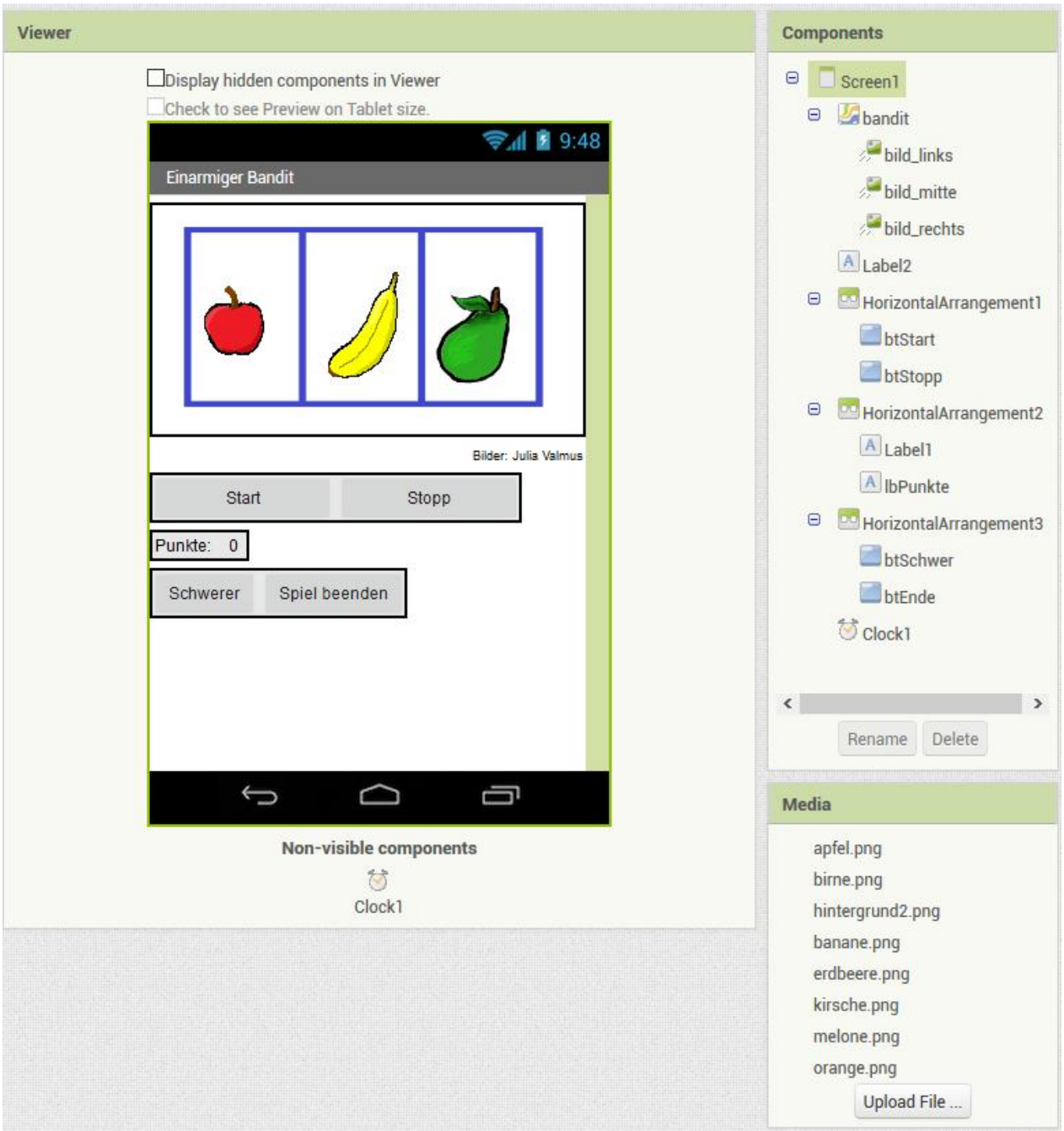
Einarmiger Bandit

Unter dem Begriff „einarmiger Bandit“ versteht man einen Spielautomaten, bei dem man einen Hebel auf einer Seite nach unten drückt (daher „einarmig“). Dadurch drehen sich drei Rollen nebeneinander und können durch einen Knopf gestoppt werden. Wenn drei gleiche Bilder nebeneinander erscheinen, hat man gewonnen. Bei manchen Automaten gibt es auch schon für zwei gleiche Bilder einen Preis.

