ragazze DIGITALI

IDEE PER UN FUTURO SMART

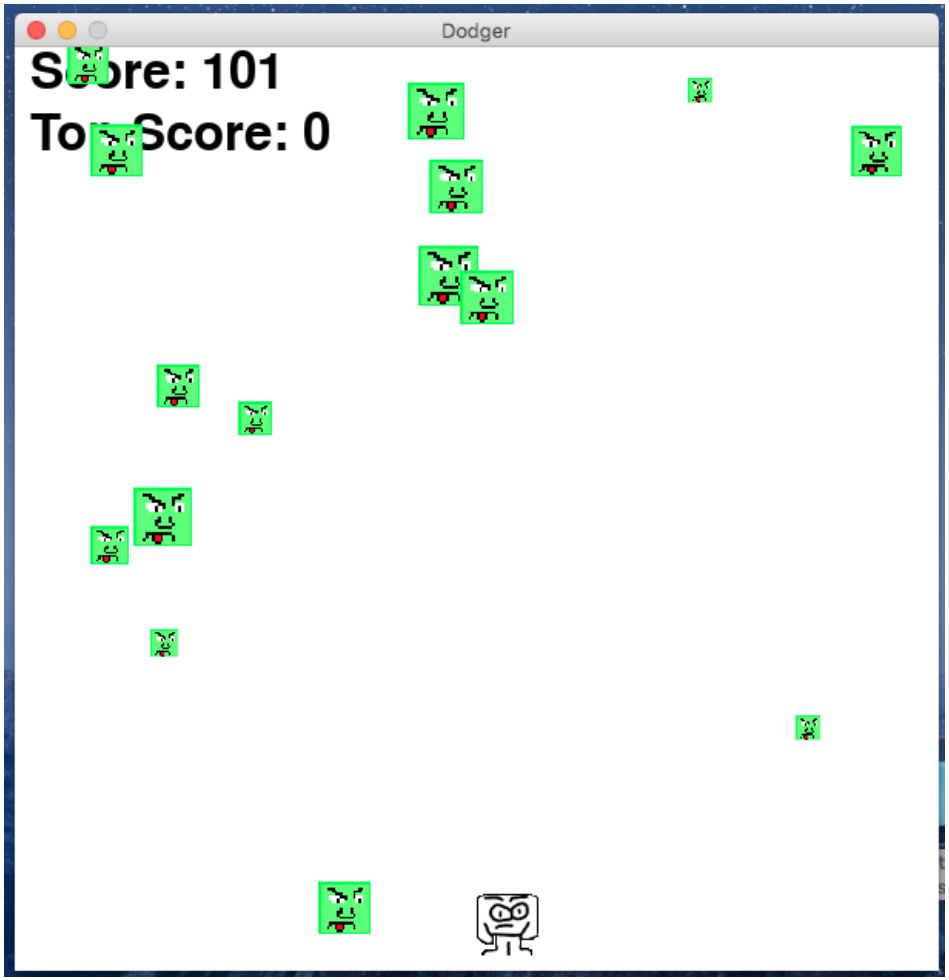
Cosa faremo oggi

Gioco finale

Oggi analizzeremo un gioco da cui potrete prendere esempio per i vostri progetti! Step:

- scaricate il codice del gioco *download*
- aprite il file con il codice su spider
- eseguite il gioco
- commentate il codice (#commento in python)
- riutilizzate il codice!

Dodger



Import dei moduli

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

Importiamo i moduli utili per creare un gioco (quelli che abbiamo già visto).

Variabili Costanti

```
WINDOWWIDTH = 600
WINDOWHEIGHT = 600
TEXTCOLOR = (0, 0, 0)
BACKGROUNDCOLOR = (255, 255, 255)
FPS = 60
```

- dimensioni finestra
- colore del testo e dello sfondo
- numero di *frame* per secondo. Più è alto questo numero più il gioco andrà veloce.

