5.1.2 Kuchuk va mushuk Salom aytadi2 - Ketma-ketlik

Oxirgi bo'limgacha biz o'yin sahnasini (orqa fon) tayyorlashni yakunladik. Endi biz sahnada aktyor sifatida ishlaydigan ikkita ob'ektni, kuchuk va mushukni yaratishdan boshlashimiz kerak. Entryda biz ob'ektlarni o'zimiz yaratishimiz shart emas edi, chunki Entry ularni biz bilmasdan biz uchun yaratdi.

Bunday aktyor obyektlarini yaratish uchun Pygame Zero Actor nomli **sinf (class)** tayyorlagan. Sinf atamasi birinchi marta paydo bo'ldi, lekin sodda qilib aytganda, keling, uni obyektlarni aniqlash uchun ishlatiladigan grammatika deb hisoblaylik. Bu biz funktsiyalarni aniqlash uchun ishlatgan def grammatikasiga o'xshaydi.

Funktsiyani aniqlash va undan foydalanish turli tushunchalar ekanligini allaqachon tushunib yetdik. Funktsiyani aniqlash funksiya yaratish degani, boshqacha qilib aytganda, funksiya chaqirilganda va foydalanilganda nima qilishini bildiruvchi ish tavsifi bilan funksiya yaratishni anglatadi. Shundan so'ng, ushbu aniqlangan (yaratilgan) funktsiyadan foydalanish uchun funktsiya o'zining nomi bilan (va ba'zan funksiya talab qiladigan argumentlar bilan) chaqiriladi va funktsiya chaqirilgan paytda u kompyuterning xotirasiga yuklanadi va funksiyada ko'rsatilgan ishni haqiqatda bajaradigan kichik, mujassamlangan dastur sifatida ishlaydi.

Sinf va obyekt orasidagi farq.

Sinflar va obyektlar o'rtasidagi munosabatlar bir xil. **Obyektni aniqlashda sinfdan foydalaniladi va haqiqiy foydalanish uchun chaqirilgan va xotirada materiallashtirilgan holat obyekt deb ataladi (aniqrog'i, bu "holat" deb ataladi).** Boshqacha qilib aytganda, siz obyektni sinf nomi bilan chaqirish orqali yaratishingiz mumkin (xuddi funktsiya nomi bilan funktsiyani chaqirish kabi). Haqiqiy kodni ko'rib chiqish orqali buni batafsil tushunamiz.

{% code lineNumbers="true" %}

```
from pgzhelper import *

TITLE = 'Kuchuk va Mushuk salom aytadi'
WIDTH = 480
HEIGHT = 270

dog = Actor('dog', (100, 150))

def draw():
    screen.fill('white')
```

{% endcode %}

Yangi qo'shilgan kod quyidagicha. U kutubxonada oldindan belgilangan Actor sinfini (o'yin sahnasida faol bo'lgan aktyor degani uchun shunday nomlangan) chaqirish orqali obyekt yaratadi va kompyuter xotirasida **yaratilgan obyektni dog** deb nomlangan o‘zgaruvchida saqlash orqali boshqaradi.

```
dog = Actor('dog', (100, 150))
```

Agar siz bu yerda Actorga diqqat bilan qarasangiz, bu funktsiya chaqiruvi emas, balki sinf chaqiruvi ekanligini darhol ko'rishingiz mumkin. Buni darhol payqashingiz sababi shundaki, **biz Python grammatikasidagi (nafaqat Python, balki boshqa dasturlash tillari) funksiya nomlari kichik harf bilan va sinf nomlari katta harf bilan boshlanishi qoidasiga amal qilganmiz.**

