

001 Štampanje teksta

Modul `pygame.font` daje opcije štampanja font-zasnovanog teksta na ekran u grafičkom modu.

Klasa koja pravi font koji se može štampati je `pygame.font.Font`.

Po difoltu, priključivanje `None` kao naziv fonta čini da konstruktor `pygame.font.Font()` učita difolt Pygame font.

Drugi parametar konstruktora je veličina fonta u tačkama.

Sledeća linija kreira difolt font sa veličinom od 30 tačaka:

```
moj_font = pygame.font.Font(None, 30)
```

Može se i specificirati naziv fonta kao osnovnog za igru:

```
moj_font = pygame.font.Font("Arial", 30)
```

Za štampanje teksta, `font.render()` funkcija kreira bitmap sa upisanim tekstem, koji se zatim iscrtava na ekranu korišćenjem `ekran.blit()`.

```
slika = font.render(tekst, True, (255,255,255))
```

```
ekran.blit(slika, (100, 100))
```

002 Klasa Kviz

Sors kod glavnog dela programa u igri je najviše odgovoran za dobijanje ulaza sa tastature u osvežavanje ekrana.

Najveći deo koda igre se nalazi u novoj klasi koja se naziva Kviz.

Prvo se importuju potrebni moduli:

```
import sys, pygame
from pygame.locals import *
```

Konstruktor `__init__()` ima naziv fajla kao parametar koji mu se dodaje, koji sadrži podatke o kvizu.

Podaci se učitavaju preko `fajl.readlines()` poziva funkcije, a zatim se ti podaci koriste u listi tokom igre.

Postoji više promenljivih (osobnosti, properties) u klasi Kviz, kojima se kontroliše logika igre.

Sva logika se izvodi metodama u klasi Kviz, a ne u glavnom programu.

program Kviz

```
import sys, pygame
from pygame.locals import *

class Kviz(object):
    def __init__(self, ime_fajla):
        self.data = []
        self.trenutno = 0
        self.ukupno = 0
        self.tacno = 0
        self.rez = 0
        self.dobio_bodove = False
        self.greska = False
        self.los_odgovor = 0
        self.boje = [bela, bela, bela, bela]

        #citanje podataka iz fajla
```

```

f = open(ime_fajla, "r")
kviz_podaci = f.readlines()
f.close()

#prebrojavanje i uredjivanje podataka
for linija_teksta in kviz_podaci:
    self.data.append(linija_teksta.strip())
    self.ukupno += 1

def prikazi_pitanje(self):
    stampaj_tekst(font1, 210, 5, "KVIZ")
    stampaj_tekst(font2, 190, 500 - 20, "Pritisni (1-4) za odgovor", ljubicasti)
    stampaj_tekst(font2, 530, 5, "REZULTAT", ljubicasti)
    stampaj_tekst(font2, 550, 25, str(self.rez), ljubicasti)

    #dobijanje tacnog odgovora iz podataka (prvo)
    self.tacno = int(self.data[self.trenutno + 5])

    #prikaz pitanja
    pitanje = self.trenutno // 6 + 1
    stampaj_tekst(font1, 5, 80, "PITANJE " + str(pitanje))
    stampaj_tekst(font2, 20, 120, self.data[self.trenutno], zuta)

    #desavanja ako je odgovor tacan
    if self.dobio_bodove:
        self.boje = [bela,bela,bela,bela]
        self.boje[self.tacno - 1] = zelena
        stampaj_tekst(font1, 230, 380, "TACNO!", zelena)
        stampaj_tekst(font2, 170, 420, "Pritisni ENTER za sledece pitanje",
            zelena)
    elif self.greska:
        self.boje = [bela,bela,bela,bela]
        self.boje[self.los_odgovor - 1] = crvena
        self.boje[self.tacno - 1] = zelena
        stampaj_tekst(font1, 220, 380, "NETACNO!", crvena)
        stampaj_tekst(font2, 170, 420, "Pritisni ENTER za sledece pitanje",
            crvena)

    #prikaz odgovora
    stampaj_tekst(font1, 5, 170, "ODGOVORI")
    stampaj_tekst(font2, 20, 210, "1 - " + self.data[self.trenutno + 1],
        self.boje[0])
    stampaj_tekst(font2, 20, 240, "2 - " + self.data[self.trenutno + 2],
        self.boje[1])
    stampaj_tekst(font2, 20, 270, "3 - " + self.data[self.trenutno + 3],
        self.boje[2])
    stampaj_tekst(font2, 20, 300, "4 - " + self.data[self.trenutno + 4],
        self.boje[3])

def rukovanje_ulazom(self, broj):
    if not self.dobio_bodove and not self.greska:
        if broj == self.tacno:
            self.dobio_bodove = True
            self.rez += 1
        else:
            self.greska = True
            self.los_odgovor = broj

def sledece_pitanje(self):
    if self.dobio_bodove or self.greska:
        self.dobio_bodove = False
        self.greska = False

