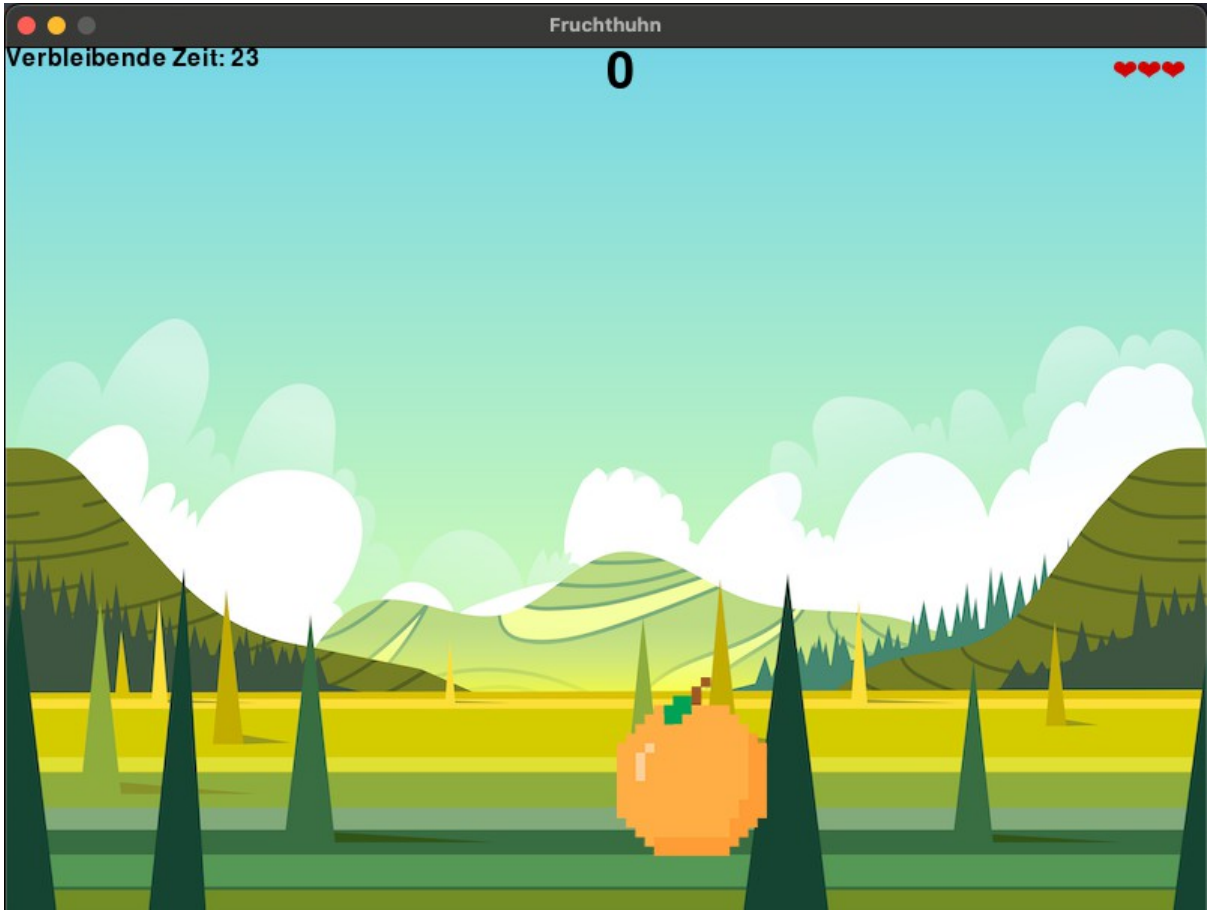
Python - Fruchthuhn Workshop

Python

Python

- Programmiersprache

Python - Fruchthuhn



Python - Fruchthuhn

- Spielscreen
- Früchte darstellen
- Maus clicks
- Score
- Zeit
- Spielerleben

Python - Fruchthuhn

- Für ein Spiel verwenden wir in Python:
  - pgzero

Falls es nicht da ist, muss es erst installiert werden.

