Greenfoot 3: KLASSISCHES Frogger-Game: Vorschläge für Unterrichtsverlauf

Die Schüler/innen haben das Spiel Frogger vorgestellt bekommen oder selbst gespielt (z.B. im Web) und kurz diskutiert, wie eine Minimalversion aussehen könnte.

Spielidee: Autos (Bus, Lastwagen …) fahren horizontal hin und her. Frosch muss das Spielfeld von unten nach oben überqueren. Kommt er oben an, erhält er einen Punkt und startet unten von Neuem. Wird er vom Auto überfahren, bekommt er einen Punkt Abzug. Über das Szenario bewegt sich eine Fliege, der der Frosch ebenfalls ausweichen muss. (Fliege wird entweder von zweitem Spieler gesteuert oder prallt in zufälligem Winkel vom Rand ab.)

Schritt 1: Grundlagen

- Neues Szenario anlegen
- Welt erstellen
- Klassen Frosch, Bus, Fliege erstellen
- Bilder für Klassen festlegen
- Neue Objekte erzeugen (Rechtsklick auf Klasse)

Themen:
Klassen, Objekte; Welt-Klasse; Hinweis auf automatische Kompilierung des Szenarios

Möglicher Arbeitsauftrag:
“Erstellen Sie ein neues Szenario »frogger« mit drei Klassen Frosch, Bus und Fliege. Geben Sie allen Klassen passende Bilder.”

Schritt 2: act, move,turn

- act-Methode
- Actor-Methoden move, turn
- this

Themen:
Methoden; act, move, turn; globaler Geschwindigkeitsregler; this; Einführung Terminologie “Parameter”

Möglicher Arbeitsauftrag:
“Bringen Sie die Klassen Frosch, Bus und Fliege dazu, sich zu bewegen.”

[Optional: Hier schon Einführung des Konstruktors, um Objekten bei ihrer Platzierung auf der Welt die gewünschte Ausrichtung zu geben.]

Schritt 3: Kollision mit Rand

- if-Abfrage: Kollision mit Rand (isAtEdge())
- zufällige Drehung

Themen:
Alternativlose if-Verzweigung; isAtEdge(); Greenfoot.getRandomNumber(…); Datentyp int

Möglicher Arbeitsauftrag:
“Der Bus soll rechts/links fahren. Wenn der Bus am Rand anstößt, soll er sich um 180 Grad drehen. Wenn die Fliege am Rand anstößt, soll sie sich um einen zufälligen Winkel drehen.”

[Optional: Hier schon Einführung der setLocation(…)-Methode, damit Frosch/Auto beim Anstoßen an den Rand auf der anderen Seite der Welt wieder auftauchen.]

Schritt 4: Objekte der Welt hinzufügen

- World-Methode addObject(…)

Themen:
Koordinatensystem der Greenfoot-Welt; addObject(…); neues Objekt erzeugen mit new Klassenname(); Wdh. Klasse, Objekt

Möglicher Arbeitsauftrag:
“Beim Erstellen der Welt soll ein Objekt der Klasse Frosch auf einer zufälligen x-Position am unteren Rand erscheinen; fünf Objekte der Klasse Auto verteilen sich am linken und rechten Rand. Ein Objekt der Klasse Fliege soll auf einer zufälligen Position auf der Welt erscheinen.”

Schritt 5: Tastatursteuerung

- Tastatursteuerung mit “Greenfoot.isKeyDown(…)”

Themen:
Greenfoot.isKeyDown(…); Datentyp String (Hochkommata!)

Möglicher Arbeitsauftrag:
“Der Frosch soll sich auf Tastendruck »w« nach oben bewegen. Die Fliege wird von einem zweiten Spieler so gesteuert: »up« = vorwärts, »left« = Drehung nach links, »right« = Drehung nach rechts.”

[Hinweis: Wurde setLocation(…) und getX()/getY() noch nicht eingeführt, muss die Aufwärtsbewegung durch eine Kombination aus move/turn gestaltet werden, z.B. turn(90);move(-2);turn(-90)]

Schritt 6: x-y-Koordinaten

- Methoden getX(), getY(), setLocation(…)

Themen:
Ausrichtung [optional: getRotation()]; Hinweis auf Methoden mit Rückgabetyp (getX() “tut” nichts, gibt nur einen Wert zurück) - vgl. getRandomNumber(…), isAtEdge()

Möglicher Arbeitsauftrag:
"Der Frosch soll sich mit »a« ein Stück nach links hüpfen, mit »d« ein Stück nach rechts."

