

Enten-Alarm – Übungsheft

Dein Spiel: Ein Boot weicht kackenden Enten aus. 5 Leben. Mit der Zeit wird es schwerer. Dieses Heft enthält alle Übungen des Tages. Kurz erklärt, Schritt für Schritt. Wenn du hängst: Hand heben.

Immer wieder: „Jetzt speichern“ tippen.

Farben der Blöcke: Bewegung blau · Aussehen lila · Klang pink · Ereignisse gelb · Steuerung orange · Fühlen hellblau · Operatoren grün · Variablen dunkelorange

Block 1: Ankommen & Scratch einrichten

Kurz erklärt: Scratch hat vier Bereiche: **Bühne** (rechts oben, hier läuft das Spiel), **Figurenliste** (rechts unten: Boot, Ente, Kacke), **Blockpalette** (links, alle Befehle nach Farben) und **Skriptbereich** (Mitte, hier baust du). Ein **Skript** ist eine Kette von Blöcken. Es startet mit einem gelben **Hut-Block** wie „Wenn  angeklickt wird“.

iPad-Bedienung: Block antippen, halten, ziehen. Einrasten, wenn grauer Schatten erscheint. Block löschen = zurück in die Palette ziehen. Lange drücken = Menü (Duplizieren, Löschen). Zoom mit den Knöpfen + – = unten rechts, nicht mit zwei Fingern.

Übung 1.1: Der Klasse beitreten

1. iPad **quer** halten. **Kamera-App** öffnen, auf den **QR-Code am Beamer** halten, **unten** auf das gelbe Feld **scratch.mit.edu** tippen.
2. Einladung zur Klasse **PWSK 26** → grüner Knopf „**Loslegen**“.
3. **Benutzername** ausdenken: 3–20 Zeichen, Buchstaben, Zahlen, - und _ . Fang mit **ente** an, z.B. ente_blitz7. Nicht deinen echten Namen nehmen.
4. **Passwort** ausdenken, mindestens 6 Zeichen. Haken bei „**Passwort anzeigen**“ hilft beim Tippen.
5. **Beides sofort auf deine Karte schreiben**. Ohne Passwort kommst du nach der Pause nicht zurück.
6. „**Nächster Schritt**“, restliche Fragen ausfüllen (Geburtsmonat und -jahr, Geschlecht, Land), abschicken.

Klappt der QR-Code nicht? Der Link steht auch auf deiner Karte.

Übung 1.2: Projekt holen

1. **Zweiten QR-Code** am Beamer scannen — er öffnet das Studio „**Entenpaddeln-Basis**“. (Der Link steht auch auf deiner Karte.)
2. Auf das Projekt „**Entenspiel Basis**“ tippen.
3. Grünen Knopf **Remixen** tippen.
4. Namen oben links ändern: Entenspiel <dein Benutzername>
5. **Jetzt speichern**.

Kein Knopf „Remixen“? Du bist nicht angemeldet — oben rechts prüfen, ob dein Nutzernamenasteht.

Übung 1.3: Erkunden

1. Tippe in der Figurenliste auf **Ente**. Was ändert sich?
2. Tab **Kostüme**: Wie viele Kostüme hat die Ente?
3. Zurück zu **Skripte**, wieder **Boot** wählen.

Übung 1.4: Das Boot sagt „Hallo!“

Figur **Boot** ausgewählt.

1. **Ereignisse** (gelb) → „Wenn  angeklickt wird“ in den Skriptbereich ziehen.
2. **Aussehen** (lila) → „sage (Hallo!) für (2) Sekunden“ darunter einrasten.
3. Grüne Flagge → Sprechblase erscheint.
4. Text und Zahl ändern, nochmal testen.

Wenn  angeklickt wird
sage (Hallo!) für (2) Sekunden

Warum funktioniert das? Der Hut-Block wartet auf ein Ereignis (Flagge). Alles, was unter ihm hängt, läuft dann von oben nach unten. Blöcke ohne Hut-Block passieren bei der Flagge nicht. Tippst du einen Block direkt an, läuft er einmal.

Herausforderungen

- Zweiten „sage“-Block anhängen: ein Gespräch.
- **Klang** (pink) → „spiele Klang (hit) ganz“ anhängen.
- **Ereignisse** → „Wenn diese Figur angeklickt wird“ + „sage“: Boot antippen.
- Der **Ente** ein eigenes Skript bauen. Jede Figur hat ihre eigenen Skripte.

