```
import asyncio
import random
import pygame

W, H = 360, 640
GROUND = H - 50
BIRD_X = 90
R = 14               # bird size
PIPE_W = 60
GAP = 165            # hole between pipes
GRAVITY = 1400
FLAP = -420

pygame.init()
screen = pygame.display.set_mode((W, H))
pygame.display.set_caption('Flappy Ball')
clock = pygame.time.Clock()
font = pygame.font.Font(None, 34)
big = pygame.font.Font(None, 64)

state = {}


def random_gap():
    return random.randint(GAP // 2 + 60, GROUND - GAP // 2 - 60)


def reset():
    state.update(y=H / 2, vy=0.0, score=0, mode='ready',
                 pipes=[{'x': W + 80 + i * 220, 'gap': random_gap(), 'passed': False}
                        for i in range(3)])
    state.setdefault('best', 0)


def flap():
    if state['mode'] == 'over':
        reset()
    elif state['mode'] == 'ready':
        state['mode'] = 'play'
        state['vy'] = FLAP
    else:
        state['vy'] = FLAP


def circle_hits_rect(cx, cy, r, rx, ry, rw, rh):
    nearest_x = max(rx, min(cx, rx + rw))
    nearest_y = max(ry, min(cy, ry + rh))
    return (cx - nearest_x) ** 2 + (cy - nearest_y) ** 2 < r * r


def say(text, f, color, x, y):
```

```
    img = f.render(text, True, color)
    screen.blit(img, img.get_rect(center=(x, y)))


def game_over():
    state['mode'] = 'over'
    state['best'] = max(state['best'], state['score'])


def update(dt):
    s = state
    s['vy'] += GRAVITY * dt
    s['y'] += s['vy'] * dt

    speed = min(150 + s['score'] * 5, 260)
    for p in s['pipes']:
        p['x'] -= speed * dt

        if p['x'] < -PIPE_W:                        # pipe left the screen
            p['x'] = max(q['x'] for q in s['pipes']) + 220
            p['gap'] = random_gap()
            p['passed'] = False

        if not p['passed'] and p['x'] + PIPE_W < BIRD_X:
            p['passed'] = True
            s['score'] += 1

        top_h = p['gap'] - GAP // 2
        bottom_y = p['gap'] + GAP // 2
        if (circle_hits_rect(BIRD_X, s['y'], R, p['x'], 0, PIPE_W, top_h) or
                circle_hits_rect(BIRD_X, s['y'], R, p['x'], bottom_y, PIPE_W, GROUND - bottom_y)):
            game_over()

    if s['y'] + R > GROUND or s['y'] - R < 0:
        game_over()


def draw():
    s = state
    screen.fill((135, 206, 235))
    for cx, cy in ((60, 90), (250, 150), (150, 260)):      # clouds
        pygame.draw.ellipse(screen, (255, 255, 255), (cx, cy, 80, 30))

    for p in s['pipes']:
        top_h = p['gap'] - GAP // 2
        bottom_y = p['gap'] + GAP // 2
        pygame.draw.rect(screen, (60, 170, 70), (p['x'], 0, PIPE_W, top_h))
        pygame.draw.rect(screen, (60, 170, 70), (p['x'], bottom_y, PIPE_W, GROUND - bottom_y))
        pygame.draw.rect(screen, (40, 130, 50), (p['x'] - 4, top_h - 20, PIPE_W + 8, 20))
        pygame.draw.rect(screen, (40, 130, 50), (p['x'] - 4, bottom_y, PIPE_W + 8, 20))

    pygame.draw.rect(screen, (150, 110, 60), (0, GROUND, W, H - GROUND))
    pygame.draw.rect(screen, (90, 190, 70), (0, GROUND, W, 10))

    by = int(s['y'])
```

```
    pygame.draw.circle(screen, (255, 217, 0), (BIRD_X, by), R)
    pygame.draw.circle(screen, (255, 255, 255), (BIRD_X + 5, by - 4), 5)
    pygame.draw.circle(screen, (0, 0, 0), (BIRD_X + 6, by - 4), 2)
    pygame.draw.polygon(screen, (255, 120, 0),
                        [(BIRD_X + 12, by), (BIRD_X + 22, by + 3), (BIRD_X + 12, by + 6)])

    say(str(s['score']), big, (255, 255, 255), W // 2, 50)
    if s['mode'] == 'ready':
        say('Tap to start', font, (0, 0, 0), W // 2, H // 2 + 60)
    elif s['mode'] == 'over':
        say('GAME OVER', big, (200, 0, 0), W // 2, H // 2 - 40)
        say(f"Score: {s['score']}   Best: {s['best']}", font, (0, 0, 0), W // 2, H // 2 + 5)
        say('Tap to play again', font, (0, 0, 0), W // 2, H // 2 + 45)


async def main():
    reset()
    last_tap = 0
    running = True
    while running:
        dt = clock.tick(60) / 1000

        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                running = False
            elif event.type in (pygame.MOUSEBUTTONDOWN, pygame.FINGERDOWN) or \
                    (event.type == pygame.KEYDOWN and event.key == pygame.K_SPACE):
                now = pygame.time.get_ticks()
                if now - last_tap > 150:     # ignore double events from one touch
                    last_tap = now
                    flap()

        if state['mode'] == 'play':
            update(dt)

        draw()
        pygame.display.flip()
        await asyncio.sleep(0)  # needed so the game also runs in the browser

    pygame.quit()


asyncio.run(main())
```