GTA Einführung Robotik mit Makecode

Mattias Schlenker

Wilhelm-Ostwald-Gymnasium

17. Dezember 2020

Wir programmieren einen Space-Shooter

Nicht jeder hat einen Roboter abbekommen, zudem haben manche Probleme mit dem UF2-Upload. Also üben wir Makecode, indem wir ein Spiel programmieren. Es soll ein klassischer Weltraumshooter werden: Wir fliegen in Richtung des oberen Bildschirmrandes. Von dort kommen und feindliche Raumschiffe entgegen, mit denen wir nicht kollidieren dürfen, die wir aber abschießen können.
Geht auf https://arcade.makecode.com/ und erstellt ein neues Projekt, bspw. „GtaAlienShooter"...

Die Szenerie

Nehmt aus ,,Effekte" den Effekt für den Startbildschirm und setzt ihn auf den Sternenfeld-Effekt. Aus den ,,Sprites" holt Ihr den Block zum erzeugen eines eigenen Sprites. Achtet darauf, dass der Typ auf ,,Player" steht. Klickt in das graue Feld und malt ein Raumschiff, das auf dem dunklen Hintergrund gut aussieht.

Figure: Startszene

Das Raumschiff bewegen

Nehmt aus ,,Controller" den Block für die Bewegung mit Knöpfen, zudem müsst Ihr sicherstellen, dass das Raumschiff innerhalb des Bildschirmes bleibt. Den benötigten Block findet Ihr unter ,,Sprites".

Figure: Mit Steuerung

Alien-Raumschiffe als Gegner

Zunächst holen wir einen Block ,,Spielupdate" und setzen ihn auf eine Sekunde. In diesen kommt ein neues Sprite, das als ,,Projektil von der Seite" (=Bildschirmrand) definiert ist. Die X-Geschwindigkeit setzen wir auf 0. Im zweiten Block wird die Position des Sprites gesetzt. Für die X-Position ziehen wir dafür aus ,,Mathematik" eine Zufallszahl, die wir auf 0 bis 160 setzen.

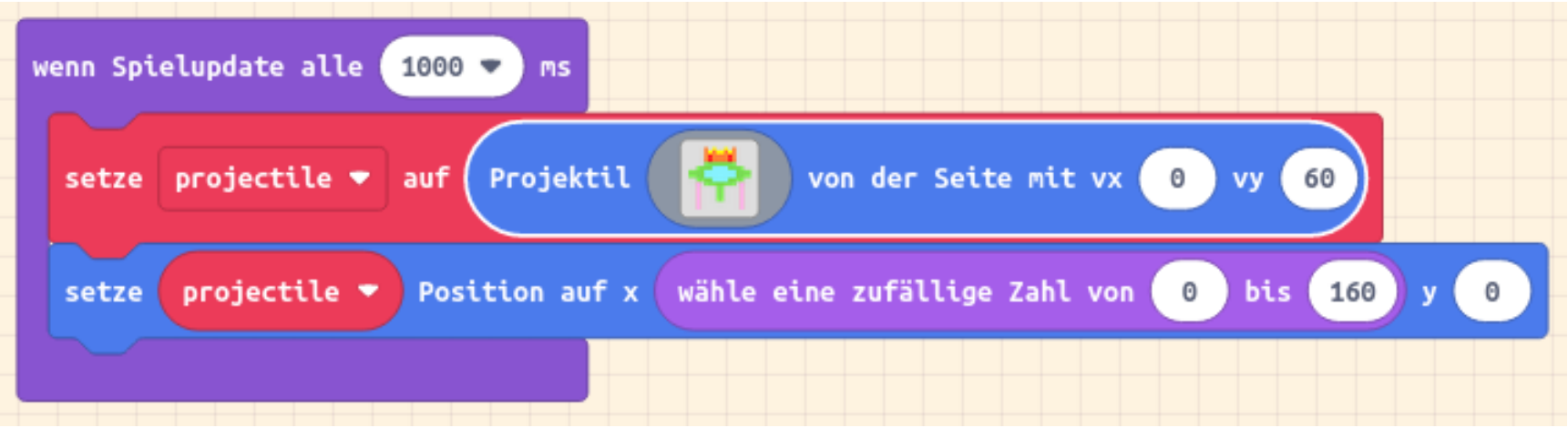

Figure: Alien-Raumschiff

Kollisionserkennung

Zuerst müssen wir das gegnerische Raumschiff einer Kategorie zuweisen: ,,Enemy" (Gegner). Dann holen wir aus ,,Sprites" den Block für ,,Kollisions-Events", hier als ,,Überlappung" definiert und setzen per Dropdown eine Überlappung des eigenen Sprites der Kategorie ,,Player" mit dem fremden Sprite der Klasse ,,Enemy".