Python - Fruchthuhn

- Installation pygame (https://www.pygame.org/wiki/GettingStarted):
  - Windows:
    - Commando Zeile:
      - py -m pip install -U pygame -- user
  - Mac:
    - Commando Zeile:
      - python3 -m pip install -U pygame -- user
  - Linux (Debian/Ubuntu/Mint):
    - Commando Zeile:
      - sudo apt-get install python3-pygame
- Einbinden:
  - import pygame

Python & Pgzero

Funktion	Beschreibung
init()	Initialisiert alle importierten Pygame-Module (gibt ein Tupel zurück, das den Erfolg und Misserfolg von Initialisierungen anzeigt).
draw()	Zeichen Funktion → um Sachen auf den Bildschirm zu zeichnen Wird immer wieder automatisch während das spiel läuft aufgerufen.
update()	Wird 60 mal in der Sekunde aufgerufen.
on_mouse_down(pos)	Wird aktiviert sobald mit der Maus auf dem monitor geklickt wird. Pos entspricht der Position der Maus.

Python – Hintergrund

Code	Beschreibung
import pgzrun	
WIDTH = 300	Höhe des Bildschirms
HEIGHT = 300	Breite des Bildschirms
def draw(): screen.fill((128,0,0))	Füllt den Hintergrund mit einer Farbe die aus den drei (rot, grün, blau) gemischt werden.

Welche Farbe hat der Hintergrund?

Versuch die Farbe zu wechseln, indem du für die drei Farben Werte von 0 bis 255 angibst.

Python – Sprite (Figuren)

Wir wollen Figuren zeichnen

Benötigt wird dafür:

- Bild welches gezeichnet werden soll
- die Actor('bild_name') Klasse von pgzero

Python – Sprite (Figuren)

Code	Beschreibung
frucht = Actor('apple')	Actor ist unsere Spielfigur – in der klammer wird ein Bild eingebunden, was „apple“ heißt.
def draw(): screen.clear() fruit.draw()	 Bildschirm leeren. Figur auf dem Bildschirm zeichnen

Python – Maus clicks

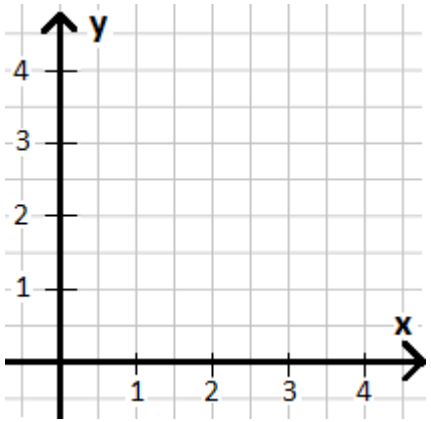
Lass die Frucht durch einen Maus click verschwinden!

- nutze die on_mouse_down(pos) Funktion

- frucht.collidepoint(pos) – prüft ob die Frucht (Sprite) an der selben stelle is wie pos

Tipp: Frucht verschwinden lassen, indem die Position der Frucht geändert wird

Python – Maus clicks



frucht.pos(x,y)

Oder

frucht.x = 160
frucht.y = 300

Python – Zufällige Position

from random import randint → sollte ganz oben verwendet werden, wird verwendet um randint benutzen zu können.

randint(minWert, maxWert) → gibt einen zufallswert zwischen minWert und maxWert

Python – Maus clicks

```
def on_mouse_down(pos):
    if frucht.collidepoint(pos):
        print("Treffer!")
        frucht.x = randint(10, 800)
        frucht.y = randint(10, 600)
    else:
        print("Daneben!")
```

Python – Score

Es soll angezeigt werden, wie viele Früchte der Spieler getroffen hat.

Sobald der Spieler eine Frucht angeklickt hat soll der Score um 1 erhöht werden.

Tipp: zum anzeigen benötigt ihr

```
screen.draw.text('text zum anzeigen', (x, y), fontsize=50, color="black")
```

Python – Score

Was passier wenn ihr die `fontsize` kleiner oder größer als 50 wählt?

Was passiert wenn ihr für `color` eine andere Farbe eingebt?

