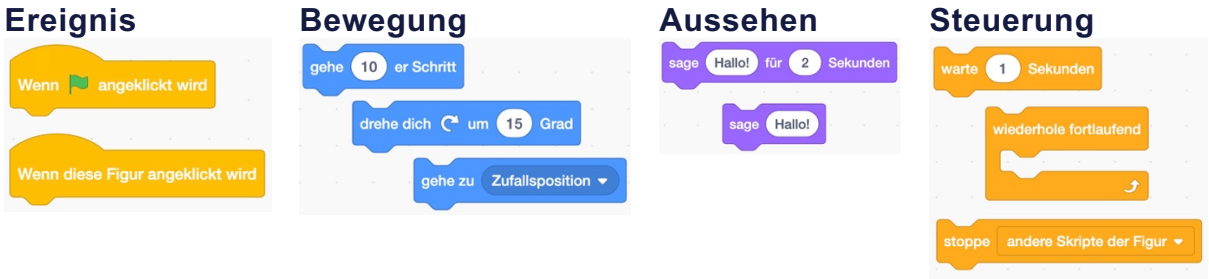
Befehle in Scratch[1]

Um ein Projekt zum Leben zu erwecken, muss man seine Charaktere und Objekte programmieren. Das bedeutet, dass man den Figuren Befehle zuweist und ihnen sagt, was und wann sie etwas machen sollen.

Scratch ist eine visuelle blockbasierte Programmiersprache und daher besonders einstiegsfreundlich. In Scratch verwendet man zum Programmieren also einzelne Blöcke, die zusammengesetzt dann einen bestimmten Code ergeben. Der Vorteil von visuellen, blockbasierten Programmiersprachen liegt darin, dass sie es ermöglichen, rein auf der Ebene der Logik zu arbeiten, ohne sich, im Gegensatz zu textbasierten Sprachen wie Python, C++ oder Java mit Syntaxproblemen (also der richtigen Reihenfolge und Struktur der Anweisungen) auseinandersetzen zu müssen.

Es gibt sehr viele verschiedene Blockarten, die alle verschiedenen Funktionen haben. Für einen besseren Überblick sind die Befehls-Blöcke in neun Kategorien eingeteilt, die jeweils durch eine Farbe gekennzeichnet werden: Bewegung, Aussehen, Klang, Ereignisse, Steuerung, Fühlen, Operatoren, Variablen und Meine Blöcke.

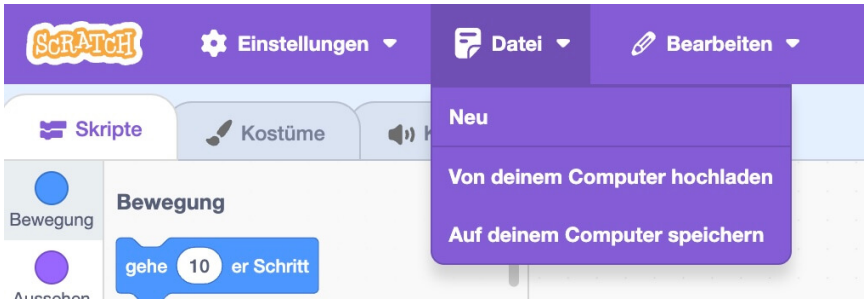
Hier seht ihr beispielhaft einige Befehle und die Kategorien, unter denen sie jeweils zusammengefasst werden.



Projekt speichern

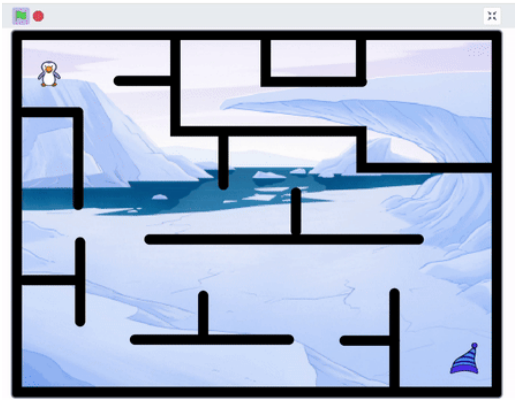
Wenn ihr in einem Scratch-Konto angemeldet seid, werden eure Projekte automatisch gespeichert. Wenn dies nicht geschieht, könnt ihr euer Projekt oben links in der Menüleiste unter **Jetzt speichern** speichern.

