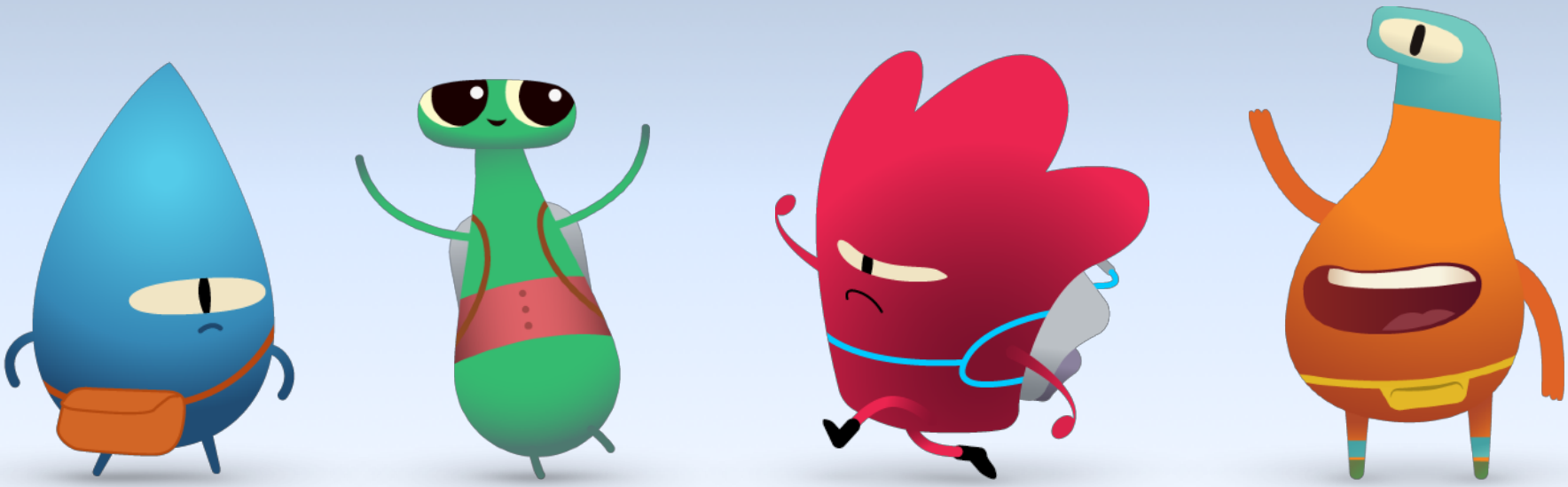
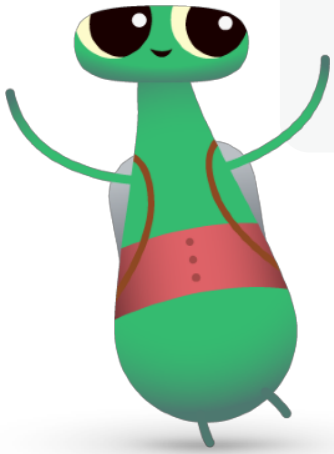
Swift Coding Club



Swift Playgrounds

Kodningsaktiviteter

Avsnitt



Välkommen till Swift-kodklubben!

De här kodningsaktiviteterna täcker in grundläggande kodningskoncept som används inom appdesign. Aktiviteterna hjälper dig att bli bättre på att koda och att förstå hur appar är uppbyggda. Det kommer du att ha nytta av när du ska utforma egna appar.

Varje avsnitt handlar om ett visst kodningskoncept och innehåller en kort introduktionsaktivitet. Sedan får du använda det kodningskonceptet för att lösa uppgifter i Swift Playgrounds.

Det finns också Gå ett steg längre-aktiviteter under varje ämne, som hjälper dig att fördjupa dina kunskaper. De här extraaktiviteterna är frivilliga och du kan välja att inte göra dem alls eller bara göra vissa av dem. En del av de frivilliga aktiviteterna kräver också annan utrustning, som en robot eller en drönare. Om du har tillgång till sådan utrustning är det ett bra sätt att tillämpa det du har lärt dig.

