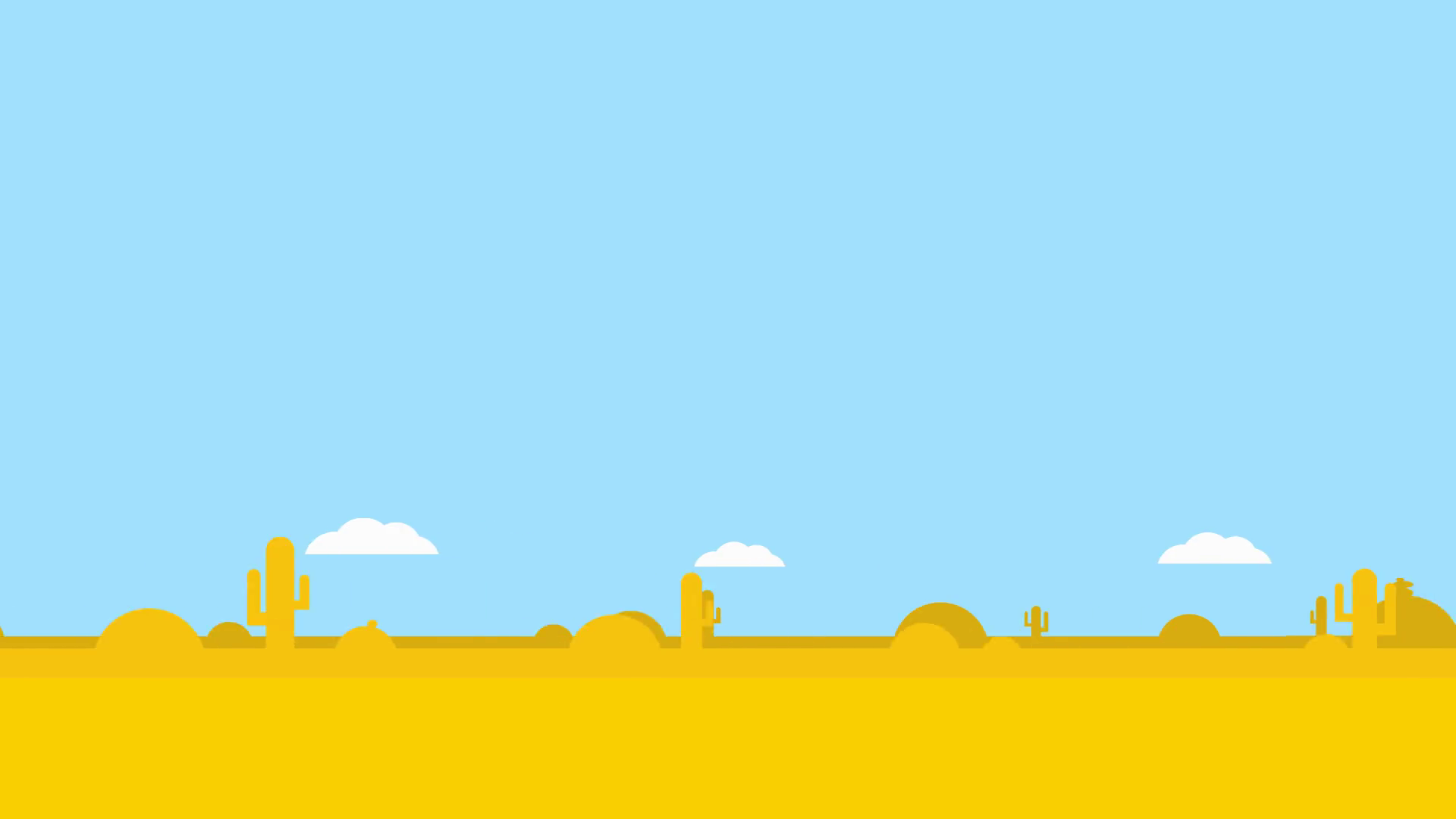
1. Install PyGame Library
   1. In PyCharm, it would be very easy to install. Otherwise, do **pip install pygame**
   2. For more help, please see: <https://www.pygame.org/wiki/GettingStarted>
2. **Note: Please use standard Python 3+ in PyCharm to run the game (Anaconda will not work correctly)**

# Basics of the Day

1. Today we are going to build a basic version of the shooting game. It has 3 things: the player, the missile you shoot, and the enemy. You control the player using arrow keys, and shoot using space key, and the enemy will move randomly.