Falls ihr kein Konto habt, könnt ihr das Projekt auch auf eurem Computer speichern. Klickt dazu oben links auf die Schaltfläche **Datei** und wählt dann **Auf deinem Computer speichern** aus. Danach könnt ihr auswählen, in welchem Ordner auf eurem Computer das Projekt abgelegt werden soll. Manchmal fragen Browser nicht nach dem Speicherort. In diesem Fall wird die Datei normalerweise im Downloads-Ordner gespeichert.



[1] https://digital.tueftellab.de/mod/page/view.php?id=222&forceview=1

Tutorial Labyrinthspiel in scratch[2]



Das fertige Spiel findest du hier Labyrinthspiel[3]

In diesem Spiel hat das Baby-Pinguinchen Mika seinen Hut verloren. Hilf ihm, seinen Winterhut zu finden.

Wir können dieses Spiel in 3 Hauptkomponenten aufteilen:

1. **Das Labyrinth:** Eine Zeichnung eines Labyrinths, das die Spieler meiden müssen, um hindurchzukommen.
    - **Gestalte das Labyrinth → wir haben eine Vorlage für dich erstellt**
2. **Der Spieler:** Der Spieler, der Pinguin, bewegt sich im Labyrinth und kehrt zum Anfang zurück, wenn er den Rand des Labyrinths berührt.
    - **Füge den Spieler hinzu und positioniere ihn**
    - **Bewege den Spieler im Labyrinth**
    - **Erkenne einen Zusammenstoß des Spielers mit dem Labyrinth**
3. **Das Ziel:** Das Ziel, der Winterhut, ist am Ende des Labyrinths platziert, und der Spieler gewinnt, wenn er das Ziel erreicht.
    - **Füge das Ziel hinzu und positioniere es**
    - **Erkenne den Zusammenstoß des Spielers mit dem Ziel**

Schritt 1: Erstelle ein neues Scratch-Projekt

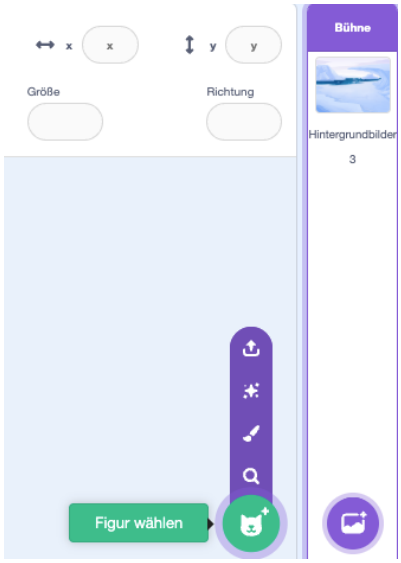
- Besuche scratch.mit.edu
- Klicke auf die Schaltfläche **Datei** und dann auf **Von deinem Computer hochladen**
- Wähle die Datei **labyrinth_vorlage.sb3** vom Schreibtisch aus
- Jetzt solltest du einen Hintergrund und die Labyrinth-Zeichnung sehen
    - Wenn ihr genug Zeit habt, könnt ihr auch probieren ein eigenes Labyrinth zu zeichnen oder den Hintergrund zu ändern

[2] https://www.codewizardshq.com/how-to-make-a-maze-in-scratch-in-7-steps/
[3] https://scratch.mit.edu/projects/1078234216/

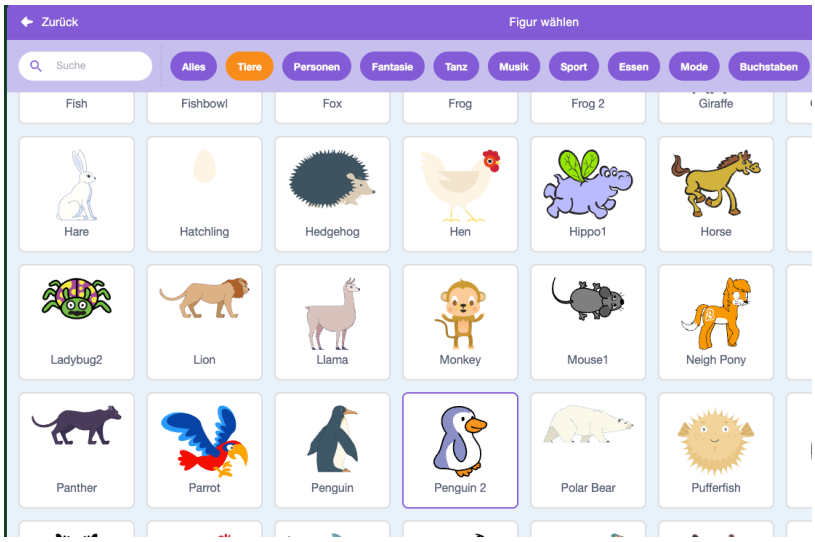
Schritt 2: Füge den Spieler hinzu und positioniere ihn

