Coderingen per project

Hier vind je de complete Python-codering voor elk project in dit boek, behalve voor de hacks en tweaks. Werken je projecten niet goed, controleer je scripts dan zorgvuldig aan de hand van de coderingen hieronder.

Schiet het fruit (blz. 48)

```
from random import randint
appel = Actor('apple')

def draw():
    screen.clear()
    appel.draw()

def plaats_appel():
    appel.x = randint(10, 800)
    appel.y = randint(10, 600)

def on_mouse_down(pos):
    if appel.collidepoint(pos):
        print('Goed schot!')
        plaats_appel()
    else:
        print('Je hebt gemist!')
        exit()

plaats_appel()
```

Muntenverzamelaar (blz. 58)

```
from random import randint

WIDTH = 400
HEIGHT = 400
score = 0
game_over = False

vos = Actor('fox')
vos.pos = 100, 100

munt = Actor('coin')
munt.pos = 200, 200
```

```
def draw():
 screen.fill('green')
 vos.draw()
 munt.draw()
 screen.draw.text('Score: ' + str(score), color='black', topleft=(10, 10))

    if game_over:
        screen.fill('pink')
        screen.draw.text('Eindscore: ' + str(score), topleft=(10, 10), fontsize=60)

def plaats_munt():
    munt.x = randint(20, (WIDTH - 20))
    munt.y = randint(20, (HEIGHT - 20))

def tijd_om():
    global game_over
    game_over = True

def update():
    global score

    if keyboard.left:
        vos.x = vos.x - 2
    elif keyboard.right:
        vos.x = vos.x + 2
    elif keyboard.up:
        vos.y = vos.y - 2
    elif keyboard.down:
        vos.y = vos.y + 2

    munt_verzameld = vos.colliderect(munt)

    if munt_verzameld:
        score = score + 10
        plaats_munt()

clock.schedule(tijd_om, 7.0)
plaats_munt()
```

Volg de getallen (blz. 68)

```
from random import randint

WIDTH = 400
HEIGHT = 400
```

```
stippen = []
lijnen = []

volgende_stip = 0

for stip in range(0, 10):
    acteur = Actor('dot')
    acteur.pos = randint(20, WIDTH - 20), \
        randint(20, HEIGHT - 20)
    stippen.append(acteur)

def draw():
    screen.fill('black')
    getal = 1
    for stip in stippen:
        screen.draw.text(str(getal), \
                         (stip.pos[0], stip.pos[1] + 12))
        stip.draw()
        getal = getal + 1
    for lijn in lijnen:
        screen.draw.line(lijn[0], lijn[1], (100, 0, 0))

def on_mouse_down(pos):
    global volgende_stip
    global lijnen
    if stippen[volgende_stip].collidepoint(pos):
        if volgende_stip:
            lijnen.append((stippen[volgende_stip - 1].pos, stippen[volgende_stip].pos))
        volgende_stip = volgende_stip + 1
    else:
        lijnen = []
        volgende_stip = 0
```

Rood alarm (blz. 80)

```
import random

FONT_KLEUR = (255, 255, 255)
WIDTH = 800
HEIGHT = 600
MIDDEN_X = WIDTH / 2
MIDDEN_Y = HEIGHT / 2
MIDDEN = (MIDDEN_X, MIDDEN_Y)
LAATSTE_LEVEL = 6
START_SNELHEID = 10
KLEUREN = ['green', 'blue']
```

```
game_over = False
game_voltooid = False
huidig_level = 1
sterren = []
animaties = []

def draw():
    global sterren, huidig_level, game_over, game_voltooid
    screen.clear()
    screen.blit('space', (0, 0))
    if game_over:
        toon_boodschap('GAME OVER!', 'Probeer nogmaals.')
    elif game_voltooid:
        toon_boodschap('JE HEBT GEWONNEN!', 'GOED GEDAAN.')