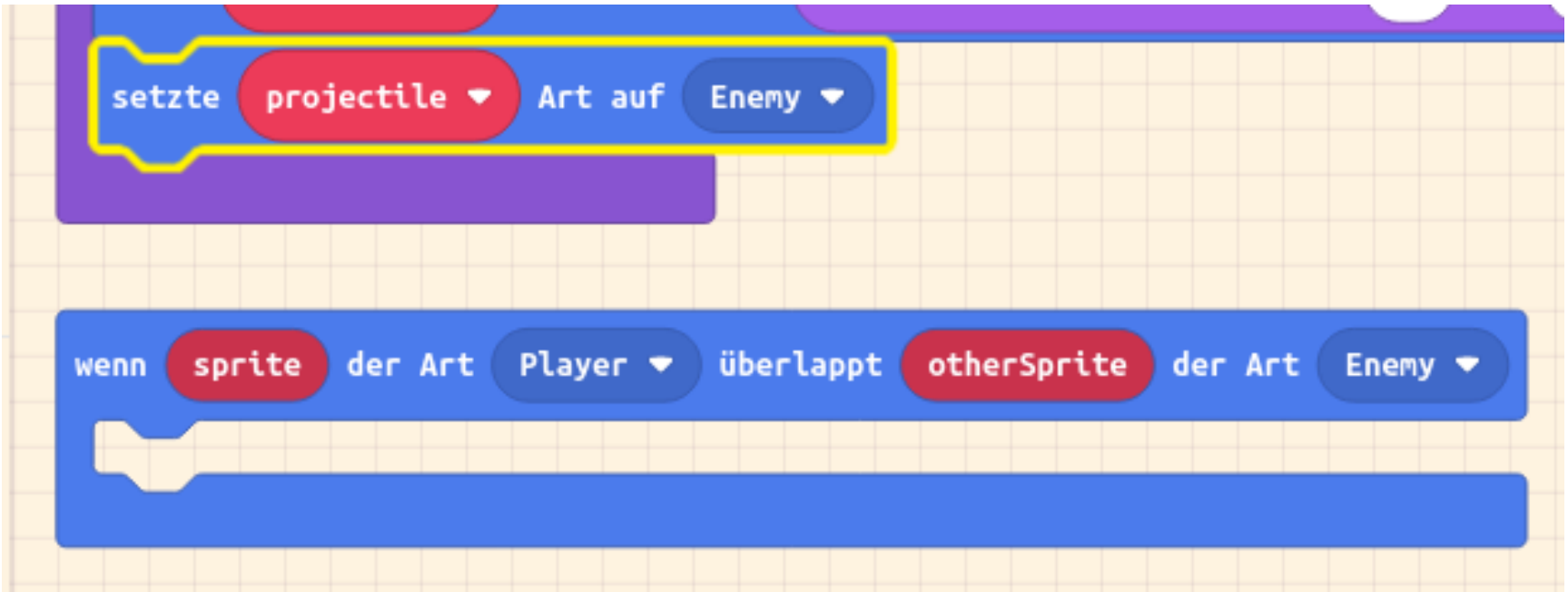

Figure: Kollisionserkennung

Sprite zerstören

Nehmt nun einen Block ,,Zerstöre" aus den ,,Sprites" und zieht dann das ovale ,,otherSprite" in das ovale Feld im ,,Zerstöre-Block" - wenn Ihr mögt, bringt ein Klick auf das Plus-Symbol Effekte zum Vorschein:

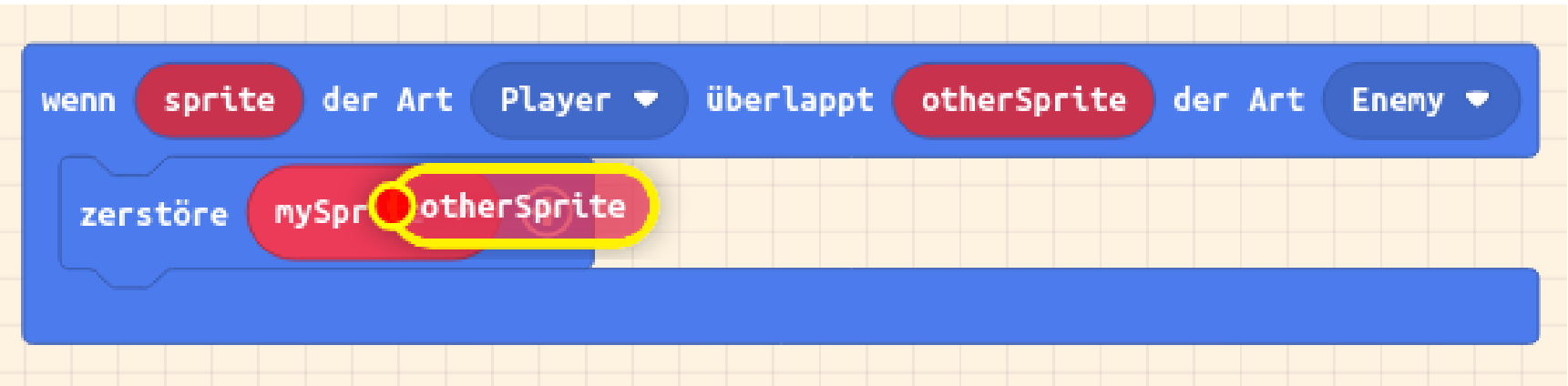

Figure: Kollisionserkennung

Weniger Leben

Unter ,,Info" findet Ihr Aktionen und Variablen, die den Spielstand betreffen. Nehmt hier einen Block ,,ändere Leben" und zieht ihn mit in den Block für das Kollisionsevent. Aus ,,Szene" könnt ihr passende Effekte auswählen.

Figure: Weniger Leben

Schießen

Schießen ist ein ,,Controller-Event": ,,Wenn A-Taste gedrückt". Holt aus den ,,Sprites" nun den Block, der ein Sprite erzeugt, der nicht am Bildschirmrand, sondern in einem anderen Sprite den Ursprung hat, hier in ,,mySprite", also an der Position des Spielers. Ändert die Leinwandgröße auf 12x12 Pixel und zeichnet etwas Schönes. Ordnet dann das Projektil der Kategorie ,,Projectile" zu. Optional könnt Ihr den Schüssen Effekte mitgeben.

