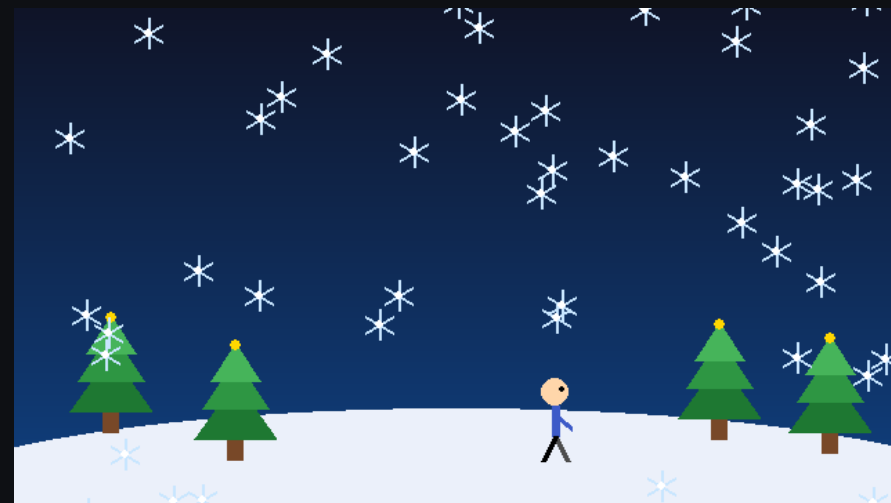


Sprajtovi, grupe i animacija lika

Kako igra drži na okupu mnogo pokretnih stvari i kako lik oživi

Danas pravimo: Čikica mrda nogama dok hoda, a sneg je grupa sprajtova



Četiri slike hoda, svaka po 0,12 sekundi

Tri pitanja za zagrevanje

1 • prošla tema: unicode

```
import pygame as pg

d = pg.event.Event(
    pg.KEYDOWN,
    key=pg.K_a,
    unicode="A")
print(d.unicode, d.key)
```

A 97

2 • tema 05: ista metoda

```
class Sneg:
    def opis(self):
        return "pada"

class Lik:
    def opis(self):
        return "hoda"

for o in [Lik(), Sneg()]:
    print(o.opis())
```

hoda
pada

3 • mešano: ostatak

```
slike = "abcd"
i = 0
for k in range(6):
    i = (i + 1) % 4
print(slike[i])
```

c

Šest novih reči za danas

sprajt

objekat igre sa slikom i mestom: lik, pahuljica, lopta

image

atribut sprajta: njegova slika (površina, Surface)

rect

atribut sprajta: pravougaonik koji kaže gde je slika

grupa

skup sprajtova koji se ažuriraju i crtaju jednim pozivom

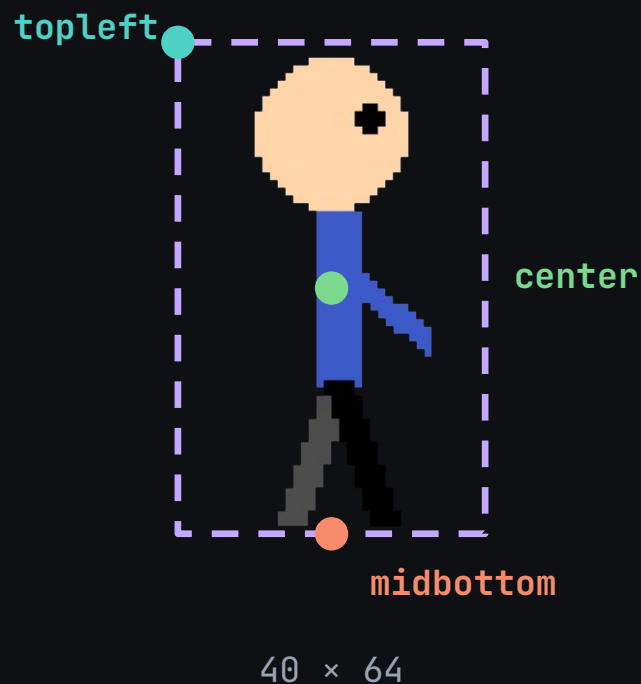
kill()

sprajt izlazi iz svih grupa: nestaje iz igre

frejm animacije

jedna slika iz niza; smenjuju se i lik „hoda“

Sprajt je slika i pravougaonik zajedno



`self.image`

slika: šta se crta (Surface)

