GTA Einführung Robotik mit Makecode

Mattias Schlenker

Wilhelm-Ostwald-Gymnasium

25. Februar 2021

Viele, viele Essensrationen

In dieser Stunde wollen wir unsere Map um Essenrationen oder Münzen erweitern, die Punkte oder zusätzliche Leben geben. Realisiert werden soll das mit Sprites, die ihre Position behalten:

Figure: Kuchenstücke

Wie realisieren wir das ohne jeden Sprite von Hand zu positionieren?

Arrays oder Listen

Die verwendete Datenstruktur nennt sich Liste oder Array, Ihr kennt sie vom Einkaufen:

- Milch
- Kartoffeln
- Nuss-Nougat-Creme
- Rosenkohl

Beim Einkaufen nehmt Ihr immer ein Element von der Liste, packt es in den Einkaufswagen und streicht es ab.

Listen in der Informatik

In Pseudocode schreibt Ihr Listen Komma getrennt in eckigen Klammern. Unsere Liste für die x-Koordinaten der Essensrationen könnte also so aussehen:

```
xPositionenKuchen = [ 30, 50, 70, 110, 130, 150 ]
```

Vorarbeit in Makecode

Um besser arbeiten zu können, setzt zunächst im Spielupdate-Block die Wahrscheinlichkeit für Haie auf 0%. Dann soll der Affe anhalten, wenn das Steuerkreuz nicht mehr gedrückt wird. Dafür wird das Controller-Event „Taste losgelassen" benutzt:

Figure: Controller-Events

Listen in MakeCode

Klappt in der Kategorieübersicht ,,Fortgeschritten" auf und sucht in ,,Arrays" nach dem Block, der einer Variable ein Array/eine Liste zuweist. Diese Liste hat zunächst zwei Elemente. Gebt der Variablen den Namen ,,xPositionenKuchen" und tragt die ersten beiden Werte der Liste von der vorletzten Folie ein.

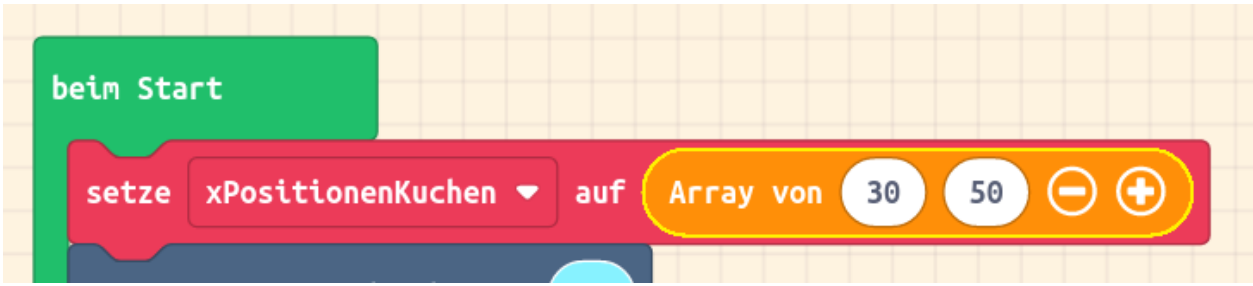

Figure: Liste mit zwei Elementen

Liste erweitern

Verwendet den Plus-Knopf, um diese Liste zu erweitern. Tragt für Eure Map sinnvolle Werte ein. Denkt daran, dass eine Kachel und unsere Standard-Sprites je 16 Pixel Seitenlänge haben:

Figure: Liste wird länger

Array-Operationen

Achtung: Hier arbeiten wir mit Listen variabler Länge, fragt in Microcontroller- oder Robitik-Projekten Eure Dozenten, ob Listen in der verwendeten Programmierumgebung feste oder variable Längen haben!
Wichtige Array-Operationen sind:

- Rufe ein durch einen Index angegebenes Element ab (bspw. das dritte hat Index 2)
- Nehme den ersten Wert der Liste (Liste wird kürzer)
- Nehme den letzten Wert der Liste (Liste wird kürzer)
- Füge am Anfang oder Ende ein Element an

Mit den letzten drei Operationen kann man eine Liste wie eine Warteschlange (vorne wegnehmen und hinten anhängen) oder einen Stapel (was zuletzt draufgelegt wurde, wird als erstes weggenommen) behandeln.