Ha det så kul!

Tänk som en dator: Kommandon och sekvenser

1

Introduktion (15 minuter)

Samarbeta i par. En är arbetsgivare och en är arbetstagare. Arbetsgivaren får hitta på en uppgift som arbetstagaren ska utföra. Det kan till exempel vara att rita en glad gubbe på tavlan eller göra fem stjärnhopp. Arbetsgivaren ska sedan få arbetstagaren att utföra uppgiften genom stegvisa instruktioner, utan att tala om vad uppgiften är. Gjorde arbetstagaren det som arbetsgivaren hade tänkt?

Arbetsgivaren gav kommandon till arbetstagaren i en viss sekvens. Det är precis det du också ska göra när du skriver kod.

Titta på den här videon om kommandon och den här om felsökning.

Tänk nu igenom instruktionerna igen, speciellt om arbetstagaren inte utförde uppgiften som det var tänkt. Var det något steg som saknades? Om du bytte ordning på stegen, skulle det bli tydligare? Den här processen kallas felsökning. Programmerare felsöker ofta sin kod för att rätta till fel och göra den bättre.

Övning (30 minuter)

Öppna nu Swift Playgrounds Lär dig programmera 1 och lös uppgifterna som har markerats med en grön bock i listan till höger.

Att fundera på: Vad finns det för likheter och skillnader mellan kommandona du använde i appen och instruktionerna från arbetsgivaren?

Kommando: En specifik handling som datorn ska utföra.

Sekvens: Ordningen som kommandon ges i.

Felsökning: Processen där man identifierar och åtgärdar buggen.

Kommandon

- Introduktion ✓
- Utfärda kommandon ✓
- Lägga till ett nytt kommando ✓
- Växla reglage ✓
- Portalövning ✓
- Hitta och fixa buggar ✓
- Felsökningsövning ✓
- Den kortaste vägen ✓

Tänk som en dator: Kommandon och sekvenser

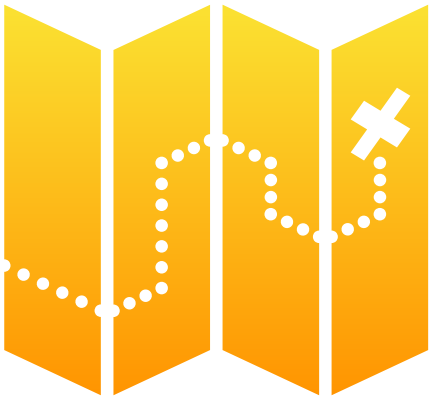
Gå ett steg längre

1

Hitta föremålet (1 klubbträff)

1. Göm ett litet föremål i eller i närheten av rummet.
2. Stå på en plats och använd iPad-kameran för att spela in dig själv medan du ger anvisningar så att någon annan kan hitta föremålet. Anvisningarna ska starta från den plats där du står.
3. Byt nu video med en annan deltagare. Titta på den andra deltagarens video och försök hitta det gömda föremålet. Hittade du det?

Att fundera på: Hur skulle instruktionerna kunna förbättras? Finns det några buggar i instruktionerna som behöver fixas? Vilka då, i så fall?



Dash (1 klubbträff)

1. Om ni har tillgång till Dash-robotar från Wonder Workshop hämtar du Dash-lektionen i Swift Playgrounds.

2. Guida Dash genom en dag på racerbanan med hjälp av kommandon.
3. Banan i Get to the Race kan vara ganska knepig. Jämför din kod med de andra deltagarnas. Om det behövs kan ni granska varandras kod och felsöka den tillsammans.
4. Se hur långt du kommer. Du kan alltid gå tillbaka till den här lektionen senare, när du har lärt dig fler kodningskoncept.
5. När du är klar kan du skapa en egen berättelse med Dash i huvudrollen.

Att fundera på: Vilka sensorer på Dash använde du för att berätta din historia? Hur bidrog sensorerna till berättelsen?

