SCHRITT FÜR SCHRITT ZUR EIGENEN APP

Deine Ideen für eine gute Zukunft

Idee: Deutsche Kinder- und Jugendstiftung (DKJS) https://www.dkjs.de

Texte: Technovation https://www.technovation.org und DKJS

Überarbeitung und Konzeption: KF Education, Dana Tretter, Lisa Krug, Jördis Dörner

Layout: KF Education, Silvana Kuhnert

Illustrationen: https://www.opendoodles.com

Icons: https://khushmeen.gumroad.com/l/doodleicons

Creative Commons

Dieses Handbuch "Schritt für Schritt zur eigenen App – Deine Ideen für eine gute Zukunft" wurde basierend auf dem "Curriculum" von Technovation (Urheberin) weiterentwickelt. Die Inhalte wurden verändert, in dem sie zusammengefasst wurden und neue Passagen dazugeschrieben wurden. Außerdem wurden die Inhalte mit neuen Illustrationen und Grafiken ergänzt, die mit CC0 lizenziert sind.

Das Vorhaben wurde umgesetzt im Rahmen von AUF!leben. AUF!leben – Zukunft ist jetzt. ist ein Programm der Deutschen Kinder- und Jugendstiftung, gefördert vom Bundesministerium für Familie, Senioren, Frauen und Jugend.

Das Programm ist Teil des Aktionsprogramms Aufholen nach Corona der Bundesregierung.

Gefördert vom:

Bundesministerium für Familie, Senioren, Frauen und Jugend

im Rahmen des Aktionsprogramms

AUFHOL PAKET – Das Förderprogramm für Kinder, Jugend & Familien.

deutsche kinder- und jugendstiftung

AUF!leben

SCHRITT FÜR SCHRITT ZUR EIGENEN APP - DEINE IDEEN FÜR EINE GUTE ZUKUNFT

Du möchtest eine eigene App entwickeln und dabei noch was Gutes tun? Dann bist du hier genau richtig! Hier lernst du, wie du deine eigene mobile Anwendung (App) entwerfen und programmieren kannst. Du hast Lust, dabei zu sein? Dann ist das alles, was du brauchst! Vorkenntnisse im Programmieren sind dabei nicht nötig. Das Handbuch hilft dir mit einer schrittweisen Anleitung, wie du von der Idee zur fertigen App kommst.

Steckbrief

Herausgeber:innen: Dieses Handbuch ist ein Produkt der Deutschen Kinder- und Jugendstiftung (DKJS). Es ist aus dem Programm Technovation Girls Germany entstanden.

Zielgruppe: Das Handbuch richtet sich vor allem an junge Menschen im Alter von zehn bis 18 Jahren und diejenigen, die sie dabei begleiten. Aber auch alle anderen, die Lust haben, Apps für einen guten Zweck zu entwickeln.

Ziel: Du findest mithilfe einer selbstentwickelten App Lösungen für soziale und ökologische Herausforderungen in deiner Lebenswelt. Dabei erlangst du digitale sowie unternehmerische Kompetenzen. Das Handbuch stärkt dich auf deinem Weg in eine digitale Zukunft und in eine gerechtere Welt. Ziel dieses Handbuchs ist es, dass du am Ende ein erstes funktionsfähiges Produkt deiner App entwickelt hast.

AUFBAU UND TIPPS

Das Handbuch ist in sechs Kapitel aufgeteilt. Es enthält neben hilfreichen Informationen, Anleitungen (Englisch: Tutorials) viele praktische Aufgaben (Aktivitäten), die du direkt im Handbuch bearbeiten kannst. Es unterstützt dich dabei, eine Idee zu entwickeln und sie zu verwirklichen.

- Zu Beginn des Handbuchs findest du die **LEARNING MAP**. Die Learning Map funktioniert wie ein Inhaltsverzeichnis und gibt dir durch Farben, Linien und Symbole (Icons) eine Führung durch das Handbuch. Durch Klicks auf die Lektionen oder Icons wirst du über eine Verlinkung automatisch an die richtige Stelle geleitet.

- Nach der Learning Map findest du das **PROJEKT CANVAS**. Das ist das zentrale Element für die Entwicklung und Umsetzung deiner App. Es ist ein Arbeitsblatt, das dir dabei hilft, den Überblick zu behalten. Hier kannst du den Teamnamen, Ziele eures Projekts und andere wichtige Dinge eintragen, die eure eigene App ausmachen.

- Danach gelangst du zum Kapitel **PROJEKTSTART** und der Lektion **Erste Schritte – Ein Team finden**. Es ist natürlich auch möglich alleine zu arbeiten, du solltest dann nur etwas mehr Zeit einplanen. Suche dir Menschen aus deinem Umfeld, die dich unterstützen. Das können Freund:innen sein, die mit dir ein Team bilden oder auch Erwachsene, die dir helfen. So könnt ihr die verschiedenen Aufgaben gut auf mehrere Schultern und je nach Interesse verteilen.

- Im Kapitel **IDEENFINDUNG** lernst du, wie du Probleme in deinem Umfeld findest und Schritt für Schritt Lösungen dazu entwickelst. Dazu gehören auch die Planung und die Organisation deines Projektes.

- Im Kapitel **UNTERNEHMERTUM** lernst du verschiedene Unternehmensformen kennen, entwickelst einen Namen und gestaltest ein Logo für euer Projekt. Zudem lernst du, wie du deine Idee und dein Produkt vorstellen und effektiv bewerben kannst. Du erfährst alles zum Thema Videoaufnahme und -schnitt und bekommst wichtige Tipps.

- Im Kapitel **CODING** (Englisch für Programmieren) lernst du Werkzeuge (Englisch: Tools) für die App-Entwicklung kennen. Du steigst schrittweise in das Programmieren ein und entwickelst deine mobile Anwendung.

- Im Zusatzkapitel **KÜNSTLICHE INTELLIGENZ (KI)** erhältst du eine Einführung in das Thema KI. Du erfährst tolle Möglichkeiten, wie du diese Technologie in einer App nutzen kannst, um soziale Probleme zu lösen. Zugleich liefert das Kapitel einen Einblick, wo heute schon überall KI drin ist und wie man gut mit ihr arbeiten kann.

Die Kapitel **PROJEKTSTART**, **IDEENFINDUNG**, **UNTERNEHMERTUM**, **CODING**, **PROJEKTZIEL** sowie das Zusatzkapitel **KÜNSTLICHE INTELLIGENZ** sind in verschiedene Lektionen unterteilt, die dieselbe Struktur haben:

In dieser Lektion ...

Schlüsselbegriffe

In jeder Lektion findest du links eine kurze Aufzählung der **Lernziele** und die wichtigsten Begriffe.

Die wichtigsten Begriffe der jeweiligen Lektion (Schlüsselbegriffe) sind in ein **Glossar** verlinkt. Solltest du sie nicht kennen, kannst du dort nachschauen, was sie bedeuten.

Zu Beginn werden die wichtigen **Fakten** und Inhalte der jeweiligen Lektionen erklärt.

In der **Aktivität** kannst du die Inhalte der Lektionen gleich praktisch anwenden. Die Aktivität besteht aus Arbeitsaufträgen, die dir bei der Umsetzung deines Projekts helfen. Die **Reflexion** am Ende jeder Lektion hilft dir noch einmal über die Lektion und das Gelernte nachzudenken und offene Fragen zu klären.

5 Tipps zum Start

1. Rechne mit ungefähr 40 Stunden, um mit dem Handbuch zu arbeiten und dein Projekt abzuschließen. Auch wenn das erst einmal nach einem hohen Zeitaufwand klingt, lohnt es sich sehr. Du wirst nicht nur viel Neues lernen und Spaß haben, sondern mit deiner Idee auch noch eine Lösung für ein wichtiges Problem finden!
2. Du kannst auch einzelne Kapitel aus dem Handbuch bearbeiten, wenn du nur über bestimmte Themen mehr erfahren möchtest. Es ist nicht zwingend notwendig, dass du das Handbuch von vorne bis hinten durcharbeitest.
3. Plane dein Projekt gründlich, wenn du im Team arbeitest: Trefft klare und verbindliche Absprachen miteinander.
4. Suche dir Hilfe bei Expert:innen, die du bei Fragen ansprechen kannst.
5. Verliere nicht den Mut, wenn es mal stockt oder nicht so läuft, wie du es dir vorgestellt hast. Das ist vollkommen normal und gehört zum Prozess dazu.

Hinweis: Du wirst in diesem Handbuch in der Mehrzahl angesprochen, weil du Teil eines Teams bist. Es ist aber auch okay, wenn du das Handbuch alleine nutzt.

Wir wünschen euch viel Spaß bei eurem Projekt!

EURE LEARNING MAP

Die Learning Map leitet euch durch das Handbuch und euer Projekt. Die verschiedenen Farben der Kapitel zeigen euch, wo ihr gerade seid und was zu den Kapiteln gehört. Alle Inhalte, die **EUER EIGENES PROJEKT**, euer Team und somit eure App betreffen sind pink dargestellt. Die weiteren Kapitel (und ihre Lektionen): **IDEENFINDUNG**, **CODING**, **UNTERNEHMERTUM** und **KÜNSTLICHE INTELLIGENZ** vermitteln euch alle wichtigen Fakten und Inhalte, um euer Projekt Schritt für Schritt von der Idee bis zur fertigen App umzusetzen.

EUER PROJEKT CANVAS gibt euch einen Überblick über euer Projekt. Wenn ihr mit dem Handbuch arbeitet, füllt ihr nach und nach die einzelnen Bausteine aus. Überall, wo ihr etwas im Projekt Canvas ausfüllen könnt, ist das mit diesem Symbol bzw. Icon gekennzeichnet:

Durch Anklicken des Icons kommt ihr jederzeit zum Projekt Canvas.

Ein Meilenstein gibt an, wenn ihr wichtige Schritte zu eurer eigenen App gemeistert habt. Wenn ihr eine Lektion fertig bearbeitet habt oder einen Meilenstein erreicht habt, könnt ihr die Checkbox daneben abhaken. So seht ihr, wo im Handbuch ihr gerade seid und was ihr schon geschafft habt. Wenn ihr auf einen Begriff in der Learning Map klickt, springt ihr in die jeweilige Lektion.

Hinweis: Damit ihr am Ende des Projekts ein erstes funktionsfähiges Produkt – nämlich eure eigene App - entwickelt habt, solltet ihr zunächst mit dem **PROJEKTSTART** und der **IDEENFINDUNG** beginnen. Bei den Kapiteln **UNTERNEHMERTUM** und **CODING** (Englisch für programmieren) ist die Reihenfolge nicht wichtig. Wollt ihr erst mit dem Programmieren beginnen, könnt ihr Unternehmertum auch danach bearbeiten und andersherum. Das Kapitel Coding enthält den Abschnitt “Fortgeschritten”. Diese Lektionen sind nicht für alle Apps wichtig und ihr braucht sie nicht unbedingt, um am Ende eine erste App zu programmieren. Für manche Apps und Funktionen können sie aber wichtig sein. Entscheidet hier selbst, wie viel ihr wissen müsst.

Legende

- Meilenstein
- Checkbox
- Link zum Projekt Canvas

EUER PROJEKT CANVAS

Auf diesen Seiten findet ihr „Euer Projekt Canvas“. Es ist euer Arbeitsblatt für die Entwicklung und Umsetzung eurer App. Ein Projekt Canvas ist ein Werkzeug, welches alle wichtigen Bausteine eures Projekts enthält und euch dabei hilft, den Überblick zu behalten.

Teamname: Wie heißt euer Team?

Lektion „Erste Schritte - Ein Team finden“

Teammitglieder: Wer ist in eurem Team?

Lektion „Erste Schritte - Ein Team finden“

Problem: Welches Problem soll eure App lösen?

Lektion „Probleme brainstormen und auswählen“

Lösung: Welche Lösung bietet eure App?

Lektion „Lösung verstehen und auswählen“

Unsere Community: Welches Umfeld habt ihr gewählt?

Lektion „Euer Umfeld verstehen“

Unsere Zielgruppe: Wer werden die Nutzer:innen eurer App sein?

Lektion „Marktforschung durchführen“

App-Funktionen: Was sind die wichtigsten Funktionen?

Lektion „Entwurf eurer App erstellen“

Unternehmensform: Welche Unternehmensform wollt ihr gründen?

Lektion „Unternehmensarten kennenlernen und Mission schreiben“

App-Name: Wie soll eure Anwendung heißen?

Lektion „Unternehmensarten kennenlernen und Mission schreiben“

Mission: Wie lautet eure Mission?

Lektion „Unternehmensarten kennenlernen und Mission schreiben“

PROJEKTSTART

ERSTE SCHRITTE – EIN TEAM FINDEN

In dieser Lektion ...

- lernt ihr, wie ihr dein Team zusammenstellt (falls ihr noch keins habt)
- legt ihr einen Plan für euer Projekt fest

Schlüsselbegriffe

- Team
- Zeitplan

Zusammen mit anderen ein Projekt umzusetzen, macht nicht nur Spaß, sondern ihr könnt zusammen auch kreativer sein. Im Team verfolgt ihr gemeinsam das gleiche Ziel und könnt eure unterschiedlichen Fähigkeiten und Ideen bündeln.

Hier sind einige Möglichkeiten, wie ihr andere für euer Team gewinnen könnt:

- Organisiert eine Infoveranstaltung oder eine AG für interessierte Schüler:innen nach der Schule.
- Erzählt euren Freund:innen von eurer Idee und zeigt ihnen dieses Handbuch.
- Startet einen Aufruf auf Social Media, in dem ihr eure Idee vorstellt und nach Mitstreiter:innen sucht.

Hinweis: Achtet sorgfältig darauf, wer sich bei euch meldet und trefft euch nie alleine mit Leuten, die ihr noch nie in echt gesehen habt. Bezieht dann immer erwachsene Vertrauenspersonen ein, also eure Eltern oder Lehrkräfte.

Sobald ihr ein Team habt, solltet ihr euch überlegen, wie ihr miteinander kommunizieren wollt. Dies kann zum Beispiel über eine Messenger-Gruppe geschehen. Ihr solltet auch regelmäßige Treffen vereinbaren, an denen ihr gemeinsam an eurem Projekt und mit diesem Handbuch arbeitet. Falls ihr euch nicht vor Ort treffen könnt oder möchtet, könnt ihr euch zum Beispiel auch zu einer Videokonferenz verabreden. Wenn ihr euch online Trefft, empfehlen wir euch, mit einem digitalenWhiteboard (z. B. IdeaBoardz, Conceptboard oder Miro) zu arbeiten und so eure Notizen und Ergebnisse gemeinsam festzuhalten.

Tipp // Zusammenarbeit im Team

Trefft klare Absprachen zur Kommunikation und Organisation eurer Teamtreffen, um Verbindlichkeit zu schaffen und euch nicht zu verzetteln. In diesem Video geben wir dir 7 Tipps für die Zusammenarbeit im Team:

VIDEO

Tipp // A-Team-Übung

Wenn ihr als Team erst zusammenfinden müsst, da ihr euch noch nicht gut oder vielleicht gar nicht kennt, hilft euch die A-Team Übung. Damit könnt damit herausfinden, was eure Stärken sind und wie ihr gut zusammenarbeiten könnt.

VIDEO

Aktivität: Ein Team zusammenstellen

Ihr braucht:

- Stifte
- Euer Projekt Canvas

Wenn du ein Team gefunden hast, macht euch gemeinsam über folgende Schritte Gedanken. Nutzt den Platz unten für Notizen oder wichtige Punkte, die ihr nicht vergessen wollt. Wenn ihr euch einig geworden seid, füllt ihr die Bausteine Teamname und Teammitglieder in „Eurem Projekt Canvas“ aus.

1. Überlegt euch einen Namen für euer Team.
2. Jedes Teammitglied erhält einen Aufgabenbereich, den er oder sie betreut und im Auge behält. Das bedeutet nicht, dass ihr nicht gemeinsam an den Aufgaben arbeiten sollt. Es hilft euch aber dabei, den Überblick zu behalten. Entscheidet bei der Verteilung der Aufgabenbereiche gemeinsam nach euren Fähigkeiten und Interessen. Die Bereiche könnten sein: Coding, Design, Marketing etc.
3. Vereinbart anschließend, wie ihr miteinander kommunizieren wollt und in welchen regelmäßigen Abständen ihr zusammenkommen wollt.
4. Zum Schluss solltet ihr euch einen Zeitplan mit allen wichtigen Arbeitsschritten für euer Projekt festlegen.

☞ **Hinweis:** Da sich euer Zeitplan im Laufe des Projekts immer wieder ändern wird, während sich eure Idee entwickelt, solltet ihr ihn regelmäßig bei euren Treffen anpassen. Bedenkt dabei folgende Punkte:

- Setzt euch Termine und Erinnerungen in euren Kalendern. Was sind die wichtigsten Schritte zum Abschluss eures Projekts? Bis zu welchem Datum sollten sie erledigt sein?
- Teilt die Aufgaben auf und arbeitet zum Beispiel paarweise zusammen.
- Setzt euch Prioritäten, das heißt, entscheidet, was am wichtigsten ist und was ihr zuerst machen wollt.

Reflexion

Seid ihr bereit? Überlegt euch, ob ihr noch andere Dinge vorbereiten müsst: Habt ihr alle Materialien, die ihr braucht? Wo wird sich euer Team treffen? Gibt es Expert:innen, die ihr um Rat fragen könnt und die euch bei der Umsetzung eures Projekts unterstützen? Dies können zum Beispiel eine Lehrkraft oder auch Eltern sein.

IDEENFINDUNG

EUER UMFELD VERSTEHEN

In dieser Lektion ...

- findet ihr heraus, was eine Community ist und welchen verschiedenen Communities ihr angehört
- wählt ihr eine Community aus, der ihr helfen möchtet

Schlüsselbegriffe

- Gemeinschaft (Englisch: Community)
- Brainstorming

VIDEO

Als Einstieg in die Ideenfindung ist es wichtig, euer Umfeld und dessen Probleme zu verstehen. Im folgenden Video geben wir euch Tipps, wie ihr Probleme und Lösungen identifizieren könnt.

Ihr solltet euch überlegen, welcher Gruppe von Menschen ihr mit eurer Idee helfen wollt. Dabei hilft es, euch anzuschauen, welche verschiedenen Communities es gibt.

Wir alle gehören zu Gruppen, wie beispielsweise unseren Mitschüler:innen oder Sportteams. Viele dieser Gruppen kann man auch als Gemeinschaften oder Communities bezeichnen. Communities könnt ihr euch auf viele verschiedene Arten vorstellen: Es können Menschen sein, die am selben Ort leben oder eine Gruppe, die ähnliche Interessen hat.

Aktivität: Community Brainstorm

Ihr braucht:

- Stifte
- Euer Projekt Canvas

Um eine Idee für eure App zu finden, müsst ihr die Bedürfnisse eurer Community kennenlernen. Oft nehmen Menschen Probleme in ihren Communities einfach hin und suchen gar nicht nach Lösungen. Eure erste Aufgabe ist es also, die Welt um euch herum und eure Community genau unter die Lupe zu nehmen. Haltet eure Beobachtungen fest (schriftlich, mit Fotos oder als Tonaufnahme), damit ihr später (z. B. bei der Produktpräsentation) erklären könnt, wie ihr zu eurer Lösung gekommen seid und warum sie auf die Bedürfnisse eurer Community wirkungsvoll eingeht.

Aber zuerst: Wie gut kennt ihr eure Community? Manchmal ist man schon lange Teil einer Gruppe, ohne sich darüber bewusst zu sein. Diese Aktivität hilft euch dabei, euch bewusst zu machen: Welchen Communities gehört ihr an? Und mit welchen Problemen können sie konfrontiert sein?

1 a) Notiert euch mindestens vier verschiedene Communities, zu denen ihr gehört oder denen ihr helfen möchtet. Tragt sie in der Tabelle ein.

A.	B.
C.	D.
E.	F.

b) Sucht euch zwei davon aus und notiert euch zusätzlich die Merkmale: Alter, Sprache, Kulturkreis oder Traditionen, Geografische Lage, Interessen.

Wie sieht eure Community aus?	Community 1	Community 2
Alter		
Sprache		
Kulturen oder Traditionen		
Geografische Lage		
Interessen		

c) Entscheidet euch für eine Community und tragt sie ein.

---

2) Tragt nun die ausgewählte Community in „Euer Projekt Canvas“ ein. Aus dieser Community wählt ihr im Laufe eures Projekts eine bestimmte Gruppe an Menschen als eure Zielgruppe aus.

Reflexion

Blickt noch einmal auf die Beobachtungen eurer Community zurück. Was hat euch überrascht? Wo habt ihr vielleicht ganz falsche Annahmen über die Menschen und ihre Probleme gehabt?

PROBLEME BRAINSTORMEN UND AUSWÄHLEN

In dieser Lektion ...

- lernt ihr verschiedene Arten von Problemen kennen
- überlegt ihr euch Probleme, die ihr in eurer Community lösen möchtet
- Prüft ihr die Dimension eurer gesammelten Probleme
- wählt ihr im Team drei Probleme aus, die ihr wichtig findet

Schlüsselbegriffe

- Ziele für nachhaltige Entwicklung (SDGs)
- Dimension

Vor dem Brainstorming von Problemen ist es hilfreich, über Problem-Kategorien nachzudenken. Die „Ziele für nachhaltige Entwicklung" der Vereinten Nationen (Sustainable Development Goals – SDGs) sind Beispiele für solche Kategorien. Fast alle Länder der Welt sind sich einig, dass diese Probleme sehr wichtig sind und gelöst werden müssen:

- Grundlegende menschliche Bedürfnisse und Rechte: Wasser, Nahrung, Schlaf, Kleidung, Unterkunft, Bildung
- Umwelt: Klimaschutz, Zugang zu sauberem Wasser, saubere Energie
- Sicherheitsbedürfnisse: Gesundheit, Wohlbefinden, Sicherheit vor Unfällen und Krankheiten
- Soziale Bedürfnisse: Freundschaften, Familie, Akzeptanz, Respekt, Produktivität
- Zusätzliche SDGs, die gut für das individuelle Handeln sind: Gleichheit, Frieden und Gerechtigkeit

VIDEO

In diesem Video könnt ihr noch mehr über die SDGs erfahren. Vielleicht stellt ihr fest, dass manche Probleme größer und wichtiger sind als andere, das nennt man Problem-Dimension, also das Ausmaß von Problemen.

Wenn ihr ein Problem auswählt, solltet ihr darauf achten, dass es wichtig und sinnvoll für eure Community ist. Stellt dafür bei jedem Problem folgende Fragen:

- Handelt es sich um ein wirklich großes Problem?
- Betrifft das Problem viele Menschen?
- Gibt es Stiftungen, Non-Profit-Organisationen, Unternehmen oder Start-ups, die sich für das Problem interessieren?
- War es in Deutschland in den Nachrichten?

Aktivität: Probleme brainstormen und auswählen

Ihr braucht:

- Stifte

Bei dieser Übung sammelt ihr im Team Probleme. Bei der Recherche nach Problemen können euch folgende Kategorien helfen.

Grundlegende Menschenrechte und Bedürfnisse
(z. B. Wasser, Nahrung, Schlaf, Bildung)

Umwelt
(z. B. Klimaschutz, sauberes Trinkwasser)

SDGs für individuelles Handeln
(z. B. Gleichheit, Frieden und Gerechtigkeit)

Sicherheitsbedürfnisse
(z. B. Gesundheit, Unfallschutz)

Soziale Bedürfnisse
(z. B. Freund:innen, Familie, Respekt)

Teil 1: Probleme finden

1 a) Ordnet zunächst diese Probleme den Kategorien zu, indem ihr das zutreffende Icon ankreuzt oder markiert. Manche Probleme können auch mehreren Kategorien zugeordnet werden.

Es ist schwer, Einwegplastik in einem Supermarkt zu vermeiden.	Es gibt in Deutschland Menschen, deren Muttersprache nicht deutsch ist, und die deshalb in der Schule Nachteile haben.	Es ist schwer, für jedes Gespräch zwischen Personen, die verschiedene Sprachen sprechen, eine:n Dolmetscher:in zu finden.	Menschen mit einer Wirbelsäulenverletzung verlieren viele Fähigkeiten.
Problemkategorie:	Problemkategorie:	Problemkategorie:	Problemkategorie:
Zu Fuß komme ich oft nicht schnell genug ans Ziel.	Kopfschmerzen tun weh.	Menschen mit Blindheit können keine Autos fahren.	Es gibt wenig bezahlbaren Wohnraum (in der Stadt).
Problemkategorie:	Problemkategorie:	Problemkategorie:	Problemkategorie:

Nicht jedes Kind kann sich eine:n persönliche:n Trainer:in für eine Sportart leisten.	Der globale Fleischkonsum wächst und sorgt ist u. a. schädlich für unser Klima.	Es gibt weniger Frauen als Männer in der Politik.	Journalist:innen sind oft Hassrede, also z. B. üblen Beschimpfungen, im Internet ausgesetzt.
Problemkategorie: 	Problemkategorie: 	Problemkategorie: 	Problemkategorie: 

b) Macht nun ein Brainstorm im Team und überlgt euch weitere Probleme. Ordnet sie dann den Kategorien zu.

Tipp // Ein gutes Brainstorming

- Achtet darauf, alle Ideen zu sammeln – auch abwegige und übertriebene.
- Vermeidet jegliche Beurteilung – sowohl von den Ideen anderer als auch von den eigenen.
- Baut gegenseitig auf euren Ideen auf und entwickelt sie weiter.
- Arbeitet gerne visuell – anstatt Worte aufzuschreiben, könnt ihr auch zeichnen.
- Unterbrecht einander nicht – ein Beitrag nach dem anderen.
- Versucht erst einmal so viele Ideen wie möglich zu finden – die Auswahl kommt später.
- Bleibt konzentriert – bleibt beim Thema.

Ein weiteres Problem:	Ein weiteres Problem:	Ein weiteres Problem:	Ein weiteres Problem:
Problemkategorie: 	Problemkategorie: 	Problemkategorie: 	Problemkategorie: 
Ein weiteres Problem:	Ein weiteres Problem:	Ein weiteres Problem:	Ein weiteres Problem:
Problemkategorie: 	Problemkategorie: 	Problemkategorie: 	Problemkategorie: 

c) Gibt es von den obenen genannten Kategorien welche, zu denen ihr noch keine Probleme gefunden habt? Versucht, Probleme zu brainstormen, die in diese Kategorien passen.

Ein weiteres Problem:	Ein weiteres Problem:	Ein weiteres Problem:	Ein weiteres Problem:
Problemkategorie: 	Problemkategorie: 	Problemkategorie: 	Problemkategorie: 

Teil 2: Probleme auswählen

2 a) Wählt nun im Team vier bis fünf „Top-Probleme“ von der Liste aus und beantwortet folgende Fragen für jedes Problem:

1. Betrifft das Problem die Community, die ihr ausgewählt habt?
2. Ist es tatsächlich ein großes Problem?
3. Betrifft das Problem viele Menschen?
4. Gibt es in Deutschland Stiftungen, Non-Profit-Organisationen, Unternehmen oder Start-ups, die sich für das Problem interessieren?
5. Wurde schon mal in den Nachrichten darüber berichtet?

Tipp // Richtig im Internet recherchieren

- Formuliert eure Suchbegriffe möglichst genau. Wenn ihr zum Beispiel nach „Fleischkonsum“ in Google sucht, erhaltet ihr 628.000 Ergebnisse. Sucht ihr nach „Fleischkonsum in Deutschland“ sind es nur noch 199.000 Ergebnisse. Gebt ihn nun noch zusätzlich „Klimabelastung“ ein, erhaltet ihr nur noch 1.730 Ergebnisse.
- Neben der Suchmaschine Google könnt ihr auch bei Bing, Yahoo, Ecosia oder auch DuckDuckGo suchen.
- Dokumentiert euch immer die Quelle. Also die Internetseite, auf der ihr die Informationen gefunden habt.
- Macht einen Quellen- und Qualitätscheck:
  - Gibt es ein Impressum? (Das ist eine Seite, die darüber informiert, wer für die Inhalte auf der Internetseite verantwortlich ist und wie man die verantwortliche Person erreichen kann.)
  - Gibt es eine:n Autor:in?
  - Wird ein Datum genannt und ist es aktuell?
  - Beziehen sich die Inhalte auf Fakten?
  - Ist der Text sachlich geschrieben oder reißerisch?
  - Gibt es andere Quellen, die die gleichen Fakten benennen?

b) Wenn ihr einen Großteil der Fragen aus der Aufgabe 2a) mit „Ja“ beantwortet habt und euer Team ein großes Interesse an dem Problem hat, dann könnt ihr es auf eure Liste setzen. Falls ihr ein Großteil Fragen mit „Nein“ beantwortet habt, macht noch ein Brainstorming und sucht nach weiteren Problemen.

c) Sucht euch jetzt im Team eure drei Top-Probleme aus.

Unsere drei Top-Probleme:

Tipp // Lokale Probleme lösen

Es ist auch wichtig, lokale Probleme zu lösen. Das Problem, das ihr auswählt, muss kein globales, also kein weltweites Problem sein. Es sollte für die Community, die es betrifft, bedeutsam sein.

Diese Aktivität wurde inspiriert von Creativity in engineering research von David Cropley und Creativity Training von Scott et al.

Reflexion

Haben euch die Übungen geholfen, neue oder andere Probleme aufzudecken? Kennt ihr andere Menschen, Initiativen oder Unternehmen, die sich auch mit eurem ausgewählten Problem befassen?

LÖSUNGEN VERSTEHEN UND AUSWÄHLEN

In dieser Lektion ...

- findet ihr heraus, was innovative Lösungen sind
- prüft ihr, ob ihr Lösungen mit einer mobilen App findet
- überlegt ihr euch eine Lösung für euer Problem

Schlüsselbegriffe

- Lösung
- Innovation
- Mobile App

Verschiedene Arten von Lösungen verstehen

Überlegt einmal, was etwas zu einer Innovation macht. Eine Innovation ist zum Beispiel ein neues Produkt, das der Welt einen Mehrwert bringt oder eine neue Art und Weise, Dinge zu tun. Innovationen können:

- etwas verbessern, das es schon gibt,
- die Kosten von etwas senken, das es schon gibt,
- dabei helfen, dass andere sich das Problem bewusst machen und ihr Verhalten ändern,
- einen Ansatz, den es schon gibt, auf eine neue Situation anwenden,
- eine komplett neue Lösung, Technologie oder Art und Weise finden, Dinge zu tun.

Eine Lösung kann zu mehreren Innovationsarten gleichzeitig gehören. Beispielsweise könnten selbstfahrende Autos eine Verbesserung gegenüber dem sein, was bereits existiert. Es handelt sich aber auch um eine völlig neue Technologie, die es vorher noch nie gab. Behaltet diese Kategorien im Hinterkopf, wenn ihr über Probleme und Lösungen nachdenkt.