Wir wollen unseren Spieler, den Pinguin, hinzufügen. Der Pinguin soll jedes Mal oben links starten, wenn das Spiel beginnt. Bevor wir den Pinguin bewegen, positionieren wir ihn oben mit unserem Code.

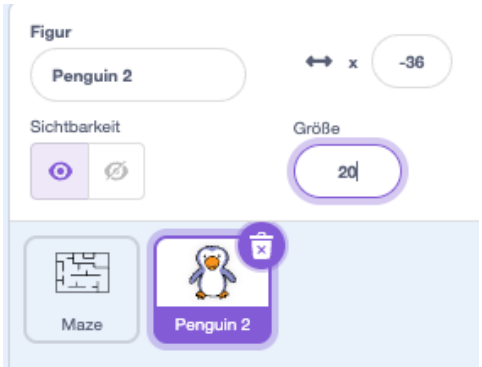
- Klicke auf die Schaltfläche **Figur wählen** in der rechten Ecke



- Wähle die „**Penguin 2**“-Figur aus der Bibliothek

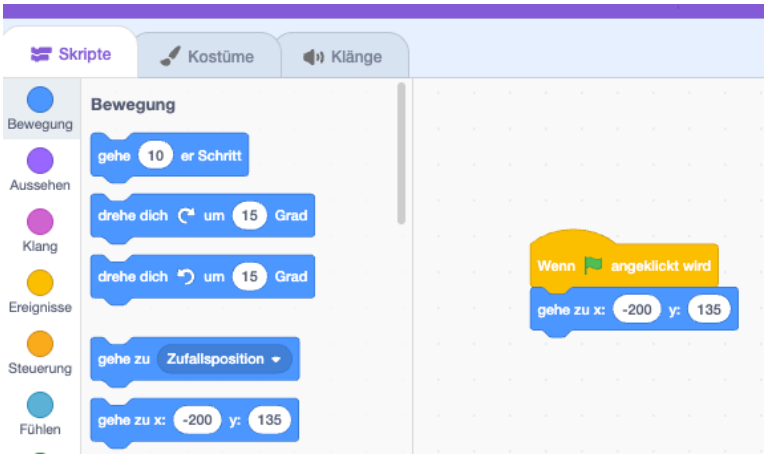


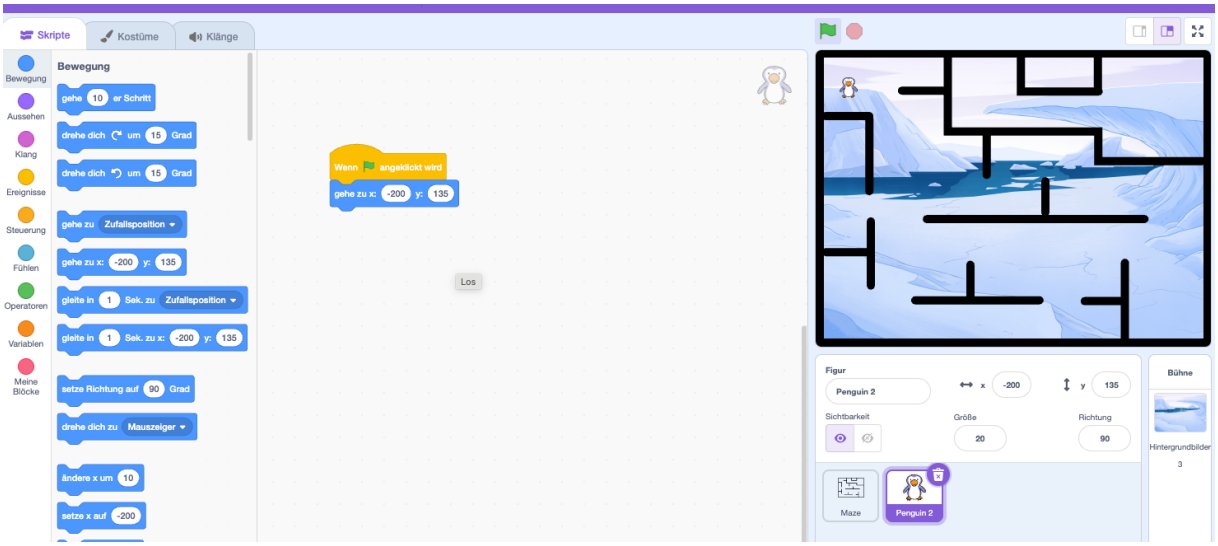
- Ändere die Größe, damit sie in dein Labyrinth passt, mein Pinguin hat die Größe **20**.



