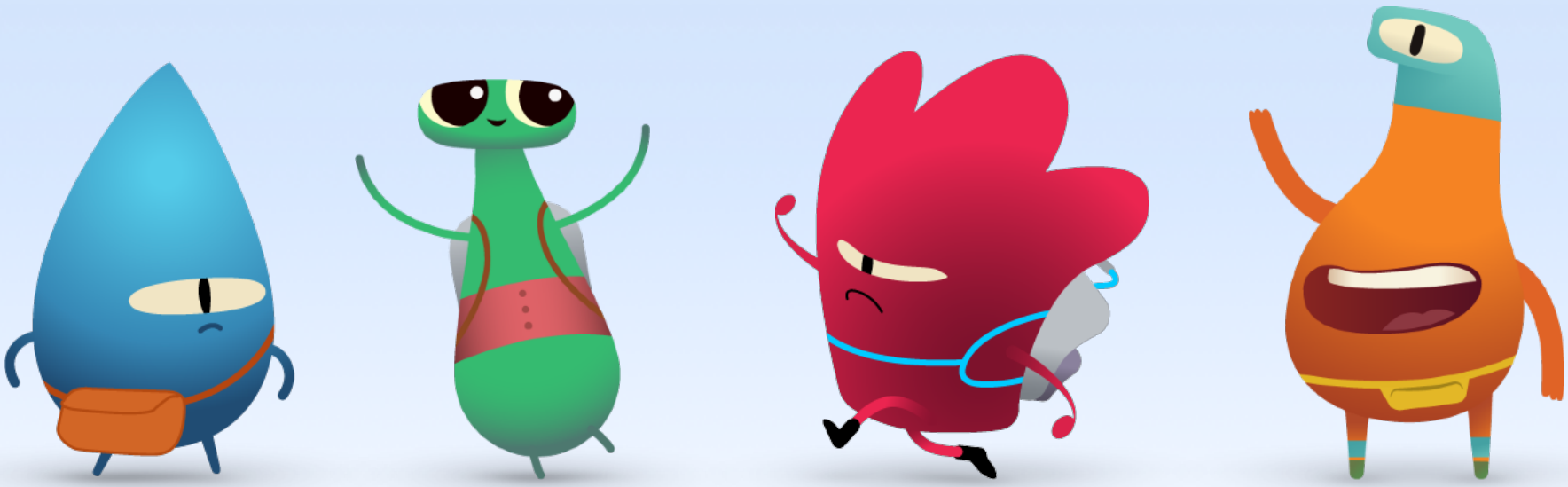
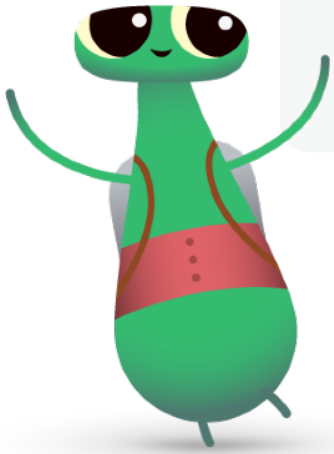
Swift Coding Club



Swift Playgrounds

Attività di programmazione

Argomenti



Ti diamo il benvenuto nello Swift Coding Club!

Queste attività illustrano i concetti di base della programmazione per accompagnare le attività di progettazione di un'app. Con queste attività, non solo imparerai a programmare, ma inizierai anche a capire come funzionano le app in modo da progettarle al meglio.

Ogni argomento ti permetterà di imparare un concetto di programmazione specifico grazie a una breve attività introduttiva, e poi lo metterai in pratica per completare i livelli di Swift Playgrounds.

Sono incluse anche delle attività "Un passo avanti" che ti aiutano ad approfondire il concetto di ciascun argomento. Queste attività aggiuntive sono facoltative e puoi scegliere di farne una, entrambe o nessuna. Ricorda che alcune richiedono un altro dispositivo, come un robot o un drone. Se li hai, usali, perché sono l'ideale per mettere in pratica quello che hai imparato.

Buon divertimento!