Lösungen brainstormen

Nachdem ihr eure Top-Probleme gefunden habt, könnt ihr anfangen, über Lösungen nachzudenken. Recherchiert weiter und versucht, innovative Lösungen zu finden. Möglicherweise habt ihr Internetseiten oder Zeitungsartikel über das Problem gefunden. Lest euch diese Quellen durch, um so viel wie möglich darüber zu erfahren. Macht euch Notizen und vergesst nicht, die verwendeten Quellen aufzuschreiben. Denkt auch an die Tipps zur Recherche und zum Quellencheck aus der vorherigen Lektion „Probleme brainstormen und auswählen". Für das Brainstorming von Lösungen gibt es unterschiedliche Wege und Möglichkeiten. Im Video „Ideation – Einstieg in die Ideenentwicklung" erhaltet ihr hilfreiche Tipps zu einem Brainstorming (ab Minute 3:55).

VIDEO

Neben Brainstorming gibt es noch andere Arten und Weisen, wie man auf Ideen kommt. In dem Video stellen wir euch drei Kreativmethoden vor.

Ihr habt eine Menge Ideen, wie ihr euer Problem angehen könnt? Dann solltet ihr nun darüber nachdenken, welche Lösungen sich denn besonders gut mit einer mobilen App umsetzen lassen und welche Lösungen davon innovativ sind. Einige Ideen beinhalten vielleicht eine mobile App, andere hingegen nicht. Denkt über alle eure Ideen nach und versucht, Wege zu finden, wie ihr die Möglichkeiten einer mobilen App nutzen könnt.

Probleme mit einer mobilen App und Smartphones lösen

Welche Vorteile hat eine mobile App auf einem Smartphone im Vergleich zu einer Webseite, die ihr über den Browser aufruft? Eigenschaften einer App, die ein Vorteil sein könnten, sind zum Beispiel:

- Sie hat Zugriff auf viele Funktionen des Smartphones (z. B. Kamera, Sensoren, Telefonate, SMS, GPS usw.). Deswegen kann eine App z. B. Push-Benachrichtigung senden oder auch vibrieren, wenn es was Neues gibt. Beispiel: Snapchat.
- Sie kann interaktiv sein. Beispielsweise kann sie bei einem Spiel auf den Beschleunigungssensor vom Smartphone zugreifen. Beispiel: Angry Birds.
- Sie kann personalisiert werden, in dem z. B. Informationen der Nutzer:innen gespeichert werden. Beispiel: eine Zyklusapp.
- Sie kann einfacher regelmäßig benutzt werden. Es ist viel bequemer, eine App zu benutzen, als jedes Mal wieder dieselbe Website im Internet aufzurufen. Beispiel: TikTok.
- Eine mobile App funktioniert offline. Da mobile Apps auf dem Smartphone gespeichert sind, können sie Funktionen ohne Internetverbindung ausführen. Beispiel: Spotify.
- Eine mobile App hat eine schönere Oberfläche. Mobile Apps sind für die Geräte entwickelt, auf denen sie verwendet werden. Vergleicht einmal die Instagram-App mit der mobilen Website.

Natürlich haben mobile Apps auch ein paar Nachteile: Man muss sie erst aus dem App Store oder Google Play Store herunterladen und sie brauchen regelmäßig Updates. Außerdem funktionieren viele Apps entweder nur auf dem iOS- oder auf dem Android-Betriebssystem und nicht auf beiden.

Prüft alle eure Ideen darauf, wie gut ihr die Möglichkeiten einer mobilen App nutzen könnt, um das Problem zu lösen.

Aktivität: Lösungen verstehen und auswählen

Ihr braucht:

- Eure drei Top-Probleme aus der Lektion „Probleme brainstormen und auswählen“
- Stifte
- Euer Projekt Canvas

Teil 1: Lösungen verstehen

Nun untersucht ihr, wie innovativ unterschiedliche Erfindungen eigentlich sind. Ordnet dazu die Lösungen einer oder mehreren Innovationsarten zu – je nachdem, um welche Art von Neuerung es sich handelt.
Wählt aus folgenden Innovationsarten:

Verbessern
(etwas weiterentwickeln, das bereits existiert)

Kosten reduzieren
(die Kosten von etwas senken, das bereits existiert)

Aufklären
(Bewusstsein schärfen und Verhaltensänderungen bei Menschen bewirken)

Anwenden
(einen bestehenden Ansatz auf eine neue Situation übertragen oder bereits vorhandene Elemente kombinieren)

Erfinden
(eine völlig neue Lösung finden, z. B. eine Technologie oder eine Arbeitsweise)

Fahrrad	Online-Videokurse	Smartphone	Aspirin
Innovationsart:	Innovationsart:	Innovationsart:	Innovationsart:
Computerschnittstellen im Hirn („Brain Computer Interface“), die Menschen mit Wirbelsäulenverletzungen helfen, einige Fähigkeiten wiederzuerlangen	Spracherkennungssystem auf dem Smartphone, das Angehörigen mitteilt, wenn sich die Person schlecht fühlt	Ein Smartphone-Spiel, das den Nutzer:innen beibringt, was sie recyceln sollen	Spracherkennungssysteme auf dem Smartphone (Alexa, Siri)
Innovationsart:	Innovationsart:	Innovationsart:	Innovationsart:

Automatische Übersetzung (z. B. Google Translate, Deep L)	Einer Ärztin eine Frage per E-Mail oder Textnachricht stellen (Telemedizin)	(Automatisierte) Online-/Film-/ Videoempfehlungen	Öffentlich zugängliche Daten zur Luftqualität
Innovationsart: 	Innovationsart: 	Innovationsart: 	Innovationsart: 
Pokemon Go	Mit dem Smartphone bezahlen	Helm	Bratpfanne
Innovationsart: 	Innovationsart: 	Innovationsart: 	Innovationsart: 
Ride-Sharing-App (z. B. Uber)	Sehbrille	Smartphone-Kamera	Instagram
Innovationsart: 	Innovationsart: 	Innovationsart: 	Innovationsart: 

Teil 2: Lösungen brainstormen

2 a) Nehmt euch jetzt für jedes eurer „Top-Probleme“ ein paar Minuten Zeit und denkt darüber nach, welche Lösungen ihr finden könnt. Schreibt alle eure Lösungsideen auf, auch wilde. Probiert zum Brainstormen einfach mal unterschiedliche Kreativitätsmethoden aus und schaut ob euch das hilft, dass ihr viele Lösungen findet. Auswählen könnt ihr danach im dritten Schritt immer noch.

Unsere drei Top-Probleme (siehe „Probleme brainstormen und auswählen“)	**Mögliche Lösungen**	**Innovationsart** (Mehrfachauswahl möglich)

b) Wählt aus, zu welcher Innovationskategorie die Lösung gehört und tragt es in die 3. Spalte der Tabelle ein. Hier sind noch einmal die Innovationsarten:

Verbessern
(etwas weiterentwickeln, das bereits existiert)

Kosten reduzieren
(die Kosten von etwas senken, das bereits existiert)

Aufklären
(Bewusstsein schärfen und Verhaltensänderungen bei Menschen bewirken)

Anwenden
(einen bestehenden Ansatz auf eine neue Situation übertragen oder bereits vorhandene Elemente kombinieren)

Erfinden
(eine völlig neue Lösung finden, z. B. eine Technologie oder eine Arbeitsweise)

Teil 3: Lösungen bewerten und auswählen

3 a) Um zu überprüfen, wie gut eure Lösungen sind, stellt euch diese Fragen:

- Löst die Lösung das Problem wirklich? Kreuzt an:
  o JA o NEIN
- Findet ihr drei Lösungen, die jemand anders (z. B. ein Unternehmen) schon umsetzt, die sehr ähnlich sind, wie das was ihr vorhabt?
  o JA o NEIN

Wenn ihr JA angekreuzt habt, was macht eure Lösung besser?

- Kennt ihr die Bedürfnisse der Personen, die die Lösungen nutzen würden (Nutzer:innen)?
  o JA o NEIN
- Kann die Lösung eine mobile App sein?
  o JA o NEIN
- Nutzt die Lösung die Funktionen von Smartphones (z. B. GPS, Beschleunigungsmesser, Kamera)?
  o JA o NEIN

Wenn nicht, überlegt, wie ihr bei eurer Lösung eine mobile App nutzen könnt.

---

b) Wählt nun ein Problem und die Lösung(en), die eure App dafür anbieten soll aus. Wählt das Problem und die Lösung aus, die euch im Team alle interessiert und am meisten begeistert. Markiert dafür einfach die entsprechenden Spalten in eurer Tabelle.

c) Beschreibung des Problems
Entscheidet euch jetzt für eines der Top-Probleme und beschreibt. Nutzt das untere Feld gerne für Notizen. Folgende Fragen können euch dabei helfen:

- Welches Problem versucht ihr mit eurem Projekt zu lösen?
- Zu welcher Problemkategorie gehört das Problem?
- Wo begegnet euch dieses Problem?
- Wo sind Menschen davon betroffen und wie wird das Problem aktuell gelöst?
- Warum wollt ihr genau dieses Problem lösen?
- Warum ist es wichtig?

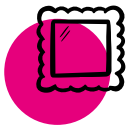
d) Beschreibung der Lösung
Fügt anschließend auch eine Beschreibung eurer Lösung in „Euer Projekt Canvas“ . Hier helfen euch diese Fragen:

- Wie sieht eure Lösung des Problems aus?
- Wann, wie oft und wofür (in welchen Situationen) sollen Nutzer:innen die App verwenden?

e) Übertragt nun das Problem und die Lösung, für die ihr euch entschieden habt in „Euer Projekt Canvas“ .

Diese Aktivität wurde inspiriert von Creativity in engineering research von David Cropley und Creativity Training von Scott et al.

☞ **Hinweis:** Wahrscheinlich wird sich eure Idee im Laufe der Zeit noch weiterentwickeln. Eure App könnte am Ende auch völlig anders aussehen als ihr es euch jetzt vorstellt. Macht euch keine Sorgen: Das ist alles Teil der Ideenentwicklung!



Reflexion

Ist es euch leicht oder schwer gefallen, euch für ein Problem zu entscheiden oder Lösungen für die Probleme zu finden? Warum?

MARKTFORSCHUNG DURCHFÜHREN

In dieser Lektion ...

- legt ihr eure genaue Zielgruppe fest
- befragt ihr eure Zielgruppe, um mehr über sie herauszufinden
- entscheidet ihr, wie ihr auf die Ergebnisse der Befragung (Nutzer:innenforschung) reagiert, um eure Idee zu verbessern

Schlüsselbegriffe

- Zielgruppe
- Nutzer:innenforschung
- Konkurrenz
- Produkt

Jetzt, wo ihr eine Idee für ein Projekt habt, wollt ihr wahrscheinlich sofort mit der Umsetzung loslegen. Doch auch wenn ihr eure Idee super findet, solltet ihr sie noch einmal überprüfen. So stellt ihr sicher, dass sie auch bei anderen gut ankommt.

Ein erster Schritt in der Marktforschung ist es, Informationen über eure Zielgruppe, also eure Nutzer:innen zu sammeln. Wie findet ihr heraus, wer genau zu euren Nutzer:innen gehört? Vielleicht löst ihr ein Problem für Teenager, ältere Menschen, Eltern oder jemand anderen in eurer Community? Schaut euch dafür nochmal den Baustein Unsere Community auf „Eurem Projekt Canvas“ an.

Hinweis: Ihr solltet die Begriffe Community und Zielgruppe nicht verwechseln. Die Zielgruppe ist zwar ein Teil der Community, die ihr euch ausgesucht habt, jedoch noch etwas spezifischer. Eine Zielgruppe könnte also zum Beispiel nur eine bestimmte Altersgruppe aus der Community sein oder nur Eltern kleiner Kinder. Je spezifischer, also genauer und detaillierter, ihr euch auf eine Zielgruppe festlegt, desto einfacher wird es, eure App speziell auf die Bedürfnisse der Zielgruppe auszurichten.

Durch Recherche könnt ihr eure App-Idee so anpassen, dass sie von mehr Menschen genutzt wird. Hier sind einige Fragen, die ihr am Ende dieser Lektion beantworten könnt:

- Gibt es genug Leute, die eure Appt nutzen werden?
- Löst unsere App das Problem?
- Wie können wir unsere Idee an die Bedürfnisse der Zielgruppe anpassen?

Nutzer:innenforschung

Bei der Nutzer:innenforschung informiert ihr euch über die Menschen, die eure App nutzen sollen, um ihre Wünsche und Bedürfnisse zu verstehen. Diese Informationen helfen euch, eine App zu entwickeln, die die Leute auch wirklich nutzen wollen. Je nachdem, welche Aussagen eure Interviewpartner:innen treffen, könnt ihr eure Ideen und die Funktionen der App anpassen.

Konkurrenzanalyse

Ein weiterer wichtiger Punkt der Marktforschung ist die Konkurrenzanalyse. Sie hilft euch, herauszufinden, welche Unternehmen es schon gibt, die dasselbe Problem lösen wie ihr. Ihr möchtet, dass eure Zielgruppe euer Produkt nutzt und kein Konkurrenzprodukt. Dafür müsst ihr sicherstellen, dass eure App besser ist als das, was es bereits gibt. Behaltet dabei im Hinterkopf, dass die Lösung, die eure Konkurrenz anbietet, vielleicht was anderes ist als eine App.

Aktivität: Nutzer:innenforschung und Konkurrenzanalyse

Ihr braucht:

- Stifte
- Euer Projekt Canvas

Teil 1: Interviews zur Nutzer:innenforschung

Tipp // Interviewpartner:innen finden

Um Menschen für eure Befragung zu finden, könntet ihr zum Beispiel einen Aushang in öffentlichen Einrichtungen der Community machen. Das kann die Pinnwand in einem lokalen Einkaufsmarkt sein oder – falls eure Zielgruppe Schüler:innen sind – ein Aushang in Schulen. In dem Aushang beschreibt ihr kurz euer Projekt und warum ihr eine Nutzer:innenbefragung durchführen möchtet. Für alle Interessierten könnt ihr dann einen Kontakt vergeben, wie sie euch erreichen können. Ihr könntet aber auch einen Stand in der Fußgängerzone aufstellen und aktiv die Menschen ansprechen. Zudem könnt ihr auch einen Aufruf über Social Media starten.

1 a) Befragt mindestens zwei Personen, von denen ihr glaubt, dass sie zur Zielgruppe für eure App gehören. Je mehr Personen ihr interviewt, desto besser! Befragt sie zu dem Problem und der Lösung, für die ihr euch entschieden habt, und beschreibt euer Vorhaben. Die Informationen, die ihr von euren Interviewpartner:innen bekommt, sind entscheidend dafür, wie und ob sich eure Ideen noch verändern werden, schließlich möchtet ihr ja eurer Zielgruppe mit eurer App helfen. Überlegt euch, was ihr sonst noch von euren Nutzer:innen wissen möchtet. Das könnten zum Beispiel Fragen über ihr Nutzer:innenverhalten sein. Also ob sie denn selbst andere Apps nutzen und wenn ja, welche Funktionen ihnen an anderen Apps gefallen.

	Person 1	**Person 2**
Ist das Problem, das unsere App angeht, ein Problem in deiner Community?		
Was tust du gegen dieses Problem?		
Erinnert dich unsere App an eine Lösung, die es schon gibt?		
Bist du daran interessiert, unsere Lösung (App) zu nutzen?		
Gibt es einen Grund, warum du diese App nicht benutzen würdest?		
Eigene Frage		

b) Was habt ihr aus der Befragung über eure Idee und ihre Funktionen gelernt? Wie wollt ihr euer Projekt – also eure App – aufgrund der Ergebnisse der Befragung ändern?

c) Überprüft nun erneut die Wahl eurer Zielgruppe, also der Menschen, die eure App nutzen sollen. Passt sie zu dem Baustein Meine Community in „Eurem Projekt Canvas“? Sind das die Menschen, deren Problem ihr lösen wollt? Legt dann endgültig eure Zielgruppe fest und tragt sie in „Euer Projekt Canvas“ ein.

Teil 2: Konkurrenzanalyse

2 a) Findet mindestens drei Unternehmen oder Produkte, die sich demselben Problem widmen wie eure App. Recherchiert im Internet und beantwortet folgende Fragen: Wie funktioniert das Produkt? Welches Problem löst es? Was ist daran besonders? Wer ist die Zielgruppe? Welchen Preis hat das Produkt?

Unternehmen / Produkt	Funktions-weise	Problem / Lösung	Zielgruppe / Preis	Besonderheit

b) Was könnt ihr davon lernen? Überlegt euch, wie ihr das, was ihr gelernt habt, nutzen könnt, um eure Idee zu verbessern.

Reflexion

Nun solltet ihr einen besseren Überblick über eure Nutzer:innen und Konkurrenz haben. Denkt daran: Es ist vollkommen in Ordnung, eure Ideen anzupassen und weiterzuentwickeln. Es ist sogar wichtig, dass ihr eure Ideen abändert – je nachdem, was ihr gelernt und erfahren habt! Reflektiert noch einmal die letzte Lektion. War es schwer, Interviewpartner:innen zu finden? Worin lag das Problem und was könntet ihr das nächste Mal besser machen?

ENTWURF EURER APP ERSTELLEN

In dieser Lektion ...

- erfahrt ihr, was ein funktionsfähiges Produkt (Englisch: Minimum Viable Product) ist und warum das wichtig ist
- wählt ihr aus, welches die wichtigsten Funktionen eures Produkts sind

Schlüsselbegriffe

- Funktionsfähiges Produkt (Englisch: Minimum Viable Product, MVP)
- Prototyp
- Testen

Minimum Viable Product (MVP)

Jetzt, wo ihr euch für eine App-Idee und eine Problemdarstellung entschieden habt, möchtet ihr wahrscheinlich sofort mit dem Programmieren beginnen. Ihr könnt eine Menge Zeit sparen, wenn ihr ein wenig plant, bevor ihr anfangt. Eine App, die ein oder zwei Dinge wirklich gut kann, ist besser als eine App, die alles nur mittelmäßig kann. Dafür erstellt ihr in dieser Lektion einen Entwurf, das heißt, ihr wählt aus, welche die wichtigsten Funktionen eurer App sind.

Ein Minimum Viable Product (MVP) ist eine App, die gerade genug Funktionen hat, um ihre Aufgabe zu erfüllen und deswegen funktionsfähig ist. So kann sie von Nutzer:innen getestet werden, damit später Verbesserungen vorgenommen werden können. Eine wichtige Funktion kann zum Beispiel ein Chat sein, wenn ihr eine App für sozialen Kontakt entwickeln wollt. Auch eine Liste kann eine erste wichtige Funktion sein, wenn ihr zum Beispiel eine Telefonbuch-App programmiert.

Ziel dieses Handbuchs ist es, dass ihr am Schluss eine erste funktionsfähige Version euer App entwickelt habt – also ein MVP eurer App. In dieser Lektion entscheidet ihr, welche Funktionen ihr in euer MVP aufnehmt und welche zukünftigen Funktionen ihr möglicherweise später erstellt. Im Moment habt ihr wahrscheinlich eine lange Wunschliste von Dingen, die eure App erledigen soll. Um ein MVP zu erstellen, entscheidet ihr, welche Funktionen am wichtigsten sind und welche ihr zuerst programmiert. Denkt darüber nach, welche dieser Funktionen euer Problem wirklich lösen und welche sich eure Nutzer:innen wünschen.

Es ist wichtig, früh zu testen, welche der Funktionen die Nutzer:innen sich wünschen und wie ihr sie am besten umsetzen könnt. Dafür ist es gut, wenn ihr einen Prototyp erstellt. Ein Prototyp ist eine allererste Version eures MVPs. Das kann am Anfang auch einfach ein auf Papier gezeichneter App-Bildschirm sein, wo ihr zeigt, wie eure App aussieht und was die Nutzer:innen damit machen können. Diese Prototypen könnt ihr dann euren Nutzer:innen oder Personen in eurem Umfeld, zum Beispiel Freund:innen oder Eltern zeigen, und nach Feedback fragen. Das nennt sich Testen. Ihr entwickelt dann die Prototypen immer weiter, bis ihr genauer wisst, was eure Nutzer:innen sich wünschen. Es ist gut, die Wünsche genauer zu kennen, bevor ihr mit der Programmierung beginnt, damit ihr keine Funktionen programmiert, die am Ende niemand braucht.

Aktivität: Sich für ein MVP entscheiden

Ihr braucht:

- Stifte/Marker
- Euer Projekt Canvas

1 a) Erstellt eine Liste mit allen Dingen, die eure App machen soll. Achtet dabei darauf, dass ihr große Ideen in kleinere Funktionen unterteilt. Zum Beispiel:

Große Idee:
Ihr wollt etwas dagegen tun, dass sich Menschen einsam fühlen. Also wollt ihr ein soziales Netzwerk entwickeln, wo sich Nutzer:innen als Freund:innen vernetzen können, Bilder posten und private Nachrichten austauschen.

Kleinere Funktionen:

- Nutzer:innen können sich mit anderen Nutzer:innen vernetzen.
- Nutzer:innen können Bilder machen und sie auf ihren Profilen posten.
- Nutzer:innen können einen Feed von Bildern sehen, die andere gepostet haben.
- Nutzer:innen können sich gegenseitig Nachrichten schreiben.

Tragt eure Überlegungen in die Liste ein:

Große Idee	Kleinere Funktionen

b) Überlegt, welche Funktionen aus der Liste die wichtigsten sind, die ihr programmieren wollt? Entscheidet euch und markiert die jeweiligen Ideen in eurer Liste. Stellt euch dazu folgende Fragen:

- Welche Funktionen sind nötig, um das Problem zu lösen?
- Welche Funktionen würden eure Nutzer:innen verwenden?
- Welche Funktionen helfen euch, euch von eurer Konkurrenz abzuheben?

c) Wenn ihr fertig seid, ordnet die Funktionen nach ihrer Wichtigkeit und nummeriert sie dazu mit 1., 2., 3. usw. Für euer MVP solltet ihr euch nur auf die ersten beiden Funktionen konzentrieren.

Teil 2: Eure Auswahl testen

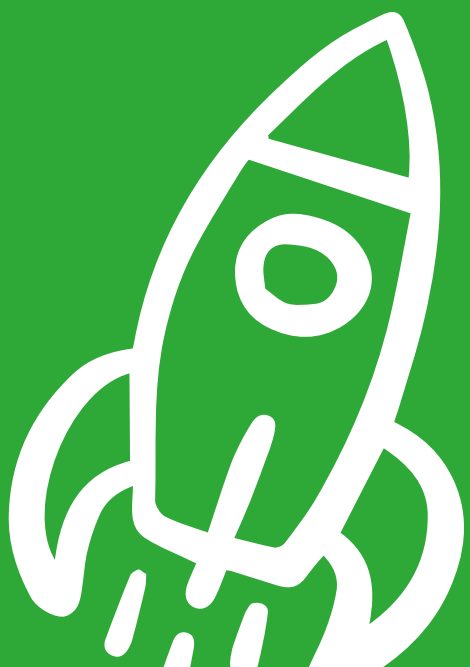
Nun ist es an der Zeit, zu überprüfen, ob euer MVP das Problem auch tatsächlich löst: Eine Person aus dem Team tut so, als wäre sie einer der Menschen, die das Problem hat, das ihr mit euer App lösen möchtet. Stellt ihr das MVP vor. Hilft es, ihr Problem zu lösen? Seid gegenüber eurem MVP so kritisch wie möglich. Wenn euer MVP das Problem nicht lösen kann, müsst ihr eure Funktionen vielleicht neu gewichten und andere auswählen.

Entscheidet euch schließlich für zwei Funktionen, die eure App haben soll und tragt diese in „Euer Project Canvas“ ein.

Reflexion

Hattet ihr Schwierigkeiten, die Funktionen für euer Projekt zu priorisieren? Warum oder warum nicht? Wie habt ihr einen Kompromiss zwischen euren Lieblingsfunktionen und den besten Funktionen zur Lösung eures Problems gefunden?

UNTERNEHMER-TUM



UNTERNEHMENSARTEN KENNENLERNEN UND MISSION SCHREIBEN

In dieser Lektion ...

- lernt ihr etwas über verschiedene Arten von Unternehmen
- wählt ihr aus, um welche Art von Unternehmen es sich bei eurem Projekt handelt
- erfahrt ihr, wie ihr eine Mission für euer Unternehmen schreibt
- findet ihr einen Namen für eure App

Schlüsselbegriffe

- Unternehmen (Englisch: Business)
- Gewinn (Englisch: Profit)
- Gemeinnützig (Englisch: Non-Profit)
- Sozialunternehmen
- Gewinnorientiert (Englisch: For-Profit)
- Mission (Englisch: Mission Statement)

Hinweis: Wenn es um Unternehmen geht, werden auch in der deutschen Sprache viele englische Begriffe verwendet wie zum Beispiel Business statt Unternehmen oder Profit statt Gewinn. Wir haben hier die englischen Begriffe, die oft verwendet werden, in Klammern gesetzt, damit ihr sie wiedererkennt, wenn sie euch mal begegnen.

Ein Unternehmen ist eine Organisation oder Person, die etwas im Tausch gegen Geld oder ein anderes Gut tut. Unternehmen können Waren herstellen, kaufen oder verkaufen (z. B. Unternehmen, die Autos herstellen) oder sie können Dienstleistungen anbieten (z. B. Friseur:in).

VIDEO

In diesem Video findet ihr eine Einführung in das Thema Unternehmertum.

VIDEO

Zur Inspiration könnt ihr euch auch dieses Interview mit einer Gründerin anschauen.

Es gibt verschiedene Arten von Unternehmen. Gewöhnlich betrachtet man ein Unternehmen als eine Möglichkeit, Geld zu verdienen. Aber Unternehmen können auch andere Ziele haben. Das können soziale Ziele sein, wie Armut und Hunger zu bekämpfen. Es können auch geschäftliche Ziele sein, wie die Herstellung umweltfreundlicher Produkte. Alle Unternehmen haben eines gemeinsam: Um weiterzuarbeiten, benötigen sie Geld.

Gewinnorientiert (For-Profit) – Geld verdienen

Gewinnorientierte Unternehmen konzentrieren sich darauf, durch den Verkauf von Waren oder Dienstleistungen einen Gewinn zu erzielen.

Sozialunternehmen – Geld verdienen und Gutes tun

Sozialunternehmen konzentrieren sich auf eine Tätigkeit, die über das Erzielen von Gewinn hinausgeht. Beispielsweise wollen sie bei der Lösung eines sozialen Problems helfen. Sie verwenden einen Teil des Gewinns, den sie erzielen, um dieses Ziel zu erreichen.

Gemeinnützig (Non-Profit) – Gutes tun

Gemeinnützige Organisationen sind dazu da, ein Problem zu lösen oder zu einer wichtigen Sache beizutragen. Sie haben nicht das Ziel, Geld zu verdienen.

Egal welche Unternehmensform ihr wählt, formuliert anschließend eine Mission für euer Unternehmen. Eine Mission ist eine Zusammenfassung der Werte eines Unternehmens, einer Organisation oder einer Einzelperson. Es hilft euch, zu bestimmen, was wichtig ist und was nicht. Es gibt klar an, wem das Unternehmen auf welche Art und Weise nützen soll. Normalerweise ist eine Mission ein kurzer und einfacher Satz, der umreißt, was der Zweck der Organisation ist und wie sie diesen Zweck erfüllt. Sie bildet das Herz eures Projekts.

Aktivität: Eine Unternehmensart, eine Mission und einen Namen auswählen

Ihr braucht:

- Stifte
- Euer Projekt Canvas

Teil 1: Eine Unternehmensform finden

1 a) Entscheidet euch nun: Soll das Unternehmen, das ihr aufbauen wollt, ein gewinnorientiertes, soziales oder gemeinnütziges Unternehmen sein? Um euch die Entscheidung zu erleichtern, besprecht die folgenden Fragen und macht euch dazu Notizen:

1. Was wollt ihr mit eurem Unternehmen erreichen?
2. Was sind einige eurer Unternehmensziele?
3. Wie könnt ihr euer Unternehmen am Laufen halten? Also woher bekommt ihr das Geld, das ihr braucht, um euer Unternehmen zu betreiben?
4. Welche Art von Unternehmen eignet sich aus eurer Sicht am besten, um das Problem zu lösen, das ihr angehen wollt?
5. Welche Unternehmensart wählt ihr aus? (gewinnorientiertes, gemeinnütziges oder soziales Unternehmen?)

b) Entscheidet euch nun für eine Unternehmensform und tragt sie in „Euer Projekt Canvas“ ein.

Teil 2: Eine Mission schreiben

2 a) Entwickelt nun gemeinsam eine Mission. Sie sollte zwei bis drei Sätze lang sein und die Ziele eures Unternehmens beschreiben. Notiert euch die wichtigsten Überlegungen. Hier sind einige Fragen, die euch dabei helfen:

- Was wollen wir tun?
- Wie machen wir das?
- Für wen tun wir das?
- Welchen Mehrwert schaffen wir?

b) Formuliert nun eure Mission und schreibt sie in „Euer Projekt Canvas“.

Teil 3: Einen App-Namen finden

3 a) Vielleicht hilft euch eure Mission auch dabei einen Namen für eure App zu finden. Wenn euch nicht sofort ein Name einfällt, macht wieder ein Gruppenbrainstorming. Seid kreativ und habt Spaß! Eure Ideen aus dem Brainstorm könnt ihr in die Notizen schreiben.

Tipp // App-Name

Es ist gut, wenn euer App-Name einprägsam ist. Gleichzeitig sollten Nutzer:innen, die eure App noch nicht kennen, gleich wissen, um welches Thema es in eurer App ungefähr geht.

b) Einigt euch nun auf einen Namen und schreibt ihn in „Euer Projekt Canvas“.

Reflexion

Was denkt ihr: Warum gibt es verschiedene Arten von Unternehmen?
Was sind eurer Meinung nach die Vorteile der verschiedenen Unternehmensarten?

EIN LOGO GESTALTEN

In dieser Lektion ...

- lernt ihr etwas über verschiedene Schriftarten
- sucht ihr eine Schriftart für eure Marke aus
- gestaltet ihr ein Logo für euer Unternehmen

Schlüsselbegriffe

- Logo
- Schriftarten
- Serifen
- Serifenschrift
- Serifenlose Schrift

Wenn ihr wollt, dass euer Unternehmen leicht wiedererkannt werden kann, dann braucht ihr neben einem tollen Namen auch ein Logo. Ein Logo ist ein Symbol, das ein Unternehmen nach außen darstellt. Das Logo eines Unternehmens soll dessen Ziele, Zweck und Charakter in einer einfachen Mischung aus Symbolen und Wörtern enthalten. Schaut euch diese Beispiele an:

Wenn ihr euer Logo entwerft, müsst ihr euch für eine dieser drei Hauptkategorien entscheiden.

- Wortmarke wie zum Beispiel facebook
- Symbol oder Bild wie zum Beispiel
- Kombination wie zum Beispiel

Schriftarten und ihre Wirkung

Außerdem müsst ihr bei der Gestaltung auf die Wahl der Schriftarten achten. Sie beeinflussen die visuelle Identität eurer Marke, das heißt, die Wirkung, die die Schriftart auf die Menschen hat, wenn sie sie anschauen. Jede Schrift hat eine „Persönlichkeit". Die beiden Hauptkategorien von Schriftarten sind Serifenschriften und serifenlose Schriften.