Check

- ☐ Ich bin bei Scratch angemeldet.
- ☐ **Nutzername und Passwort stehen auf meiner Karte.**
- ☐ Ich habe mein eigenes Projekt Entenspiel <Benutzername> (Remix).
- ☐ Ich weiß, wo Bühne, Figurenliste, Blockpalette und Skriptbereich sind.
- ☐ Das Boot sagt etwas, wenn ich die grüne Flagge drücke.
- ☐ Ich habe auf „Jetzt speichern“ getippt.

Block 2: Das Boot steuern

Kurz erklärt: Scratch beschreibt jeden Punkt der Bühne mit **x** (links -240 bis rechts 240) und **y** (unten -180 bis oben 180). Die Mitte ist x: 0, y: 0. Eine **Endlosschleife** („wiederhole fortlaufend“) führt ihren Inhalt immer wieder aus, etwa 30 Mal pro Sekunde. Eine **Bedingung** („falls ... dann“) führt ihren Inhalt nur aus, wenn die Frage im sechseckigen Loch mit Ja beantwortet wird.

iPad: „Maustaste gedrückt?“ = dein Finger berührt die Bühne. „Maus x-Position“ = wo dein Finger links/rechts ist.

Übung 2.1: Koordinaten ablesen

1. **Boot** wählen. Unter der Figurenliste stehen die Felder **x** und **y**.
2. Boot mit dem Finger auf der Bühne herumziehen. Die Zahlen ändern sich.
3. Boot ganz nach unten in die Mitte ziehen. Ungefähr x: 0, y: -140?

Übung 2.2: Boot an den Start setzen

Figur **Boot**. Falls der „sage“-Block aus Block 1 noch unter dem Flaggen-Block hängt: abziehen und in die Palette werfen.

1. **Bewegung** (blau) → „gehe zu x: () y: ()“ unter „Wenn  angeklickt wird“.
2. Eintragen: **x: 0, y: -140**.
3. Boot irgendwohin ziehen, grüne Flagge. Es springt zurück.

```
Wenn  angeklickt wird
gehe zu x: (0) y: (-140)
```

Übung 2.3: Das Boot folgt dem Finger

1. **Steuerung** (orange) → „wiederhole fortlaufend“ unter „gehe zu“.
2. **Steuerung** → „falls < > dann“ in die Schleife.
3. **Fühlen** (hellblau) → „Maustaste gedrückt?“ in das sechseckige Loch.
4. **Bewegung** → „gehe zu x: () y: ()“ in das „falls“. Bei **y** die **-140** eintragen.
5. **Fühlen** → „Maus x-Position“ in das runde Loch bei **x**.
6. Grüne Flagge. Finger auf die Bühne legen und nach links/rechts ziehen. Speichern.

```
Wenn  angeklickt wird
gehe zu x: (0) y: (-140)
wiederhole fortlaufend
  falls <Maustaste gedrückt?> dann
    gehe zu x: (Maus x-Position) y: (-140)
```

Warum funktioniert das? Die Schleife prüft ständig: Liegt ein Finger auf der Bühne? Wenn ja, wird x des Boots auf x des Fingers gesetzt. y bleibt -140, deshalb bleibt das Boot unten. Sechseckige Blöcke sind Fragen (Ja/Nein), runde Blöcke sind Werte (Zahl/Text).

Herausforderungen

- „falls“ herausnehmen, nur den „gehe zu“-Block in der Schleife. Was ist anders?
- **Auch hoch und runter:** die -140 bei **y** durch „Maus y-Position“ ersetzen.
- **Weich fahren (★ ★):** statt dem „gehe zu“-Block ändere x um $((\text{Maus x-Position}) - (\text{x-Position})) / (4))$. Zwei grüne Operatoren: der „-“-Block steckt im „/“-Block. Das Boot zieht weich nach, statt zu springen. Statt 4 auch 2 oder 10 probieren.

Check

- ☐ Ich weiß, was x und y auf der Bühne bedeuten.
- ☐ Mein Boot steht beim Start unten in der Mitte (x: 0, y: -140).
- ☐ Ich habe eine „wiederhole fortlaufend“-Schleife im Skript.
- ☐ Das Boot folgt meinem Finger nach links und rechts.
- ☐ Ich habe gespeichert.

Block 3: Animation

Kurz erklärt: Jede Figur kann mehrere Bilder haben, die **Kostüme**. „nächstes Kostüm“ springt zum nächsten Bild, nach dem letzten wieder zum ersten. „warte“ bremst die Schleife, sonst siehst du nur Flackern. Eine Figur kann **mehrere Skripte gleichzeitig** laufen lassen: eins steuert, eins paddelt.

Übung 3.1: Kostüme anschauen

1. **Boot** wählen, Tab **Kostüme**: Boot1 und Boot2 antippen.
2. Dasselbe bei der **Ente**: Ente1 und Ente2.
3. Zurück zum Tab **Skripte**.

