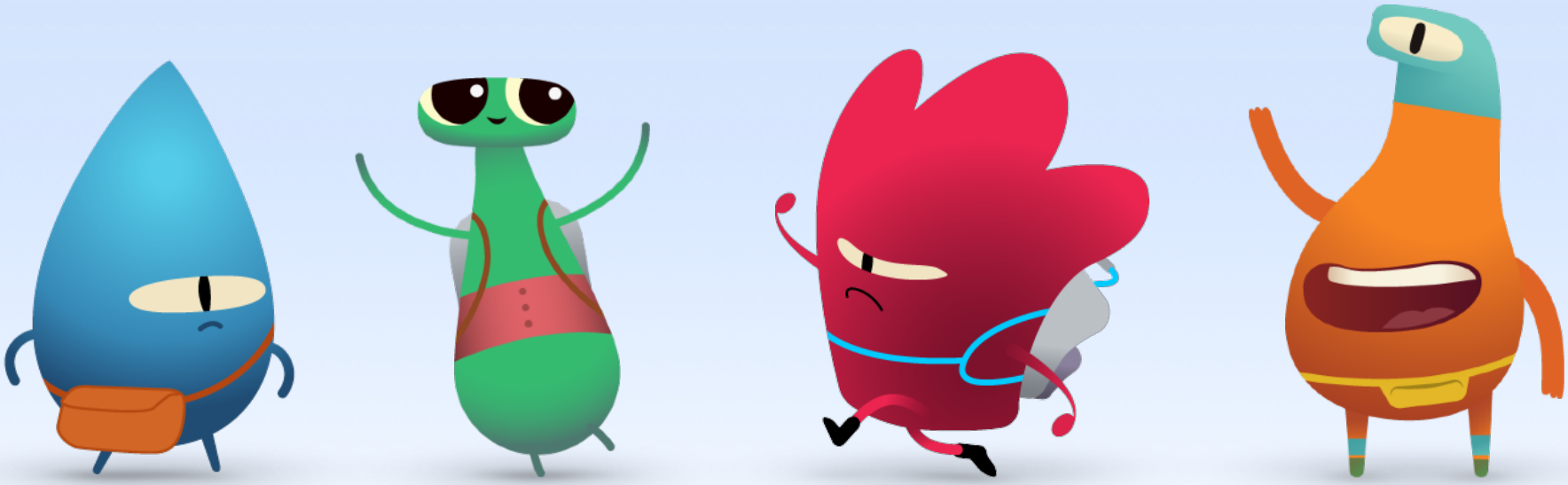
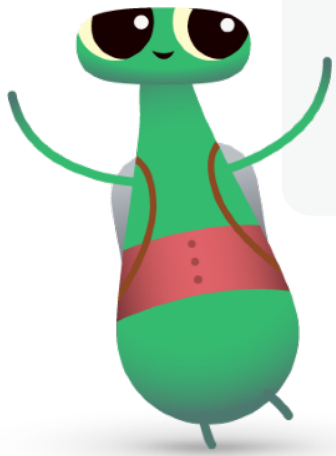
Swift Coding Club



Swift Playgrounds

Programmieraktivitäten

Themen



Willkommen beim Swift Coding Club!

Diese Programmieraktivitäten behandeln grundlegende Programmierkonzepte, die zu den App-Design-Aktivitäten passen. Mithilfe dieser Aktivitäten baust du nicht nur deine Programmierfähigkeiten auf, sondern beginnst auch zu verstehen, wie Apps funktionieren. Dadurch kannst du bessere Apps entwickeln.

Bei jedem Thema lernst du in einer kurzen Einführungsaktivität etwas über bestimmte Programmierkonzepte. Dann kannst du diese Konzepte in Swift Playgrounds anwenden, um Rätsel zu lösen.

Unter „Das Gelernte vertiefen“ findest du zu jedem Thema weitere Aktivitäten, mit denen du mehr über die Konzepte lernen kannst. Diese zusätzlichen Aktivitäten sind freiwillig und du kannst wählen, ob du eine, beide oder keine davon machen willst. Für manche dieser Aktivitäten brauchst du ein weiteres Gerät wie einen Roboter oder eine Drohne. Wenn diese Geräte verfügbar sind, bieten sie dir eine tolle Möglichkeit, das Gelernte anzuwenden.