    else:
        for ster in sterren:
            ster.draw()

def update():
    global sterren
    if len(sterren) == 0:
        sterren = maak_nieuwe_sterren(huidig_level)

def maak_nieuwe_sterren(aantal_extra_sterren):
    kleuren_te_maken = krijg_kleuren_te_maken(aantal_extra_sterren)
    nieuwe_sterren = maak_sterren(kleuren_te_maken)
    layout_sterren(nieuwe_sterren)
    animeer_sterren(nieuwe_sterren)
    return nieuwe_sterren

def krijg_kleuren_te_maken(aantal_extra_sterren):
    kleuren_te_maken = ['red']
    for i in range(0, aantal_extra_sterren):
        random_kleur = random.choice(KLEUREN)
        kleuren_te_maken.append(random_kleur)
    return kleuren_te_maken

def maak_sterren(kleuren_te_maken):
    nieuwe_sterren = []
    for kleur in kleuren_te_maken:
        ster = Actor(kleur + '-star')
        nieuwe_sterren.append(ster)
    return nieuwe_sterren

def layout_sterren(sterren_layout):
    aantal_openingen = len(sterren_layout) + 1
    opening_grootte = WIDTH / aantal_openingen
```

```
    random.shuffle(sterren_layout)
    for index, ster in enumerate(sterren_layout):
        nieuw_x_pos = (index + 1) * opening_grootte
        ster.x = nieuw_x_pos

def animeer_sterren(sterren_te_animeren):
    for ster in sterren_te_animeren:
        duur = START_SNELHEID - huidig_level
        ster.anchor = ('center', 'bottom')
        animatie = animate(ster, duration=duur, on_finished=handle_game_over, y=HEIGHT)
        animaties.append(animatie)

def handle_game_over():
    global game_over
    game_over = True

def on_mouse_down(pos):
    global sterren, huidig_level
    for ster in sterren:
        if ster.collidepoint(pos):
            if 'red' in ster.image:
                red_ster_klik()
            else:
                handle_game_over()

def red_ster_klik():
    global huidig_level, sterren, animaties, game_voltooid
    stop_animaties(animaties)
    if huidig_level == LAATSTE_LEVEL:
        game_voltooid = True
    else:
        huidig_level = huidig_level + 1
        sterren = []
        animaties = []

def stop_animaties(animaties_stoppen):
    for animatie in animaties_stoppen:
        if animatie.running:
            animatie.stop()

def toon_boodschap(titeltekst, subtiteltekst):
    screen.draw.text(titeltekst, fontsize=60, center=MIDDEN, color=FONT_KLEUR)
    screen.draw.text(subtiteltekst,
                     fontsize=30,
                     center=(MIDDEN_X, MIDDEN_Y + 30),
                     color=FONT_KLEUR)
```

Grote quiz (blz. 98)

```
WIDTH = 1280
HEIGHT = 720

hoofd_vak = Rect(0, 0, 820, 240)
timer_vak = Rect(0, 0, 240, 240)
antwoord_vak1 = Rect(0, 0, 495, 165)
antwoord_vak2 = Rect(0, 0, 495, 165)
antwoord_vak3 = Rect(0, 0, 495, 165)
antwoord_vak4 = Rect(0, 0, 495, 165)

hoofd_vak.move_ip(50, 40)
timer_vak.move_ip(990, 40)
antwoord_vak1.move_ip(50, 358)
antwoord_vak2.move_ip(735, 358)
antwoord_vak3.move_ip(50, 538)
antwoord_vak4.move_ip(735, 538)
antwoord_vakken = [antwoord_vak1, antwoord_vak2, antwoord_vak3, antwoord_vak4]

score = 0
tijd_over = 10

v1 = ['Wat is de hoofdstad van Frankrijk?',
      'Londen', 'Parijs', 'Berlijn', 'Tokyo', 2]