Füge dann diesen Code zum Pinguin hinzu, damit er jedes Mal zur Startposition zurückkehrt, wenn das Spiel beginnt. Achte dabei darauf, dass du den Pinguin auch ausgewählt hast. Er ist dann lila markiert:

- Füge einen **Wenn grüne Flagge angeklickt wird**-Block aus der Kategorie **Ereignisse** hinzu.
- Füge einen **Gehe zu x y-Position**-Block aus der Kategorie **Bewegung** hinzu. Verwende die x- und y-Werte deiner Figur an der Startposition.





Jetzt probiere aus, deinen Spieler mit der Maus von der Startposition wegzubewegen. Wenn du die grüne Flagge klickst, sollte er dorthin zurückkehren. Wenn dein Test funktioniert, bist du bereit für den nächsten Schritt.

Hinweis: Du kannst kreativ werden und deine eigene Figur zeichnen, indem du über die Schaltfläche **Figur wählen** fährst und **Malen** wählst.

Schritt 3: Bewege den Spieler im Labyrinth

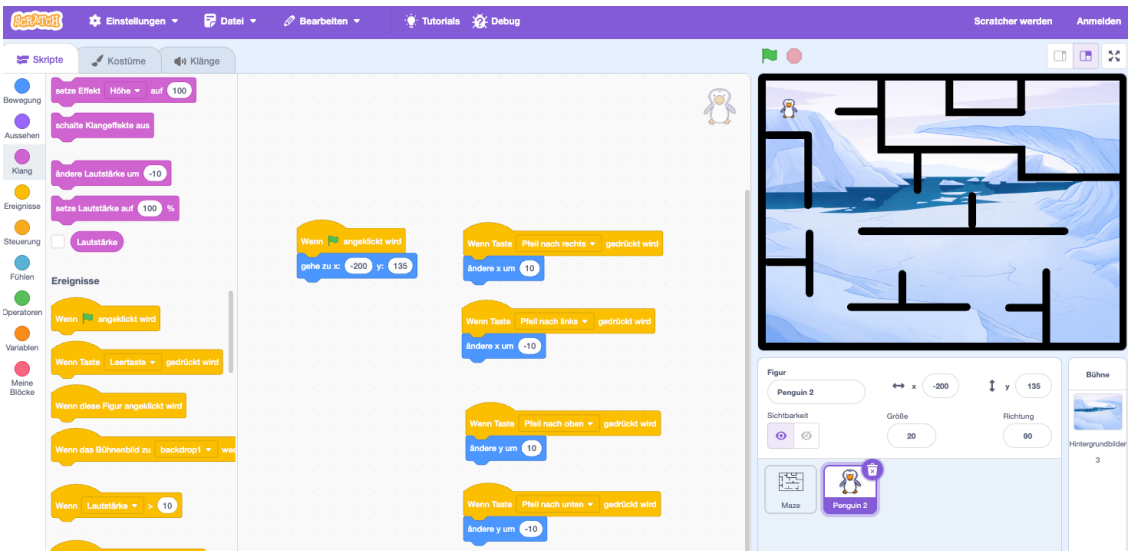
Der Pinguin ist auf dem Bildschirm, aber noch kann er sich nicht bewegen. Wir wollen, dass der Pinguin sich nach oben, unten, rechts und links bewegen, wenn wir die Pfeiltasten drücken.

