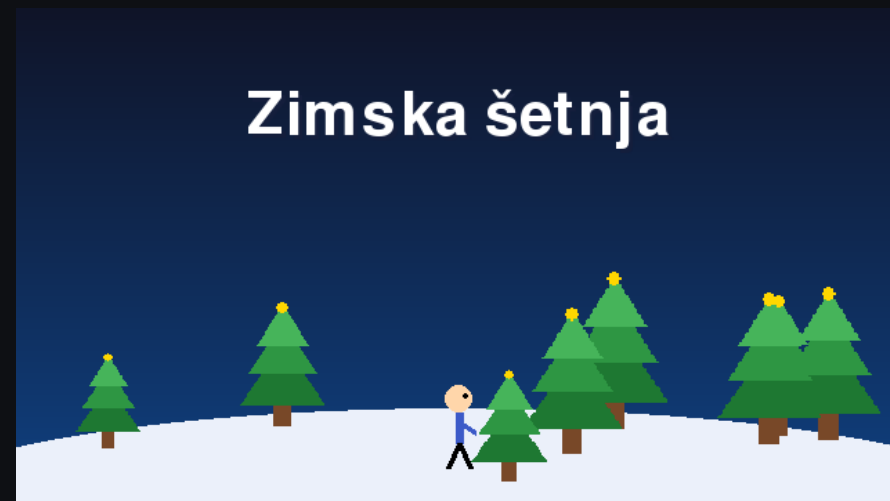


Prozor, crtanje, slike i tekst

Brzo ponavljamo pygame iz PZU1, pa učimo da slažemo scenu od površina, slika i teksta

Danas pravimo: prvi korak igre „Zimska šetnja“, mirnu zimsku scenu



Jelkice nasumične veličine, Čikica i naslov sa senkom; svaki klik daje novu scenu

Tri pitanja za zagrevanje

1. prošla tema: `__len__`

```
class Lik:
    def __len__(self):
        return 7

print(len(Lik()) + 1)
```

8

2. tema 03: atribut klase

```
class Jelka:
    broj = 0
    def __init__(self):
        Jelka.broj += 1

a = Jelka()
b = Jelka()
print(Jelka.broj)
```

2

3. mešano: Rect iz PZU1

```
import pygame as pg

r = pg.Rect(10, 20, 30, 40)
print(r.right, r.bottom)
```

40 60

Šest novih reči za danas

površina (Surface)

pravougaona slika od piksela; prozor je takođe površina

blit

„lepljenje“ jedne površine na drugu, na zadato mesto

okvir (Rect)

pravougaonik slike: zna gde su centar, ivice i uglovi

alfa kanal

četvrti broj boje: koliko je piksel providan (0 do 255)

convert

priprema slike za brže crtanje; convert_alpha čuva providnost

petlja vođena događajima

čeka događaj (klik, X) i tek tada nešto uradi

Iz PZU1 već znamo šest stvari

prozor

```
ekran = pg.display.set_mode((640, 360))
```

koordinate

(0, 0) je gore levo; y raste nadole

boje

```
"white", (64, 85, 135), "#405587"
```

oblici

```
pg.draw.rect, circle, line, polygon
```

slike

```
slika = pg.image.load("jelkica.png")
```

tekst

```
font.render("Zdravo", True, "white")
```

Danas ovo brzo ponavljamo, pa dodajemo novo: površinu kao platno, okvir slike, convert i skaliranje.

U PZU2 koristimo pygame-ce

U komandnoj liniji (terminalu):

```
pip uninstall pygame  
pip install pygame-ce
```

Provera, u Pythonu:

```
import pygame as pg  
  
print(pg.version.ver)
```

Dva paketa, jedan uvoz

pygame i pygame-ce se oba uvoze kao import pygame, pa ne smeju biti instalirana zajedno.

