5.7 Ko'rsichqon oyini yasash - Keng qamrovli

Va nihoyat, sizni ushbu kitobning so‘nggi misoliga yetib kelganingiz bilan sizni tabriklayman. Oldingi misollarni muvaffaqiyatli o‘zlashtirib, shu bosqichga yetib kelganingizda, Python kodlashning zavqini ham asta-sekin his qila boshlagan bo‘lsangiz kerak. Kutilganidan ko‘ra matnli kodlash juda qiyin yoki zerikarli bo‘lmaganiga ishonaman. Agar shunday bo'lsa, siz bu kitobning asosiy maqsadi bo‘lgan matnli kodlash olamiga kirish vazifasini to‘liq bajarganingizni hisoblasangiz bo‘ladi.

So‘nggi misol hozirgacha bo‘lgan misollarning umumiy ko‘rinishi bo‘lib, oldingilariga qaraganda biroz murakkabroq. Ammo, vaqt ajratib, bosqichma-bosqich tushunib borsangiz, buni ham uddalaysiz.

{% code lineNumbers="true" %}

```
from pgzhelper import *
import random

WIDTH = 960
HEIGHT = 540

hammer = Actor('toy_hammer', (WIDTH / 2, HEIGHT / 2))
hammer.scale = 0.5
hammer.angle = 40

score= 0
hammer_pressed = False

GAP_FROM_SCR = 50
moles = []
for _ in range(6):
    mole = Actor('mole')
    mole.anchor = ('left', 'top')
    x = random.randint(GAP_FROM_SCR, WIDTH - mole.width + GAP_FROM_SCR)
    y = random.randint(GAP_FROM_SCR, HEIGHT - mole.height + GAP_FROM_SCR)
    mole.pos = (x, y)
    mole.scale = 0.5
    mole.visible = False
    moles.append(mole)


def draw():
    global score
    screen.blit('field', (0, 0))

    for mole in moles:
        if mole.visible:
            mole.draw()
        if hammer_pressed and mole.visible and mole.collide_pixel(hammer):
            sounds.toi.play()
            moles.remove(mole)
            score += 1

    hammer.draw()
```

```
    screen.draw.text('Hisob: ' + str(score), (20, 20), color='black')


def update():
    if random.randint(0, 10) == 0:
        if len(moles) != 0:
            mole_list = random.sample(moles, 1)
            mole_list[0].visible = not mole_list[0].visible
        else:
            game.exit()


def on_mouse_move(pos):
    hammer.centerx, hammer.centery = pos


def on_mouse_down():
    global hammer_pressed
    hammer_pressed = True
    animate(hammer, angle=75, tween='accelerate', duration=0.1, on_finished=animation_done)


def animation_done():
    global hammer_pressed
    animate(hammer, angle=40, tween='accelerate', duration=0.1)
    hammer_pressed = False
```

{% endcode %}

Entry versiyasidagi ko'rsichqonlarni tutish o‘yinidan farqli jihati shundaki, Entry versiyasida ko'rsichqonlarning joylashuvi o‘zgarmas, lekin Pygame Zero versiyasida ko'rsichqonlar boshlang‘ich holatda tasodifiy (random) tarzda joylashtiriladi. Avval tasodifiylikdan foydalanish uchun alohida tashqi modul, ya'ni **random** moduli kerak bo‘ladi, shuning uchun 2-qatorda ushbu modul **import** qilinmoqda. 14~24-qatorlardagi kodlar ekranda tasodifiy joylashishi kerak bo‘lgan jami 6 ta ko'rsichqon Actor obyektlarini yaratib, bu yaratilgan 6-tasini **moles** (ko'rsichqonlar) degan ro‘yxatga joylashtirib ishlatadi.

19-20-qatorlarda foydalanilgan **randint** metodini Entryda foydalanilgan "tasodifiy son" bloki bilan solishtirsak, katta farqi yo‘q. U tasodifiy son tanlash uchun diapazonning eng kichik va eng katta qiymatlarini argument sifatida qabul qiladi. Lekin, metod nomida int (integer, butun son) bo‘lganligi sababli, u faqat butun sonlar diapazonida tanlov qilishi mumkin. Agar suzuvchi nuqta (float) qiymatlari kerak bo‘lsa, **randfloat** metodidan foydalanish mumkin.

randint metodiga yuborilayotgan eng kichik va eng katta qiymatlar uchun **GAP_FROM_SCR** deb nomlangan 50 piksellik bo‘shliq (margin) qiymati ishlatilmoqda. Maqsad shuki, tasodifiy joylashuv paytida o‘yin ekrani chetiga juda yaqin joylashib qolmasligi uchun, ko'rsichqonlar tasvirlari ekran chetida chiqib ketmasligi va qirqilib qolmasligi uchun ekrandan 50 piksellik ichkariga kirib tasodifiy qiymatlarni aniqlash kerak.

