Code Library · Stage 5

Six tiny programs about BULLETS, LISTS, and one-shot key presses. Copy, paste, tweak.

First, a tiny bit of background

A **list** is one variable that holds many items in order - like a shopping basket. You start empty, you add items with append, you loop through with for, you remove with remove. That's all there is to it.

The other new idea: **KEYDOWN**. In Lesson 4 we used key.get_pressed() because we wanted the player to keep moving while you held the arrow. For shooting we want **one bullet per press** - hold SPACE down, still only fires once. That's what KEYDOWN inside the event loop gives us.

```
# the heart of Stage 5
bullets = []                                # empty list
bullets.append(pygame.Rect(x, y, 6, 15))    # add a new bullet
for b in bullets[:]:                        # loop safely (copy!)
    b.y -= 8                                # move it up
    if b.y < 0:
        bullets.remove(b)                   # tidy up off-screen ones
```

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 key - see what happens.

Game 1 · One Bullet (no list)

What it does

A single yellow bullet that flies upward forever. No list, no firing - just one moving rectangle. This is the building block.

```
import pygame

pygame.init()
screen = pygame.display.set_mode((600, 400))
pygame.display.set_caption("One Bullet")

YELLOW = (255, 215, 0)
SPACE  = (0, 0, 50)

# just ONE bullet, no list. It starts near the bottom.
bullet = pygame.Rect(290, 350, 20, 30)
bullet_speed = 8

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

    bullet.y -= bullet_speed   # slide it upward

    screen.fill(SPACE)
    pygame.draw.rect(screen, YELLOW, bullet)
    pygame.display.flip()

pygame.quit()
```

Try these one by one

One 1 Make it crawl: change bullet_speed to 1.

One 2 Make it red and tiny.

```
RED = (255, 0, 0)
bullet = pygame.Rect(290, 350, 4, 8)
```

One 3 When it flies off the top, send it back to the bottom.

```
if bullet.y < 0: bullet.y = 350
```

Game 2 · Press SPACE to Fire (introducing lists)

What it does

A player at the bottom. Press SPACE - a yellow bullet appears at the top of the player and flies upward. Press again - another one. Look at all those bullets piling up - we'll clean them in Game 3.

```
import pygame

pygame.init()
screen = pygame.display.set_mode((600, 400))
pygame.display.set_caption("Press SPACE to Fire")

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

player = pygame.Rect(270, 340, 60, 40)
bullets = []          # <-- the magic: an empty list

running = True
while running:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False
        # ONE bullet per SPACE press (KEYDOWN, not get_pressed)
        if event.type == pygame.KEYDOWN and event.key == pygame.K_SPACE:
            bullets.append(pygame.Rect(player.x + 27, player.y, 6, 15))

    # move every bullet up
    for b in bullets:
        b.y -= 8

    screen.fill(SPACE)
    pygame.draw.rect(screen, GREEN, player)
    for b in bullets:
        pygame.draw.rect(screen, YELLOW, b)
    pygame.display.flip()

pygame.quit()
```

Try these one by one

Fire 1 Add LEFT/RIGHT player movement using keys = pygame.key.get_pressed() (from Lesson 4).

Fire 2 Print the length of the bullets list after every press.

```
print("Bullets in air:", len(bullets))
```

Fire 3 Use the F key instead of SPACE.

```
if event.key == pygame.K_f:
```

Game 3 · Disappearing Bullets (the safe-loop trick)

What it does

Same as Game 2 + bullets vanish when they leave the top of the screen. We loop over bullets[:] (a copy) so we can safely .remove(b) from the real list. Crucial pattern - you'll use it again and again.

```
import pygame

pygame.init()
WIDTH, HEIGHT = 600, 400
screen = pygame.display.set_mode((WIDTH, HEIGHT))
pygame.display.set_caption("Disappearing Bullets")

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

player = pygame.Rect(270, 340, 60, 40)
bullets = []

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))

    # the safe way to remove items while looping: copy with [:]
    for b in bullets[:]:
        b.y -= 8
        if b.y < 0:
            bullets.remove(b)

    screen.fill(SPACE)
    pygame.draw.rect(screen, GREEN, player)
    for b in bullets:
        pygame.draw.rect(screen, YELLOW, b)
    pygame.display.flip()

pygame.quit()
```

Try these one by one

Vanish 1 What happens if you forget the [:] and write 'for b in bullets:'? Try it - Python may behave strangely. (This is why we copy.)