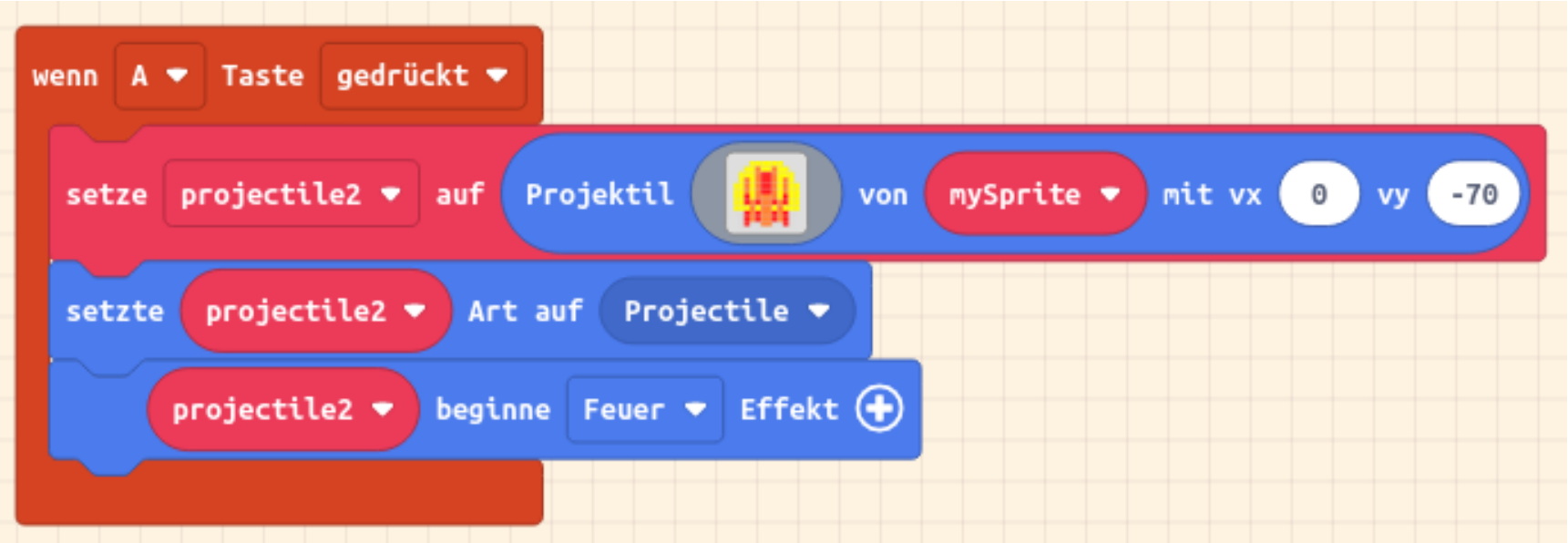

Figure: Schießen mit A

Punkte

Um mit Spielständen arbeiten zu können, müsst Ihr aus ,,Info" den Block zur Initialisierung der Punktzahl in den grünen Block ,,beim Start" ziehen:

Figure: Punktzahl

Treffer landen

Für den Treffer benötigt Ihr wieder ein Block für das Überlappungs-Event. Hier sind die Klassen ,,Projectile" und ,,Enemy" auszuwählen. Zerstört beide Sprites und erhöht die Punktzahl um 1.