FRS

Serifenschriften vermitteln ein Gefühl von Geschichte, Macht und Förmlichkeit. Wenn ihr euch für eine Serifenschrift entscheidet, könnt ihr ein Gefühl von Weisheit und Alter vermitteln. Die roten Markierungen auf den drei Buchstaben werden als Serifen bezeichnet.

FRS

Serifenlose Schriften sind viel jüngere Schriftarten. Sie stehen für Jugendlichkeit und eine neue Denkweise.

Tipp // Farben

Wusstet ihr, dass auch Farben beeinflussen können, wie Menschen sich fühlen? Hier findet ihr einige Farben und mögliche Assoziationen – also Gefühle oder Gedanken – die mit den Farben in Verbindung gebracht werden. Diese Assoziationen können sich jedoch in verschiedenen Ländern oder Kulturen unterscheiden. Deswegen solltet ihr nochmal recherchieren, was die Farben für eure Zielgruppe bedeuten.

Farbe	Beschreibung
ROT	leidenschaftlich, aggressiv, wichtig, revolutionär
ORANGE	spielerisch, energisch, billig, lebhaft
GELB	glücklich, freundlich, warnend, weise
GRÜN	natürlich, stabil, wohlhabend, ehrlich
BLAU	gelassen, vertrauenswürdig, einladend, loyal
VIOLETT	luxuriös, mysteriös, romantisch, inspirierend
BRAUN	erdig, robust, rustikal
SCHWARZ	leistungsstark, anspruchsvoll, trendy, formal
WEISS	sauber, tugendhaft, gesund, unschuldig
GRAU	neutral, formal, trübsinnig, bescheiden

Aktivität: Ein Logo gestalten

Ihr braucht:

- Stifte
- Computer oder Tablet
- ggf. Grafikprogramm (z. B. Paint, Gimp)

Teil 1: Erstellt ein Logo für eure App!

1 a) So könnt ihr für euer Logo brainstormen: Holt euch Inspiration. Recherchiert im Internet und sammelt ein paar Bilder, von denen ihr denkt, dass sie das Problem zeigen, das ihr lösen wollt. Denkt an eure Zielgruppe und die von euch vorgeschlagene Lösung. Als Inspiration könnt ihr auch Bilder und Logos von anderen Unternehmen nutzen. Notiert euch die Marken, der Logos, die euch gefallen.

b) Entscheidet euch, ob ihr das Logo als Wortmarke, als Symbol oder als eine Kombination aus beiden machen wollt. Wenn ihr euch für ein Logo mit Schrift entscheidet, dann wählt eine Schriftart aus.

c) Jedes Mitglied eures Teams zeichnet Skizzen, wie das Logo aussehen könnte. Versucht die Skizzen so schnell wie möglich zu machen – ohne groß darüber nachzudenken. Und denkt daran: Keine Skizze ist schlecht!

Tipp // Skizzen erstellen

Erstellt die Skizzen in schwarz-weiß. Am besten zeichnet ihr mit der Hand, bevor ihr zum Computer wechselt. Nach 15 bis 30 Minuten solltet ihr eine große Sammlung von Skizzen haben.

d) Schaut zusammen auf eure Skizzen. Macht ein rotes Häkchen neben die fünf Skizzen, die euch am besten gefallen. Wählt Skizzen, die vielversprechend sind – oder die Eigenschaften haben, die ihr kombinieren könnt, um ein stärkeres Logo zu kreieren. Entwickelt nun jede der fünf Skizzen weiter: Zeichnet weitere Versionen davon und schaut, wie sie aussehen.

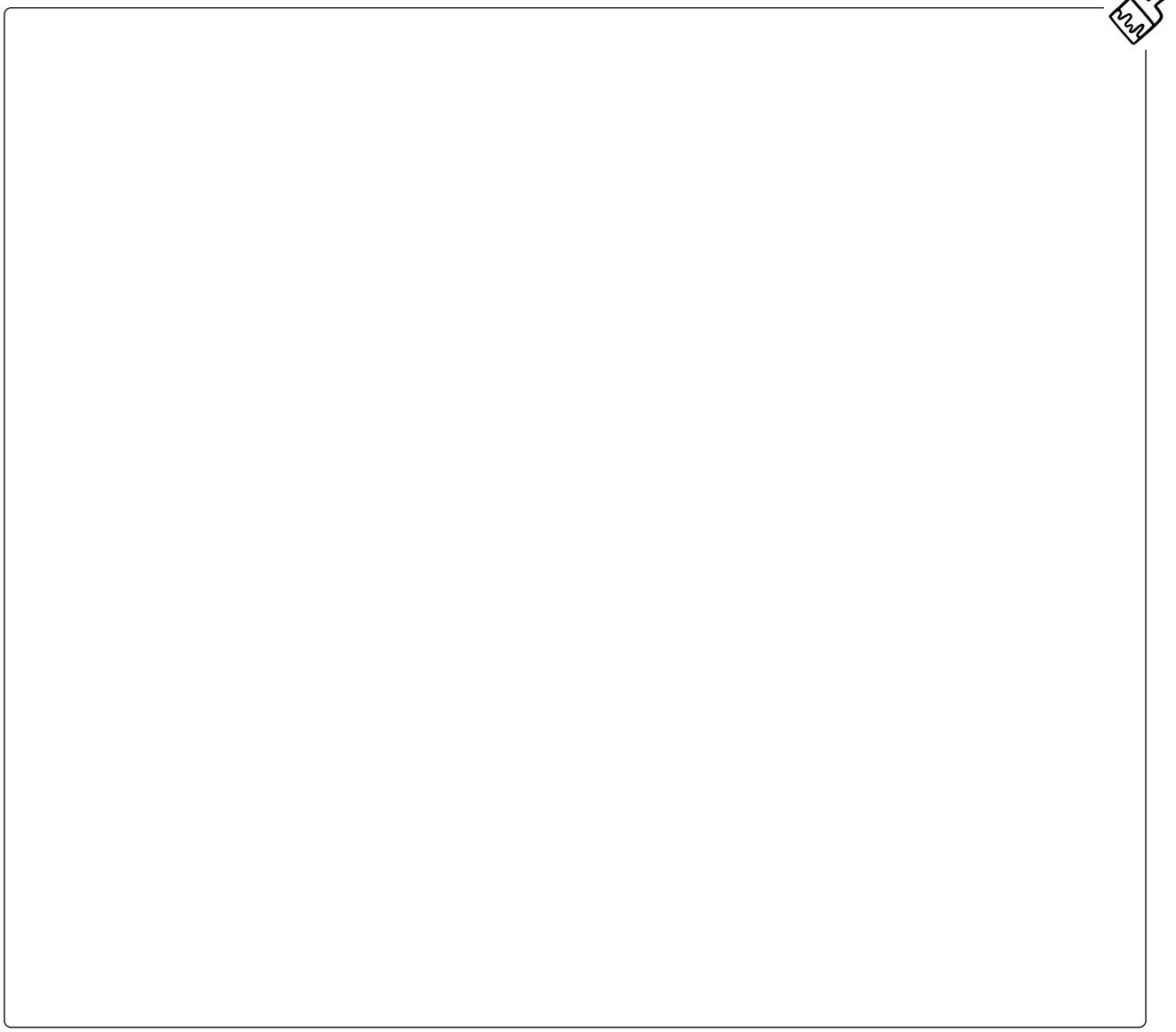
e) Wählt nun die Idee, die euch am besten gefällt aus. Versucht, diese Idee auf unterschiedliche Weise herauszuarbeiten. Wenn ihr digitale Zeichenwerkzeuge verwenden möchtet, könnt ihr das jetzt tun. Denkt daran, alles in schwarz-weiß zu halten und noch keine Farben hinzuzufügen. Bedenkt dabei, dass euer Logo auch gut aussehen sollte, wenn es wirklich klein ist (z. B. auf einem Smartphone-Bildschirm).

f) Entscheidet euch für ein Logo. Konzentriert euch dazu auf das Logo, das euch am besten gefällt. Zeichnet das Logo entweder von Hand oder digital – so lange, bis ihr zufrieden seid und es „genau richtig“ ist.

g) Ihr möchtet Wörter in euer Logo aufnehmen? Dann experimentiert jetzt mit der Wortplatzierung. Wenn euer Logo bereits Wörter enthält oder keine Schrift enthalten soll, könnt ihr diesen Schritt überspringen.

h) Nachdem ihr nun ein Logo habt, das euch in schwarz-weiß gefällt, überlegt euch, welche Farben zum Charakter eurer App passen. Wählt erstmal eine Hauptfarbe aus. Dann könnt ihr noch ein bis zwei weitere Farben hinzufügen.

i) Stellt euer Logo fertig. Wählt dazu eine oder zwei farbige Variationen eures Logos aus. Erstellt ein größeres Logo und ein kleineres Logo. Das größere könnt ihr zum Beispiel für euren Startbildschirm eurer App verwenden - das kleinere für Visitenkarten oder eure App.



Teil 2: Holt euch Feedback

Zeigt euer Logo einer oder mehreren Personen aus eurer Zielgruppe. Fragt sie, woran sie denken, wenn sie das Logo sehen. Ergibt die Antwort für eure Marke Sinn? Nutzt das Feedback, um euer Logo nochmal anzupassen.

Tipp // Logo-Maker

Es gibt auch kostenlose Programme, die für euch ein Logo erstellen. Ihr werdet sehen, dass diese Programme ähnlichen Schritten folgen. Ihr könnt euch davon inspirieren lassen und Ideen für euer Logo sammeln. Beispiele sind: Hatchful und Logo.com

Reflexion

Wie und wieso habt ihr die Farben für euer Logo ausgewählt? Welche Schriftart habt ihr gewählt und warum? Was hatten die Personen, denen ihr euer Logo gezeigt habt für einen Eindruck? Habt ihr euch dasselbe vorgestellt?

GESCHÄFTSPLAN SCHREIBEN

In dieser Lektion ...

- erstellt ihr einen Plan, damit mehr Menschen eure App nutzen

Schlüsselbegriffe

- Marketing
- Strategie
- Geschäftsplan

Jedes Unternehmen sollte einen Geschäftsplan haben. In einem Geschäftsplan stehen die Nutzer:innen im Mittelpunkt. Darin wird beschrieben, welche Nutzer:innen die App testen und wie deren Feedback dazu aussieht. Und ihr beschreibt, wie ihr mehr Menschen dazu bringen könnt, die App auszuprobieren.

Euer Geschäftsplan sollte Folgendes beinhalten:

- Eine Übersicht eurer Nutzer:innen, einschließlich ihrer Anzahl und ihrer demografischen Daten. Demografische Daten sind: Alter, Geschlecht, Wohnort, Beruf, Familienstand usw.
- Beschreibung, wie ihr die Nutzer:innen angesprochen habt.
- Beschreibung, wie sich eure Idee nach dem Feedback eurer Zielgruppe verändert hat.
- Zusätzlich könnt ihr auch eine Konkurrenzanalyse machen, um dadurch zu erfahren, wie ihr die Nutzer:innen dazu bringt, euer Produkt statt das der Konkurrenz zu nutzen.

Im zweiten Teil eures Geschäftsplans beschreibt ihr, wie ihr euer Produkt bei möglichen Nutzer:innen bekannter macht. Es geht auch darum, möglichst genau zu beschreiben, wie Menschen von eurem Produkt erfahren, also zum Beispiel durch Werbeplakate oder eine Aktion auf Social Media. Ihr könnt auch beschreiben, wie die Nutzer:innen die Möglichkeit bekommen, euer Produkt zu erhalten und vielleicht auch zu kaufen.

Ihr sollt also hauptsächlich erklären, wie ihr den Menschen euer Produkt vorstellen wollt, wie ihr sie dazu bringt, es auch zu benutzen und was passiert, nachdem die Menschen euer Produkt ausprobiert haben. Um diese Punkte bearbeiten zu können, benötigt ihr:

- Eine eindeutige Sprache, die den Zweck eures Produkts erklärt.
- Eine genaue Strategie, die aufzeigt, wie euer Produkt die Zielgruppe erreicht. Strategie heißt, eine Abfolge von Schritten, um ein Ziel zu erreichen.

Tipp // Anpassungen vornehmen

Es ist normal, dass ihr am Anfang noch nicht alles wisst, was in den Geschäftsplan geschrieben werden soll. Arbeitet während der Entwicklung eures MVPs einfach immer wieder an eurem Geschäftsplan und füllt die Sachen ein, die ihr lernt. Ihr könnt euer Unternehmen auch jederzeit anpassen und zum Beispiel die Unternehmensart ändern, während ihr euer Produkt und euren Geschäftsplan entwickelt.

☞ **Hinweis:** Wenn ihr ein Unternehmen gründen wollt, dann wäre der nächste Schritt ein Businessplan. Dieser enthält ausführliche Informationen über euch und euer Produkt. Ihr beschreibt darin genau, wie ihr euer Vorhaben umsetzen wollt, wer eure Kund:innen sind und mit wem ihr konkurriert, eure Marketingstrategie (Preise, Vertrieb und Werbung) sowie Risiken und Chancen. Zusätzlich erstellt ihr einen detaillierten Finanzplan, also einen Kosten- und Finanzierungsplan, der eure geschätzten Einnahmen und Ausgaben für die nächsten drei Jahre enthält.

Aktivität: Plant und schreibt euren Geschäftsplan (1-2 Seiten)

Ihr braucht:

- Stifte

Nun könnt ihr beginnen einen Geschäftsplan zu schreiben. Möglicherweise seid ihr gerade an einem Punkt in eurem Projekt, an dem ihr noch nicht alles ausfüllen könnt. Das ist kein Problem und ganz normal. In dem Fall könnt ihr die fehlenden Inhalte nachtragen, sobald ihr soweit seid.

Geschäftsplan

Wie heißt eure App und was sind ihre wichtigsten Funktionen? 1 bis 2 Sätze

Wie habt ihr Menschen dazu gebracht, eure App zu nutzen? 3 bis 5 Sätze
(Durch Einladung von Freund:innen und Familien, die App zu testen, Zusammenarbeit mit Unternehmen, Verbreitung auf Social Media usw.)

Wie viele Menschen haben eure App bis jetzt genutzt? 1 Satz

Welches Feedback habt ihr von ihnen erhalten? 3 bis 10 Sätze
(Beschreibt das Feedback so genau wie nur möglich, auch welches davon euch wahrscheinlich bei Änderungen am Design/Entwürfen beeinflusst hat.)

Wie wollt ihr in diesem Jahr Menschen oder Unternehmen dazu bringen, eure App zu nutzen?
3 bis 10 Sätze *(z. B. mit Marketing auf Social Media oder Präsentationen an der Schule)*

Reflexion

Einen Geschäftsplan anzufertigen, ist eine wichtige Aufgabe. Besonders, weil dadurch sichergestellt wird, dass euer Produkt von den Menschen genutzt wird, die es wirklich brauchen! Was konntet ihr schon in die Vorlage des Geschäftsplans eintragen? Und was müsst ihr noch herausfinden?

PRODUKTPRÄSENTATION ERSTELLEN

In dieser Lektion ...

- lernt ihr die Anforderungen an eine Produktpräsentation kennen
- erstellt ihr ein Storyboard für eure Präsentation

Schlüsselbegriffe

- Produktpräsentation
- Produkt-Video
- Pitch
- Demo
- Storyboard

Die Präsentation, in der ihr eure App vorstellt und für sie werbt, sollte maximal vier bis sechs Minuten lang sein. Traut euch dabei ruhig, eure persönliche Motivation und Begeisterung, die ihr für die Lösung des Problems habt, zu zeigen! Ziel eurer Präsentation ist es, die Zuschauer:innen, zukünftige Kund:innen und Geldgeber:innen von euren Ideen zu überzeugen und euer Produkt vorzustellen.
Es gibt verschiedene Formen von Produktpräsentationen. In einem Pitch (Kurzpräsentation) geht es zum Beispiel darum, das Problem und die Lösung so kurz und überzeugend wie möglich vorzustellen. Ein Pitch ist circa vier Minuten lang.

VIDEO Nähere Informationen zum Pitching und hilfreiche Tipps findet ihr in diesem Clip.

VIDEO In diesem Video bekommt ihr weitere praktische Einblicke von einer Expertin.

VIDEO Und hier ist ein Beispiel für ein Pitch einer mobilen App im Videoformat.

Demo-Video

VIDEO Eine Demo (kurz für Demonstration, also Vorführung) ist circa 2 Minuten lang. Hier wird die Technologie und die Funktion eurer App dargestellt, die App wird also vorgestellt. Ein Beispiel für eine Demo, wieder im Videoformat, gibt es hier.

Nun seid ihr dran, euch Gedanken zu eurer Produktpräsentation zu machen. Ihr müsst euch dabei nicht zwischen einem Pitch oder einer Demo entscheiden, sondern könnt auch beide Formen vereinen. Wichtig ist, dass folgende Fragen beantwortet werden:

- Was für eine App habt ihr genau entwickelt? Wie funktioniert die App?
- Welches Problem löst die App und warum ist das für eure Zielgruppe wichtig?

- Hängt das Problem mit den UN-Zielen für nachhaltige Entwicklung (SDGs) zusammen?
- Wie hilft eure App das Problem zu lösen? Was ist die Technologie dahinter?
- Warum ist ausgerechnet eure Lösung die beste Option?
- Woher habt ihr das Nutzer:innen-Feedback zum Problem und eurer Lösung erhalten? Wie habt ihr darauf reagiert? Habt ihr die Idee verändert/angepasst?
- Welche Ziele möchtet ihr mit eurem Projekt erreichen?

Länge der Präsentation

Am besten ist für die Produktpräsentation eine Länge von maximal vier bis sechs Minuten, da durch lange Vorträge das Interesse des Publikums kleiner wird. Die Einhaltung eines vorgegebenen Zeitrahmens bedeutet auch, dass man sein Projekt sehr gut kennt. Diese Fähigkeit ist sehr nützlich und wichtig!

Folgende Punkte sollte eure Präsentation enthalten:

1. Kurze Einführung
2. Problem und Lösung vorstellen
3. Technologie und Funktion
4. Positive Wirkung erläutern
5. Wie habt ihr das Produkt entwickelt?
6. Abschluss und Ziele euer App

Format der Präsentation

Während der Präsentation könnt ihr entweder frei sprechen oder Präsentationsfolien und andere Materialien verwenden, um eure Ideen besser zu veranschaulichen. Zur Übung empfehlen wir euch die Produktpräsentation als Video aufzuzeichnen. So könnt ihr es euch selbst noch einmal anschauen und sehen, wie ihr rüberkommt. Außerdem könnt ihr so leichter Feedback von anderen Personen einholen und ihr habt eine Erinnerung an ein tolles Projekt, das ihr gerade macht. Dabei könnt ihr entscheiden, ob ihr euch selbst während der Präsentation filmen wollt, oder auch andere Videoausschnitte und Bilder verwenden wollt. Da ihr nun wisst, was alles nötig ist, könnt ihr euch einmal Gedanken zu machen, wie ihr das Ganze als Präsentation zusammenstellen könnt.

Das Storyboard

Mit einem Storyboard (Ablaufplan/Drehbuch) könnt ihr die Inhalte und die Gestaltung eurer Präsentation planen. Es ist ein gutes Mittel, die Bildideen aus eurem Kopf für euch und andere sichtbar zu machen. Das hilft euch auch später, weil ihr die Szenen schon komplett durchdacht habt und euch so besser organisieren könnt. Wir zeigen euch ein einfaches Beispiel eines Storyboards. Dazu genügt eine Skizze der wichtigsten Szenen, mit dem Text/Geräuschen dazu und wichtigen Hinweisen/Notizen.

Skizze/Bild/Folie/Inhalte	Text/Tonspur	Dauer	Notizen
	Man hört Geräusche von Videos – dann ein Signalgeräusch einer App	Ca. 10 Sekunden	Person liegt gelangweilt auf der Couch, schaut Videos, als plötzlich eine Benachrichtigung auf dem Handy erscheint.

Skizze/Bild/Folie/Inhalte	Text/Tonspur	Dauer	Notizen
	Voice Over: „Biete mein Gemüse zur Abholung an“.	Ca. 7 Sekunden	Handy wird ins Bild gehoben, Text erscheint auf dem Bildschirm mit Voice Over.

Aktivität: Entwickelt das Storyboard für eure Präsentation

Ihr braucht:

- Stifte

Mithilfe des Storyboards könnt ihr ein aussagekräftiges Konzept für eure App-Vorstellung entwickeln und gleichzeitig eine überzeugende Geschichte erzählen.

Geht in diesen Schritten vor:

1 a) Zeichnet in die Kästchen auf dem Arbeitsblatt eure Bildideen für die einzelnen Szenen, die ihr im Video aufnehmen wollt oder eure Ideen für Präsentationsfolien oder -materialien.

b) Schreibt daneben die Texte, die gesprochen werden und/oder auch die Geräusche, die zu hören sein sollen.

c) In einem weiteren Kästchen findet ihr Platz für zusätzliche Informationen zu den Szenen:

- Welche Requisiten (Ausstattung, z. B. Dekoration) werden benötigt?
- Gibt es Hinweise zu Beleuchtung, Geräuschen, Hintergrundmusik?
- Schreibt euch alles auf, was ihr bei der Präsentation beachten müsst.

Skizze/Bild/Folie/Inhalte	Text/Tonspur	Dauer	Notizen

Skizze/Bild/Folie/Inhalte	Text/Tonspur	Dauer	Notizen

Reflexion

Ihr habt den Plan für eure Produktpräsentation gemacht und ein Storyboard erarbeitet. Diese Schritte sind sehr wichtig, um für die Präsentation oder die Aufnahme eures Videos bereit zu sein. In den nächsten Lektionen geht es um die Videoaufnahme eurer Produktpräsentation. Habt ihr euch schon entschieden, ob ihr ein Video aufnehmen wollt oder ihr es lieber persönlich präsentieren wollt, zum Beispiel vor eurer Klasse? Wenn ihr euch entschließt, dass ihr kein Video machen wollt, dann übt eure Präsentation vor anderen Personen, die euch Feedback geben können.

VIDEO AUFNEHMEN

ⓘ In dieser Lektion ...

- nehmt ihr eure Produktpräsentation als Video auf, um eure Idee zu zeigen
- macht ihr Screenshots und/oder Videoaufnahmen von eurer App und fügt sie eurem Video hinzu

Schlüsselbegriffe

- Produktion
- B-Roll-Filmmaterial
- Screenshots
- Bildschirmaufnahme

Licht, Kamera, Action!

Seid ihr bereit, euer Video zu produzieren? Macht euch erstmal noch keine Gedanken um den Schnitt. Sammelt alle Aufnahmen, die ihr in eurem Video haben möchtet und macht Screenshots oder Videos von eurer App. Denn so kann das Publikum sehen, wie sie funktioniert. Zusätzlich zu eurem Hauptmaterial könnt ihr auch B-Roll Material, also alternatives Material, aufnehmen. Das könnt ihr später in euer Video einbauen.

Ein Beispiel: „Im Video wird eine belebte Straßenszene gezeigt. Währenddessen spricht eine Person, die nicht gezeigt wird, über ein Verkehrsproblem."

Tipps für die Umsetzung:

- Sprecht laut und deutlich.
- Haltet Blickkontakt mit der Kamera.
- Nutzt zum Beispeil auch Bildmaterial oder Animationen. Hier findet ihr eine nützliche Website, die euch zeigt, wie ihr Screenshots erstellen könnt.
- Macht mehrere Aufnahmen einer Szene.
- Nutzt viel Licht.
- Dreht im Querformat.
- Achtet auf den Schutz der Privatsphäre.
- Baut schauspielerische Elemente ein, zum Beispiel wie Menschen eure App in der Realität nutzen.

☞ **Hinweis:** Programme für Bildschirmaufnahmen, die ihr verwenden könnt, sind zum Beispiel: Quicktime Player auf Mac, VLC Player auf Windows, Bildschirmaufnahme auf dem iPhone, Microsoft Stream auf Windows, Game Bar auf Windows, Bildschirmrekorder auf Android.

Aktivität: Euer Video aufnehmen

Ihr braucht:

- Computer oder Tablet
- Kamera oder Smartphone
- Mikrofon
- Evtl. zusätzliches Licht
- Storyboard

Jetzt könnt ihr mit der Videoaufnahme beginnen! Dazu braucht ihr ein Gerät zum Filmen (z. B. ein Smartphone mit Kamerafunktion) und wenn möglich ein Mikrofon. Vereinbart Termine mit allen Menschen, die ihr filmen wollt.

Hier sind ein paar Empfehlungen für euer Video:

- Alle Teammitglieder sollten während des Videos gezeigt werden und sprechen.
- Das Video sollte maximal vier bis sechs Minuten lang sein.
- Schützt euch und die Privatsphäre anderer: Zeigt keine Namen (also z. B. nur Vorname oder Spitzname), Adressen, Telefonnummern oder Accounts, durch die man eine Person erkennen könnte. Ihr müsst die Zustimmung aller Personen haben, die ihr in eurem Video zeigt.
- Kennzeichnet fremde Bilder und Informationen: Wenn ihr Bilder oder Informationen nutzt, die von anderen Personen stammen, müsst ihr sie um Erlaubnis bitten und/oder sie erwähnen.

LINK

Hier könnt ihr mehr über Copyright (Urheberrecht), faire Nutzung und öffentliche Domains erfahren.

Reflexion

Nun ist euer Video fast fertig. In der nächsten Lektion erhaltet ihr Tipps zur Bearbeitung eures Videos. Auf welche Herausforderungen seid ihr beim Filmen gestoßen? Welche Aufnahmetechniken waren neu für euch? Welche waren für euer Video am hilfreichsten?

VIDEO FERTIGSTELLEN

In dieser Lektion ...

- bearbeitet ihr euer Video nach, um es ansprechend zu gestalten

Schlüsselbegriffe

- Nachbearbeitung/Post-produktion (Englisch: Post Production)

Der letzte Schritt zur Fertigstellung eures Videos ist die sogenannte Post-Produktion, das heißt die Bearbeitung eures Videos und das Hinzufügen von Effekten, Titeln und Ton. Dadurch könnt ihr euer Video noch ansprechender gestalten. Wahrscheinlich wollt ihr einige Teile herausschneiden und ein paar Screenshots, Untertitel, Erzählungen und vielleicht sogar Spezialeffekte in das Video einbauen. Denkt daran: Ihr habt vier bis sechs Minuten für euer Video.

Bearbeiten (Editieren)

Ihr könnt unterschiedliche Techniken nutzen, um euer Video zu bearbeiten (editieren). Da die Videobearbeitung viel Zeit in Anspruch nehmen kann, solltet ihr die wichtigsten Dinge zuerst machen. Wenn danach noch Zeit ist, könnt ihr zusätzliche Arbeit in die Bearbeitung stecken.

Hier sind ein paar Tipps:

- Seht euch euer gesamtes Filmmaterial an und wählt aus, was in euer Video rein soll.
- Wählt einen Video-Editor (Videobearbeitungs-Tool) aus. Zum Beispiel: Android-Editor: FilmoraGo | Mac- oder iPhone-Editor: iMovie | Windows-Editor: ShotCut.
- Schneidet euer Filmmaterial und entscheidet, wo ihr zum Beispiel Screenshots oder Bildschirmaufnahmen oder Töne einfügen möchtet.
- Fügt Titel, Untertitel, Grafiken und Effekte hinzu.
- Stockfotos sind auch eine Möglichkeit, Bilder einzubinden, kostenlose Stockfotos findet ihr zum Beispiel hier: https://www.pexels.com/
- Das Hinzufügen von Zahlen, Diagrammen oder Statistiken lenkt die Aufmerksamkeit der Zuschauer:innen auf die Informationen. Ihr könnt sie erstellen, indem ihr sie abzeichnet und abfotografiert oder sie digital erstellt.
- Hintergrundmusik kann das Video interessanter machen und sorgt für mehr Spaß beim Anschauen. Hier ist eine Sammlung https://www.youtube.com/audiolibrary/

Aktivität: Euer Video bearbeiten

Ihr braucht:

- Computer oder Tablet
- Eure Filmaufnahmen und Fotos

Plant für diese Aktivität ausreichend Zeit ein. Beginnt zuerst mit den wichtigsten Bearbeitungsschritten, die ihr an eurem Video vornehmen müsst.

Hier sind die Schritte, die ihr in dem Fall tun solltet. Bei den Tipps in der Lektion findet ihr noch weitere Möglichkeiten, wie ihr euer Video bearbeiten und aufpeppen könnt.

Das solltet ihr tun:

1. Wählt euer Filmmaterial aus.
2. Wählt einen Video-Editor aus.
3. Bearbeitet euren Inhalt.
4. Fügt Titel und Effekte hinzu.
5. Fügt Untertitel in einer anderen Sprache hinzu, wenn ihr das möchtet.
6. Gibt es noch etwas, das ihr tun möchtet, um euer Video aufregender zu gestalten? Denkt daran: Ein einfaches Video ist vollkommen in Ordnung und zu viele Spezialeffekte können von eurer Geschichte ablenken.

Tipp // Feedback einholen

Bittet Personen außerhalb eures Teams, sich euer Video anzusehen und hört auf ihr Feedback. Ihr könnt sie Folgendes fragen: Verstehst du, was das Problem ist und wie wir es lösen möchten? Verstehst du, wie die App funktioniert? Lasst sie die App nutzen und fragt sie, ob im Video etwas Wichtiges gefehlt hat. Überarbeitet euer Video, bis ihr mit dem Endprodukt zufrieden seid.

Reflexion

Das Video könnt ihr verwenden, um andere für euer Projekt zu begeistern. Ihr könnt das Video auch als Übung sehen, wenn ihr eure Idee anderen Leuten persönlich vorstellen wollt. Denn jetzt habt ihr eine Geschichte, die ihr zu eurer Idee erzählen könnt. Wie hat sich euer Video von der Erstellung des Storyboards bis zur endgültigen Version verändert? Vor welchen Herausforderungen standet ihr während der Bearbeitung?

CODING

EINFÜHRUNG – APP-BUILDER KENNENLERNEN

In dieser Lektion ...

- erfahrt ihr mehr über App-Builder und Programmiersprachen
- lernt ihr den App-Builder Thunkable kennen
- programmiert ihr eure erste App mit Thunkable

Schlüsselbegriffe

- App-Builder
- Code
- Drag and Drop
- Programmiersprache
- Anleitung (Englisch: Tutorial)

App-Builder und Programmiersprachen

Programmiersprachen geben euch die Möglichkeit, mit einem Computer zu sprechen und ihm zu sagen, was er tun soll. Bevor ihr beginnt, müsst ihr euch entscheiden, mit welcher Programmiersprache ihr eure App programmieren möchtet.

Tipp // Stellt Fragen

Sucht für alle Wörter, die ihr nicht kennt, im Internet die Erklärung. Professionelle Programmierer:innen schlagen auch dauernd Sachen im Internet nach, die sie nicht wissen. Fragen zu haben ist gut!

