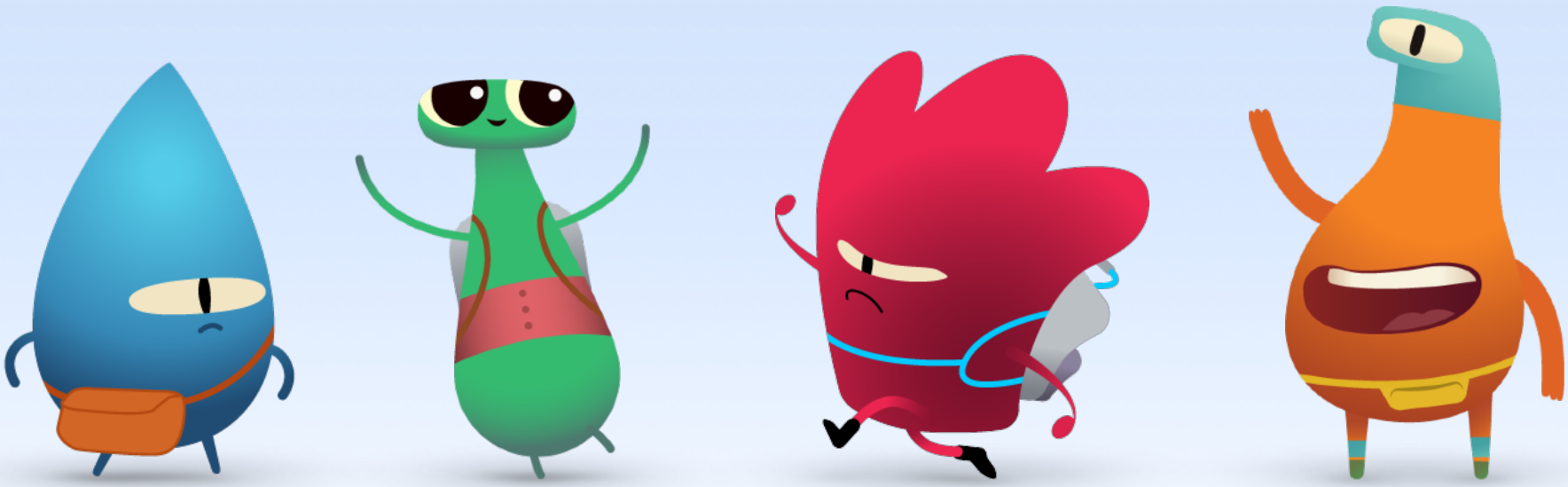
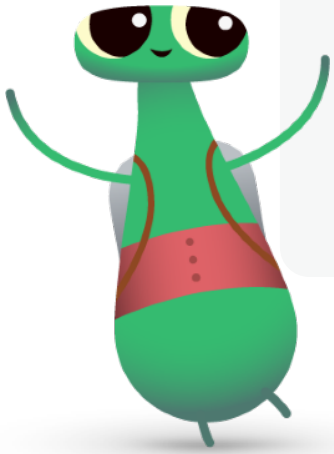
Swift Coding Club



Swift Playgrounds

Ćwiczenia z programowania

Tematy



Witaj w Swift Coding Club!

Prezentowane ćwiczenia z programowania obejmują podstawowe pojęcia związane z pisaniem kodu, które pomogą Ci w projektowaniu aplikacji. Rozwiązując te ćwiczenia, nie tylko zdobędziesz nowe umiejętności w zakresie programowania, ale zaczniesz też rozumieć, jak działają aplikacje. Dzięki temu będziesz tworzyć lepsze aplikacje.

Każdy temat jest poświęcony konkretnemu pojęciu związanemu z programowaniem i obejmuje krótkie ćwiczenie wprowadzające. Po jego wykonaniu możesz wykorzystać poznaną koncepcję do rozwiązania zagadek w aplikacji Swift Playgrounds.