ISPIS

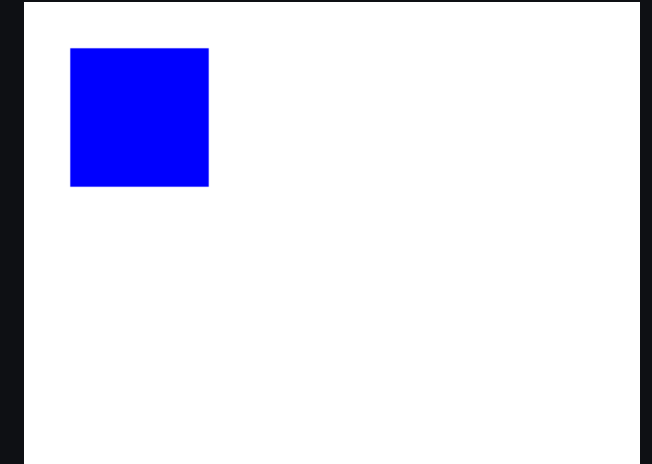
2.5.8

Prvi prozor: nacrtaj jednom, pa čekaj

```
import pygame as pg

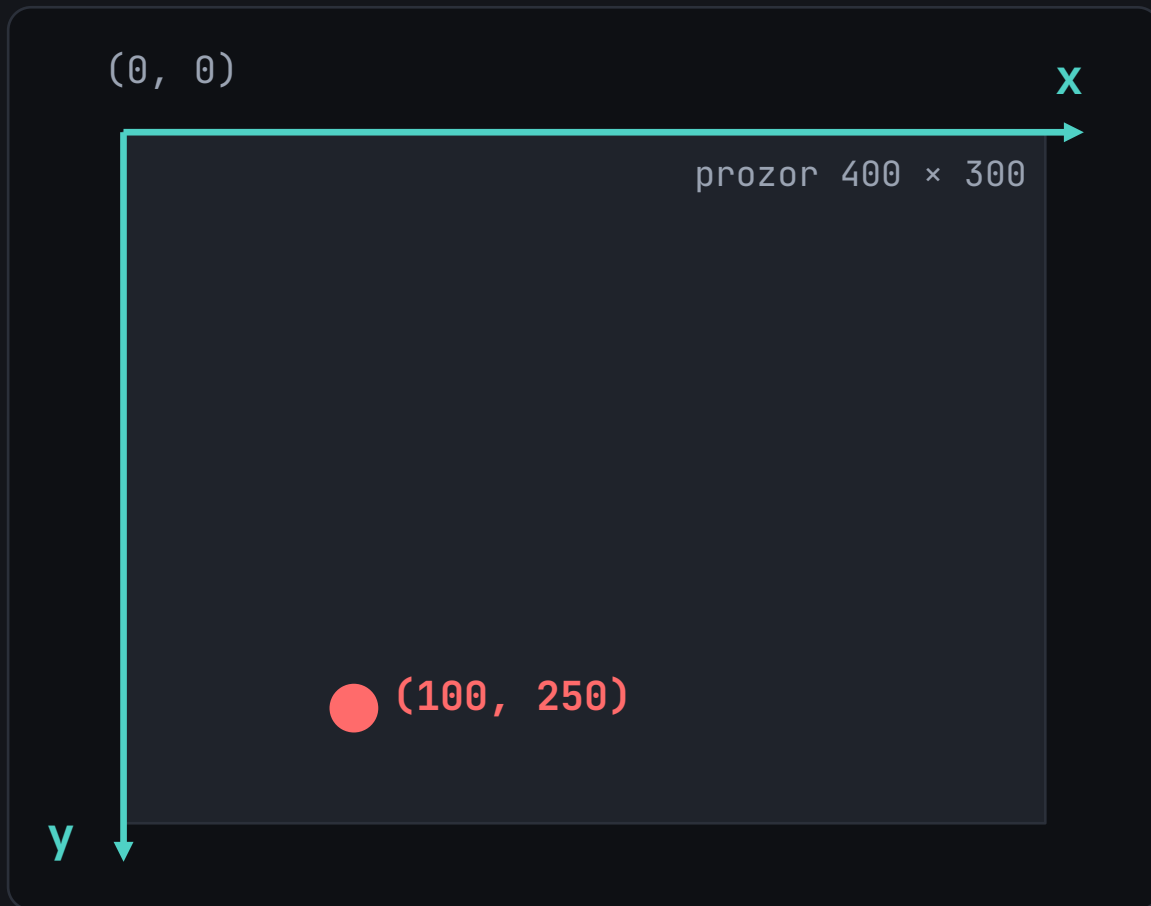
pg.init()
(SIRINA, VISINA) = (400, 300)
ekran = pg.display.set_mode((SIRINA, VISINA))
pg.display.set_caption("Naša prva igra")
ekran.fill("white")
pg.draw.rect(ekran, "blue", (30, 30, 90, 90))
pg.display.flip()

radi = True
while radi:
    dogadjaj = pg.event.wait()    # čeka događaj
    if dogadjaj.type == pg.QUIT:  # klik na X
        radi = False
pg.quit()
```



- flip() prikaže sve što je nacrtano.
- event.wait() čeka sledeći događaj.
- Klik na X šalje događaj QUIT.

Početak je gore levo, a boja ima tri broja



"white"

ime boje (na engleskom)



(64, 85, 135)

crvena, zelena, plava: od 0 do 255

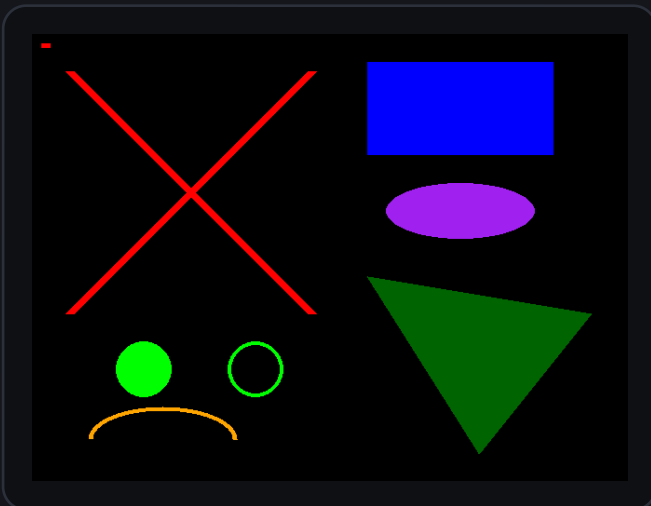


"#405587"

isti brojevi u hex zapisu

Modul draw: jedna funkcija za svaki oblik

```
pg.draw.rect(ekran, boja, (x, y, sirina, visina), debljina)
pg.draw.circle(ekran, boja, (x, y), poluprecnik, debljina)
pg.draw.line(ekran, boja, (x1, y1), (x2, y2), debljina)
pg.draw.polygon(ekran, boja, [(x1, y1), (x2, y2), ...], debljina)
pg.draw.ellipse(ekran, boja, (x, y, sirina, visina), debljina)
pg.draw.arc(ekran, boja, (x, y, sirina, visina), od, do, debljina)
```



- $(x, y, sirina, visina)$: x i y su gornji levi ugao.
- $debljina$ 0 ili bez debljine: pun oblik.
- Koordinate i poluprečnik smeju da budu realni brojevi.
- arc : uglovi od i do su u radijanima (3.14 je pola kruga).