In dieser Lektion lernt ihr App-Builder und Programmiersprachen kennen. Das Handbuch enthält Beispielcodes für den App-Builder Thunkable.

Tipp // Drag and Drop

Alle App-Builder verfügen in der Regel über eine sogenannte „Drag and Drop"-Schnittstelle, mit der ihr Apps erstellen könnt, ohne dass ihr Code eingeben müsst. Auch wenn der Code in App-Buildern einfach aussieht, werdet ihr wertvolle Programmiergrundlagen lernen.

Bevor ihr mit dem Programmieren eurer App beginnen könnt, zeigen euch die nächsten beiden Aktivitäten, wie ihr Thunkable einrichtet.

VIDEO

Schaut euch vorher ein allgemeines Einstiegsvideo zum Thema Coding an.

VIDEO

Und hier findet ihr ein Interview mit einer richtigen Programmiererin. Sie erzählt euch, wie sie mit dem Coding angefangen hat und gibt euch Tipps zum Programmieren.

Aktivität: Thunkable einrichten und benutzen

Ihr braucht:

- Computer oder Tablet
- Internet
- Mail-Konto: Ihr braucht ein Mail-Konto, um euch bei Thunkable anzumelden
- Optional: mobiles Endgerät (z. B. ein Handy oder ein Tablet)

Hinweis: Falls ihr keine E-Mail-Adresse habt, könnt ihr auch eure Eltern oder Erziehungsberechtigen fragen, ob ihr deren E-Mail-Adresse nutzen könnt.

Teil 1: Erstellt ein Konto und euer erstes Projekt auf Thunkable

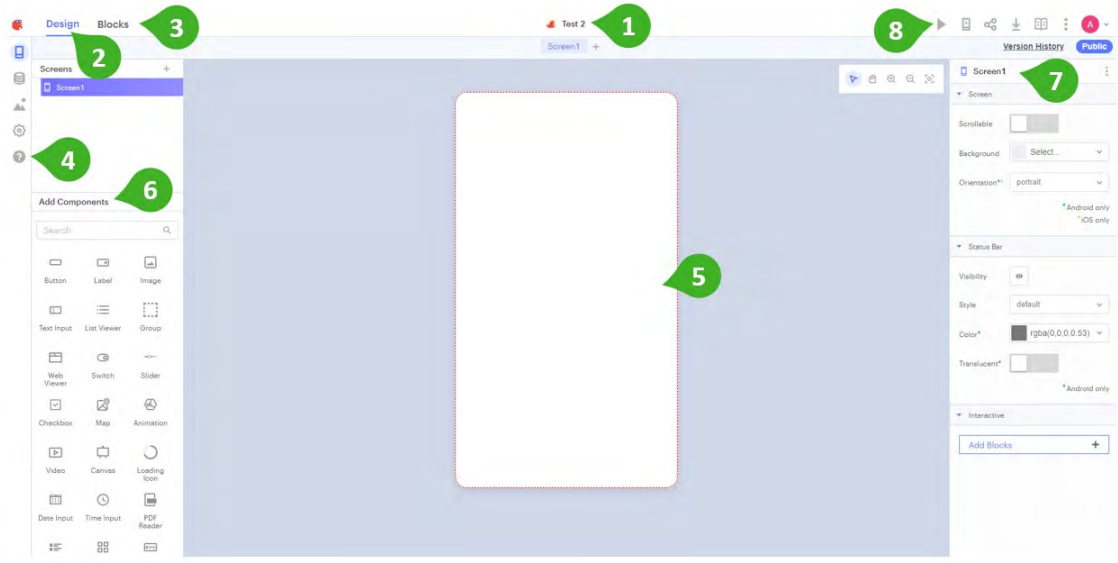
Thunkable ist ein plattformübergreifender App-Builder. Plattformübergreifend bedeutet: Jede App, die ihr auf Thunkable erstellt, funktioniert sowohl für Android- als auch für iOS-Geräte. Erstellt zuerst ein Konto bei https://thunkable.com und richtet euer Gerät mit den folgenden Schritten ein:

1. Klickt oben rechts auf den Button “Sign Up”, um ein Konto zu erstellen.
2. Es öffnet sich eine Seite, auf der ihr auswählen könnt, mit welcher Mailadresse oder Account (z. B. Google, Apple) ihr euch anmelden wollt. Wählt eine davon aus und gebt eure Mailadresse ein.
3. Klickt dann auf den Log-In Link in der Mail.

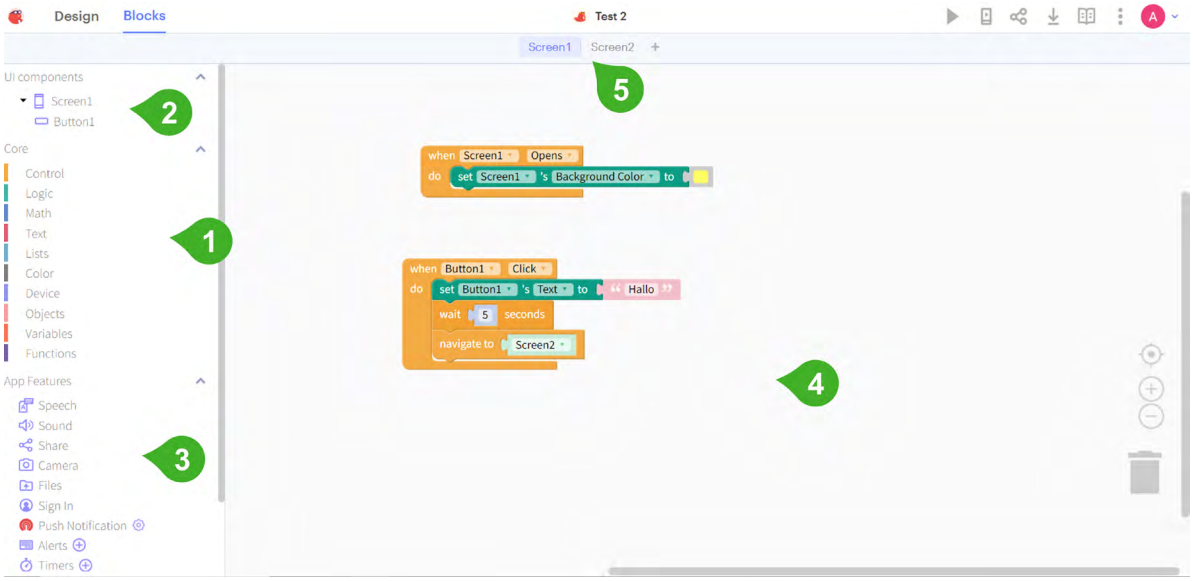
Wenn ihr auf der Startseite in eurem Thunkable Account seid, klickt auf die Fläche „Create New App“. Benennt euer Projekt und stellt es auf „Privat“ oder „Öffentlich“. Öffentliche Projekte sind für alle sichtbar. Private Projekte sind nur für ihre Ersteller:innen sichtbar. Für diese Option braucht ihr allerdings einen besonderen Account (Upgrade), was Geld kostet.

Teil 2: Lernt Thunkable kennen und programmiert eure erste App

Lernt nun die Oberfläche von Thunkable kennen. Thunkable ist in zwei Bereiche aufgeteilt, den Design- und den Block-Bereich. Hier ist ein Überblick über beide Bereiche und wo ihr was findet.



1. Klickt hier, um euer Projekt umzubenennen.
2. Design-Bereich: Thunkable hat zwei Bereiche, die ihr zur Programmierung verwendet. Im Design-Bereich gestaltet ihr die Benutzeroberfläche, also den Teil eurer App, den eure Nutzer:innen sehen und mit dem sie interagieren, wie zum Beispiel Textfelder oder Schaltflächen (Englisch: Buttons).
3. Block-Bereich: Das ist der zweite Bereich. Hier findet das Programmieren statt, indem ihr die Programmierblöcke mittels Drag and Drop anordnet. Ihr braucht sie, damit alles, was ihr im Design-Bereich gestaltet, auch so funktioniert, wie ihr euch das vorstellt.
4. Hilfebereich: Hinter dem Fragezeichen findet ihr den Hilfebereich (Englisch: Help Center) mit vielen Anleitungen (Englisch: Tutorials) als Text oder Video. Diese helfen euch, wenn ihr Probleme habt, euch bei Thunkable zurecht zu finden.
5. Wireframe: Ein Wireframe ist wie eine Aufnahme des Bildschirms eurer App. Hier könnt ihr gestalten, wie eure App aussehen soll, also zum Beispiel welche Farbe der Hintergrund haben soll. Außerdem könnt ihr hier die Komponenten anordnen, also wo und wie groß zum Beispiel ein Button oder ein Textfeld auf dem Bildschirm sein soll.
6. Components: Von hier könnt ihr Komponenten, also zum Beispiel ein Textfeld oder Buttons, auswählen und sie mittels Drag and Drop auf den Wireframe platzieren. Ihre Größe könnt ihr anpassen, indem ihr an den Ecken mit der Maus zieht.
7. Properties Panel: Die Größe könnt ihr auch hier noch genauer anpassen. Hier könnt ihr nämlich alle Eigenschaften (Englisch: Properties) eurer Komponenten verändern. Klickt dazu im Wireframe auf die Komponente, die ihr verändern möchtet, und ändert zum Beispiel ihre Größe oder Farbe. Jede Komponente hat andere Eigenschaften, die ihr hier verändern könnt. Bei einem Button könnt ihr zum Beispiel einen Text auf dem Button hinzufügen.
8. Preview: Das ist eine Möglichkeit euch eine Vorschau (Englisch: Preview) eurer App anzeigen zu lassen. Wechselt in diesen Modus, um auszuprobieren was den Nutzer:innen angezeigt wird, wenn sie zum Beispiel auf einen Button klicken. So könnt ihr überprüfen, ob eure App so funktioniert, wie ihr euch das vorstellt. Wenn ihr dann wieder weiterprogrammieren wollt, klickt nochmal auf dieselbe Stelle und wechselt in den Bearbeitungsmodus.



1. Core: Hier findet ihr die allgemeinen Programmierblöcke, die ihr mittels Drag and Drop in euren Arbeitsbereich ziehen könnt. Sie sind in Kategorien geordnet, wie zum Beispiel in Kontrollelemente (Englisch: Control) oder Textelemente.
2. UI Components: Alle Komponenten der Benutzeroberfläche (Englisch: User Interface/UI Components), die ihr im Design-Bereich in eure Wireframes platziert, haben hier spezielle Programmierblöcke. Diese verwendet ihr, um zu programmieren, was jede Komponente tun soll, also zum Beispiel ein Button.
3. App Features: Hier werden alle unsichtbaren Komponenten (Englisch: Non-visible components) angezeigt, also Funktionen, die zwar keine Schaltfläche im Wireframe haben, aber trotzdem zu eurer App gehören. Das kann zum Beispiel ein Alarm, ein Timer oder ein Sensor sein. Die Programmierblöcke könnt ihr genauso in den Arbeitsbereich ziehen wie die anderen.
4. Arbeitsbereich: Hier ist euer Arbeitsbereich, auf den ihr eure Programmierblöcke anordnet. Ihr könnt sie rumschieben und wie Puzzlestücke zusammenstecken. Wenn ihr Programmierblöcke nicht mehr braucht, könnt ihr sie auf die Mülltonne ziehen und löschen.
5. Screens: Wenn ihr mehrere Bildschirme (Englisch: Screens) in eurer App gestaltet habt, also zum Beispiel einen Startbildschirm, Log-In Bildschirm usw., dann könnt ihr hier zwischen den Bildschirmen wechseln. Jeder Bildschirm wird in einem neuen Arbeitsbereich programmiert, weil er vermutlich auch unterschiedliche Komponenten enthält.

VIDEO

Jetzt da ihr einen Überblick habt, klickt in der Symbolleiste auf der linken Seite auf das Fragezeichen und klappt die Tutorial-Liste aus. Folgt dann dem Tutorial „Platform Overview“ und versucht diese Anleitung nachzumachen.

☞ **Hinweis:** In Thunkable ist das meiste auf Englisch, auch Anleitungen (Englisch: Tutorials) und Tipps. Bei Videos auf Youtube könnt ihr die englischen Untertitel anschalten und diese automatisch auf Deutsch übersetzen lassen. Klickt dafür unten rechts im Video auf das vierte Symbol von rechts, dann auf Untertitel, dann auf „Automatisch übersetzen“. Hier wählt ihr Deutsch oder die Sprache, die ihr am besten versteht, aus. Wenn ihr Schwierigkeiten beim Verstehen habt, fragt eure Teammitglieder, eur:e Mentor:in, Eltern, Lehrkräfte oder andere Personen in eurem Umfeld, die gut Englisch sprechen können, um Hilfe.

Die Anleitung hilft euch, eure erste App mit Thunkable zu erstellen und zeigt euch nochmal, wie Thunkable aufgebaut ist. Denkt daran, dass dies eure allererste App ist. Es ist also in Ordnung, falls ihr mal nicht weiterkommt oder mehrere Versuche braucht!

Reflexion

Wie fühlt es sich an, zum ersten Mal etwas Eigenes zu programmieren? Was war neu für euch? Auf welche Herausforderungen seid ihr gestoßen? Wie habt ihr sie überwunden? In den folgenden Lektionen werdet ihr noch viel tiefer in die Programmierung einsteigen und umfangreichere Apps als Übung programmieren – und zum Schluss natürlich eure eigene App!

EINFÜHRUNG – ALGORITHMEN UND PSEUDOCODE

In dieser Lektion ...

- lernt ihr, was Algorithmen und Pseudocode sind

Schlüsselbegriffe

- Algorithmus (Mehrzahl: Algorithmen)
- Pseudocode

VIDEO

Die meisten Kinder, die in Nordamerika aufwachsen, lieben Weißbrot mit Erdnussbutter und Marmelade. Schaut zu, wie Johanna und Evan ihrem Vater Josh Anweisungen geben, ein Brot mit Erdnussbutter und Marmelade zu bestreichen.

Hinweis: Das Video ist auf Englisch. Die Anleitung, wie ihr deutsche Untertitel hinzufügt findet ihr in der Aktivität der Lektion: Thunkable einrichten und benutzen.

Was ist passiert?

Josh verhält sich wie ein Computer: Er tut nichts – es sei denn, es wird ihm in einfachen Schritt-für-Schritt-Befehlen mitgeteilt. Ein Mensch wüsste, dass die Anweisung „Streiche etwas Marmelade auf das Brot" in Wirklichkeit bedeutet, das Marmeladenglas zu öffnen, das Messer hineinzustecken, es wieder herauszuziehen und es zu benutzen, um etwas Marmelade auf das Brot zu streichen. Ein Computer jedoch täte, was Josh tat: Er würde einfach das ganze Marmeladenglas auf das Brot streichen.

Im Gegensatz zu Menschen können Computer nicht von selbst auf irgendetwas schließen oder Vermutungen anstellen. Sie können nur genau das tun, was ihnen gesagt wird. Ein Algorithmus ist eine Reihe von Schritt-für-Schritt-Anweisungen. Um einen Computer dazu zu bringen, etwas zu tun, muss man einen Algorithmus schreiben, den er versteht. Mit diesem Handbuch lernt ihr, wie ihr Algorithmen schreibt, um eine mobile App zu erstellen.

Pseudocode

Pseudocode bedeutet, dass ein Algorithmus in einfacher Sprache und nicht in Code geschrieben wird. „Pseudo" bedeutet „falsch". Ihr könnt euch Pseudocode also als falschen beziehungsweise „Schein-Code" vorstellen. Pseudocode könnt ihr verwenden, um zu planen, was euer Code tun soll. In dieser Lektion üben wir das Schreiben von Pseudocode, damit ihr euren eigenen Code schreiben könnt, wenn ihr eure App programmiert.

Aktivität: Pseudocode schreiben

Ihr braucht:

- Stifte

Ahmt die Denkweise eines Computers nach!
Schreibt einen Algorithmus für eine alltägliche Aktivität. Das kann Fahrradfahren, ein belegtes Brot machen, ein Spiel spielen oder etwas anderes sein. Wenn ihr fertig seid, gebt euren Algorithmus einem Freund oder einer Freundin und lasst die Person genau das tun, was der Algorithmus sagt – genau wie Josh im Video mit dem Erdnussbutterbrot.

Reflexion

Als ihr euren Algorithmus geschrieben habt: Sind die Dinge so gelaufen, wie ihr es erwartet habt? Beschreibt, wodurch sich die Programmiersprache von der Sprache unterscheidet, die ihr im Alltag gewohnt seid. Warum muss eine Programmiersprache auf diese Weise erstellt werden?

GRUNDLAGEN – EREIGNISGESTEUERTE PROGRAMMIERUNG

In dieser Lektion ...

- versteht ihr, was Ereignisse (Englisch: Events) in Apps sind
- erfahrt ihr mehr über ereignisgesteuerte Programmierung

Schlüsselbegriffe

- Benutzeroberfläche
- Ereignis (Englisch: Event)
- Ereignisgesteuerte Programmierung
- Event Handler
- Schaltfläche (Englisch: Button)

Die Programmierung mobiler Apps wird gewöhnlich „ereignisgesteuerte Programmierung" genannt. Dies liegt an den drei Dingen, die dabei eine Rolle spielen: Benutzeroberfläche, Ereignisse (Englisch: Events) und Ereignisbehandlung (Event Handling).

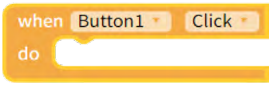
Benutzeroberfläche

Die Benutzeroberfläche der App ist alles, womit Benutzer:innen interagieren, also was sie beeinflussen oder steuern können. Das können zum Beispiel Schaltflächen, Navigationsleisten oder Textfelder sein. Wenn eine Person eure App nutzt, interagiert sie mit ihrer Benutzeroberfläche, indem sie auf Schaltflächen klickt, Text eingibt usw.

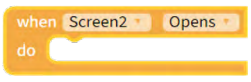
Ereignisse (Englisch: Events)

Ein Ereignis ist etwas, das geschieht. In der Programmierung spricht man von einem Ereignis, wenn etwas geschieht, das die Ausführung des Codes auslöst. Bei mobilen Apps ist ein Ereignis in der Regel etwas, das geschieht wenn ein:e Nutzer:in etwas tut.

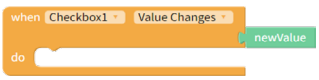
Das bedeutet, dass ein:e Nutzer:in beispielsweise eine Schaltfläche anklickt und daraufhin etwas Bestimmtes passiert. Weitere Ereignisse könnten sein: eine Änderung der Ausrichtung des Telefonbildschirms (durch Drehen des Smartphones) oder die Eingabe von Text in ein Textfeld. Beispiele dafür, wie Ereignisse in Thunkable als Programmierblöcke aussehen sind hier:



Klicken (Englisch: click) auf eine Schaltfläche (Englisch: Button)



Öffnen (Englisch: open) des Bildschirms (Englisch: Screen)



Ändern (Englisch: change), also Aktivieren oder Deaktivieren, eines Kontrollkästchens (Englisch: Check Box)

Ereignisbehandlung (Event Handling)

Ein „Event Handler" ist das, was der Code tut, wenn ein Ereignis eintritt. Ihr selbst könnt mit euer Programmierung bestimmen, was die App beim Eintreten eines Ereignisses machen soll (z. B., wenn eine Person auf eine Schaltfläche klickt). Ihr könnt also der App mit einer Schritt-für-Schritt-Anweisung (dem Algorithmus) sagen, was sie dann tun soll.

Aktivität: Tutorials für Einsteiger:innen

Ihr braucht:

- Computer oder Tablet
- Internet

Jetzt könnt ihr alles üben, was ihr bisher gelernt habt. Erstellt eine App, in der sich jemand mindestens drei verschiedene Bilder anzeigen lassen kann. Die App soll am Ende eine Galerie zeigen, in der die Nutzer:innen jedes Bild nacheinander anschauen können.

LINK

Öffnet diesen Link und klickt oben rechts entweder auf “Remix” oder auf die drei Punkte und wählt “Projekt kopieren” (Englisch: Duplicate project). Hier sind schon die Bilder, die ihr gleich für die Galerie braucht, hochgeladen. Versucht nun, eine Galerie als App zu programmieren.

VIDEO

Als Hilfe könnt ihr euch dieses Tutorial anschauen und es Schritt für Schritt nachprogrammieren.

Hinweis: Das Video ist auf Englisch. Die Anleitung, wie ihr deutsche Untertitel hinzufügt findet ihr in der Aktivität der Lektion: Thunkable einrichten und benutzen.

LINK

Wenn ihr nicht weiterkommt, könnt ihr hier einen Blick auf die Lösung werfen. Probiert es aber erst selbst aus, bevor ihr nachschaut!

Reflexion

Was habt ihr in dieser App zu eurer Benutzeroberfläche hinzugefügt? Welche Arten von Ereignissen gab es in dieser App?

GRUNDLAGEN – DATEN UND FUNKTIONEN

In dieser Lektion ...

- lernt ihr, was Daten sind
- lernt ihr verschiedene Funktionen kennen, die ihr in Thunkable verwenden könnt

Schlüsselbegriffe

- Daten
- Zeichenkette (Englisch: String)
- Funktion
- Ausgabe
- Eigenschaften (Englisch: Properties)
- Zahl
- Boolean
- Eingabe
- Komponenten (Englisch: Components)
- Etikett (Englisch: Label)

Bis jetzt habt ihr gelernt, dass ihr eurer App sagen müsst, dass sie bestimmte Dinge tun soll, wenn ein Ereignis eintritt. Um den Computer dazu zu bringen, etwas zu tun, müsst ihr ihm das auf eine Weise sagen, die ein Computer versteht. Dafür braucht ihr sogenannte Daten.

Ihr habt das Wort Daten wahrscheinlich schon einmal gehört. In der Informatik sind Daten die Informationen, die eure App verstehen und nutzen kann.

Drei grundlegende Arten von Daten solltet ihr kennen:

Zahlen: ein Datentyp, der eine Zahl ist.

Zeichenketten: ein Datentyp, der Zeichen verwendet. Ihr könnt euch die Zeichenketten als Text vorstellen, zum Beispieldie Wörter „Hallo“ und „Welt“.
Booleans: eine Art von Information, die einer von zwei Werten

sein kann: entweder „wahr“ (= true oder ja) oder „falsch“ (= false oder nein). Booleans werden zum Beispiel bei bedingten Anweisungen eingesetzt. Darüber lernst du noch mehr in der Coding-Lektion “If/Else - Bedingte Anweisungen”.

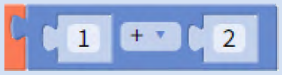
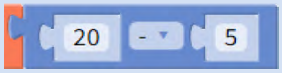
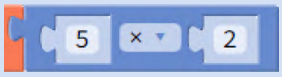
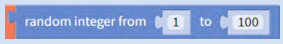
Funktionen

Ihr könnt eurer App über eine sogenannte Funktion mitteilen, was sie tun soll. Eine Funktion ist ein Code-Block, der eine bestimmte Aufgabe erfüllt. Funktionen nehmen eine Eingabe an und erzeugen eine Ausgabe. Hier sind ein paar Funktionen, die ihr in Thunkable verwenden könnt:

Mathematische Funktionen

Mathematische Funktionen nehmen Zahlen, verrechnen sie und zeigen euch dann die Antwort. Es gibt viele verschiedene mathematische Funktionen, die ihr verwenden könnt.

Hier sind einige Beispiele:

Funktion	Eingabe	Ausgabe	Beispiele
Hinzufügen von	zwei oder mehr Zahlen	die Summe dieser Zahlen	
Subtrahieren	zwei Zahlen	die Differenz dieser Zahlen	
Vervielfachen	zwei oder mehr Zahlen	das Produkt dieser Zahlen	
Zufällige ganze Zahlen (Englisch: random integer)	zwei Zahlen	eine Zufallszahl, die zwischen diesen beiden Zahlen liegt	

Hier sind Beispiele, wann ihr so eine mathematische Funktion verwenden könnt:

- Punkte in einem Spiel zählen
- Hinzufügen von Geld auf einem Bankkonto

Text-Funktionen

Eine Zeichenkette ist ein Informationstyp, der Zeichen verwendet. Sie wird auf Englisch String genannt. Ihr könnt euch die Strings als Text vorstellen, wie zum Beispiel die Wörter „Hallo“ und „Welt“. Es gibt auch viele Funktionen, die Text verwenden. Hier sind Beispiele:

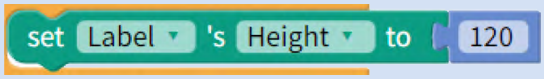
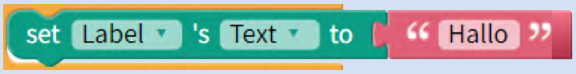
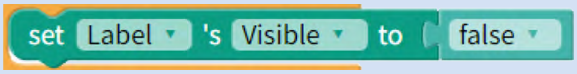
Funktion	Eingabe	Ausgabe	Beispiele
Verbinden (Englisch: join)	zwei oder mehr Zeichenketten	die Summe dieser Zahlen	
Länge (Englisch: Length)	eine Zeichenkette	die Anzahl der Zeichen in dieser Zeichenkette	

Beispiele für Situationen, in denen ihr Text-Funktionen verwenden könnt:

- das Anzeigen eines Wortes
- die Überprüfung, wie lang ein Passwort ist
- die Überprüfung, ob jemand das richtige Passwort verwendet

Funktionen der Komponenten

Alle Komponenten (Bausteine wie z. B. Buttons oder Bilder) in Thunkable haben Eigenschaften, die geändert werden können. Ihr seht alle Eigenschaften der unterschiedlichen Komponenten, wenn ihr sie in dem Design-Bereich anklickt. Außerdem könnt ihr die Eigenschaften auf im Block-Bereich mit Hilfe von Funktionen, also Programmierblöcken, ändern. Beispiele dafür sind hier:

Eigenschaft	Was besagt oder tut diese Eigenschaft?	Funktion bzw. Programmierblock zum Ändern
Höhe (Englisch: Height)	wie hoch das Label (Englisch für Etikett) auf dem Screen ist	
Schriftgröße (Englisch: Font-Size)	die Schriftgröße des Textes im Label	
Text (Englisch: Text)	welchen Text das Label anzeigt	
Sichtbarkeit (Englisch: visible)	ob das Label auf dem Screen zu sehen ist oder nicht	

Aktivität: Bildeigenschaften verwenden

Ihr braucht:

- Computer oder Tablet
- Internet

LINK

Jetzt könnt ihr euer Wissen über Funktionen direkt anwenden. Geht zurück zu eurer App mit der Bildergalerie. Hier ist der Start-Code, falls ihr ihn nochmal braucht.

Ihr könnt auch mit den Eigenschaften eurer Bilder spielen, zum Beispiel mit der Größe oder der Position. Schafft außerdem noch eine Möglichkeit, dass gar kein Bild angezeigt wird, wenn ihr auf eine vierte Schaltfläche mit der Bezeichnung „nichts“ (Englisch: none) klickt.

Hinweis: Ihr müsst hier die Boolean-Blöcke unter “Logic” „true“ und „false“ verwenden.

LINK

Seht euch anschließend die Thunkable-Lösung an.

Reflexion

Welche Eigenschaft eures Bildes habt ihr in der Aktivität verändert, sodass kein Bild angezeigt wurde? Habt ihr in dieser Lektion etwas gelernt, das ihr in eurer App verwenden möchtet?

GRUNDLAGEN – VARIABLEN

In dieser Lektion ...

- lernt ihr, was Variablen sind und wie man sie verwendet

Schlüsselbegriffe

- Variablen

Einer der wichtigsten Begriffe beim Programmieren ist „Variable“. Eine Variable ist ein Name für eine Dateneinheit, die sich ändern kann. So kann man es sich gut merken: Eine Variable kann variieren, also ihren Wert verändern. Es gibt viele alltägliche Informationen, die für euch wichtig sein können und die ihren Wert verändern.

Hier sind ein paar Beispiele:
- euer Alter
- das Wetter
- das Datum

Stellt euch die Variable als Schachtel vor. In der Schachtel könnt ihr Informationen ablegen und abrufen. Wie zum Beispiel eine Postkartensammlung. Wenn ihr eine Postkarte aus der Schachtel lesen wollt, könnt ihr die Schachtel öffnen und eine Karte rausnehmen. Ihr könnt auch neue Karten hineinlegen oder die Karten verändern. Wichtig ist, dass ihr den Namen des Inhalts (hier: Postkarten) auf die Schachtel schreibt. Dieser Name bleibt immer der gleiche – egal was sich darin befindet. Der Inhalt der Schachtel variiert.

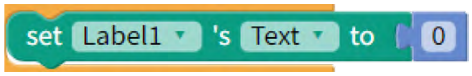
Für eure App sind Variablen immer dann nützlich, wenn die App sich an bestimmte Informationen „erinnern“ soll und wenn diese Informationen sich ändern könnten.

Zum Beispiel:
- die Antwort einer Person auf eine Quizfrage
- der Fortschritt von Spieler:innen in einem Spiel
- Dinge, die sich beim Online-Shopping im Warenkorb von jemandem befinden

Genau wie die Schachtel brauchen eure Variablen dafür zwei Dinge:
1. einen Namen, damit eure App sie finden kann
2. Informationen, die darin gespeichert werden

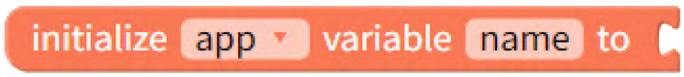
Ein Beispiel: Wie ihr Variablen verwendet, um den Spielstand in einem Spiel zu speichern.

Stellt euch vor, ihr entwickelt ein Spiel, bei dem die Spieler:innen Punkte sammeln und auch wieder verlieren können. Ihr möchtet, dass die Spieler:innen ihren Punktestand sehen können. Also gebt ihr den Punktestand in ein Etikett (Englisch: Label) ein. Zu Beginn des Spiels haben die Spieler:innen null Punkte, also fügt ihr dem Spiel einen Block wie diesen hinzu:



Die Punktzahl der Spieler:innen ändert sich jedoch im Laufe des Spiels. Jedes Mal, wenn sie Punkte gewinnen oder verlieren, müsst ihr das Textfeld aktualisieren. Wenn mehr Punkte hinzukommen, fügt ihr einen Block wie diesen hinzu:

Das funktioniert zwar, aber was macht ihr, wenn die Spieler:innen wieder 10 Punkte gewinnen? Dann müsst ihr den Text erneut aktualisieren, um daraus 20 zu machen. Was ist dann, wenn die Spieler:innen 5 Punkte verlieren? Dann müsst ihr die Punktzahl auf 15 ändern – und so weiter. Das kann sehr schnell verwirrend werden.
In Situationen wie dieser könnt ihr eine Variable erstellen, um den Spielstand zu verfolgen. Ihr könnt eine Variable namens Punktestand erstellen, die immer den Wert des aktuellen Punktestandes der Spieler:innen anzeigt. Während sie das Spiel spielen, könnt ihr mit Hilfe der Variable Punkte addieren und vom Punktestand abziehen. Ihr müsst euch also nicht darum kümmern, den Überblick über den aktuellen Punktestand zu behalten.
Zieht folgenden Block aus dem Variablen-Abschnitt in eure Arbeitsfläche:



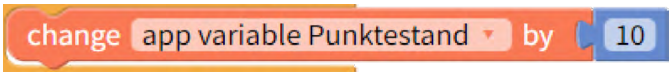
Dieser Block teilt der App mit, dass es eine Variable geben wird, sobald die App startet. Er fordert euch auf, für die Variable einen Namen und einen Wert festzulegen (Englisch: initialize). Ändert den Namen der Variable in "Punktestand" um. So sähe der Block für ein Spiel aus, bei dem der Punktestand der Spieler:innen bei „null" beginnt:

Mit diesem Block teilt ihr der App also mit, dass es ein Datenelement namens "Punktestand" geben wird, das sich während des Spiels ändert. Jedes Mal, wenn die App startet, erstellt (Englisch: initialize) sie eine Variable namens Punktestand und setzt sie auf „null".
Jetzt könnt ihr eure Punktestand-Variable anzeigen lassen, indem ihr das Label (Etikett) als Punktestand "app variable Punktestand" setzt (Englisch: set):

> **Hinweis:** Beim Programmieren ist es sehr wichtig, alles immer eindeutig zu benennen. Zum Beispiel die Variablen, Textfelder oder auch Buttons. Thunkable hilft euch dabei und nummeriert für euch automatisch die Buttons, wenn ihr mehrere habt. Bei den Variablen fügt Thunkable immer automatisch die Bezeichnung "app variable" vor dem Namen der Variable ein.