Schritt 7: Kollidierende Objekte

- Kollisionsabfrage mit isTouching(…)

Themen:
Alternativlose Verzweigung bei isTouching(…); Wdh. Rückgabetyp, Wdh. setLocation(…)

Möglicher Arbeitsauftrag:
"Wenn der Frosch ein Auto berührt, soll er wieder auf seinen Ausgangspunkt gesetzt werden. (Alternativ: »…auf der gleichen x-Position an den unteren Bildschirmrand gesetzt werden …«) Wenn die Fliege den Frosch berührt, wird das Spiel mit Greenfoot.stop() beendet."

[Optional: Vor Stop Textausgabe mit getWorld().showText(…) für "Game over"]

Zwischenübung

Spätestens hier sollten die Schüler/innen das Gelernte in einem eigenen, komplett neuen Projekt anwenden, Vorgabe einer Checkliste (ausgedruckt, zum Abhaken), z.B.

Was wir bisher können:

- *ein neues Szenario erzeugen*
- *Klassen anlegen*
- *zu Beginn des Szenarios mindestens Objekte der Welt hinzufügen (feste und/oder zufällige Position)*
- *Verwendung von this*
- *move(), turn()*
- *Tastensteuerung*
- *Kollision mit Rand*
- *Kollision mit anderem Objekt*
- *Generierung von Zufallszahlen, z.B. für*
  - *zufällige Drehungen*
  - *zufällige Positionierung*
- *setLocation(...), auch mit getX() / getY()*

Für dieses Projekt sollten nicht (wesentlich) mehr als zwei Doppelstunden verwendet werden.

Sehr häufig kommt hier das Bedürfnis nach zufällig neu auftauchenden Objekten auf. Dann Vorgabe z.B. in der act()-Methode der World-Klasse:

```
if(Greenfoot.getRandomNumber(100) > 98)
{
    this.addObject(new Gegner(),
Greenfoot.getRandomNumber(this.getWidth()),
Greenfoot.getRandomNumber(this.getHeight()));
}
```

Weiter mit Frogger:

Schritt 8: Eigene Bilder verwenden

- eigene Bilder verwenden (im Ordner images)
- Bilder zur Laufzeit ändern mit setImage(…)

Themen:
setImage(…); Wdh. Datentyp String

Möglicher Arbeitsauftrag:
"Wenn der Frosch vom Auto überfahren wird, erhält er ein neues Bild. Wenn der Frosch die Fliege schnappt, erhält er ein neues Bild."
[Falls Fliege von Spieler gesteuert wird, fällt letzter Teil weg; vgl. Schritt 3 & 5)]
Achtung: PNGs mit Transparenz verwenden.

Schritt 9: Attribute

- Frosch erhält in Variable gespeicherte Anzahl Leben
- Lastwagen erhält in Variable gespeicherte Geschwindigkeit
- Frosch: Anzahl Leben verringern bei Kollision mit Lastwagen
- Lastwagen: Geschwindigkeit erhöhen bei Kollision mit Rand

Themen:
int-Attribute deklarieren (Kleinschreibung von Attributen!); Initialwert; Attributwerte verändern

Möglicher Arbeitsauftrag
"Der Frosch hat zu Beginn des Spiels 5 Leben. Jedes Mal, wenn er mit einem Lastwagen zusammenstößt, wird ihm ein Leben abgezogen.
Jeder Lastwagen erhält zu Beginn des Spiels eine zufällige Geschwindigkeit zwischen 1 und 4. Wenn er am Rand anstößt, wird seine Geschwindigkeit um 1 erhöht."

Schritt 10: Attribute in if-Verzweigungen

- Frosch speichert Anzahl erfolgreicher Überquerungen in Attribut
- Wenn Anzahl Überquerungen == 5, dann Textausgabe "Spiel gewonnen!"
- Wenn Anzahl Leben < 1, dann Textausgabe "Game over!" und Greenfoot.stop()
- Ausgabe von Punkten und Leben an unterschiedlichen Positionen

Themen:
Textausgabe mit showText(…), if-Verzweigung mit Vergleichsoperator, Abfrage von getX() als Kriterium für "oben angekommen"

Möglicher Arbeitsauftrag:
"Der Frosch erhält ein Attribut, in dem er die Anzahl erfolgreicher Überquerungen speichert. Wenn er die Straße erfolgreich überquert hat (d.h.: oben angekommen ist), wird der Wert dieses Attributs um 1 erhöht und der Frosch wieder nach unten gesetzt. Sobald der Frosch die Straße 5 Mal überquert hat, ist das Spiel vorbei (Greenfoot.stop() und Textausgabe »Game over«

Am Ende jeden act()-Zyklus‘ werden per showText(…) an unterschiedlichen Stellen Leben und Punkte ausgegeben (»Punkte: 2«, »Leben: 4«).”