Dla każdego zagadnienia przygotowano też ćwiczenia pt. „Następne kroki”, które pozwalają dowiedzieć się więcej o danym pojęciu. Te dodatkowe ćwiczenia są opcjonalne, możesz zrobić więc oba ćwiczenia lub jedno z nich albo nie robić żadnego. Pamiętaj, że do wykonania niektórych z tych ćwiczeń potrzebne jest dodatkowe urządzenie, takie jak robot lub dron. Jeśli dysponujesz takimi urządzeniami, będą one świetnymi narzędziami do sprawdzenia zdobytej wiedzy w praktyce.

Baw się dobrze!

Myśl jak komputer: polecenia i sekwencje

1

Wprowadzenie (15 minut)

Dobierz się w parę z partnerem. Jedna osoba jest reżyserem, a druga wykonawcą. Reżyser musi wymyślać, co ma robić wykonawca. Zadaniem może być na przykład narysowanie uśmiechniętej twarzy na tablicy lub zrobienie pięciu pajacyków. Reżyser może wydawać tylko pojedyncze polecenia, nie mówiąc przy tym wykonawcy, na czym polega całe zadanie. Czy wykonawca wykonał zadanie zgodnie z zamiarem reżysera?

Reżyser wydawał wykonawcy polecenia w określonej sekwencji, tak samo jak podczas pisania kodu.

Obejrzyj to wideo, by dowiedzieć się więcej o poleceniach, i zobacz wideo o debugowaniu.

Zastanów się jeszcze raz nad instrukcjami reżysera, zwłaszcza wtedy, jeśli wykonawca nie był w stanie wykonać założonego zadania. Czy w instrukcjach pominięto jakiś etap? Czy zmiana kolejności niektórych czynności ułatwiłaby zadanie? Ten proces nazywa się debugowaniem. Programiści często korzystają z debugowania do naprawiania i ulepszania swojego kodu.

Ćwiczenie (30 minut)

W środowisku Learn to Code 1 aplikacji Swift Playgrounds rozwiąż teraz zagadki, które po prawej stronie oznaczono zielonymi symbolami.

Zastanów się: W jaki sposób polecenia zastosowane przez Ciebie w aplikacji przypominają instrukcje, których udzielał reżyser?

Polecenie: konkretna czynność, którą komputer ma wykonać.

Sekwencja: kolejność wydawania poleceń.

Debugowanie: proces wykrywania i naprawiania błędów.

Polecenia

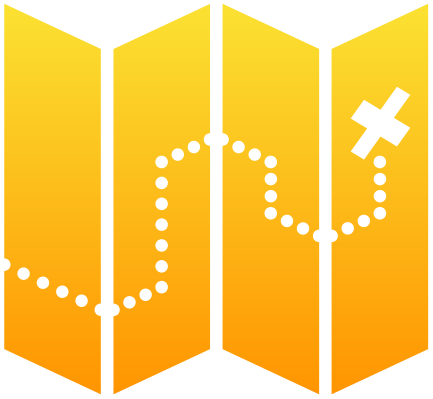
Myśl jak komputer: polecenia i sekwencje
Następne kroki

1

Schowaj i szukaj (1 sesja)

1. Ukryj w sali lub jej pobliżu mały przedmiot.
2. Zatrzymaj się w jednym miejscu i za pomocą aparatu iPada nagraj swoje instrukcje dla osoby, która będzie szukać przedmiotu. Wskazówki powinny opisywać drogę od miejsca, w którym stoisz, do kryjówki.
3. Wymień się nagraniami z innym uczniem. Obejrzyj jego film i spróbuj znaleźć schowany przez tę osobę przedmiot. Udało Ci się?

Zastanów się: W jaki sposób można byłoby ulepszyć instrukcje? Czy instrukcje wymagają debugowania? Jeśli tak, w jaki sposób trzeba to zrobić?



Dash (1 sesja)

1. Jeśli masz do dyspozycji roboty Dash firmy Wonder Workshop, przejdź przez lekcję Dash w aplikacji Swift Playgrounds.

2. Poprowadź Dasha w dniu wyścigu, wykorzystując do tego polecenia.
3. Trasa wyścigu może być trudna. Porównaj swój kod z propozycjami innych uczniów. Jeśli to potrzebne, przejrzyjcie swoje kody i debugujcie je wspólnie.
4. Przekonaj się, jak daleko uda Ci się dotrzeć. Zawsze możesz wrócić do tej lekcji, gdy poznasz więcej pojęć związanych z programowaniem.
5. Następnie możesz stworzyć własną historię z Dashem w roli głównego bohatera.