Kuchen platzieren 1

Nun bietet es sich an, solange X-Koordinaten von der Liste wegzunehmen, bis die Liste leer ist und erst dann zum nächsten Block zu gehen. Wir fügen also einen Block ,,während" ein, der aus Logik einen Vergleichsoperator ,,größer" mitbekommt:

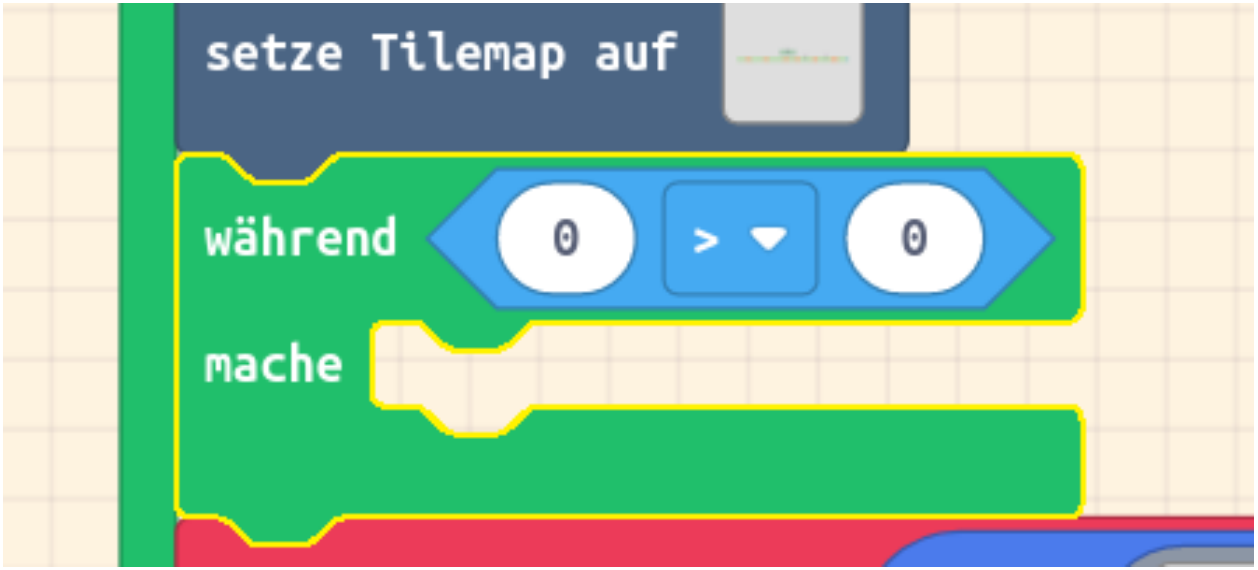

Figure: Vorbereitung der Prüfung auf Listenlänge

Kuchen platzieren 2

An die erste Stelle des Vergleichsoperators setzen wir nun die Länge des Arrays „xPositionenKuchen”:

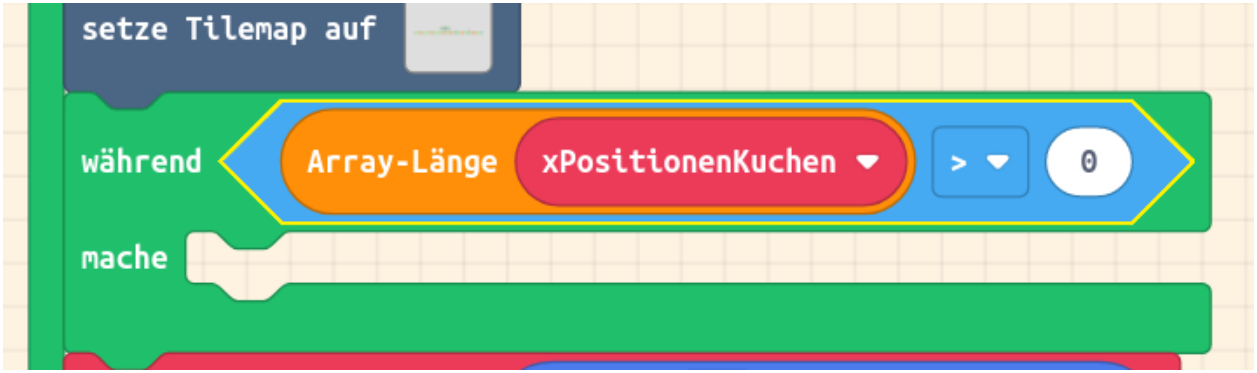

Figure: Prüfung auf Listenlänge – sind noch Werte in der Liste?

Da die Liste nie kürzer wird, bleibt Euer Bildschirm schwarz, wenn Ihr das Spiel jetzt startet...

Kuchen platzieren 3

Jetzt kommt ein Sprite der Art ,,Food" ins Spiel. Als X-Position bekommt er den ersten Wert der Liste ,,xPositionenKuchen", die Y-Position weit genug vom oberen Rand, damit der Kuchen durch Hüpfen erreichbar ist . Die Liste wird dadurch in jedem Durchgang kürzer. Ist die Länge 0, wird der nächste Block angesprungen:

Figure: Liste verkürzen

Hmmmm...

Alle Kuchen auf derselben Höhe? Das ist doch langweilig oder?
Klar, wir hatten immer den gleichen Y-Wert...