Pensare come un computer: comandi e sequenze

1

Introduzione (15 minuti)

Lavora in coppia con un compagno. Uno di voi dà le istruzioni e l'altro le esegue. Chi dà le istruzioni deve pensare a qualcosa da far fare all'altro. Per esempio, disegnare uno smiley sulla lavagna o fare cinque jumping jack. Senza dire qual è il compito complessivo, dà al compagno solo istruzioni passo passo. Quest'ultimo è riuscito a completare l'attività come avrebbe dovuto?

Le istruzioni sono state date all'interno di una sequenza, che è esattamente quello che devi fare quando scrivi codice.

Guarda questo video per saperne di più sui comandi e quest'altro per scoprire come fare il debug.

Ora pensa di nuovo alle istruzioni, soprattutto se il compagno che le riceveva non è stato in grado di completare l'attività come previsto. Mancava qualche passaggio nelle istruzioni? Oppure, se avessi invertito l'ordine di alcuni passaggi, le istruzioni sarebbero state più chiare? Questo processo è chiamato debug. I programmatori spesso eseguono il debug per correggere e migliorare il codice.

Esercitazione (30 minuti)

Ora usa "Impara a programmare 1" in Swift Playgrounds per completare i livelli con i segni di spunta nell'elenco a destra.

Prova a pensarci: qual è la differenza tra i comandi che hai scritto nell'app e le istruzioni che sono state date nel gioco di prima?

Comando: un'azione specifica da far eseguire al computer.

Sequenza: l'ordine in cui vengono forniti i comandi.

Debug: il processo di identificazione e correzione dell'errore.

Comandi

- Introduzione ✓
- Creazione dei comandi ✓
- Aggiunta di un nuovo comando ✓
- Azionare un interruttore ✓
- Esercitazione con i portali ✓
- Individuazione e correzione di bug ✓
- Caccia ai bug ✓
- La strada più breve ✓

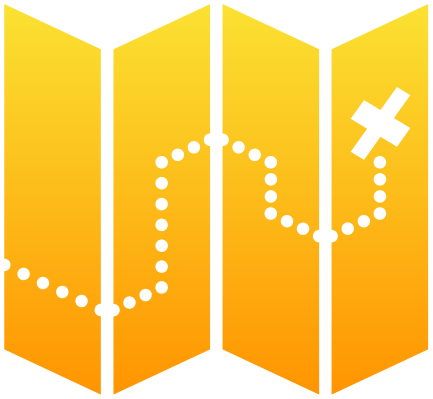
Pensare come un computer: comandi e sequenze
Un passo avanti

1

Nascondino (1 sessione)

1. Nascondi un piccolo oggetto dentro o vicino alla stanza.
2. Posizionati in un punto e usa la fotocamera dell'iPad per registrarti mentre dai indicazioni a qualcun altro affinché trovi l'oggetto. Le indicazioni dovranno iniziare dal punto in cui ti trovi.
3. Ora scambia il tuo video con quello di un altro studente. Guarda il video per trovare l'oggetto nascosto. Lo hai trovato?

Prova a pensarci: come potrebbero essere migliorate le istruzioni? Hanno bisogno di un debug? Se sì, come?



Dash (1 sessione)

1. Se hai a disposizione dei robot Dash di Wonder Workshop, apri la lezione Dash in Swift Playgrounds.

2. Usa i comandi per guidare Dash in "Race Day".
3. Il percorso fino a "Get to the Race" è un po' complicato. Confronta il tuo codice con quello di altri studenti; se necessario, controllatevi il codice a vicenda e fate il debug insieme.
4. Vedi fino a dove riesci ad arrivare. Puoi sempre tornare a questa lezione quando avrai imparato altri concetti di programmazione.
5. Quando te la senti, puoi creare una tua storia con Dash come protagonista.

Prova a pensarci: quali sensori di Dash hai usato per aiutarti a raccontare la storia? A cosa sono serviti i sensori nel racconto?

Pensare in modo efficiente: funzioni e introduzione ai cicli

2

Introduzione (5 minuti)

Pensa ad alcune mosse di ballo. Descrivi le mosse ai compagni o, ancora meglio, chiedi loro di eseguirle. È stato facile descriverle?