Gde je donji desni ugao pravougaonika?

```
# prozor je 400 x 300  
pg.draw.rect(ekran, "red", (100, 50, 80, 40))
```

Pravougaonik je 80 piksela širok i 40 visok.

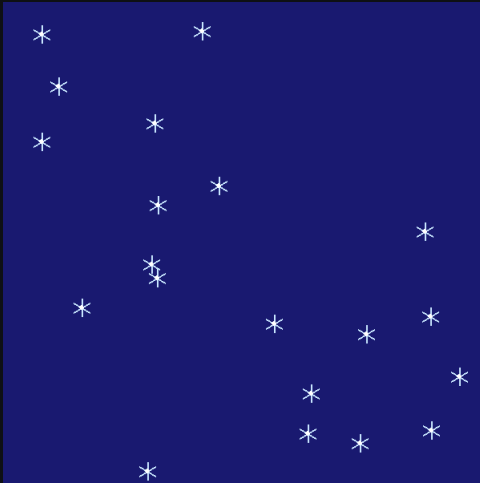
A tačno
(180, 90)

B
(80, 40)

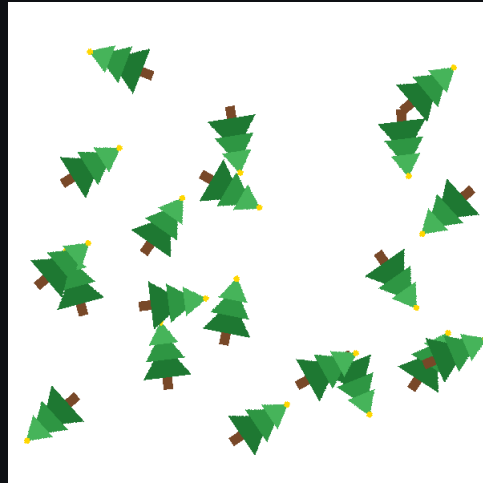
C
(100, 50)

D
(180, 50)

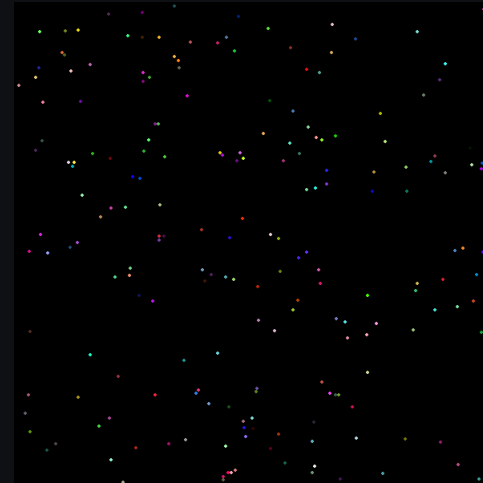
Stari zadaci, sada u primerima



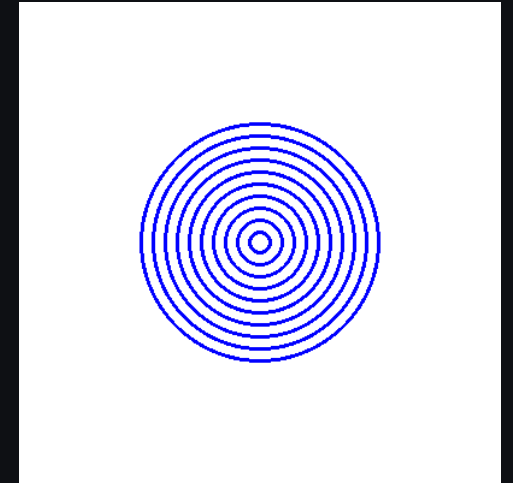
20 pahuljica



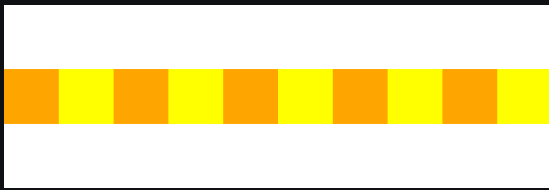
rotirane jelkice



200 tačkica



krugovi



kvadrati

Svi su iz starih beleški, ispravljani: ništa ne izlazi iz prozora, ugao je od 0 do 359, prvi kvadrat je narandžast.

