Inhaltsverzeichnis

1 Was ist PyGame ?

PyGame ist eine Sammlung von vielen Modulen und dient zur Spielprogrammierung. Sie enthält Module zum Abspielen und Steuern von Grafiken, Sound sowie auch Abfragen von Eigabegeräten (Tastatur, Maus).

2 Grundgerüst

Im folgenden Code Block siehst du das Grundgerüst eines Spieles. Hier wird ein schwarzes Fenster erstellt, wenn man rechts oben auf den roten **x** Button klickt, wird das Fenster geschlossen.

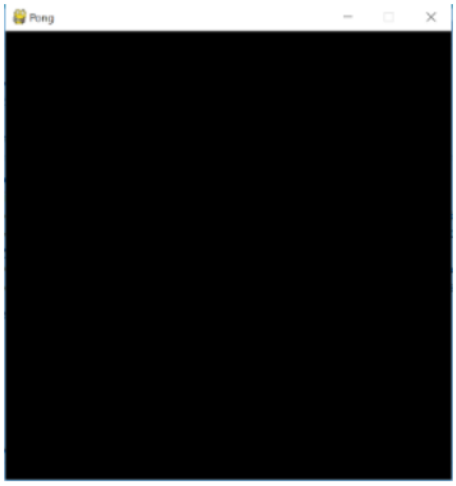
```
import pygame

x = 500
y = 500

title = "Pong"

run = True

pygame.init()
screen = pygame.display.set_mode((x, y))
pygame.display.set_caption(title)

while run:
        for event in pygame.event.get():
                if event.type == pygame.QUIT:
                run = False
        pygame.display.flip()
```

Als erstes initialisierst du drei Variablen x, y und title. Diese bestimmen die Breite und die Höhe des Fensters sowie den Titel des Spieles, wie man aus dem Variable-Namen herauslesen kann.

Danach lädst du das Framework, bzw. initialisierst alle Module. Mit dem nächsten Befehl erstellst du ein Fenster mit der vorgegebenen Höhe und Breite, also mit x und y und setzt den Titel des Spieles.

Die While-Schleife überprüfst du, solange das Spiel läuft, ob der x Button geklickt wurde und beendest dann das Spiel.

Das erzeugte Fenster sieht so aus:



3 Pong

Jetzt weisst du wie man ein Fenster erstellt. Nun können wir mit dem programmieren des Spieles beginnen.

Als erstes solltest du einen neuen Ordner in deinem Workspace hinzufügen. In diesem Ordner erstellst du ein neues File, vergesse das **.py** am Ende des Namens nicht. Erstelle das Fenster, wie oben beschrieben

Der Schläger sollte sich am unteren Ende des Spielfeldes bewegen. Er sollte nur nach links und rechts bewegt werden können und darf nicht über den Spielrand hinweg.

```
racketWidth = 100
racketHeight = 15

racketX = 200
racketY = 450

...

pygame.draw.rect(screen, (255, 40, 0), (racketX, racketY, racketWidth,
racketHeight), 0)
```

Mit diesem Code wird ein roter Schläger am unteren Ende gezeichnet. Die Zahlen (255, 40, 0) stehen für die Farbe. Du kannst auch mit RGB (Red-Green-Blue) eine andere Farbe aussuchen. Mithilfe von draw können wir ein Rechteck sehr einfach zeichnen, du kannst auch andere geometrische Figuren mithilfe von draw zeichnen.

Die Initialisierungen der Variablen des Schlägers kommen unter den vorherigen Initialisierungen für das Fenster und die draw Methode kommt in die While-Schleife vor der Methode pygame.display.flip().

Jetzt muss sich der Schläger auch bewegen lassen.

```
speed = 0

...

def moveRacket():
    global racketX
    racketX += speed


def racketBlock():
    global speed
    if racketX <= 0 or racketX >= x - racketWidth:
        speed = 0

...

               if event.type == pygame.KEYDOWN:
                       if event.key == pygame.K_LEFT:
                       speed = -2
                       if event.key == pygame.K_RIGHT:
                       speed = 2
    screen.fill((0, 0, 0))
    moveRacket()
    racketBlock()

...
```

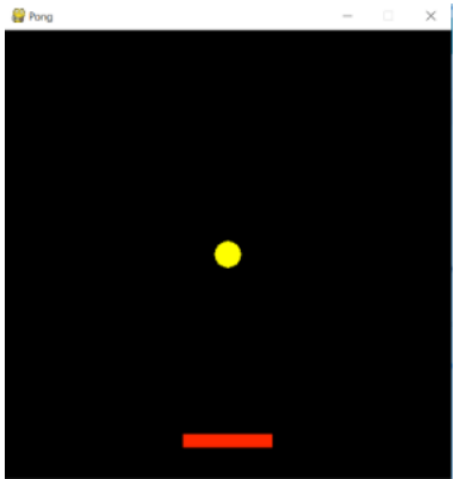
Als erstes musst du noch eine Varible namens speed initialisieren. Sie bestimmt die Geschwindigkeit des Schlägers. Danach erstellst du zwei neue Methoden. MoveRacket() um den Schläger zu bewegen und RacketBlock() um den Rand des Spielfeldes zu setzten. Beide Methoden kommen oberhalb der while-Schleife. In der While-Schleife überprüfst du, ob die Taste mit dem Pfeil nach links oder rechts geklickt wurde und passt die Geschwindigkeit je nach dem an.