Zastanów się: Które czujniki Dasha pomogły Ci opowiedzieć swoją historię? Co wnosi do historii wykorzystanie tych czujników?

Myśl sprawnie: funkcje i trochę pętli

2

Wprowadzenie (5 minut)

Przypomnij sobie kroki jakiegoś tańca. Opisz te kroki innym. Najlepiej będzie, jeśli spróbują je też wykonać. Czy łatwo było je opisać?

W programowaniu czasami łatwiej jest utworzyć nowe zachowanie, łącząc istniejące już polecenia. Taki proces nazywa się kompozycją. Żeby utworzyć funkcję, musisz nazwać nowe zachowanie, by można było go użyć ponownie. Kiedy mówisz programowi, by uruchomił jakąś funkcję, „wywołujesz" ją. Ktoś, kto mówi: „wykonaj Macarenę", wywołuje w ten sposób funkcję o nazwie „Macarena".

Obejrzyj to wideo o pętlach i funkcjach.

Ćwiczenie (40 minut)

W środowisku Learn to Code 1 aplikacji Swift Playgrounds rozwiąż teraz zagadki, które po prawej stronie oznaczono zielonymi symbolami.

Zastanów się: Ile ruchów wykonał bohater w wybranej zagadce? Ile poleceń trzeba było w tym celu napisać? Kiedy i po co powinno się tworzyć funkcje i pętle?

Funkcja: zbiór zgrupowanych poleceń, któremu nadano nazwę.

Pętla for: wielokrotnie wykonuje ten sam blok kodu określoną liczbę razy (z ang. dla, na przykład „dla kolejnych wartości zmieniających się od ... do ... zrób ...").

Funkcje

- Introduction ✓
- Composing a New Behavior ✓
- Creating a New Function ✓
- Collect, Toggle, Repeat
- Across the Board
- Nesting Patterns ✓
- Slotted Stairways ✓
- Treasure Hunt

Pętle for

- Introduction ✓
- Using Loops ✓
- Looping All the Sides ✓
- To the Edge and Back
- Loop Jumper
- Branch Out
- Gem Farm
- Four Stash Sweep

Myśl sprawnie: funkcje i trochę pętli
Następne kroki

2

Tworzenie wzorców (1 sesja)

1. Za pomocą aplikacji do rysowania, takiej jak Art Set lub Pages, stwórz jakiś wzór, używając do tego różnych kształtów, obiektów i kolorów. Wzór może mieć dowolną długość.
2. W aplikacji zapisz swój wzór słownie 20 razy, na przykład w ten sposób: „czerwony, żółty, niebieski; czerwony, żółty, niebieski...” itd.
3. Nazwij powtarzającą się część wzoru (na przykład czerwony, żółty, niebieski = Kolory podstawowe), następnie zapisz wzór raz jeszcze, używając do tego tylko nowej nazwy.
4. O ile mniej pracy i kroków trzeba było wykonać, by zapisać ten wzór?
5. Ile razy powtarza się wzór Kolory podstawowe? Opisz teraz swój wzór jednym krokiem. Udało Ci się właśnie napisać pętlę for!

Zastanów się: W jaki sposób to ćwiczenie wiąże się z programowaniem?

Wywiad z robotem (1 sesja)

1. Przejdź do środowiska Answers Starting Point w aplikacji Swift Playgrounds.

2. Na stronie Text (Tekst) dostępne są funkcje „show” (pokaż) i „ask” (zapytaj). Stuknij opcję Run My Code (Uruchom mój kod) i wpisz swoje imię, a następnie stuknij opcję Submit (Wyślij), by zobaczyć, co się stanie. Funkcje mogą zwrócić wynik, który wyświetla się w widoku na żywo.
3. Na stronie Types (Typy) możesz zapoznać się także z innymi funkcjami „show” i „ask”.
4. Dobierz się w parę z innym członkiem klubu i napiszcie serię różnych funkcji „show” i „ask”, które każde z Was ma wykonać.
5. Na podstawie wyników swoich funkcji możesz napisać opowiadanie, artykuł w formie wywiadu lub krótką biografię.