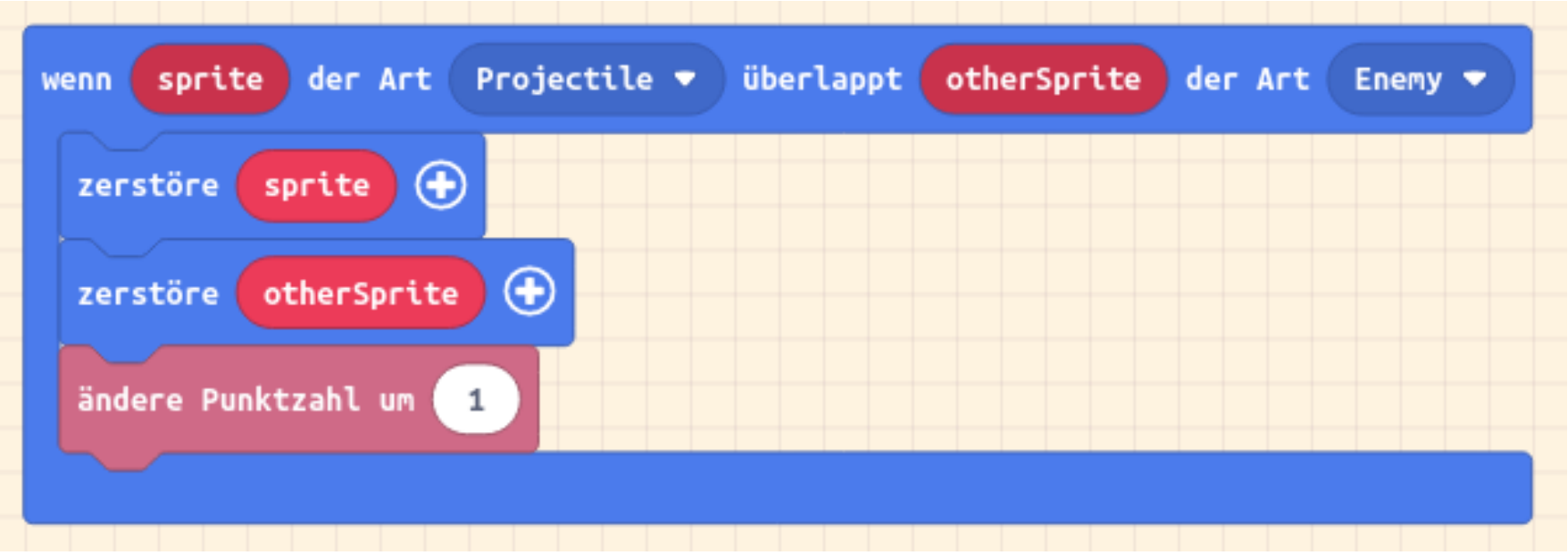

Figure: Punktzahl

Mehr Leben

Holt diese vier Blöcke aus ,,Mathemathik", ,,Logik" und ,,Info":

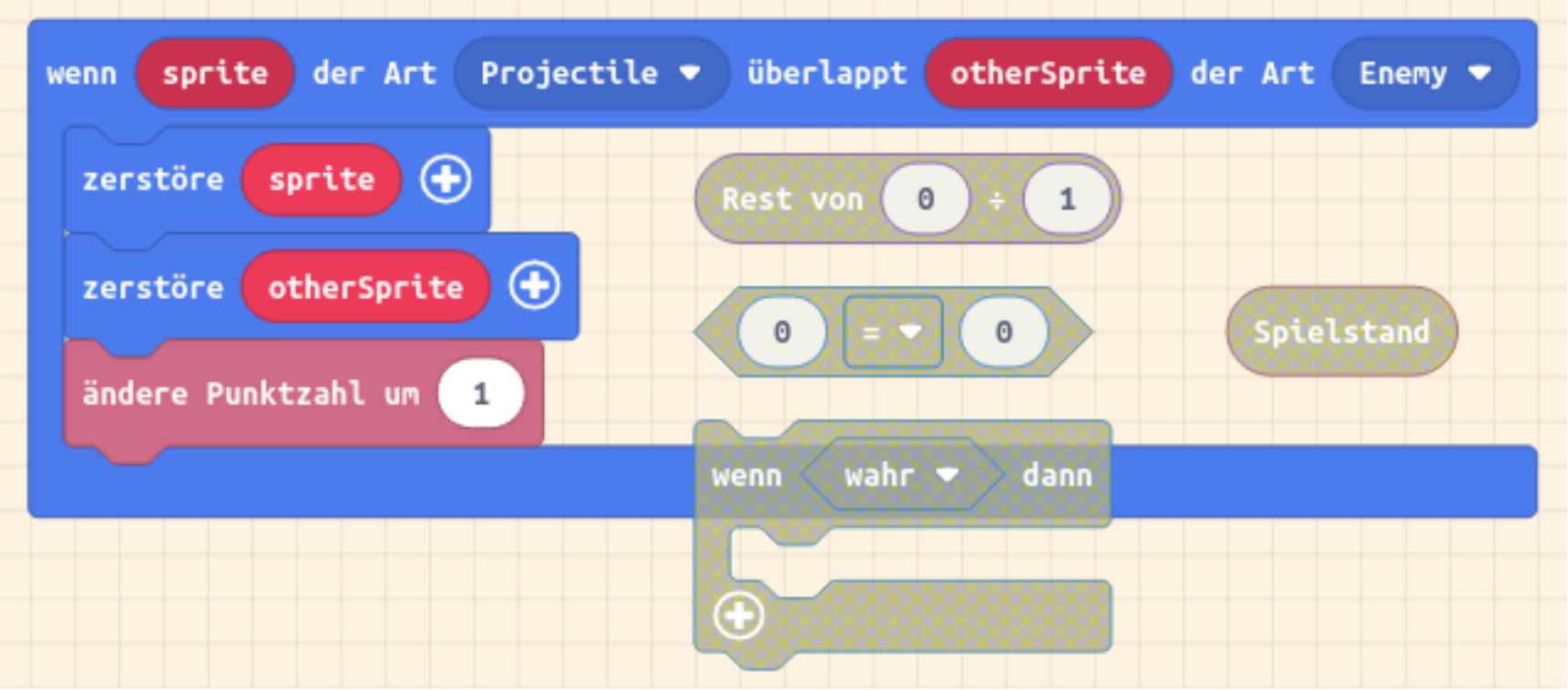

Figure: Vier Blöcke...