Talvolta nella programmazione è più facile combinare dei comandi esistenti per creare un nuovo comportamento. Questo processo è chiamato composizione. Quando assegni un nome al nuovo comportamento per poterlo riutilizzare, crei una funzione. Quando dici a un programma di eseguire una funzione, la stai "chiamando". Quindi se qualcuno dice "Balla la Macarena", sta chiamando la funzione "Macarena".

Dai un'occhiata a questo video sulle funzioni e i cicli.

Esercitazione (40 minuti)

Ora usa "Impara a programmare 1" in Swift Playgrounds per completare i livelli con i segni di spunta nell'elenco a destra.

Prova a pensarci: in un dato livello, quante mosse ha fatto il personaggio? E quanti comandi hai scritto? Quando e perché dovresti creare delle funzioni e dei cicli?

Funzione: un insieme di comandi a cui è stato assegnato un nome.

Ciclo for: esegue ripetutamente un blocco di codice per un determinato numero di volte.

Funzioni

- Introduzione ✓
- Composizione di una nuova azione ✓
- Creazione di una nuova funzione ✓
- Raccogli, aziona, ripeti
- A tutto campo
- Schema di annidamento ✓
- Scale separate ✓
- Caccia al tesoro

Cicli for

- Introduzione ✓
- Uso dei cicli ✓
- Un ciclo per tutti i lati ✓
- Verso i bordi e di nuovo al centro
- Cicli e portali
- Amplia i tuoi orizzonti
- Fabbrica di gemme
- Pulizia in quattro movimenti

Pensare in modo efficiente: funzioni e introduzione ai cicli

Un passo avanti

2

Creare uno schema (1 sessione)

1. Crea uno schema usando un'app di disegno come Art Set o Pages e incorporando forme, oggetti e colori diversi. Lo schema può essere lungo quanto vuoi.
2. Nell'app, scrivi 20 volte lo schema a parole. Per esempio, "rosso, giallo, blu; rosso, giallo, blu..." e così via.
3. Dai un nome alla parte dello schema che si ripete (per esempio, rosso, giallo e blu = Colori primari) e poi scrivi di nuovo lo schema a parole usando solo il nuovo nome.
4. Quanti passaggi in meno hai usato per scrivere lo schema?
5. Quante volte si ripete Colori primari? Ora descrivi lo schema in un solo passaggio. Hai scritto un ciclo for!

Prova a pensarci: qual è il collegamento tra questa attività e la programmazione?

Intervista con il robot (1 sessione)

1. Apri il modello Risposte in Swift Playgrounds.

2. Sulla pagina Testo, "show" e "ask" sono delle funzioni. Tocca "Esegui codice" e inserisci il tuo nome, quindi tocca Invia e vedi cosa succede. Le funzioni possono avere un risultato, che è quello che vedi nella vista attiva.
3. Nella pagina Tipi, esercitati con altre funzioni "show" e "ask".
4. In coppia con un compagno, scrivete una serie di funzioni "show" e "ask" diverse da far completare all'altro.
5. Usa i risultati delle tue funzioni per scrivere una storia, un'intervista o una breve biografia di fantasia.

Prova a pensarci: cosa succederebbe se scrivessi le funzioni "show" e "ask" in una sequenza diversa? Che effetto avrebbe questo sulla tua storia o intervista?

Pensare in modo logico: codice condizionale

3

Introduzione (10 minuti)

In gruppo, fate un paio di round del gioco “Indovina cosa vedo”. Uno studente sceglie un oggetto e ne descrive agli altri giocatori solo una parte. I giocatori devono guardarsi intorno e indovinare l’oggetto. Lo studente che indovina sceglierà l’oggetto nel round successivo.

Quali decisioni avete dovuto prendere? Qual è stato il vostro processo mentale mentre cercavate di indovinare l’oggetto? Per giocare a “Indovina cosa vedo”, hai dovuto pensare in termini di condizioni.

Guarda questo video per saperne di più sul codice condizionale.

Esercitazione (35 minuti)

Ora usa “Impara a programmare 1” in Swift Playgrounds per completare i livelli con i segni di spunta nell’elenco a destra.

Prova a pensarci: quali tipi di decisioni ha preso il codice usando l’istruzione if? Come hai combinato i cicli for e le istruzioni if? Perché?