Površina je platno

Crtamo na svojoj površini, pa je lepimo gde hoćemo

Sve što vidimo je površina zalepljena na prozor

nebo

Surface 640 × 360
`image.load`

blit

jelkica

Surface 60 × 88
`Surface((60, 88))`

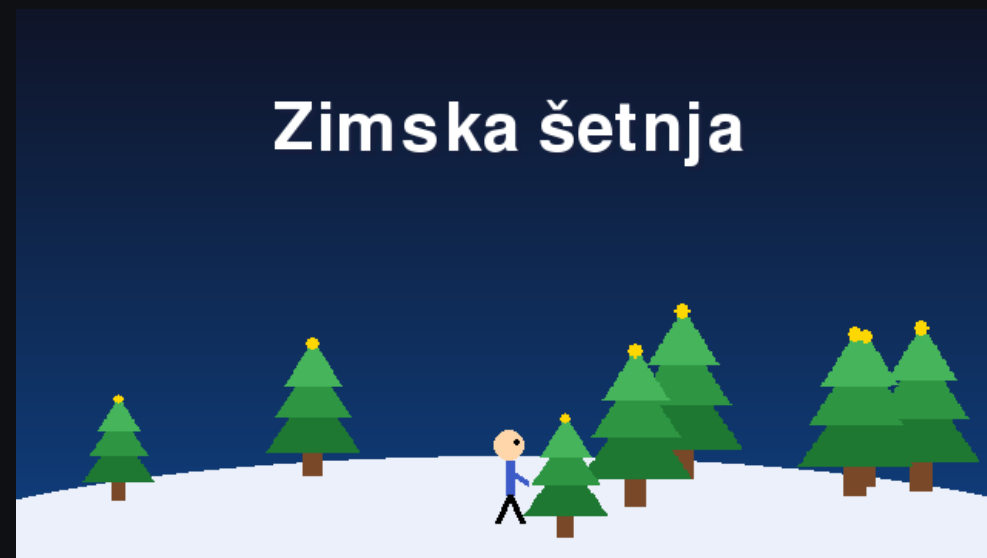
blit

natpis

Surface sa tekstom
`font.render`

blit

ekran (prozor)

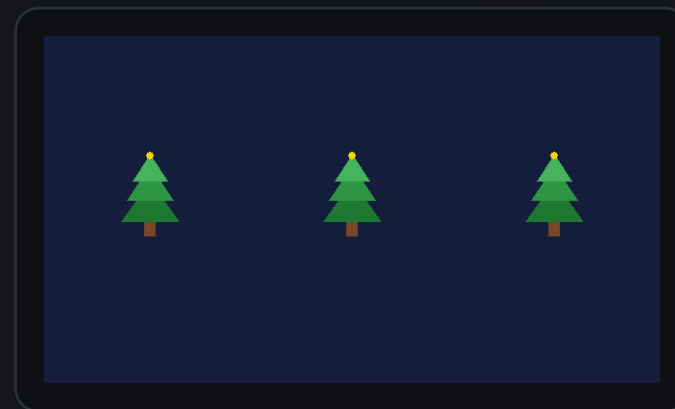


i ekran je površina: `set_mode` je vraća

Jedan crtež na svojoj površini, tri otiska

```
jelkica = pg.Surface((60, 88), pg.SRCALPHA)
pg.draw.rect(jelkica, STABLO, (24, 70, 12, 18))
pg.draw.polygon(jelkica, TAMNA, [(30, 30), (60, 72), (0, 72)])
...                               # još dva trougla i zvezda

for x in (80, 290, 500):         # jedan crtež, tri otiska
    ekran.blit(jelkica, (x, 120))
```



- SRCALPHA: pozadina je providna.
- Koordinate u crtežu su unutar jelkice: (0, 0) je njen gornji levi ugao.
- Ovako 06_01 pravi jelkica.png.

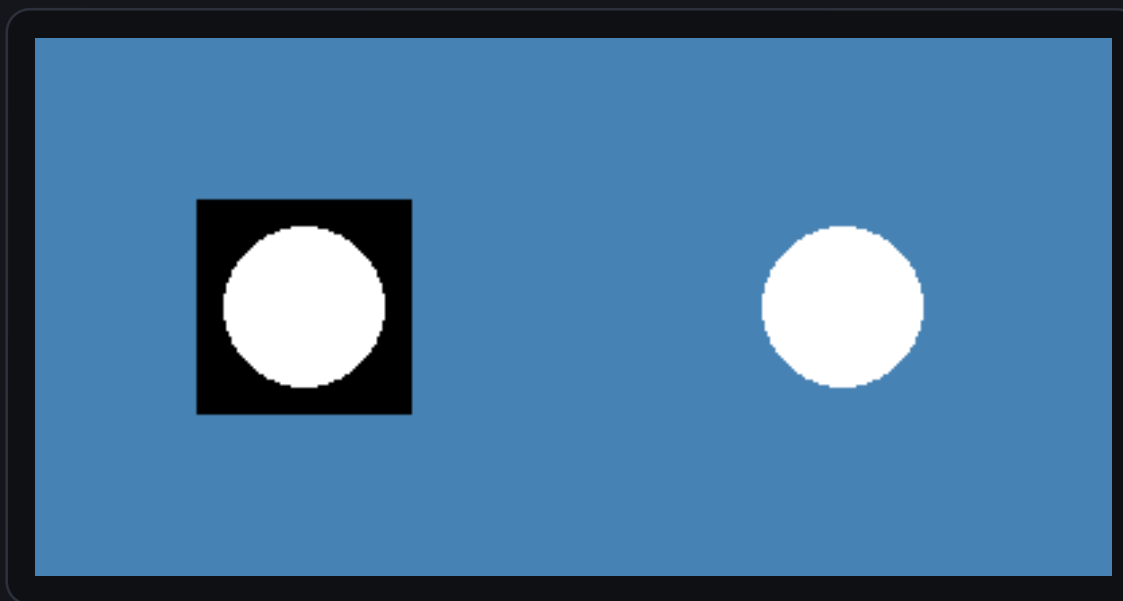
Bez SRCALPHA nova površina je crna

Bez SRCALPHA

```
krug = pg.Surface((80, 80))  
pg.draw.circle(krug, "white", (40, 40), 30)
```