Tänk effektivt: Funktioner och lite loopar

2

Introduktion (5 minuter)

Tänk på några dansrörelser. Beskriv dem för varandra eller, ännu bättre, få någon annan att utföra dem. Hur enkelt var det att beskriva?

Inom programmering är det ibland enklare att kombinera befintliga kommandon för att skapa ett nytt beteende. Den här processen heter sammanställning. När du ger det nya beteendet ett namn så att du kan använda det igen i framtiden har du skapat en funktion. När du säger till ett program att köra en funktion ”anropar” du funktionen. Så om någon säger ”Dansa Macarena” anropar personen funktionen ”Macarena”.

Titta på den här videon om funktioner och loopar.

Övning (40 minuter)

Öppna nu Swift Playgrounds Lär dig programmera 1 och lös uppgifterna som har markerats med en grön bock i listan till höger.

Att fundera på: Hur många rörelser gjorde karaktären i ett visst pussel? Och hur många kommandon skrev du? När och varför är det bra att skapa funktioner och loopar?

Funktion: En samling kommandon som har grupperats ihop och fått ett namn.

For-loop: Kör ett block med kod om och om igen ett givet antal gånger.

Funktioner

- Introduktion ✓
- Sammanställa ett nytt beteende ✓
- Skapa en ny funktion ✓
- Hämta, växla, upprepa
- Över brädet
- Nästlingsmönster ✓
- Täta trappor ✓
- Skattjakt

For-loopar

- Introduktion ✓
- Använda loopar ✓
- Loopa alla sidorna ✓
- Till kanten och tillbaka
- Loophoppare
- Utspritt
- Stenfarm
- Fyra samlingar

Tänk effektivt: Funktioner och lite loopar

Gå ett steg längre

2

Skapa mönster (1 klubbträff)

1. Skapa ett mönster i en ritapp som Art Set eller Pages. Använd olika former, föremål och färger. Mönstret kan vara hur långt du vill.
2. Skriv mönstret 20 gånger med ord i appen, till exempel ”rött, gult, blått, rött, gult, blått ...” och så vidare.
3. Ge ett namn till den del av mönstret som upprepas (till exempel rött, gult, blått = Primärfärger). Skriv sedan mönstret igen med ord och använd bara det nya namnet.
4. Hur mycket mindre arbete eller hur många färre steg krävdes för att skriva mönstret?
5. Hur många gånger upprepas Primärfärger? Beskriv nu mönstret i ett enda steg. Du har skrivit en for-loop!

Att fundera på: Vad har den här aktiviteten för likheter med kodning?

Robotintervju (1 klubbträff)

1. Hämta startpunkten Svar i Swift Playgrounds.

2. På sidan Text är ”show” och ”ask” funktioner. Tryck på Kör min kod och fyll i ditt namn. Tryck sedan på Skicka och se vad som händer. Funktioner kan ha ett resultat, vilket är vad du ser i livevyn.
3. På sidan Typer kan du utforska olika show- och ask-funktioner.
4. Jobba i par. Skriv en serie olika show- och ask-funktioner som din kompis får fylla i.
5. Använd resultaten från funktionerna för att skriva en berättelse, en intervju eller en kort biografi.

Att fundera på: Vad skulle hända om du skrev show- och ask-funktionerna i en annan ordning? Hur skulle det påverka berättelsen eller intervjun?

Tänk logiskt: Villkorlig kod

3

Introduktion (10 minuter)

Lek Ett skepp kommer lastat några gånger med hela gruppen. I leken väljer en person ett föremål och beskriver bara en del av det för deltagarna. Deltagarna får sedan titta sig omkring och gissa vad personen såg. Den som gissar rätt får sedan välja ett föremål.

Vilka beslut behövde du ta? Hur tänkte du när du försökte lista ut vad personen tänkte på? När du lekte Ett skepp kommer lastat tänkte du i villkor.

I den här videon får du lära dig mer om villkor.

Övning (35 minuter)

Öppna nu Swift Playgrounds Lär dig programmera 1 och lös uppgifterna som har markerats med en grön bock i listan till höger.

Att fundera på: Vilka slags beslut fattade din kod med hjälp av if-satsen? Hur kombinerade ni for-loopar och if-uttryck? Varför?