Zastanów się: Co stałoby się, gdyby funkcje „show” i „ask” były zapisane w innej sekwencji? Jak wpłynęłoby to na Twoją historię lub Twój wywiad?

Myśl logicznie: kod warunkowy

3

Wprowadzenie (10 minut)

Rozegraj z resztą grupy kilka rund gry w szpiega. Gracz, który jest szpiegiem, wybiera jakiś obiekt, a następnie opisuje reszcie tylko jedną jego część. Gracze muszą następnie rozejrzeć się wokół i zgadnąć, co zobaczył szpieg. Osoba, która zgadnie, zostaje nowym szpiegiem.

Jakie decyzje trzeba było podjąć, by rozwiązać zagadkę? Jak przebiegał Twój tok rozumowania podczas odgadywania, co widział szpieg? Grając w szpiega, trzeba sprawdzać, czy możliwa odpowiedź spełnia różne warunki.

Obejrzyj to wideo, by dowiedzieć się więcej o kodzie warunkowym.

Ćwiczenie (35 minut)

W środowisku Learn to Code 1 aplikacji Swift Playgrounds rozwiąż teraz zagadki, które po prawej stronie oznaczono zielonymi symbolami.

Zastanów się: Jakie decyzje podejmuje kod, wykonując instrukcję if? W jak sposób są połączone Twoje pętle for i instrukcje if? Dlaczego właśnie tak?

Warunek: sprawdzasz, czy jego wynik jest prawdziwy czy fałszywy.

Instrukcja warunkowa: blok kodu, który działa tylko wtedy, gdy coś jest prawdą.

Wartość logiczna: wartość, która może być tylko prawdziwa albo fałszywa.

Operator logiczny: symbol lub słowo, takie jak „i", „lub" oraz „nie".

Kod warunkowy

- Introduction ✓
- Checking for Switches ✓
- Using else if ✓
- Looping Conditional Code ✓
- Conditional Climb
- Defining Smarter Functions ✓
- Boxed In
- Decision Tree

Oparatory logiczne

- Introduction ✓
- Using the NOT Operator ✓
- Serial of NOT
- Checking This AND That ✓
- Checking This OR That ✓
- Logical Labyrinth

Myśl logicznie: kod warunkowy
Następne kroki

3

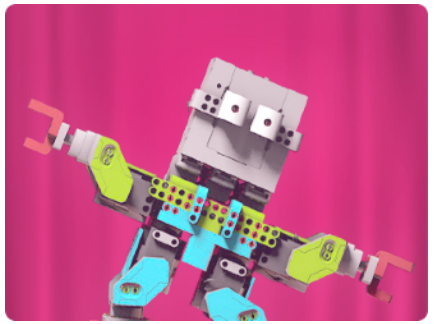
Łowcy rupieci (1 sesja)

1. Każdy uczeń zapisuje na oddzielnych kartkach papierudwa warunki — na przykład „ma kształt prostokąta” lub „zaczyna się na literę C”. Następnie kartki wkłada się do kapelusza.
2. Uczniowie dzielą się na małe grupy i wyjmują po dwa pytania z kapelusza, a następnie wykonują od trzech do pięciu zdjęć znajdujących się w sali przedmiotów, które odpowiadają każdemu z warunków.
3. Za pomocą aplikacji do tworzenia prezentacji, takiej jak na przykład Keynote, należy utworzyć album fotograficzny, w którym każda część będzie poświęcona osobnemu warunkowi. Nie należy jeszcze opisywać warunków.
4. Album trzeba najpierw pokazać innej grupie, by zobaczyć, czy jej członkowie potrafią odgadnąć, jakie warunki ilustrują zdjęcia. Jeśli gracze udzielą poprawnej odpowiedzi, na danej stronie albumu należy umieścić instrukcję warunkową, na przykład „jeśli ma kolor niebieski, zrób temu zdjęcie”.