Übung 3.2: Das Boot paddelt

Figur **Boot**. Ein **neues** Skript neben das Steuer-Skript legen, nicht anhängen.

1. **Ereignisse** → „Wenn  angeklickt wird“ an eine freie Stelle.
2. **Steuerung** → „wiederhole fortlaufend“ darunter.
3. **Aussehen** → „nächstes Kostüm“ in die Schleife.
4. **Steuerung** → „warte (1) Sekunden“ darunter in die Schleife. Auf **0.3** ändern (Punkt, kein Komma).
5. Grüne Flagge. Speichern.

```
Wenn  angeklickt wird
wiederhole fortlaufend
  nächstes Kostüm
  warte (0.3) Sekunden
```

Übung 3.3: Die Ente flattert

Figur **Ente**. Gleiches Skript, Wartezeit **0.2**.

```
Wenn  angeklickt wird
wiederhole fortlaufend
  nächstes Kostüm
  warte (0.2) Sekunden
```

Abkürzung: Skript des Boots mit dem Finger auf die **Ente** in der Figurenliste ziehen. Es wird kopiert. Dann nur die Zahl ändern.

Herausforderungen

- Tab Kostüme, Boot2 lange drücken → „Duplizieren“, etwas dazu malen. Drei Bilder.
- Ente: drittes Skript mit „ändere Größe um (5)“, „warte (0.2)“, „ändere Größe um (-5)“, „warte (0.2)“ in einer Schleife.

- Boot: „nächstes Kostüm" und „warte" in ein „falls <Maustaste gedrückt?> dann" legen. Paddelt nur, wenn der Finger auf der Bühne liegt.
- Schleife mit „ändere Effekt [Farbe] um (10)" (Aussehen). Regenbogen-Ente.

Check

- ☐ Ich weiß, wo die Kostüme einer Figur sind.
- ☐ Das Boot wechselt beim Spielen zwischen Boot1 und Boot2.
- ☐ Die Ente flattert.
- ☐ Ich verstehe: Eine Figur kann mehrere Skripte gleichzeitig laufen lassen.
- ☐ Ich habe gespeichert.

Block 4: Die Ente fliegt

Kurz erklärt: Jede Figur hat eine **Richtung** wie ein Kompass: 90 = rechts, -90 = links, 0 = oben, 180 = unten. „gehe ()er Schritt" bewegt sie in diese Richtung. „pralle vom Rand ab" dreht die Richtung am Bühnenrand um. Der **Drehtyp** bestimmt, wie das Bild dabei aussieht: „rundum" dreht mit (kopfüber), „links-rechts" spiegelt nur. Eine **Zufallszahl** würfelt jedes Mal neu.

Übung 4.1: Richtung ablesen

1. **Ente** wählen. Unter der Bühne das Feld **Richtung** antippen. Ein Drehrad erscheint.
2. Auf 180 drehen. Was passiert? Zurück auf **90**.

Übung 4.2: Die Ente fliegt los

Figur **Ente**. Neues Skript neben dem Flatter-Skript.

1. **Ereignisse** → „Wenn  angeklickt wird".
2. **Bewegung** → „gehe zu x: (-200) y: (130)".
3. **Bewegung** → „setze Richtung auf (90)".
4. **Steuerung** → „wiederhole fortlaufend".
5. **Bewegung** → „gehe (10)er Schritt" in die Schleife, auf **4** ändern.
6. Grüne Flagge. Die Ente fliegt nach rechts und bleibt am Rand hängen.

```
Wenn  angeklickt wird
gehe zu x: (-200) y: (130)
setze Richtung auf (90)
wiederhole fortlaufend
  gehe (4)er Schritt
```

Übung 4.3: Am Rand umdrehen

1. **Bewegung** → „pralle vom Rand ab" in die Schleife unter „gehe (4)er Schritt".
2. Grüne Flagge. Schau hin: Auf dem Rückweg fliegt sie **kopfüber**.
3. **Bewegung** → „setze Drehtyp auf [links-rechts]" unter „setze Richtung auf (90)", vor der Schleife.
4. Grüne Flagge. Jetzt spiegelt sie sich richtig. Speichern.

```
Wenn  angeklickt wird
gehe zu x: (-200) y: (130)
setze Drehtyp auf [links-rechts]
setze Richtung auf (90)
wiederhole fortlaufend
  gehe (4)er Schritt
  pralle vom Rand ab
```

Warum funktioniert das? „pralle vom Rand ab" hat die Bedingung „berühre ich den Rand?" schon eingebaut und dreht dann die Richtung um. Der Drehtyp „links-rechts" sorgt dafür, dass das Bild nur gespiegelt und nicht gedreht wird.

