Måns Jonasson

Barnhack!

– kom igång med Scratch del 2

Välkommen till kom igång med Scratch del 2!

Det har hänt en del sedan jag skrev den första delen av min Scratch-kurs. Jag har hållit många Barnhack sedan dess och haft förmånen att vägleda hundratals barn genom grunderna i Scratch. Scratch 2 har också lämnat beta-stadiet och går nu att köra i skarpt läge direkt i webbläsaren.

Under mina Barnhack använder jag innehållet i den första kursen för att lära ut grunderna i Scratch under en timme. Den andra timmen låter jag sen barnen fortsätta på egen hand. De brukar då antingen fortsätta med kattspelet, eller så börjar de på ett eget spel medan jag går runt och hjälper till. Vissa frågor brukar komma upp varje gång, och de har jag med i den här fortsättningen. Dessutom brukar jag visa lite olika typer av andra spel som är enkla att bygga i Scratch för att sporra kreativiteten i rummet, och även de spelen finns med i den här delen.

Jag hoppas att ni blir inspirerade att fortsätta med Scratch och att ni delar ut era bästa spel i Scratch-communityn!

Bläckfiskspelet

I denna övning ska vi göra ett litet undervattensspel – vi ska styra en bläckfisk som simmar fram och tillbaka över havsbotten, försöka fånga stjärnor för att få poäng samtidigt som vi måste akta oss för hajar och krabbor som lurar överallt!

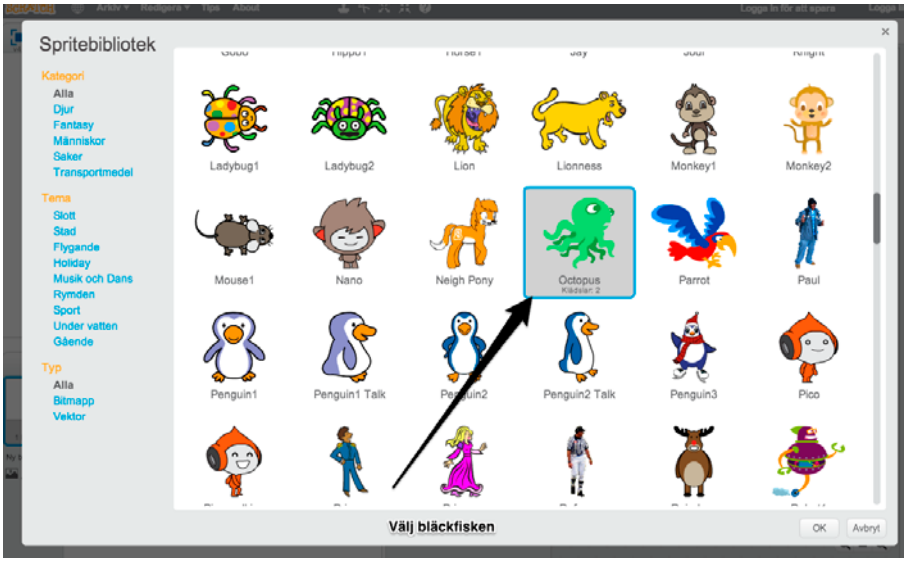
Att använda piltangenterna för att styra en sprite

I det första kattspelet vi byggde lät vi katten springa fram och tillbaka av sig själv medan vi jagade poäng genom att klicka på katten. Nu är det dags att se hur vi kan använda piltangenterna för att styra en sprite, och dessutom testar vi lite andra figurer än katten!

Först och främst, starta ett nytt och tomt Scratchprojekt. Som vanligt laddas katten Scratch in, men vi tar bort honom för nu ska vi använda andra figurer. Högerklicka på katten i sprite-listan och välj ”Radera”

Nu är det dags att ladda in en annan sprite. Scratch kommer med en massa förinstallerade figurer, och för att välja bland dessa klickar du på den lilla figuren bredvid ”Ny sprite”.

Nu får du upp en lång lista med alla inbyggda sprites. Här finns djur, människor och prylar att välja bland, men vi ska göra ett undervattenspel, så scrolla ner en bit tills du ser den gröna bläckfisken och dubbelklicka på den för att ladda den.