Sa SRCALPHA

```
krug = pg.Surface((80, 80), pg.SRCALPHA)  
pg.draw.circle(krug, "white", (40, 40), 30)
```



Gde je na ekranu centar belog kruga?

```
platno = pg.Surface((100, 100))  
pg.draw.circle(platno, "white", (10, 10), 5)  
ekran.blit(platno, (50, 20))
```

Platno lepimo na ekran tako da mu gornji levi ugao bude u (50, 20).

A

(10, 10)

B

(50, 20)

C

tačno

(60, 30)

D

(40, 10)

Slike

Učitavamo jednom, pripremamo za prozor, postavljamo okvirom

convert briše providnost, convert_alpha je čuva

```
import pygame as pg

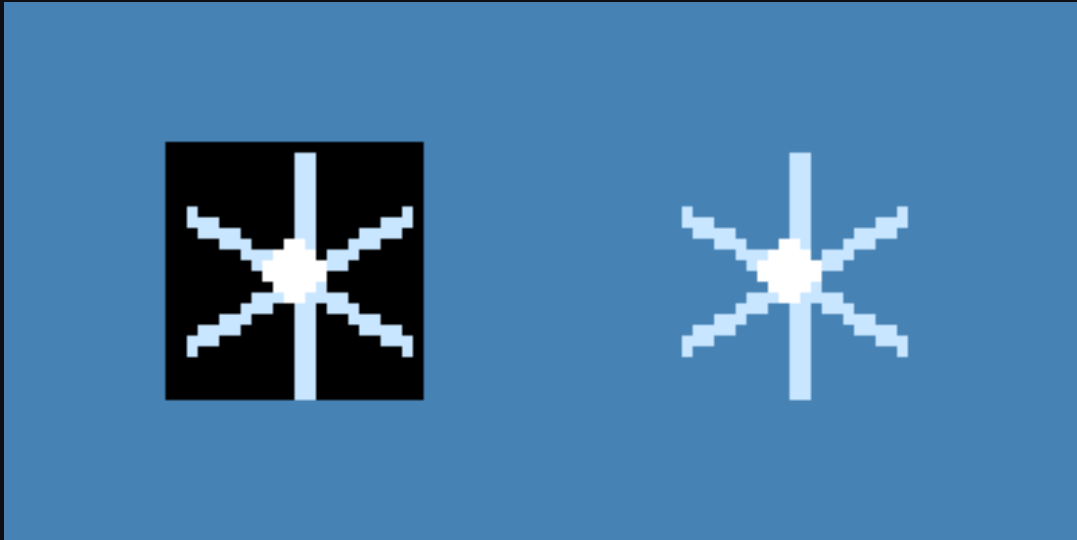
pg.init()
ekran = pg.display.set_mode((640, 360)) # prvo prozor!
slika = pg.image.load("pahuljica.png")
foto = slika.convert() # bez providnosti
sprajt = slika.convert_alpha() # sa providnošću
print(slika.get_at((0, 0)))
print(foto.get_at((0, 0)))
print(sprajt.get_at((0, 0)))
```

ISPIS: PİKSEL (0, 0)

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

Četvrti broj je alfa: 0 je potpuno providno, 255 neprovidno.

Pozadina dobija `convert`, a sličica `convert_alpha`



levo `convert()`, desno `convert_alpha()`

`convert()`

pozadine i fotografije, bez providnih delova

`convert_alpha()`

sličice sa providnom pozadinom: likovi, jelkice

pre oba

mora da postoji prozor (`set_mode`)

BEZ PROZORA

```
pygame.error: cannot convert without pygame.display initialized
```

Čudovište: nema datoteke

```
import pygame as pg

pg.init()
ekran = pg.display.set_mode((640, 360))
jelka = pg.image.load("jelka.png")
```

PORUKA

```
FileNotFoundError:
No file 'jelka.png' found
in working directory '...'.
```

Kako izgleda

FileNotFoundError već u redu sa image.load, pre nego što se išta nacрта.