Viel Spaß!

Wie ein Computer denken: Befehle und Sequenzen

1

Einführung (15 Minuten)

Arbeite mit einem Partner zusammen. Einer von euch ist der Ansager, einer der Macher. Der Ansager lässt sich etwas einfallen, das der Macher machen kann, z. B. einen Smiley an die Tafel zeichnen oder 5-mal den Hampelmann machen. Der Ansager darf dem Macher nicht sagen, was das Ziel ist, sondern gibt ihm Schritt-für-Schritt-Anweisungen. Hat der Macher die Aufgabe wie vorgesehen abgeschlossen?

Der Ansager hat dem Macher Befehle in einer Sequenz gegeben. Genau das musst du auch tun, wenn du Code schreibst.

Schau dir dieses Video an, um mehr über Befehle zu erfahren, und dieses hier zum Thema Debugging.

Denk jetzt noch einmal über die Anweisungen nach – insbesondere dann, wenn der Macher die Aufgabe nicht wie vorgesehen abschließen konnte. Hat in den Anweisungen vielleicht ein Schritt gefehlt? Oder wäre es klarer gewesen, wenn die Reihenfolge einiger Schritte geändert worden wäre? Über so etwas nachzudenken, wird als „Debugging" bezeichnet. Programmierer debuggen oft, um Fehler in ihrem Code zu beheben und ihn zu verbessern.

Übung (30 Minuten)

Löse jetzt in Swift Playgrounds: Programmieren lernen 1 die Rätsel, die in der Liste rechts ein grünes Häkchen haben.

Denk darüber nach: Was fällt dir auf, wenn du die Befehle, die du in der App verwendet hast, mit denen vergleichst, die der Ansager gegeben hat?

Befehl: Eine bestimmte Aktion, die der Computer ausführen soll.

Sequenz: Die Reihenfolge, in der die Befehle gegeben werden.

Debugging: Das Suchen und Beheben von Fehlern.

Befehle

- Einführung ✓
- Befehle erteilen ✓
- Einen neuen Befehl hinzufügen ✓
- Einen Schalter betätigen ✓
- Portalübung ✓
- Bugs finden und beheben ✓
- Übung zum Beheben von Bugs ✓
- Der kürzeste Weg ✓

Wie ein Computer denken: Befehle und Sequenzen

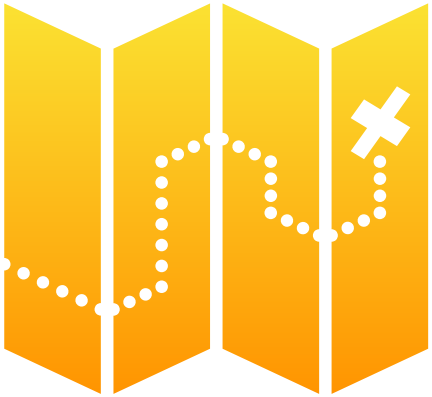
Das Gelernte vertiefen

1

Verstecken (1 Sitzung)

1. Verstecke im oder in der Nähe des Zimmers einen kleinen Gegenstand.
2. Stell dich an einen Ort, an dem du mit der iPad Kamera aufnimmst, wie du jemandem Anweisungen für den Weg zum versteckten Gegenstand gibst. Die Anweisungen sollten von dort, wo du stehst, beginnen.
3. Tausche dein Video jetzt mit einem anderen Schüler. Schau dir sein Video an und versuche, den versteckten Gegenstand zu finden. Hast du ihn gefunden?

Denk darüber nach: Wie hätten die Anweisungen verbessert werden können? Müssen die Anweisungen debuggt werden? Wenn ja, wie?



Dash (1 Sitzung)

1. Falls der Dash Roboter von Wonder Workshops verfügbar ist, lade die Dash Lektion in Swift Playgrounds.

2. Verwende Befehle, um Dash durch den Renntag zu steuern.
3. Der Weg in „Ab zum Rennen" kann ziemlich knifflig sein. Vergleiche deinen Code mit dem anderer Schüler. Falls nötig, überprüft gegenseitig euren Code und helft einander beim Debugging.
4. Schaut, wie weit ihr kommt. Du kannst immer wieder zu dieser Lektion zurückkehren, wenn du weitere Programmierkonzepte kennengelernt hast.
5. Wenn du bereit bist, kannst du deine eigene Geschichte mit Dash als Hauptcharakter erzählen.