Lass uns die Logik für die Pfeiltasten hinzufügen. Achte dabei darauf, dass du den Pinguin auch ausgewählt hast - er ist dann lila markiert. Um die Figur nach rechts und links zu bewegen, füge diesen Code hinzu:

- Füge aus der Kategorie **Ereignisse** einen **Wenn Taste Leertaste gedrückt**-Block hinzu.
- Klicke auf **Leertaste** und ändere auf **Pfeil nach rechts**
- Füge einen **Ändere x um**-Block aus der Kategorie **Bewegung** hinzu und ändere den Wert auf **10**
- Klicke mit der rechten Maustaste auf den **Wenn Taste gedrückt**-Block und dupliziere den gesamten Codeblock
- Ändere dann das Dropdown-Menü auf **Pfeil nach links** und ändere **x um** auf **-10**

Um die Figur nach oben und unten zu bewegen, füge diesen Code hinzu:

- Füge aus der Kategorie **Ereignisse** wieder einen **Wenn Leertaste Taste gedrückt**-Block hinzu und ändere die Taste auf **Pfeil nach oben**
- Füge einen **Ändere y um**-Block aus der Kategorie **Bewegung** hinzu und ändere den Wert auf **10**.
- Klicke mit der rechten Maustaste auf den **Wenn Taste gedrückt**-Block und dupliziere den gesamten Codeblock.
- Ändere dann das Dropdown-Menü auf **Pfeil nach unten** und ändere **y um** auf **-10**.



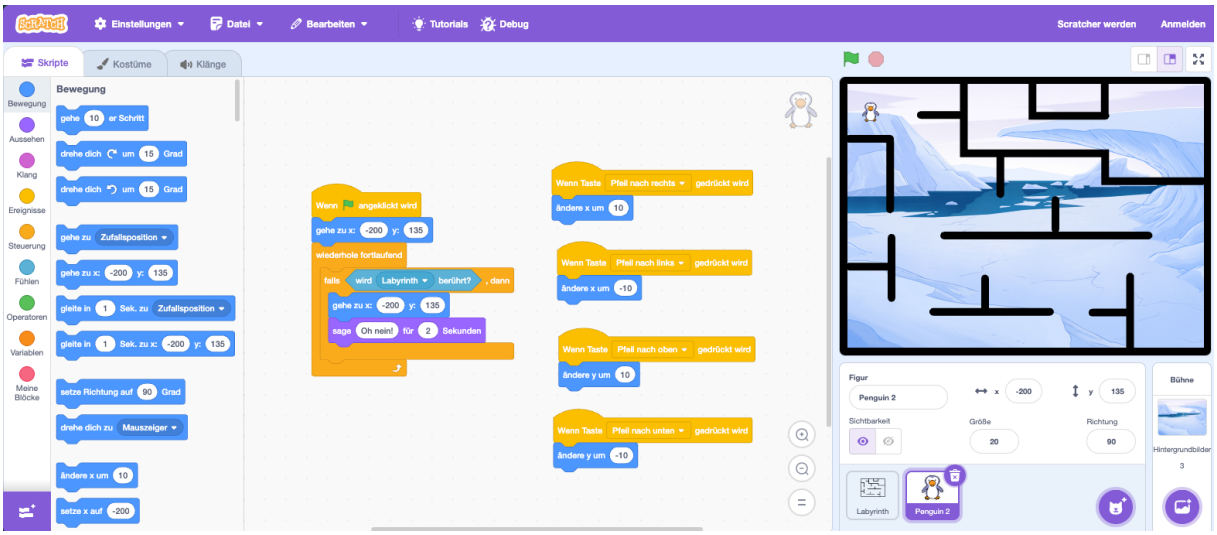
Jetzt kann sich dein Pinguin bewegen! Probiere es aus, indem du die Pfeiltasten drückst und sicherstellst, dass der Pinguin in die richtige Richtung bewegt wird.

Hinweis: Um den Spieler schneller oder langsamer zu bewegen, ändere die x- und y-Werte.

Schritt 4: Erkenne einen Zusammenstoß mit dem Labyrinth

Wir müssen programmieren, was passiert, wenn der Spieler das Labyrinth berührt. Der Pinguin sollte "Oh nein!" sagen und zur Startposition zurückkehren.

- Wähle den Pinguin aus und füge aus der **Steuerung** Kategorie einen **wiederhole fortlaufend**-Block an den Block mit der grünen Flagge hinzu.
- Füge innerhalb des **wiederhole fortlaufend**-Blocks einen **falls, dann**-Block hinzu.
- Füge aus der Kategorie **Fühlen** den **wird Mauszeiger berührt**-Block innerhalb des **falls, dann**-Blocks hinzu. Ändere das Dropdown-Menü auf **Labyrinth**.
- Füge einen **Gehe zu**-Block hinzu und verwende die gleichen x- und y-Positionen wie die Startposition.
- Füge aus der Kategorie **Aussehen** einen **sage Hallo! für 2 Sekunden**-Block hinzu und aktualisiere die Nachricht auf **„Oh nein!“**



Jetzt kannst du versuchen, das Labyrinth mit deinem Spieler zu berühren. Kehrt er zur Startposition zurück?