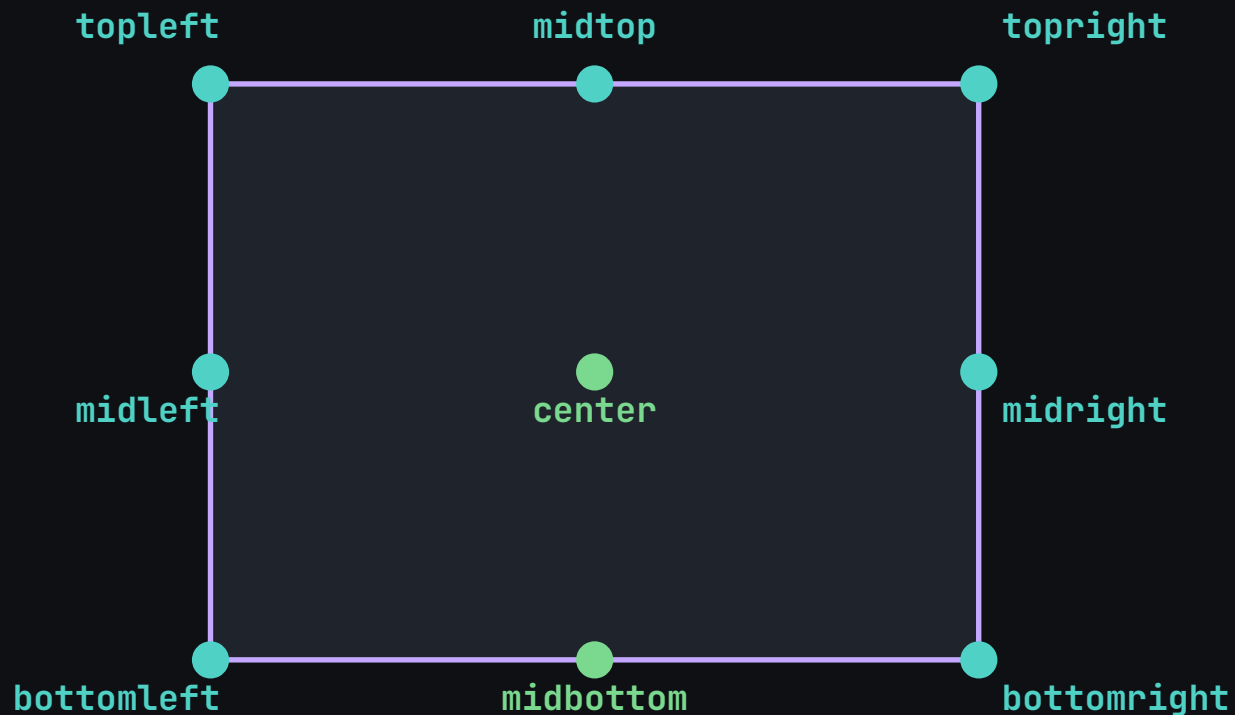
Zašto se pojavljuje

Slika nije u radnom folderu, ime je pogrešno napisano ili program nije pokrenut iz foldera sa slikama.

Kako se pobeđuje

Proverite ime (i velika slova), stavite sliku pored programa i pokrenite program iz tog foldera.

Okvir slike zna devet tačaka



```
get_rect(center=(x, y))
```

slika je centrirana u tački

```
get_rect(midbottom=(x, y))
```

slika „stoji“ na tački: jelkica na snegu

```
blit(slika, okvir)
```

blit prima i okvir, ne samo (x, y)

get_rect računa ugao umesto nas

```
import pygame as pg

jelkica = pg.Surface((60, 88))
okvir = jelkica.get_rect(center=(320, 180))
print(okvir.topleft)
okvir = jelkica.get_rect(midbottom=(320, 340))
print(okvir.topleft)
print(okvir.center, okvir.right)
```

ISPIS

```
(290, 136)
(290, 252)
(320, 296) 350
```

Stari način (stare beleške):

```
x = (SIRINA - slika.get_width()) / 2
y = (VISINA - slika.get_height()) / 2
ekran.blit(slika, (x, y))
```

Isto, preko okvira:

```
centar = ekran.get_rect().center
okvir = slika.get_rect(center=centar)
ekran.blit(slika, okvir)
```

Šta ispisuje ovaj program?

```
jelkica = pg.Surface((60, 88))  
okvir = jelkica.get_rect(midbottom=(200, 300))  
print(okvir.topleft)
```

Jelkica je 60 piksela široka i 88 visoka.

A

(200, 300)

B

tačno

(170, 212)

C

(170, 256)

D

(140, 212)

scale pravi novu sliku, a original ostaje

```
import pygame as pg

jelkica = pg.Surface((60, 88))
for k in (0.5, 1, 1.5):
    nova = pg.transform.scale(jelkica, (60 * k, 88 * k))
    print(k, nova.get_size())
print("original:", jelkica.get_size())
```

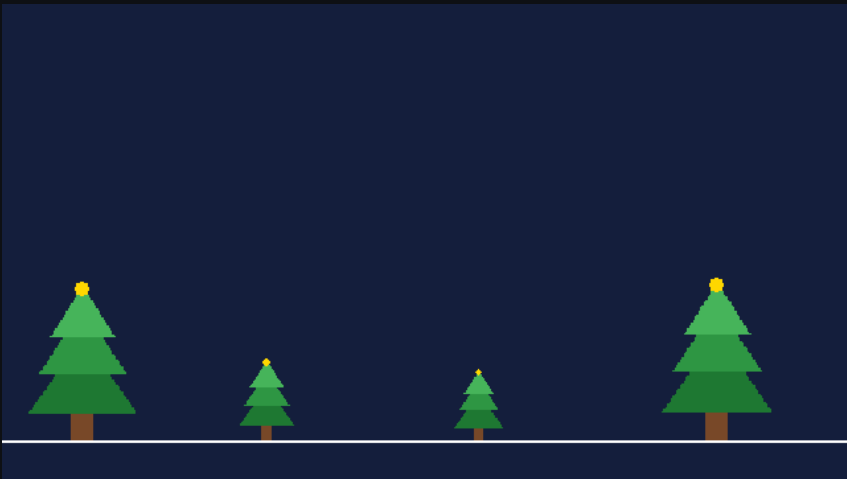
ISPIS

```
0.5 (30, 44)
1 (60, 88)
1.5 (90, 132)
original: (60, 88)
```

Širinu i visinu množimo istim brojem k, pa jelkica ostaje iste razmere. Rezultat scale moramo da sačuvamo u promenljivu: original se ne menja. Ista pravila važe za rotate iz PZU1.