Xuddi funktsiya chaqiruvi kabi, sinf chaqiruvida uni yaratishda sinfga uzatilishi kerak bo'lgan argumentlar bo'lishi yoki bo'lmasligi mumkin. Biroq, Actor ikkita argument talab qiladi. Birinchi qiymat ('dog') Actor obyekti tashqi ko‘rinishi uchun foydalanadigan rasm fayli nomi (dog.png), ikkinchi qiymat (100, 150) esa rasmning qayerda chizilishi haqidagi ma’lumotdir (aniqrog'i, ekrandagi tasvir markazining joylashuvi haqidagi ma'lumot). Ular obyekt yaratish uchun argumentlar sifatida ishlatiladi. Agar siz Actor sinfi haqida ko'proq bilmoqchi bo'lsangiz, PygameZero kutubxonasi qo'llanmasiga (dokumentatsiyaga) murojaat qilishingiz mumkin. Kutubxona muallifining o'zi sinfning maqsadi va ishlatilishini batafsil tushuntiradi.

```
from pgzhelper import *

TITLE = 'Kuchuk va Mushuk salom aytadi'
WIDTH = 480
HEIGHT = 270

dog = Actor('dog', (100, 150))
dog.scale = 0.5

def draw():
    screen.fill('white')
    dog.draw()
```

Bu yerga qadar obyekt muvaffaqiyatli yaratilgan bo‘lsa, endi ushbu obyektni ekranda aks ettirish (chizish) kerak. O‘tgan darsda ekranni butunlay oq rang bilan to‘ldirgan edik, endi shu fon ustiga obyektni chizamiz (12-qator). Shu sababli, *7-qatorida yaratilgan "kuchuk" obyektini ifodalovchi (uni ko‘rsatib turgan)* ***dog*** *o‘zgaruvchisi* orqali obyekt ichidagi **draw** funksiyasini chaqirish kifoya. Kod shaklida bu quyidagicha yoziladi: **dog.draw()**. Bu yerda biz avvaldan kutubxona ichidagi funksiyalar yoki o‘zgaruvchilarni chaqirish uchun tez-tez ishlatgan **" . " (nuqta operatori)** yordamida, "kuchuk" obyekti ichidagi **draw()** funksiyasini chaqirayapmiz.

Endi savol tug‘iladi, 8-qatoridagi kod nima uchun kerak? **dog.scale = 0.5** degan kodning ma'nosi shuki, bu orqali "kuchuk" obyekti ichidagi **scale** o‘zgaruvchisiga 0.5 qiymati tayinlanmoqda. Agar bu kodni qo‘shmasdan, hozirgi kodni ishga tushirsangiz, "kuchuk" obyektining tasviri ekranda yaxshi chizilishini ko‘rasiz. Ammo bir muammo mavjud: "kuchuk" obyektining tasviri juda katta bo‘ladi. Shuning uchun, bu muammoni hal qilish, ya’ni tasvir hajmini yarmiga qisqartirish uchun ushbu kod kerak edi. Ushbu kodni qo‘shib, dasturni qaytadan ishga tushirsangiz, obyekt tasviri mos hajmda ko‘rinadi.

Obyekt nima?

Dastlab obyektni yaratib va foydalanib ko‘rganingizda, yangi paydo bo‘lgan ushbu obyekt tushunchasi aslida nima ekanligi haqida qanday fikrga keldingiz? *Obyekt ichida funksiyalar va o‘zgaruvchilar mavjud bo‘lib, ulardan foydalanish uchun* `"."` *(nuqta operatori) ishlatiladi, degan fikrga ko‘ra, ehtimol siz obyektni kutubxonaga juda o‘xshash va balki kichraytirilgan kutubxona,* deb o‘ylashingiz mumkin. **Foydalanish jihatidan kutubxonaga juda o‘xshash bo‘lsada, yaratilish maqsadi va obyektni bevosita loyihalash hamda yaratish jarayoni o‘rtasida katta farq bor.** Tez-tez takrorlanadigan funksiyalarni biror joyga

(kutubxonaga) to‘plab, undan foydalanish kodning qayta ishlatilishi jihatidan qulay bo‘lishi uchun yaratilgani kutubxona deb hisoblanadi. Obyekt ham oxir-oqibatda kodning qayta ishlatilishini oshirish maqsadini ko‘zlaydi, ammo shunga qo‘shimcha ravishda, dasturiy ta’minot ishlab chiqishning murakkabligini kamaytirish maqsadidan kelib chiqadi, deyish mumkin. Agar bu mavzuni batafsilroq bilishni istasangiz, avvalgi kitobda keltirilgan obyektga yo‘naltirilgan dasturlash (OOP) paradigmasi haqida yana bir bor o‘qib chiqishingiz foydali bo‘ladi.