2. Ask students to use Google Images to download 4 images (jpg or png) and copy them to the PyGame project folder:

1. A small image for your player, such as an icon of a plane;
2. A small image for your missle, such as an icon of a missle;
3. A small image for the enemy, such as an icon of a monster;
4. A bigger image for the background of the game.



3. The steps to run the lessons are as follows:

1. Pygame init and setup, import (done)
2. Set screen size (done)
3. Build a loop (done)
4. Handle quit event
5. Load the background image and render it
6. Scale the background image if the size is not right
7. Load the 3 other images
8. Render the player image
9. Implement the DOWN key press to move the player down
10. Let the students to figure out the other keys left, right, and up
11. Let’s work on the shooting.
12. Implement press SPACE key and show the missile.
13. It’s not working, and we need to bring a flag is\_shooting
14. Adjust the starting position of the missle

**import** pygame

**import** sys

pygame.init()

size = (1200, 800)

screen = pygame.display.set\_mode(size)

bg\_image = pygame.image.load(**"background.png"**)

bg\_image = pygame.transform.scale(bg\_image, (1200, 800))

bg\_image\_rect = bg\_image.get\_rect()

player\_image = pygame.image.load(**"plane.png"**)

player\_image = pygame.transform.scale(player\_image, (100, 100))

player\_image\_rect = player\_image.get\_rect()

ml\_image = pygame.image.load(**"missile.png"**)

ml\_image\_rect = ml\_image.get\_rect()

en\_image = pygame.image.load(**"enemy.png"**)

en\_image\_rect = en\_image.get\_rect()

is\_shooting = **False**

**while True**:

**for** event **in** pygame.event.get():

**if** event.type == pygame.QUIT:

sys.exit()

**if** event.type == pygame.KEYDOWN:

**if** event.key == pygame.K\_DOWN:

player\_image\_rect = player\_image\_rect.move([0, 10])

**if** event.type == pygame.KEYDOWN:

**if** event.key == pygame.K\_UP:

player\_image\_rect = player\_image\_rect.move([0, -10])

**if** event.type == pygame.KEYDOWN:

**if** event.key == pygame.K\_LEFT:

player\_image\_rect = player\_image\_rect.move([-10, 0])

**if** event.type == pygame.KEYDOWN:

**if** event.key == pygame.K\_RIGHT:

player\_image\_rect = player\_image\_rect.move([10, 0])

**if** event.type == pygame.KEYDOWN:

**if** event.key == pygame.K\_SPACE:

is\_shooting = **True**

ml\_image\_rect.centerx = player\_image\_rect.centerx + 100

ml\_image\_rect.centery = player\_image\_rect.centery

screen.blit(bg\_image, bg\_image\_rect)

screen.blit(player\_image, player\_image\_rect)

**if** is\_shooting:

ml\_image\_rect = ml\_image\_rect.move([10, 0])

screen.blit(ml\_image, ml\_image\_rect)

pygame.display.flip()

4. Next we will create a variable to check if the over or not, we called is is\_running we will put that as the outer if statement to control the flow of the game

**if** is\_running:

**if** is\_shooting:

ml\_image\_rect = ml\_image\_rect.move([10, 0])

screen.blit(ml\_image, ml\_image\_rect)

screen.blit(score, score\_rect)

pygame.display.flip()

5. Create two variables for size of screen to manage the placement of Rect objects on the screen

*size\_x = 1200*

*size\_y = 800*

6. Place them into the size object

size = (size\_x, size\_y)

7. We will use random library to place randomly show the enemy, also create addition two variables as upper limit and lower limit of the random place.