`self.rect`

pravougaonik: gde se crta i koliko je velika slika

`pg.sprite.Sprite`

klasa koju nasleđujemo; daje add, kill, alive i rad sa grupama

Modul `pygame.sprite` ima tri alata

Sprite

klasa koju nasleđuju naši objekti igre: Igrac, Pahuljica, Lopta

danas

Group

skup sprajtova: `update` i `draw` za sve odjednom

danas

sudari

da li se sprajt dodiruje sa drugim sprajtom ili grupom

tema 10

Sprajt je ono što smo do sada pisali kao klasu sa `azuriraj` i `crtaj`, samo u obliku koji grupa razume.

Sprajt

Klasa sa slikom i pravougaonikom

Prvi sprajt: nasleđujemo pg.sprite.Sprite

```
import pygame as pg

class Pahuljica(pg.sprite.Sprite):
    def __init__(self, x, y):
        super().__init__() # obavezno
        self.image = pg.Surface((24, 24))
        self.rect = self.image.get_rect(center=(x, y))

p = Pahuljica(100, 50)
print(p.rect)
print(p.alive())
```

ISPIS

```
Rect(88, 38, 24, 24)
False
```

- `super().__init__()` je obavezan, kao u temi 04.
- `get_rect(center=...)` pravi rect na pravom mestu.
- `alive()` je `False`: sprajt još nije ni u jednoj grupi.

Slika sprajta: učitana ili nacrtana kodom

Učitana iz fajla (pahuljica, Čikica)

```
# posle set_mode:  
SLIKA = pg.image.load("pahuljica.png").convert_alpha()  
self.image = SLIKA
```

convert_alpha()

čuva providnost; poziva se posle set_mode

pg.SRCALPHA

nova površina je providna, pa se vidi samo krug

Nacrtana na novoj providnoj površini (lopta)

```
self.image = pg.Surface((30, 30), pg.SRCALPHA)  
pg.draw.circle(self.image, "orange", (15, 15), 15)
```

sliku učitavamo jednom

pre petlje, ne za svaki sprajt i ne u svakom frejmu

Grupe

Jedan poziv za sve sprajtove

Grupa poziva update za svaki sprajt

```
import pygame as pg

class Tacka(pg.sprite.Sprite):
    def __init__(self, x):
        super().__init__()
        self.image = pg.Surface((4, 4))
        self.rect = self.image.get_rect(topleft=(x, 0))

    def update(self, dt):
        self.rect.y += 100 * dt

grupa = pg.sprite.Group()
for x in (10, 20, 30):
    grupa.add(Tacka(x))
grupa.update(0.5)          # svima: update(0.5)
print(len(grupa))
print([t.rect.y for t in grupa])
```

ISPIS

```
3
[50, 50, 50]
```

grupa.update(dt)

pozove update(dt) svakog sprajta

grupa.draw(ekran)

nacrtaj image svakog sprajta na mestu rect

Šta ispisuje ovaj program?

```
import pygame as pg

a = pg.sprite.Sprite()
b = pg.sprite.Sprite()
c = pg.sprite.Sprite()
grupa = pg.sprite.Group(a, b, c)
druga = pg.sprite.Group(a)
a.kill()
print(len(grupa), len(druga), a.alive())
```

Sprajt a je u dve grupe.