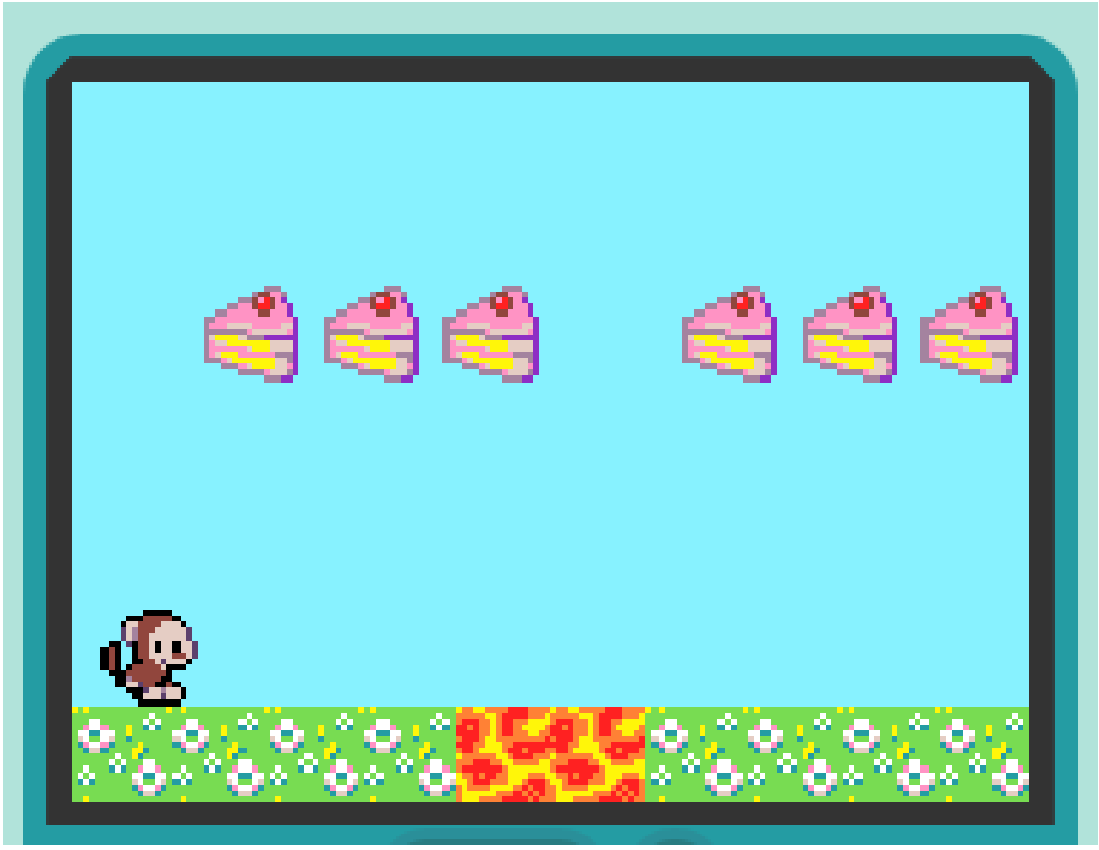

Figure: Kuchenhöhe in eigener Liste

Liste für die Y-Werte

OK, erstellen wir durch Kopieren und umbenennen eine Liste ,,yPositionenKuchen" in der wir die Höhe zwischen 80 und 60 Pixeln vom oberen Rand variieren:

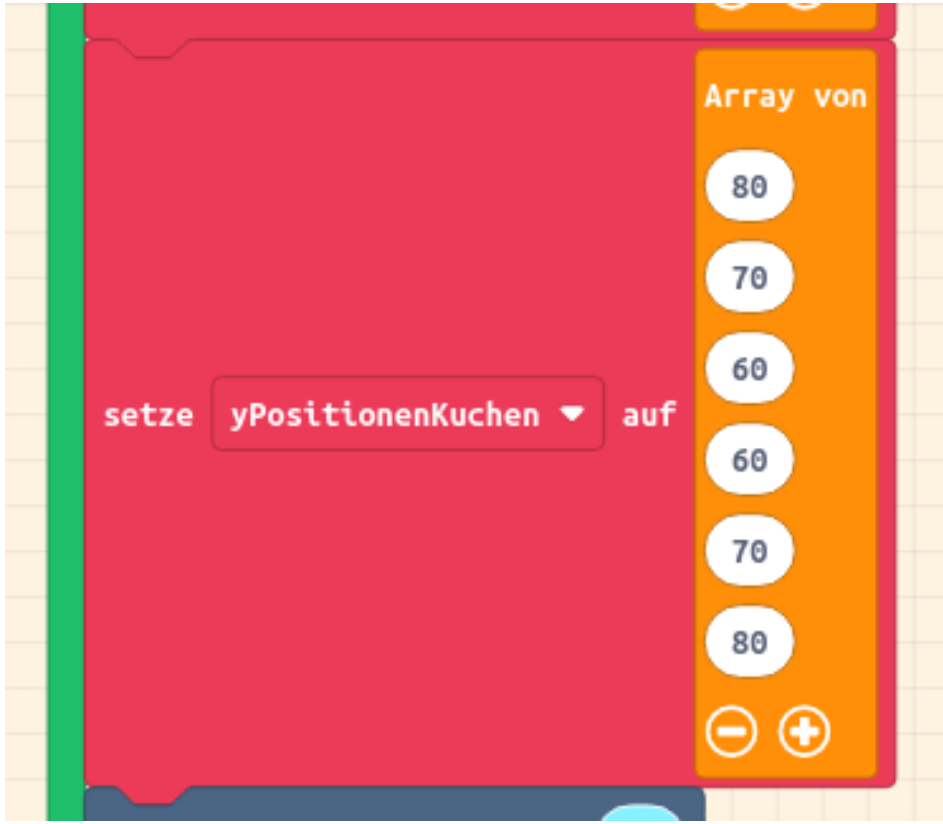

Figure: Kuchenhöhe als Liste

Y-Werte aus Liste entnehmen

Für die Y-Koordinate kopieren wir den Block ,,rufe den ersten Wert ab und lösche ihn von xPositionenKuchen" und benennen die Variable um in ,,yPositionenKuchen". Dann schieben wir ihn an die Stelle der bisher fixen Höhe:

Figure: Kopiert und umbenannt...