**import** time

**import** random

randomUpperBound = size\_y - 200

randomLowerBound = 50

We will also need to seed the random function in the beginning of the program

random.seed()

We know the enemy is showing up from the right so flying into the screen so will use two variables we created before to manipulate with the placement of the enemy object

en\_image\_rect.centerx = size\_x + 400

en\_image\_rect.centery = random.randrange(randomLowerBound, randomUpperBound)

8. Next we will create a bool variable for checking if there is enemy on screen

is\_enemy = **False**

We also want to generate enemy in duration, so we have to time it; record a starting time

start = time.process\_time() this is in float

showEnemyCounter = 0

Then we will give a speed based on the difference between, we need this is\_enemy variable because we want to move the enemy forward direction

showEnemyCounter = float(time.process\_time() - start)

**if** is\_enemy == **False and** showEnemyCounter > 0.2:

is\_enemy = **True**

start = time.process\_time()

So we will make the x direction -10 as it move toward the 0 and also show the en\_image\_rect -- the enemy -- the method is similar to how we show the missles

**if** is\_enemy == **True**:

en\_image\_rect.centerx = en\_image\_rect.centerx -10

en\_image\_rect.centery = en\_image\_rect.centery

screen.blit(en\_image, en\_image\_rect)

9. How do we know the enemy is fly out the screen? When the enemy\_rec’s x is reach 0, then we need to tell that is\_enemy = False, generated the new enmey

**if** en\_image\_rect.centerx == 0:

is\_enemy = **False**

en\_image\_rect.centerx = 1600

en\_image\_rect.centery = random.randrange(randomLowerBound, randomUpperBound);

10. Next we will check the collision on player and enemy, if the player collided with enmey

**if** player\_image\_rect.colliderect(en\_image\_rect):

is\_running = **False**

11. Next we will check the collision on missile and enemy, if the missile collided with enmey

We will set two objects location to be out of screen. Then redraw them

**if** en\_image\_rect.colliderect(ml\_image\_rect):

ml\_image\_rect.centerx = 1700

ml\_image\_rect.centery = 1700

en\_image\_rect.centerx = 1800

en\_image\_rect.centery = 1800

is\_enemy = **False**

is\_shooting = **False**

pygame.draw.rect(screen, red, ml\_image\_rect)

pygame.draw.rect(screen, red, en\_image\_rect)

12. Fire sound when missile is fired,

<https://freesound.org/people/sandyrb/sounds/35643/>

Create a variable to hold the sound

crash\_sound = pygame.mixer.Sound(**"bomb.wav"**)

When the space event is press fire the sound

**if** event.type == pygame.KEYDOWN:

**if** event.key == pygame.K\_SPACE:

is\_shooting = **True**

ml\_image\_rect.centerx = player\_image\_rect.centerx + 100

ml\_image\_rect.centery = player\_image\_rect.centery

pygame.mixer.Sound.play(crash\_sound)

13. Create a text area for showing score

Need to create a kills, variable to record scores.

kills = 0

red = 178,34,34

score = basicfont.render(**"Score: "** + str(kills), **True**, red)

score\_rect = score.get\_rect()

score\_rect.centerx = size\_x / 2

score\_rect.centery = size\_y - size\_y / 8

We will place them in the end prior to pygame render all the objects.

showEnemyCounter = float(time.process\_time() - start)

**if** is\_running:

**if** is\_enemy == **False and** showEnemyCounter > 0.2:

is\_enemy = **True**

start = time.process\_time()