Jelkice različite veličine stoje na istom tlu

```
for x in (60, 200, 360, 540):  
    k = random.uniform(0.6, 1.4)          # koliko puta veća  
    slika = pg.transform.scale(JELKICA, (60 * k, 88 * k))  
    okvir = slika.get_rect(midbottom=(x, 330))  
    ekran.blit(slika, okvir)
```



- `random.uniform(0.6, 1.4)`: `k` je realan broj.
- `midbottom = (x, 330)`: donja ivica je uvek na liniji tla.
- Bez `midbottom` bi male jelkice „lebdele“.

Tekst

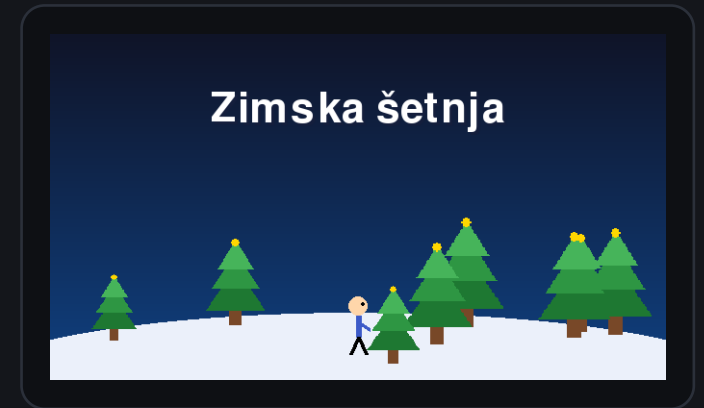
Tekst je sličica koju pravi font

Naslov sa senkom: isti tekst, dve boje

```
font = pg.font.Font(None, 64) # None: font iz pygame-a
natpis = font.render("Zimska šetnja", True, "white")
senka = font.render("Zimska šetnja", True, (20, 30, 60))

centar = (SIRINA // 2, 80)
ekran.blit(senka, senka.get_rect(center=(323, 83)))
ekran.blit(natpis, natpis.get_rect(center=centar))
```

- Font(None, 64): font koji dolazi sa pygame-om, veličine 64.
- render(tekst, True, boja): True znači glatke ivice slova.
- Senka je ista sličica, 3 piksela desno i dole, nacrtana prva.

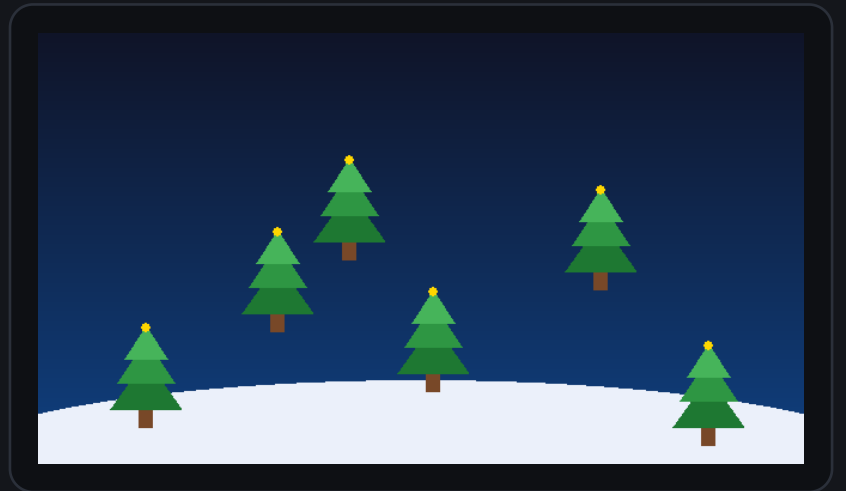


Petlja vođena događajima

Program čeka i crta samo kad se nešto desi

Klik sadi jelkicu: crtamo samo posle događaja

```
ekran.blit(NEBO, (0, 0))
pg.display.flip()
radi = True
while radi:
    dogadjaj = pg.event.wait()
    if dogadjaj.type == pg.QUIT:
        radi = False
    elif dogadjaj.type == pg.MOUSEBUTTONDOWN:
        okvir = JELKICA.get_rect(midbottom=dogadjaj.pos)
        ekran.blit(JELKICA, okvir)
        pg.display.flip()          # samo posle klika
```



- dogadjaj.pos: mesto klika.
- Između klikova program ne radi ništa.
- Šest klikova, šest jelkica.

Zimska šetnja, korak 1: zimska scena

Cilj: za 25 minuta mirna zimska scena koju svaki klik slaže iznova.