v2 = ['Wat is 5+7?',
      '12', '10', '14', '8', 1]

v3 = ['Wat is de zevende maand van het jaar?',
      'April', 'Mei', 'Juni', 'Juli', 4]

v4 = ['Welke planeet staat het dichtst bij de zon?',
      'Saturnus', 'Neptunus', 'Mercurius', 'Venus', 3]

v5 = ['Waar liggen de piramiden?',
      'India', 'Egypte', 'Marokko', 'Canada', 2]

vragen = [v1, v2, v3, v4, v5]
vraag = vragen.pop(0)

def draw():
    screen.fill('dim grey')
    screen.draw.filled_rect(hoofd_vak, 'sky blue')
    screen.draw.filled_rect(timer_vak, 'sky blue')

    for vak in antwoord_vakken:
        screen.draw.filled_rect(vak, 'orange')
```

```
    screen.draw.textbox(str(tijd_over), timer_vak, color=('black'))
    screen.draw.textbox(vraag[0], hoofd_vak, color=('black'))

    index = 1
    for vak in antwoord_vakken:
        screen.draw.textbox(vraag[index], vak, color=('black'))
        index = index + 1

def game_over():
    global vraag, tijd_over
    boodschap = 'Game over. Je hebt %s vragen juist' % str(score)
    vraag = [boodschap, '-', '-', '-', '-', 5]
    tijd_over = 0

def juist_antwoord():
    global vraag, score, tijd_over

    score = score + 1
    if vragen:
        vraag = vragen.pop(0)
        tijd_over = 10
    else:
        print('Einde van de vragen')
        game_over()

def on_mouse_down(pos):
    index = 1
    for vak in antwoord_vakken:
        if vak.collidepoint(pos):
            print('Geklikt op antwoord' + str(index))
            if index == vraag[5]:
                print('Je hebt het goed!')
                juist_antwoord()
            else:
                game_over()
        index = index + 1

def update_tijd_over():
    global tijd_over

    if tijd_over:
        tijd_over = tijd_over - 1
    else:
        game_over()

clock.schedule_interval(update_tijd_over, 1.0)
```

Ballonvaart (blz. 116)

```
from random import randint

WIDTH = 800
HEIGHT = 600

ballon = Actor('balloon')
ballon.pos = 400, 300

vogel = Actor('bird-up')
vogel.pos = randint(800, 1600), randint(10, 200)

huis = Actor('house')
huis.pos = randint(800, 1600), 460

boom = Actor('tree')
boom.pos = randint(800, 1600), 450

vogel_omhoog = True
omhoog = False
game_over = False
score = 0
aantal_updates = 0

scores = []

def update_hoogste_scores():
    global score, scores
    bestandsnaam = r'/Users/bharti/Desktop/python-games/ballonvaart/hoogste-scores.txt'
    scores = []
    with open(bestandsnaam, 'r') as bestand:
        lijn = bestand.readline()
        hoogste_scores = lijn.split()
        for hoogste_scores in hoogste_scores:
            if(score > int(hoogste_scores)):
                scores.append(str(score) + ' ')
                score = int(hoogste_scores)
            else:
                scores.append(str(hoogste_scores) + ' ')
    with open(bestandsnaam, 'w') as bestand:
        for hoogste_scores in scores:
            bestand.write(hoogste_scores)

def toon_hoogste_scores():
    screen.draw.text('HOOGSTE SCORES', (350, 150), color='black')
    y = 175
    positie = 1
```

Onthoud dat je deze grijze code moet aanpassen aan de plaats waar high-scores.txt file op je computer staat.