Im MIT App Inventor könnte der Automat etwa so aussehen:



In jede der drei Bildkomponenten (ImageSprites aus Drawing and Animation) wurde zunächst eines der hochgeladenen Bilder in der Eigenschaft Picture gespeichert.

Beim Start des Programms (Klick auf den Button btStart) sollen die Bilder im Sekundenabstand zufällig wechseln. Wird der Button btStopp geklickt, bleibt das aktuelle Bild sichtbar und es muss ausgewertet werden, ob der Spieler / die Spielerin etwas gewonnen hat. Gestartet wird zunächst mit fünf unterschiedlichen Bildern.

Bei zwei gleichen Bildern soll es 10 Punkte, bei drei gleichen 100 Punkte geben.

Sind alle drei Bilder verschieden können auch Punkte abgezogen werden.

Natürlich sind eurer Phantasie (fast) keine Grenzen gesetzt. Ihr könnt den Schwierigkeitsgrad erhöhen, indem ihr die Zeitphase kürzer macht, die Anzahl der unterschiedlichen Bilder erhöht, etc.

Auftrag für Expertinnen und Experten

1. *Gestalte den Screen wie oben und lade zunächst einmal fünf Bilder hoch. Diese findest du im Tauschordner.*
2. *Überlege dir, was alles im Programm passieren muss und schreibe die Schritte auf.*
3. *Vergleiche deine Schritte mit einer Partnerin / einem Partner nach Wahl und besprecht eure Ideen. Entscheidet dann, wie ihr es versuchen wollt.*
4. *Programmiert euren eigenen „einarmigen Bandit“ Schritt für Schritt gemeinsam. Wechselt euch dazu immer alle zehn Minuten am PC ab.*

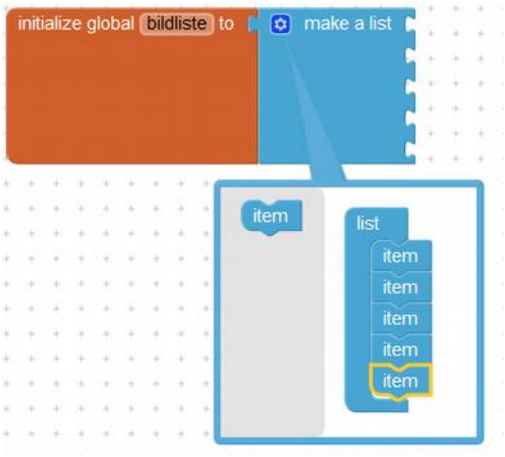
Platz für Ideen, Skizzen, Überlegungen:

Wenn ihr Hilfe braucht... :)

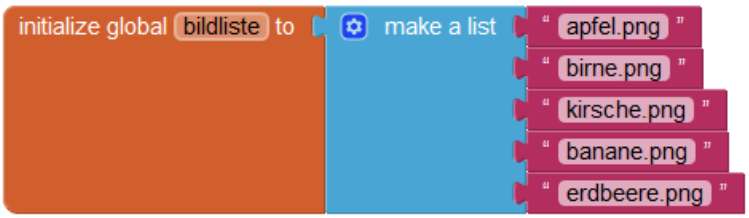
1. Gestalte den Screen wie oben und lade zunächst einmal fünf Bilder hoch. Diese findest du im Tauschordner.

Die Bilder müssen wir irgendwo abrufbar haben. Dazu legen wir eine Liste an, in der wir die Dateinamen ablegen.

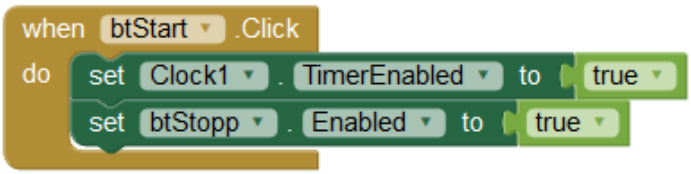
Um mehr als zwei Elemente in die Liste aufnehmen zu können, erweitern wir die Liste zunächst.