Villkor: Något du testar som får resultatet ”sant” eller ”falskt”.

Villkorlig kod: Ett block med kod som endast körs om något är sant.

Boolesk: Ett värde som bara kan vara antingen sant eller falskt.

Logisk operator: En symbol eller ett ord, som ”and”, ”or” och ”not”, som kopplar ihop värden.

Villkorlig kod

- Introduktion ✓
- Leta efter reglage ✓
- Använda else if ✓
- Villkorlig loopande kod ✓
- Villkorlig trappgång
- Definiera smartare funktioner ✓
- Instängd
- Beslutsträd

Logiska operatorer

- Introduktion ✓
- Använda NOT-operatorn ✓
- Seriell NOT
- Kontrollera med AND ✓
- Kontrollera med OR ✓
- Logisk labyrint

Tänk logiskt: Villkorlig kod
Gå ett steg längre

3

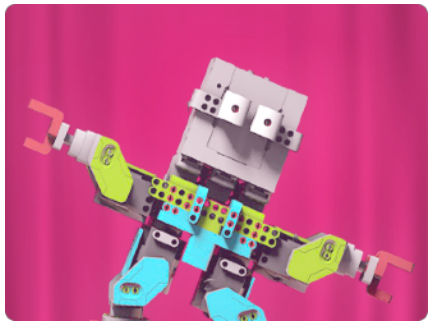
Skattjakt (1 klubbträff)

1. Alla deltagare får skriva två villkor på separata lappar, till exempel ”Är en rektangel” eller ”Börjar med bokstaven K”. Lägg sedan lapparna i en hatt.
2. Bilda mindre grupper. Ta två lappar ur hatten och fotografera sedan tre till fem saker i rummet som uppfyller vardera villkor.
3. Skapa ett fotoalbum i en presentationsapp som Keynote, med ett avsnitt för varje villkor. Skriv inte ut villkoren ännu.
4. Visa fotoalbumet för en annan grupp och se om de kan gissa villkoren. Om de gissar rätt skriver ni in villkorssatsen, till exempel ”om det är blått, så ta en bild” på albumsidan.

Att fundera på: Fanns det fall där det var svårt att avgöra om en bild uppfyllde villkoret? Hur hade en dator hanterat de fallen?

MeeBot Dances (1 klubbträff)

1. Om ni har en MeeBot-robot från UBTECH hämtar du lektionen MeeBot Dances i Swift Playgrounds.



2. Gå igenom lektionen och se hur du kan få MeeBot att dansa genom att använda funktioner, for-loopar och villkor.
3. Sätt ihop en egen dans. Du kan även välja egen musik på sista sidan.

Att fundera på: Hur använde du kod för att anpassa dansen efter takten i musiken?

Tänk om och om igen: While-loopar

4

Introduktion (5 minuter)

I avsnitt 2 lärde du dig om funktioner och for-loopar. Vilken dans var det ni gjorde tillsammans? Hur skulle du skriva en funktion för dansen med for-loopar?

Tänk dig nu att du vill dansa den till musik. Hur ska du veta att du ska sluta dansa när låten tar slut?

Du kan använda ett villkor: om låten spelas, så dansa. Eller tydligare uttryckt med en while-loop: så länge låten spelas, dansa.

Det är något annat än en for-loop, som säger till datorn att köra ett block med kod ett visst antal gånger, till exempel dansa i en ring tio gånger. En while-loop säger till datorn att köra ett block med kod tills något händer. Till exempel dansa i en ring tills låten slutar spela.

I den här videon får du lära dig mer om while-loopar.

Övning (40 minuter)

Öppna nu Swift Playgrounds Lär dig programmera 1 och lös uppgifterna som har markerats med en grön bock i listan till höger.

Att fundera på: När använde du for-loopar och while-loopar? Hur bestämde ni vilka ni skulle använda?

While-loop: En loop som kör ett block med kod så länge ett visst villkor är sant. När villkoret är falskt slutar loopen att köras.