Condizione: qualcosa che verifichi e che dà come risultato vero o falso.

Codice condizionale: un blocco di codice che verrà eseguito solo se qualcosa è vero.

Valore booleano: un valore che può essere solo vero o falso.

Operatore logico: un simbolo o parole come “e”, “o” e “non”.

Codice condizionale

- Introduzione ✓
- Controllo degli interruttori ✓
- Uso di else if ✓
- Un ciclo per il codice condizionale ✓
- Scalata condizionale
- Funzioni più intelligenti ✓
- In scatola
- Albero delle decisioni

Operatori logici

- Introduzione ✓
- Uso dell’operatore NOT ✓
- Spirale di NOT
- Uso dell’operatore AND ✓
- Uso dell’operatore OR ✓
- Labirinto logico

Pensare in modo logico: codice condizionale
Un passo avanti

3

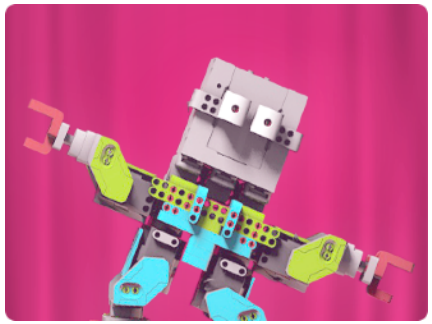
Caccia al tesoro (1 sessione)

1. Ogni studente scrive due condizioni su foglietti distinti, per esempio “È un rettangolo” o “Inizia con la lettera C”, poi mette i foglietti in un cappello.
2. Divisi in piccoli gruppi, estraete due condizioni dal cappello e fate da tre a cinque foto nell’aula che corrispondano a ciascuna condizione.
3. Realizzate un album di foto usando un’app per presentazioni come Keynote, creando una sezione per ciascuna condizione. Non date ancora un nome alle condizioni.
4. Presentate il vostro album fotografico a un altro gruppo per vedere se riesce a indovinare le condizioni. Se indovina, aggiungete l’istruzione condizionale alla pagina dell’album, per esempio “Se è blu, fai una foto”.

Prova a pensarci: in alcuni casi è stato difficile giudicare l’effettiva corrispondenza di una foto? Come gestirebbe questi casi un computer?

Il ballo di MeeBot (1 sessione)

1. Se hai un robot MeeBot di UBTECH, scarica la lezione “MeeBot Dances” in Swift Playgrounds.



2. Esplora la lezione e vedi come puoi usare le funzioni, i cicli for e il codice condizionale per far ballare MeeBot.
3. Crea la tua routine di ballo. Sull’ultima pagina puoi anche scegliere la musica che preferisci.

Prova a pensarci: come hai usato il codice per rispecchiare il ritmo della musica?

Pensare ripetutamente: cicli while

4

Introduzione (5 minuti)

Nell'argomento 2, hai visto le funzioni e i cicli for. Qual era il ballo che avete fatto insieme? Come usereste i cicli for per scrivere la funzione per il ballo?

Ora immagina di volerlo ballare alla festa della scuola. Come sapresti quando smettere di ballare alla fine della canzone?

Useresti un codice condizionale: "Se sento la canzone, ballo". O in modo più chiaro, con un ciclo while: "Finché sento la canzone, ballo".

Al contrario di un ciclo for, che dice a un computer di eseguire un blocco di codice per un determinato numero di volte (per esempio, ballare in cerchio per 10 volte), un ciclo while dice al computer di eseguire un blocco di codice finché non succede qualcosa. Quindi, ballo in cerchio finché la canzone non si interrompe.

Guarda questo video per saperne di più sui cicli while.

Esercitazione (40 minuti)

Ora usa "Impara a programmare 1" in Swift Playgrounds per completare i livelli con i segni di spunta nell'elenco a destra.

Prova a pensarci: quando hai usato i cicli for e i cicli while nell'app? Come lo hai deciso?

Ciclo while: un ciclo che esegue un blocco di codice fintanto che una data condizione è vera. Quando la condizione diventa falsa, il ciclo si interrompe.