A

tačno

2 0 False

B

2 1 False

C

3 1 True

D

2 1 True

Pahuljica iz teme 07 postaje sprajt

```
class Pahuljica(pg.sprite.Sprite):  
    def __init__(self, y=-20):  
        super().__init__()  
        self.image = SLIKA  
        self.x0 = random.uniform(0, SIRINA) # sredina njihanja  
        self.polozaj = pg.Vector2(self.x0, y)  
        self.brzina = random.uniform(40, 120) # px u sekundi  
        self.faza = random.uniform(0, 6.28)  
        self.vreme = 0 # svoj sat  
        self.rect = self.image.get_rect(center=self.polozaj)
```

y=-20

podrazumevano: nova pahuljica
kreće iznad prozora

self.polozaj

Vector2 sa realnim brojevima

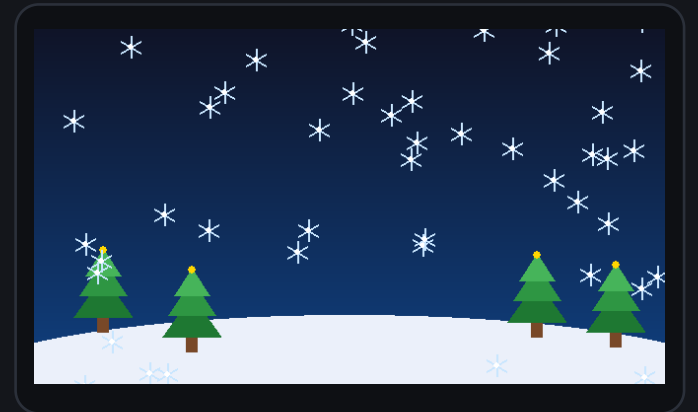
self.vreme

svaka pahuljica meri svoje
vreme, za sinus

Pahuljica koja ispadne sama nestaje

```
def update(self, dt):
    self.vreme += dt
    self.polozaj.y += self.brzina * dt
    njihanje = 15 * math.sin(2 * self.vreme + self.faza)
    self.polozaj.x = self.x0 + njihanje
    self.rect.center = self.polozaj      # Rect: celi brojevi
    if self.rect.top > VISINA:           # ispala je dole
        self.kill()                     # izlazi iz svih grupa
```

Isto kao u temi 07, samo što su položaj i rect zajedno, a kraj pada je kill().



- Ime metode je update, jer je tako zove grupa.
- Tajmer sada dodaje pahuljicu svakih 0,1 s; stare same odlaze.

Čudovište: sprajt bez image

```
class Pahuljica(pg.sprite.Sprite):  
    def __init__(self):  
        super().__init__()  
        self.slika = pg.Surface((24, 24)) # nije image  
        self.rect = self.slika.get_rect()
```

```
sneg = pg.sprite.Group(Pahuljica())  
sneg.draw(ekran)
```

PORUKA (pygame-ce)

```
TypeError: Source objects must be a surface  
bez rect: TypeError: invalid destination position for blit
```

Kako izgleda

Program pada tek pri crtanju grupe, sa porukom o površini.

Zašto se pojavljuje

Grupa traži tačno imena image i rect.

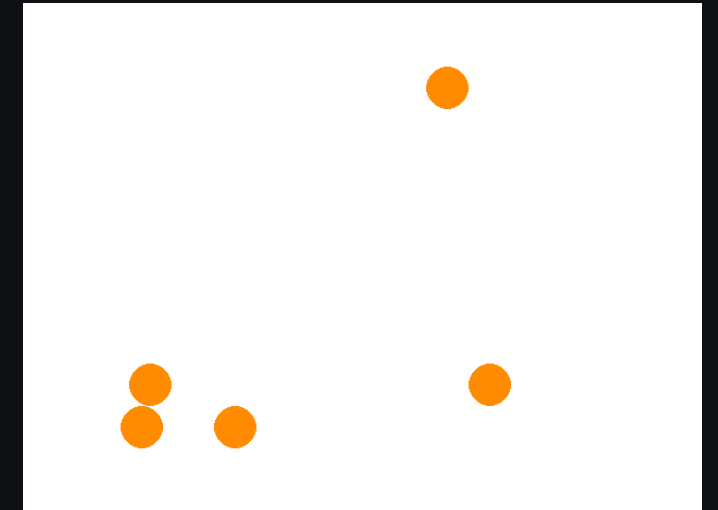
Kako se pobeđuje

U `__init__` uvek: `self.image = ...` i `self.rect = ...`

Lopte skočice: grupa radi sav posao

```
def update(self, dt):
    self.polozaj += self.brzina * dt
    self.rect.center = self.polozaj
    if self.rect.left < 0 or self.rect.right > SIRINA:
        self.brzina.x = -self.brzina.x
    if self.rect.top < 0 or self.rect.bottom > VISINA:
        self.brzina.y = -self.brzina.y
    self.rect.clamp_ip(ekran.get_rect()) # vrati u prozor
    self.polozaj = pg.Vector2(self.rect.center)
```

```
lopte = pg.sprite.Group()
for i in range(5):
    lopte.add(Lopta())
...
lopte.update(dt) # sve lopte se pomere
lopte.draw(ekran) # sve lopte se nacrtaju
```