Python – Score

```
frucht = Actor("apple")
# add score variable
score = 0
WIDTH = 800
HEIGHT = 600

def draw():
    screen.clear()
    screen.blit("background", (0, 0))
    # zeige den Score oben links an
    screen.draw.text(str(score), (400, 0), fontsize=50, color="black")
    frucht.draw()

def place_fruit():
    frucht.x = randint(10, WIDTH)
    frucht.y = randint(10, HEIGHT)

def on_mouse_down(pos):
    if frucht.collidepoint(pos):
        print("Treffer!")
        place_fruit()
        # update score
        score += 1
    else:
        print("Daneben!")

# setze alle start werte auf den entsprechenden Anfangswert
def init():
    global score
    score = 0
    place_fruit()
```

Python – Verschiedene Früchte

Es sollen verschiedene Früchte angezeigt werden, nicht nur der Apfel.

Dafür benötigen wir verschiedene Frucht Bilder, die im selben Ordner wie das Apfelbild Gespeichert sind.

Tipp: Ein Array mit Fruchtbildnamen und dann dem Actor zufällig eins der Bilderübergeben
Die Frucht muss auch zufällig neu gesetzt werden, wenn die Frucht angeklickt wurde

Python – Verschiedene Früchte

```
# array mit den frucht bildern
fruechte = ['apple', 'orange', 'pineapple']
# waehle ein bild zufaellig aus 'fruechte' aus
frucht = Actor(fruechte[randint(0, len(fruechte) – 1)])



def place_fruit():
    frucht.x = randint(10, WIDTH)
    frucht.y = randint(10, HEIGHT)

    # aendere das bild der aktuellen frucht durch ein neues zufaelliges bild aus
'fruechte'
    frucht.image = fruechte[randint(0, len(fruechte) - 1)]
```

Python – Timer

Es soll ein Timer mit der verbleibenden Zeit angezeigt werden.

Der Timer soll bei 30 Sekunden starten und langsam runterzählen.

Tipp: Verwendet hierfür

```
def update(dt):
```

Python – Timer

```
# neue variable timer mit 30
timer = 30

# standard methode updated den screen etc.
def update(dt):
    global timer
    if timer >= 0:
        timer -= dt

def draw():
    screen.clear()
    screen.blit("background", (0, 0))
    screen.draw.text(str(score), (400, 0), fontsize=50, color="black")
    frucht.draw()

    # zeigt die verbleibende Zeit in der mitte des bildschirms an
    screen.draw.text("Verbleibende Zeit: %s" % round(timer), (0, 0), color="black")

    # zeigt ein Game Over wenn der Timer abgelaufen ist
    if timer < 0:
        screen.draw.textbox("Game Over!\n %s Punkte erreicht" % score, Rect(100, 100, 600, 400),
background="black")

def on_mouse_down(pos):
    global score, timer

    # reagiere nur auf maus klicks wenn der timer noch nicht abgelaufen ist
    if timer > 0:
        if frucht.collidepoint(pos):
            print("Treffer!")
            place_fruit()
            score += 1
        else:
            print("Daneben!")
```

Python – Restart

Nachdem der Timer abgelaufen ist und der Spieler ein Game over angezeigt bekommt.

Soll es nun auch möglich sein über einen Tasten druck das Spiel neu zu starten.
Dafür soll nicht immer das ganze Programm neugestartet werden.

Tipp: Verwendet hierfür

```
def on_key_down(key):
```

Python – Restart

```
# standard methode wird aufgerufen bei jedem Tasten druck
def on_key_down(key):

    # pruefe ob R gedrückt wurde um jederzeit das spiel neu zu starten
    if key == keys.R:
        init()]
```

Python – Spielerleben

Der Spieler soll 3 Leben bekommen.

Wenn der Spieler die Frucht nicht trifft soll ihm ein Leben abgezogen werden.

Tipp: Es wird eine Variable benötigt, in der die Anzahl der Spieler Leben gespeichert werden.

Python – Spielerleben

```
# neue variable für die Anzahl der Leben
player_lives = 3

WIDTH = 800
HEIGHT = 600

def update(dt):
    global timer, player_lives

    # update den timer nur wenn Leben da sind und timer nicht 0
    if timer >= 0 and player_lives > 0:
        timer -= dt


def draw():
    screen.clear()
    screen.blit("background", (0, 0))
    screen.draw.text(str(score), (400, 0), fontsize=50, color="black")
    frucht.draw()
    screen.draw.text("Verbleibende Zeit: %s" % round(timer), (0, 0), color="black")

    draw_lives()
    if player_lives == 0 or timer < 0:
        draw_game_over()


# funktion welche die Leben anzeigt auf der rechten seite
def draw_lives():
    for life in range(player_lives):
        screen.blit('heart', (WIDTH-30-(life*16), 10))
```