While-loopar

- Introduktion ✓
- Kör kod medan... ✓
- Skapa smartare while-loopar ✓
- Välja rätt verktyg ✓
- Fyra gånger fyra
- Omvänd
- Möjligheternas land
- Nästla loopar ✓
- Slumpmässiga rektanglar
- Du har alltid rätt

Tänk om och om igen: While-loopar

Gå ett steg längre

4

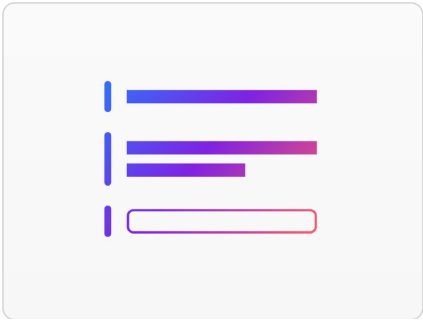
Hitta föremålet igen (1 klubbträff)

1. Göm ett litet föremål i eller i närheten av rummet.
2. Stå på en plats och använd iPad-kameran för att spela in dig själv medan du ger anvisningar så att någon annan kan hitta föremålet. Anvisningarna ska starta från den plats där du står. Använd funktioner, for-loopar och while-loopar när det går.
3. Byt nu video med en annan deltagare. Titta på den andra deltagarens video och försök hitta det gömda föremålet. Hittade du det?
4. Titta igenom båda videorna tillsammans. Skriv ner de tillfällen där ni använde for-loopar och while-loopar i anvisningarna.

Att fundera på: Blev det enklare den här gången, när ni använde loopar? Blev det svårare i något fall?

21 frågor (1 klubbträff)

1. Vi tar en titt på startpunkten Svar igen. Hämta den här versionen av den.



2. Den här svarsmallen är förberedd för att spela 21 frågor. Ta en titt på koden först. Vad kommer koden att göra?
3. Spelledaren väljer ett föremål och skriver in det som svar i koden. Spelaren kan ställa 21 ja- eller nej-frågor till spelledaren.
4. Efter varje fråga kan spelaren mata in sin gissning i live-vyn och kontrollera om det är rätt. Spelledaren måste tänka på att räcka över sin playground med live-vyn i helskärmsläge, så att spelaren inte kan se koden och det rätta svaret.

Att fundera på: Vilka kodningskoncept användes i denna playground, och på vilket sätt?

Tänk samma idé: Algoritmer

5

Introduktion (5 minuter)

Jobba tillsammans i hela gruppen. Försök komma på några saker som ni gör ofta och som kräver flera steg – till exempel att borsta tänderna eller bre en smörgås. De är alla algoritmer.

Välj ett exempel och låt flera elever ge anvisningar om hur de gör. Var anvisningarna likadana? Vilka skillnader fanns det? Gav alla anvisningar samma slutresultat?

I den här videon får du lära dig mer om algoritmer.

Övning (40 minuter)

Öppna nu Swift Playgrounds Lär dig programmera 1 och lös uppgifterna som har markerats med en grön bock i listan till höger.

Att fundera på: På hur många sätt går det att lösa varje pussel, tror du? Vem hade den kortaste algoritmen? Vem hade den intressantaste?

Algoritmer

- Introduktion ✓
- Högerhandsregeln ✓
- Justera din algoritm ✓
- Besegra en labyrint ✓
- Vilken väg att vända?
- Rulla åt höger, rulla åt vänster

Algoritm: En stegvis uppsättning med regler eller instruktioner.

Pseudokod: En informell beskrivning av kod eller ett koncept som är avsett att läsas av människor.

Tänk samma idé: Algoritmer
Gå ett steg längre

5

Vem är längst? (1 klubbträff)

1. Dela in gruppen i några mindre grupper. Varje grupp ska hitta på en metod eller algoritm som kan användas för att avgöra vem som är längst i klubben. Att man bara vet det eller kan se det direkt räknas inte som en lösning!
2. Använd era kunskaper i kodning när ni hittar på de olika stegen i algoritmen. Ni kan skriva algoritmen med en egen, påhittad pseudokod, men använd så mycket kodningsterminologi som möjligt.
3. Be grupperna att presentera sina algoritmer. Alla i gruppen ska medverka i presentationen.