Mit „set" könnt ihr einen neuen Wert für die Punktzahl der Spieler:innen festlegen. Ihr könnt die App anweisen, das jedes Mal zu tun, wenn sie Punkte sammeln. In diesem Beispiel wird die Punktezahl um 10 Punkte erhöht:

Oder ihr könnt die Punktezahl mit “change” um 10 Punkte so verändern:



Damit weist ihr eure App an, den Spielstand auf den jetzigen Wert zu ändern und 10 hinzuzufügen. Wenn die aktuelle Punktzahl null ist, ist die neue Punktzahl 10. Wenn die aktuelle Punktzahl 25 ist, ist die neue Punktzahl 35. Jetzt müsst ihr euch nicht mehr darum kümmern, den Punktestand jedes Mal zuzuweisen, wenn die Spieler:innen Punkte gewinnen oder verlieren. Die App nimmt einfach den aktuellen Punktestand und fügt ihm 10 hinzu.

Variablen speichern

Vielleicht fällt euch auf, dass ihr statt “App” auch “stored” oder “Cloud” auswählen könnt. Wenn ihr “app” auswählt, sind die Werte der Variablen während der Nutzung eurer App gespeichert. Sobald ihr die App schließt und neu öffnet, wird die Variable neu festgelegt und ihr müsst die Werte neu vergeben.

“Stored” (Englisch für gespeichert) bedeutet, dass sich die App auch die Werte der Variable merkt und sich beim nächsten Öffnen der App daran erinnert. Das wäre zum Beispiel hilfreich, wenn ihr den Punktestand auch speichern möchtet, wenn die App in der Zwischenzeit geschlossen und neu geöffnet wird.

Bei “App” und bei “stored” werden die Daten nur auf dem eigenen Smartphone gespeichert, das heißt, wenn die App von einer anderen Person mit einem anderen Smartphone verwendet wird, kann die Person nur ihre eigenen Daten abrufen.

Bei “Cloud” ist das anders: Hier werden eure Werte in einer Cloud gespeichert und können auch von anderen Personen abgerufen werden. Bei einem Spiel ist das zum Beispiel eine Liste der höchsten Punktestände von allen, die das Spiel verwendet haben.

> 👉 **Hinweis:** Es ist nicht schlimm, wenn die Variablen euch immer noch ein bisschen verwirren! Sie sind nicht so leicht zu verstehen. Der beste Weg ist, zu üben und es immer wieder neu zu probieren. Ihr werdet die Variablen auch in den nächsten Coding-Lektionen wieder benutzen und dadurch vertrauter damit werden.

Aktivität: Zähl-App

Ihr braucht:

- Computer oder Tablet
- Internet

LINK

Jetzt könnt ihr die Verwendung von Variablen üben. Hier ist eine App, die es euch ermöglicht, in 1er, 5er und 10er Schritten zu zählen.

Aber Achtung! Diese App ist unvollständig! Das Problem ist, dass sich der Zähler in der App nicht zurücksetzen lässt. Repariert die App, sodass ihr den Zähler zurücksetzen und wieder bei „null“ anfangen könnt. Was müsst ihr dafür zur App hinzufügen? Braucht es eine weitere Komponente/ Button? Welche Funktion muss dieser Button haben?

LINK

Schaut hier im Lösungscode nach, wenn ihr selbst nicht auf die Lösung kommt.

Reflexion

Welche anderen Variablen in eurem Alltag fallen euch ein? Wie könnt ihr Variablen in eurer App verwenden?

GRUNDLAGEN – LISTEN

In dieser Lektion ...

- organisiert ihr Daten in Listen
- verwendet ihr Listen in einer App

Schlüsselbegriffe

- Feld (Englisch: Array)
- Index
- Listen

Listen und Felder

Wenn ihr eure App erstellt, benötigt ihr vielleicht eine Möglichkeit, Daten und Informationen zu organisieren. Glücklicherweise bietet die Informatik Möglichkeiten, Daten so zu organisieren, dass ihr sie leicht finden und verwenden könnt. Fällt euch ein Beispiel dafür ein, wie ihr im Alltag Informationen und Gegenstände organisiert?

Hier sind ein paar Beispiele:

- die Telefonnummer einer Freundin in eine Kontaktliste aufnehmen
- Hausaufgaben in einem Kalender oder Hausaufgabenheft notieren
- eine Einkaufsliste schreiben
- Kleidung in einen Schrank packen

In Thunkable könnt ihr eine Liste verwenden, um Daten zu organisieren. Listen können mehrere Daten enthalten und es ist einfach, Daten aus Listen abzufragen. Bestimmt habt ihr schon einmal eine To-Do-Liste oder eine Einkaufsliste geschrieben. Beim Programmieren sind Listen sehr ähnlich.

Name der Liste: Einkauf

- Äpfel
- Bananen
- Orangen

Listen sind nützlich, um eine große Menge Informationen zu speichern. Jedes Element, also jeder Stichpunkt (Englisch: Item), in einer Liste hat einen Index. Ein Index ist eine Zahl, welche die Position des Elements in der Liste angibt. Das erste Element in einer Liste hat einen Index von 1, das zweite einen Index von 2 und so weiter. Hier ist ein Beispiel:

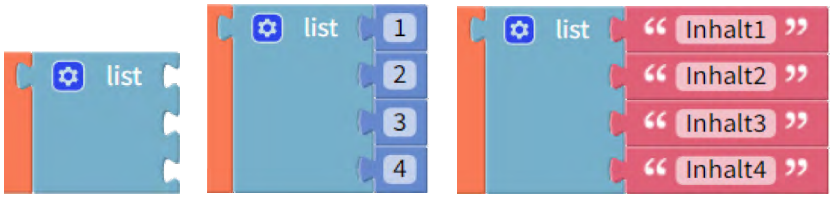
Name der Liste: Einkauf

- Äpfel (Index = 1)
- Bananen (Index = 2)
- Orangen (Index = 3)

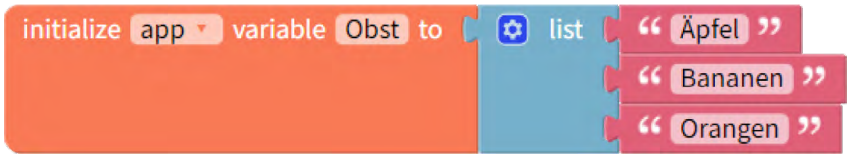
Hinweis: In vielen Programmiersprachen werden Listen als Felder (Englisch: Arrays) bezeichnet. Indizes (das ist die Mehrzahl von Index) beginnen bei 0, nicht bei 1.

Listen in Thunkable verwenden

In Thunkable könnt ihr auf die Blöcke gehen und „Listen“ (Englisch: Lists) auswählen. Listen sehen wie folgt aus:



So könnt ihr eine Liste von Obst für euren Einkauf erstellen:



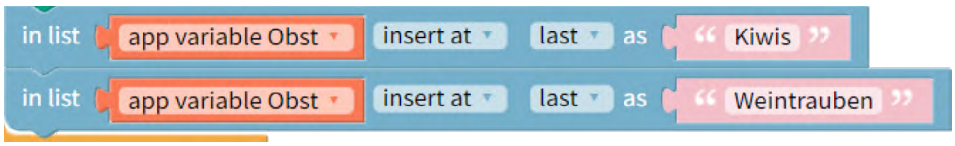
Eure App versteht diese Informationen so:

Name der Liste: Obst

- Äpfel (Index = 1)
- Bananen (Index = 2)
- Orangen (Index = 3)

Ihr könnt eine Sache aus einer Liste entnehmen, ohne gleich die ganze Liste zu verwenden. Wollt ihr statt der gesamten Liste nur die Zeichenfolge „Bananen“ in ein Textfeld eingeben, müsst ihr der App mitteilen, dass sie sich Index 2 in dieser Liste ansehen soll. So könnt ihr „Bananen“ aus der oben gezeigten Liste mit Thunkable in ein Label, hier ein Textfeld, einfügen:

Ihr könnt auch Dinge in Listen hinzufügen, entfernen und ersetzen. Wenn ihr zum Beispiel vergessen habt, „Kiwis“ und „Weintrauben“ zur Obstliste hinzuzufügen, könnt ihr das so nachholen:



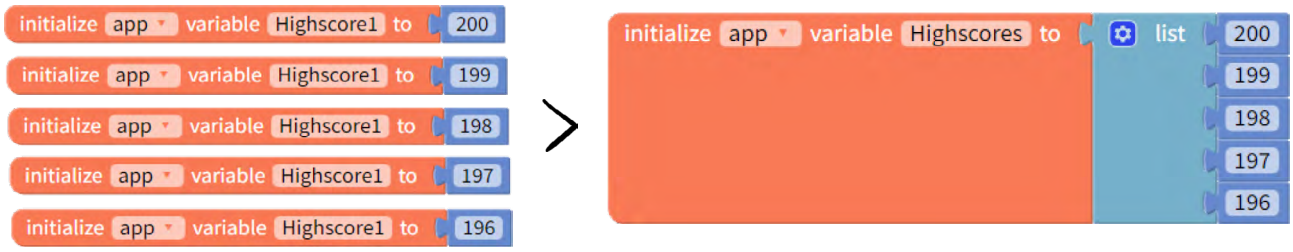
Dieser Programmierblock fügt (Englisch: insert) “Kiwis” und “Weintrauben” an die letzte (Englisch: last) Stelle der Liste ein. Ihr könntet im Block auch eine andere Stelle oder ersetzen (Englisch: set) auswählen. Nun sieht eure Liste so aus:

Name der Liste: Obst

- Äpfel (Index = 1)
- Bananen (Index = 2)
- Orangen (Index = 3)
- Kiwis (Index = 4)
- Weintrauben (Index = 5)

Listen solltet ihr immer dann verwenden, wenn ihr mehrere Informationen habt, die ihr unter dem gleichen Variablennamen aufnehmen möchtet. Wenn ihr zum Beispiel die fünf besten Ergebnisse von Leuten, die ein Spiel gespielt haben (Englisch: Highscore) anzeigen möchtet, könntet ihr fünf Variablen mit Namen wie HighScore1, HighScore2, HighScore3 usw. erstellen, bis ihr zu HighScore5 gelangt.

Der einfachere Weg wäre dieser: Ihr erstellt eine Liste namens „Highscores“, die die fünf Highscores enthält und zeigt sie dann mithilfe ihrer Indizes an. Indem ihr Listen verwendet, könnt ihr auch weitere Highscores leichter und hinzufügen oder ersetzen. Dadurch könnt ihr mit einer Liste viel Zeit sparen.



Aktivität: Orakel-App

Ihr braucht:

- Computer oder Tablet
- Internet

LINK

Jetzt könnt ihr in Thunkable üben, was ihr zum Thema Listen gelernt habt. Diese App funktioniert wie eine Art Orakel: Ihr könnt ihr eine Frage stellen, auf „Fragen“ drücken und sie gibt euch eine Antwort (Englisch: answer).

Im Moment kann die App nur mit „ja“, „nein“ und „nicht sicher“ antworten. Arbeitet an der App weiter, damit das Orakel auch mit „wahrscheinlich“, „vielleicht“ und „ich weiß nicht“ antworten kann.

LINK

Schaut hier im Lösungscode nach, wenn ihr selbst nicht auf die Lösung kommt.

Reflexion

In dieser Lektion habt ihr Listen als eine Möglichkeit zum Speichern und Abrufen von Informationen kennengelernt. Wie könnt ihr Listen in eure App aufnehmen? Und: Wie habt ihr die Orakel-App so verändert, dass sie mehr Antworten geben kann?

GRUNDLAGEN – IF/ELSE – BEDINGTE ANWEISUNGEN

In dieser Lektion ...

- erfahrt ihr etwas über bedingte Anweisungen (Englisch: Conditional Statements) und lernt, wie man sie programmiert
- verbessert ihr eure Orakel-App, indem ihr sicherstellt, dass die Nutzer:innen Fragen in das Textfeld eingeben

Schlüsselbegriffe

- Bedingung (Englisch: Condition)
- Bedingte Anweisungen (Englisch: Conditional Statements)
- wenn/sonst (Englisch: if/else)
- Bedingungen

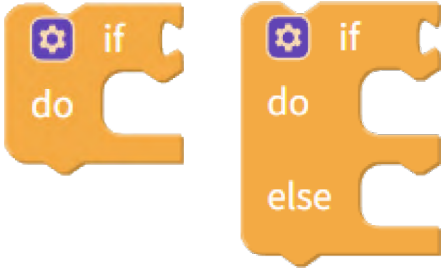
Bedingungen sind für uns Menschen oft die Grundlage von Entscheidungen oder Handlungen. Zum Beispiel: **Wenn** es später nicht regnet, **dann** gehe ich mit dem Hund Gassi. Genauso können wir dem Computer oder einer App Bedingungen eingeben, sodass sie etwas tun. Zum Beispiel: **Wenn** der Punktestand gleich 100 ist, **dann** wird ein Ton abgespielt.

Denkt an eure Orakel-App. Momentan könnt ihr in der App auch ohne eine Frage einzugeben auf „fragen" drücken und ihr bekommt eine Antwort. Besser wäre, wenn die App nur antwortet, wenn eine Frage gestellt wurde. Dafür müsst ihr sie so programmieren, dass sie eine Bedingung (Englisch: Condition) überprüfen kann: Wenn „fragen" gedrückt wird, überprüfe die Bedingung „Text in Textbox". Wenn sie wahr ist (= Ja), dann gib eine Antwort. Wenn sie falsch ist (= nein), dann gib keine Antwort. Mithilfe der Bedingung kann die App also überprüfen, ob die Nutzer:innen einen Text eingegeben haben.

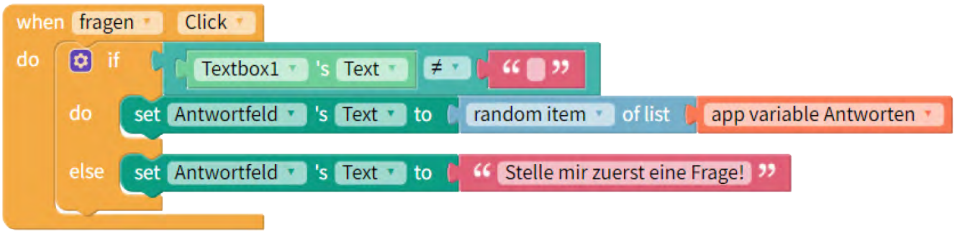
Erinnert ihr euch auch an den Datentyp „Boolean" aus der Coding-Lektion: Daten und Funktionen? Das ist ein Datentyp, der wahr oder falsch sein kann. Wenn eure App eine Bedingung überprüft, gibt sie die Antwort in Form eines „Boolean".

Bedingte Anweisungen

Jetzt wisst ihr, was eine Bedingung ist. Aber wie könnt ihr sie verwenden? Das macht ihr durch bedingte Anweisungen. Diese haben immer einen wenn-Teil (Englisch: if), der der App sagt, was zu tun ist, wenn die Bedingung erfüllt ist. Bedingte Anweisungen haben normalerweise auch einen sonst-Teil (Englisch: else). Er sagt der App, was zu tun ist, wenn die Bedingung nicht erfüllt ist. Wenn ihr den else-Teil weglasst und die Bedingung falsch ist, macht die App gar nichts. So sehen die Blöcke für bedingte Anweisungen in Thunkable aus:



Für das Beispiel eurer Orakel-App, die nur antwortet, wenn eine Frage gestellt wird, sieht der Code in Thunkable so aus:



So funktionieren die Blöcke:
Ihr setzt eine Bedingung neben **if**:
„Wenn der Text in der Textbox nicht gleich leer ist,“

Neben **do** sagt ihr eurer App, was sie tun soll, wenn die Bedingung erfüllt ist:
dann gib einen zufälligen Eintrag (Englisch: random Item) aus der Liste „app variable Antworten“ aus,

Neben else sagt ihr der App, was sie tun soll, wenn die Bedingung nicht erfüllt ist:
sonst, sage „Stelle mir zuerst eine Frage!“.

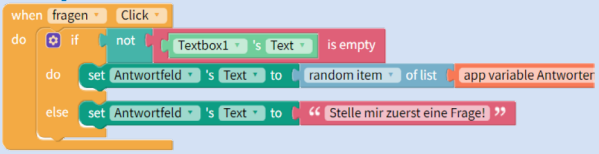
Ist die Bedingung erfüllt, wird nur der Code neben **do** ausgeführt und der gesamte Code neben **else** wird ignoriert. Ist eure Bedingung nicht erfüllt, wird der Code neben **do** ignoriert und der Code neben **else** wird ausgeführt.

Es gibt verschiedene Möglichkeiten, diese bedingte Anweisung zu schreiben. Alle funktionieren, es gibt nicht nur eine richtige Antwort. Hier findet ihr vier weitere Beispiele. Welche findet ihr am besten?

Beispiel 1:

Bedingung: Die Länge der Zeichenfolge im Textfeld ist nicht gleich null.

Beispiel 2:

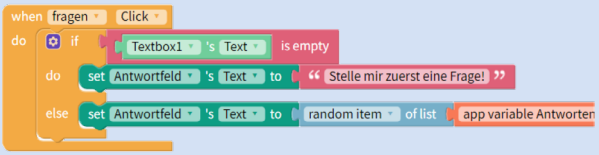


Bedingung: Das Textfeld ist nicht leer.

Beispiel 3:

Bedingung: Die Länge der Zeichenfolge im Textfeld ist gleich Null.
(Beachtet, dass die Blöcke neben **then** und **else** im Vergleich zu Beispiel 1 vertauscht sind.)

Beispiel 4:



Bedingung: Das Textfeld **ist** leer.
(Beachtet, dass die Blöcke neben **then** und **else** im Vergleich zu Beispiel 1 vertauscht sind.)

Aktivität: Bedingte Anweisungen in Thunkable schreiben

Ihr braucht:

- Computer oder Tablet
- Internet

Teil 1: Eine schlauere Orakel-App

Versucht jetzt, selbst den Fehler in eurer App zu beheben, sodass die App nur antwortet, wenn eine Frage gestellt wurde. Wenn keine Frage gestellt wurde, dann soll die App sagen "Stell mir zuerst eine Frage!". Dafür braucht ihr eine bedingte Anweisung. Jedes Mal, wenn jemand „fragen" drückt, soll unser Code die Bedingung überprüfen, ob Text eingegeben wurde.

LINK

Ihr könnt diesen Code als Startcode nehmen.

LINK

Hier ist der Lösungs-Code, falls ihr Hilfe braucht oder euer Ergebnis überprüfen wollt.

Teil 2: Eine verbesserte Zähl-App

Diese Aktivität baut auf der Aktivität aus „Coding-Lektion: Variablen" auf. Ihr habt diese Aktivität noch nicht durchgeführt? Dann versucht es jetzt.

Probiert nun aus, eine eurer älteren Apps, die Zähl-App, zu verbessern und wandelt sie in ein Spiel um: Versucht, den Zählvorgang von vorne beginnen zu lassen, sobald die Nutzer:innen genau 100 erreichen. Wenn sie einen Button antippen, um die Zahl zu vergrößern, solltet ihr sicherstellen, dass die Zahl nicht größer wird als 100. Was denkt ihr: Wo solltet ihr eure bedingte Anweisung einfügen?

LINK

Verwendet diesen Code als Start-Code.

Habt ihr Schwierigkeiten? Hier ist ein Code, der fast fertig ist, aber einen Fehler (Englisch: Bug) enthält. Die App funktioniert manchmal und manchmal nicht. Manchmal sagt sie den Nutzer:innen, dass sie gewonnen haben (Englisch: You win!) und manchmal zählt sie einfach über 100 hinaus. Um das auszuprobieren, drückt zehn Mal auf „Addiere 10" – bis ihr genau auf 100 landet.

Als nächstes drückt ihr zuerst „Addiere 1“ und dann zehn Mal „Addiere 10“. Findet ihr heraus, was nicht stimmt?

LINK

Hier der Code mit Fehler.
Habt ihr herausgefunden, warum der Code nicht immer funktioniert? Es liegt nur an einer Stelle.

LINK

Ihr könnt hier im Lösungscode nachschauen und die Stelle suchen, die anders ist als in dem Code mit Fehler.

Nun habt ihr gesehen, wie viel eine einzige Stelle in eurem Code ausmacht. In dem Fehlercode ist in der Funktion, die die Summe (Englisch: Count) aktualisiert (Englisch: Update), die Bedingung gesetzt, dass die App den Gewinn verkündet, wenn die Summe “=” 100 ist. Wenn jedoch die Summe größer als 100 ist, ist die Bedingung nicht erfüllt und die App zählt weiter, obwohl die Spieler:innen dann auch gewinnen würden. Für die Lösung müsst ihr hier also das “ist gleich” (=) Zeichen mit dem “ist größer gleich” (≥) ersetzen.

Ist euch im Lösungs-Code auch der lila-farbige Programmierbaustein aufgefallen? Hier wurde eine Funktion zum Verändern der Punktezahl (Englisch: Update Count) erstellt. Das ist praktisch, wenn ihr dieselben Schritte mehrmals in eurem Code ausführen wollt. Ihr könnt auch dann selbst eine Funktion erstellen, wie hier die Funktion “updateCount” und sie dann einfach mit einem Block aufrufen, anstatt jedes Mal wieder dasselbe zu programmieren.

Reflexion

Welche bedingten Anweisungen könnt ihr euch noch vorstellen, die ein Computer regelmäßig anwendet? Ein Beispiel: Wenn dieser Link angeklickt wird, öffne die Webseite im Browser.

GRUNDLAGEN – IF / ELSE / ELSE IF – BEDINGTE ANWEISUNGEN

In dieser Lektion ...

- lernt ihr, wie man eine „wenn / sonst / sonst wenn bedingte Anweisung“ (Englisch: if / else / else if Conditional) schreibt
- verbessert ihr eure Orakel-App weiter, indem ihr den Nutzer:innen nicht erlaubt, die gleiche Frage zweimal hintereinander zu stellen

Schlüsselbegriffe

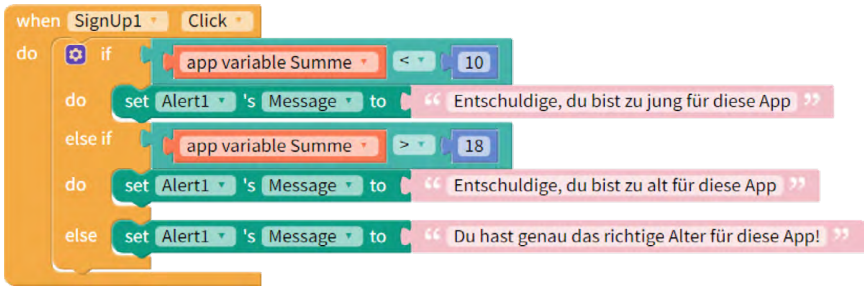
- wenn / sonst / sonst wenn (Englisch: if / else / else if)

If / else / else / if sind bedingte Anweisungen, die mehr als eine Bedingung prüfen und mehr als zwei mögliche Ergebnisse haben. Erst wenn die erste Bedingung falsch ist, wird die zweite Bedingung geprüft. Und wenn die zweite Bedingung ebenfalls falsch ist, dann wird die App auf „else“ gesetzt oder sie tut nichts.

Sehen wir uns ein Beispiel an:
Ihr entwerft eine App, die nur für Nutzer:innen im Alter von 10-18 Jahren gedacht ist. Ihr wollt, dass eure App euren Nutzer:innen sagt, ob sie zu jung oder zu alt sind, um eure App zu nutzen. Ihr habt drei mögliche Ergebnisse:

1. Der/die Nutzer:in ist zu jung
2. Der/die Nutzer:in ist zu alt
3. Der/die Nutzer:in hat das richtige Alter

Ihr lasst die Nutzer:innen das Alter in die App eingeben und speichert es in einer Variable namens Alter). Auf diese Weise könnt ihr später eine if/else/else if -Anweisung verwenden, um das Alter der Nutzer:innen zu überprüfen.



Die erste Bedingung prüft, ob die Nutzer:innen jünger als 10 Jahre sind. Trifft das zu, dann werden sie von der App darauf aufmerksam gemacht, dass sie zu jung sind. Wenn sie nicht jünger als 10 sind, prüft die App, ob sie älter als 18 sind. Ist dies der Fall, werden sie in der App darauf hingewiesen, dass sie zu alt sind. Ist dies nicht der Fall, wird in der App angezeigt, dass sie das richtige Alter haben.

Diese Punkte solltet ihr bei der Verwendung von if/else/else if-Anweisungen beachten:

- Ihr könnt so viele Bedingungen überprüfen lassen, wie ihr wollt.
- Der Code geht die Bedingungen von oben nach unten durch, also stellt die Bedingung, die zuerst getestet werden soll, an den Anfang.
- Der einzige Code, der ausgeführt wird, ist der Code hinter der ersten für wahr befundenen Aussage.
- Wenn keine der Bedingungen wahr ist, wird der Code im anderen („else“) Teil ausgeführt.

Aktivität: Eine noch schlauere Orakel-App

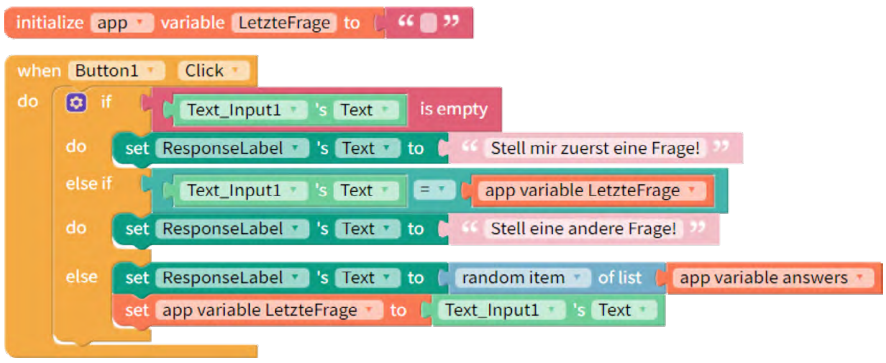
Ihr braucht:

- Computer oder Tablet
- Internet

In dieser Aktivität verbessert ihr eure Orakel-App noch weiter. In der letzten Lektion habt ihr dafür gesorgt, dass die Nutzer:innen einen Text eingeben müssen, bevor die App antwortet. Man kann aber der App zweimal dieselbe Frage stellen und erhält dann unterschiedliche Antworten

Um das zu ändern, könnt ihr zuerst mit einer if / else / else if-Bedingung prüfen, ob die Nutzer:innen einen Text eingegeben haben. Danach, also wenn die Person Text eingegeben hat, könnt ihr prüfen, ob die Frage eine andere ist als die zuletzt gestellte Frage.

Richtet eine Variable namens „LetzteFrage“ ein, die sich an die letzte Frage erinnert, die die Nutzer:innen gestellt haben.



So funktioniert der Code: Jedes Mal, wenn die App eine Antwort gibt, ersetzt der Code die Variable „LetzteFrage“ durch das, was im Textfeld steht. Das passiert im Screenshot oben hinter der Bedingung “else”.

Im nächsten Schritt müsst ihr die App noch dazu bringen, zu überprüfen, ob die „LetzteFrage“ mit dem übereinstimmt, was jetzt in dem Textfeld steht. Dafür müsst ihr eine if / else / else if-Anweisung erstellen. Wenn die dieselbe Frage zweimal gestellt wird, soll die App sagen: “Stell eine andere Frage!”

LINK

Ladet euch diesen Beispielcode herunter und versucht ihn anzupassen.

LINK

Hier findet ihr die Lösung in Thunkable.

Reflexion

Überlegt, ob das Orakel für eure App geeignet oder nicht geeignet ist. Und wieso ist es sinnvoll oder weniger sinnvoll? Was für andere Bedingungen könnten die Orakel-App gut ergänzen?

GRUNDLAGEN – ERWEITERTE LOGIK UND BEDINGUNGEN

In dieser Lektion ...

- lernt ihr, wie ihr bedingte Anweisungen schreibt, die eine Logik verwenden
- verbessert ihr eure Orakel-App, so dass die App nur Fragen beantwortet, die ein Fragezeichen haben.

Schlüsselbegriffe

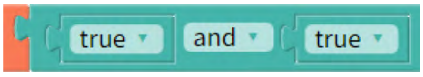
- Logik-Operatoren
- und (Englisch: and)
- oder (Englisch: or)
- nicht (Englisch: not)

Bedingungen und Logik-Operatoren

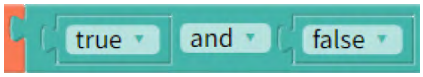
In dieser Lektion lernt ihr etwas über logische Operatoren. Logik-Operatoren ermöglichen es Computern, Entscheidungen basierend auf mehreren Bedingungen zu treffen. Es gibt drei wichtige Logik-Operatoren, die ihr in diesem Abschnitt kennenlernt: **and**, **or** und **not**.

And-Operator

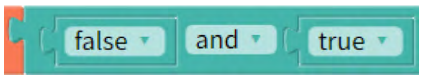
Der Operator „and" (Englisch: und) zeigt "wahr" (Englisch: true) an, wenn alle Eingabebedingungen wahr sind. Wenn eine der Eingabebedingungen falsch ist, gibt er das Ergebnis "falsch" (Englisch: false) aus. Das hier sind alle möglichen Ergebnisse, wenn ihr den **and**-Operator verwendet:



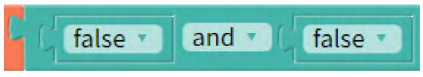
wahr und wahr = wahr



wahr und falsch = falsch



falsch und wahr = falsch



falsch und falsch = falsch

So könnt ihr euch die Verwendung von **and** in einer bedingten Anweisung vorstellen:
Wenn Bedingung 1 und Bedingung 2 wahr sind, dann mache dies.
Wenn eine oder beide Bedingungen falsch sind, dann mache jenes oder nichts.