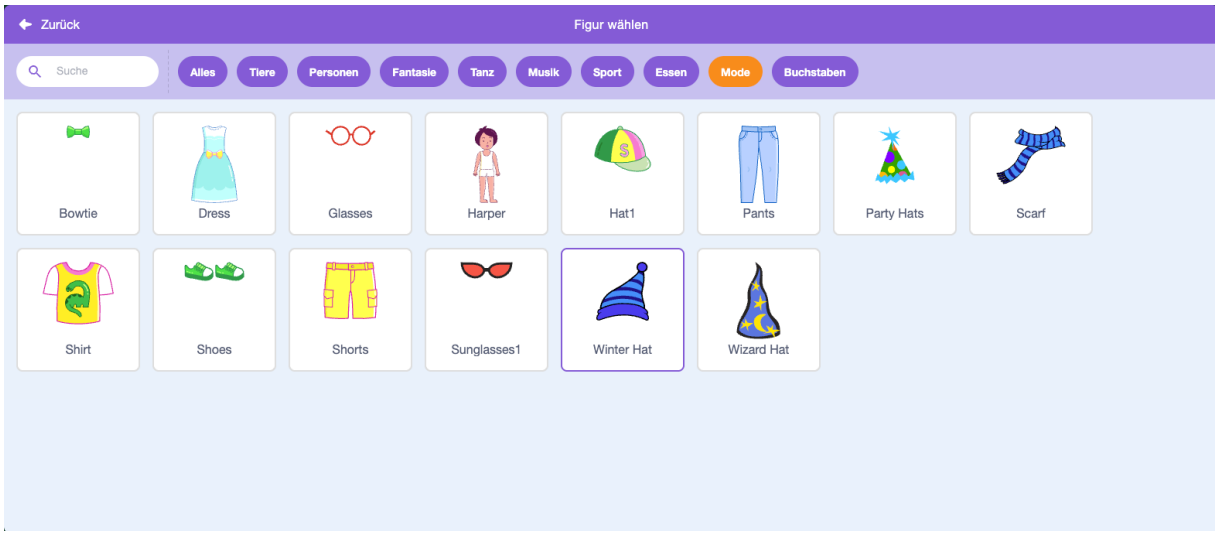
Wenn es funktioniert, können wir weitermachen!

Hinweis: Du kannst die Nachricht, die deine Figur sagt, anpassen oder ändern, was passiert, wenn der Pinguin das Labyrinth berührt hat.

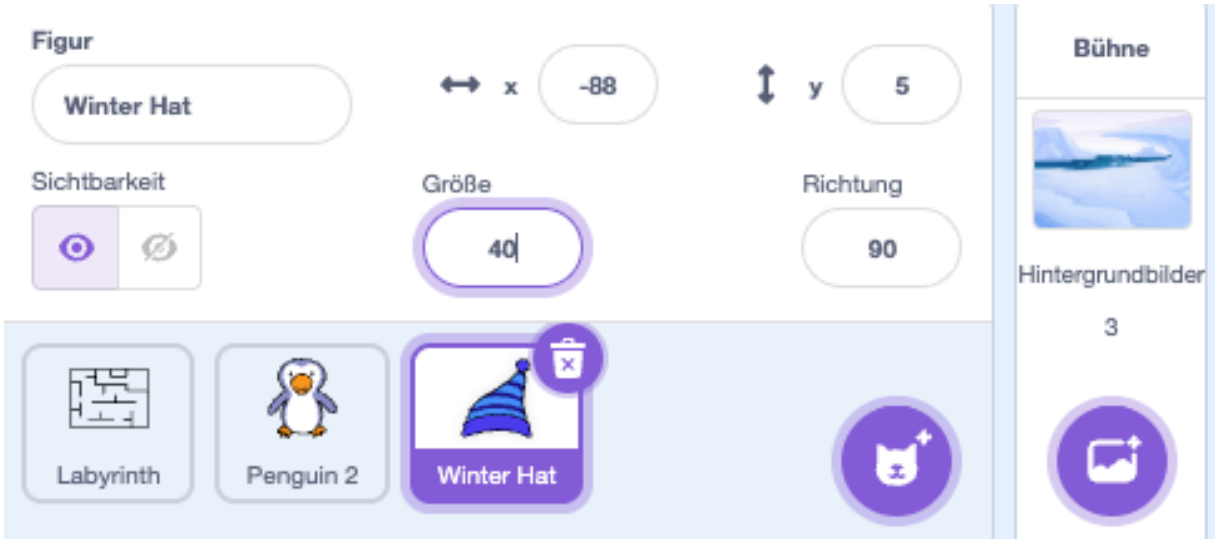
Schritt 5: Füge das Ziel hinzu und positioniere es