- 1 Učitavamo nebo, jelkicu i Čikicu, posle set_mode
- 2 Na kopiji neba: osam jelkica, svaka svoje veličine
- 3 Jelkice stoje na snegu: midbottom
- 4 Čikica na sredini, naslov sa senkom
- 5 Petlja vođena događajima: klik daje novu scenu

KAKO IZGLEDA NA KRAJU



nebo.png, jelkica.png i desno.png pravi program
06_01

Scena se slaže na kopiji neba

```
def napravi_scenu():  
    scena = NEB0.copy() # crtamo na kopiji neba  
    for y in sorted(random.sample(range(300, 345), 8)):  
        k = random.uniform(0.6, 1.3)  
        slika = pg.transform.scale(JELKICA, (60 * k, 88 * k))  
        x = random.randint(30, SIRINA - 30)  
        scena.blit(slika, slika.get_rect(midbottom=(x, y)))  
    scena.blit(CIKICA, CIKICA.get_rect(midbottom=(320, 335)))  
    naslov(scena, "Zimska šetnja")  
    return scena
```

NEB0.copy()

nova površina; original ostaje čist

random.sample

[308, 336, 304, 316]: 4 različita broja

sorted(...)

dalje (više) jelkice prve, bliže preko njih

Petlja čeka klik i crta novu scenu

```
ekran.blit(napravi_scenu(), (0, 0))
pg.display.flip()
radi = True
while radi:
    dogadjaj = pg.event.wait()
    if dogadjaj.type == pg.QUIT:
        radi = False
    elif dogadjaj.type == pg.MOUSEBUTTONDOWN: # nova scena
        ekran.blit(napravi_scenu(), (0, 0))
        pg.display.flip()
pg.quit()
```

napravi_scenu()

vraća gotovu površinu, celu scenu

MOUSEBUTTONDOWN

pritisak tastera miša

flip()

tek posle crtanja, jednom

Pronađi grešku: sve jelkice su iste

```
for x in (60, 200, 360, 540):  
    k = random.uniform(0.6, 1.4)  
    pg.transform.scale(JELKICA, (60 * k, 88 * k))  
    okvir = JELKICA.get_rect(midbottom=(x, 330))  
    ekran.blit(JELKICA, okvir)
```



Svaka jelkica dobija svoj k , a na ekranu su sve iste veličine. Program ne javlja grešku.

scale vraća NOVU sliku, a ovde je rezultat bačen. Ispravka: `slika = pg.transform.scale(JELKICA, ...)`, pa okvir i blit sa promenljivom slika.

Izaberite nivo i počnite

LEVEL 1

Potpis u uglu

Na zimskom nebu napišite svoje ime u donjem desnom uglu, 10 piksela od ivica.

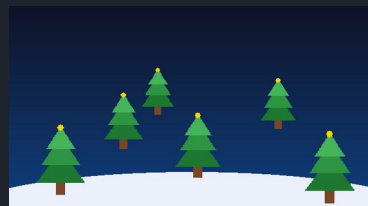


Pomoć: bottomright.

LEVEL 2

Jelkice u perspektivi

Klik sadi jelkicu. Što je klik niže, jelkica je veća, kao da je bliže.



Pomoć: k zavisi od y.

LEVEL 3

Piksel-art

Nacrtajte zimsku scenu na površini 32×18 , pa je uvećajte 20 puta.



Pomoć: set_at i scale.

Sledeći put sneg pada

```
sat = pg.time.Clock()
while radi:
    ...                                # događaji
    dt = sat.tick(60) / 1000
    for p in pahuljice:
        p.azuriraj(dt)                # pada brzina × dt
    ekran.blit(POZADINA, (0, 0))
    ...                                # crtanje
    pg.display.flip()
```

U temi 07 naša scena oživljava. Petlja više ne čeka događaj, nego crta 60 slika u sekundi, a pahuljice padaju brzinom u pikselima u sekundi, pomnoženom sa dt.

Rezime

površina

```
s = pg.Surface((60, 88), pg.SRCALPHA)
ekran.blit(s, (x, y))
```

Crtamo na njoj, pa je lepimo.

slike

```
pozadina: .convert()
sličica: .convert_alpha()
```

Uvek posle set_mode.

okvir

```
okvir = s.get_rect(midbottom=(x, y))
ekran.blit(s, okvir)
```

Centar, ivice, uglovi.

scale i tekst

```
nova = pg.transform.scale(s, (w, h))
font.render(tekst, True, boja)
```

Oba vraćaju novu površinu.

Pitanje pre odlaska

```
slika = pg.Surface((40, 64))  
okvir = slika.get_rect(midbottom=(100, 200))  
print(okvir.top, okvir.left)
```

Čikica je 40 piksela širok i 64 visok. Šta ispisuje program?

Pošaljite dva broja pre nego što izađete.



Literatura

- Dokumentacija za pygame-ce, moduli display, draw, image, transform, font i klase Surface i Rect: pyga.me/docs
- Kafle, S. Learning Python by Building Games, Packt, 2019 (poglavlja o PyGame-u, posebno 11, 12 i 13)
- Sweigart, A. Invent Your Own Computer Games with Python, 4. izdanje, No Starch Press, 2016 (besplatno na inventwithpython.com)

Dodatna literatura i zanimljivi sajtovi

pygame-ce: Surface

pyga.me/docs/ref/surface.html

Površina: blit, convert, convert_alpha, get_rect i zastavica SRCALPHA.

pygame-ce: Rect

pyga.me/docs/ref/rect.html

Sve tačke okvira (center, midbottom, topright...) i kako se pomeraju.

pygame-ce: transform

pyga.me/docs/ref/transform.html

scale, rotate, flip: svaka funkcija vraća novu površinu.

pygame-ce: font

pyga.me/docs/ref/font.html

Font(None, ...), SysFont i render, koji pravi sličicu sa tekstom.

Lospec: palete boja

lospec.com/palette-list

Biblioteka gotovih paleta boja za piksel-art.

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