Zadatak iz starih beleški (beachBounce),
prerađen: pet lopti u grupi, sa dt.

Animacija lika

Niz slika koje se smenjuju u vremenu

Hod su četiri slike: noge u četiri položaja



HOD[1][0]

prednja noga napred



HOD[1][1]

noge se mimoilaze



HOD[1][2]

zadnja noga napred



HOD[1][3]

noge se mimoilaze



STOJI[1]

kad ne hoda

Crna je prednja noga, tamnosiva zadnja. Posle četvrte slike ide opet prva: 0, 1, 2, 3, 0, 1...

Frejmove hoda crta funkcija, ne ruka

```
def cikica(smer, prednja, zadnja, ruka):
    s = pg.Surface((40, 64), pg.SRCALPHA)
    pg.draw.circle(s, KOZA, (20, 12), 10)
    pg.draw.circle(s, "black", (20 + 5 * smer, 10), 2)
    pg.draw.line(s, PLAVA, (20, 22), (20, 44), 6)
    pg.draw.line(s, PLAVA, (20, 28), (20 + ruka * smer, 38), 4)
    pg.draw.line(s, "gray30", (20, 44), (20 + zadnja * smer, 62), 4)
    pg.draw.line(s, "black", (20, 44), (20 + prednja * smer, 62), 4)
    return s
```

```
KORACI = [(9, -9, -6), (2, -2, 4), (-9, 9, 12), (-2, 2, 4)]
```

U 09_01 je dodato još samo podizanje stopala: noga koja ide napred je malo kraća.

prednja, zadnja

pomeraj stopala od
sredine; + je u smeru
hoda

smer

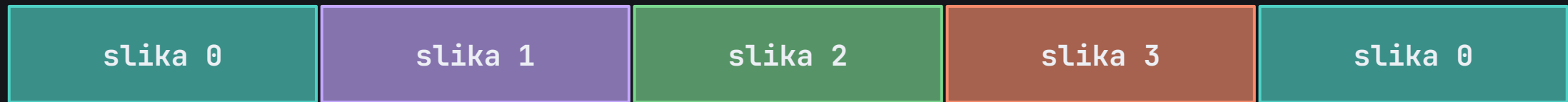
1 desno, -1 levo: ista
funkcija crta obe strane

KORACI

četiri položaja nogu i ruke

Nova slika na svakih 0,12 s, po satu

frejmovi igre ($dt \approx 0,017$ s)



- vreme_frejma sabira dt ; kad pređe 0,12, ide sledeća slika.
- Isti broj slika u sekundi pri 30, 60 ili 144 FPS.
- Posle slike 3 opet ide slika 0.

animiraj meri vreme i bira sliku

```
def animiraj(self, dt, hoda):  
    if not hoda:                                # stoji  
        self.image = STOJI[self.smer]  
        return  
    self.vreme_frejma += dt  
    if self.vreme_frejma >= TRAJANJE_FREJMA:    # 0,12 s  
        self.vreme_frejma -= TRAJANJE_FREJMA  
        self.indeks = (self.indeks + 1) % 4  
        self.image = HOD[self.smer][self.indeks]
```