Att fundera på: Vilken grupp verkade ha den effektivaste algoritmen? Vad skulle ni ändra i algoritmen om ni ville hitta den kortaste eleven?

Parrot (1 klubbträff)

1. Om ni har tillgång till en drönare från Parrot hämtar du lektionen Parrot i Swift Playgrounds.

2. Använd alla dina kunskaper i kodning för att få drönaren att lyfta, landa, röra sig åt alla håll och göra akrobatiska trick.
3. Skapa till sist en algoritm för drönarens bana från punkt A till punkt B.

Att fundera på: Hur påverkade flyghastigheten drönarens bana?

Tänk som en nyhetsbot: Variabler

6

Introduktion (5 minuter)

Föreställ dig att du håller på att flytta. Du packar dina saker i lådor och märker varje låda med en etikett som beskriver innehållet i lådan. Om en låda till exempel innehåller prispokaler kanske du märker den med Prispokaler. Det här exemplet illustrerar tre viktiga egenskaper hos en behållare.

När vi programmerar en dator använder vi något som kallas för variabler istället för lådor. Variabler liknar flyttlådor på så vis att de har en etikett (ett namn) och ett innehåll (ett värde). Värdet eller innehållet kan förändras, men inte etiketten eller namnet. Vi kan ta reda på en variabels värde eller innehåll genom att hitta den variabel som har rätt namn eller etikett.

I den här videon får du lära dig mer.

Övning (40 minuter)

Öppna nu Swift Playgrounds Lär dig programmera 2 och lös uppgifterna som har markerats med en grön bock i listan till höger.

Att fundera på: På vilket sätt var det till hjälp att använda variabler i appen?

Variabel: En namngiven behållare som innehåller ett värde. Värdet kan förändras med tiden.

Variabel

- Introduktion ✓
- Hålla koll ✓
- Höja värdet
- Öka värdet ✓
- Söka sju stenar ✓
- Tre stenar, fyra reglage
- Leta efter lika värden ✓
- Samla in reglagen
- Hämta totalen

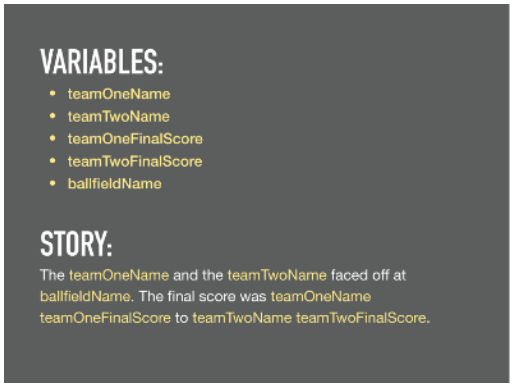
Tänk som en nyhetsbot: Variabler
Gå ett steg längre

6

Nyhetsbot (1 klubbträff)

1. I den här aktiviteten ska du skapa en nyhetsbot som är en robot som kan skriva en kort artikel automatiskt. Föreställ dig en nyhets- eller sporthändelse och fundera sedan igenom vilken information du skulle behöva för att skriva om händelsen.
2. Brainstorma fram fyra till sex variabler, till exempel lagets namn, antal mål och datum för matchen. Skriv en nyhetsartikel på två eller tre meningar med variablerna, som din nyhetsbot sedan kan fylla i med rätt information om liknande händelser.
3. Gå ihop två och två. En talar om vad variablerna ska stå för och den andra fyller i dem i artikeln. Byt sedan roller. Har ni nu två kompletta nyhetsartiklar som fungerar?

Att fundera på: Fanns det några variabelnamn som fungerade bättre än andra? Varför eller varför inte?



Sphero Arcade (1 klubbträff)

1. Om ni har tillgång till en Sphero SPRK+-robot hämtar du lektionen Sphero Arcade i Swift Playgrounds.

2. Använd funktioner och variabler för att sikta, upptäcka krockar och till sist bygga din egen version av Pong.
3. På sidan Play Sphero Pong ska du analysera koden innan du kör den eller redigerar den. Vad gör varje kommando?
4. Vilka delar av spelet kunde du förändra med hjälp av variabler?