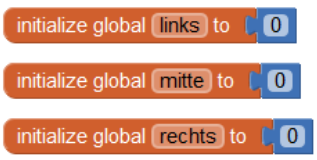
Jetzt können die Dateinamen eingetragen werden.



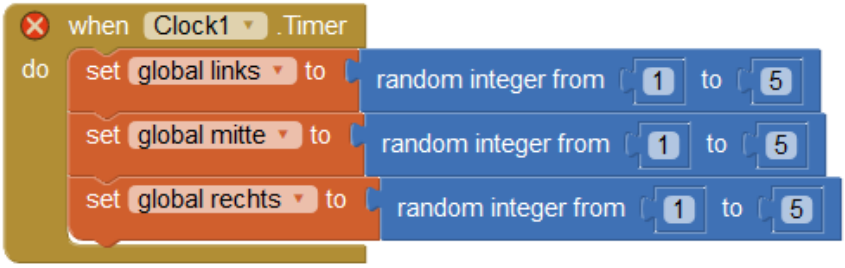
Den Wechsel der Bilder programmieren wir am besten in einer Timerkomponente (Clock aus Sensors). Gestartet wird der Timer beim Klick auf den Start-Button (Die Eigenschaft TimerEnabled muss also vorher im Designer deaktiviert werden). Außerdem soll dann der Stopp-Button aktiviert werden, den wir ja nur brauchen, wenn die Bilder wechseln.



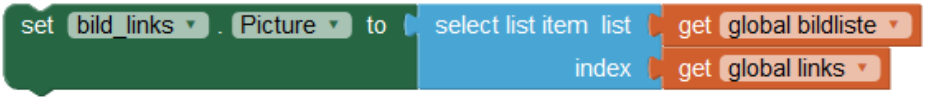
Welches Element der Bildliste gewählt wird, soll eine Zufallszahl bestimmen. Wir nutzen hier drei globale Variable links, mitte und rechts.



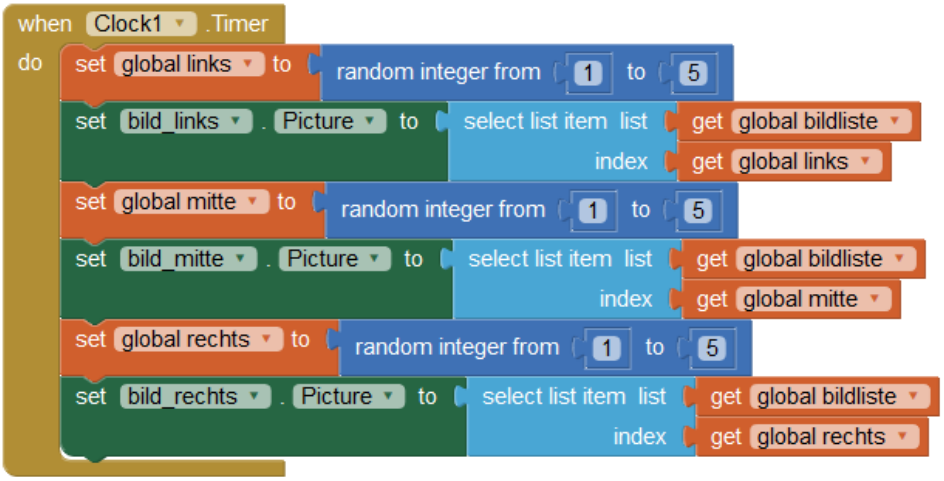
Diese Variablen bekommen dann im Timer-Ereignis die Zufallszahl zugewiesen.



Noch werden die Bilder nicht verändert. Dazu müssen wir den Bildkomponenten noch das richtige Bild übergeben. Für das linke Bild z.B.

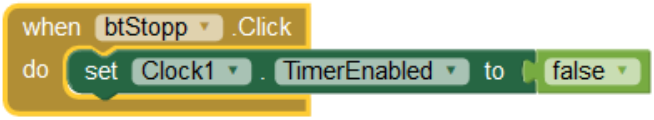


Insgesamt sollte es dann so aussehen:



Jetzt sollte der Bildwechsel schon funktionieren. Nur lässt sich das ganze noch nicht stoppen. Dazu müssen wir uns überlegen, was beim Stoppen passieren soll.

Zunächst muss der Timer gestoppt werden.