**if** player\_image\_rect.colliderect(en\_image\_rect):

is\_running = **False**

**if** en\_image\_rect.colliderect(ml\_image\_rect):

ml\_image\_rect.centerx = 1700

ml\_image\_rect.centery = 1700

en\_image\_rect.centerx = 1800

en\_image\_rect.centery = 1800

is\_enemy = **False**

is\_shooting = **False**

kills+=1

score = basicfont.render(**"Score: "** + str(kills), **True**, red)

pygame.draw.rect(screen, red, ml\_image\_rect)

pygame.draw.rect(screen, red, en\_image\_rect)

**if** en\_image\_rect.centerx == 0:

is\_enemy = **False**

en\_image\_rect.centerx = 1600

en\_image\_rect.centery = random.randrange(randomLowerBound, randomUpperBound);

**if** is\_enemy == **True**:

en\_image\_rect.centerx = en\_image\_rect.centerx -10

en\_image\_rect.centery = en\_image\_rect.centery

screen.blit(en\_image, en\_image\_rect)

**if** is\_shooting:

ml\_image\_rect = ml\_image\_rect.move([10, 0])

screen.blit(ml\_image, ml\_image\_rect)

screen.blit(score, score\_rect)

pygame.display.flip()

14. Now we need to find a way to determine we win the game or lose

Create kill variable to determine how many enemy killed

kills = 0

Create is\_win variable to determine player win or not, we will set the a inner if inside the is\_runner to check if we win

**if** kills **is** 5:

is\_win = **True**

We will also display the winning message on the screen

**if is\_win:**

**display\_text = basicfont.render("YOU WIN", True, red)**

**screen.blit(display\_text, text\_rect)**

**is\_running = False**

We can also use the same text area to display the losing message on the screen

**if player\_image\_rect.colliderect(en\_image\_rect):**

**display\_text = basicfont.render("YOU LOST", True, red)**

**screen.blit(display\_text, text\_rect)**

**is\_running = False**

15. We need to recode the kills when the missiles hit the enemy

Create the text and its rect

score = basicfont.render(**"Score: "** + str(kills), **True**, red)

score\_rect = score.get\_rect()

score\_rect.centerx = size\_x / 2

score\_rect.centery = size\_y - size\_y / 8

Add the incrementing kill variable when the condition meeted

**if** en\_image\_rect.colliderect(ml\_image\_rect):

ml\_image\_rect.centerx = 1700

ml\_image\_rect.centery = 1700

en\_image\_rect.centerx = 1800

en\_image\_rect.centery = 1800

is\_enemy = **False**

is\_shooting = **False**

kills+=1

score = basicfont.render(**"Score: "** + str(kills), **True**, red)

pygame.draw.rect(screen, red, ml\_image\_rect)

pygame.draw.rect(screen, red, en\_image\_rect)