Variabili Costanti

```
BADDIEMINSIZE = 10
BADDIEMAXSIZE = 40
BADDIEMINSPEED = 1
BADDIEMAXSPEED = 8
ADDNEWBADDIERATE = 6
PLAYERMOVERATE = 5
```

- dimensioni e velocità dei cattivi
- velocità con cui vengono creati nuovi cattivi
- velocità di movimento del giocatore

Funzioni

```
def terminate():
    pygame.quit()
    sys.exit()
```

Per chiudere il gioco è necessario chiamare entrambe queste funzioni. Può essere utile raggrupparle in un'unica funzione *terminate()*.

Funzioni

```
def waitForPlayerToPressKey():
    while True:
        for event in pygame.event.get():
            if event.type == QUIT:
                terminate()
            if event.type == KEYDOWN:
                if event.key == K_ESCAPE:
                    terminate()
                return
```

Per mettere in pausa il gioco (inizio e Game Over) è possibile richiamare la funzione *waitForPlayerToPressKey()*. Questa funzione verifica se l'utente vuole chiudere la finestra (QUIT o K_ESCAPE) o se vuole continuare il gioco cliccando qualunque altro tasto.

Funzioni

```
def playerHasHitBaddie(playerRect, baddies):
    for b in baddies:
        if playerRect.colliderect(b['rect']):
            return True
    return False
```

Per sapere se il giocatore si è scontrato con un cattivo si può utilizzare la funzione *playerHasHitBaddie(playerRect, baddies)*. Se il "rettangolo" del giocatore collide (si "tocca") con il rettangolo anche di un solo nemico allora la funzione restituisce "True" altrimenti "False".
I nemici nel gioco sono tanti quindi baddies è una lista.

NB colliderect(rect) è una funzione di pygame che ci dice quando due rettangoli collidono.

Funzioni

```
def drawText(text, font, surface, x, y):
    textobj = font.render(text, 1, TEXTCOLOR)
    textrect = textobj.get_rect()
    textrect.topleft = (x, y)
    surface.blit(textobj, textrect)
```

Per mostrare un testo (i.e. per il Game Over o per il punteggio) si può utilizzare la funzione *drawText(text, font, surface, x ,y)*. Per prima cosa si crea un rettangolo su cui verrà disegnato il testo. Posizioniamo il testo nelle coordinate x,y. Poi nell'ultima linea disegnamo il rettangolo del testo sopra a surface (lo sfondo). Per fare questo abbiamo bisogno del rettangolo del testo (textobj) e delle sue dimensioni (textrect).

Impostazione di pygame e della finestra di gioco

```
pygame.init()
mainClock = pygame.time.Clock()
windowSurface =
    pygame.display.set_mode((WINDOWWIDTH,
    WINDOWHEIGHT))#, pygame.FULLSCREEN)
pygame.display.set_caption('Dodger')
pygame.mouse.set_visible(False)
```

Utilizzeremo la variabile mainClock per gestire il tempo del nostro gioco.
La variabile windowSurface rappresenta la finestra di gioco.
La funzione *pygame.display.set_caption()* permette di modificare il *titolo* della finestra di gioco.
L'ultima funzione *pygame.mouse.set_visible(False)* dice a pygame di ñascondereïl cursore.

Impostazione del Font

```
# Set up the fonts.
font = pygame.font.SysFont(None, 48)
```

Passando *None* alla funzione *pygame.font.SysFont()* si utilizza il font di default. Al suo posto si può mettere qualsiasi altro font (Arial, Times New Roman,...)

Impostazione della musica e dei suoni

```
# Set up sounds.
gameOverSound=pygame.mixer.Sound('gameover.wav')
pygame.mixer.music.load('background.mid')
```

La variabile *gameOverSound* rappresenta la musichetta che il gioco riproduce quando il giocatore perde. La musica di sottofondo viene caricata con la funzione *pygame.mixer.music.load()*.
Al posto di gameover.wav e background.mid si possono scaricare e utilizzare altre musichette.
Step

- scaricare un file .wav o .midi da google
- spostare il file scaricato nella directory del vostro gioco (dove c'è il file gioco.py)
- cambiare il nome del file nel codice sorgente

Impostazione delle immagini

```
# Set up images.
playerImage = pygame.image.load('player.png')
playerRect = playerImage.get_rect()
baddieImage = pygame.image.load('baddie.png')
```

Come per i suoni potete utilizzare le vostre immagini per i personaggi del gioco. Queste immagini devono essere salvate nella directory del gioco e devono avere l'estensione .png o .jpeg. Infine ricordatevi di cambiare il nome dell'immagine nel codice.