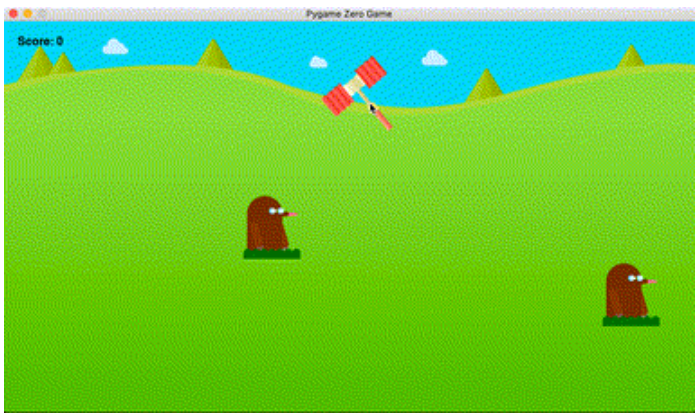
23-qatordagi **visible** (ko‘rinuvchan) deb nomlangan yangi obyekt xususiyati ishlatilmoqda, **bu Actor obyektida mavjud bo‘lmagan ichki xususiyat. Biz o‘yinda kerak bo‘lganimiz uchun qo‘ziqorinlar ekranda ko‘rinishi yoki ko‘rinmasligini aniqlash uchun bu xususiyatni qo‘shimcha qilib oldik. Shu kabi siz ham kerak bo‘lsa, mavjud obyektlarga o‘z xususiyatlaringizni qo‘shishingiz mumkin.** *Maqsad, ko'rsichqonlar o‘yinida qo‘ziqorinlar ko‘rinib yoki ko‘rinmasligini farqlash, bu o‘tgan bobda ko‘rsatilgan flag (bayroq) o‘zgaruvchisi kabi ishlaydi.*

O‘yinni qiziqarli qilish uchun tasodifiy ko'sichqonning ko‘rinishi yoki ko‘rinmasligi ham tasodifiy bo‘lishi kerak. Buni amalga oshiruvchi kod **update** funksiyasida 44~47-qatorlarda joylashgan. Biz o‘yinda ishlatiladigan obyektlarning joylashuvi, holati va boshqa ma'lumotlarini yangilab turadigan update (yangilash) funksiyasi juda tez-tez chaqirilishini bilamiz, lekin qancha tez-tezligini yaxshi bilmasdik. Endi nihoyat, bu tezlik haqida gapirishimiz kerak, bu soniyada 60 kadr (60FPS) tezlikda amalga oshiriladi. Bu juda tez tezlikda, soniyasiga 60 marta chaqiriladi. Masala shundaki, 60FPS tezligi biz uchun juda tez, bizga juda tez yoki juda sekin bo‘lmagan holda o‘yin qiziqarli bo‘lishi uchun qolgan ko'rsichqonlar orasidan tasodifiy tarzda ko‘rinish/ko‘rinmasligini aniqlash kerak. Ushbu tezlik (0-10 oralig‘idagi qiymatlarda 0 tanlanganida) 44-qatorda amalga oshirilgan. Tirik qolgan ko'rsichqonlardan birini tasodifiy tanlash **random.sample** metodi yordamida 45-46-qatorlarda amalga oshirildi va 47-qatorda tanlangan ko'rsichqonning ko‘rinish xususiyatini o‘zgartirish orqali ko‘rinish yoki ko‘rinmasligi boshqariladi.