Zastanów się: Czy w niektórych przypadkach trudno było zgadnąć, czy zdjęcie pasuje do warunku? W jaki sposób poradziłby sobie z takimi przypadkami komputer?

Taniec z robotem MeeBot (1 sesja)

1. Jeśli masz robota MeeBot firmy UBTECH, pobierz w aplikacji Swift Playgrounds lekcję MeeBot Dances.



2. Przejdź lekcję i zobacz, jak za pomocą funkcji, pętli for i kodu warunkowego sprawić, by robot MeeBot zatańczył.
3. Stwórz własną choreografię taneczną. Na ostatniej stronie możesz nawet wybrać własną muzykę.

Zastanów się: W jaki sposób należało wykorzystać kod, by oddawał tempo muzyki?

Myśl w kółko: pętle while

4

Wprowadzenie (5 minut)

W temacie nr 2 poznaliśmy funkcje i pętle for. Jaki taniec udało Ci się stworzyć wspólnie z grupą? W jaki sposób można wykorzystać pętle for do napisania funkcji dla tego tańca?

Wyobraź sobie, że chcesz wykonać ten taniec na szkolnej dyskotece. Skąd masz wiedzieć, że po zakończeniu piosenki trzeba przestać tańczyć?

Służy do tego kod warunkowy — jeśli piosenka gra, tańczysz. Mówiąc dokładniej, nie kod, a pętla while — tak długo jak (ang. while) gra piosenka, tańczysz.

Tym właśnie różni się ta pętla od pętli for, która każe komputerowi wykonać blok kodu określoną liczbę razy — na przykład zatańczyć w kółku 10 razy. Pętla while natomiast każe komputerowi wykonywać blok kodu tak długo, aż jakiś warunek przestanie być spełniony — na przykład tańczyć do momentu zakończenia piosenki.

Obejrzyj to wideo, by dowiedzieć się więcej o pętlach while.

Ćwiczenie (40 minut)

W środowisku Learn to Code 1 aplikacji Swift Playgrounds rozwiąż teraz zagadki, które po prawej stronie oznaczono zielonymi symbolami.

Zastanów się: W jakiej sytuacji zostały przez Ciebie użyte pętle for i while? Dlaczego właśnie wtedy?

Pętla while: pętla, która wykonuje blok kodu tak długo, jak prawdziwy jest dany warunek. Kiedy warunek jest fałszywy, pętla przestaje działać.

Pętle while

Myśl w kółko: pętle while
Następne kroki

4

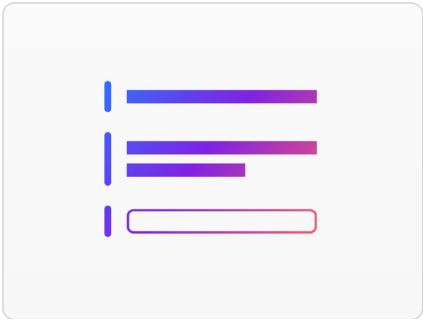
Schowaj i szukaj raz jeszcze (1 sesja)

1. Ukryj w sali lub jej pobliżu mały przedmiot.
2. Zatrzymaj się w jednym miejscu i za pomocą aparatu iPada nagraj swoje instrukcje dla osoby, która będzie szukać przedmiotu. Wskazówki powinny opisywać drogę od miejsca, w którym stoisz, do kryjówki. Gdy to możliwe, posługuj się funkcjami, pętlami for i pętlami while.
3. Wymień się nagraniami z innym uczniem. Obejrzyj jego film i spróbuj znaleźć schowany przez tę osobę przedmiot. Udało Ci się?
4. Przeanalizujcie razem swoje filmy. Wypisz przypadki, w których zostały przez Ciebie użyte pętle for i while.

Zastanów się: Czy zastosowanie pętli sprawiło, że zadanie było teraz łatwiejsze? Czy w niektórych sytuacjach okazało się jednak trudniejsze?