Denk darüber nach: Welche von Dashs Sensoren hast du verwendet, um deine Geschichte zu erzählen? Wie haben diese Sensoren zur Geschichte beigetragen?

Effizient denken: Funktionen und ein paar Schleifen

2

Einführung (5 Minuten)

Denk dir ein paar Tanzbewegungen aus. Beschreibe die Tanzbewegungen jemand anderem, oder noch besser, lass sie jemand anderen ausführen. Wie einfach war es, die Bewegungen zu beschreiben?

Beim Programmieren ist es manchmal einfacher, vorhandene Befehle zu kombinieren und so ein neues Verhalten zu schaffen. Dieser Vorgang wird als „Komposition" bezeichnet. Wenn du diesem neuen Verhalten einen Namen gibst, damit du es in Zukunft wiederverwenden kannst, dann hast du eine Funktion. Wenn du ein Programm anweist, eine Funktion auszuführen, nennen wir das einen „Aufruf". Wenn also jemand sagt „Tanz die Macarena", ruft er die Funktion „Macarena" auf.

Schau dir dieses Video zu Funktionen und Schleifen an.

Übung (40 Minuten)

Löse jetzt in Swift Playgrounds: Programmieren lernen 1 die Rätsel, die in der Liste rechts ein grünes Häkchen haben.

Denk darüber nach: Wie viele Aktionen hat der Charakter in einem bestimmten Rätsel ausgeführt? Und wie viele Befehle hast du geschrieben? Wann und warum solltest du Funktionen und Schleifen erstellen?

Funktion: Eine Reihe von Befehlen, die unter einem Namen zusammengefasst werden.

for-Schleife: Führt einen Codeblock eine vorgegebene Anzahl Mal aus.

Funktionen

- Einführung ✓
- Ein neues Verhalten zusammenst... ✓
- Eine neue Funktion erstellen ✓
- Einsammeln, Auslösen, Wiederholen
- Über das Brett
- Muster verschachteln ✓
- Tolle Treppen ✓
- Schatzsuche

for-Schleifen

- Einführung ✓
- Schleifen verwenden ✓
- Eine Schleife für alle Seiten ✓
- An den Rand und zurück
- Schleifenspringer
- Ausbreiten
- Edelsteinfarm
- Vier Lager leerräumen

Effizient denken: Funktionen und ein paar Schleifen

Das Gelernte vertiefen

2

Mustermacher (1 Sitzung))

1. Verwende eine Zeichenapp wie Art Set oder Pages, um ein Muster zu erstellen, das aus verschiedenen Formen, Objekten und Farben besteht. Das Muster darf so lang sein, wie du möchtest.
2. Schreibe dein Muster in der App zwanzigmal mit Wörtern auf – z. B. „rot, gelb, blau; rot, gelb, blau …“ und so weiter.
3. Gib dem Teil des Musters, der sich wiederholt, einen Namen (z. B. „rot, gelb, blau“ = „Primärfarben“). Schreibe das Muster dann nochmals in Wörtern, diesmal aber nur mit dem neuen Namen.
4. Wie viel weniger Arbeit bzw. Schritte brauchst du jetzt, um das Muster zu schreiben?
5. Wie oft wiederholt sich „Primärfarben“? Beschreibe dein Muster jetzt in einem Schritt. Du hast eine for-Schleife erstellt!

Denk darüber nach: Was hat diese Aktivität mit Programmieren zu tun?

Roboterinterview (1 Sitzung)

1. Lade den Ausgangspunkt „Antworten“ in Swift Playgrounds.

2. Auf der Seite „Text“ sind „show“ und „ask“ Funktionen. Tippe auf „Meinen Code ausführen“, um deinen Namen einzugeben, und tippe dann auf „Senden“, um zu sehen, was passiert. Funktionen können ein Ergebnis haben, das man in der Live-Ansicht sieht.
3. Erkunde auf der Seite „Typen“ verschiedene Arten von show- und ask-Funktionen.
4. Arbeitet in Zweiergruppen und schreibt euch gegenseitig eine Reihe von show- und ask-Funktionen, die ihr vervollständigen müsst.
5. Nutzt die Ergebnisse der Funktionen, um eine Geschichte, ein Interview oder eine kurze Biografie zu schreiben.