```
    for hoogste_score in scores:
        screen.draw.text(str(positie) + '. ' + hoogste_score, (350, y), color='black')
        y += 25
        positie += 1

def draw():
    screen.blit('background', (0, 0))
    if not game_over:
        ballon.draw()
        vogel.draw()
        huis.draw()
        boom.draw()
        screen.draw.text('Score: ' + str(score), (700, 5), color='black')
    else:
       toon_hoogste_scores()

def on_mouse_down():
    global omhoog
    omhoog = True
    ballon.y -= 50

def on_mouse_up():
    global omhoog
    omhoog = False

def fladder():
    global vogel_omhoog
    if vogel_omhoog:
        vogel.image = 'bird-down'
        vogel_omhoog = False
    else:
        vogel.image = 'bird-up'
        vogel_omhoog = True

def update():
    global game_over, score, aantal_updates
    if not game_over:
        if not omhoog:
            ballon.y += 1

        if vogel.x > 0:
            vogel.x -= 4
            if aantal_updates == 9:
                fladder()
                aantal_updates = 0
            else:
                aantal_updates += 1
```

```
        else:
            vogel.x = randint(800, 1600)
            vogel.y = randint(10, 200)
            score += 1
            aantal_updates = 0

        if huis.right > 0:
            huis.x -= 2
        else:
            huis.x = randint(800, 1600)
            score += 1

        if boom.right > 0:
            boom.x -= 2
        else:
            boom.x = randint(800, 1600)
            score += 1

        if ballon.top < 0 or ballon.bottom > 560:
            game_over = True
            update_hoogste_scores()

        if ballon.collidepoint(vogel.x, vogel.y) or \
           ballon.collidepoint(huis.x, huis.y) or \
           ballon.collidepoint(boom.x, boom.y):
                game_over = True
                update_hoogste_scores()
```

Dansfestijn (blz. 136)

```
from random import randint

WIDTH = 800
HEIGHT = 600
MIDDEN_X = WIDTH / 2
MIDDEN_Y = HEIGHT / 2

beweging_lijst = []
toon_lijst = []

score = 0
huidige_beweging = 0
tellen = 4
dans_lengte = 4

zeg_dans = False
toon_aftellen = True
```

```
bewegingen_voltooid = False
game_over = False

danser = Actor('dancer-start')
danser.pos = MIDDEN_X + 5, MIDDEN_Y - 40

omhoog = Actor('up')
omhoog.pos = MIDDEN_X, MIDDEN_Y + 110
rechts = Actor('right')
rechts.pos = MIDDEN_X + 60, MIDDEN_Y + 170
omlaag = Actor('down')
omlaag.pos = MIDDEN_X, MIDDEN_Y + 230
links = Actor('left')
links.pos = MIDDEN_X - 60, MIDDEN_Y + 170

def draw():
    global game_over, score, zeg_dans
    global tellen, toon_aftellen
    if not game_over:
        screen.clear()
        screen.blit('stage', (0, 0))
        danser.draw()
        omhoog.draw()
        omlaag.draw()
        rechts.draw()
        links.draw()
        screen.draw.text('Score: ' +
                         str(score), color='black',
                         topleft=(10, 10))
        if zeg_dans:
            screen.draw.text('Dans!', color='black',
                             topleft=(MIDDEN_X - 65, 150), fontsize=60)
        if toon_aftellen:
            screen.draw.text(str(tellen), color='black',
                             topleft=(MIDDEN_X - 8, 150), fontsize=60)
    else:
        screen.clear()
        screen.blit('stage', (0, 0))
        screen.draw.text('Score: ' +
                         str(score), color='black',
                         topleft=(10, 10))
        screen.draw.text('GAME OVER!', color='black',
                         topleft=(MIDDEN_X - 130, 220), fontsize=60)
    return

def reset_danser():
    global game_over
```

```
    if not game_over:
        danser.image = 'dancer-start'
