Einstieg Spiele-Programmierung LÖVE Einführung

1 Vorbereitung

1. Extrahiere **StartGamedev** und öffne den Texteditor mithilfe der `open-editor` Datei.
2. Lese Aufgaben aufmerksam, tippe Code (*Quelltext*) ab und teste Ergebnisse.
3. Innerhalb einer Aufgabe (z.B. 1.1) wird der Code kontinuierlich erweitert. Wenn Du eine neue Aufgabe (z.B. 1.2) anfängst, solltest Du deinen Texteditor leeren.
4. Funktionen (z.B. `function love.draw() ... end`) dürfen nur ein mal vorkommen.
5. Benutze die Tabulatortaste um einzurücken (links neben Q), halte Deinen Code lesbar.
6. Funktionen, Schleifen und Konditionen enden mit `end`. Die Zeilen davor sind der Körper.
7. Deine Zeilen können anders durchnummeriert sein, als auf diesem Übungsblatt.

2 Malen, wie auf einer Leindwand

2.1 Dein Lieblingsrechteck

Ein Rechteck an der Position `x=100`, `y=200`. 300 Pixel breit und 150 hoch.

```
1 function love.draw()
2   love.graphics.rectangle("fill",100,200,300,150)
3 end
```

1. Zeichne das Rechteck an einer anderen Stelle.
2. Male das Rechteck „genau" in der rechten oberen Ecke. Das Fenster hat die Größe 800x600.
3. Ersetze `"fill"` mit `"line"`. Was passiert nun?
4. Male ein zweites Rechteck woanders. Kopiere nur Zeile 2.
5. Mache das gesamte Fenster Weiß.

2.2 Zwei Rechtecke

```
1 function love.draw()
2   love.graphics.setColor(0,255,0)
3   love.graphics.rectangle("fill",100,200,300,150)
4   love.graphics.setColor(255,255,255)
5   love.graphics.rectangle("fill",300,400,100,50)
6 end
```

1. Ändere die Zahlen in Zeile 2. Was passiert?
2. Diese Farben-Repräsentation mit drei Zahlen (0-255) wird RGB (Rot-Grün-Blau) genannt. Färbe das kleinere Rechteck blau.
3. Bewege die Rechtecke, sodass sie überlappen. Welches ist oben?
4. Vertausche die Zeilen 3 und 5. Was ändert sich?

2.3 Ein Paar Linien

```
function love.draw()
  love.graphics.line(100,0,100,200)
  love.graphics.line(0,200,100,200)
  love.graphics.rectangle("fill",100,200,300,150)
end
```

1. Bewege das Rechteck. Passe die Linien entsprechend an.
2. Das war anstrengend. Variablen können das automatisch machen! Fahre fort.

2.4 Variablen

```
x = 100
y = 200

function love.draw()
  love.graphics.line(100,0,x,y)
  love.graphics.line(0,200,x,y)
  love.graphics.rectangle("fill",x,y,300,150)
end
```

1. Was passiert, wenn man die Zahlen für `x` und `y` vertauscht?
2. Ändere Zeile 2 zu `y = x`. Was bedeutet dies?
3. Ändere Zeile 2 zurück zu `y = 200`. Ändere Zeile 1 zu: `x = y`. Es wird eine Fehlermeldung erscheinen. Kannst Du den Code korrigieren?
4. Führe eine Variable für die Breite des Rechtecks ein.

3 Interaktion

3.1 Ein bewegtes Bild

```
x = 100
y = 200

function love.draw()
  love.graphics.line(100,0,x,y)
  love.graphics.line(0,200,x,y)
  love.graphics.rectangle("fill",x,y,300,150)
end

function love.mousepressed()
  x = x + 10
end
```

1. Versuche, im Spiel den Mausknopf zu drücken. Etwas sollte geschehen.
2. Lasse das Rechteck sich rückwärts bewegen.
3. Lasse das Rechteck sich nach oben bewegen.
4. Vergrößere das Reckteck mit jedem Mausknopfdruck.