Unser Spieler ist platziert und kann sich im Labyrinth bewegen. Jetzt platzieren wir das Ziel am Ende des Labyrinths.

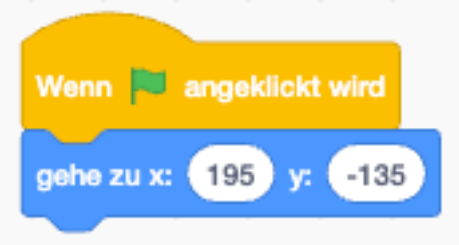
- Klicke auf die Schaltfläche **Wähle Figur** in der rechten Ecke.
- Wähle eine Figur aus der Bibliothek aus. Ich verwende eine aus der Scratch-Bibliothek namens **Winter Hat**.

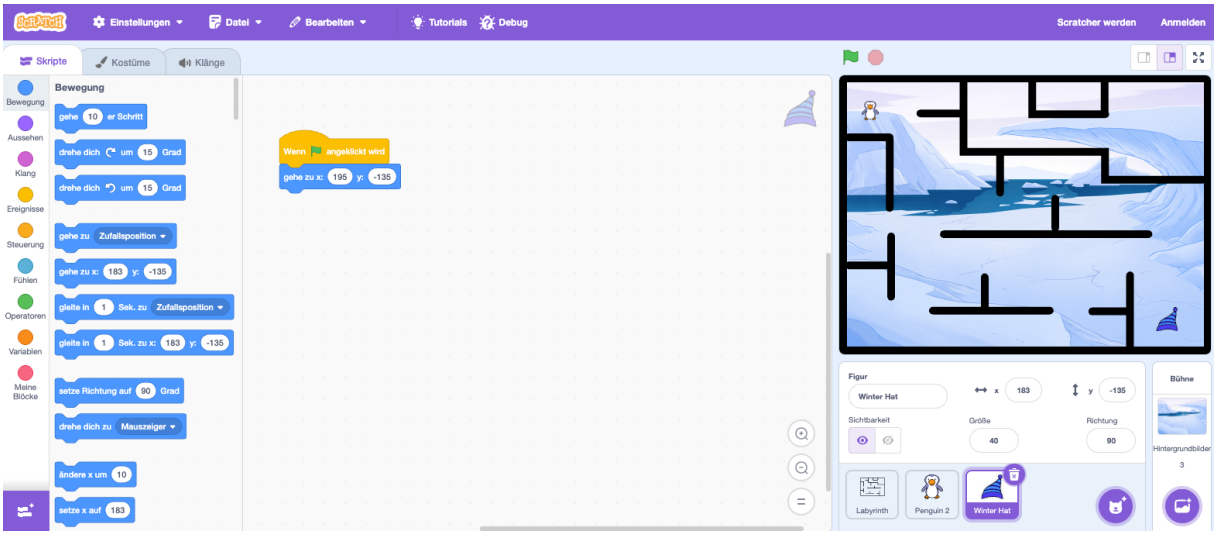


- Ändere die Größe, damit es in dein Labyrinth passt. Meiner hat die Größe 40.



- Stelle sicher, dass du den Hut ausgewählt hast und füge einen **Wenn grüne Flagge angeklickt wird**-Block aus **Ereignissen** hinzu.
- Positioniere den Hut in der Vorschau am Ende des Labyrinths. Füge dann einen **Gehe zu x y-Position**-Block aus **Bewegung** hinzu, der diese Koordinaten verwendet. Meiner ist bei **x = 195** und **y = -135** positioniert.