        omhoog.image = 'up'
        rechts.image = 'right'
        omlaag.image = 'down'
        links.image = 'left'
    return

def update_danser(beweging):
    global game_over
    if not game_over:
        if beweging == 0:
            omhoog.image = 'up-lit'
            danser.image = 'dancer-up'
            clock.schedule(reset_danser, 0.5)
        elif beweging == 1:
            rechts.image = 'right-lit'
            danser.image = 'dancer-right'
            clock.schedule(reset_danser, 0.5)
        elif beweging == 2:
            omlaag.image = 'down-lit'
            danser.image = 'dancer-down'
            clock.schedule(reset_danser, 0.5)
        else:
            links.image = 'left-lit'
            danser.image = 'dancer-left'
            clock.schedule(reset_danser, 0.5)
    return

def toon_bewegingen():
    global beweging_lijst, toon_lijst, dans_lengte
    global zeg_dans, toon_aftellen, huidige_beweging
    if toon_lijst:
        deze_beweging = toon_lijst[0]
        toon_lijst = toon_lijst[1:]
        if deze_beweging == 0:
            update_danser(0)
            clock.schedule(toon_bewegingen, 1)
        elif deze_beweging == 1:
            update_danser(1)
            clock.schedule(toon_bewegingen, 1)
        elif deze_beweging == 2:
            update_danser(2)
            clock.schedule(toon_bewegingen, 1)
        else:
            update_danser(3)
            clock.schedule(toon_bewegingen, 1)
```

```
    else:
        zeg_dans = True
        toon_aftellen = False
    return

def aftellen():
    global tellen, game_over, toon_aftellen
    if tellen > 1:
        tellen = tellen - 1
        clock.schedule(aftellen, 1)
    else:
        toon_aftellen = False
        toon_bewegingen()
    return

def genereer_bewegingen():
    global beweging_lijst, dans_lengte, tellen
    global toon_aftellen, zeg_dans
    tellen = 4
    beweging_lijst = []
    zeg_dans = False
    for beweging in range(0, dans_lengte):
        rand_beweging = randint(0, 3)
        beweging_lijst.append(rand_beweging)
        toon_lijst.append(rand_beweging)
    toon_aftellen = True
    aftellen()
    return

def volgende_beweging():
    global dans_lengte, huidige_beweging, bewegingen_voltooid
    if huidige_beweging < dans_lengte - 1:
        huidige_beweging = huidige_beweging + 1
    else:
        bewegingen_voltooid = True
    return

def on_key_up(key):
    global score, game_over, beweging_lijst, huidige_beweging
    if key == keys.UP:
        update_danser(0)
        if beweging_lijst[huidige_beweging] == 0:
            score = score + 1
            volgende_beweging()
        else:
            game_over = True
    elif key == keys.RIGHT:
```

```
        update_danser(1)
        if beweging_lijst[huidige_beweging] == 1:
            score = score + 1
            volgende_beweging()
        else:
            game_over = True
    elif key == keys.DOWN:
        update_danser(2)
        if beweging_lijst[huidige_beweging] == 2:
            score = score + 1
            volgende_beweging()
        else:
            game_over = True
    elif key == keys.LEFT:
        update_danser(3)
        if beweging_lijst[huidige_beweging] == 3:
            score = score + 1
            volgende_beweging()
        else:
            game_over = True
    return

genereer_bewegingen()
music.play('vanishing-horizon')

def update():
    global game_over, huidige_beweging, bewegingen_voltooid
    if not game_over:
        if bewegingen_voltooid:
            genereer_bewegingen()
            bewegingen_voltooid = False
            huidige_beweging = 0
    else:
        music.stop()
```

Blije tuin (blz. 154)

```