Mehr Leben

Ordnet sie nun so an, dass die Zahl der Leben um 1 erhöht wird, wenn der Rest der Division des Spielstandes durch 20 gleich 0 ist (die Operation ,,Rest nach Division" heisst auch ,,Modulo"):

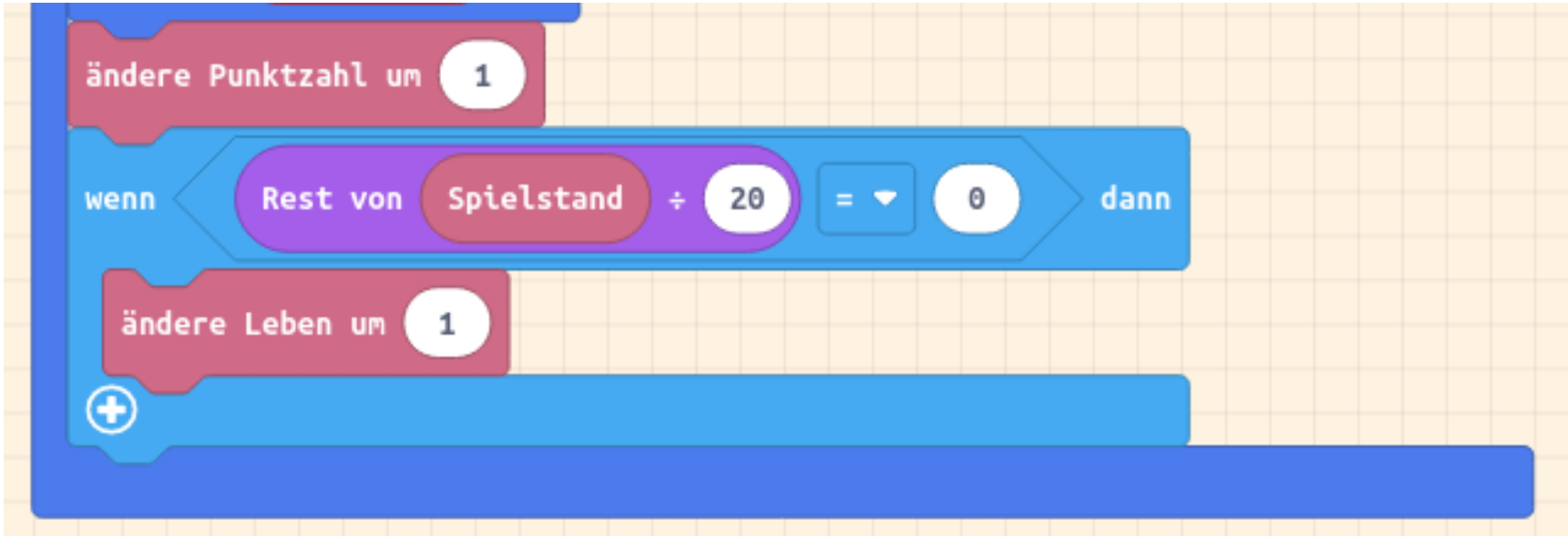

Figure: Mehr Leben

Schnellere Gegner

Um die Geschwindigkeit der Gegner variabel zu machen, benötigen wir eine - richtig Variable. Erzeugt unter ,,Variablen" eine neue Variable ,,vGegner" und initialisiert diese im Startblock mit 60.

Figure: vGegner

Schnellere Gegner

Im sekündlichen Update wird diese Variable nun ins Feld für ,,vy" eingesetzt:

Figure: vGegner

Schnellere Gegner

Und im Block für die Überlappung zwischen ,,Projectile" und ,,Enemy" wird mit einer weiteren Modulo-Operation dafür gesorgt, dass Ihr alle 10 Treffer eine neues Leben bekommt.

