Code Library · Stage 6

Six tiny programs about ENEMIES, spawning, recycling, and collisions. Copy, paste, tweak.

First, a tiny bit of background

Three new tricks appear in this lesson:

- **Loop to create many.** Instead of five copy-paste lines, write for i in range(5): and build all five aliens in one go.
- **random.randint(a, b)** gives a random whole number from a to b. Use NEGATIVE y values so enemies start above the visible window and slide in from space.
- **Recycle, don't remove.** When an enemy reaches the bottom, just send it back to the top with a new random x. The list stays the same size, but the action never stops.

```
# the heart of Stage 6
enemies = []
for i in range(5):                        # build many in one loop
    enemies.append(pygame.Rect(random.randint(0, WIDTH - 50),
                               random.randint(-300, 0),
                               50, 50))
for e in enemies:                         # each frame: move + recycle
    e.y += enemy_speed
    if e.y > HEIGHT:
        e.y = random.randint(-200, -50)
        e.x = random.randint(0, WIDTH - 50)
```

Sneak peek: every Rect has a .colliderect() method that returns True if two rectangles overlap. That's how we detect a bullet hitting an enemy. We'll use it in Game 5 and Game 6.

How to use this handout

Each game has three parts: **what it does**, **the code**, and **challenges**. Copy any code block into the Trinket editor on the lesson page, press the green ■ **Run** button, then change a number, a colour, a speed - see what happens.

Game 1 · One Falling Enemy

What it does

A single red square that drifts down forever (off the bottom of the screen, eventually). The building block - no list, no recycle, no random.

```
import pygame

pygame.init()
WIDTH, HEIGHT = 600, 400
screen = pygame.display.set_mode((WIDTH, HEIGHT))
pygame.display.set_caption("One Falling Enemy")

RED   = (230, 60, 60)
SPACE = (0, 0, 50)

# one enemy starts at the top, drifts down
enemy = pygame.Rect(270, 0, 50, 50)
enemy_speed = 3

running = True
while running:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False

    enemy.y += enemy_speed       # slide down

    screen.fill(SPACE)
    pygame.draw.rect(screen, RED, enemy)
    pygame.display.flip()

pygame.quit()
```

Try these one by one

- **Fall 1** Make it crawl: change enemy_speed to 1.
- **Fall 2** Make it dive: enemy_speed = 12.
- **Fall 3** Move it to start at the top-LEFT instead of centred.

```
enemy = pygame.Rect(0, 0, 50, 50)
```

Game 2 · Recycle the Enemy

What it does

Same enemy as Game 1, but when it leaves the bottom of the screen, we teleport it back to the top at a NEW random x. Watch it fall forever from different places. This is the recycling pattern.

```
import pygame
import random

pygame.init()
WIDTH, HEIGHT = 600, 400
screen = pygame.display.set_mode((WIDTH, HEIGHT))
pygame.display.set_caption("Recycle the Enemy")

RED   = (230, 60, 60)
SPACE = (0, 0, 50)

enemy = pygame.Rect(random.randint(0, WIDTH - 50), 0, 50, 50)
enemy_speed = 3

running = True
while running:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False

    enemy.y += enemy_speed

    # when it falls off the bottom, send it back to the top
    if enemy.y > HEIGHT:
        enemy.y = random.randint(-100, 0)
        enemy.x = random.randint(0, WIDTH - 50)

    screen.fill(SPACE)
    pygame.draw.rect(screen, RED, enemy)
    pygame.display.flip()

pygame.quit()
```

Try these one by one

Recycle 1 Make it appear ABOVE the screen each time (random.randint(-300, -100)) so it slides in from space.

```
enemy.y = random.randint(-300, -100)
```

Recycle 2 Speed up the enemy every time it recycles by 1.

```
enemy_speed += 1
```

Recycle 3 Pick a random colour each recycle: define colours = [...] and use random.choice(colours) inside the if-block.