Übung 4.4: Zufall ausprobieren

1. **Operatoren** (grün) → „Zufallszahl von (1) bis (10)“ in der Palette mehrmals antippen. Jedes Mal eine andere Zahl.
2. Den Block in das runde Loch von „gehe (4)er Schritt“ ziehen. Auf **2 bis 6** ändern.
3. Grüne Flagge. Die Ente ruckelt, weil sie in jedem Durchlauf anders weit springt.
4. Zufallszahl-Block wieder herausziehen, **4** eintragen.

Merke: Runde Wert-Blöcke passen in jedes runde Loch, auch anstelle einer Zahl.

Herausforderungen

- Wackelflug: „ändere y um (Zufallszahl von (-2) bis (2))“ zusätzlich in die Schleife.
- „warte (Zufallszahl von (0) bis (0.05)) Sekunden“ in die Schleife: mal schneller, mal langsamer.
- Im „gehe zu x: y:“ das y durch „Zufallszahl von (100) bis (150)“ ersetzen. Andere Starthöhe.
- Tab Klänge → Lautsprecher → Bibliothek → „Duck“. In der Schleife: „falls <wird [Rand] berührt?> dann spiele Klang [Duck]“.

Check

- ☐ Die Ente startet oben links und fliegt nach rechts.
- ☐ Sie dreht am Rand um.
- ☐ Sie fliegt auf dem Rückweg **nicht** kopfüber (Drehtyp links-rechts).
- ☐ Ich weiß, was der Zufallszahl-Block macht und wo er hinpasst.
- ☐ Ich habe gespeichert.

Block 5: Es kackt!

Kurz erklärt: Es gibt nur eine Figur „Kacke“, aber viele sollen gleichzeitig fallen. Dafür gibt es **Klone**: Kopien, die das Spiel selbst erzeugt, während es läuft. Die Figur ist der Stempel und bleibt versteckt. Jeder Klon startet mit „Wenn ich als Klon entstehe“ sein eigenes Skript und verschwindet mit „lösche diesen Klon“. Alle drei Blöcke stehen in **Steuerung** ganz unten.

Übung 5.1: Die Ente stempelt

Figur **Ente**. Drittes Skript.

1. **Ereignisse** → „Wenn  angeklickt wird“.
2. **Steuerung** → „wiederhole fortlaufend“.
3. **Steuerung** → „warte (1) Sekunden“ in die Schleife, auf **2** ändern.
4. **Steuerung** → „erzeuge Klon von (mir selbst)“ darunter. Dropdown → **Kacke**.
5. Grüne Flagge. Noch passiert nichts Sichtbares. Die Klone wissen noch nicht, was sie tun sollen.

```
Wenn  angeklickt wird
wiederhole fortlaufend
  warte (2) Sekunden
  erzeuge Klon von [Kacke]
```

Übung 5.2: Das Original versteckt sich

Figur **Kacke**.

1. **Ereignisse** → „Wenn  angeklickt wird“.
2. **Aussehen** → „verstecke dich“ darunter.

```
Wenn  angeklickt wird
verstecke dich
```

Übung 5.3: Der Klon fällt

Figur **Kacke**, neues Skript.

1. **Steuerung** → „Wenn ich als Klon entstehe“.
2. **Bewegung** → „gehe zu (Zufallsposition)“. Dropdown → **Ente**.
3. **Bewegung** → „ändere y um (10)“, auf **-20** ändern.

4. **Aussehen** → „zeige dich“.
5. **Steuerung** → „wiederhole bis < >“.
6. **Operatoren** → „() < (50)“ in das sechseckige Loch.
7. **Bewegung** → runder Block „y-Position“ in das linke Loch von „<“. Rechts **-170**.
8. **Bewegung** → „ändere y um (10)“ in die Schleife, auf **-5** ändern.
9. **Steuerung** → „lösche diesen Klon“ **unter** die Schleife, nicht hinein.
10. Grüne Flagge. Alle 2 Sekunden fällt eine Kacke. Speichern.

```
Wenn ich als Klon entstehe
gehe zu [Ente]
ändere y um (-20)
zeige dich
wiederhole bis <(y-Position) < (-170)>
  ändere y um (-5)
lösche diesen Klon
```

Warum funktioniert das? Der Klon erbt alles vom Original, auch das Versteckt-Sein, darum „zeige dich“. „wiederhole bis“ fällt so lange, bis die Frage „y-Position < -170“ (fast ganz unten) wahr ist. „berührt Rand?“ wäre unzuverlässig: Die Ente prallt am Rand ab, ein Klon dort würde sofort gelöscht.