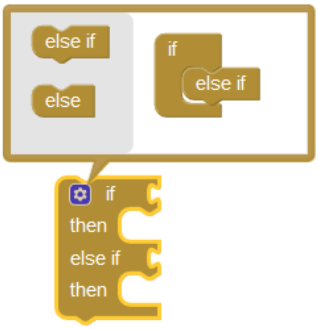
Dann folgt die Auswertung.

Überlege dir in eigenen Worten, was passieren soll!

Vielleicht hast du ungefähr folgendes gedacht: „Falls alle drei Bilder gleich sind, bekommt man 100 Punkte dazu, ansonsten muss ich noch schauen, ob zwei Bilder gleich sind. Sind zwei Bilder gleich, bekommt man 10 Punkte dazu, sonst passiert erst einmal nichts.“

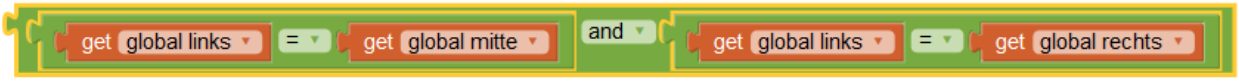
Du weißt bestimmt schon, welche Kontrollstruktur hier gebraucht wird.

Wir nutzen eine Verzweigung, bei der wir zweimal eine Bedingung abfragen können:



Wie lautet die erste Bedingung? Müssen wir jedes Bild mit jedem vergleichen?

Es reicht folgende Bedingung aus. Überlege dir, warum.



Wenn das linke und das mittlere Bild gleich sind **und** das linke und das rechte Bild, dann müssen natürlich auch das mittlere und das rechte Bild gleich sein.

Da hier beide Bedingungen gleichzeitig gelten müssen, wurden sie mit **„and“** verknüpft.

Die zweite Bedingung ist etwas umfangreicher. Schaffst du es trotzdem, sie zu finden?

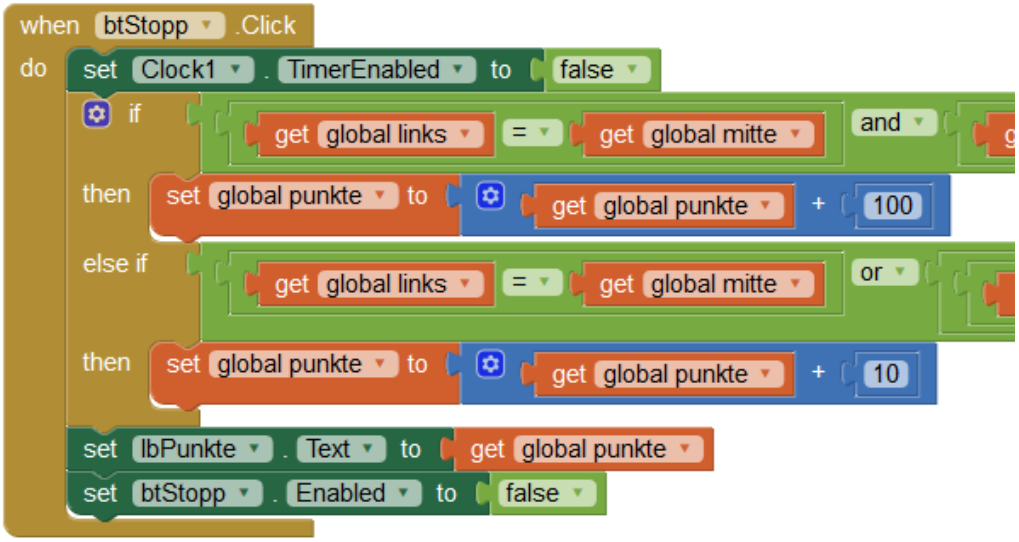
Hier werden die einzelnen Bedingungen mit oder („or“) verknüpft, weil eine schon ausreicht.

Hast du schon eine Variable für die Punkte angelegt? Wenn nicht, lege eine globale Variable punkte an.

Das Ergebnis deiner Verzweigung könnte dann so aussehen:

Was fehlt noch?

Die Punkte müssen angezeigt und der Stopp-Button deaktiviert werden.



Den in der Abbildung fehlenden Code auf der rechten Seite siehst du auf der vorherigen Seite.

Viel Spaß beim Austesten.