Game 3 · Five Enemies (loop to create many)

What it does

Now five aliens at once. The for-loop in the setup builds all of them in one go - change range(5) to range(50) and you have 50 enemies without writing more code.

```
import pygame
import random

pygame.init()
WIDTH, HEIGHT = 800, 600
screen = pygame.display.set_mode((WIDTH, HEIGHT))
pygame.display.set_caption("Five Enemies (loop to create many)")
clock = pygame.time.Clock()

RED   = (230, 60, 60)
SPACE = (0, 0, 50)

# build all 5 enemies in one loop instead of five copy-pasted lines
enemies = []
for i in range(5):
    e = pygame.Rect(random.randint(0, WIDTH - 50),
                    random.randint(-300, 0),
                    50, 50)
    enemies.append(e)
enemy_speed = 2

running = True
while running:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False

    # move + recycle all five
    for e in enemies:
        e.y += enemy_speed
        if e.y > HEIGHT:
            e.y = random.randint(-200, -50)
            e.x = random.randint(0, WIDTH - 50)

    screen.fill(SPACE)
    for e in enemies:
        pygame.draw.rect(screen, RED, e)
    pygame.display.flip()
    clock.tick(60)

pygame.quit()
```

Try these one by one

Five 1 Make it 20 enemies.

```
for i in range(20):
```

Five 2 Change the alien size to a random width and height.

```
pygame.Rect(random.randint(0, WIDTH - 50), random.randint(-300, 0), random.randint(20, 80), random
```

Five 3 Slow them down so they all drift gently: enemy_speed = 1.

Game 4 · Asteroid Field

What it does

15 asteroids - each with its OWN random colour and OWN random speed. We store [rect, colour, speed] for each one instead of just the rect. Same pattern as confetti from Lesson 5.

```
import pygame
import random

pygame.init()
WIDTH, HEIGHT = 800, 600
screen = pygame.display.set_mode((WIDTH, HEIGHT))
pygame.display.set_caption("Asteroid Field")
clock = pygame.time.Clock()

SPACE = (0, 0, 50)
colours = [(230, 60, 60), (250, 160, 50), (200, 130, 90), (180, 180, 180)]

# 15 asteroids, each with its OWN colour and speed
asteroids = []
for i in range(15):
    rect = pygame.Rect(random.randint(0, WIDTH - 40),
                       random.randint(-600, 0),
                       random.randint(20, 50),
                       random.randint(20, 50))
    colour = random.choice(colours)
    speed  = random.randint(2, 6)
    asteroids.append([rect, colour, speed])

running = True
while running:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False

    for a in asteroids:
        a[0].y += a[2]                          # move at its own speed
        if a[0].y > HEIGHT:
            a[0].y = random.randint(-200, -50)
            a[0].x = random.randint(0, WIDTH - 50)

    screen.fill(SPACE)
    for rect, colour, speed in asteroids:
        pygame.draw.rect(screen, colour, rect)
    pygame.display.flip()
    clock.tick(60)

pygame.quit()
```

Try these one by one

Field 1 100 asteroids!

```
for i in range(100):
```

Field 2 Speed range from 1 to 12 - some crawl, some fly.

```
speed = random.randint(1, 12)
```

Field 3 Add a black background and white asteroids for a snow effect.

```
screen.fill((0,0,0))
colour = (255, 255, 255)
```

Game 5 · Touch Detector (meet colliderect)

What it does

Move your mouse to control a green square. An enemy falls from the top. When your square touches the enemy, your square turns RED. That magic test is cursor.colliderect(enemy) - one line, returns True or False.

```
import pygame
import random

pygame.init()
WIDTH, HEIGHT = 600, 400
screen = pygame.display.set_mode((WIDTH, HEIGHT))
pygame.display.set_caption("Touch Detector - dodge the enemy!")

RED   = (230, 60, 60)
GREEN = (60, 220, 100)
SPACE = (0, 0, 50)

enemy = pygame.Rect(random.randint(0, WIDTH - 50),
                    random.randint(-200, 0), 50, 50)
enemy_speed = 3

running = True
touching = False
while running:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False

    enemy.y += enemy_speed
    if enemy.y > HEIGHT:
        enemy.y = random.randint(-200, -50)
        enemy.x = random.randint(0, WIDTH - 50)

    # follow the mouse with a small green square
    mx, my = pygame.mouse.get_pos()
    cursor = pygame.Rect(mx - 15, my - 15, 30, 30)

    # the secret weapon: colliderect()
    touching = cursor.colliderect(enemy)

    screen.fill(SPACE)
    pygame.draw.rect(screen, RED, enemy)
    pygame.draw.rect(screen, RED if touching else GREEN, cursor)
    pygame.display.flip()

pygame.quit()
```