Denk darüber nach: Was wäre passiert, wenn du deine show- und ask-Funktionen in einer anderen Sequenz geschrieben hättest? Wir würde sich das auf die Geschichte oder das Interview auswirken?

Logisch denken: Bedingte Anweisungen

3

Einführung (10 Minuten)

Spielt als Gruppe einige Runden des Spiels „Ich sehe was, was du nicht siehst“. In diesem Spiel wählt ein Spieler einen Gegenstand aus, von dem er nur eine Eigenschaft nennt. Die Mitspieler müssen sich umsehen und erraten, was gemeint ist. Wer den Gegenstand errät, stellt das nächste Rätsel.

Welche Entscheidungen musstet ihr treffen? Was war euer Denkprozess, während ihr herauszufinden versucht habt, was gemeint war? In diesem Spiel habt ihr in Bedingungen gedacht.

Schau dir dieses Video an, um mehr über bedingte Anweisungen zu erfahren.

Übung (35 Minuten)

Löse jetzt in Swift Playgrounds: Programmieren lernen 1 die Rätsel, die in der Liste rechts ein grünes Häkchen haben.

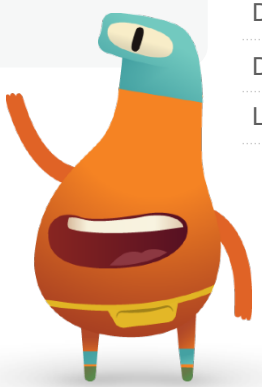
Denk darüber nach: Was für Entscheidungen traf dein Code bei der Ausführung der if-Anweisung? Wie hast du for-Schleifen und if-Anweisungen kombiniert? Warum?

Bedingung: Etwas, das man prüft und das wahr oder falsch ist.

Bedingter Code: Ein Codeblock, der nur dann ausgeführt wird, wenn etwas wahr ist.

Boolescher Wert: Ein Wert, der nur entweder wahr oder falsch sein kann.

Logische Operatoren: Ein Symbol oder Wörter wie „und“, „oder“, und „nicht“.



Bedingte Anweisungen

- Einführung ✓
- Nach Schaltern suchen ✓
- „else if“ verwenden ✓
- Schleifen mit bedingten Anweisungen ✓
- Bedingtes Klettern
- Definieren intelligenter Funktionen ✓
- Eingesperrt
- Entscheidungsbaum

Logische Operatoren

- Einführung ✓
- Den „NOT“-Operator verwenden ✓
- Die NOT-Spirale
- Dies UND das überprüfen ✓
- Dies ODER das überprüfen ✓
- Logisches Labyrinth

Logisch denken: Bedingte Anweisungen
Das Gelernte vertiefen

3

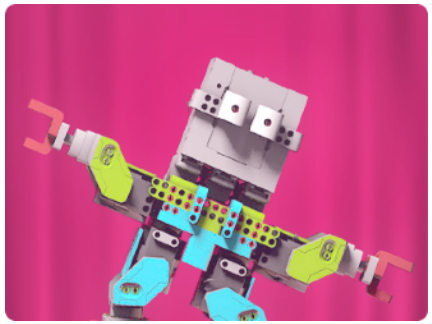
Schnitzeljagd (1 Sitzung)

1. Jeder Schüler schreibt zwei Bedingungen auf je ein Stück Papier – z. B. „Es ist rechteckig.“ oder „Es beginnt mit einem C.“ – und legt den Zettel dann in einen Hut.
2. Bildet kleine Gruppen, zieht in der Gruppe zwei Zettel aus dem Hut und macht jeweils drei bis fünf Fotos von Gegenständen im Raum, die die jeweilige Bedingung erfüllen.
3. Erstellt ein Fotoalbum in einer Präsentationsapp wie Keynote und legt für jede Bedingung einen Abschnitt an. Benennt die Bedingungen aber noch nicht.
4. Zeigt euer Fotoalbum anderen Gruppen und seht, ob sie eure Bedingungen erraten können. Wenn sie richtig raten, fügt ihr der Seite im Album eine bedingte Anweisung hinzu, z. B. „Wenn blau, dann Foto machen“.