Schermata iniziale

```
windowSurface.fill(BACKGROUNDCOLOR)
drawText('Dodger', font, windowSurface,
    (WINDOWWIDTH / 3), (WINDOWHEIGHT / 3))
drawText('Press a key to start.', font,
    windowSurface, (WINDOWWIDTH / 3) - 30,
    (WINDOWHEIGHT / 3) + 50)
pygame.display.update()
waitForPlayerToPressKey()
```

All'inizio del gioco vogliamo mostrare una schermata con il nome del gioco e la scritta "Spingi un tasto per inziare". Dopodichè ci mettiamo in attesa dell'azione richiamando la funzione *waitForPlayerToPressKey()*.

Inizio del gioco

Ora che abbiamo definito tutte le funzioni utili e *l'ambiente* di partenza possiamo iniziare a scrivere il nostro gioco!

Inizio del gioco

```
topScore = 0
while True:
    # Set up the start of the game.
    baddies = []
    score = 0
    playerRect.topleft = (WINDOWWIDTH / 2,
        WINDOWHEIGHT - 50)
    moveLeft = moveRight = moveUp = moveDown =
        False
    reverseCheat = slowCheat = False
    baddieAddCounter = 0
    pygame.mixer.music.play(-1, 0.0)
```

Inizio del gioco

- creiamo una variabile per salvare il punteggio migliore
- iniziamo il *loop* del gioco
- i nemici sono 0
- il punteggio della partita è 0
- posiziono il personaggio nel centro della finestra
- il personaggio è fermo
- i cattivi sono fermi
- creiamo una variabile per contare quanti cattivi ci sono
- chiamando la funzione *pygame.mixer.music.play(-1, 0.0)* con -1 diciamo a python di riprodurre la musica continuamente. Il secondo paramentro dice a python quanto aspettare prima di riprodurre il brano.

Partita

```
while True: # The game loop runs while the
    game part is playing.
     score += 1 # Increase score.
```

Questo è il *loop* di una partita. in questa parte di codice dovremo inserire tutte le azioni da fare per far funzionare il gioco. Prima di tutto ad ogni *ciclo* aumentiamo il punteggio della partita.

Partita

```
for event in pygame.event.get():
            if event.type == QUIT:
                terminate()
```

Se premo...

La x della finestra il gioco si chiude

Partita

```
if event.type == KEYDOWN:
                    if event.key == K_z:
                        reverseCheat = True
                    if event.key == K_x:
                        slowCheat = True
```

Se premo...

Z modifico la variabile reverseCheat
X modifico la variabile slowCheat

Partita

```
if event.key == K_LEFT or event.key == K_a:
                    moveRight = False
                    moveLeft = True
                if event.key == K_RIGHT or
                   event.key == K_d:
                    moveLeft = False
                    moveRight = True
                if event.key == K_UP or
                   event.key == K_w:
                    moveDown = False
                    moveUp = True
                if event.key == K_DOWN or
                   event.key == K_s:
                    moveUp = False
                    moveDown = True
```

Partita

```
if event.type == KEYUP:
                    if event.key == K_z:
                        reverseCheat = False
                        score = 0
                    if event.key == K_x:
                        slowCheat = False
                        score = 0
                    if event.key == K_ESCAPE:
                            terminate()
```

Se premo...

KYEUP vs KEYDOWN ??

K_ESCAPE = **ESC**

Partita

```
if event.type == MOUSEMOTION:
                    playerRect.centerx = event.pos[0]
                    playerRect.centery = event.pos[1]
```

Se muovo il mouse...

MOUSEMOTION rileva quando il mouse si muove, in particolare riposiziona il personaggio in base al suo movimento.
Se volessi fare qualcosa cliccando il mouse??

Aggiungere cattivi

```
if not reverseCheat and not slowCheat:
                baddieAddCounter += 1
```

Se NON stiamo premenso Z o X allora incrementiamo la variabile *baddieAddCounter* ad ogni *ciclo*.