from random import randint
import time

WIDTH = 800
HEIGHT = 600
MIDDEN_X = WIDTH / 2
MIDDEN_Y = HEIGHT / 2

game_over = False
beeindigd = False
```

```
tuin_blij = True
vangbloem_raken = False

tijd_verstreken = 0
start_tijd = time.time()

koe = Actor('cow')
koe.pos = 100, 500

bloem_lijst = []
verwelkt_lijst = []
vangbloem_lijst = []
vangbloem_vy_lijst = []
vangbloem_vx_lijst = []

def draw():
    global game_over, tijd_verstreken, beeindigd
    if not game_over:
        screen.clear()
        screen.blit('garden', (0, 0))
        koe.draw()
        for bloem in bloem_lijst:
            bloem.draw()
        for vangbloem in vangbloem_lijst:
            vangbloem.draw()
        tijd_verstreken = int(time.time() - start_tijd)
        screen.draw.text(
            'Tuin blij gedurende: ' +
            str(tijd_verstreken) + ' seconden',
            topleft=(10, 10), color='black'
        )
    else:
        if not beeindigd:
            koe.draw()
            screen.draw.text(
                'Tuin blij gedurende: ' +
                str(tijd_verstreken) + ' seconden',
                topleft=(10, 10), color='black'
            )
            if (not tuin_blij):
                screen.draw.text(
                    'TUIN NIET BLIJ - GAME OVER!', color='black',
                    topleft=(10, 50)
                )
                beeindigd = True
            else:
                screen.draw.text(
```

```
                'VANGBLOEM AANVAL - GAME OVER!', color='black',
                topleft=(10, 50)
            )
            beeindigd = True
    return

def nieuwe_bloem():
    global bloem_lijst, verwelkt_lijst
    bloem_nieuw = Actor('flower')
    bloem_nieuw.pos = randint(50, WIDTH - 50), randint(150, HEIGHT - 100)
    bloem_lijst.append(bloem_nieuw)
    verwelkt_lijst.append('blij')
    return

def toevoegen_bloemen():
    global game_over
    if not game_over:
        nieuwe_bloem()
        clock.schedule(toevoegen_bloemen, 4)
    return

def check_verwelk_tijden():
    global verwelkt_lijst, game_over, tuin_blij
    if verwelkt_lijst:
        for verwelkt_sinds in verwelkt_lijst:
            if not verwelkt_sinds == 'blij':
                tijd_verwelkt = int(time.time() - verwelkt_sinds)
                if (tijd_verwelkt) > 10.0:
                    tuin_blij = False
                    game_over = True
                    break
    return

def verwelk_bloem ():
    global bloem_lijst, verwelkt_lijst, game_over
    if not game_over:
        if bloem_lijst:
            rand_bloem = randint(0, len(bloem_lijst) - 1)
            if (bloem_lijst[rand_bloem].image == 'flower'):
                bloem_lijst[rand_bloem].image = 'flower-wilt'
                verwelkt_lijst[rand_bloem] = time.time()
        clock.schedule(verwelk_bloem, 3)
    return

def check_bloem_raken():
    global koe, bloem_lijst, verwelkt_lijst
    index = 0
```

```
    for bloem in bloem_lijst:
        if (bloem.colliderect(koe) and
                bloem.image == 'flower-wilt'):
            bloem.image = 'flower'
            verwelkt_lijst[index] = 'blij'
            break
        index = index + 1
    return

def check_vangbloem_raken():
    global koe, vangbloem_lijst, vangbloem_raken
    global game_over
    for vangbloem in vangbloem_lijst:
        if vangbloem.colliderect(koe):
            koe.image = 'zap'
            game_over = True
            break
    return

def snelheid():
    random_richting = randint(0, 1)
    random_snelheid = randint(2, 3)
    if random_richting == 0:
        return -random_snelheid
    else:
        return random_snelheid

def muteer():
    global bloem_lijst, vangbloem_lijst, vangbloem_vy_lijst