Herausforderungen

- „ändere y um (-5)“ auf -8 oder -12 ändern. Ab wann wird es unfair?
- Bei der Ente „warte (2)“ auf 1 oder 0.5 ändern.
- In der Fall-Schleife „ändere x um (Zufallszahl von (-2) bis (2))“: Kacke trudelt.
- „drehe dich um (5) Grad“ in der Schleife. Dann am Anfang „setze Richtung auf (90)“.

Check

- ☐ Ich kann in einem Satz erklären, was ein Klon ist.
- ☐ Die Ente erzeugt alle 2 Sekunden einen Klon der Kacke.
- ☐ Das Original der Kacke ist versteckt.
- ☐ Die Kacke-Klone fallen von der Ente nach unten und verschwinden.
- ☐ Ich habe auf „Jetzt speichern“ getippt.

Block 6: Treffer & Leben

Kurz erklärt: Eine **Variable** ist ein beschrifteter Becher: Name „Leben“, Inhalt eine Zahl. „setze“ füllt, „ändere“ nimmt raus oder legt dazu, der runde Block zeigt den Inhalt. **Nachrichten** sind eine Lautsprecher-Durchsage: „sende (Treffer) an alle“ ruft, „Wenn ich (Treffer) empfangen“ reagiert. So meldet die Kacke nur, und das Boot entscheidet, was beim Treffer passiert.

Übung 6.1: Variable „Leben“ anlegen

Bühne wählen (rechts neben den Figuren).

1. **Variablen** (dunkelorange) → „Neue Variable“ → Name Leben → **Für alle Figuren** → OK.
2. Häkchen neben dem runden Block „(Leben)“ anlassen. Die Variable wird auf der Bühne angezeigt.
3. **Ereignisse** → „Wenn  angeklickt wird“. **Variablen** → „setze (Leben) auf (0)“, auf **5** ändern.

```
Wenn  angeklickt wird
setze [Leben] auf (5)
```

Übung 6.2: Die Kacke merkt den Treffer

Figur **Kacke**, Klon-Skript erweitern.

1. **Operatoren** → „< > oder < >“ an eine freie Stelle.
2. Den Block „(y-Position) < (-170)“ aus „wiederhole bis“ ins **linke** Loch von „oder“ ziehen.
3. **Fühlen** → „wird (Mauszeiger) berührt?“ ins **rechte** Loch. Dropdown → **Boot**.
4. Den ganzen „oder“-Block zurück in „wiederhole bis“.
5. **Steuerung** → „falls < > dann“ zwischen Schleife und „lösche diesen Klon“.

6. **Fühlen** → „wird (Boot) berührt?“ ins Loch von „falls“.

7. **Ereignisse** → „sende (Nachricht1) an alle“ ins „falls“. Dropdown → „Neue Nachricht“ → Treffer.

```
Wenn ich als Klon entstehe
gehe zu [Ente]
ändere y um (-20)
zeige dich
wiederhole bis <(y-Position) < (-170)> oder <wird [Boot] berührt?>
  ändere y um (-5)
falls <wird [Boot] berührt?> dann
  sende [Treffer] an alle
lösche diesen Klon
```

Übung 6.3: Das Boot reagiert

Figur **Boot**, neues Skript.

1. **Ereignisse** → „Wenn ich (Treffer) empfangen“.
2. **Variablen** → „ändere (Leben) um (1)“, auf **-1** ändern.
3. **Steuerung** → „wiederhole (10) mal“, auf **3** ändern.
4. **Aussehen** → „setze Effekt (Farbe) auf (0)“ in die Schleife. Dropdown → **Durchsichtigkeit**, Wert **80**.
5. **Steuerung** → „warte (0.1) Sekunden“.
6. Effekt-Block lange drücken → „Duplizieren“, Wert auf **0**. Noch ein „warte (0.1)“.
7. Grüne Flagge, treffen lassen: Blinken, Leben 5 → 4. Speichern. (Ein Geräusch kommt in Block 9 dazu.)

```
Wenn ich [Treffer] empfangen
ändere [Leben] um (-1)
wiederhole (3) mal
  setze Effekt [Durchsichtigkeit] auf (80)
  warte (0.1) Sekunden
  setze Effekt [Durchsichtigkeit] auf (0)
  warte (0.1) Sekunden
```

Übung 6.4: Game Over

Bühne, Leben-Skript erweitern:

1. **Steuerung** → „warte bis <>“ unter „setze Leben auf 5“.
2. **Operatoren** → „() < (50)“ ins Loch. **Variablen** → „(Leben)“ links, rechts **1**.
3. **Ereignisse** → „sende () an alle“ darunter. Dropdown → „Neue Nachricht“ → Game Over.

```
Wenn  angeklickt wird
setze [Leben] auf (5)
warte bis <(Leben) < (1)>
sende [Game Over] an alle
```

Boot, neues Skript:

1. **Ereignisse** → „Wenn ich (Game Over) empfangen“.
2. **Aussehen** → „sage (Hallo!) für (2) Sekunden“, Text Game Over!.
3. **Steuerung** → „stoppe (alle)“ darunter.

```
Wenn ich [Game Over] empfangen
sage (Game Over!) für (2) Sekunden
stoppe [alle]
```

Kacke, neues Skript:

1. **Ereignisse** → „Wenn ich (Game Over) empfangen“.
2. **Steuerung** → „lösche diesen Klon“ darunter.