**Reference**

**import pygame**

**import sys**

**import time**

**import random**

**pygame.init()**

**random.seed()**

**is\_shooting = False**

**is\_enemy = False**

**is\_win = False**

**is\_running = True**

**red = 178,34,34**

**crash\_sound = pygame.mixer.Sound("bomb.wav")**

**size\_x = 1200, size\_y = 800**

**randomUpperBound = size\_y - 200, randomLowerBound = 50, kills = 0**

**size = (size\_x, size\_y)**

**screen = pygame.display.set\_mode(size)**

**bg\_image = pygame.image.load("background.png")**

**bg\_image = pygame.transform.scale(bg\_image, (size\_x, size\_y))**

**bg\_image\_rect = bg\_image.get\_rect()**

**player\_image = pygame.image.load("plane.png")**

**player\_image = pygame.transform.scale(player\_image, (90, 90))**

**player\_image\_rect = player\_image.get\_rect()**

**ml\_image = pygame.image.load("missile.png")**

**ml\_image = pygame.transform.scale(ml\_image, (30, 10))**

**ml\_image\_rect = ml\_image.get\_rect()**

**en\_image = pygame.image.load("enemy.png")**

**en\_image = pygame.transform.scale(en\_image, (90, 90))**

**en\_image\_rect = en\_image.get\_rect()**

***# Fonts***

**basicfont = pygame.font.SysFont(None, 100)**

**display\_text = basicfont.render("", True, red)**

**text\_rect = display\_text.get\_rect()**

**text\_rect.centerx = size\_x / 3**

**text\_rect.centery = size\_y / 2**

**score = basicfont.render("Score: " + str(kills), True, red)**

**score\_rect = score.get\_rect()**

**score\_rect.centerx = size\_x / 2**

**score\_rect.centery = size\_y - size\_y / 8**

**start = time.process\_time()**

**showEnemyCounter = 0**

***# from right***

**en\_image\_rect.centerx = size\_x + 400**

***# top is zero***

**en\_image\_rect.centery = random.randrange(randomLowerBound, randomUpperBound)**

**while True:**

**for event in pygame.event.get():**

**if event.type == pygame.QUIT:**

**sys.exit()**

**if event.type == pygame.KEYDOWN:**

**if event.key == pygame.K\_DOWN:**

**player\_image\_rect = player\_image\_rect.move([0, 10])**

**if event.type == pygame.KEYDOWN:**

**if event.key == pygame.K\_UP:**

**player\_image\_rect = player\_image\_rect.move([0, -10])**

**if event.type == pygame.KEYDOWN:**

**if event.key == pygame.K\_LEFT:**

**player\_image\_rect = player\_image\_rect.move([-10, 0])**

**if event.type == pygame.KEYDOWN:**

**if event.key == pygame.K\_RIGHT:**

**player\_image\_rect = player\_image\_rect.move([10, 0])**

**if event.type == pygame.KEYDOWN:**

**if event.key == pygame.K\_SPACE:**

**is\_shooting = True**

**ml\_image\_rect.centerx = player\_image\_rect.centerx + 100**

**ml\_image\_rect.centery = player\_image\_rect.centery**

**pygame.mixer.Sound.play(crash\_sound)**

**screen.blit(bg\_image, bg\_image\_rect)**

**screen.blit(player\_image, player\_image\_rect)**

**showEnemyCounter = float(time.process\_time() - start)**

**if is\_running:**

**if kills is 5:**

**is\_win = True**

**if is\_enemy == False and showEnemyCounter > 0.2:**

**is\_enemy = True**

**start = time.process\_time()**

**if player\_image\_rect.colliderect(en\_image\_rect):**

**display\_text = basicfont.render("YOU LOST", True, red)**

**screen.blit(display\_text, text\_rect)**

**is\_running = False**

**if en\_image\_rect.colliderect(ml\_image\_rect):**

**ml\_image\_rect.centerx = 1700**

**ml\_image\_rect.centery = 1700**

**en\_image\_rect.centerx = 1800**

**en\_image\_rect.centery = 1800**

**is\_enemy = False**

**is\_shooting = False**

**kills+=1**

**score = basicfont.render("Score: " + str(kills), True, red)**

**pygame.draw.rect(screen, red, ml\_image\_rect)**

**pygame.draw.rect(screen, red, en\_image\_rect)**

**if en\_image\_rect.centerx == 0:**

**is\_enemy = False**

**en\_image\_rect.centerx = 1600**

**en\_image\_rect.centery = random.randrange(randomLowerBound, randomUpperBound);**

**if is\_enemy == True:**

**en\_image\_rect.centerx = en\_image\_rect.centerx -10**

**en\_image\_rect.centery = en\_image\_rect.centery**

**screen.blit(en\_image, en\_image\_rect)**

**if is\_shooting:**

**ml\_image\_rect = ml\_image\_rect.move([10, 0])**

**screen.blit(ml\_image, ml\_image\_rect)**

**if is\_win:**

**display\_text = basicfont.render("YOU WIN", True, red)**

**screen.blit(display\_text, text\_rect)**

**is\_running = False**

**screen.blit(score, score\_rect)**

**pygame.display.flip()**