Endi esa dasturlashda obyekt deganda nimani anglatishi haqida tushuntirish kerak. **Obyekt — bu uning nomidan ham tushunish mumkin bo‘lganidek, real dunyoda(real world) mavjud bo‘lgan obyektlar(masalan, chashka, stol, stul, gulvazo, televizor, avtomobil va hokazo)ni dasturlash olamiga(programming world) olib kirish uchun ularning xususiyatlari (attribute) va harakatlarini (behavior) modellashtirish orqali yaratilgan kod tuzilmasidir.** *Shuni yodda tutingki, obyektning xususiyatlari va harakatlarini kod orqali ifodalashda* ***xususiyatlar o‘zgaruvchilar yordamida, harakatlar esa funksiyalar orqali (ularni metodlar deb ham atashadi) ifodalanadi.***

Bizning birinchi obyektimiz sifatida foydalanilgan Aktyor (Actor) obyekti aslida nimani modellashtiradi? Aslida, ushbu obyekt real dunyoda mavjud emas, balki dasturlash olamidagi o‘yin sahnasida aktyor sifatida bir rolni bajarish uchun yaratilgan. Agar siz ushbu obyektni ilk bor dizayn qilayotgan bo‘lsangiz, uni qanday loyihalashtirgan bo‘lar edingiz? Ushbu obyektning o‘yin sahnasi ichida bajarishi kerak bo‘lgan rollari va mas'uliyatlari nima? U qanday xususiyatlarga (o‘zgaruvchilarga) ega bo‘lishi va qanday harakatlar(funksiyalar)ni bajarishi kerakligini o‘ylab ko‘rish kerak. Birinchi navbatda kerakli xususiyatlarni ko‘rib chiqaylik: aktyor obyektining tashqi ko‘rinishi tasvir (rasm) orqali bo‘lgani sababli, tashqi ko‘rinishni qanday tasvir faylidan yaratishni belgilash uchun **rasm nomi (image name) xususiyati** kerak bo‘lishi mumkin. Ba'zan ushbu tasvirning o‘lchami o‘yin sahnasiga mos kelishi uchun uni belgilash imkonini beruvchi **o‘lcham (scale)** xususiyati foydali bo‘lishi mumkin. Bundan tashqari, tasvirni qaysi joyga joylashtirishni belgilash uchun **joylashuv (position)** xususiyati kerak bo‘ladi. Agar ushbu tasvirni aylantirib ko‘rsatishni xohlasak, burchakni belgilash uchun **burchak (angle)** xususiyati ham bo‘lishi mumkin. Endi esa aktyor obyektining bajarishi kerak bo‘lgan harakatlar haqida o‘ylab ko‘raylik. Aktyor asosiy harakat sifatida ekranda **ko‘rinishi (chizilishi) kerak**. U **harakatlanishi lozim** va o‘yin sahnasi xususiyatlari sababli **boshqa obyektlar bilan to‘qnashuv** yoki ularga tegishni aniqlaydigan vazifalarni bajarishi kerak bo‘ladi.

Actor obyekti biz yuqorida taxmin qilgan darajadagi xususiyatlar va harakatlarni o‘z ichiga olish bilan birga, aslida undan ham ko‘p xususiyatlar va harakatlarga ega. Agar hozirning o‘zida Actor obyektining qanday xususiyatlarga ega ekanligi va nimalarni qila olishini bilmoqchi bo‘lsangiz, bu obyektning yaratuvchisi tomonidan yozilgan qo‘llanmani ko‘rib chiqishingiz mumkin. Ammo, bu kitobning misollarini bosqichma-bosqich o‘rganib borar ekansiz, tabiiy ravishda bularni tushunib olasiz, shuning uchun shoshilishga hojat yo‘q.

Ushbu bo‘limda siz zamonaviy dasturlash tushunchalari ichida eng muhim bo‘lgan obyekt tushunchasini o‘rgandingiz. Bu yutuq bilan sizni tabriklaymiz! Obyekt haqida hozircha shu darajadagi tushuncha yetarli, va yanada chuqurroq mavzularni keyinchalik ko‘rib chiqamiz. Endi esa qolgan kodlarni davom ettiraylik!