Wenn ich [Game Over] empfangen
lösche diesen Klon

Grüne Flagge, fünfmal treffen lassen. „Game Over!“, alles stoppt. Speichern.

Warum funktioniert das? Die Bühne steht bei „warte bis“ und wartet, bis der Becher „Leben“ leer ist. Dann ruft sie „Game Over“, und alle, die zuhören, reagieren: Boot sagt Bescheid und stoppt alles, jeder Kacke-Klon löscht sich.

Herausforderungen

- Boot sagt beim Treffer kurz „Autsch!“ („sage ... für 0.5 Sekunden“ im Treffer-Skript).
- Ente hört auch auf „Treffer“: „ändere Größe um 20“, warte 0.2, „ändere Größe um -20“.
- Statt Blinken schrumpft das Boot: „ändere Größe um -10“. Am Anfang „setze Größe auf 100 %“.
- Mit 3 Leben starten. Oder jede Kacke zieht 2 Leben ab. Was passiert mit „Leben < 1“?

Check

- ☐ Die Variable „Leben“ ist bei der Bühne angelegt, „Für alle Figuren“, und wird auf der Bühne angezeigt.
- ☐ Am Start steht Leben auf 5.
- ☐ Trifft eine Kacke das Boot: Blinken, Leben -1.
- ☐ Bei 0 Leben sagt das Boot „Game Over!“ und alles stoppt.
- ☐ Ich habe auf „Jetzt speichern“ getippt.

Block 7: Schwieriger mit der Zeit

Kurz erklärt: Bisher kackt die Ente fest alle 2 Sekunden. Kommt die Zahl in eine Variable **Pause**, kann die Bühne sie während des Spiels immer kleiner drehen. Die Ente liest bei jedem Durchgang neu nach. Eine Untergrenze verhindert, dass die Pause negativ wird und die Ente hunderte Kacken pro Sekunde stempelt.

Übung 7.1: Punkte zählen

Bühne, zweites Skript.

1. **Variablen** → „Neue Variable“ → Punkte → Für alle Figuren. Noch einmal → Pause.
2. Häkchen: **Punkte** an, **Pause** aus.
3. **Ereignisse** → „Wenn  angeklickt wird“.
4. **Variablen** → „setze (Punkte) auf (0)“, dann „setze (Pause) auf (2)“.
5. **Steuerung** → „wiederhole fortlaufend“. Darin „warte (1) Sekunden“ und „ändere (Punkte) um (1)“.

```
Wenn  angeklickt wird
setze [Punkte] auf (0)
setze [Pause] auf (2)
wiederhole fortlaufend
  warte (1) Sekunden
  ändere [Punkte] um (1)
```

Übung 7.2: Die Stellschraube einbauen

Ente, Stempel-Skript: **Variablen** → runder Block „(Pause)“ in das Zahlenfeld von „warte (2) Sekunden“ ziehen.

```
Wenn  angeklickt wird
wiederhole fortlaufend
  warte (Pause) Sekunden
  erzeuge Klon von [Kacke]
```

Bühne, Punkte-Skript, in der Schleife unter „ändere Punkte um 1“:

1. **Steuerung** → „falls < > dann“.
2. **Operatoren** → „() > (50)“ ins Loch. **Variablen** → „(Pause)“ links, rechts **0.4**.
3. **Variablen** → „ändere (Pause) um (1)“ ins „falls“, Wert **-0.05**.

4. Grüne Flagge, durchhalten: nach 20 Sekunden Pause 1, nach 30 Sekunden 0.5. Speichern.

```
Wenn  angeklickt wird
setze [Punkte] auf (0)
setze [Pause] auf (2)
wiederhole fortlaufend
  warte (1) Sekunden
  ändere [Punkte] um (1)
  falls <(Pause) > (0.4)> dann
    ändere [Pause] um (-0.05)
```

Warum funktioniert das? „ändere Pause um -0.05“ heißt Pause = Pause - 0.05, jede Sekunde. Das „falls > 0.4“ stoppt das Schrumpfen bei 0.4. Ohne Grenze würde die Pause negativ, und „warte -1 Sekunden“ wartet gar nicht mehr.