Vanish 2 Make bullets vanish slightly EARLIER - at y < 50 instead of y < 0.

```
if b.y < 50: bullets.remove(b)
```

Vanish 3 Give every bullet width 20 and height 5 - lasers!

```
bullets.append(pygame.Rect(player.x + 20, player.y, 20, 5))
```

Game 4 · Bullet Rain

What it does

Press SPACE and 10 blue drops appear at the top, falling down. The same list pattern as bullets - except the y goes UP each frame, and we use random.randint to spread them across the screen.

```
import pygame
import random

pygame.init()
WIDTH, HEIGHT = 600, 400
screen = pygame.display.set_mode((WIDTH, HEIGHT))
pygame.display.set_caption("Bullet Rain - press SPACE to add drops")
clock = pygame.time.Clock()

BLUE  = (120, 180, 255)
SPACE = (0, 0, 50)

drops = []

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:
            for _ in range(10):                        # add 10 drops at once
                x = random.randint(0, WIDTH - 4)
                drops.append(pygame.Rect(x, 0, 4, 18))

    for d in drops[:]:
        d.y += 6                                       # falling down
        if d.y > HEIGHT:
            drops.remove(d)

    screen.fill(SPACE)
    for d in drops:
        pygame.draw.rect(screen, BLUE, d)
    pygame.display.flip()
    clock.tick(60)

pygame.quit()
```

Try these one by one

Rain 1 Make it pour: spawn 50 drops per press.

```
for _ in range(50):
```

Rain 2 Slower drops: change d.y += 6 to d.y += 2.

Rain 3 Random colours per drop: import random, then random.choice([(120, 180, 255), (200, 200, 255), (60, 200, 255)]).

Game 5 · Click to Shoot

What it does

No player, just a screen. Click anywhere - a bullet spawns at the mouse click and flies up. Combines Lesson 2's MOUSEBUTTONDOWN with this lesson's bullet list.

```
import pygame

pygame.init()
WIDTH, HEIGHT = 600, 400
screen = pygame.display.set_mode((WIDTH, HEIGHT))
pygame.display.set_caption("Click to Shoot")

YELLOW = (255, 215, 0)
SPACE  = (0, 0, 50)

bullets = []

running = True
while running:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False
        # bullet spawns at the mouse click, then flies up
        if event.type == pygame.MOUSEBUTTONDOWN:
            mx, my = event.pos
            bullets.append(pygame.Rect(mx - 3, my, 6, 15))

    for b in bullets[:]:
        b.y -= 8
        if b.y < 0:
            bullets.remove(b)

    screen.fill(SPACE)
    for b in bullets:
        pygame.draw.rect(screen, YELLOW, b)
    pygame.display.flip()

pygame.quit()
```

Try these one by one

Click 1 Make the bullet fly DOWN instead of up.

```
b.y += 8
```

Click 2 Make the bullet go LEFT toward the screen edge.

```
b.x -= 8
```

Click 3 Change the bullet to a circle: replace draw.rect with pygame.draw.circle(screen, YELLOW, b.center, 6).

Game 6 · Confetti Cannon (bonus)

What it does

Press SPACE - 40 random-coloured pieces of confetti shoot upward and outward from the bottom centre, then slowly fall back down with gravity. Same list pattern, but each item carries its own colour and direction.

```
import pygame
import random

pygame.init()
WIDTH, HEIGHT = 600, 400
screen = pygame.display.set_mode((WIDTH, HEIGHT))
pygame.display.set_caption("Confetti Cannon - press SPACE")
clock = pygame.time.Clock()

SPACE = (0, 0, 50)

# each piece of confetti = (rect, colour, dx, dy)
pieces = []

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:
            for _ in range(40):
                r = pygame.Rect(WIDTH // 2, HEIGHT - 20, 6, 6)
                colour = (random.randint(80, 255),
                          random.randint(80, 255),
                          random.randint(80, 255))
                dx = random.randint(-6, 6)
                dy = random.randint(-12, -4)
                pieces.append([r, colour, dx, dy])

    for piece in pieces[:]:
        r, colour, dx, dy = piece
        r.x += dx
        r.y += dy
        piece[3] += 1     # gravity: pull dy down a bit each frame
        if r.y > HEIGHT:
            pieces.remove(piece)

    screen.fill(SPACE)
    for r, colour, dx, dy in pieces:
        pygame.draw.rect(screen, colour, r)
    pygame.display.flip()
    clock.tick(60)

pygame.quit()
```

Try these one by one

Confetti 1 Make the cannon shoot 200 pieces per press!

```
for _ in range(200):
```

Confetti 2 No gravity: delete the line piece[3] += 1. They fly forever in straight lines.

Confetti 3 Confetti only goes in two colours - pick your favourites and use random.choice([COLOUR_A, COLOUR_B]).

Where bullets take us next

You can now manage many things at once with a list. In Lesson 6 we'll create ENEMIES - another list, full of red rectangles falling from the top of the screen. Same pattern, brand new feel.

Challenge: combine Game 3 (firing bullets up) with Game 4 (drops falling down). When a bullet and a drop touch, remove BOTH from their lists. That's how every shooter game starts!