Cicli while

- Introduzione ✓
- Il tuo primo ciclo while ✓
- Cicli while più intelligenti ✓
- Scelta dello strumento corretto ✓
- Quattro per quattro
- Girotondo
- Terra dell'abbondanza
- Annidamento dei cicli ✓
- Rettangoli casuali
- Hai sempre ragione

Pensare ripetutamente: cicli while

Un passo avanti

4

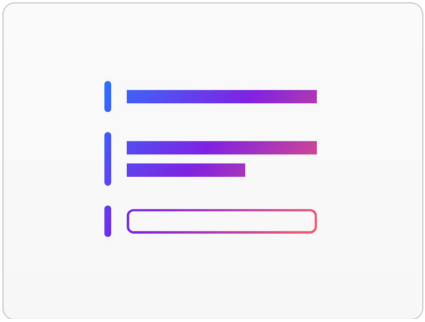
Nascondino (1 sessione)

1. Nascondi un piccolo oggetto dentro o vicino alla stanza.
2. Posizionati in un punto e usa la fotocamera dell'iPad per registrarti mentre dai indicazioni a qualcun altro affinché trovi l'oggetto. Le indicazioni dovranno iniziare dal punto in cui ti trovi. Usa funzioni, cicli for e cicli while quando puoi.
3. Ora scambia il tuo video con quello di un altro studente. Guarda il video per trovare l'oggetto nascosto. Lo hai trovato?
4. Guardate entrambi i video insieme. Scrivete i punti in cui avete usato cicli for e cicli while nelle indicazioni.

Prova a pensarci: questa volta, l'uso dei cicli ha reso le cose più facili? Ci sono stati dei casi in cui le ha rese più difficili?

21 domande (1 sessione)

1. Diamo di nuovo un'occhiata al modello Risposte. Scarica questa versione.



2. Il modello Risposte è impostato in modo da eseguire 21 domande. Dai prima un'occhiata al codice. Cosa farà?
3. Il creatore sceglie un oggetto o qualcosa da inserire come risposta nel codice. Il giocatore può porre al creatore 21 domande con risposta sì o no.
4. Dopo ciascuna domanda, il giocatore può inserire la propria soluzione nella vista attiva per verificarla. Assicurati che il creatore consegni il playground con la vista attiva aperta a schermo intero, in modo che il giocatore non possa vedere il codice e la risposta corretta.

Prova a pensarci: quali concetti di programmazione utilizzava questo playground e in che modo vengono usati?

Pensare la stessa idea: algoritmi

5

Introduzione (5 minuti)

Lavorando in gruppo, nominate alcune cose che fate spesso che richiedono più passaggi per essere completate, per esempio lavarsi i denti o farsi un panino. Questi sono degli algoritmi.

Scegli un esempio e chiedi a vari studenti di dare indicazioni su come farlo. Le indicazioni sono state le stesse? Quali sono state le differenze? Tutte le indicazioni hanno portato allo stesso risultato?

Guarda questo video per saperne di più sugli algoritmi.

Esercitazione (40 minuti)

Ora usa "Impara a programmare 1" in Swift Playgrounds per completare i livelli con i segni di spunta nell'elenco a destra.

Prova a pensarci: secondo te, quanti modi ci sono per completare ogni livello? Chi aveva l'algoritmo più breve? Chi aveva quello più interessante?

Algoritmo: una serie di regole o istruzioni passo passo.

Pseudocodice: una descrizione informale di una porzione di codice o di un concetto pensata per essere letta dall'uomo.

Algoritmi

- Introduzione ✓
- La regola della mano destra ✓
- Modifica dell'algoritmo ✓
- Conquista di un labirinto ✓
- Da che parte girare?
- Gira a destra, gira a sinistra

Pensare la stessa idea: algoritmi
Un passo avanti

5

Chi è più alto? (1 sessione)

1. Dividi la classe in piccoli gruppi. Ogni gruppo penserà a un modo, o "algoritmo", per permettere a qualcuno di determinare qual è lo studente più alto. Non importa se lo sapete già o se vi basta guardare per scoprirlo!
2. Usa le tue conoscenze di programmazione per trovare i passaggi dell'algoritmo. Puoi inventare uno pseudocodice per scrivere gli algoritmi usando il maggior numero possibile di termini di programmazione.
3. Chiedi a ciascun gruppo di presentare il proprio algoritmo. Dovrebbero farlo tutti insieme.