-= TRAJANJE_FREJMA

ostatak vremena se ne gubi, pa ritam ostaje tačan

% 4

posle 3 ide 0: slike ukrug

hoda

False kad nijedna strelica nije dole

Po vremenu: noge mrdaju isto na svakom računaru

```
for fps in (30, 60):  
    dt = 1 / fps  
    po_frejmu = 0  
    po_vremenu = 0  
    vreme_frejma = 0  
    for frejm in range(1, fps + 1):      # 1 sekunda  
        if frejm % 4 == 0:                # svaki 4. frejm  
            po_frejmu += 1  
            vreme_frejma += dt  
            if vreme_frejma >= 0.12:      # svakih 0,12 s  
                vreme_frejma -= 0.12  
                po_vremenu += 1  
    print(fps, "FPS:", po_frejmu, po_vremenu)
```

ISPIS

30 FPS: 7 8

60 FPS: 15 8

Po frejmu: 7 promena pri 30 FPS,
a 15 pri 60 FPS: na brzom računaru
Čikica „trči u mestu“.

Po vremenu: 8 promena u sekundi,
svuda.

Šta ispisuje ovaj program?

```
indeks = 2  
for promena in range(5):  
    indeks = (indeks + 1) % 4  
print(indeks)
```

Slika hoda se promenila pet puta.

A
7

B
3

tačno

C
1

D
0

Smer bira niz slika: rečnik sa 1 i -1

```
HOD = {1: hod_desno, -1: hod_levo}      # liste od 4 slike  
STOJI = {1: stoji_desno, -1: stoji_levo}  
  
# levo može i kao ogledalo desne slike:  
levo = pg.transform.flip(desno, True, False)
```

`flip(slika, True, False)` je ogledalo levo-desno; `flip(slika, False, True)` okreće sliku naopako.

Mi levo crtamo istom funkcijom ($smer = -1$), jer Čikica gleda u smeru hoda i oko je na pravoj strani.



`HOD[1][0]`



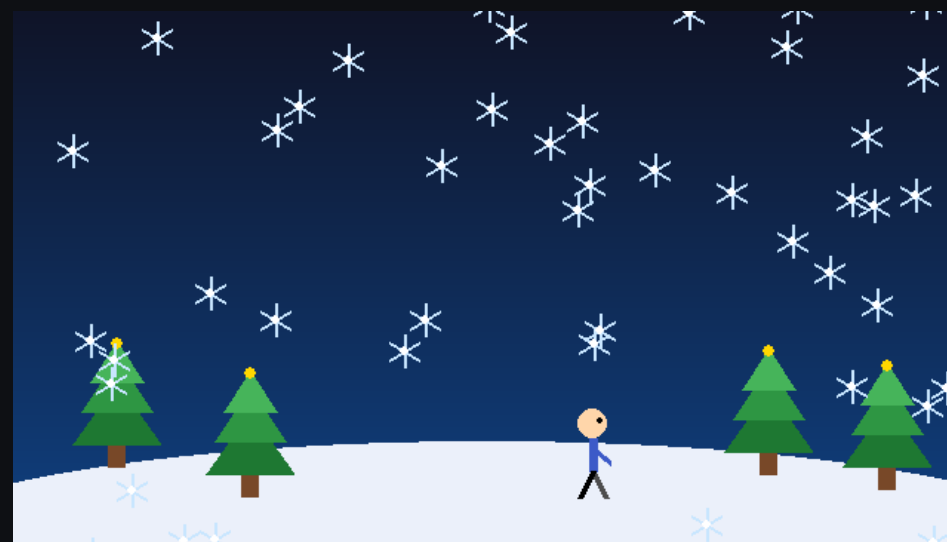
`HOD[-1][0]`

Zimska šetnja, korak 4: Čikica hoda

Cilj: Igrac i Pahuljica su sprajтови, sneg je grupa, a Čikica mrda nogama.

- 1 Pokrenemo 09_01: nastaju slike hoda
- 2 Pahuljica(pg.sprite.Sprite) sa kill
- 3 Igrac(pg.sprite.Sprite): image i rect
- 4 animiraj: nova slika na 0,12 s
- 5 Petlja: update i draw za grupe

KAKO IZGLEDA NA KRAJU



Isti program kao u temi 08, sa klasama koje nasleđuju Sprite.

