

Za radoznale: Događaji i unos

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. Ponavljanje tastera: set_repeat

Na času smo videli da KEYDOWN stiže samo jednom, koliko god dugo držimo taster. Funkcija `pg.key.set_repeat(kasnjenje, razmak)` to menja: posle kašnjenja u milisekundama KEYDOWN počne da stiže ponovo, na svakih „razmak“ milisekundi, kao kad držimo slovo u programu za pisanje teksta. `get_repeat` kaže kakvo je podešavanje.

```
import pygame as pg

pg.init()
pg.display.set_mode((200, 200))
print("pre:", pg.key.get_repeat())
pg.key.set_repeat(400, 50)      # posle 0,4 s, pa na svakih 0,05 s
print("posle:", pg.key.get_repeat())
pg.key.set_repeat()             # bez argumenata: ponavljanje isključeno
print("opet:", pg.key.get_repeat())
```

Ispis:

```
pre: (0, 0)
posle: (400, 50)
opet: (0, 0)
```

Podrazumevano je (0, 0): nema ponavljanja. Sa `set_repeat(400, 50)` taster koji držimo pošalje prvi KEYDOWN odmah, drugi posle 0,4 s, a zatim po jedan na svakih 0,05 s.

Zašto ovo nismo radili na času: za kretanje lika ponavljanje nije dobro: prvih 0,4 s lik stoji, pa krene trzajem, a brzina zavisi od razmaka ponavljanja, a ne od dt. Za kretanje koristimo `get_pressed`; `set_repeat` je za unos teksta i za menije u kojima se strelicom ide kroz stavke.

Fajl: `primeri/za_radoznale/08_r1_set_repeat.py`

2. Sopstveni događaj sa podacima

Tajmer šalje sopstveni događaj bez podataka. Mi možemo i sami da pošaljemo sopstveni događaj, sa podacima koje izaberemo, funkcijom `pg.event.post`. Tako jedan deo igre može da „javi“ drugom da se nešto desilo, na primer da je pahuljica uhvaćena i koliko poena vredi, a da ti delovi ne znaju jedan za drugi.

```
import pygame as pg

pg.init()
pg.display.set_mode((200, 200))
pg.event.clear()
UHVACENA = pg.event.custom_type()

# negde u kodu za pahuljice:
pg.event.post(pg.event.Event(UHVACENA, poeni=10, boja="bela"))
pg.event.post(pg.event.Event(UHVACENA, poeni=50, boja="zlatna"))

# u petlji igre:
ukupno = 0
for dogadjaj in pg.event.get():
    if dogadjaj.type == UHVACENA:
        ukupno += dogadjaj.poeni
        print("uhvaćena", dogadjaj.boja, "pahuljica")
print("poeni:", ukupno)
```

Ispis:

```
uhvaćena bela pahuljica
uhvaćena zlatna pahuljica
poeni: 60
```

Podaci se zadaju kao imenovani argumenti pri pravljenju događaja, a čitaju se tačkom, isto kao key ili pos kod događaja koje šalje biblioteka.

Zašto ovo nismo radili na času: u maloj igri sve je u jednom programu, pa je jednostavnije da odmah povećamo poene. Ovaj način se isplati u većim igrama, sa više modula (tema 12).

Fajl: `primeri/za_radoznale/08_r2_sopstveni_dogadjaj.py`

3. Tajmer koji pošalje samo jedan događaj

Treći argument funkcije `set_timer`, `loops`, kaže koliko događaja tajmer pošalje. Sa `loops = 1` tajmer pošalje jedan događaj i sam se isključi. Tako se pravi „uradi ovo za 2 sekunde“, a da prozor sve vreme reaguje, kao u rešenju igre slova.

```
import pygame as pg

pg.init()
pg.display.set_mode((200, 200))
KRAJ = pg.event.custom_type()
pg.time.set_timer(KRAJ, 200, 1) # jedan događaj, posle 0,2 s
sat = pg.time.Clock()
vreme = 0
stiglo = 0
while vreme < 1: # gledamo celu sekundu
    vreme += sat.tick(60) / 1000
    for dogadjaj in pg.event.get():
        if dogadjaj.type == KRAJ:
            stiglo += 1
print("stiglo događaja:", stiglo)
```

Ispis:

```
stiglo događaja: 1
```

Za celu sekundu stigao je samo jedan događaj, iako bi tajmer bez trećeg argumenta za to vreme poslao pet događaja (na svakih 0,2 s).

Zašto ovo nismo radili na času: na času smo tajmer koristili za ponavljanje (nova pahuljica na svakih 0,7 s); jednokratni tajmer je potreban tek kad igra ima kraj, o čemu pričamo u temi 12.

Fajl: `primeri/za_radoznaLe/08_r3_jednokratni_tajmer.py`