Denk darüber nach: Gab es Fälle, in denen es schwierig zu beurteilen war, ob ein Foto passte oder nicht? Wie würde ein Computer mit solchen Fällen umgehen?

MeeBot Tänze 1 Session

1. Wenn du den MeeBot Roboter von UBTECH hast, lade die Lektion „MeeBot Tänze“ in Swift Playgrounds.



2. Schau dir die Lektion an, um zu erfahren, wie du Funktionen, for-Schleifen und bedingte Anweisungen einsetzen kannst, um den MeeBot zum Tanzen zu bringen.
3. Erstelle deine eigene Tanzroutine. Auf der letzten Seite kannst du sogar deine eigene Musik auswählen.

Denk darüber nach: Wie hast du Code eingesetzt, um das Tempo der Musik aufzugreifen?

Wieder und wieder denken: while-Schleifen

4

Einführung (5 Minuten)

In Thema 2 hast du Funktionen und for-Schleifen kennengelernt. Welchen Tanz habt ihr zusammen vorgeführt? Wie würdest du die Funktion für den Tanz mit for-Schleifen schreiben?

Stell dir nun vor, du wolltest den Tanz auf einem Schulfest tanzen. Woher wüsstest du, wann du am Ende des Songs aufhören musst zu tanzen?

Du würdest eine bedingte Anweisung verwenden: „Wenn der Song läuft, dann tanzen." Oder klarer, eine while-Schleife: „Solange der Song läuft, tanzen."

Das ist anders als bei der for-Schleife, die einem Computer sagt, wie oft er einen Codeblock ausführen soll (z. B., 10-mal im Kreis tanzen). Eine While-Schleife sagt einem Computer, er solle einen Codeblock ausführen, bis etwas geschieht (z. B., im Kreis tanzen, bis der Song aufhört zu spielen).

Schau dir dieses Video an, um mehr über while-Schleifen zu erfahren.

Übung (40 Minuten)

Löse jetzt in Swift Playgrounds: Programmieren lernen 1 die Rätsel, die in der Liste rechts ein grünes Häkchen haben.

Denk darüber nach: Wann hast du for-Schleifen und wann while-Schleifen verwendet? Wie hast du das entschieden?

while-Schleife: Eine Schleife, die einen Codeblock ausführt, solange eine Bedingung wahr ist. Sobald die Bedingung falsch ist, wird die Schleife nicht weiter ausgeführt.

while-Schleifen

- Einführung ✓
- Code ausführen, solange ... ✓
- Intelligentere while-Schleifen ers... ✓
- Das richtige Werkzeug auswählen ✓
- Vier mal Vier
- Umgedreht
- Schlaraffenland
- Schleifen verschachteln ✓
- Zufällige Rechtecke
- Du hast immer Recht

Wieder und wieder denken: while-Schleifen

Das Gelernte vertiefen

4

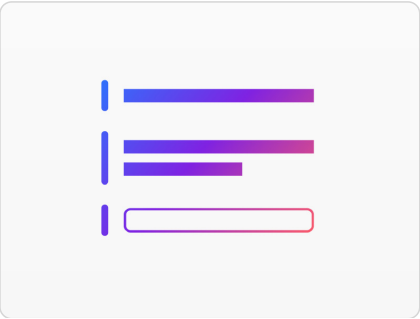
Wieder Verstecken 1 Sitzung

1. Verstecke im oder in der Nähe des Zimmers einen kleinen Gegenstand.
2. Stell dich an einen Ort, an dem du mit der iPad Kamera aufnimmst, wie du jemandem Anweisungen für den Weg zum versteckten Gegenstand gibst. Die Anweisungen sollten von dort, wo du stehst, beginnen. Verwende diesmal wo immer möglich Funktionen sowie for- und while-Schleifen.
3. Tausche dein Video jetzt mit einem anderen Schüler. Schau dir sein Video an und versuche, den versteckten Gegenstand zu finden. Hast du ihn gefunden?
4. Schaut euch die beiden Videos zusammen an. Notiert euch, wo ihr in den Anweisungen for- und while-Schleifen verwendet habt.

Denk darüber nach: Haben die Schleifen die Aufgabe diesmal leichter gemacht? Gab es Momente, in denen sie die Aufgabe schwieriger gemacht haben?