21 pytań (1 sesja)

1. Wróćmy do środowiska Answers Starting Point. Stąd pobierzesz jego odpowiednią wersję.



2. Ten szablon odpowiedzi pozwala na grę w 21 pytań. Zacznij od przeanalizowania kodu. Jak będzie działał?
3. Twórca wybiera obiekt lub przedmiot, który zostanie wprowadzony w kodzie jako odpowiedź. Gracz może zadać twórcy 21 pytań, na które odpowiedź to „tak” lub „nie”.
4. Po każdym pytaniu gracz może wpisać swoją odpowiedź w widoku na żywo i sprawdzić, czy jest poprawna. Należy upewnić się, że twórca przekazuje swoje środowisko z widokiem na żywo na całym ekranie, tak aby gracz nie mógł zobaczyć kodu ani poprawnej odpowiedzi.

Zastanów się: Jakie pojęcia związane z programowaniem i w jaki sposób zastosowano w tym środowisku?

Przemyśl pomysł kilka razy: algorytmy

5

Wprowadzenie (5 minut)

Wspólnie z grupą wskaż codzienne czynności, których zrobienie wymaga wykonania kilku kroków, takie jak umycie zębów lub przygotowanie kanapki. To wszystko to właśnie algorytmy.

Wybierzcie jeden przykład i niech kilku uczniów udzieli instrukcji jak wykonać wybraną czynność. Czy ich instrukcje były takie same? Czym się różniły? Czy ostatecznie wszyscy wykonali tę samą rzecz?

Obejrzyj to wideo, by dowiedzieć się więcej o algorytmach.

Ćwiczenie (40 minut)

W środowisku Learn to Code 1 aplikacji Swift Playgrounds rozwiąż teraz zagadki, które po prawej stronie oznaczono zielonymi symbolami.

Zastanów się: Na ile różnych sposobów można rozwiązać każdą zagadkę? Czyj algorytm był najkrótszy? Czyj algorytm był najciekawszy?

Algorytm: zestaw rozpisanych krok po kroku zasad lub instrukcji.

Pseudokod: nieformalny zapis kodu lub pojęcia, który jest czytelny dla człowieka.

Algorytmy

- Introduction ✓
- The Right-Hand Rule ✓
- Adjusting Your Algorithm ✓
- Conquering a Maze ✓
- Which Way to Turn?
- Roll Right, Roll Left

Przemyśl pomysł kilka razy: algorytmy
Następne kroki

5

Kto jest najwyższy? (1 sesja)

1. Podzielcie się w grupie na kilka mniejszych zespołów. Każdy zespół ma za zadanie przedstawić sposób — algorytm — na wskazanie najwyższej osoby. Nie wystarczy, że wiecie lub po prostu widzicie, że ktoś jest najwyższy!
2. Wykorzystajcie swoją znajomość programowania do wskazania kolejnych kroków w Waszym algorytmie. Możecie wymyślić własny pseudokod do opisania algorytmów za pomocą jak największej liczby terminów związanych z programowaniem.
3. Niech każdy zespół przedstawi swój algorytm. Powinna go wykonać cała grupa.

Zastanów się: Który algorytm wydaje się najskuteczniejszy? Co należy zmienić w algorytmie, by wskazać najniższego ucznia?

Parrot (1 sesja)

1. Jeśli masz do dyspozycji drona firmy Parrot, pobierz w aplikacji Swift Playgrounds lekcję Parrot.

2. Wykorzystaj wszystkie zdobyte do tej pory umiejętności związane z programowaniem, by wznieść drona w powietrze, wylądować, skierować go w różne strony i zmusić go do wykonania różnych figur akrobatycznych.
3. Na koniec utwórz algorytm, za pomocą którego dron przeleci z punktu A do punktu B.

Zastanów się: W jaki sposób prędkość drona wpływa na jego tory lotu?

Myśl jak dziennikarski bot: zmienne

6

Wprowadzenie (5 minut)

Wyobraź sobie, że przeprowadzasz się do nowego domu. Prawdopodobnie spakujesz swoje rzeczy do pudeł i opiszesz ich zawartość jednym lub kilkoma słowami. Pudełko z pucharami możesz na przykład opisać słowem „puchary". Ten przykład pokazuje trzy ważne cechy kontenera.

