Asteroidespillet, Del 3

Erfaren Python

Introduksjon

I denne oppgaven skal vi jobbe videre med romskipsspillet vårt fra del 2. Denne gangen skal vi få inn noen farlige asteroider!

Vi starter med den samme koden som vi hadde sist, og med de samme bildene liggende sammen med koden.

1. Lag spillskjelettet

Koden fra del 2 skal se slik ut:

```
import pygame

# Variable for vindusstørrelsen
display_width = 600
display_height = 800

# Start opp pygame slik at det kan brukes. Viktig!
pygame.init()

class Spaceship(pygame.sprite.Sprite):
    def __init__(self):
        pygame.sprite.Sprite.__init__(self)

        self.image = pygame.image.load("spaceship.png")

        self.rect = self.image.get_rect()

        self.rect.centerx = 300
        self.rect.centery = 700

    def updatePosition(self, x_speed):
        self.rect.centerx += x_speed

        if self.rect.right > display_width:
            self.rect.right = display_width

        if self.rect.left < 0:
            self.rect.left = 0

# Start pygame-klokka. Denne holder rede på tiden i spillet vårt
clock = pygame.time.Clock()

# Sett opp spillvinduet
gameDisplay = pygame.display.set_mode((display_width,display_height))
pygame.display.set_caption('Asteroid Run')

# Les inn bakgrunnsbildet
bgImg = pygame.image.load("background.png")

# Tegn bakgrunnsbildet
def drawBackground():
    gameDisplay.blit(bgImg, (0, 0))

ship = Spaceship()

# Lag en liste av "sprites" og legg til romskipet vårt i listen.
sprites_list = pygame.sprite.Group()
sprites_list.add(ship)
```

```
x_change = 0

# Start "hovedløkka" i spillet
finished = False
while not finished:
    # Sørg for at denne løkken går 60 ganger i sekundet
    clock.tick(60)

    # Sjekk hendelser
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            finished = True
        if event.type == pygame.KEYDOWN:
            if event.key == pygame.K_LEFT:
                x_change = -5
            if event.key == pygame.K_RIGHT:
                x_change = 5
        if event.type == pygame.KEYUP:
            if event.key == pygame.K_LEFT or event.key == pygame.K_RIGHT:
                x_change = 0


    ship.updatePosition(x_change)

    drawBackground()
    sprites_list.draw(gameDisplay)

    # Oppdater vinduet med all grafikken som skal tegnes
    pygame.display.update()


# Når vi hopper ut av løkka, avslutter vi spillet.
pygame.quit()
```

I denne versjonen har vi lagt inn en test for å hindre at romskipet går utenfor skjermen. Se på Spaceship-klassens updatePosition-funksjon:

```
        if self.rect.right > display_width:
            self.rect.right = display_width

        if self.rect.left < 0:
            self.rect.left = 0
```

2. Lag en asteroideklasse

I del 2 lagde vi en klasse for romskipet. Denne gangen skal vi lage en klasse for å tegne asteroider, og den blir ganske lik.

Først av alt trenger vi et nytt bilde som du kan laste ned her:

https://raw.githubusercontent.com/kkringerike/asteroidrun/master/asteroid.png

(Høyreklikk på bildet, velg "Lagre som..." og legg det i samme mappe som koden din.)

Skriv så inn følgende kode etter Spaceship-klassen din

```
class Asteroid(pygame.sprite.Sprite):
    def __init__(self):
        pygame.sprite.Sprite.__init__(self)

        self.image = pygame.image.load("asteroid.png")

        self.rect = self.image.get_rect()

        self.rect.centerx = random.randrange(0, display_width)
        self.rect.centery = -200

    def updatePosition(self, y_speed):
        self.rect.centery += y_speed
```

Denne klassen arver også fra "Sprite"-klassen og vi setter den opp med asteroidebildet vårt i `__init__`-metoden. Her setter vi også en startposisjon for asteroiden. Vi vil at asteroiden skal starte på et tilfeldig sted slik at det ikke blir likt hele tiden, derfor har vi skrevet:

```
        self.rect.centerx = random.randrange(0, display_width)
```

På denne linjen sier vi at x-posisjonen skal være et tilfeldig tall mellom 0 og `display_width` (bredden på vinduet).

For at dette skal virke trenger vi `random`-biblioteket som lager tilfeldige tall, så vi må legge inn dette helt øverst i programmet:

```
import random
```

Vi vil også at asteroiden skal starte utenfor skjermen og komme inn ovenfra. Derfor har vi satt y-posisjonen til -200:

```
        self.rect.centery = -200
```

3. Lag et asteroide-objekt

Vi skal nå lage oss en asteroide med den nye klassen. Skriv inn følgende rett etter at du lager romskip-obkektet (like før hovedløkka):

```
asteroid = Asteroid()
sprites_list.add(asteroid)
```

Hvis du prøver å kjøre programmet ditt nå, vil du ikke se noen forskjell. Du har laget en asteroide, men den tegnes utenfor skjermen (vi satte jo y til å være -200).

For at vi skal se asteroiden, må vi flytte den nedover skjermen. Vi lager oss en variabel for farten til asteroiden. Legg den f.eks sammen med `x_change`-variablen som styrer romskipet:

```
y_speed = 5
```

Og så må vi oppdatere posisjonen til romskipet hver gang vi skal tegne det. Altså, legger vi inn denne linjen i hovedløkka, rett før vi tegner bakgrunnen og spritene:

```
    asteroid.updatePosition(y_speed)
```

- [ ] Kjør programmet og se hva som skjer. En asteroide skal fly sakte nedover skjermen.

4. Flere asteroider!

Det dumme med denne asteroiden er jo at når den har fløyet forbi, så ser vi den aldri igjen. Vi trenger derfor å lage flere asteroider. Vi lager en *if*-sjekk for å se om asteroiden har fløyet forbi nederste kant av skjermen, og i så fall lager vi oss en ny asteroide. SKriv inn denne koden i slutten av hovedløkka

```
    if asteroid.rect.top > display_height:
        sprites_list.remove(asteroid)

        asteroid = Asteroid()
        sprites_list.add(asteroid)
```

Det vi gjør her er altså å ta den gamle asteroiden ut av sprite-listen vår, så lager vi en ny asteroide som vi putter inn i lista.

- [ ] Kjør programmet og se hva som skjer. Du skal nå få nye asteroider hele tiden, på forskjellige steder

Utfordringer

- [ ] Kan du få asteroidene til å fly stadig fortere? Den neste asteroiden skal være litt raskere enn den forrige.

  Hint