21 Fragen (1 Sitzung)

1. Schauen wir uns noch einmal den Ausgangspunkt „Antworten“ an. Lade diese Version davon.



2. Diese Vorlage für „Antworten“ ist so eingerichtet, dass du mit einem Partner „21 Fragen“ spielen kannst. Schaut euch zuerst den Code an. Was wird er tun?
3. Einer von euch lässt sich nun einen Gegenstand oder eine Sache einfallen und gibt sie als Antwort in den Code ein. Der anderer kann ihm dann 21 Ja/Nein-Fragen stellen.
4. Nach jeder Frage kann der Fragesteller raten und seine Antwort in der Live-Ansicht eingeben, um zu prüfen, ob sie stimmt. Achtet darauf, dass der Schüler, der die Antwort festgelegt hat, den Playground mit der Live-Ansicht im Vollbildmodus weitergibt. Sonst kann der Fragesteller den Code und die richtige Antwort sehen.

Denk darüber nach: Welche Programmierkonzepte hat dieser Playground verwendet? Wie hat er sie verwendet?

Dieselbe Idee denken: Algorithmen

5

Einführung (5 Minuten)

Nennt als Gruppe Dinge, die ihr oft macht und für deren Ausführung mehrere Schritte erforderlich sind – z. B. Zähneputzen oder ein Brötchen schmieren. All dies sind Algorithmen.

Wählt ein Beispiel aus. Mehrere Schüler sollen nun Anweisungen geben, wie sie das tun. Waren alle Anweisungen gleich? Worin unterschieden sie sich? Führten sie letztendlich alle zum selben Ergebnis?

Schau dir dieses Video an, um mehr über Algorithmen zu erfahren.

Übung (40 Minuten)

Löse jetzt in Swift Playgrounds: Programmieren lernen 1 die Rätsel, die in der Liste rechts ein grünes Häkchen haben.

Denk darüber nach: Wie viele verschiedene Arten gibt es, die einzelnen Rätsel zu lösen? Wer hatte den kürzesten Algorithmus? Wer den interessantesten?

Algorithmus: Eine Reihe von Schritt-für-Schritt-Regeln oder -Anweisungen.

Pseudocode: Eine informelle, für den Menschen bestimmte Beschreibung eines Codes oder Konzepts.

Algorithmen

- Einführung ✓
- Die Rechte-Hand-Regel ✓
- Deinen Algorithmus anpassen ✓
- Ein Labyrinth durchqueren ✓
- In welche Richtung drehen?
- Rolle nach rechts, Rolle nach links

Dieselbe Idee denken: Algorithmen

Das Gelernte vertiefen

5

Wer ist am größten? (1 Sitzung)

1. Teilt euch in kleine Gruppen auf. Jede Gruppe denkt sich einen Weg oder Algorithmus aus, mit dem sich feststellen lässt, wer am größten ist. Einfach nur Hinschauen zählt nicht!
2. Greift auf das zurück, was ihr gelernt habt, um euren Algorithmus zu entwickeln. Ihr könnt euren eigenen Pseudocode erfinden, um eure Algorithmen mit so vielen Programmierbegriffen wie möglich zu schreiben.
3. Jede Gruppe präsentiert nun ihren Algorithmus und dann sollt ihr in alle zusammen ausführen.

Denk darüber nach: Welcher Algorithmus erschien dir am effizientesten? Was würdest du in eurem Algorithmus ändern, um herauszufinden, wer in der Klasse am kleinsten ist?

Parrot (1 Sitzung)

1. Wenn eine Drohne von Parrot verfügbar ist, lade die Parrot Lektion in Swift Playgrounds.

2. Verwende alle bisher gelernten Programmierfähigkeiten, um die Drohne starten, landen, in alle Richtungen fliegen und akrobatische Manöver ausführen zu lassen.
3. Erstelle am Ende einen Algorithmus für einen Flugplan, der die Drohne von Punkt A nach Punkt B bringt.

Denk darüber nach: Wie hat sich die Geschwindigkeit der Drohne auf deine Flugpläne ausgewirkt?