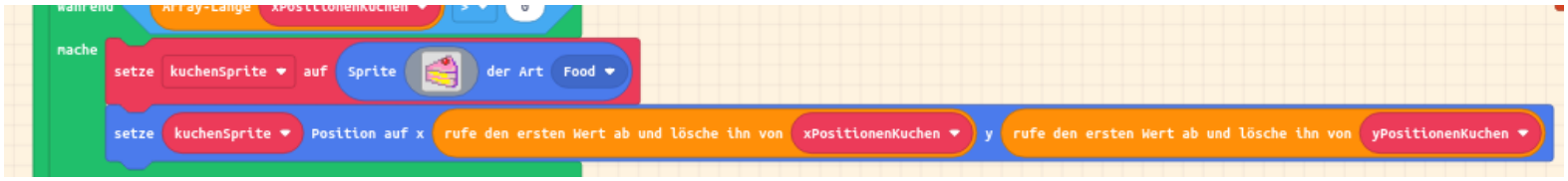

Figure: ...und an der richtigen Stelle eingefügt

Kollisionserkennung

Die Kollisionserkennung funktioniert wie beim Shooter. Nichts neues hier. Vergesst nicht, im Start-Block die Punktzahl auf 0 zu setzen:

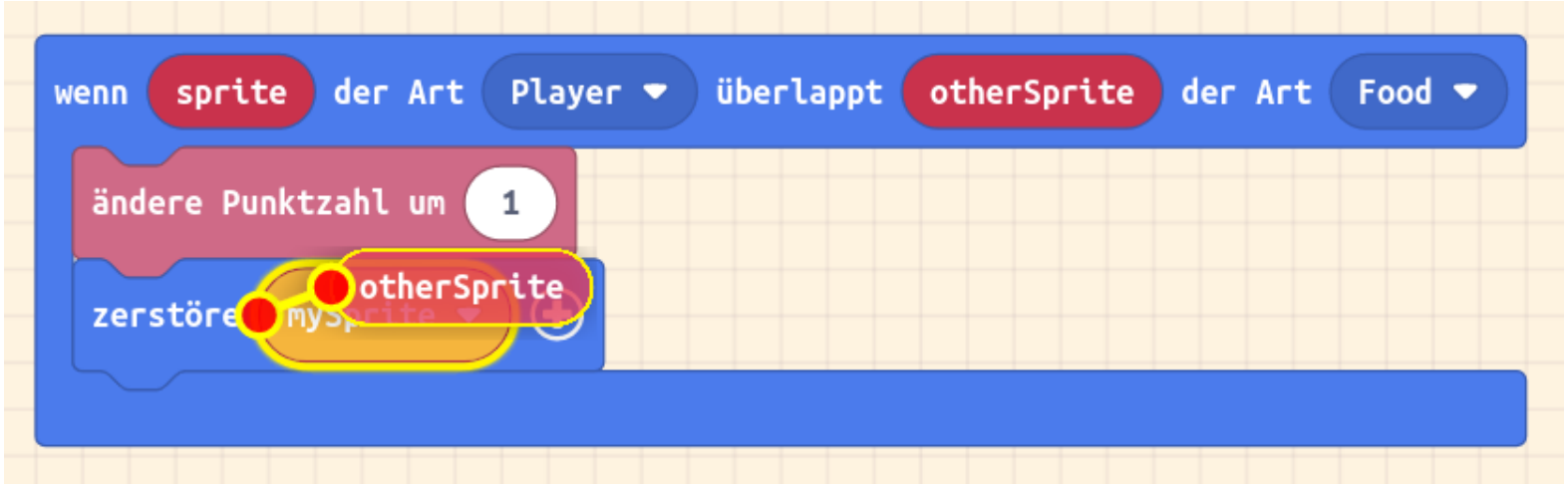

Figure: Zerstören des Food-Sprites und Inkrementieren der Punkte

Download des Musterspiels

Wie die letzten Termine steht die UF2-Datei in LernSax, alternativ könnt Ihr hier direkt weitermachen:
https://makecode.com/_Jjh8rqMsa8y0
Wer arbeitet an einer spannenderen Map mit mehr Essenrationen und vielleicht Zusatzleben nach Kontakt mit einem Gegner?
In den nächsten Sessions arbeiten wir weiter an den eingereichten Vorschlägen.