Kiedy programujemy komputer, zamiast pudełek używamy tak zwanych zmiennych. Zmienne są podobne do pudełek — mają etykietę (nazwę) i zawartość (wartość). Wartość lub zawartość może się zmienić, ale etykieta bądź nazwa są cały czas te same. Wartość lub zawartość zmiennej można określić, wyszukując zmienną o właściwej nazwie lub etykiecie.

Obejrzyj to wideo, aby dowiedzieć się więcej.

Ćwiczenie (40 minut)

W środowisku Learn to Code 2 aplikacji Swift Playgrounds rozwiąż teraz zagadki, które po prawej stronie oznaczono zielonymi symbolami.

Zastanów się: W jaki sposób zastosowanie zmiennych pomogło Ci w pracy z aplikacją?

Zmienna: nazwany kontener, w którym przechowywana jest wartość. Wraz z czasem wartość może się zmieniać.

Zmienna

- Introduction ✓
- Keeping Track ✓
- Bump Up the Value
- Incrementing the Value ✓
- Seeking Seven Gems ✓
- Three Gems, Four Switches
- Checking for Equal Values ✓
- Round Up the Switches
- Collect the Total

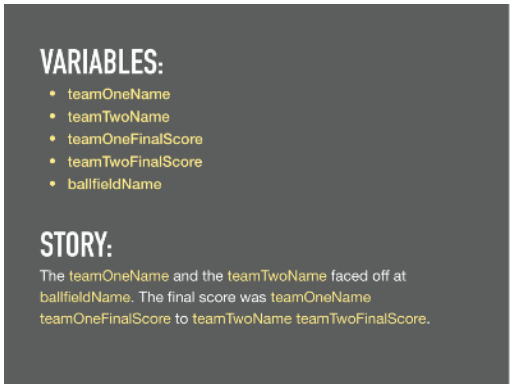
Myśl jak dziennikarski bot: zmienne
Następne kroki

6

Dziennikarski bot (1 sesja)

1. W tym ćwiczeniu opracujesz dziennikarskiego bota, czyli robota, który może automatycznie pisać krótkie artykuły. Zacznij od wybrania jakiejś aktualności lub jakiegoś wydarzenia sportowego i zastanów się, jakie informacje mogą przydać się do napisania tekstu na ten temat.
2. Wymyśl od czterech do sześciu zmiennych, takich jak nazwa drużyny, wynik i data wydarzenia. Wykorzystując zmienne, napisz dwu- lub trzyzdaniowy artykuł, który dziennikarski bot będzie mógł uzupełnić, by opisać podobne wydarzenie.
3. Znajdź parę. Niech jedna osoba z pary poda informację dla każdej zmiennej, którą należy uzupełnić w artykule. Następnie zamieńcie się. Czy udało Wam się otrzymać dwa sensowne artykuły?

Zastanów się: Czy któreś nazwy zmiennych sprawdziły się lepiej niż inne? Dlaczego tak lub dlaczego nie?



Automat do gier Sphero (1 sesja)

1. Jeśli masz do dyspozycji robota Sphero SPRK+, pobierz w aplikacji Swift Playgrounds lekcję Sphero Arcade.

2. Za pomocą funkcji i zmiennych celuj i wykrywaj kolizje w grze Pong, a na koniec zbuduj jej własną wersję.
3. Przed uruchomieniem lub edytowaniem kodu przeanalizuj go na stronie Play Sphero Pong. Do czego służy każde polecenie?
4. Jakie elementy gry udało Ci się zmienić za pomocą zmiennych?

Zastanów się: Jakie inne gry można zmodyfikować tak jak grę Pong?

Myśl jak architekt: typy

7

Wprowadzenie (5 minut)

Ile typów budynków przychodzi Ci do głowy? Wybierz jeden typ. Co go wyróżnia? Innymi słowy, jakie są charakterystyczne cechy tego typu budynku? Co się w nim zazwyczaj robi? Te czynności nazywane są zachowaniami. W szkole na przykład są sale (cechy), a między kolejnymi lekcjami odzywa się dzwonek (zachowanie). Czy inni też zgadzają się co do tych właściwości i zachowań? Dlaczego tak lub dlaczego nie?