    global vangbloem_vx_lijst, game_over
    if not game_over and bloem_lijst:
        rand_bloem = randint(0, len(bloem_lijst) - 1)
        vangbloem_pos_x = bloem_lijst[rand_bloem].x
        vangbloem_pos_y = bloem_lijst[rand_bloem].y
        del bloem_lijst[rand_bloem]
        vangbloem = Actor('fangflower')
        vangbloem.pos = vangbloem_pos_x, vangbloem_pos_y
        vangbloem_vx = snelheid()
        vangbloem_vy = snelheid()
        vangbloem = vangbloem_lijst.append(vangbloem)
        vangbloem_vx_lijst.append(vangbloem_vx)
        vangbloem_vy_lijst.append(vangbloem_vy)
        clock.schedule(muteer, 20)
    return

def update_vangbloemen():
    global vangbloem_lijst, game_over
```

```
    if not game_over:
        index = 0
        for vangbloem in vangbloem_lijst:
            vangbloem_vx = vangbloem_vx_lijst[index]
            vangbloem_vy = vangbloem_vy_lijst[index]
            vangbloem.x = vangbloem.x + vangbloem_vx
            vangbloem.y = vangbloem.y + vangbloem_vy
            if vangbloem.left < 0:
                vangbloem_vx_lijst[index] = -vangbloem_vx
            if vangbloem.right > WIDTH:
                vangbloem_vx_lijst[index] = -vangbloem_vx
            if vangbloem.top < 150:
                vangbloem_vy_lijst[index] = -vangbloem_vy
            if vangbloem.bottom > HEIGHT:
                vangbloem_vy_lijst[index] = -vangbloem_vy
            index = index + 1
    return

def reset_koe():
    global game_over
    if not game_over:
        koe.image = 'cow'
    return

toevoegen_bloemen()
verwelk_bloem()

def update():
    global score, game_over, vangbloem_raken
    global bloem_lijst, vangbloem_lijst, tijd_verstreken
    vangbloem_raken = check_vangbloem_raken()
    check_verwelk_tijden()
    if not game_over:
        if keyboard.space:
            koe.image = 'cow-water'
            clock.schedule(reset_koe, 0.5)
            check_bloem_raken()
        if keyboard.left and koe.x > 0:
            koe.x -= 5
        elif keyboard.right and koe.x < WIDTH:
            koe.x += 5
        elif keyboard.up and koe.y > 150:
            koe.y -= 5
        elif keyboard.down and koe.y < HEIGHT:
            koe.y += 5
        if tijd_verstreken > 15 and not vangbloem_lijst:
            muteer()
        update_vangbloemen()
```

Slapende draken (blz. 176)

```
import math

WIDTH = 800
HEIGHT = 600
MIDDEN_X = WIDTH / 2
MIDDEN_Y = HEIGHT / 2
MIDDEN = (MIDDEN_X, MIDDEN_Y)
FONT_KLEUR = (0, 0, 0)
EIEREN_DOEL = 20
HELD_START = (200, 300)
AANVAL_AFSTAND = 200
DRAAK_WEKTIJD = 2
EIEREN_VERBERGTIJD = 2
BEWEEG_AFSTAND = 5

levens = 3
eieren_verzameld = 0
game_over = False
game_voltooid = False
reset_vereist = False

makkelijke_grot = {
    'draak': Actor('dragon-asleep', pos=(600, 100)),
    'eieren': Actor('one-egg', pos=(400, 100)),
    'ei_tellen': 1,
    'ei_verborgen': False,
    'ei_verberg_teller': 0,
    'slaap_lengte': 10,
    'slaap_teller': 0,
    'wakker_teller': 0
}

medium_grot = {
    'draak': Actor('dragon-asleep', pos=(600, 300)),
    'eieren': Actor('two-eggs', pos=(400, 300)),
    'ei_tellen': 2,
    'ei_verborgen': False,
    'ei_verberg_teller': 0,
    'slaap_lengte': 7,
    'slaap_teller': 0,
    'wakker_teller': 0
}

moeilijke_grot = {
    'draak': Actor('dragon-asleep', pos=(600, 500)),
    'eieren': Actor('three-eggs', pos=(400, 500)),
```

```
    'ei_tellen': 3,
    'ei_verborgen': False,
    'ei_verberg_teller': 0,
    'slaap_lengte': 4,
    'slaap_teller': 0,
    'wakker_teller': 0
}

grotten = [makkelijke_grot, medium_grot, moeilijke_grot]
held = Actor('hero', pos=HELD_START)

def draw():
    global grotten, eieren_verzameld, levens, game_voltooid