Herausforderungen

A: Ente wird schneller (★★★)

1. Bühne: Neue Variable Tempo, Häkchen aus. Im Punkte-Skript oben „setze (Tempo) auf (4)“, in der Schleife „ändere (Tempo) um (0.1)“.
2. Ente, Flug-Skript: „(Tempo)“ in das Feld von „gehe (4)er Schritt“.

B: Zweite Ente nach 30 Sekunden (★★★★) Ein Klon der Ente reicht nicht: Der Kacke-Klon geht mit „gehe zu (Ente)“ immer zum Original.

1. **Ente** in der Figurenliste lange drücken → „Duplizieren“ → Ente2.
2. **Kacke** lange drücken → „Duplizieren“ → Kacke2. Bei Kacke2: „gehe zu (Ente)“ → **Ente2**.
3. Ente2, Stempel-Skript: „erzeuge Klon von (Kacke)“ → **Kacke2**. Vor der Schleife „warte (30) Sekunden“.
4. Ente2, Flug-Skript: unter dem Hut-Block „verstecke dich“, „warte (30) Sekunden“, „zeige dich“. Start x auf **200**.

C: Untergrenze 0.4 auf 0.2 ändern. 60 Sekunden spielen. Noch fair?

Check

- ☐ Die Variable „Punkte“ zählt jede Sekunde hoch und wird angezeigt.
- ☐ Die Variable „Pause“ steuert die Wartezeit der Ente.
- ☐ Die Pause wird jede Sekunde kleiner, aber nie kleiner als 0.4.
- ☐ Ich kann erklären, warum die Untergrenze wichtig ist.
- ☐ Ich habe auf „Jetzt speichern“ getippt.

Block 8: Feinschliff

Kurz erklärt: „() mod ()“ (Operatoren) ist der Rest beim Teilen: $23 \text{ mod } 10 = 3$, $30 \text{ mod } 10 = 0$. Ist der Rest 0, ist die Zahl durch 10 teilbar. „verknüpfe () und ()“ klebt Text und Zahl zu einem Satz zusammen. Alles, was sich im Spiel verändert, muss am Start einmal zurückgesetzt werden: der **Startzustand**.

Übung 8.1: Hintergrundmusik

Bühne, neues Skript.

```
Wenn  angeklickt wird
wiederhole fortlaufend
  spiele Klang [music_loop] ganz
```

Übung 8.2: Belohnungs-Sound alle 10 Punkte

Bühne, Punkte-Skript, direkt nach „ändere Punkte um 1“:

1. **Steuerung** → „falls < > dann“.
2. **Operatoren** → „() = ()“ ins Loch.
3. **Operatoren** → „() mod ()“ ins **linke** Feld von „=“.
4. **Variablen** → „(Punkte)“ links in „mod“, rechts **10**. Rechtes Feld von „=“: **0**.

5. **Klang** → „spiele Klang (point)" ins „falls" (ohne „ganz").

```
wiederhole fortlaufend
  warte (1) Sekunden
  ändere [Punkte] um (1)
  falls <((Punkte) mod (10)) = (0)> dann
    spiele Klang [point]
  falls <(Pause) > (0.4)> dann
    ändere [Pause] um (-0.05)
```

Tipp: Den „mod"-Block erst frei im Skriptbereich fertig bauen, dann als Ganzes in „=" ziehen, dann ins „falls".

Übung 8.3: Game Over mit Punktzahl

Boot, Skript „Wenn ich (Game Over) empfangen":

1. **Operatoren** → „verknüpfe (apple) und (banana)" ins Textfeld des „sage"-Blocks.
2. Links: Game Over! Punkte: (Leerzeichen am Ende). Rechts: **Variablen** → „(Punkte)".
3. Zeit auf **3**.

```
Wenn ich [Game Over] empfangen
sage (verknüpfe (Game Over! Punkte: ) und (Punkte)) für (3) Sekunden
stoppe [alle]
```

Übung 8.4: Sauberer Neustart

Boot, Steuer-Skript, nach „gehe zu x: 0 y: -140" zwei Blöcke aus **Aussehen** einfügen:

```
Wenn  angeklickt wird
gehe zu x: (0) y: (-140)
setze Effekt [Durchsichtigkeit] auf (0)
wechsle zu Kostüm [Boot1]
wiederhole fortlaufend
  falls <Maustaste gedrückt?> dann
    gehe zu x: (Maus x-Position) y: (-140)
```

Speichern. Dein Basis-Spiel ist fertig.