Python – Spielerleben

```
# Funktion, die prüft was für ein Game over angezeigt werden soll
def draw_game_over():
    global timer, score
    if timer > 0:
        screen.draw.textbox("Game Over!\n %s Punkte erreicht\n mit %s Sekunden Restzeit!" % (score, round(timer)), Rect(100, 100, 600,
400), background="black")
    else:
        screen.draw.textbox("Game Over!\n %s Punkte erreicht" % score, Rect(100, 100, 600, 400), background="black")

def place_fruit():
    frucht.x = randint(10, WIDTH)
    frucht.y = randint(10, HEIGHT)
    frucht.image = fruechte[randint(0, len(fruechte) - 1)]

def on_mouse_down(pos):
    global score, timer, player_lives
    if timer > 0:
        if frucht.collidepoint(pos):
            print("Treffer!")
            place_fruit()
            score += 1
        else:
            # ziehe ein Leben ab.
            player_lives -= 1
            print("Daneben!")

def on_key_down(key):
    if key == keys.R:
        init()

def init():
    global score, timer, player_lives
    score = 0
    place_fruit()
    timer = 30
    # setze das leben wieder auf 3
    player_lives = 3
```

Python – Früchte bewegen sich

Damit der Spieler es nicht so einfach hat, die Früchte zu erwischen,
Sollen die Früchte sich bewegen.

- Lass die Früchte sich nach rechts oder links bewegen

- Lass die Früchte sich nach oben oder unten bewegen

Tipp: Ändere hierfür in der draw() Funktion die x und y werte der Frucht

Python – Früchte bewegen sich

Was passiert mit der Frucht wenn ihr x und y gleichzeitig ändert?

Python – Früchte bewegen sich

```
def draw():
    screen.clear()
    screen.blit("background", (0, 0))
    screen.draw.text(str(score), (400, 0), fontsize=50, color="black")
    frucht.draw()
    screen.draw.text("Verbleibende Zeit: %s" % round(timer), (0, 0), color="black")
    move_frucht()
    draw_lives()
    if player_lives == 0 or timer < 0:
        draw_game_over()


def move_frucht():
    # addiere 1 zur Position der Frucht hinzu
    frucht.x += 1
    frucht.y += 1
```

Python – Früchte bewegen sich

Achte darauf, dass die Früchte sich innerhalb des vorgegebenen Spielebildschirms Bewegen.

- Prüfe dass die Frucht nichts rechts/links außerhalb des Bildschirm angezeigt wird

- Prüfe dass die Frucht nichts oben/unten außerhalb des Bildschirm angezeigt wird

Python – Früchte bewegen sich

```
def move_frucht():
    # addiere zufälligen wert zwischen -20 und 20 zur Position
der Frucht
    frucht.x += randint(-20, 20)
    frucht.y += randint(-20, 20)

    if frucht.x <= 10:
        frucht.x = 30

    if WIDTH - 20 <= frucht.x:
        frucht.x = WIDTH - 30

    if frucht.x <= 10:
        frucht.y = 30

    if HEIGHT - 20 <= frucht.y:
        frucht.y = HEIGHT - 20
```

Python – tanzende Früchte

Um den Schwierigkeitsgrad für den Spieler zu erhöhen, lassen wir die Früchte tanzen.

- setze den x Wert der Frucht auf einen zufälligen wert zwischen -10 und 10
  → benutze hierfür die random() Funktion

→ was kannst du beobachten?

Python – tanzende Früchte

- setze den y Wert der Frucht auf einen zufälligen wert zwischen -10 und 10
  → benutze hierfür die random() Funktion

→ was kannst du beobachten?

Python – tanzende Früchte

- lass x und y zufällig einen Wert zwischen -20 und 20 aus wählen

→ was passiert mit der Frucht?

Python – Früchte bewegen sich

```
def move_frucht():
    # addiere zufälligen wert zwischen -20 und 20 zur Position der Frucht
    frucht.x += randint(-20, 20)
    frucht.y += randint(-20, 20)

    if WIDTH - 20 <= frucht.x <= 0:
        frucht.x = 10
    if HEIGHT - 20 <= frucht.y <= 0:
        frucht.y = 10
```