

Arbeitsblatt „Pong“

Wir werden mit Hilfe des Pakets `pygame` ein einfaches Spiel in Python programmieren, das sich so verhält, wie das Beispiel auf der zweiten Folie des Lehrvideos: Ein "Ball" in Form des Ohm-Logos wandert über das Spielfeld und muss am unteren Rand durch einen vom Spieler mit den Pfeiltasten bewegten Schläger zurückgeschlagen werden. Gelingt dies nicht, endet das Spiel.

Ein Einführungsvideo, Installationshinweise und ein Beispielprogramm zu `pygame` finden Sie im E-Learning Portal unter 2D-Spiele.

Leeres Fenster

Erstellen Sie ein erstes „Spiel“, das nach dem Start ein leeres Fenster der Größe 640x400 Pixel anzeigt. Der Fenstertitel soll „PONG!“ lauten. Sobald die Taste ESC gedrückt wird, soll sich das Fenster schließen.

Hinweise:

Der Fenstertitel wird mithilfe der Funktion `pygame.display.set_caption` gesetzt.

Das Ereignis "ESC gedrückt" kann mit

```
if event.type == pygame.KEYDOWN:
    if event.key == pygame.K_ESCAPE:
        pass
```

im Rahmen der Input-Verarbeitung überprüft werden.

Statische OHM-Grafik

Im nächsten Schritt soll die Grafik `ohm.png` in der Mitte des Bildschirms angezeigt werden. Verwenden Sie zur Ermittlung der richtigen Koordinaten die Eigenschaften `width` und `height` des Fensters.

Bewegte OHM-Grafik

Bei jedem Durchlauf der Game Loop soll sich die OHM-Grafik nun bewegen. Die Anzahl der Pixel, um die die Grafik verschoben werden soll, ist beim Programmstart in der Funktion `init` zufällig festzulegen.

Sobald die Grafik den Fensterrand berührt, soll sie von diesem abprallen, d.h. beim linken und beim rechten Rand invertiert sich die Verschiebung der x-Achse, beim oberen und beim unteren Rand invertiert sich die Verschiebung der y-Achse.

Führen Sie die notwendigen Aktualisierungen in der Funktion `update` durch. In der Funktion `draw` soll nur die Grafik an der berechneten Stelle ausgegeben werden.

Hinweise:

Für die Ermittlung der zufälligen Startgeschwindigkeit kann der Zufallsgenerator aus der Python-Standardbibliothek verwendet werden:

```
from random import randint  
wuerfel = randint(1, 7) # Ganze Zufallszahl aus [1, ..., 7[
```

Statischer Schläger

Neben der bewegten OHM-Grafik soll auch ein Schläger im Fenster dargestellt werden. Dazu soll ein rotes Rechteck am unteren Bildschirmrand eingeblendet werden. Bitte experimentieren Sie, mit welchen Koordinaten sich ein brauchbares Layout ergibt.

Bewegter Schläger

Ist die Taste „Pfeil nach links“ gedrückt (`pygame.K_LEFT`), soll sich der Schläger nach links bewegen; ist die Taste „Pfeil nach rechts“ gedrückt (`pygame.K_RIGHT`), bewegt sich der Schläger nach rechts.

Registrieren Sie die Tastendrücke in der Funktion `user_input`, berechnen Sie die neue Position in der Funktion `update` und zeichnen Sie den Schläger in der Funktion `draw`.

Kollision

Trifft die OHM-Grafik auf den Schläger und ist sie gerade auf dem Weg nach unten, so soll sie vom Schläger abprallen, d.h. die y-Verschiebung invertiert werden.

Hinweis: `pygame.Rect` besitzt die Methoden `collidect`, mit der eine Kollision mit einem anderen `Rect` ermittelt werden kann. Diese Überprüfung sollte ebenfalls in der `update`-Funktion stattfinden.



Game Over

Erreicht die OHM-Grafik den unteren Bildschirmrand, soll eine Grafik mit dem Text „Game Over“ eingeblendet werden. Einen Hinweis, wie eine solche Grafik erstellt werden kann, finden Sie im Lehrvideo.