Wie ein NewsBot denken: Variablen

6

Einführung (5 Minuten)

Stell dir vor, du würdest umziehen. Du würdest vermutlich deine Sachen in Kartons verpacken und diese mit ein oder zwei Wörtern beschriften, die den Inhalt beschreiben. Wenn z. B. ein Karton deine Trophäen enthält, könntest du ihn mit „Trophäen“ beschriften. Dieses Beispiel macht uns drei wichtige Eigenschaften eines Containers klar.

Wenn wir einen Computer programmieren, verwenden wir anstelle von Kartons sogenannte „Variablen“. Variablen sind ähnlich wie Kartons – sie haben eine Beschriftung (einen Namen) und einen Inhalt (einen Wert). Der Inhalt (der Wert) kann sich ändern, aber die Beschriftung (der Name) bleibt immer gleich. Wir können den Wert bzw. Inhalt einer Variablen finden, indem wir die Variable mit dem richtigen Namen bzw. der richtigen Beschriftung finden.

Schau dir dieses Video an, um mehr zu erfahren.

Übung (40 Minuten)

Löse jetzt in Swift Playgrounds: Programmieren lernen 2 die Rätsel, die in der Liste rechts ein grünes Häkchen haben.

Denk darüber nach: Wie hat dir der Einsatz von Variablen in der App geholfen?

Variable: Ein benannter Container, der einen Wert enthält. Der Wert kann sich mit der Zeit verändern.

Variablen

- Einführung ✓
- Den Überblick behalten ✓
- Den Wert hochtreiben
- Einen Wert erhöhen ✓
- Auf der Suche nach sieben Edels... ✓
- Drei Edelsteine, vier Schalter
- Nach gleichen Werten suchen ✓
- Betätige die Schalter
- Gesamtwert berechnen

Wie ein NewsBot denken: Variablen

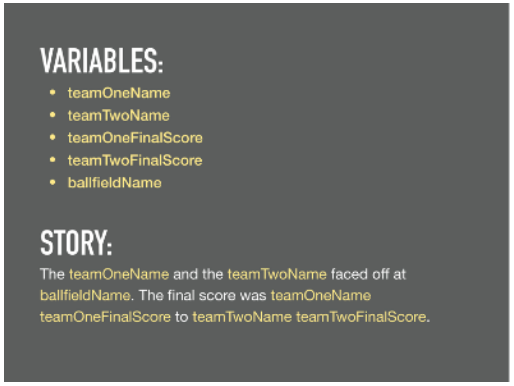
Das Gelernte vertiefen

6

NewsBot (1 Sitzung)

1. Für diese Aktivität erstellst du NewsBot – ein Roboter, der automatisch kurze Artikel schreiben kann. Denke zuerst über eine Nachricht oder ein Sportereignis nach und überlege dir, worüber du schreiben müsstest.
2. Trage vier bis sechs Variablen zusammen, z. B. Teamnamen, Spielergebnis und Datum des Ereignisses. Schreibe in zwei bis drei Sätzen einen kurzen Artikel über das Ereignis und verwende dabei die Variablen, die NewsBot später für ähnliche Ereignisse füllen kann.
3. Such dir einen Partner. Einer von euch liefert nun die Informationen, um alle Variablen im Artikel des anderen auszufüllen. Dann wechselt ihr. Habt ihr nun zwei vollständige Geschichten, die einen Sinn ergeben?

Denk darüber nach: Funktionieren gewisse Variablennamen besser als andere? Warum bzw. warum nicht?



Sphero Arcade (1 Sitzung)

1. Wenn ein Sphero SPRK+ Roboter verfügbar ist, lade die Sphero Arcade Lektion in Swift Playgrounds.

2. Verwende Funktionen und Variablen, um zu zielen, Kollisionen zu erkennen und am Ende deine eigene Version von Pong zu erstellen.
3. Analysiere auf der Seite „Spiele Sphero Pong“ den Code, bevor du ihn ausführst oder bearbeitest. Was machen die einzelnen Befehle?
4. Welche Spielelemente hast du mithilfe von Variablen verändert?

Denk darüber nach: Was für Spiele könntest du noch so modifizieren, wie du es mit Pong gemacht hast?