> **Hinweis:** Die Verwendung des **and**-Operators unterscheidet sich von der Verwendung einer **else if**-Anweisung, da beide Bedingungen gleichzeitig ausgewertet werden und nicht eine nach der anderen.

And-Operator Beispiele

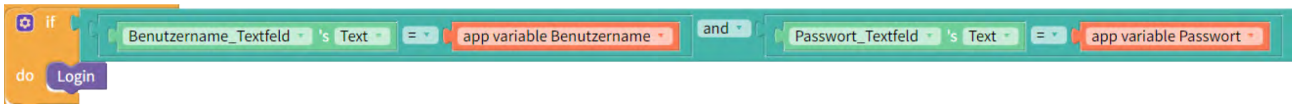
Wenn zwei Bedingungen wahr sein müssen, damit etwas passiert, solltet ihr **and** verwenden. Hier sind ein paar Beispiele, wann ihr **and** in eurer App verwenden solltet. Stellt euch vor, eure App erlaubt es Personen, sich anzumelden. Ihr möchtet, dass eure App sie nur dann anmeldet, wenn ihr Benutzername und ihr Passwort korrekt sind. Das hier sollte eure App tun:

- **Wenn/if** der Benutzername korrekt ist **und/and** das Passwort wahr (hier: richtig) ist, dann kann die Person sich anmelden.
- Wenn eines von beiden falsch ist, dann kann die Person sich nicht anmelden.

Hier sind beide Bedingungen in Thunkable:

Hier sind diese beiden Bedingungen in einem **and**-Block:

Jetzt könnt ihr eine **if**-Anweisung einfügen. Diese **if**-Anweisung meldet die Person nur an, wenn *beide* Bedingungen erfüllt sind.

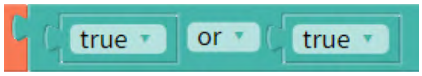


Hier sind weitere Beispiele, wann ihr **and** verwenden könnt.

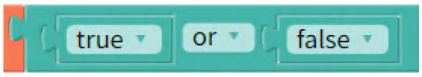
- Ein Spiel gewinnen: „Wenn" eine Person das Level beendet und die Zeit nicht abgelaufen ist, dann gewinnt sie.
- Einen Supermarkt in der Nähe finden: „Wenn" das Geschäft geöffnet ist und das Geschäft in der Nähe der Nutzer:in ist, dann wird es in den Suchergebnissen angezeigt.
- Bilder mit Untertitel posten: „Wenn" die Person ein Bild ausgewählt hat und die Person einen Untertitel eingegeben hat, dann wird das Bild gepostet.
- Fallen euch weitere Beispiele ein?

Or-Operator

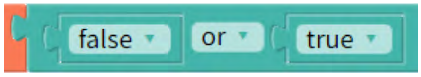
Damit der **or**-Operator (Englisch für oder) "wahr" ausgibt, muss nur eine der Eingaben wahr sein. Das hier sind alle möglichen Ergebnisse, wenn ihr den **or**-Operator verwendet.



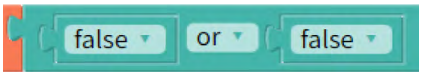
wahr oder wahr = wahr



wahr oder falsch = falsch



falsch oder wahr = falsch



falsch oder falsch = falsch

So könnt ihr euch die Verwendung von **or** in einer bedingten Anweisung vorstellen:
Wenn Bedingung 1 oder Bedingung 2 wahr sind, dann mache dies.
Wenn beide Bedingungen falsch sind, dann mache jenes oder nichts.

☞ **Hinweis:** Der or-Operator kommt euch vielleicht auch ähnlich vor wie else if. Der or-Operator unterscheidet sich aber von else if, weil beide Bedingungen gleichzeitig ausgewertet werden und nicht eine nach der anderen. Den or-Operator verwendet ihr am besten, wenn ihr zwei Bedingungen habt, die das gleiche Ergebnis haben sollen, wenn sie wahr sind. Die folgenden Beispiele erklären das noch ein bisschen besser.

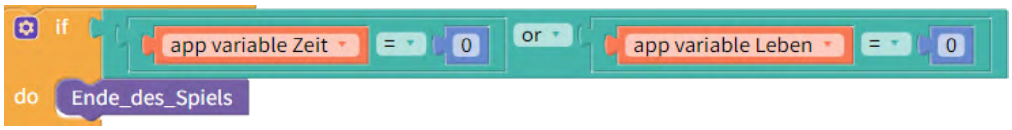
Or-Operator Beispiele

Stellt euch vor, ihr erstellt ein Spiel. Ihr wollt, dass das Spiel zu Ende ist, wenn entweder die Zeit (Englisch: Time) abgelaufen ist oder die spielende Person alle ihre Leben (Englisch: Lives) verloren hat.

Hier sind eure beiden Bedingungen in Thunkable.

Hier sind diese beiden Bedingungen in einem **or**-Block:

Jetzt könnt ihr sie in eine **if**-Anweisung einfügen. Diese if-Anweisung wird das Spiel beenden, wenn *mindestens* eine der Bedingungen wahr ist.



Weitere Beispiele, wann ihr **or** verwenden könnt:

- Suchergebnisse anzeigen: „Wenn" der Titel passt oder die Beschreibung passt, dann wird es in den Suchergebnissen angezeigt.
- Empfehlen von Videos an dich als Nutzer:in: „Wenn" deine Freund:innen es mögen *oder* es ähnlich wie etwas ist, was du magst, dann wird es dir empfohlen.

Not-Operator

Der **not**-Operator (Englisch für nicht) schaltet den Wert einer Eingangsbedingung auf das Gegenteil von dem um, was er ist.

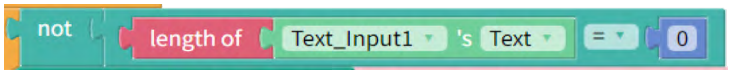
nicht wahr = falsch

nicht falsch = wahr

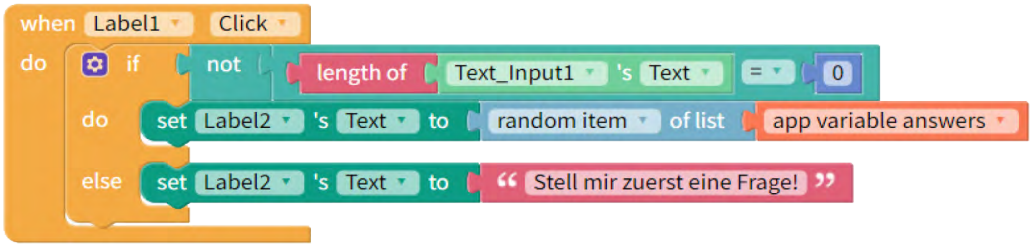
Ihr könnt es euch in einer bedingten Anweisung so vorstellen:
„Wenn" Bedingung 1 nicht wahr ist, dann mache dies.
Ansonsten, mache jenes oder nichts.

Not-Operator Beispiele

In eurer Orakel-App habt ihr schon überprüft, ob eine Frage in das Textfeld eingegeben wurde. Ihr könnt den not-Operator verwenden, um sicherzustellen, dass das Textfeld nicht leer ist. So sieht diese Bedingung in Thunkable aus:



Hier seht ihr, wie ihr diese Bedingung in eine **if**-Anweisung einfügen könnt:



Diese **if**-Anweisung gibt den Nutzer:innen nur dann eine Antwort, wenn das Textfeld nicht leer ist (also, die Länge des Textes nicht 0 ist). Ansonsten gibt die App die Anweisung, dass erst eine Frage gestellt werden soll. Hier sind weitere Beispiele, in denen ihr den **not**-Operator verwenden könnt.

Ihr möchtet Suchergebnisse über Welpen ausschließen, also sucht nach dem Wort „Welpe“, indem ihr den Operator **not** verwendet.

- „Wenn“ der Artikel nicht das Wort „Welpe“ enthält, dann wird das Ergebnis angezeigt.

Ihr könnt logische Operatoren auch kombinieren. So könntet ihr zum Beispiel Suchergebnisse anzeigen, die das Wort „Kätzchen“ enthalten und das Wort „Welpe“ nicht enthalten.

- „Wenn“ der Artikel das Wort „Welpe“ nicht enthält und das Wort „Kätzchen“ enthält, dann wird das Ergebnis angezeigt.

Aktivität: Verbesserung der Orakel-App - Logik-Operatoren

Ihr braucht:

- Computer oder Tablet
- Internet

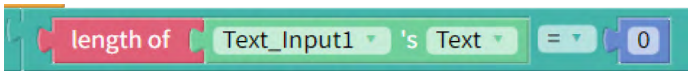
In dieser Aktivität verbessert ihr eure Orakel-App nochmal mit Hilfe von Logik-Operatoren. Eure App soll den Nutzer:innen sagen, dass sie „zuerst eine Frage stellen sollen“, wenn

1. die Nutzer:innen eine Frage stellen, die kein Fragezeichen hat.
2. das Textfeld leer ist.

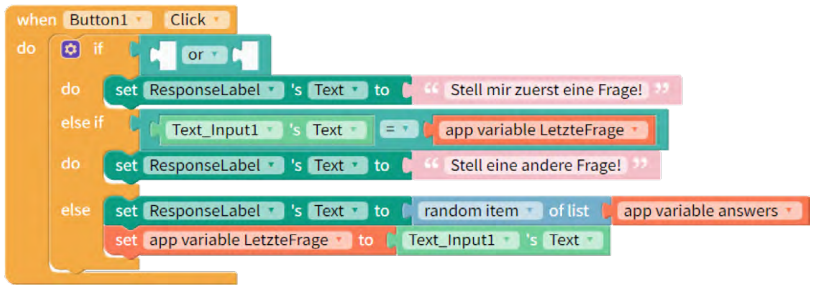
LINK

Hier ist der Start-Code für diese Aktivität.

Schritt 1: Überlegt euch, welche beiden Bedingungen ihr überprüfen müsst und wie ihr diese Bedingungen in Thunkable schreiben könnt.
Lösung: Bedingung 1: Das Textfeld ist leer. Bedingung 2: Der Text enthält kein Fragezeichen.



Schritt 2: Überlegt euch, ob ihr dafür den **and**-Operator oder den **or**-Operator verwenden müsst. Denkt auch darüber nach, an welche Stelle ihr die Bedingungen und den Operator in euren Code einbauen müsst.
Lösung: Den Codebaustein des Operators nehmt ihr so in euren Start-Code auf:



Schritt 3: Ergänzt nun die beiden Bedingungen aus Schritt 1 in euren Logik-Operator.

LINK

Hier findet ihr den Lösungs-Code.

Reflexion

In dieser Lektion habt ihr gelernt, wie man logische Operatoren verwendet, um erweiterte bedingte Anweisungen zu erstellen. An welchen Stellen in eurer eigenen App würdet ihr sie einbauen? Mit welchen weiteren Bedingungen könntet ihr die Orakel-App ergänzen?

GRUNDLAGEN – DEBUGGING-TIPPS

In dieser Lektion ...

- bekommt ihr Tipps zum „Debuggen“ eures Codes in Thunkable

Schlüsselbegriffe

- Debugging
- Fehler (Englisch: Bug)
- Testdaten
- Kommentare
- Versionskontrolle
- Alarm (Englisch: Alert) Funktion

Debugging ist der Prozess, mit dem Programmierer:innen herausfinden, warum ihr Code nicht funktioniert und wo der Fehler (Englisch: Bug) liegt. Die Fehlersuche ist das A und O beim Debugging!

Debugging ist ein natürlicher Teil des Programmierprozesses. Lasst euch also nicht entmutigen, wenn eure App beim ersten Versuch nicht perfekt funktioniert. Oft verbringt ihr den Großteil eurer Zeit mit der Suche nach Fehlern. Plant in eurem Team immer Extra-Zeit für das Debugging ein.

Selbst erfahrene Programmierer:innen machen Fehler – auch wenn sie genau wissen, was sie tun. Deswegen ist es wichtig, sich einen „Werkzeugkasten“ mit Debugging-Techniken zuzulegen. Das hilft euch, wenn ihr euch nicht sicher seid, was schiefgelaufen ist. Jede:r hat eine andere Methode, um den eigenen Code zu „debuggen“, also den Fehler zu suchen. Und auch ihr werdet eure eigene Methode finden, wenn ihr ein bisschen Übung habt. Oft ist das Debuggen der zeitaufwendigste Teil des Programmierens.

Hier findet ihr Tipps, die euch das Debugging erleichtern:

Testdaten nutzen

Ihr verwendet viele und komplizierte Daten in eurer App? Dann ist es hilfreich, Testdaten zum Debuggen zu nutzen. Testdaten sind einfachere Daten, mit denen ihr sicherstellen könnt, dass eure App korrekt funktioniert. Nehmen wir an, ihr erstellt eine App, die den Nutzer:innen anzeigt, wo sich das nächstgelegene Restaurant befindet. Ihr verwendet Tabellen, um die Namen der Restaurants, ihre Standortdaten und Telefonnummern zu speichern. Um mit dem Programmieren eurer App zu beginnen, müsst ihr nicht gleich eine ganze Liste von allen Restaurants in eurer Tabelle anlegen. Ihr könnt einfach ein oder zwei Test-Restaurants einfügen und eure App zum Laufen bringen, bevor ihr die echten Daten hinzufügt.

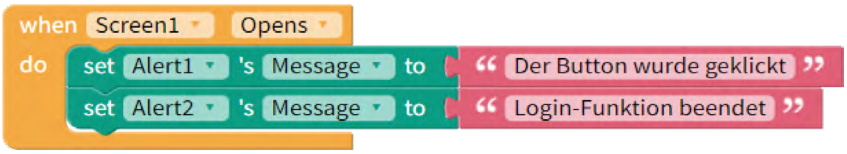
Verwendung der Alarm-Funktion

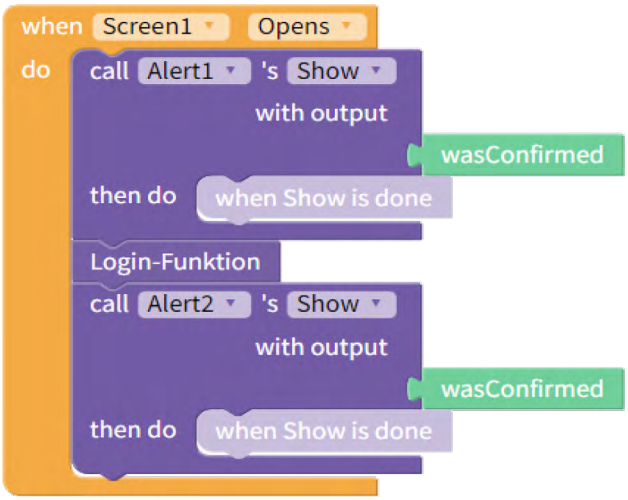
In Thunkable gibt es Funktionen, mit denen ihr herausfinden könnt, warum euer Code fehlerhaft ist: „Alert“ (Englisch für Alarm). Ihr findet die Progammierblöcke des Alerts im Block-Bereich unter App Features (Englisch für App Funktionen). Er erscheint als nicht-sichtbare Komponente (Englisch: non-visible Components) und sieht so aus:

Wenn ihr auf Alert drückt, findet ihr Programmierblöcke und könnt sie in eurem Code einbauen. Wenn ihr eure App dann in Thunkable testet (Englisch: Preview) wird euch die Nachricht des Alerts als kleines Kästchen angezeigt, das ihr bestätigen müsst (Englisch: confirm). Sollte die Nachricht des Alerts, den ihr programmiert habt, nicht erscheinen, so wisst ihr, dass an dieser Stelle etwas in eurem Code nicht stimmt.

Hier sind Beispiele, in denen der Alert angewendet wird:

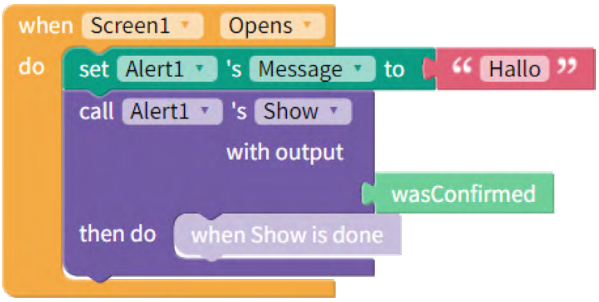
Beispiel 1
Hier wurde eine Fehlersuche programmiert, wenn nach dem Klicken auf eine Schaltfläche (Englisch: Button) nichts passiert oder die Login-Funktion nicht funktioniert:





Erklärung: Wenn der Button erfolgreich geklickt wurde, wird die Meldung “Button wurde geklickt” angezeigt. Wenn dann anschließend die Login-Funktion beendet wurde wird der zweite Alert als Nachricht “Login-Funktion beendet” angezeigt.

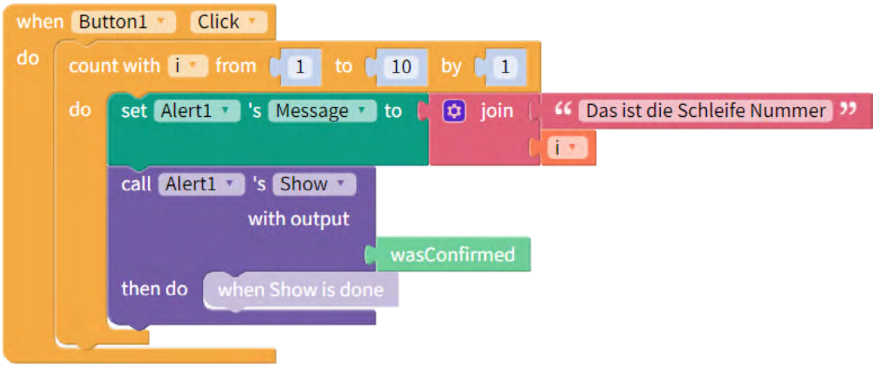
Beispiel 2
Hier wurde ein Alert programmiert, um zu testen, ob ein Screen lädt oder stecken bleibt:



Erklärung: Wenn der Screen geöffnet wird, wird der Alert als Nachricht “Hallo” ausgegeben.

Beispiel 3

Hier wurde ein Alert programmiert, um zu testen, ob eine Schleife (Englisch: Loop) hängen bleibt:

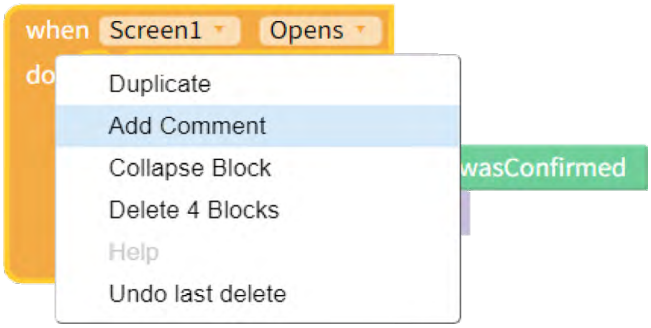


Erklärung: Wenn die Schleife von 1 bis 10 durchzählt, wird bei jeder Schleife der Satz "Das ist die Schleife Nummer" und die Zahl des Durchlaufs als Alert ausgegeben. Also ist der Alert beim ersten Durchlauf "Das ist die Schleife Nummer 1" und beim zehnten Durchlauf ist der Alert "Das ist die Schleife Nummer 10".

Kommentare

Programmierer:innen hinterlassen oft Kommentare in ihrem Code. Damit können sie an bestimmten Stellen erklären, was der Code dort genau macht. Kommentare sind hilfreich, wenn andere Personen sich euren Code anschauen. Kommentare helfen euch auch, wenn ihr euch zu einem späteren Zeitpunkt euren Code anschaut und plötzlich vergessen habt, was manche Teile des Codes tun.

Um einen Kommentar hinzuzufügen, klickt mit der rechten Maustaste auf einen Block und wählt „add comment" (Englisch für Kommentar hinzufügen) aus:



In der Ecke des Blocks wird nun ein Fragezeichen angezeigt und ihr könnt Text hinzufügen. Der Kommentar wird angezeigt, wenn ihr auf das Fragezeichen klickt.

Versionskontrolle

Stellt euch vor, ihr habt einen Teil eurer App gebaut und er funktioniert. Nun fügt ihr einen neuen Code hinzu und plötzlich funktioniert nichts mehr. Ihr löscht den neuen Code wieder, aber die App funktioniert immer noch nicht. Ihr seid euch nicht sicher, was schiefgelaufen ist und wünscht euch, ihr hättet eine „Rückgängig"-Taste.

Das lässt sich vermeiden, indem ihr verschiedene Versionen eurer App erstellt. Wenn ihr einen Fehler gemacht habt und nicht wisst, wie ihr ihn beheben könnt, geht ihr einfach zur zuletzt gespeicherten funktionierenden Version zurück. Ihr könnt die Versionskontrolle auch verwenden, um mit neuen Funktionen zu experimentieren – ohne, dass ihr euch Sorgen machen müsst, dass eure App „kaputt geht".

In Thunkable könnt ihr zu jedem Zeitpunkt eine Kopie eurer App erstellen. Klickt dazu in der oberen rechten Menüleiste auf die Schaltfläche „Projekt duplizieren" (Englisch: duplicate Project). Benennt dann eure gespeicherten Kopien um, sodass ihr zwischen den einzelnen Versionen eurer App leicht den Überblick behalten könnt, zum Beispiel mit dem aktuellen Datum nach der Version: "Version 2023_06_23".

Aktivität: Einen Kommentar hinzufügen

Ihr braucht:

- Computer oder Tablet
- Internet

Fügt mindestens einen Kommentar zu einer eurer selbstprogrammierten Apps hinzu. Der Kommentar soll erklären, was eine Gruppe von Blöcken tut. Wählt eine Gruppe von Blöcken aus, von der ihr denkt, dass sie schwer zu verstehen ist (z. B. für ein anderes Teammitglied). Hier ist ein Beispiel für einen Kommentar. Hilft er euch, den Abschnitt des Codes zu verstehen?

Reflexion

Wann musstet ihr, während ihr eure App programmiert habt, schon mal debuggen? Findet ihr die Alert und Kommentar Funktion hilfreich? Wollt ihr diese Funktionen für eure App nutzen?

FORTGESCHRITTEN – SCHLEIFEN

In dieser Lektion ...

- lernt ihr Schleifen (Englisch: Loops) kennen
- erstellt ihr eine App, die ihr auf Geburtstagspartys mitnehmen könnt

Schlüsselbegriffe

- Schleifen (Englisch: Loops)
- Für-Schleife (Englisch: For-Loop)
- Für-jede-Schleife (Englisch: For-Each-Loop)
- Während-Schleife (Englisch: While-Loops)

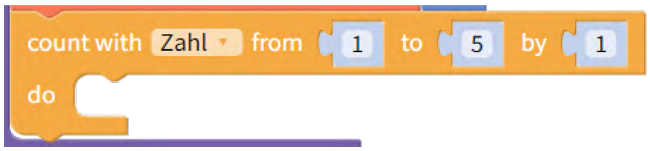
Schleifen

Stellt euch vor, ihr müsst euren Namen 100-mal hintereinander aufschreiben. Das würde bestimmt sehr lange dauern und vielleicht würdet ihr auch Fehler machen, weil eure Konzentration irgendwann nachlässt. Für einen Computer ist das die perfekte Aufgabe, weil er sie sehr schnell und fehlerfrei erledigen kann. Computer sind gut darin, sehr schnell Dinge immer und immer wieder zu tun. Ihr könnt euch diese Eigenschaft zunutze machen, indem ihr Schleifen verwendet. Eine Schleife ist ein Code-Block, der sich immer und immer wieder wiederholt.

Es gibt drei Arten von Schleifen:

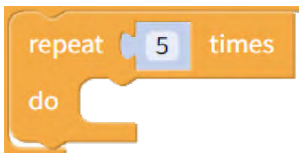
- For-Schleifen wiederholen sich eine bestimmte Anzahl von Malen.
- For-Each-Schleifen sind eine bestimmte Art von „For-Schleifen", die sich für jedes Element in einer Liste einmal wiederholen.
- While-Schleifen wiederholen sich, bis eine Bedingung nicht mehr erfüllt ist.

For-Schleifen heißen so, weil man ihnen sagen kann, wie oft (Englisch: for how many times) sie den Code wiederholen sollen. Ihr sagt eurer App zum Beispiel „Wiederhole dies 17-mal" oder „Wiederhole dies 5-mal". Sie nutzen eine Variable, um zu zählen, wie oft der Code wiederholt wurde. Die Variable nennen wir Zähler. Ihr kontrolliert, wie oft die Schleife wiederholt wird, indem ihr festlegt, wo der Zähler beginnt und endet. Ihr könnt auch festlegen, um wie viel der Zähler bei jeder Wiederholung des Codes ansteigt. In den meisten Fällen möchtet ihr den Zähler bei jeder Wiederholung der Schleife wahrscheinlich um 1 erhöhen. In Thunkable sehen For-Schleifen so aus:

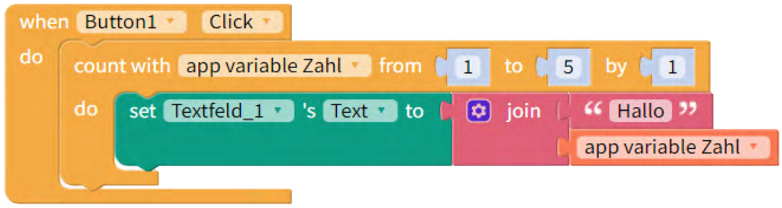


Der Teil, auf dem Zahl steht, ist der Zähler (zählen in Englisch: count). Im Moment heißt der Zähler „Zahl", das könnt ihr aber ändern. Im Beispiel beginnt die „Zahl" bei 1 und hört auf, wenn die „Zahl" gleich 5 ist. Jedes Mal, wenn der Code innerhalb der Schleife wiederholt wird, erhöht sich die „Zahl" um 1. Diese Schleife wird den Code innerhalb der Schleife also fünfmal wiederholen. Im Moment macht die Schleife aber gar nichts, da der Teil „do" (Englisch für tun) leer ist. Thunkable hat auch eine Wiederholungsschleife. Sie funktioniert auf die gleiche Weise, aber

ohne Zähler („Zahl“). Sie wiederholt (Englisch: repeat) den ganzen Block zum Beispiel 5 mal (Englisch: times). Sie kann anstelle vom Block „Zahl“verwendet werden und sieht so aus:



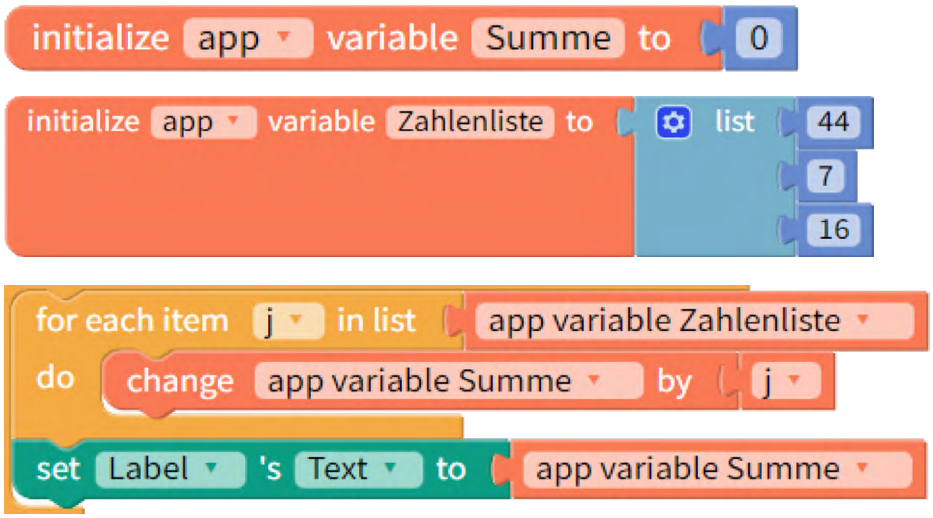
Bislang haben wir nichts an der Zähler-Variable („Zahl“) geändert. Jetzt fügen wir einen Code-Block zum „do“-Teil der Schleife hinzu. Jedes Mal, wenn sie diese Schleife durchläuft, sagt die App „Hallo“ zu ihrer Nutzerin oder ihrem Nutzer – also wird die Person fünfmal begrüßt.



In dieser For-Schleife verwenden wir die Variable „Zahl“ im Code, indem wir sie an das Wort „Hallo“ anhängen. Die „Zahl“ erhöht sich jedes Mal um 1, sodass die App bei jedem Durchlauf etwas anderes ausgibt. Das erste Mal würde die App also “Hallo1” ausgeben, das zweite Mal “Hallo2” und so weiter, bis zum fünften Mal “Hallo5” ausgegeben wird. Die App zählt also für euch mit, wie oft die Nutzer:innen mit “Hallo” begrüßt werden.

Ein weiterer nützlicher Typ von For-Schleifen, den ihr in Thunkable verwenden könnt, sind **For-Each-Schleifen**:

Hier heißt die Zähler-Variable „item“ (Englisch für Element) „j“ und die Schleife ist bereits so eingestellt, dass sie sich um die Anzahl der Elemente in einer Liste wiederholt. „j“ ist dabei ein Platzhalter, der von Thunkable automatisch erstellt wird. Diese Schleifen sind immer dann sehr nützlich, wenn ihr etwas mit einer Liste tun müsst. Nehmen wir an, ihr habt eine Liste mit Zahlen und wollt alle Zahlen addieren und in einer Variable namens „sum“ (Englisch für Summe) speichern. Dann würdet ihr dies mit einer For each-Schleife umsetzen.



Wie funktioniert dieser Code?

Jedes Mal, wenn die Schleife läuft, wird der Variablen „Summe“ ein Element aus Zahlenliste hinzugefügt. Die Schleife stoppt automatisch, wenn alle Zahlen in der Liste hinzugefügt wurden. So läuft die Schleife ab:

- Es gibt ein Element auf der Liste, Startschleife: „app variable Summe“ = 0 + 44
- Es gibt noch Elemente in der Liste, Schleife wiederholen: „app variable Summe“ = 44 + 7 (= 51)
- Es gibt noch Elemente in der Liste, Schleife wiederholen: „app variable Summe“ = 51 + 16 (= 67)
- Es sind keine Elemente mehr in der Liste, Schleife verlassen: Text von Label auf 67 setzen

While-Schleifen sind Schleifen, die so lange laufen, bis eine Bedingung nicht mehr erfüllt ist. Sie werden deshalb While-Schleifen genannt, weil der Code wiederholt wird, während (Englisch: while) eine Bedingung noch erfüllt ist. Ihr könnt euch While-Schleifen so vorstellen, dass ihr eurer App sagt: „Solange dies geschieht, wiederhole dies“ oder „Solange sich dies nicht geändert hat, wiederhole dies“.