Igrač kao sprajt: x i y su noge

```
class Igrac(pg.sprite.Sprite):  
    def __init__(self, x):  
        super().__init__()   
        self.x = x  
        self.y = TL0          # noge (donja ivica slike)  
        self.brzina = 220      # px u sekundi  
        self.smer = 1  
        self.brzina_y = 0  
        self.indeks = 0        # koja slika hoda  
        self.vreme_frejma = 0  # koliko traje ta slika  
        self.image = STOJI[1]  
        self.rect = self.image.get_rect(midbottom=(x, TL0))
```

`self.y = TL0`

y je sada donja ivica slike: noge na snegu

`indeks, vreme_frejma`

koja slika hoda i koliko je već na ekranu

`midbottom`

rect se postavlja za sredinu donje ivice

update: hod, skok, animacija, pa rect

```
def update(self, dt):
    tasteri = pg.key.get_pressed()
    hoda = tasteri[pg.K_LEFT] or tasteri[pg.K_RIGHT]
    if tasteri[pg.K_LEFT]:
        self.x -= self.brzina * dt
        self.smer = -1
    if tasteri[pg.K_RIGHT]:
        self.x += self.brzina * dt
        self.smer = 1
    self.x = pg.math.clamp(self.x, 20, SIRINA - 20)
    self.brzina_y += GRAVITACIJA * dt
    self.y = min(self.y + self.brzina_y * dt, TL0)
    if self.y == TL0:
        self.brzina_y = 0
    self.animiraj(dt, hoda)
    self.rect.midbottom = (self.x, self.y)
```

get_pressed() u update

igrač sam čita tastere, pa grupa zove samo update(dt)

hoda

True dok je bar jedna strelica dole

rect na kraju

rect prati x i y posle svih promena

U petlji su samo grupe: update, pa draw

```
igrac = Igrac(320)
likovi = pg.sprite.Group(igrac)
pahuljice = pg.sprite.Group()
for i in range(40):
    pahuljice.add(Pahuljica(random.uniform(0, VISINA)))
pg.time.set_timer(NOVA_PAHULJICA, 100)
...
    likovi.update(dt)
    pahuljice.update(dt)
    ekran.blit(POZADINA, (0, 0))
    likovi.draw(ekran)
    pahuljice.draw(ekran)
```

likovi

grupa sa jednim sprajtom: Čikicom

0,1 s

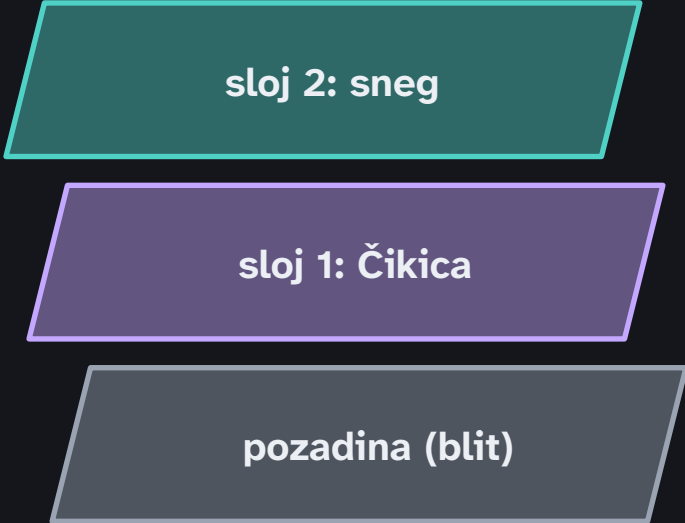
tajmer češće: pahuljice same odlaze

redosled draw

ko se crta kasnije, ispred je

LayeredUpdates: grupa sa slojevima

```
sve = pg.sprite.LayeredUpdates()
sve.add(igrac, layer=1)
for p in pahuljice:
    sve.add(p, layer=2)          # sneg ispred lika
sve.update(dt)
sve.draw(ekran)                 # prvo sloj 1, pa sloj 2
```



sloj 2: sneg

sloj 1: Čikica

pozadina (blit)

Samo pominjemo: u našoj igri su dovoljne dve grupe nacrtane pravim redom. Slojevi pomažu kad ima mnogo vrsta sprajtova (pozadina, likovi, efekti, natpisi).