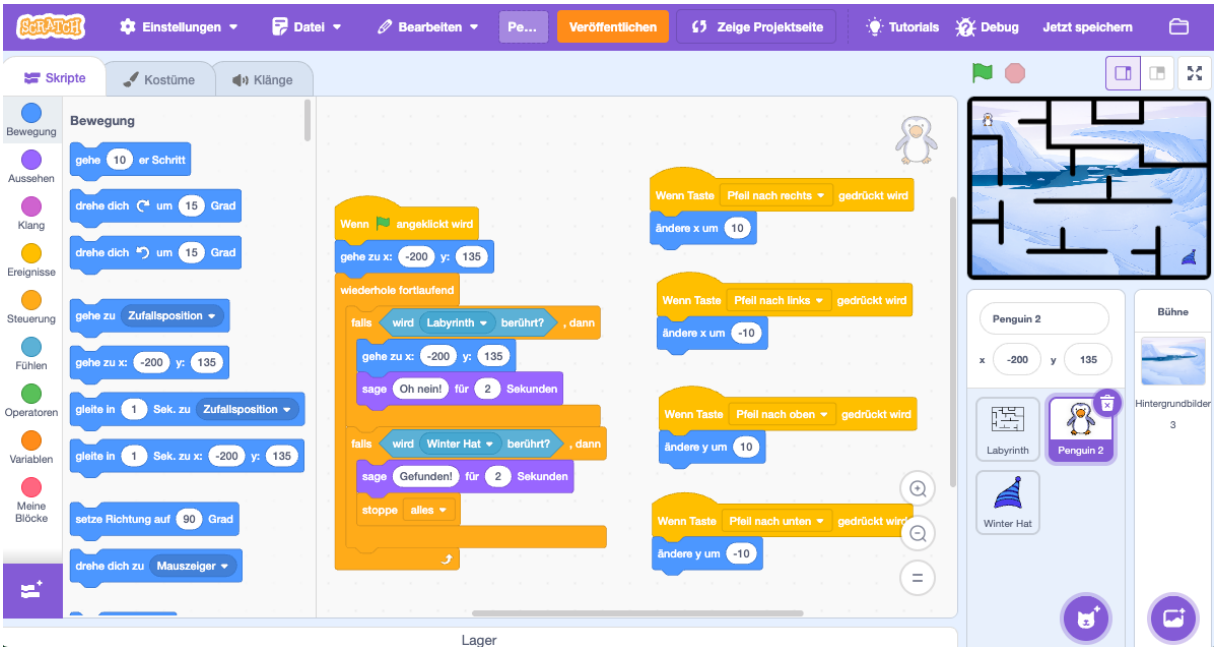
Sobald du fertig bist, klicke auf die grüne Flagge, um dein Spiel zu testen. Dein Hut sollte zum Ende des Labyrinths gehen, wenn die grüne Flagge geklickt wird.

Hinweis: Du kannst die Ziel-Figur anpassen, um zu deinem Thema zu passen, indem du eine eigene Figur malst oder hochlädst.

Schritt 6: Erkenne den Zusammenstoß des Spielers mit dem Ziel

Um das Spiel zu gewinnen, muss der Spieler das Ziel erreichen. Wenn der Spieler das Ziel erreicht, sagt er "**Gefunden!**" und das Spiel endet.

- Wähle die Spieler-Figur aus und füge ihr innerhalb des schon bestehenden **wiederhole fortlaufend**-Blocks eine weitere **falls, dann**-Schleife aus der **Steuerung** Kategorie hinzu.
- Füge aus der Kategorie **Fühlen** den **wird berührt**-Block innerhalb der **falls, dann**-Schleife hinzu. Ändere das Dropdown-Menü auf "**Winter Hat**".
- Füge einen **sage für 2 Sekunden**-Block aus dem **Aussehen**-Block hinzu und aktualisiere die Nachricht auf "**Gefunden!**"
- Füge aus der Kategorie **Steuerung** einen **Stoppe alle**-Block hinzu, um das Spiel zu beenden.



Hinweis: Du musst das Labyrinth nicht vollständig abschließen, um dein Spiel zu testen. Ziehe einfach deine Spieler-Figur ans Ende in deiner Spielvorschau.

Dein Labyrinthspiel ist komplett! Jetzt kannst du das Labyrinthspiel spielen!

Klicke auf die grüne Flagge und probiere es aus.

Wenn du zuhause noch weitere Projekte in scratch ausprobieren willst, schaue doch mal hier vorbei:

https://digital.tueftellab.de/mod/page/view.php?id=143%2F