Hier sind zwei Beispiele zum Nachdenken:
Ihr feiert eine Party und möchtet, dass die Musik so lange spielt, bis alle Gäste gegangen sind. Ihr könntet eure Party als diese While-Schleife beschreiben:

- While (Gäste auf Party > 0), Do: weiter Musik spielen

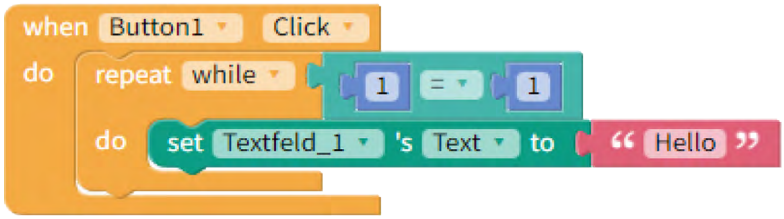
Was, wenn ihr auch möchtet, dass eure Musik aufhört zu spielen, wenn es später wird als Mitternacht? Ihr könnt While-Schleifen programmieren und sie mit Logik-Operatoren kombinieren, um mehrere Bedingungen zu verknüpfen, bevor die Schleife beendet wird. Jetzt könnt ihr eure Party so beschreiben:

- While (Gäste auf Party > 0) und (Zeit < Mitternacht), Do: weiter Musik spielen

In diesem Fall würde die Musik aufhören, sobald alle die Party verlassen haben oder wenn es nach Mitternacht ist.

Um eine While-Schleife zu verwenden, müsst ihr eine Bedingung aufstellen, die mit „wahr“ beginnt. Wenn die Bedingung falsch ist, wird die Schleife nicht ausgeführt. Die Schleife prüft die Bedingung jedes Mal, bevor sie wiederholt wird. So stellt sie sicher, dass die Bedingung immer noch wahr ist. Hier seht ihr, wie While-Schleifen in Thunkable aussehen:

Beachtet: Bei While-Schleifen können Fehler passieren! Wenn ihr eine Bedingung wählt, die niemals falsch sein wird, dann endet eure Schleife niemals. Das nennt man eine Endlosschleife. Hier ist ein Beispiel:



Da 1 immer gleich 1 sein wird, kann diese Bedingung niemals falsch sein. Bei so einer Endlosschleife kann es vorkommen, dass ihr eine Meldung bekommt, dass Thunkable nicht mehr funktioniert. Keine Sorge: Wenn ihr programmiert, gehen ständig Dinge kaputt oder hängen sich auf! Startet eure App einfach neu und versucht es noch einmal.

> **Hinweis:** Mobile Apps sind ereignisgesteuert. Das heißt, der Code wird ausgeführt, sobald ein Ereignis stattfindet (z. B. ein Klick oder Swipe (Englisch für wischen) mit dem Finger). Das bedeutet, dass Schleifen nicht häufig verwendet werden. Der Schleifenblock der am häufigsten für Apps verwendet wird, ist der For Each-Block – und das auch nur, wenn in eurer App Listen verwendet werden.

Aktivität: Wie alt bist du?

Ihr braucht:

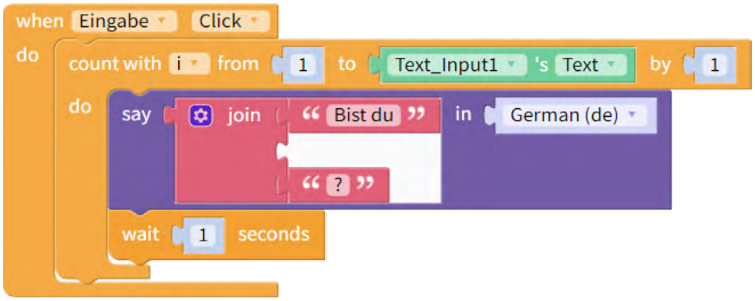
- Computer oder Tablet
- Internet

LINK

In manchen englischsprachigen Ländern gibt es ein Geburtstagslied, in dem die Lebensjahre einzeln gesungen werden. Ist jemand zum Beispiel vier Jahre alt, wird gesungen: Bist du 1, bist du 2, bist du 3, bist du 4? Auf Englisch ist das: *„... are you 1, are you 2, are you 3, are you 4“?* Falls ihr es noch nie gehört habt, so hört es euch hier an.

Dieses Lied kann also echt lange dauern. Besonders wenn die Person 98 Jahre alt ist, wie die Oma im Video. Ein Computer könnte das viel schneller machen, indem er Schleifen verwendet. Stellt euch vor, ihr könntet eine App erstellen, in der die Person einfach ihr Alter eingibt und die App singt das Lied für sie!

In dieser Aktivität erstellt ihr eine App, die ihr zu Geburtstagsfeiern mitnehmen könnt. Dafür nutzt ihr eine **For-Schleife**:



Diese For-Schleife zählt von 1 bis zu der Zahl, die die Nutzer:innen in das Textfeld eingeben. Jedes Mal, wenn die Schleife läuft, erhöht sich der Zähler um 1. Dieser Code ist fast vollständig. Leider gibt er nicht bei jedem Zählvorgang das Alter an. Findet heraus, wie ihr diesen Block dazu bringt, das Alter bei jeder Zählung anzugeben.

LINK

Nutzt dafür diesen Start Code.

LINK

Hier ist der Lösungs-Code.

Reflexion

In dieser Lektion habt ihr etwas über For-Schleifen, While-Schleifen und For each-Schleifen gelernt. Außerdem habt ihr eine App erstellt, die eine Schleife verwendet. Fandet ihr die Verwendung von Schleifen einfach oder schwierig? Wie könnt ihr Schleifen in eurer eigenen App nutzen?

FORTGESCHRITTEN – SENSOREN UND KOMPONENTEN

In dieser Lektion ...

- lernt ihr, was Sensoren sind
- lernt ihr verschiedene Komponenten kennen, die ihr in Thunkable verwenden könnt
- wählt ihr einen Sensor oder eine Komponente für eure App aus und baut sie ein

Schlüsselbegriffe

- Medienkomponenten
- Sensoren
- Soziale Komponenten
- Verbindungskomponenten

Smartphone-Sensoren und Komponenten

Jetzt geht es darum, Sensoren oder Komponenten, also Teile oder Funktionen eures Smartphones, wie zum Beispiel Kamera oder Internetverbindung, für eure App auszuwählen und zu programmieren. Sensoren und Komponenten ermöglichen es eurer App, viele verschiedene Dinge zu tun.

Hier sind einige wichtige Komponenten von Smartphones und ihre Funktionsweise als Beispiele:

Komponenten	Was sie tun
Kamera, Lautsprecher, Mikrofon	erlauben das Aufnehmen von Bildern, Videos und Ton
GPS	zeigt den Standort des Smartphones an
Speicher auf einem Smartphone	ermöglicht das Speichern von Einstellungen, Bildern und Tönen
Verbindung zum Internet	ermöglicht die Verbindung des Smartphones mit Informationen aus dem Internet
Beschleunigungsmesser, Gyroskop	zeigen an, wie schnell sich das Smartphone bewegt
Anrufe, Textnachrichten, Kontaktlisten	ermöglichen das Tätigen von Anrufen, das Senden von Textnachrichten und das Verbinden mit Personen

In Thunkable gibt es auch noch „unsichtbare“ Komponenten. Sie heißen „App Features“ (Englisch für App-Funktionen) und ihr findet sie im Block-Bereich. Es kommen immer neue Funktionen dazu. Hier sind die Wichtigsten:

Medienkomponenten

Benötigt eure App Funktionen, die sich auf Dinge wie Fotos, Audio und Video beziehen? Dann sind diese Komponenten wahrscheinlich hilfreich für euch:

- Kamera & Medien (Englisch: Camera & Media): Video aufnehmen, Foto aufnehmen, Fotos aus der Galerie auswählen, Bildbeschreibung generieren.
- Sprache (Englisch: Speech): Übersetzen, Text vorlesen.
- Ton (Englisch: Sound): Ton aufnehmen, lange und kurze Töne abspielen

Sensoren

Soll eure App Informationen aus der Umgebung um sie herum oder von den Nutzer:innen sammeln können? Dafür gibt es ein paar nützliche Sensoren, auf die ihr in Thunkable zugreifen könnt. Beispiele für Sensoren sind der Standortsensor (Englisch: Location Sensor) oder der Beschleunigungsmesser (Englisch: Accelerometer). Der Standortsensor sammelt Breiten- und Längengrad des Handystandorts und wird zum Beispiel bei Google Maps verwendet. Der Beschleunigungssensor kann feststellen, ob das Smartphone wackelt, ob es aufrecht oder verkehrt herum gehalten wird, und wird zum Beispiel bei einer Schrittzähler-App verwendet.
Beachtet, dass nicht alle Smartphones oder mobile Geräte (z. B. ein Tablet) über alle diese Sensoren verfügen. Wenn ihr euch nicht sicher seid, ob eure Zielgruppe Zugang zu Geräten mit diesen Sensoren hat, fügt die Funktionen lieber nicht hinzu.

Soziale Komponenten

Soll eure App Anrufe tätigen, E-Mails senden, SMS verschicken und bestimmte Arten von Informationen teilen können? Dann könnten diese zwei sozialen Komponenten hilfreich sein: Teilen (Englisch: Share): Damit können Nutzer:innen aus eurer App heraus einen Anruf tätigen, eine Textnachricht an das Smartphone von anderen Nutzer:innen senden oder Nachrichten, Bilder oder andere Inhalte in eurer App mit anderen Apps auf ihrem Smartphone teilen. Werbeanzeigen (Englisch: Ads): Thunkable ermöglicht es euch, Anzeigen zu eurer App hinzuzufügen. Alle Apps müssen zuerst von Thunkable genehmigt werden, bevor sie heruntergeladen oder veröffentlicht werden können.

Verbindungskomponenten

Verbindungskomponenten ermöglichen es eurer App zum Beispiel mit dem Internet oder anderen Apps zu interagieren. Beispiele von Verbindungskomponenten sind die Karte (Englisch: Map) und Web-Schnittstelle (Englisch: WebAPI). Mit der Kartenkomponenten könnt ihr eine Karte in eure App einbetten, wenn ihr zum Beispiel den Weg zum nächsten Krankenhaus anzeigen wollt. Die Web-Schnittstelle ermöglicht es eurer App, Informationen von anderen Websites zu erhalten oder an sie zu senden. Wenn ihr zum Beispiel innerhalb eurer App das Wetter anzeigen lassen möchtet, braucht ihr keine eigene Messstation aufzubauen. Ihr könnt die Informationen mit Hilfe der Web-Schnittstelle von einer Wetter-Website erhalten und in der App anzeigen.

Aktivität: Lernt, wie ihr Sensoren und Komponenten nutzt

Ihr braucht:

- Computer oder Tablet
- Internet

Gute Recherche, also das Suchen von passenden Informationen zu einem Thema, ist eine der wichtigsten Fähigkeiten von Programmierer:innen. Dazu gehört, Ressourcen oder Quellen zu finden, die euch helfen, wenn ihr bei einem Problem nicht weiterkommt.

Für diese Aktivität wählt ihr mindestens einen Sensor oder eine Komponente aus, die ihr in eurer App verwenden möchtet. Findet dafür ein passendes Tutorial (Englisch für Anleitung) im Internet und lernt damit, wie ihr den Sensor oder die Komponente in eure App einbaut.

Hier sind Tipps für die Suche nach Anleitungen:

- Ihr solltet mit einer Online-Suche über eine Suchmaschine (z. B. Google oder Ecosia) beginnen. Achtet darauf, dass ihr Stichwörter (Englisch: Keywords) und Suchbegriffe verwendet, die den Namen eurer Komponente beinhalten, wie zum Beispiel „Google Maps Thunkable“.
- Sucht nach Videos auf YouTube. Achtet darauf, dass ihr Stichwörter verwendet, genau wie bei einer Online-Suche.
- Oft wird für Anleitungen auf Deutsch auch der englische Begriff, also Tutorials verwendet. Diesen Begriff könnt ihr auch zusammen mit euren Stichwörtern eingeben.

Thunkable

LINK

Thunkable hat viele Tutorial-Videos auf dem Youtube-Kanal.

LINK

Thunkable hat einen Hilfebereich auf Thunkable Docs.

> 👉 **Hinweis:** Das Video ist auf Englisch. Die Anleitung, wie ihr deutsche Untertitel hinzufügt findet ihr in der Aktivität der Lektion: Thunkable einrichten und benutzen.

Reflexion

Wie habt ihr die Anleitung oder Informationen gefunden, die ihr gesucht habt? Wie nutzt ihr eure Ergebnisse aus dieser Lektion in eurer App? Gibt es andere Bereiche in eurem Leben, in denen ihr diese Fähigkeit, selbstständig eine Anleitung für etwas zu finden, anwenden könnt?

FORTGESCHRITTEN – DATENSPEICHERUNG

In dieser Lektion ...

- lernt ihr, wie ihr Daten auf eurem Gerät in Thunkable speichern könnt
- verwendet ihr gespeicherte Variablen in Thunkable

Schlüsselbegriffe

- Lokale Speicherung
- (Lokale) Datenbank
- Datenbankaufruf
- Schlüssel

Informationen auf eurem Gerät speichern

Lokale Speicherung bedeutet, dass ihr Informationen direkt auf eurem Gerät speichert. Hier sind Beispiele für Datenspeicher, die ihr vielleicht sogar täglich benutzt:

- Kontaktliste: Ihr speichert die Telefonnummern von Freund:innen, damit ihr sie später verwenden könnt.
- Messenger-App: Die meisten Messenger-Apps speichern automatisch alte Nachrichten, damit ihr sie später lesen könnt.
- Dateiablage: Hier speichert ihr Dokumente und Papierkram.

In eurer App könnt ihr den lokalen Speicher nutzen, um Informationen zwischen den Zeitpunkten zu speichern, an denen eure Nutzer:innen die App öffnen und schließen.

Datenbanken

Ihr habt in den Lektionen Variablen und Listen schon zwei Beispiele kennengelernt, wie ihr Daten in der App und auf eurem Smartphone speichert. Wenn die Daten komplizierter und es viele sind, dann kann es hilfreich sein, eine andere Datenquelle, zum Beispiel eine Datenbank, zu erstellen.

Der lokale Speicher wird manchmal auch als lokale Datenbank bezeichnet. Eine Datenbank ist eine organisierte Sammlung von Informationen. Der Zugriff auf eine Datenbank wird als Aufruf bezeichnet. Wenn ihr eine Datenbank aufruft, könnt ihr Informationen abrufen oder löschen, neue Informationen speichern oder Informationen aktualisieren. Denkt nochmal an das Postkartenbeispiel für Variablen in Coding Lektion Variablen. Genauso wie Variablen einen eindeutigen Namen bekommen, der dann wie der Name einer Schachtel ist, bekommen auch eure Daten einen genauen Namen. Dieser Name wird bei Datenbanken „Schlüssel“ genannt.

Die nächsten beiden Abschnitte zeigen euch, wie ihr die Speicherinformationen auf eurem Gerät in Thunkable verwenden könnt.

Verwendung der lokalen Speicherkomponente in Thunkable

In Thunkable könnt ihr unter dem Menü Datenquellen (Englisch: Data Sources), das ihr in der linken Menüleiste findet, Speicheroptionen für Daten auswählen. Wenn ihr neben “Data Sources” auf das “+” klickt und dann auf “Create New” (Englisch für Neue erstellen) könnt ihr eine neue Datenquelle einrichten.

☞ **Hinweis:** Es ist möglich, Thunkable mit externen Datenquellen wie z. B. Google Sheets, Air Table und Webflow zu verknüpfen. Wenn ihr mehr über externe Datenquellen wissen wollt, schaut in den Thunkable Docs nach.

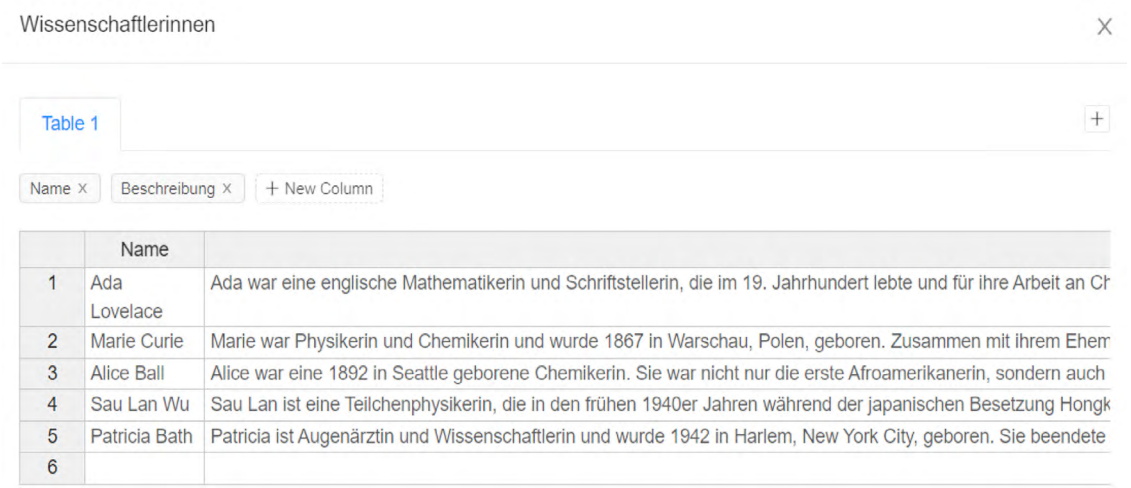
LINK

Fokus dieser Lektion ist die lokale Speicherung, das heißt, dass ihr Daten lokal, also auf dem Smartphone eurer Nutzer:innen speichert. Die Daten sind jedes Mal da, wenn sich die Nutzer:innen in eurer App anmelden. Das heißt auch, dass die Daten dann nur auf dem eigenen Smartphone verfügbar sind und auf keinem anderen.
Denkt auch hier nochmal an die Lektion Variablen zurück. Für Variablen gibt es auch die Möglichkeit Werte lokal, also innerhalb der App oder dem Smartphone und in einer Cloud zu speichern. Das ist auch für andere Daten ähnlich.

Um eine lokale Datenquelle in Thunkable zu erstellen, klickt ihr auf “Create your own table” (Englisch für Erstelle deine eigene Tabelle).

Im nächsten Schritt gebt ihr einen Namen eurer Datenquelle ein und klickt auf “create” (Englisch für erstellen). Nun könnt ihr eure Datentabelle befüllen, neue Spalten (Englisch: Columns) hinzufügen oder löschen und seht eure Tabelle unter dem Datenquellen-Menü. Wenn ihr Daten anpassen oder ändern wollt, könnt ihr das immer über das Menü Datenquellen machen oder über Programmierblöcke.

Beispiel: Eine Lexikon-App erstellen
Angenommen, ihr wollt eine Lexikon-App erstellen, die ihren Nutzer:innen Informationen über berühmte Wissenschaftlerinnen anzeigt. Ihr erstellt also unter dem Menü Datenquellen eine eigene Tabelle (Englisch: create your own table) mit dem Namen „Wissenschaftlerinnen“. Die erste Spalte (Englisch: Column) benennt ihr in „Name“ um und die zweite Spalte in „Beschreibung“. Nun füllt ihr die Tabelle mit Namen und Beschreibungen von Wissenschaftlerinnen. Das sieht dann in Thunkable so aus:

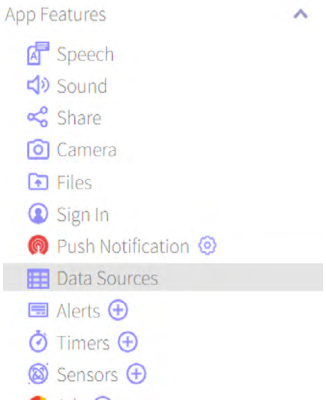


	Name	
1	Ada Lovelace	Ada war eine englische Mathematikerin und Schriftstellerin, die im 19. Jahrhundert lebte und für ihre Arbeit an Ch
2	Marie Curie	Marie war Physikerin und Chemikerin und wurde 1867 in Warschau, Polen, geboren. Zusammen mit ihrem Ehem
3	Alice Ball	Alice war eine 1892 in Seattle geborene Chemikerin. Sie war nicht nur die erste Afroamerikanerin, sondern auch
4	Sau Lan Wu	Sau Lan ist eine Teilchenphysikerin, die in den frühen 1940er Jahren während der japanischen Besetzung Hongk
5	Patricia Bath	Patricia ist Augenärztin und Wissenschaftlerin und wurde 1942 in Harlem, New York City, geboren. Sie beendete
6		

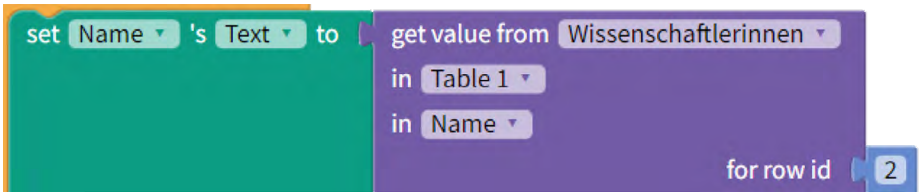
Thunkable hat unter der Datenquelle „Wissenschaftlerinnen“ eine erste Tabelle (Englisch: Table 1) angelegt. Ihr seht in dieser Tabelle, dass die Zeilen (Englisch: Rows) beginnend von 1 durchnummeriert sind, hier in diesem Beispiel von 1 bis 6. Diese Nummerierung in Kombination mit der Spaltenbeschreibung ordnet jeder Information in eurer Datentabelle einen eindeutigen Schlüs-

sel zu. So sind zum Beispiel alle Informationen (Name und Beschreibung) über Marie Curie unter der Zeilennummer „2“ gespeichert. Das ist wichtig, wenn ihr Informationen eurer Datentabelle mit Hilfe der App aufrufen, ändern, aktualisieren oder löschen wollt.

Um mit Hilfe eurer App mit eurer Datenquelle zu kommunizieren, gibt es im Block-Bereich von Thunkable, sobald ihr eine Datenquelle hinzufügt, Programmierblöcke für das Aufrufen eurer Datenbank. Ihr findet sie im Blockbereich unter den App-Features bei Datenquellen (Englisch: Data Sources).

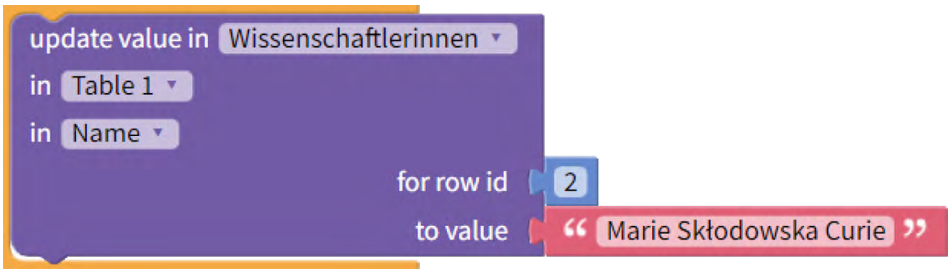


Wollt ihr zum Beispiel in einem Etikett „Name“ (Englisch: Label) den Namen einer Wissenschaftlerin aus eurer Datentabelle anzeigen, dann könnt ihr das mit dem Block „get value“ (Englisch für hole Wert):



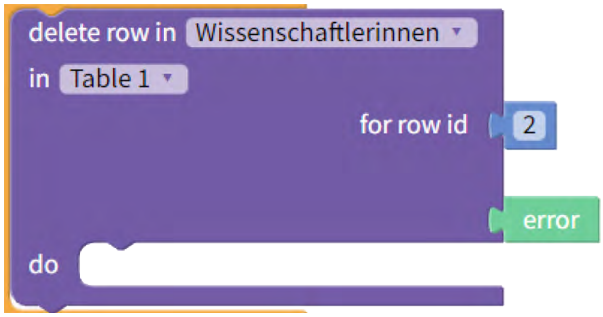
Dieser Block schaut in eure Datentabelle „Wissenschaftlerinnen“, und zwar in die erste Tabelle (Englisch: Table 1) und in die Spalte „Name“. Bei „for row id“ (Englisch für Zeilennummer) benötigt dieser Block nun als eindeutigen Schlüssel die Zeilennummer, aus der Informationen angezeigt werden sollen. In diesem Beispiel wird die Zeile mit der Nummer „2“ ausgewählt und damit wird der Name „Marie Curie“ im Etikett angezeigt.

So ähnlich funktioniert es, wenn ihr eine Information in eurer Datenquelle aktualisieren oder löschen wollt. Zum Aktualisieren verwendet ihr den Block „update value“ (Englisch für Wert aktualisieren):



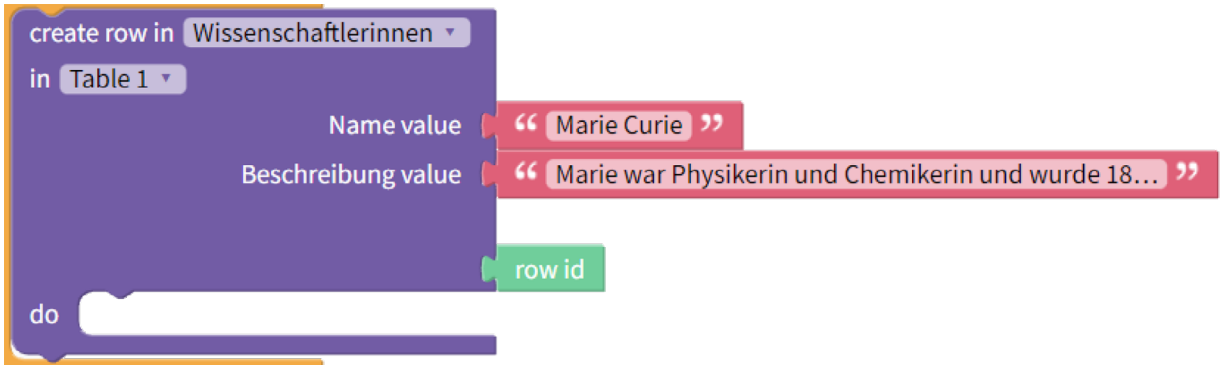
Dieser Block ändert in eurer Datentabelle „Wissenschaftlerinnen“, in der ersten Tabelle, in der Spalte „Name“ und in der Zeile „2“ den Namen von Marie Curie auf den Wert (Englisch: Value) „Marie Skłodowska Curie“. Ihr habt also den Namen der Wissenschaftlerin auf ihren vollen Namen aktualisiert.

Zum Löschen verwendet ihr den Block „delete row“ (Englisch für Zeile löschen):



Dieser Block löscht alle Informationen in eurer Datentabelle „Wissenschaftlerinnen“, in der ersten Tabelle und in der Zeile „2“. Damit wird sowohl der Name „Marie Curie“ als auch die Beschreibung über Marie Curie gelöscht, also alles, was in Zeile 2 steht.

Wenn ihr es euren Nutzer:innen ermöglichen wollt, dass sie einen neuen Eintrag in der Datentabelle erstellen können, dann verwendet ihr den Block „create row“ (Englisch für Zeile erstellen):



Wieder wird die erste Tabelle eurer Datentabelle „Wissenschaftlerinnen“ aufgerufen. Thunkable fügt diesem Block automatisch einen Platz für alle Spalten in eurer Datentabelle hinzu, hier also für eure Spalten „Name“ und „Beschreibung“. Dahinter könnt ihr dann jeweils zum Beispiel mit einem Textblock neue Informationen in eurer Datentabelle speichern. Hier wird nun eine neue Zeile mit dem Namen „Marie Curie“ und einer Beschreibung eingefügt. Achtung: Die Zeile wird am Ende der Tabelle eingefügt. Wollt ihr also in Zukunft mehr über Marie Curie wissen, ist der Schlüssel, um die Informationen aufzurufen, nicht mehr die Zeilennummer „2“!

Aktivität: Erstellt eine Lexikon-App

Ihr braucht:

- Computer oder Tablet
- Internet

Erstellt nun selbst eine Lexikon-App, die ihren Nutzer:innen hilft, sich über jedes gewünschte Thema zu informieren. Die App soll auf einem Bildschirm die Namen anzeigen. Wenn die Nutzer:innen dann auf einen der Namen klicken, soll eure App ihnen mehr Informationen geben.

LINK

Ihr könnt diesen Start-Code verwenden oder komplett von vorne beginnen.

Schritt 1: Datenquelle erstellen

Ihr könnt euch selbst ein Thema für eure Lexikon-App aussuchen, das euch interessiert. Das können berühmte Personen oder Orte, beliebte Filme, bevorstehende Veranstaltungen oder etwas anderes sein. Ihr könnt auch, so wie in der Lektion, eine Liste über Wissenschaftlerinnen erstellen. Erstellt nun eine eigene Tabelle und gebt drei bis sechs Zeilen Informationen ein.
Wenn ihr eine Lexikon-App über Wissenschaftlerinnen erstellen wollt, könnt ihr diese Informationen in eure Tabelle kopieren:

Name	Beschreibung
Ada Lovelace	Ada war eine englische Mathematikerin und Schriftstellerin, die im 19. Jahrhundert lebte und für ihre Arbeit an Charles Babbages frühem mechanischen Allzweckcomputer – der Analytical Engine – bekannt ist. Ada wird oft als die erste Programmiererin überhaupt bezeichnet, weil sie den ersten Algorithmus geschrieben hat, der von einer Maschine ausgeführt werden sollte.
Marie Curie	Marie war Physikerin und Chemikerin und wurde 1867 in Warschau, Polen, geboren. Zusammen mit ihrem Ehemann Pierre entdeckte sie zwei neue radioaktive Elemente. Marie gewann 1903 den Nobelpreis für Physik. Sie entdeckte auch, dass Radiumgas zur Krebsbehandlung verwendet werden kann.
Alice Ball	Alice war eine 1892 in Seattle geborene Chemikerin. Sie war nicht nur die erste Afroamerikanerin, sondern auch die erste Frau, die ihren Abschluss an der Universität von Hawaii machte. Im Alter von nur 23 Jahren entwickelte Alice ein Heilmittel für Lepra, eine Krankheit, die vorher als unheilbar galt.

Sau Lan Wu	Sau Lan ist eine Teilchenphysikerin, die in den frühen 1940er Jahren während der japanischen Besetzung Hongkongs geboren wurde. Ihren Doktortitel erwarb sie in Harvard. Sie leitete das Team, das das „Gluon" entdeckte. Sie ist eine der wichtigsten Teilchenphysikerinnen auf ihrem Gebiet und hat viele bahnbrechende Entdeckungen gemacht.
Patricia Bath	Patricia ist Augenärztin und Wissenschaftlerin und wurde 1942 in Harlem, New York City, geboren. Sie beendete die High School in nur zweieinhalb Jahren und wusste, dass sie Ärztin werden wollte. 1985 schloss sie eine Erfindung ab, die den Grauen Star heilen und das Sehvermögen von Menschen auf der ganzen Welt wiederherstellen kann.