Pronađi grešku: lik skoči u gornji levi ugao

```
def update(self, dt):  
    self.vreme_frejma += dt  
    if self.vreme_frejma >= 0.12:  
        self.vreme_frejma -= 0.12  
        self.indeks = (self.indeks + 1) % 4  
    self.image = self.frejmovi[self.indeks]  
    self.rect = self.image.get_rect()
```

Program se ne ruši. Animacija radi, ali lik se posle prvog frejma vidi samo u uglu prozora.

ISPIS: `rect.midbottom`

na početku: (320, 330)
posle update: (20, 64)

get_rect() nov pravougaonik, na (0, 0)

rešenje ne praviti rect iznova: slike su iste veličine

ili novi rect sa midbottom na starom mestu

Izaberite nivo i počnite

LEVEL 1

Lopte u boji

Svaka lopta ima slučajnu boju, a klik dodaje novu loptu u grupu, na mestu klika.

Pomoć: `Lopta(x, y)` i `lopte.add(...)`.

LEVEL 2

Čikica trči

Dok je Shift dole, Čikica ide dva puta brže i dva puta brže mrda nogama.

Pomoć: `pg.K_LSHIFT` i kraće trajanje frejma.

LEVEL 3

Skok i disanje

U vazduhu posebna slika skoka; kad stoji, dve slike na 0,5 s, pa lagano „diše“.

Pomoć: `skok_desno.png` iz 09_01.

Sledeći put Čikica hvata pahuljice

```
uhvacene = pg.sprite.spritecollide(igrac, pahuljice, True)  
poeni += len(uhvacene)
```

U temi 10 sudar Čikice i pahuljice daje poen, a pahuljica nestaje: to je isti kill koji smo danas koristili. Prepreke koje padaju oduzimaju život.

Rezime

sprajt

```
class P(pg.sprite.Sprite):
    super().__init__()
```

Uvek image i rect.

grupa

```
g = pg.sprite.Group()
g.update(dt)
g.draw(ekran)
```

Jedan poziv za sve sprajtove.

kill()

```
if self.rect.top > VISINA:
    self.kill()
```

Izlazi iz svih grupa.

animacija

```
vreme_frejma += dt
indeks = (indeks + 1) % 4
```

Nova slika na 0,12 s, ne po frejmu.

Pitanje pre odlaska

```
TRAJANJE_FREJMA = 0.12 # sekundi  
# lik hoda tačno 3 sekunde
```

Koliko puta se slika hoda promeni za 3 sekunde hoda, na računaru sa 60 FPS? A sa 30 FPS?

Pošaljite dva broja pre nego što izađete.



Literatura

- Dokumentacija za pygame-ce, modul sprite (Sprite, Group, LayeredUpdates): pyga.me/docs
- Kafle, S. Learning Python by Building Games, Packt, 2019 (poglavlja o PyGame-u)
- Sweigart, A. Invent Your Own Computer Games with Python, 4. izdanje, No Starch Press (besplatno na inventwithpython.com)
- Williams, R. The Animator's Survival Kit, Faber & Faber (hod lika, za one koje zanima animacija)

Dodatna literatura i zanimljivi sajtovi

pygame-ce: modul sprite

pyga.me/docs/ref/sprite.html

Sprite, Group, kill, alive i LayeredUpdates, sa svim metodama.

Piskel

piskelapp.com

Besplatan editor u pregledaču za piksel-art i animirane sprajtove.

Kenney: Toon Characters 1

kenney.nl/assets/toon-characters-1

Likovi u više poza za igre, besplatni pod licencom CC0.

OpenGameArt

opengameart.org

Grafike, muzika i zvukovi za igre; licenca piše uz svaki rad.

Hvala!

Pitanja su uvek dobrodošla: na času, u četvrtu ili međim.

Slajdovi, primeri i kartica za podsećanje su na sajtu predmeta.

PREDAVANJA

dr Staša Vujičić Stanković

stasa.vujicic.stankovic@math.rs

www.matf.bg.ac.rs/~stasa.vujicic.stankovic

VEŽBE

Milan Kocić

milan.kocic@matf.bg.ac.rs

poincare.matf.bg.ac.rs/~milan.kocic