Herausforderungen

- **Highscore:** Neue Variable Highscore (Für alle Figuren, Häkchen an). Im Game-Over-Skript des Boots vor „sage": „falls <(Punkte) > (Highscore)> dann setze (Highscore) auf (Punkte)". **Nicht** bei der Flagge auf 0 setzen.
- **Startbildschirm:** Boot vor der Schleife „sage (Tippe auf die Bühne und weiche den Enten aus!) für (2) Sekunden". Bei Ente und Bühne in jedes Schleifen-Skript nach dem Hut-Block „warte bis <Maustaste gedrückt?>".
- **Blinkende Ente:** Ente: „Wenn ich (Treffer) empfangen" → „setze Effekt (Farbe) auf (50)" → „warte (0.2)" → „setze Effekt (Farbe) auf (0)".

Check

- ☐ Die Musik läuft in Dauerschleife.
- ☐ Alle 10 Punkte kommt ein Sound.
- ☐ Beim Game Over zeigt das Boot die Punktzahl.
- ☐ Nach Neustart ist das Boot wieder voll sichtbar und hat Kostüm Boot1.
- ☐ Ich habe gespeichert.

Block 9: Kreativzeit

So gehst du vor: Erst drei Minuten still überlegen, was deinem Spiel fehlt, und die Idee auf deine Ideenkarte schreiben. Wer mag, stellt sie kurz vor. Dann bauen: lieber klein und fertig als groß und halb.

Mit dem einfachsten Teil anfangen. Nach jedem Block testen. Alle 5 Minuten speichern. Vieles, was du brauchst, hast du in Block 2 bis 8 schon gebaut. Schau dort nach.

Check

- ☐ Ich habe mir eine Idee ausgesucht.
- ☐ Mindestens ein Teil davon funktioniert im Spiel.
- ☐ Das Spiel läuft noch ohne Fehler durch (grüne Flagge bis Game Over).
- ☐ Ich habe gespeichert.

Block 10: Präsentation & Abschluss

Kurz erklärt: Dein Workshop-Konto ist ein Schüler-Konto und wird nach dem Kurs geschlossen. Dein Projekt darin verschwindet dann. Dein Spiel bleibt dir nur als **Datei**, deshalb sichern wir es jetzt und schicken es an den Laptop der Leitung.

Übung 10.1: Als Datei sichern

1. **Jetzt speichern.**
2. **Datei** → **Auf deinen Computer herunterladen.**
3. Safari fragt: **Laden.**
4. Die Datei Entenspiel <Benutzername>.sb3 liegt in der App **Dateien** unter Downloads.
5. In **Dateien** die Datei lange drücken → **Teilen** → **AirDrop** → iPad der Leitung wählen.
6. **E-Mail und Dateiname** aufschreiben und der Leitung geben.

Öffnen auf jedem Computer: scratch.mit.edu → Entwickeln → Datei → Von deinem Computer hochladen.

Übung 10.2: Dein Spiel vorstellen

Wer will, zeigt sein Spiel, etwa eine Minute. Niemand muss. Du spielst von deinem iPad aus am Beamer und beantwortest:

1. **Was habe ich Neues eingebaut?**
2. **Was war schwer, und wie habe ich es gelöst?**
3. **Was würde ich als Nächstes bauen?**

iPad auf den Beamer (Apple TV): Bühne auf Vollbild. Vom rechten oberen Rand nach unten wischen → **Bildschirmsynchronisation** → Apple TV wählen → Code von der Leinwand eintippen. Danach dort wieder **Synchronisation stoppen**.

Beim Zuhören: Merk dir eine Idee aus einem anderen Spiel.

Zu Hause weitermachen

- **scratch.mit.edu** im Browser, kostenlos.
- Mit einem Erwachsenen ein **eigenes Scratch-Konto** anlegen.
- Deine .sb3-Datei über „Datei → Von deinem Computer hochladen“ öffnen und weiterbauen.
- Studio **Entenpaddeln-Basis** über den Link auf deinem Zettel: Startprojekt und alle Zwischenstände zum Remixen, falls die Datei mal fehlt.
- Ideen: die ★★☆☆-Karten aus Block 9.

Check

- ☐ Ich habe „Jetzt speichern“ getippt.
- ☐ Ich habe die .sb3-Datei heruntergeladen und an den Laptop geschickt.
- ☐ Ich habe mein Spiel vorgestellt, oder ich habe zugeschaut und mir eine Idee gemerkt.
- ☐ Ich weiß, wie ich zu Hause weitermache.

Geschafft. Du hast heute ein Spiel von null auf fertig programmiert: Steuerung, Animation, Klone, Variablen, Nachrichten, Schwierigkeitskurve. Aus diesen Bausteinen bestehen auch große Spiele.