Prova a pensarci: quale algoritmo sembra più efficiente? Se volessi trovare lo studente più basso, che cosa modificheresti nel tuo algoritmo?

Parrot (1 sessione)

1. Se hai a disposizione un drone Parrot, scarica la lezione Parrot in Swift Playgrounds.

2. Usa tutte le abilità di programmazione che hai imparato finora per farlo decollare, atterrare e volteggiare, e per fargli cambiare direzione.
3. Infine, crea un algoritmo per il percorso di volo che dovrà fare il drone per arrivare dal punto A al punto B.

Prova a pensarci: in che modo la velocità del drone ha influenzato i percorsi di volo?

Pensare come un NewsBot: variabili

6

Introduzione (5 minuti)

Immagina di doverti trasferire in una nuova casa. Probabilmente metteresti tutto quello che hai in degli scatoloni, etichettandoli con una o due parole che ne descrivano il contenuto. Per esempio, se uno scatolone contiene i tuoi trofei, potresti etichettarlo con la parola "Trofei". Questo esempio suggerisce tre importanti funzioni di un contenitore.

Quando programmiamo un computer, anziché usare gli scatoloni, usiamo le variabili. Le variabili sono simili agli scatoloni: hanno sia un'etichetta (un nome) sia un contenuto (un valore). Il valore o i contenuti possono cambiare, ma l'etichetta o il nome no. Possiamo trovare il valore o il contenuto di una variabile individuando la variabile con il nome o l'etichetta corretti.

Guarda questo video per saperne di più.

Esercitazione (40 minuti)

Ora usa "Impara a programmare 2" in Swift Playgrounds per completare i livelli con i segni di spunta nell'elenco a destra.

Prova a pensarci: in che modo le variabili ti hanno aiutato nell'app?

Variabile: un contenitore denominato che archivia un valore. Il valore può cambiare nel tempo.

Variabile

- Introduzione ✓
- Tenere traccia ✓
- Su col valore
- Incrementare il valore ✓
- Cercare sette gemme ✓
- Tre gemme, quattro interruttori
- Controllare i valori uguali ✓
- Pareggiare gli interruttori
- Raccogliere il totale

Pensare come un NewsBot: variabili

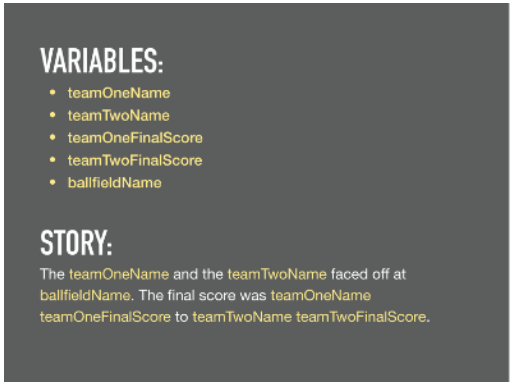
Un passo avanti

6

NewsBot (1 sessione)

1. Per questa attività, creerai NewsBot, un robot in grado di scrivere automaticamente un breve articolo. Per prima cosa, pensa a una notizia o a un evento sportivo, e a quali informazioni potrebbero servirti per scriverci un articolo.
2. Pensa a minimo quattro e massimo sei variabili, come nome della squadra, punteggio e data dell'evento. Scrivi un testo di due o tre frasi usando le variabili, che poi NewsBot potrà inserire per altri eventi simili.
3. Lavorate in coppia. Uno di voi fornisce le informazioni per ciascuna variabile da inserire nella storia. Poi fate cambio. I due articoli completati sono comprensibili?

Prova a pensarci: ci sono nomi di variabili che hanno funzionato meglio di altri? Perché o perché no?



Sphero Arcade (1 sessione)

1. Se hai a disposizione un robot Sphero SPRK+, scarica la lezione Sphero Arcade in Swift Playgrounds.

2. Usa funzioni e variabili per prendere la mira, rilevare collisioni e infine costruire la tua versione di Pong.
3. Sulla pagina Play Sphero Pong, analizza il codice prima di eseguirlo o modificarlo. Cosa fa ciascun comando?
4. Quali elementi del gioco hai cambiato usando le variabili?