```

```

        self.tacno = 0
        self.boje = [bela, bela, bela, bela]
        self.trenutno += 6
        if self.trenutno >= self.ukupno:
            self.trenutno = 0

def stampaj_tekst(font, x, y, tekst, boja=(255, 255, 255), senka = True):
    if senka:
        slika_teksta = font.render(tekst, True, (0, 0, 0))
        ekran.blit(slika_teksta, (x - 2, y - 2))

        slika_teksta = font.render(tekst, True, boja)
        ekran.blit(slika_teksta, (x, y))

#glavni deo programa
pygame.init()
ekran = pygame.display.set_mode((600,500))
pygame.display.set_caption("Kviz")
font1 = pygame.font.Font(None, 40)
font2 = pygame.font.Font(None, 24)
bela = 255,255,255
purpurna = 0,255,255
zuta = 255,255,0
ljubicasta = 255,0,255
zelena = 0,255,0
crvena = 255,0,0

#ucitavanje fajla sa pitanjima i odgovorima
kviz = Kviz("kviz_data.txt")

#glavna petlja
while True:
    for dogadjaj in pygame.event.get():
        if dogadjaj.type == QUIT:
            sys.exit()
        elif dogadjaj.type == KEYUP:
            if dogadjaj.key == pygame.K_ESCAPE:
                sys.exit()
            elif dogadjaj.key == pygame.K_1:
                kviz.rukovanje_ulazom(1)
            elif dogadjaj.key == pygame.K_2:
                kviz.rukovanje_ulazom(2)
            elif dogadjaj.key == pygame.K_3:
                kviz.rukovanje_ulazom(3)
            elif dogadjaj.key == pygame.K_4:
                kviz.rukovanje_ulazom(4)
            elif dogadjaj.key == pygame.K_RETURN:
                kviz.sledece_pitanje()

#cisti ekran
ekran.fill((0,0,200))

#prikaz pitanja
kviz.prikazi_pitanje()

#updejtovanje prikaza na ekranu
pygame.display.update()

```

003 Tekst fajl kviza

Posle učitavanja podataka, podaci se uređuju (parsuju iz objekta lista pod imenom kviz_podaci) i kopiraju linija po linija u novu listu Kviz.data.

Na ovaj način se mogu skidati prazna mesta na svakoj od linija.

Podaci za kviz imaju samo pet pitanja, tema je astronomija.

Fajla sa ulaznim podacima je kviz_data.txt:

Kako se zove cetvrta planeta od Sunca?

Saturn

Mars

Zemlja

Venera

2

Koja planeta ima najvise satelita u Suncevom sistemu?

Uran

Saturn

Neptun

Jupiter

4

Koliki je precnik Sunca, otprilike?

65.000 milja

45.000.000 milja

1.000.000 milja

825.000 milja

3

Kolika je prosečna udaljenost Zemlje od Sunca?

13.000.000 milja

93.000.000 milja

250.000 milja

800.000 milja

2

Sta izaziva talase na Zemlji?

Mesec

Sunce

Zemljina kora

Kiseonik

1

004 Opis rada programa

Najveći deo posla se obavlja u metodi `Kviz.prikaz_pitanja()`.

Ovde se crta ceo ekran igre: naslov, tekst na dnu, rezultat, pitanja, odgovori i bojenje odgovora u zavisnosti od korisnikovog unosa.

Kada korisnik unese tačan odgovor, odgovor je zelen.

Ako korisnik unese netačan odgovor, štampa se u crvenom, a tačan odgovor je u zelenom.

Bojenje je pojednostavljeno korišćenjem liste sa četiri boje kojima se boji tekst sa svakim od odgovora.

Ključ indeksiranja iz jednog zapisa pitanja u drugo je polje `Kviz.trenutno`.

Program čeka odgovor od korisnika koji pritiska dugmad 1, 2, 3 ili 4 u zavisnosti od željenog odgovora.

Kada korisnik izabere odgovor, `Kviz.rukovanje_ulazom()` metod se poziva.

Ako odgovor nije već izabran, korisnički ulaz će se uporediti za tačnim odgovorom i ili `self.dobio_bodove` ili `self.pogresio` će biti postavljeni na `True`.

Igra odgovara na ova dva flega i postavlja se u stanje čekanja sve dok korisnik ne pritisne ENTER za nastavak na sledeće pitanje.

Posle toga poziva se metod `Kviz.sledece_pitanje()`.

Ako je igra u stanju čekanja između pitanja, onda su flegovi resetovani, boje su resetovane, i igra skače na sledeće pitanje.

Pošto postoje 6 linija po pitanju u fajlu sa podacima, `Kviz.trenutno` polje se inkrementira za 6 za preskok na sledeće pitanje.

Glavni deo koda sadrži pomoćnu funkciju `odstampaj_tekst()` koji ima kao prvi parametar objekat fonta.

Sledeće je kod za inicijalizaciju koji kreira Pygame prozor i vrši postavke za igru.

Zatim se kreira kviz objekat (korišćenjem `Kviz` klase) i fajl sa podacima `kviz_podaci.txt` se učitava.

Petlja `while` čini da se igra odvija; može se smatrati i petljom igre.

Najveći deo koda je uključen u dobijanje korisničkog ulaza sa događajima aktiviranim preko tastature.

Zatim se samo očisti ekran i poziva `kviz.prikazi_pitanje()` za updejtovanje trenutnog stanja igre.