    screen.clear()
    screen.blit('dungeon', (0, 0))
    if game_over:
        screen.draw.text('GAME OVER!', fontsize=60, center=MIDDEN, color=FONT_KLEUR)
    elif game_voltooid:
        screen.draw.text('JE HEBT GEWONNEN!', fontsize=60, center=MIDDEN, color=FONT_KLEUR)
    else:
        held.draw()
        teken_grotten(grotten)
        teken_tellers(eieren_verzameld, levens)

def teken_grotten(grotten_te_tekenen):
    for grot in grotten_te_tekenen:
        grot['draak'].draw()
        if grot['ei_verborgen'] is False:
            grot['eieren'].draw()

def teken_tellers(eieren_verzameld, levens):
    screen.blit('egg-count', (0, HEIGHT - 30))
    screen.draw.text(str(eieren_verzameld),
                     fontsize=40,
                     pos=(30, HEIGHT - 30),
                     color=FONT_KLEUR)
    screen.blit('life-count', (60, HEIGHT - 30))
    screen.draw.text(str(levens),
                     fontsize=40,
                     pos=(90, HEIGHT - 30),
                     color=FONT_KLEUR)
```

```
def update():
    if keyboard.right:
        held.x += BEWEEG_AFSTAND
        if held.x > WIDTH:
            held.x = WIDTH
    elif keyboard.left:
        held.x -= BEWEEG_AFSTAND
        if held.x < 0:
            held.x = 0
    elif keyboard.down:
        held.y += BEWEEG_AFSTAND
        if held.y > HEIGHT:
            held.y = HEIGHT
    elif keyboard.up:
        held.y -= BEWEEG_AFSTAND
        if held.y < 0:
            held.y = 0
    check_voor_raken()

def update_grotten():
    global grotten, held, levens
    for grot in grotten:
        if grot['draak'].image == 'dragon-asleep':
            update_slapende_draak(grot)
        elif grot['draak'].image == 'dragon-awake':
            update_wakkere_draak(grot)
        update_ei(grot)

clock.schedule_interval(update_grotten, 1)

def update_slapende_draak(grot) :
    if grot['slaap_teller'] >= grot['slaap_lengte']:
        grot['draak'].image = 'dragon-awake'
        grot['slaap_teller'] = 0
    else:
        grot['slaap_teller'] += 1

def update_wakkere_draak(grot) :
    if grot['wakker_teller'] >= DRAAK_WEKTIJD:
        grot['draak'].image = 'dragon-asleep'
        grot['wakker_teller'] = 0
    else:
        grot['wakker_teller'] += 1

def update_ei(grot):
    if grot['ei_verborgen'] is True:
        if grot['ei_verberg_teller'] >= EIEREN_VERBERGTIJD:
```

```
            grot['ei_verborgen'] = False
            grot['ei_verberg_teller'] = 0
        else:
            grot['ei_verberg_teller'] += 1

def check_voor_raken():
    global grotten, eieren_verzameld, levens, reset_vereist, game_voltooid
    for grot in grotten:
        if grot['ei_verborgen'] is False:
            check_voor_ei_raken(grot)
        if grot['draak'].image == 'dragon-awake' and reset_vereist is False:
            check_voor_draak_raken(grot)

def check_voor_draak_raken(grot):
    x_afstand = held.x - grot['draak'].x
    y_afstand = held.y - grot['draak'].y
    afstand = math.hypot(x_afstand, y_afstand)
    if afstand < AANVAL_AFSTAND:
        handle_draak_raken()

def handle_draak_raken():
    global reset_vereist
    reset_vereist = True
    animate(held, pos=HELD_START, on_finished=verminder_leven)

def check_voor_ei_raken(grot):
    global eieren_verzameld, game_voltooid
    if held.colliderect(grot['eieren']):
        grot['ei_verborgen'] = True
        eieren_verzameld += grot['ei_tellen']
        if eieren_verzameld >= EIEREN_DOEL:
            game_voltooid = True

def verminder_leven():
    global levens, reset_vereist, game_over
    levens -= 1
    if levens == 0:
        game_over = True
    reset_vereist = False
```