Prova a pensarci: quali altri giochi potresti personalizzare come hai fatto con Pong?

Pensare come un architetto: tipi

7

Introduzione (5 minuti)

Quanti tipi di edifici ti vengono in mente? Scegline uno. Che cosa lo rende unico? In altre parole, quali sono alcune caratteristiche specifiche di quel particolare tipo di edificio? In genere, cosa succede al suo interno? Questi sono quelli che chiamiamo comportamenti. Per esempio, una scuola ha delle aule (caratteristiche) e la campanella suona tra una lezione e l'altra (comportamento). Siete tutti d'accordo sulle proprietà e i comportamenti? Perché o perché no?

Se usiamo un programma per aiutarci a costruire l'edificio, dobbiamo essere molto specifici. Dobbiamo definire il suo tipo fornendo le proprietà (ciò che abbiamo chiamato *caratteristiche*) e i metodi (ciò che abbiamo chiamato *comportamenti*).

Guarda questo video per saperne di più.

Esercitazione (40 minuti)

Ora usa "Impara a programmare 2" in Swift Playgrounds per completare i livelli con i segni di spunta nell'elenco a destra.

Prova a pensarci: quali erano i tipi nell'app? Che cosa hai inizializzato?

Tipo: un gruppo denominato di proprietà (caratteristiche) e metodi (comportamenti) di una classe di dati.

Inizializzazione: l'atto di creare una nuova istanza di un tipo, che include la definizione dei valori iniziali per le proprietà del tipo.

Tipi

- Introduzione ✓
- Disattivare un portale ✓
- Attivare e disattivare i portali
- Impostare il portale corretto ✓
- Angoli del mondo
- Gemme casuali ovunque

Inizializzazione

- Introduzione ✓
- Inizializzare l'esperto ✓
- Addestrare l'esperto
- Usare istanze di tipi diversi ✓
- In due è meglio

Pensare come un architetto: tipi
Un passo avanti

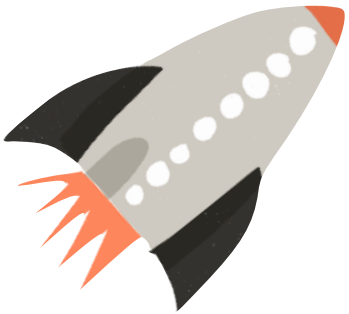
7

Essere un architetto (1 sessione)

1. Scegli un tipo di edificio, reale o immaginario.
2. Pensa a cinque o sei variabili per descrivere il tipo di edificio. Queste variabili non devono contenere alcun valore, ma devono essere descrittive: per esempio “altezza” o “numeroDiFinestre”.
3. Poi aggiungi dei valori accanto alle variabili. Questo passaggio descrive un’istanza specifica del tipo di edificio. Nella terminologia Swift, stai inizializzando un’istanza del tipo di edificio.
4. Completa l’inizializzazione usando un’app di disegno come Note per disegnare il tipo di edificio, in modo che rifletta le variabili e i valori.
5. Trova un compagno e scambiatevi l’elenco con le variabili e i valori dei tipi di edifici. Disegnate ognuno il tipo di edificio dell’altro, e poi guardate i disegni. Quanto sono simili?

Prova a pensarci: ti vengono in mente alcuni modi per rendere i disegni più simili?

Tipo: casaRazzo
numeroDiMotori = 4
altezza = 15 metri
numeroDiFinestre = 8
colore = Grigio siderale
formaPorta = Rettangolo arrotondato



Sasso, carta, forbici (1 sessione)

1. Scarica la sfida “Sasso, carta, forbici” in Swift Playgrounds.

2. Esplora la lezione. Come funziona il codice? Quanti concetti di programmazione riesci a riconoscere?
3. Ora prova a creare una versione del gioco in cui usi le tue regole e aggiungi nuove azioni. Per farlo, dovrai definire il tipo del gioco e inizializzarlo con le proprietà e i metodi associati.

Prova a pensarci: quali erano le proprietà e i metodi del gioco originale? Quali sono le nuove proprietà e i metodi che hai aggiunto per personalizzare il codice?