3.2 Wird richtig/rechts geklickt?

```
1  a = 100
2  b = 200
3
4  function love.draw()
5    love.graphics.rectangle("fill",a,b,300,150)
6  end
7
8  function love.mousepressed(mx, my)
9    local dir = "right"
10   if mx < 400 then dir = "left" end
11   if dir == "right" then a = a + 10 end
12 end
```

1. Wo muss man klicken, um das Rechteck zu bewegen?
2. Kopiere Zeile 11 und ändere sie so, dass `a` verringert wird wenn `dir == left"`.
3. Lasse das Rechteck den Rand berühren, aber nich darüber hinaus gehen (z.B. ergänze `and a < 500` vor `then`).

3.3 Es soll von selbst etwas tun

```
1  x = 100
2  y = 200
3
4  function love.draw()
5    love.graphics.rectangle("fill",x,y,300,150)
6  end
7
8  function love.update()
9    y = y - 1
10 end
```

Alles innerhalb vom `love.update`-Block wird 60 mal je Sekunde ausgeführt.

1. Halte das Rechteck am oberen Rand an. `if y > 200 then` ... `end` o.ä. könnte helfen.
2. Oben im Code, füge `velocity = 1` (Geschwindigkeit = 1) ein. Benutze `y = y - velocity` anstelle von `y = y - 1` um das Rechteck zu bewegen.
3. Reduziere `velocity` dauerhaft um 0.01. Das simuliert Erdanziehung.
4. Erhöhe `velocity`, wenn diese angeklickt wird (benutze `love.mousepressed()`).
5. Halte das Rechteck am unteren Rand an.
6. Zeige die Geschwindigkeit mit `love.graphics.print(velocity,10,10)`
7. Setze `velocity` gleich 0, wenn das Rechteck den oberen Rand berührt.
8. Gebe der Spielerin ein Ziel. Zeige an, wenn die Spielerin das Ziel erreicht hat. Z.B.: Ein einfaches Parken-Spiel. Zeichne eine Linie auf der Höhe 100, ändere die Farbe des Rechteckes, `if 0.5 > velocity and velocity > -0.5 and 105 > y and y > 95 then`.

3.4 Schleifen (Loops)

Die **while**-Schleife (während-loop) führt das Programm in dessen Rumpf/Körper aus, so lange die Schleifen-/Laufbedingung `y < 500` wahr ist.

```
1  x = 0
2
3  function love.draw()
4    y = 0
5
6    while y < 500 do
7      love.graphics.rectangle("fill",x,y,300,150)
8      y = y + 200
9    end
10 end
11
12 function love.mousepressed(mx, my)
13   x = x + 50
14 end
```

1. Vertausche Zeilen 7 und 8. Verstehst Du den Unterschied?
2. Verschiebe Zeile 4 in Zeile 2. Der Bildschirm sollte schwarz werden. Wieso?
3. Mache die Änderungen rückgängig. Dann zeichne mehr aber kleinere Rechtecke vertikal mithilfe der while-Schleife.
4. Füge eine neue Variable `z = 0` hinzu. Füge eine neue while-Schleife hinzu. Lasse diese `z` erhöhen und Rechtecke horizontal zeichnen.
5. Bewege `z = 0` und die neue Schleife in den Körper der alten Schleife. Nun wird durch alle z-Werte hindurchgegangen, wenn `y` erhöht wird! Zeichne so ein Schachbrettmuster.
6. Bewege das Schachbrett auf Mausklick.

3.5 Listen

```
1  a = {100,200,500}
2
3  function love.draw()
4    i = 1
5    while i <= 3 do
6      love.graphics.rectangle("fill",a[i],a[i],10,10)
7      i = i + 1
8    end
9  end
```

`a` ist eine Liste (Tabelle). `a[1]` ist gleich `100`. `1` ist hier der *Index* von `100` in `a`.

1. Füge eine Nummer in die Liste `a` hinzu. `#a` ist die Länge von `a`. Zeichne alle 4 Elemente.
2. `a[5] = 5 * 10` erstellt ein neues Element. Lasse mithilfe einer while-Schleife `a` gleich `{10,20,....,200}` sein.
3. `a[#a+1] = v` ist das gleiche, wie `v` zur Liste hinzuzufügen. Wenn die Maus geklickt wird, füge die x-Koordinate des Klicks zur Liste hinzu.