Try these one by one

Touch 1 Add a print every time you start touching: if touching: print("hit!").

Touch 2 Make your cursor smaller (15x15) - harder to hit, easier to dodge.

```
cursor = pygame.Rect(mx - 7, my - 7, 15, 15)
```

Touch 3 Add a SECOND falling enemy and check collision for both.

Game 6 · Mini Defender (everything together)

What it does

A complete mini-shooter - player, bullets, enemies, recycling, collisions. No score or game-over yet (those are Lessons 7 and 8), but everything you've learned is here in one program. This is what your Galaxy Defender ALREADY looks like.

```
import pygame
import random

pygame.init()
WIDTH, HEIGHT = 800, 600
screen = pygame.display.set_mode((WIDTH, HEIGHT))
pygame.display.set_caption("Mini Defender - move with LEFT/RIGHT, shoot with SPACE")
clock = pygame.time.Clock()

GREEN  = (0, 220, 100)
YELLOW = (255, 215, 0)
RED    = (255, 0, 0)
SPACE  = (0, 0, 50)

player = pygame.Rect(370, 520, 60, 40)
bullets = []
enemies = []
for i in range(5):
    enemies.append(pygame.Rect(random.randint(0, WIDTH - 50),
                               random.randint(-300, 0), 50, 50))

running = True
while running:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False
        if event.type == pygame.KEYDOWN and event.key == pygame.K_SPACE:
            bullets.append(pygame.Rect(player.x + 27, player.y, 6, 15))

    keys = pygame.key.get_pressed()
    if keys[pygame.K_LEFT]  and player.x > 0:                player.x -= 5
    if keys[pygame.K_RIGHT] and player.x < WIDTH - 60:       player.x += 5

    # move bullets
    for b in bullets[:]:
        b.y -= 10
        if b.y < 0: bullets.remove(b)

    # recycle enemies
    for e in enemies:
        e.y += 2
        if e.y > HEIGHT:
            e.y = random.randint(-200, -50)
            e.x = random.randint(0, WIDTH - 50)

    # bullet vs enemy
    for b in bullets[:]:
        for e in enemies:
            if b.colliderect(e):
                if b in bullets: bullets.remove(b)
                e.y = random.randint(-200, -50)
                e.x = random.randint(0, WIDTH - 50)

    screen.fill(SPACE)
    pygame.draw.rect(screen, GREEN, player)
    for b in bullets:  pygame.draw.rect(screen, YELLOW, b)
    for e in enemies:  pygame.draw.rect(screen, RED, e)
    pygame.display.flip()
    clock.tick(60)

pygame.quit()
```

Try these one by one

Mini 1 Make the player blue and twice as wide.

```
BLUE = (0, 130, 255)
player = pygame.Rect(340, 520, 120, 40)
```

Mini 2 10 enemies instead of 5.

```
for i in range(10):
```

Mini 3 Print the message "Boom!" every time a bullet hits an enemy.

```
print("Boom!")
```

Where this takes us next

You now have a real shooter without a score! Lesson 7 adds **score and lives** - a number that counts up when you hit an alien, and three little hearts that go down when an alien escapes past you. After that, only Game States (8), Helpers (9), and the Game Loop (10) are left.

Challenge: pick Game 6, change five things (colours, speeds, count, sizes, key bindings), screenshot the result and share it with someone. That's YOUR Galaxy Defender, not mine!