Nun kannst du den Schläger bewegen. Jetzt brauchst du nur noch einen Ball.

```
ballX = int(x/2)
ballY = int(y/2)

ballRadius = 15

...

pygame.draw.circle(screen, (255, 255, 0), (ballX, ballY), ballRadius, 0)

...
```

Die Initialisierungen der Position und des Radiusses kommen unter den Anderen hin und die draw Methode für den Ball kommt unter die für den Schläger. Wie du siehst, soll der Ball in der Mitte des Fensters gezeichnet werden. Die Koordinaten bestimmen immer den Mittelpunkt des Kreises.

Nun sollte dein Fenster wie folgt aussehen:



Jetzt sollte sich aber der Ball zufällig in irgendeine Richtung bewegen und jeweils an der Rändern (rechts, links, oben) und dem Schläger abprallen.

Deshalb bestimmst du den Rand für den Ball als erstes. Dies machst du in folgender Methode.

```
def ballBlock():
    global ballYSpeed, ballXSpeed
    if ballY - ballRadius <= 0:
        ballYSpeed *= -1
    if ballX - ballRadius <= 0:
        ballXSpeed *= -1
    if ballX + ballRadius >= x:
        ballXSpeed *= -1
    if ballY >= 435 and ballY <= 440:
        if ballX >= racketX -15 and ballX <= racketX + racketWidth + 15:
            ballYSpeed *= -1
```

Hier überprüfst du jeweils dieselbe Seite des Balles mit dem Rand, falls der Ball am Rand ist, sollte er in die Gegenrichtung abprallen, dies machst du indem du die Geschwindigkeit *(-1) rechnest.

Du brauchst noch zusätzliche Variablen für die neue Position sowie eine Methode moveBall().

```
ballXSpeed = 1
ballYSpeed = -2

...

def moveBall():
    global ballX, ballY
    ballX += ballXSpeed
    ballY += ballYSpeed

...

moveBall()
ballBlock()

...
```

Die Methoden rufst du am Ende in der While-Schleife auf, doch vo dem Zeichnen des Balles.

Damit das Spiel nicht beendeet wird, wenn man den Ball nicht "fängt" brauchst du noch eine reset Methode.

```
import random

...

def reset():
    global ballYSpeed, ballXSpeed, life, ballX, ballY, racketX, racketY,
speed
    racketX = 200
    racketY = 450


    ballX = int(x/2)
    ballY = int(y/2)

    speed = 0
    ballXSpeed = random.randint(-2, 2)
    if ballXSpeed == 0:
        ballXSpeed = 1
    ballYSpeed = -2
    screen.fill((0, 0, 0))
    pygame.draw.circle(screen, (255, 255, 0), (ballX, ballY), ballRadius, 0)
    pygame.draw.rect(screen, (255, 40, 0), (racketX, racketY, racketWidth,
racketHeight), 0)
    pygame.display.flip()
    pygame.time.wait(1000)
```

Am Anfang musst du noch import random hinzufügen, damit du eine zufällige Zahl generieren kannst.

Diese rufst du in der Methode ballBlock() bei der letzten Überprüfung in else auf. Nun musst du diese Methoden auch in der While-Schleife aufrufen.

Am Ende sollte dein Code so aussehen:

```
import pygame
import random


run = True

x = 500
y = 500

racketWidth = 100
racketHeight = 15

racketX = 200
racketY = 450

ballX = int(x/2)
ballY = int(y/2)

ballRadius = 15


speed = 0

ballXSpeed = 1
ballYSpeed = -2

pygame.init()
screen = pygame.display.set_mode([x,y])


def reset():
    global ballYSpeed, ballXSpeed, life, ballX, ballY, racketX, racketY,
speed
    racketX = 200
    racketY = 450

    ballX = int(x/2)
    ballY = int(y/2)

    speed = 0
    ballXSpeed = random.randint(-2, 2)
    if ballXSpeed == 0:
        ballXSpeed = 1
       ballYSpeed = -2
    screen.fill((0, 0, 0))
    pygame.draw.circle(screen, (255, 255, 0), (ballX, ballY), ballRadius, 0)
    pygame.draw.rect(screen, (255, 40, 0), (racketX, racketY, racketWidth,
racketHeight), 0)
    pygame.display.flip()
    pygame.time.wait(1000)


def racketBlock():
    global speed
    if racketX <= 0 or racketX >= x - racketWidth:
        speed = 0

def moveBall():
```

```
    global ballX, ballY
    ballX += ballXSpeed
    ballY += ballYSpeed


def ballBlock():
    global ballYSpeed, ballXSpeed
    if ballY - ballRadius <= 0:
        ballYSpeed *= -1
    if ballX - ballRadius <= 0:
        ballXSpeed *= -1
    if ballX + ballRadius >= x:
        ballXSpeed *= -1
    if ballY >= 435 and ballY <= 440:
        if ballX >= racketX -15 and ballX <= racketX + racketWidth + 15:
            ballYSpeed *= -1
        else:
            reset()

def moveRacket():
    global racketX
    racketX += speed

while run:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            run = False
        if event.type == pygame.KEYDOWN:
            if event.key == pygame.K_LEFT:
                speed = -2
            if event.key == pygame.K_RIGHT:
                speed = 2
    screen.fill((0, 0, 0))
    moveRacket()
    racketBlock()
    pygame.draw.rect(screen, (255, 40, 0), (racketX, racketY, racketWidth,
racketHeight), 0)
    moveBall()
    ballBlock()
    pygame.draw.circle(screen, (255, 255, 0), (ballX, ballY), ballRadius, 0)
    pygame.display.flip()
    pygame.time.wait(5)
pygame.quit()
```

Funktioniert alles?

Mache eine Anzahl leben und schaue, dass das Spiel abbricht, sobald man alle Leben verloren hat.

Zwei kleine Tipp

- Prüfe ob du noch ein Leben hast mit der While-Schlaufe
- Schaue dort, wo sich das Spiel erneuert, ein Leben zu entziehen