

Za radoznale: Prozor, crtanje, slike i tekst

Ovo nije obavezno i neće biti na ispitu. Za one koji žele da vide kako se ista stvar radi naprednije. Svi ispisi su dobijeni pokretanjem u pygame-ce.

1. set_colorkey i set_alpha: providnost bez SRCALPHA

Na času smo providnost dobijali na dva načina: SRCALPHA za površine koje sami crtamo i convert_alpha za PNG slike. Postoje još dva. set_colorkey(boja) kaže da se jedna boja uopšte ne crta; tako su se nekad pravili likovi na JPG slikama sa jednobojnom pozadinom (često magenta, jer je u crtežu skoro nema). set_alpha(broj) čini celu površinu providnom, kao „providnost sloja“ u programu za crtanje.

```
import pygame as pg

pg.init()
ekran = pg.display.set_mode((100, 100))

lik = pg.Surface((20, 20))          # obična površina, bez SRCALPHA
lik.fill("magenta")
pg.draw.circle(lik, "white", (10, 10), 8)
lik.set_colorkey("magenta")        # magenta se ne crta
ekran.fill("black")
ekran.blit(lik, (0, 0))
print("ugao:", ekran.get_at((0, 0)), "sredina:", ekran.get_at((10, 10)))

duh = pg.Surface((20, 20))
duh.fill("white")
duh.set_alpha(128)                  # pola providno
ekran.fill("black")
ekran.blit(duh, (0, 0))
print("duh na crnom:", ekran.get_at((5, 5)))
```

Ispis:

```
ugao: Color(0, 0, 0, 255) sredina: Color(255, 255, 255, 255)
duh na crnom: Color(127, 127, 127, 255)
```

Ugao lika je ostao crn, jer se magenta nije nacrtala, a sredina je bela. Beli „duh“ sa alfa 128 na crnoj pozadini daje sivu: pola bele i pola crne.

Zašto ovo nismo radili na času: sličice u ovom predmetu crta kod, sa providnom pozadinom, pa su SRCALPHA i convert_alpha dovoljni; set_alpha koristimo u temi 11, za čestice koje blede.

Fajl: `primeri/za_radoznale/06_r1_colorkey_i_alpha.py`

2. subsurface: deo velike slike

Mnoge igre drže sve sličice lika u jednoj velikoj slici (list sprajtova, na engleskom sprite sheet): frejmovi hoda su poređani jedan do drugog. Metoda subsurface vraća površinu koja je samo jedan pravougaoni deo velike slike, bez kopiranja piksela.

```
import pygame as pg

# „list“ od tri frejma, svaki 40 × 40, jedan do drugog
traka = pg.Surface((120, 40))
for i, boja in enumerate(["red", "green", "blue"]):
    traka.fill(boja, (i * 40, 0, 40, 40))

frejmovi = []
for i in range(3):
    frejmovi.append(traka.subsurface((i * 40, 0, 40, 40)))

for f in frejmovi:
    print(f.get_size(), f.get_at((0, 0)))
```

Ispis:

```
(40, 40) Color(255, 0, 0, 255)
(40, 40) Color(0, 255, 0, 255)
(40, 40) Color(0, 0, 255, 255)
```

Svaki frejm je površina 40 × 40, a gornji levi piksel svakog frejma ima boju svog dela trake. Pošto subsurface deli piksele sa originalom, crtež na traci se vidi i u frejmovima.

Zašto ovo nismo radili na času: u temi 09 frejmove hoda Čikice crta kod, svaki na svojoj površini, pa list sprajtova nije potreban; koristan je kad grafiku preuzmete sa sajta kao što je OpenGameArt.

Fajl: `primeri/za_radoznale/06_r2_subsurface.py`

3. font.size: koliko je tekst širok pre crtanja

Ponekad treba znati koliko će tekst biti širok pre nego što ga nacrtamo, na primer da bismo izabrali veličinu fonta tako da naslov stane u prozor. Metoda `font.size(tekst)` vraća širinu i visinu, bez pravljenja sličice.

```
import pygame as pg

pg.init()
for velicina in (32, 64, 96):
    font = pg.font.Font(None, velicina)
    print(velicina, font.size("Zimska šetnja"))

# najveći font za koji naslov staje u 300 piksela
velicina = 120
while pg.font.Font(None, velicina).size("Zimska šetnja")[0] > 300:
    velicina -= 4
print("staje:", velicina)
```

Ispis:

```
32 (155, 23)
64 (306, 46)
96 (456, 67)
staje: 60
```

Širina raste skoro srazmerno veličini fonta. Petlja smanjuje veličinu za 4 dok naslov ne stane u 300 piksela. Brojevi zavise od fonta, pa sa drugim fontom mogu da budu malo drugačiji.

Zašto ovo nismo radili na času: na času je naslov imao stalnu veličinu; računanje veličine potrebno je tek kad tekst menja dužinu, na primer kod imena igrača ili rezultata.

Fajl: `primeri/za_radoznale/06_r3_velicina_teksta.py`