Jeśli używamy programu komputerowego do zaprojektowania budynku, musimy być bardzo dokładni. Musimy określić jego typ, wskazując właściwości (które nazywa się *cechami*) i metody (które nazywa się *zachowaniami*).

Obejrzyj to wideo, aby dowiedzieć się więcej.

Ćwiczenie (40 minut)

W środowisku Learn to Code 2 aplikacji Swift Playgrounds rozwiąż teraz zagadki, które po prawej stronie oznaczono zielonymi symbolami.

Zastanów się: Jakie typy można było zauważyć w aplikacji? Co udało Ci się zainicjalizować?

Typ: nazwana grupa właściwości (cech) i metod (zachowań) charakteryzujących wybrany rodzaj danych.

Inicjalizacja: utworzenie nowej instancji danego typu, w tym określenie wartości początkowych dla wszystkich właściwości danego typu.

Typy

- Introduction ✓
- Deactivating a Portal ✓
- Portal On and Off
- Setting the Right Portal ✓
- Corners of the World
- Random Gems Everywhere

Inicjalizacja

- Introduction ✓
- Initializing Your Expert ✓
- Train Your Expert
- Using Instances of Different Types ✓
- It Takes Two

Myśl jak architekt: typy
Następne kroki

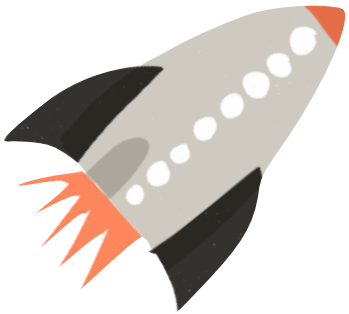
7

Zostań architektem (1 sesja)

1. Wybierz typ budynku, prawdziwy lub wymyślony.
2. Wybierz od pięciu do sześciu zmiennych, które opisują to, jak wygląda ten typ budynku. Te zmienne nie powinny mieć żadnych wartości; zamiast tego powinny składać się z opisów wartości, takich jak „wysokość” lub „liczbaOkien”.
3. Umieść wartości obok zmiennych. Stanowią one opis konkretnej instancji tego typu budynku. Posługując się terminologią języka Swift, można powiedzieć, że inicjalizujesz właśnie instancję swojego typu budynku.
4. Zakończ inicjalizację, korzystając z aplikacji do rysowania, takiej jak Notatki, i naszkicuj w niej swój typ budynku, w którym uwzględnisz wymyślone zmienne i wartości.
5. Znajdź partnera i wymieńcie się listami zmiennych i wartości swoich typów budynków. Niech każdy z Was narysuje budynek wybranego przez drugą osobę typu, a następnie pokażcie sobie swoje rysunki. Czy są do siebie podobne?

Zastanów się: Czy możesz wymyślić sposoby, by sprawić, żeby rysunki byłyby do siebie bardziej podobne?

Typ: hangarRakiety
liczbaSilników = 4
wysokość = 15 metrów
liczbaOkien = 8
kolor = gwiezdna szarość
kształtDrzwi = zaokrąglony prostokąt



Papier, nożyce, kamień (1 sesja)

1. W aplikacji Swift Playgrounds pobierz wyzwanie Rock, Paper, Scissors.

2. Przejdź lekcję. W jaki sposób działa w niej kod? Jakie pojęcia związane z programowaniem rozpoznajesz?
3. Spróbuj teraz stworzyć nową wersję tej gry, w której zastosujesz własne reguły i wprowadzisz nowe działania. By to zrobić, musisz określić typ gry i zainicjalizować go wraz z powiązanymi z nim właściwościami i metodami.

Zastanów się: Jakie właściwości i metody były obecne w oryginalnej grze? Jakie nowe właściwości i metody zostały przez Ciebie dodane w celu spersonalizowania gry?