Schritt 2: Namen auf dem Bildschirm anzeigen
Ihr wollt nun, dass alle Namen auf eurem Bildschirm als eine Liste angezeigt werden, sodass die Nutzer:innen auf den Namen klicken können, über den sie mehr erfahren wollen. Dafür fügt ihr im Design-Bereich auf eurem ersten Bildschirm das Element „List-Viewer" hinzu. Im Block-Bereich müsst ihr nun programmieren, dass die Daten aus eurer Datentabelle in diesem List-Viewer angezeigt werden, sobald sich der Bildschirm öffnet. Alle Werte einer Spalte könnt ihr in eurer Datenbank über den Block „list of values" (Englisch für Liste der Werte) aufrufen. In diesem Block wählt ihr dann eure erstellte Datentabelle und die richtige Spalte (also zum Beispiel die Spalte „Name") aus.

Schritt 3: Einen Schlüssel festlegen
Ihr wisst nicht, auf welchen Namen eure Nutzer:innen klicken werden, also führt ihr eine Variable als Schlüssel in eurer App ein. Ihr verwendet die Variable „Zeilennummer", weil die Zeilennummer Teil von eurem eindeutigen Schlüssel ist, welche Informationen ihr aus eurer Datentabelle aufrufen wollt. Die Zeile in eurer Datentabelle ist dieselbe wie die Zeile in eurer Liste im List-Viewer, die den Nutzer:innen angezeigt wird. Die Zeile im List-Viewer wird als Index bezeichnet (hier könnt ihr auch nochmal in der Lektion Listen nachschauen). Ihr setzt also den Index, also das Element des List-Viewers, auf das geklickt wird, als den Wert der Variable „Zeilennummer". Damit stellt ihr sicher, dass immer die richtigen Informationen angezeigt werden.

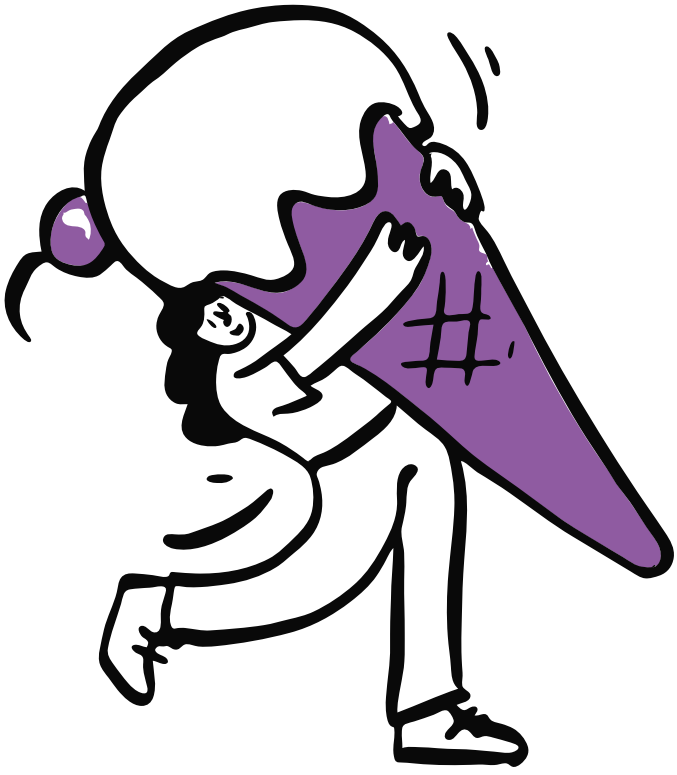
Schritt 4: Informationen aus Datenbank aufrufen
Um die Informationen über den Namen anzuzeigen, wenn Nutzer:innen auf den Namen klicken, richtet ihr einen zweiten Screen ein. Wenn also auf einen Namen geklickt wird, öffnet sich ein zweiter Bildschirm. Auf diesem platziert ihr ein Etikett für jede Spalte eurer Datentabelle. Für jedes Etikett programmiert ihr dann einen Aufruf eurer Daten aus der Datenbank mit der ausgewählten Zeilennummer als Schlüssel. Dafür verwendet ihr den Block „get value" (Englisch für Erhalte Wert).

LINK

Habt ihr Schwierigkeiten? Hier findet ihr den Lösungs-Code.

Reflexion

War es leicht für euch, Datenspeicherung zu verstehen? Welche Art von Daten könntet ihr in eurer App speichern?



KÜNSTLICHE INTELLIGENZ

KÜNSTLICHE INTELLIGENZ KENNENLERNEN

In dieser Lektion ...

- lernt ihr, was eine Künstliche Intelligenz (KI) ist und wo sie bereits zur Anwendung kommt
- erfahrt ihr, woran man I-Technologien erkennen kann
- Handlungen einer KI analysieren

Schlüsselbegriffe

- Künstliche Intelligenz (KI)
- KI-Modell
- Muster
- Vorhersage

Wichtig ist es, sich zuerst die Bedeutung der Begriffe „künstlich“ und „Intelligenz“ anzuschauen. Künstlich meint, das etwas nicht echt ist, sondern vom Menschen erschaffen wurde und nur eine Kopie von etwas Echtem ist. Intelligenz bedeutet, den Verstand zu nutzen, um zu lernen, zu verstehen und zu erkennen. Bei Künstlicher Intelligenz handelt es sich um etwas, das von Menschen geschaffen wurde und das Ziel verfolgt, das menschliche Gehirn bei der Problemlösung zu simulieren, das heißt nachzuahmen.

Was hat KI mit App-Entwicklung zu tun?

KI existiert überall um euch herum! Sie ist zu einem Teil unseres Alltags geworden. Auch in Apps kommt sie sehr oft vor. Beispiele sind eine App zur Gesichtserkennung oder zum Erkennen von Vogelgesang. Auch hinter einer Übersetzungsapp kann KI stecken, wie zum Beispiel bei DeepL. Es kann auch für euch als App-Entwickler:innen interessant sein zu überlegen, wie ihr KI für eure App nutzen könnt und wo es sinnvoll ist.

Im Wesentlichen kann Künstliche Intelligenz vier Dinge:

- Verstehen oder wahrnehmen – Beispiel: Sprachassistenten, wie Siri oder Alexa, erkennen, wenn ihr deren Namen ruft, und werden aktiviert.
- Mit Menschen interagieren – Beispiel: Sprachassistenten und Chatbots wie Siri oder Google Assistant reagieren auf das, was Menschen ihnen sagen, und können sich so mit Menschen unterhalten.
- Lernen – Beispiel: Plattformen wie Netflix und YouTube merken sich, was für Videos ihr mögt und schlagen andere Videos vor, die euch gefallen könnten.
- Modelle verwenden, um Entscheidungen zu treffen – Beispiel: Google Maps verwendet Straßen-Modelle zur Berechnung von Strecken

Was ist ein KI-Modell und was ist mit Muster gemeint?

Ein KI-Modell ist eine Art Computerprogramm, das darauf trainiert wird, bestimmte Dinge zu lernen, indem es große Mengen von Daten analysiert und Muster in den Daten erkennt. Ein Muster meint hier eine Art von Regelmäßigkeit, die in den Daten gefunden wird. Zum Beispiel kann ein Muster darin bestehen, dass in Fotos von Hunden oft Ohren und Pfoten zu sehen sind. Das KI-Modell kann diese Muster lernen und sie dann verwenden, um Vorhersagen zu treffen oder Entscheidungen zu treffen. Ein Beispiel dafür ist ein KI-Modell, das trainiert wurde, Hunde auf Fotos zu erkennen, indem es nach Mustern wie Ohren und Pfoten sucht.

Aktivität: KI-Modelle brainstormen

Ihr braucht:

- Stifte
- Internet

Macht einen Teambrainstorm und überlegt euch drei weitere KI-Modelle, die euch bereits im Alltag umgeben. Welche Apps kennt ihr, die KI nutzen? Ihr könnt dazu auch im Internet recherchieren.

Reflexion

Wart ihr überrascht, wie oft KI schon zum Einsatz kommt? Was hat euch am meisten überrascht? Welche Gefahren seht ihr?

MIT KÜNSTLICHER INTELLIGENZ ARBEITEN

In dieser Lektion ...

- erfahrt ihr, wie Künstliche Intelligenz funktioniert

Schlüsselbegriffe

- Datensatz
- Vorhersage

KI hat 3 grundlegende Bestandteile:

Damit KI funktioniert, braucht sie eine Menge Daten, aus denen sie lernen kann. Der technologische Fortschritt hat es ermöglicht, mehr Informationen schneller als je zuvor zu sammeln. Das ist eine Sache, die KI jetzt möglich macht.
Wie ihr erfahren habt, lernt KI aus Daten und findet von sich aus Muster. Damit kann sie Vorhersagen treffen. Denkt über euch selbst nach: Welche Art von Vorhersagen machen Menschen? Welche Daten und Muster berücksichtigen sie bei diesen Vorhersagen?
Manche Menschen versuchen, das Wetter vorherzusagen, andere die Ergebnisse von Sportspielen oder auch, was in einem Film als nächstes passiert.
Daten können in verschiedenen Formen vorliegen, zum Beispiel in Form von Tönen, Zahlen, Texten und Bildern.

Sehen wir uns als Beispiel Google Maps an: Wie nutzt Google Maps KI, um euch den besten Weg zu eurem Ziel zu zeigen?

Schritt 1

Beispiel: Die KI in Google Maps findet den für euch „besten" Weg

Google sammelt folgende Daten (das nennt man auch „Input"), um eine Vorhersage zu machen:

- Aktueller Standort
- Reiseziel
- Verkehrsmittel (zu Fuß, Auto, Fahrrad, öffentliche Verkehrsmittel)
- Verkehrsaufkommen

Schritt 2

Aber: Hört KI an diesem Punkt auf? Kann KI nur Vorhersagen machen oder noch mehr?

Google Maps kann dann für jeden möglichen Weg vorhersagen, wie lange ihr braucht.

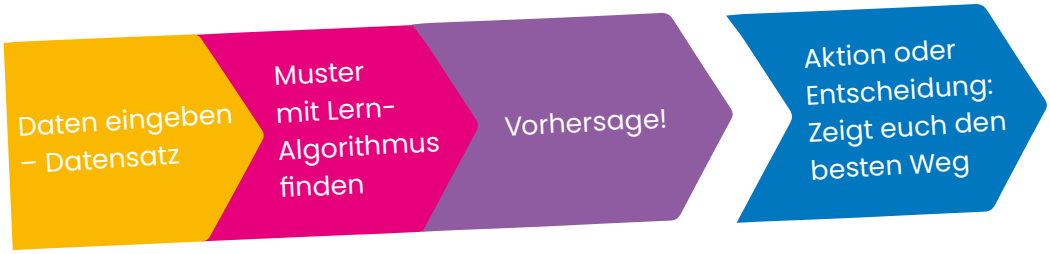
Schritt 3

KI-Technologien nutzen Vorhersagen, um gewisse Dinge zu tun.

Google Maps kann dann mit Hilfe der Vorhersagen Entscheidungen treffen.

Schritt 4

Beispiel: Google Maps findet den für euch „besten" Weg.



Google Maps trifft dann eine Entscheidung mit einer Aktion: Es zeigt euch den besten Weg.

Google Maps kann nur so gute Vorhersagen treffen, weil es so viele Leute benutzen und Google so extrem viele Daten sammeln kann. Viele Daten sind aber nicht das Einzige, was wichtig ist für eine Vorhersage. Eine Vorhersage ist nur dann gut, wenn auch die Daten gut sind.

Warum sind gute Daten wichtig?

KI-Modelle sind programmierte Algorithmen. Sie sollen die menschliche Entscheidungsfindung nachahmen und werden deswegen mit Daten trainiert. KI-Modelle benötigen eine Menge Daten, um gute und genaue Vorhersagen zu treffen. Es gibt drei beliebte Methoden, Daten (z. B. Bilder, Zahlen, Töne oder Text) für das Training und die Verwendung in KI-Modellen zu sammeln.

Hier sind sie:

- Eigene Trainingsdaten aus der Community sammeln
- Daten mit Sensoren oder Benutzereingaben (Informationen, die Benutzer:innen eingeben, z. B. Alter) sammeln: Es gibt Geräte, die selbstständig Daten sammeln können, wie zum Beispiel ein Smartphone, wenn das GPS aktiviert ist
- Daten aus öffentlichen Datensätzen als Trainingsdaten verwenden

Doch was macht einen „guten" Datensatz eigentlich aus?

Hierzu drei wichtige Punkte:

1. Es wird eine sehr große Datenmenge benötigt
2. Es müssen sehr viele unterschiedliche Daten sein
3. Die Art der Daten sollte stimmen

Ein Beispiel zur Erklärung: Stellt euch vor, ihr möchtet ein KI-Modell entwickeln, das erkennen soll, ob jemand eine Mund-Nasen-Maske trägt. Dazu solltet ihr viele Bilder sammeln, die verschiedene Beispiele abbilden:

- Verschiedene Arten und Farben von Mund-Nasen-Bedeckungen
- Unterschiedliche Menschen – Geschlecht, Aussehen, Alter
- Verschiedene Bildhintergründe – drinnen, draußen, hell, dunkel
- Unterschiedliche Kopfhaltung im Bild – nah, fern, links, rechts

Denn: Was passiert, wenn ihr euer Modell nur mit Bildern von weißen Männern mit blauen OP-Masken trainieren würdet? Was geschieht, wenn euer Modell dann entscheiden soll, ob eine Frau mit einer dunkleren Hautfarbe, die eine lilafarbene Maske trägt, eine Maske trägt oder nicht? Würde das Modell dann erkennen, dass die Frau eine Maske trägt? Wahrscheinlich nicht. Es wäre besser, das Modell in der Trainingsphase mit vielen unterschiedlichen Bildern zu füttern.

Und selbstverständlich muss ein Datensatz die richtige Art von Daten haben. KI-Modelle können mit Zahlen, Text, Bildern und Tönen trainiert werden. Das heißt, wenn euer Modell erkennen soll, ob jemand eine Maske trägt, hilft es nicht, wenn ihr das Modell mit Tönen trainiert. Indem ihr darüber entscheidet, welche Informationen in eurem Datensatz sind, habt ihr einen sehr großen Einfluss auf das KI-Modell. Achtet unbedingt darauf, dass ihr eine große Menge verschiedener Daten und die richtige Datenart verwendet. Sonst besteht die Gefahr, dass euer KI-Modell nicht sonderlich genau ist. Das würde höchstwahrscheinlich auch dazu führen, dass es falsche Vorhersagen und Entscheidungen trifft. Nehmt euch also genügend Zeit für die Sammlung der Daten.

Aktivität: KI-Modelle ausprobieren

Ihr braucht:

- Stifte
- Internet

Schaut euch mindestens zwei der drei folgenden Websites an und erhaltet einen Eindruck von dem, was KI leisten kann.

LINK

Imaginary Soundscape: Stellt euch vor, ihr seid auf Reisen und besucht eine andere Stadt. Die meisten Menschen stellen sich dann vor, wie dieser Ort aussehen würde. Aber habt ihr schon einmal darüber nachgedacht, wie er sich anhören würde? Dieses KI-Modell tut genau das. Basierend auf einem Bild erzeugt die KI das, was sie "glaubt", was ihr hören würdet, wenn ihr tatsächlich dort wärt.

LINK

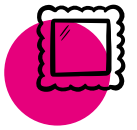
AutoDraw: Wie oft hattet ihr beim Zeichnen schon ein klares Bild im Kopf, aber das Ergebnis war ganz anders als ihr es euch vorgestellt habt? Diese KI nimmt eure Kritzeleien und sagt voraus, was ihr zu zeichnen versucht. Probiert es mal aus!

LINK

X Degrees of Separation: Diese Anwendung gibt euch einen Eindruck davon, wie eine KI „denkt". Dazu nimmt sie zwei Kunstwerke und zeigt uns eine Mischung aus ähnlichen Kunstwerken, die beide miteinander verbindet.

1) Überlegt euch, welche Daten dort verwendet wurden, um die KI zu trainieren? Könnt ihr euch vorstellen, für welche anderen Anwendungen man diese Daten auch verwenden könnte?

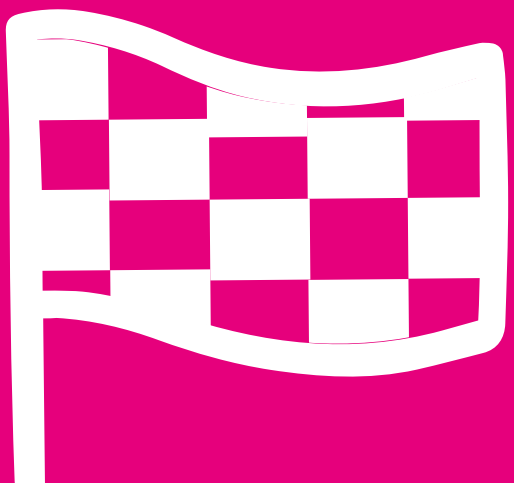
Name	Daten	andere Verwendung



Reflexion

Nachdem ihr nun einen ersten Einblick in das Thema Künstliche Intelligenz erhalten habt, könnt ihr darüber nachdenken, wie KI nützlich sein könnte, um euer ausgewähltes Problem zu lösen. Denkt an die drei Teile der KI: Daten, Muster, Vorhersage. Wie würdet ihr alle drei in eurer Lösung berücksichtigen?

PROJEKTZIEL



EURE EIGENE APP

In dieser Lektion ...

- schaut ihr auf euer Projekt und die Entwicklung eurer Idee zurück und reflektiert, was ihr gelernt habt

Herzlichen Glückwunsch! Ihr habt eure eigene App programmiert, mit der ihr ein soziales Problem löst und den Menschen helft. In dieser letzten Lektion schaut ihr noch einmal auf eure Arbeit zurück und reflektiert im Team eure gemeinsame Reise von der Idee zur fertigen App.

Aktivität: Auf euer Projekt zurückblicken

Ihr braucht:

- Stifte

Erinnert euch daran, als ihr als Team aufgebrochen seid und was ihr in der Zwischenzeit alles erreicht habt, um euer Projekt zum Laufen zu bringen und abzuschließen. Es ist wahrscheinlich, dass ihr selbst, euer Team und eure Appidee sich im Laufe des Prozesses verändert hat. Setzt euch zusammen, um rückblickend zu besprechen, wie ihr und euer Projekt euch entwickelt habt.

Stellt euch dazu folgende Fragen:
Wie habt ihr das Projekt oder eure Ideen nach der Nutzer:innenforschung, Konkurrenzanalyse oder den Testphasen verändert?

Was für technisches Wissen habt ihr während der Bearbeitung des Handbuchs gesammelt?

Wie hat sich euer Team weiterentwickelt? Was waren die Schwierigkeiten? Woran seid ihr als Team gewachsen?

Gab es Herausforderungen, die ihr nur gemeinsam bewältigen konntet? Welche waren das und wie habt ihr sie gelöst?

Habt ihr euch durch das Projekt verändert? Worin besteht die Änderung?

Was würdet ihr beim nächsten Mal anders machen?

Reflexion

Gibt es sonst noch Punkte, an die ihr euch gerne erinnert möchtet?
Ihr könnt zum Beispiel auch noch Bilder von eurer App und eurem Team hier einfügen oder die Erfolge benennen, auf die ihr besonders stolz seid.

Glossar

Alarm-Funktion (Englisch: Alert): Funktion in Thunkable, mit der man Fehler im Code findet

Algorithmus: Schritt-für-Schritt-Anleitung, die erklärt, was der Computer tun soll

Anleitung (Englisch: Tutorial): ein Video mit einer Anleitung, wie etwas funktioniert

App-Builder (z. B. Thunkable): Programme, mit denen man Anwendungen (z. B. eine App) erstellen kann, ohne Code eingeben zu müssen. Normalerweise haben sie eine „Drag and Drop"-Schnittstelle, mit der ihr vorgefertigte Codes zusammenfügen könnt

Ausgabe: Informationen, die aus einer Funktion herauskommen

B-Roll-Filmmaterial: Filmmaterial, das nicht zum Hauptthema gehört, aber später in das Video geschnitten werden kann

Bedingte Anweisungen (Englisch: Conditional Statements): bedeutet, dass ein Code oder Befehl nur ausgeführt wird, wenn eine bestimmte Bedingung erfüllt ist

Bedingung (Englisch: Condition): ein Zustand oder eine Situation, in der sich etwas befindet

Benutzeroberfläche: alle Elemente in einer App, mit denen ein:e Nutzer:in interagieren kann

Bildschirmaufnahme: Videoaufzeichnung der verschiedenen App-Funktionen, die in der Regel mit einem gesprochenen Text erklärt wird

Boolean: ein Datentyp, der einer von zwei Werten (wahr oder falsch) sein kann

Brainstorming: eine Möglichkeit, schnell viele Ideen zu sammeln

Code: die Art und Weise, wie Menschen mit Computern sprechen und sie dazu bringen, verschiedene Dinge zu tun

Daten: Informationen, die eine App verstehen und nutzen kann

Datenbank: eine organisierte Sammlung von Informationen

Datenbankaufruf: Zugriff auf eine Datenbank, dadurch können Informationen in der Datenbank gespeichert, abgerufen, aktualisiert oder gelöscht werden

Datensatz: eine organisierte Sammlung von Daten zu einem Thema

Debugging: eine Fehlersuche – hilft herauszufinden, warum der Code nicht funktioniert

Demo: kurz für "Demonstration", also Vorführung der Technologie und der Funktionen einer App

Dimension: das Ausmaß oder die Größe der Probleme oder Lösungen

Drag and Drop: eine Funktion, mit der ihr etwas anklicken und mit gedrückter Maustaste an eine andere Stelle bewegen könnt

Eigenschaften (Englisch: Properties): Beschreibungen von verschiedenen Komponenten in eurer App

Eingabe: Informationen, die in eine Funktion einfließen

Ereignis (Englisch: Event): etwas, das passiert und die Ausführung des Codes auslöst

Ereignisgesteuerte Programmierung: Programmierung, die auf Ereignissen basiert

Etikett (Englisch: Label): Code-Block zur Programmierung euer App in Thunkable eures Videos und das Hinzufügen von Effekten, Titeln und Ton

Event Handler: Code, der der App mitteilt, was sie tun soll, wenn ein Ereignis eintritt

Fehler (Englisch: Bug): Fehler im Code

Feld (Englisch: Array): eine Möglichkeit der Informatik, Daten zu organisieren

Funktion: ein Code-Block, der eine bestimmte Aufgabe erfüllt - er nimmt eine Eingabe auf und wandelt sie in eine Ausgabe um funktionsfähig ist.

Für-Jede-Schleife (Englisch: For-Each-Loop): wird für jedes Element in einer Liste einmal wiederholt

Für-Schleife (Englisch: For-Loop): wiederholt einen Code-Block eine bestimmte Anzahl von Malen

Geschäftsplan: ein Dokument, in dem genau beschrieben wird, wie ihr Menschen dazu bringt, eure App zu nutzen

Gewinnorientiert (Englisch: For-Profit): ein Unternehmen, das vor allem das Ziel hat, Geld zu verdienen

Funktion: ein Code-Block, der eine bestimmte Aufgabe erfüllt - er nimmt eine Eingabe auf und wandelt sie in eine Ausgabe um

Funktionsfähiges Produkt (Englisch: Minimum Valuable Produkt, MVP): ist eine App, die gerade genug Funktionen hat, um ihre Aufgabe zu erfüllen und von Nutzer:innen getestet zu werden

Gemeinnützig (Englisch: Non-Profit): ein Unternehmen, das andere Ziele hat als Geld zu verdienen und der Gesellschaft nützen soll

Gemeinschaft (Englisch: Community): eine Gruppe von Menschen, die etwas gemeinsam hat

Geschäftsplan: ein Dokument, in dem genau beschrieben wird, wie ihr Menschen dazu bringt, eure App zu nutzen

Gewinn (Englisch : Profit): Geld, das ein Unternehmen verdient, wenn alle Kosten für den Betrieb des Unternehmens abgezogen sind

Gewinnorientiert (Englisch : For-Profit): ein Unternehmen, das vor allem das Ziel hat, Geld zu verdienen

If/Else/Else If (wenn / sonst / sonst wenn bedingte Anweisungen): Anweisungen, die mehr als eine Bedingung haben

If/Else (wenn/sonst): häufige Form von „Conditional Statements" in der Programmierung. Sie sagt dem Computer Folgendes: Wenn die Bedingung wahr ist, tut er etwas. Sonst (d. h. wenn die Bedingung falsch ist) tut er eine andere Sache

Index: eine Zahl, die angibt, wo sich eine Dateneinheit in einer Liste befindet

Innovation: ein neues Produkt, das der Welt einen Mehrwert bringt oder eine neue Art und Weise, Dinge zu tun

KI-Modell: ein KI-Modell ist eine Art Computerprogramm, das darauf trainiert wird, bestimmte Dinge zu lernen

Kommentare: Text, der im Code enthalten ist und erklärt, was der Code macht

Komponenten (Englisch: Components): Teile eurer App

Konkurrenz: Personen oder Unternehmen, die ähnliche Dinge anbieten wie ihr mit eurer App

Künstliche Intelligenz (KI): Maschinen/Programmierung, die Aufgaben übernehmen können/kann, die traditionell eher von Menschen erledigt werden

Listen: eine Möglichkeit, mehrere Daten in einer App zu organisieren

Logik-Operatoren: ermöglichen es Computern, Entscheidungen auf der Grundlage mehrerer Bedingungen zu treffen

Logo: ein Symbol, das ein Unternehmen oder ein Business repräsentiert

Lokale Datenbank: eine organisierte Sammlung von Informationen, die auf einem Gerät gespeichert sind

Lokale Speicherung (auch lokale Datenbank): eine Möglichkeit, Informationen auf einem Gerät zu speichern. Man spricht hier auch von einer lokalen Datenbank

Lösung: eine App, die ein Problem angeht

Marketing: Menschen von eurem Business erzählen und sie dazu bringen, euer Produkt zu nutzen

Medienkomponenten: Beispiele für Medienkomponenten sind Foto, Audio und Video

Mission (Englisch: Mission Statement): ein Leitbild der Werte eines Unternehmens, einer Organisation oder Einzelperson

Mobile App: ein Programm, das auf einem mobilen Gerät installiert ist (z. B. Smartphone, Tablet)

Muster: eine Regelmäßigkeit, die von einer KI in Daten erkannt wird

Nachbearbeitung/Postproduktion (Englisch: post Production): die Bearbeitung eines aufgenommenen Videos und das Hinzufügen von Effekten, Titeln und Ton

Nicht (Englisch: not): gibt das Gegenteil der eingegebenen Bedingung aus

Nutzer:innenforschung: die Zielgruppe oder die Nutzer:innen eures Produkts analysieren, um ihre Wünsche und Bedürfnisse zu verstehen

Oder (Englisch: or): ein Logik-Operator, der “wahr” anzeigt, wenn mindestens eine der Eingabebedingungen wahr ist

Pitch: eine kurze Präsentation einer Idee, damit man andere davon überzeugt

Prioritätensetzung: sich entscheiden, was am wichtigsten ist

Produkt-Video: die Präsentation euer App und ihrer Funktionen in Form eines Videos

Produkt: wird von Unternehmen den Kund:innen auf dem Markt angeboten. In eurem Fall ist das eine App.

Produktion: die Erstellung von etwas, Videoproduktion heißt also die Erstellung von Videomaterial

Produktpräsentation: eine Präsentation, in der ihr eure App und ihre Funktionen vorstellt

Programmiersprache: ein Weg, mit einem Computer oder Smartphone zu sprechen und ihnen zu sagen, was sie tun sollen

Projekt Canvas: ein Werkzeug zur Planung eines Projekts

Prototyp: ist die allererste Version eurer App

Pseudocode: Schreiben eines Algorithmus in einfacher Sprache statt in Code

Schaltfläche (Englisch: Button): ein grafisches Element, mit dem ihr eine digitale Benutzeroberfläche wie eure App oder Thunkable steuern könnt.

Schleifen (Englisch: Loops): damit lässt sich ein Computer anweisen, etwas viele Male hintereinander zu tun (einen Code-Block viele Male hintereinander auszuführen)

Schlüssel: eindeutiger Name einer zugeordneten Variablen in Datenbanken

Schriftart: eine bestimmte Schrift, die einen eigenen Namen hat

Schriftschnitt: ein bestimmter Stil einer Schriftart (z. B. kursiv)

Screenshots: statische (sich nicht bewegende) Bilder, die zeigen, wie die App auf dem Smartphone aussehen wird

Sensoren: Hardware auf dem Smartphone, die es ihm ermöglicht, mit der Welt um es herum zu interagieren und Daten für verschiedene Zwecke zu sammeln

Serifen: eine feine Linie, die einen Buchstabenstrich abschließt

Serifenlose Schrift: Schrift ohne Serifen (z. B. Arial)

Serifenschrift: Schrift mit Serifen (z. B. Times New Roman)

Soziale Komponenten: Funktionen, die es Nutzer:innen ermöglichen, über eure App zu telefonieren, E-Mails zu verschicken, SMS zu schreiben und Inhalte zu teilen

Sozialunternehmen: ein Unternehmen, bei dem nicht der Gewinn im Vordergrund steht, sondern z. B. die Lösung eines sozialen Problems

Storyboard: ein Storyboard hilft euch bei der Entwicklung eines überzeugenden Pitchs für euer Projekt und bei der Vorbereitung einer überzeugenden Geschichte zu eurer Idee

Strategie: eine Abfolge von Schritten, um ein Ziel zu erreichen

Team: Gruppe von Menschen, die gemeinsam an einer Aufgabe arbeiten

Testdaten: ein einfacherer Datensatz, mit dem ihr sicherstellen könnt, dass eure App korrekt funktioniert

Testen: euren Prototypen der App an anderen Menschen ausprobieren.

Und (Englisch: and): ein Logik-Operator, der “wahr” anzeigt, wenn alle Eingabebedingungen wahr sind

Unternehmen (Englisch: Business): eine Organisation oder Person, die etwas im Tausch gegen Geld oder ein anderes Gut tut

Variablen: eine Dateneinheit, die sich ändern kann

Verbindungskomponenten: ermöglichen eurer App die Interaktion mit Dingen außerhalb eurer App, wie z. B. dem Internet und anderen Apps

Versionskontrolle: Speichern von Zwischenversionen eurer App, während ihr daran weiterarbeitet

Vorhersage: eine zukünftige Tatsache, die eine KI aufgrund bestimmter Datenmuster voraussagt

Vorproduktion: Vorbereitung und Planung eines Videos vor Beginn der Aufnahme

Wettbewerbsforschung: Informationen über eure Konkurrenz sammeln

Während-Schleife (Englisch: While-Loop): wiederholt einen Code-Block, bis eine Bedingung nicht mehr erfüllt ist

Zahl: ein Datentyp, der eine Zahl verwendet

Zeichenkette (Englisch: String): ein Datentyp, der Zeichen verwendet

Zeitplan: ein Plan, der euch bei eurem Projekt hilft, und der festlegt, was bis wann erledigt sein soll

Ziele für nachhaltige Entwicklung (auf Englisch: Sustainable Development Goals, SDGs): Zielsetzungen der Vereinten Nationen, z. B. um Armut zu reduzieren oder der Umwelt zu helfen

Zielgruppe: Menschen, die euer Produkt nutzen werden