Aggiungere cattivi

```
if baddieAddCounter == ADDNEWBADDIERATE:
    baddieAddCounter = 0
    baddieSize = random.randint(BADDIEMINSIZE,
        BADDIEMAXSIZE)
    newBaddie = {'rect':
        pygame.Rect(random.randint(0, WINDOWWIDTH
        - baddieSize), 0 - baddieSize,
        baddieSize, baddieSize),
         'speed': random.randint(BADDIEMINSPEED,
            BADDIEMAXSPEED),
         'surface':pygame.transform.scale(baddieImag
            (baddieSize, baddieSize)),
                          }

    baddies.append(newBaddie)
```

Muovere i personaggi

```
if moveLeft and playerRect.left > 0:
    playerRect.move_ip(-1 * PLAYERMOVERATE, 0)
if moveRight and playerRect.right < WINDOWWIDTH:
    playerRect.move_ip(PLAYERMOVERATE, 0)
if moveUp and playerRect.top > 0:
    playerRect.move_ip(0, -1 * PLAYERMOVERATE)
if moveDown and playerRect.bottom < WINDOWHEIGHT:
    playerRect.move_ip(0, PLAYERMOVERATE)
```

Usiamo sempre i booleani per verificare in che direzione spostare il giocatore. Per muoverlo chiamiamo una funzione move_ip(x,y) in cui x è lo spostamento sull'asse orizzonatle e y su quello verticale.

Muovere i cattivi

```
for b in baddies:
    if not reverseCheat and not slowCheat:
        b['rect'].move_ip(0, b['speed'])
    elif reverseCheat:
        b['rect'].move_ip(0, -5)
    elif slowCheat:
        b['rect'].move_ip(0, 1)
```

I nemici si muovono in 3 modalità

- normale verso il basso
- piano verso il basso (X)
- verso l'alto (Z)

Cancellare i cattivi

```
for b in baddies[:]:
    if b['rect'].top > WINDOWHEIGHT:
        baddies.remove(b)
```

Quando un cattivo "esce" dalla finestra di gioco lo eliminiamo anche dalla lista dei cattivi.

Aggiornare la schermata

```
windowSurface.fill(BACKGROUNDCOLOR)
```

Come prima cosa coloriamo lo sfondo.

Aggiornare la schermata

```
drawText('Score: %s' % (score), font,
    windowSurface, 10, 0)
drawText('Top Score: %s' % (topScore), font,
    windowSurface, 10, 40)
```

Punteggio

Per mostrare il punteggio usiamo la funzione *drawText*. I parametri sono: il testo, il font, lo sfondo, la posizione (x,y))

Aggiornare la schermata

```
windowSurface.blit(playerImage, playerRect)

for b in baddies:
    windowSurface.blit(b['surface'], b['rect'])
```

Personaggio e cattivi

per ogni cattivo (b) presente nella lista di cattivi (baddies) chiamo la funzione *windowSurface.blit(b['surface'], b['rect'])*.

Aggiornare la schermata

```
pygame.display.update()
```

Ricordiamoci di aggiornare la schermata!

Collisioni

```
if playerHasHitBaddie(playerRect, baddies):
    if score > topScore:
        topScore = score # set new top score
    break

mainClock.tick(FPS)
```

Alla fine del ciclo verifico se il giocatore ha toccato un cattivo. In caso affermativo e se il punteggio corrente è maggiore del migliore, aggiorno quest'ultimo. Infine termino la partita.

Game Over

```
pygame.mixer.music.stop()
gameOverSound.play()

drawText('GAME OVER', font, windowSurface,
    (WINDOWWIDTH / 3), (WINDOWHEIGHT / 3))
drawText('Press a key to play again.', font,
    windowSurface, (WINDOWWIDTH / 3) - 80,
    (WINDOWHEIGHT / 3) + 50)
pygame.display.update()
waitForPlayerToPressKey()

gameOverSound.stop()
```

Una volta terminata la partita è necessario mostrare la scermata di game over.

- stoppo la musica

Consigli!

- riutilizzare le funzioni già viste
- leggere bene gli errori
- utilizzare la documentazione https://www.pygame.org/docs/
- chiedere