Endi qolgan kodning katta qismi bolg‘aning animatsiyasi bilan bog‘liq. Agar oldingi misolda animatsiya bitta Actor ichida bir nechta tasvirlarning aylanishi orqali amalga oshirilgan bo‘lsa, bu safargi animatsiya hozirgi joydan belgilangan joyga yoki burchakka ma’lum bir vaqt ichida harakat qilishni o‘z ichiga oladi. Rezina bolg‘a sichqoncha bosilgan vaqtda bolg‘aning urish harakatini taqlid qilish uchun oldinga taxminan 75 daraja tezda egiladi, so‘ngra yana hozirgi burchak (40 daraja) holatiga tezda qaytadi. 59-qatorda **animate** metodi birinchi marta ishlatilgan bo‘lib, jami 5 ta argument qabul qiladi. Bularni quyidagicha umumlashtirish mumkin: qaysi obyekt (bu yerda hammer), uning qaysi xususiyati (bu yerda angle), qancha vaqt ichida (bu yerda 0,1 soniya), qanday animatsiya turi (bu yerda accelerate) bajariladi. Animatsiyaning **accelerate** turi vaqt o‘tishi bilan tezlik oshib borishini ta’minlovchi animatsiyadir. Bunday turlarning jami 10 tasi mavjud bo‘lib, ularning har biri haqida batafsil ma’lumotni kutubxona qo‘llanmasidan topishingiz mumkin. Argumentlar orasida oxirgi **on_finished** parametri ham qiziqish uyg‘otishi mumkin. Bu yerda animation_done nomli funksiya nomi (62-qatorda aniqlangan) qiymat sifatida uzatilgan. Ha, bu parametrning nomi kabi, animatsiya tugagach, uzatilgan funksiyani qayta chaqirishni (callback) ta’minlaydi. Buning turli maqsadlari bo‘lishi mumkin, lekin bizning maqsadimiz shundaki, rezina bolg‘aning egilishi animatsiyasi tugagandan so‘ng, darhol hozirgi joyga qaytish animatsiyasi davom ettiriladi.

Foydalanuvchi ko'rsichqonni qachon urganligini qanday aniqlashimiz mumkin? 34-qatorda bu savolga javob mavjud. 3 ta shart bir vaqtda bajarilishi kerak. Rezina bolg‘aning animatsiyasi oldinga egilib turgan bo‘lishi kerak (ya'ni bolg‘a ko‘tarilayotgan paytda emas), ko'rsichqon yashirin holatda emas, balki ekranda ko‘rinib turishi kerak, va oxirgi shart, bolg‘a va ko'rsichqon o‘rtasida to‘qnashuv tekshiruvi muvaffaqiyatli o‘tishi kerak. Ushbu 3 ta shart bajarilganda ko'rsichqon urilgan hisoblanadi. Muvaffaqiyatli zarbadan so‘ng effektli ovoz chiqadi (35-qator), urilgan ko'rsichqon moles (ko'rsichqonlar) ro‘yxatidan o‘chirib tashlanadi (36-qator) va oxirgi bosqichda o‘yinda to‘plangan ochkolar hisoblanadi (37-qator).

Shu bilan ko'rsichqon o‘yinining barcha kodlari haqida tushuntirish yakunlandi va oxirgi bosqichda o‘yin natijasini ko‘rish bilan tugatamiz.



Ushbu kitobni shu misolgacha tugatganlarni yana bir bor tabriklayman. Bu bilan siz nihoyat Pygame Zero boshlang'ich darajasidan o'tdingiz. Shunday qilib, endi siz Pygame Zero-ning yanada ilg'or darajasiga o'tishga tayyorsiz. Kitobning keyingi darajasida siz Python kodlash ko'nikmalaringizni biz hozirgacha ko'rgan misollardan farq qiladigan va o'yinga o'xshash(?) o'yinlar yaratish orqali yanada qiziqarli tarzda rivojlantirasiz. Siz hayajonlana boshladingizmi? Matnli dasturlash bo'yicha birinchi sayohatingizda sizga hamroh bo'lganimdan xursand bo'ldim va muallif sizning sayohatingizda sizni xursand qilishda davom etadi. Bundan tashqari, men kodlash sayohatingizda sizga amaliy yordam beradigan va o'rganishingizga yordam beradigan kodlash vositalari va kitoblarni tayyorlashni davom etaman. Va nihoyat, agar sizda tushunarsiz misollar haqida savollaringiz yoki qo'shimcha fikrlaringiz bo'lsa, iltimos, onlayn hamjamiyat orqali bog'laning. Sizning sayohatingiz hech qachon yolg'iz emas!

☀ Muallif tomonidan 2024 yilning jazirama yozi oxirida chop etilgan ☀