[Optional: Mehrfach-Verzweigung - Bei <1 Leben Textausgabe “Game over”, bei <3 Leben Textausgabe “Jetzt wird’s eng”, ansonsten “Noch genug Leben!” - Achtung: Immer gleiche Position für Textausgabe benutzen.
Optional: Ausgabe des Variablenwerts in Form - »Game over, Straße überquert: 3«]

Schritt 11: Schuss abfeuern

- Frosch feuert mit “space” einen Schuss nach oben ab
- Wenn der Schuss einen Lastwagen trifft, verschwindet dieser (this.removeTouching(…)) und auch der Schuss (this.getWorld().removeObject(this))

Themen: getWorld()-Methode (um addObject(…)-Methode vom Actor aus aufrufen zu können), removeTouching(…), removeObject(…)

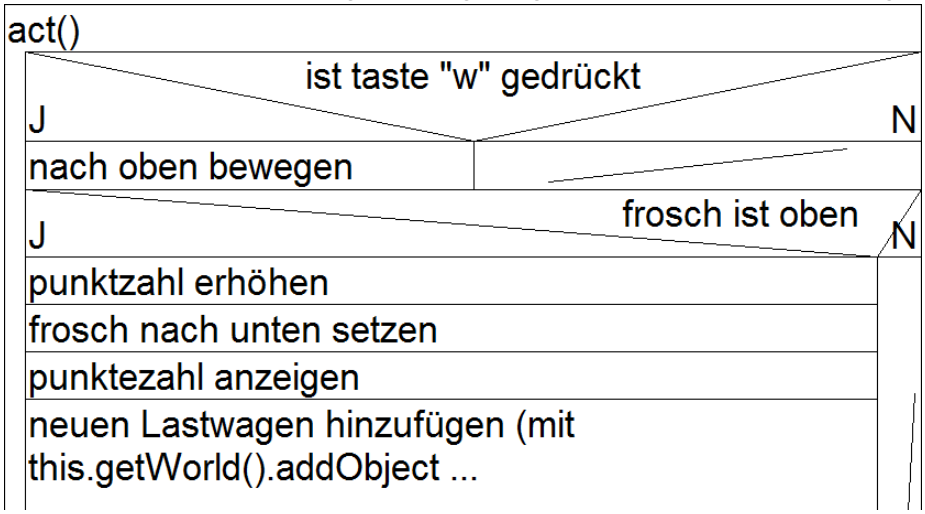
Möglicher Arbeitsauftrag:
“Bei Drücken der Taste »space« feuert der Frosch einen Schuss ab. Wenn der Schuss einen Lastwagen trifft, verschwindet sowohl der Lastwagen als auch der Schuss.”

Hinweise:

- Mit Greenfoot.isKeyDown(…) werden mehrere Schüsse abgefeuert (pro act-Zyklus einer). Besser ist die Verwendung von Greenfoot.getKey(), siehe z.B. http://greenfoot3.de/spielfigur-schiesst/ , dort auch Hinweise zu einstellbarem Dauerfeuer.
- Häufiger Fehler: Wenn der Schuss zuerst verschwindet, kann er den Lastwagen nicht verschwinden lassen -> Reihenfolge beachten

Struktogramm - Einführung

Im Verlauf können Struktogramme gezeigt und in den Unterricht eingebracht werden, z.B.



Im Anschluss sollten die erarbeiteten Inhalte vertieft werden, evtl. wieder mit Checkliste. Die Schüler/innen arbeiten frei an einem eigenen Projekt. Alte Computerspiele der 80er eignen sich wegen ihrer Einfachheit recht gut als Vorlagen, z.B.
Burning Rubber: https://www.youtube.com/watch?v=bMUEyk2D_wA
Asteroids: https://www.youtube.com/watch?v=kIMplupTJm0

Häufig wird ein zufälliges Auftauchen von Gegnern o.ä. benötigt, siehe dazu oben "Zwischenübung".

Wichtiger Hinweis für die Schüler/innen: Keine zu komplizierten Szenarien planen!