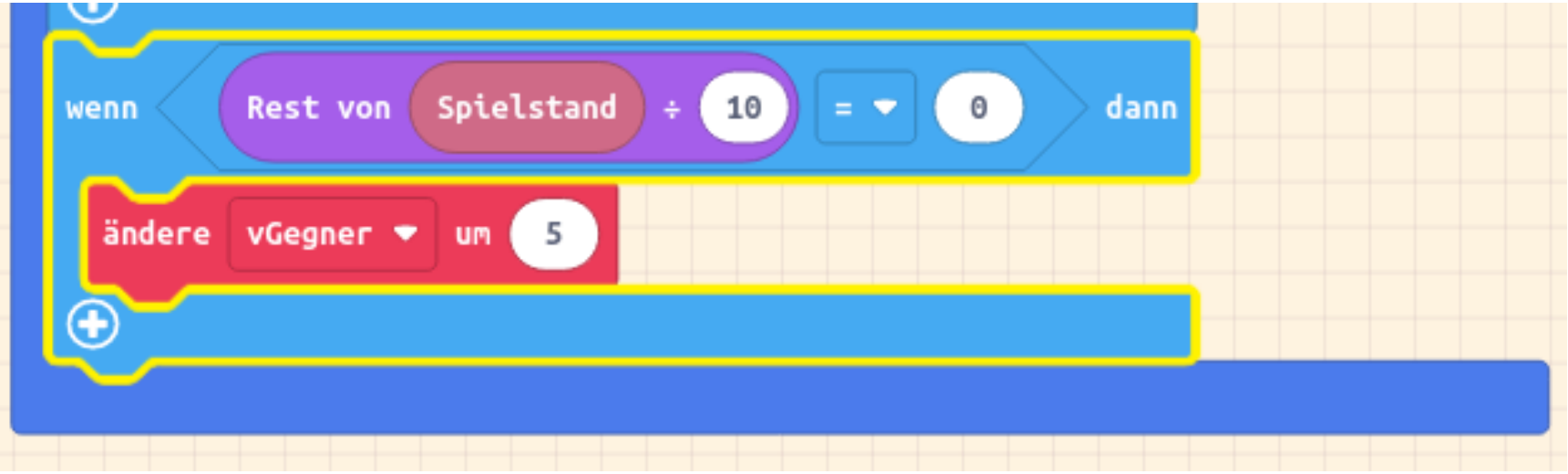

Figure: vGegner

Fertig!

Wenn Ihr Euren Code mit meinem vergleichen wollt oder einfach nur spielen möchtet:
https://makecode.com/_6bUK5eAqF2pu

Was habt Ihr gelernt?

- Rechteckige Blöcke stehen für Schleifen und Events
- Blöcke mit Puzzle-Teil oben und unten gehören in rechteckige Blöcke
- Blöcke mit spitzer Klammer stehen für Aussagenlogik
- Ovale Blöcke rufen Variablen ab oder liefern Ergebnisse von Berechnungen
- Farbe hilft bei der Zuordnung zu Gruppen
- Der Modulo-Operator ist praktisch für gegelmäßige Ereignisse

Ideen für Erweiterungen

Das wurde genannt:

- Asteroiden, denen man ausweichen muss, die man aber nicht abschießen kann
- Versorgungsraumschiffe, die die Zahl der Leben erhöhen
- Spezielle Objekte, die alle Gegner für eine Zeit verlangsamen
- Mehr Effekte
- Schöne Start- und Endbildschirme
- Animation der Sprites, z.b. leichtes Kippen nach rechts oder links bei Bewegung
- schräg fliegende oder die Richtung ändernde Objekte, denen man ausweichen muss
- Geschwindigkeitserhöhung nach Zeit und nicht nach Treffern
- Begrenzte Schusszahl pro Zeiteinheit (ähnlich Token Bucket)

Links

Und hier die Hardware, auf der die Spiele direkt laufen können:

- Meow Bit
- PyGamer
- Raspberry Pi

Da bereits der Wunsch geäußert wurde, so ein Arcade-Cabinet zu bauen, ich habe alles vorrätig und wenn sich im Schulclub ein Platz dafür findet, warum nicht einen eigenen Automaten für die Best-of-WOG-Games?!