Wie ein Architekt denken: Typen

7

Einführung (5 Minuten)

Wie viele verschiedene Gebäudetypen gibt es? Wähle einen Typ aus. Was macht ihn einzigartig? In anderen Worten, was sind die Besonderheiten dieser speziellen Art von Gebäude? Was geschieht normalerweise darin? Wir nennen das Verhalten. Eine Schule hat z. B. Klassenzimmer (Besonderheiten) und zwischen den Unterrichtsstunden klingelt es (Verhalten). Sind alle mit den Eigenschaften und Verhalten einverstanden? Warum bzw. warum nicht?

Wenn wir zum Bau eines Gebäudes ein Computerprogramm verwenden, müssen wir sehr präzise sein. Wir müssen den Typ definieren, indem wir die Eigenschaften(*Besonderheiten*) und die Methoden (*Verhalten*) angeben.

Schau dir dieses Video an, um mehr zu erfahren.

Übung (40 Minuten)

Löse jetzt in Swift Playgrounds: Programmieren lernen 2 die Rätsel, die in der Liste rechts ein grünes Häkchen haben.

Denk darüber nach: Welche Typen kamen in der App vor? Was hast du initialisiert?

Typ: Eine benannte Gruppierung von Eigenschaften (die Besonderheiten) und Methoden (das Verhalten) einer Art von Daten.

Initialisierung: Das Erstellen einer neuen Instanz eines Typs, wobei für jede Eigenschaft des Typs der Anfangswert festgelegt wird.

Typen

- Einführung ✓
- Ein Portal deaktivieren ✓
- Portal Ein und Aus
- Das richtige Portal aktivieren ✓
- Ecken der Welt
- Edelsteine zufällig überall verteilen

Initialisierung

- Einführung ✓
- Deinen Experten initialisieren ✓
- Deinen Experten trainieren
- Instanzen verschiedener Typen v... ✓
- Dazu gehören immer zwei

Wie ein Architekt denken: Typen
Das Gelernte vertiefen

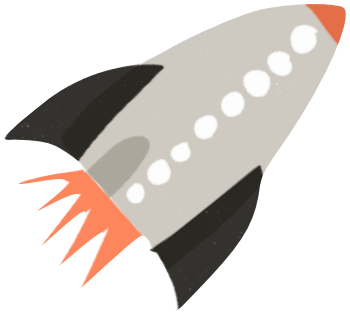
7

Architekt sein (1 Sitzung)

1. Wähle einen Gebäudetyp. Er muss nicht real sein.
2. Überlege dir fünf bis sechs Variablen, mit denen sich das Gebäude beschreiben lässt. Diese Variablen sollen noch keine Werte enthalten, aber sie sollten beschreiben, um was für Werte es geht – z. B. „höhe“ oder „anzahlFenster“.
3. Füge nun neben den Variablen Werte hinzu. Damit beschreibst du ein bestimmtes Exemplar eines Gebäudetyps. In Swift Terminologie hieße das, du initialisierst eine Instanz des Gebäudetyps.
4. Schließe die Initialisierung ab, indem du in einer Zeichenapp wie Notizen den Gebäudetyp skizzierst und dabei die Variablen und Werte wiedergibst.
5. Bildet Zweiergruppen und tauscht die Listen mit den Variablen und Werten eurer Gebäudetypen. Zeichnet die Gebäudetypen eurer Partner und teilt dann eure Zeichnungen miteinander. Wie ähnlich sind sie sich?

Denk darüber nach: Was könnte man tun, damit sich die Zeichnungen ähnlicher werden?

Typ: raketenHaus
anzahlTriebwerke = 4
höhe = 15 Meter
anzahlFenster = 8
farbe = metallisch-grau
formDerTür = abgerundetes Rechteck



Stein, Schere, Papier (1 Sitzung)

1. Lade die Herausforderung „Stein, Schere, Papier“ in Swift Playgrounds.

2. Erkunde die Lektion. Wie funktioniert der Code? Welche Programmierkonzepte erkennst du wieder?
3. Versuche jetzt, eine eigene Version des Spiels zu erstellen, die eigene Regeln und neue Aktionen verwendet. Dafür musst du den Typ des Spiels definieren und diesen mit den zugehörigen Eigenschaften und Methoden initialisieren.

Denk darüber nach: Was waren die Eigenschaften und Methoden im ursprünglichen Spiel? Mit welchen neuen Eigenschaften und Methoden hast du das Spiel personalisiert?