Att fundera på: Vilka andra spel skulle du kunna anpassa på samma sätt som du gjorde med Pong?

Tänk som en arkitekt: Typer

7

Introduktion (5 minuter)

Hur många sorters byggnader kan ni komma på? Välj en typ av byggnad. Vad gör den unik? Med andra ord, vilka specifika kännetecken har den sortens byggnad? Vad händer vanligtvis i den? Vi kallar detta för beteenden. En skola, till exempel, har klassrum (egenskap) och en klocka som ringer mellan lektionerna (beteende). Håller alla med om egenskaperna och beteendena? Varför eller varför inte?

Om vi använder ett datorprogram till hjälp för att konstruera byggnaden måste vi vara väldigt specifika. Vi måste definiera typen genom att ange byggnadens *egenskaper* och metoder (det vi kallade *beteenden*).

Titta på den här videon så får du veta mer.

Övning (40 minuter)

Öppna nu Swift Playgrounds Lär dig programmera 2 och lös uppgifterna som har markerats med en grön bock i listan till höger.

Att fundera på: Vilka typer fanns det i appen? Vad initierade ni?

Typer

- Introduktion ✓
- Avaktivera en portal ✓
- Slå av och på en portal
- Ställa in rätt portal ✓
- Världens hörn
- Slumpade stenar överallt

Initiering

- Introduktion ✓
- Initiera din expert ✓
- Träna din expert
- Använda instanser av olika typer ✓
- Det krävs ett par

Datatyp: En namngiven grupp egenskaper (funktioner) och metoder (beteenden) hos ett slags data.

Initiering: När man skapar en ny instans av en datatyp, vilket inbegriper att skapa inledande värden för datatypens egenskaper.

Tänk som en arkitekt: Typer
Gå ett steg längre

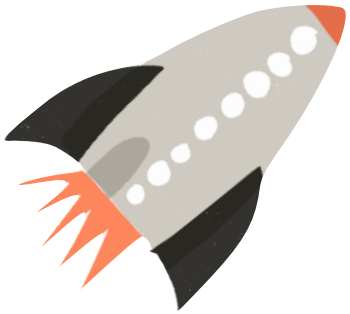
7

Var en arkitekt (1 klubbträff)

1. Välj en typ av byggnad, verklig eller påhittad.
2. Tänk ut fem eller sex variabler för att beskriva hur denna typ av byggnad ser ut. Variablerna ska inte innehålla några värden. De ska istället beskriva värdet, till exempel ”höjd” eller ”antalFönster”.
3. Lägg till värden bredvid variablerna. Det här beskriver en specifik instans av byggnadens typ. Med Swift-termer initierar vi en instans av byggnadens typ.
4. Slutför initieringen genom att skissa upp byggnadstypen i Anteckningar eller någon annan ritapp, med de angivna variablerna och värdena.
5. Gå ihop med en kompis och byt era listor över byggnadstypens variabler och värden med varandra. Rita upp varandras byggnadstyper och visa sedan ritningarna för varandra. Hur lika är de?

Att fundera på: Kan du komma på några sätt att göra ritningarna mer lika?

Typ: raketHus
antalMotorer = 4
höjd = 15 meter
antalFönster = 8
färg = rymdgrå
dörrForm = avrundad rektangel



Sten, sax och påse (1 klubbträff)

1. Hämta utmaningen Sten, sax, påse i Swift Playgrounds.

2. Gå igenom lektionen. Hur fungerar koden? Vilka kodningskoncept känner du igen?
3. Försök nu att skapa en version av spelet där du använder dina egna regler och lägger till nya åtgärder. För att göra det måste du definiera spelets typ och initiera det tillsammans med tillhörande egenskaper och metoder.

Att fundera på: Vilka var egenskaperna och metoderna i det ursprungliga spelet? Vilka nya egenskaper och metoder lade du till i din egen variant av spelet?