Bläckfisken laddas nu och lägger sig mitt på scenen. Den är ganska stor, och för att få bättre plats med annat på scenen behöver vi förminska den. I Scratch finns två verktyg högst upp i menyraden för att förstora och förminska, de ser ut som fyra pilar som pekar åt olika håll. Den högra av dem är fyra pilar som pekar inåt, och det är ”förminska”.

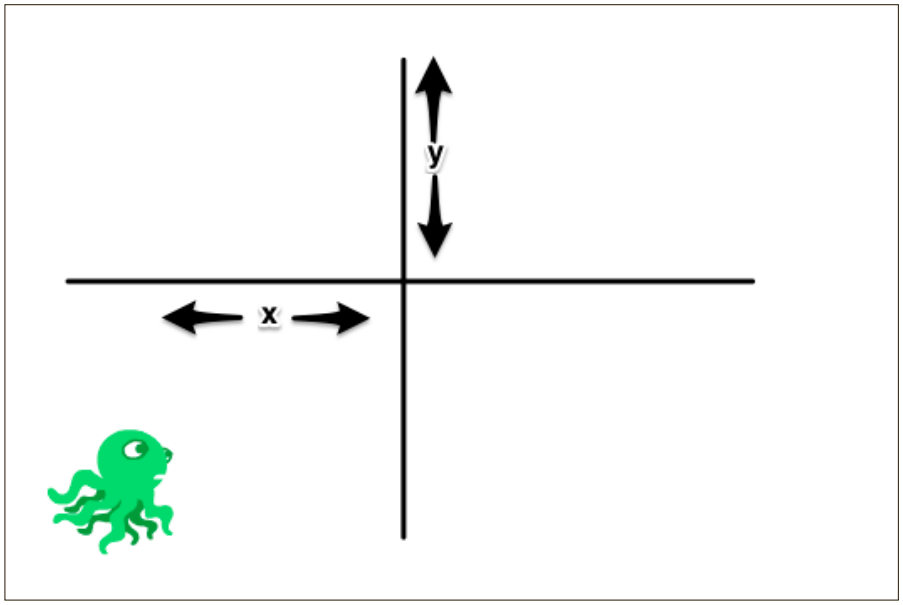
Klicka först på den ikonen för att välja verktyget ”förminska” och sen på bläckfisken flera gånger tills den är lagom stor, och dra sen ner den till botten av scenen:

Nu ska vi se till att bläckfisken kan röra på sig, men istället för att låta den springa fram och tillbaka på skärmen av sig själv ska vi använda piltangenterna för att styra den själva.

Koordinater

Den här gången ska vi flytta på bläckfisken på ett annat sätt än vi flyttade katten tidigare. Istället för att använda blocket ”gå 10 steg” ska vi använda blocket ”ändra x med 10”. Men först måste vi prata lite om koordinater.

På scenen kan vi flytta på saker genom att ändra eller sätta deras x-värde och y-värde. Vad är då detta? Jo, alla sprites på scenen har en position som anges i koordinater, x och y. En sprites x-koordinat representerar den horisontella position som den befinner sig på, och y-koordinaten anger den vertikala positionen.



På detta sätt är hela scenen uppstyckad i tusentals pixlar, som alla kan anges med ett x- och ett y-värde. Mittpunkten på scenen är nollpunkt för både x och y, och när saker ligger till höger om mitten har de ett positivt x-värde medan allt till vänster om mitten har ett negativt x-värde. På samma sätt är allt ovanför mitten på scenen på positivt y-värde och allt under mittlinjen har negativa y-värden.

Genom att berätta för Scratch vilka x- och y-värden vi vill att en sprite ska ha kan vi alltså flytta på en figur. Nu när vi har lagt bläckfisken längst ner till vänster har den både negativt x-värde och y-värde. Du kan peka på den med musen för att se ungefär vilka x- och y-positioner den befinner sig på – koordinaterna visas alltid nere till höger under scenkanten.

Tänk på att det alltid är muspekarens x- och y-värde som visas där, inte var en viss sprite befinner sig! För att ta reda på exakt vilka koordinater en sprite befinner sig på behöver vi gå in på inställningarna för den spriten genom att klicka på den lilla ”i”-knappen bredvid bilden i sprite-listan, där vi tidigare ställde in rotationsstilen för en sprite.

Där ser vi vilken position en sprite har vid just detta tillfälle. Just nu spelar det ingen roll exakt var bläckfisken ligger, så stäng inforutan igen genom att klicka på den blå pilen i vänstra hörnet.

Så genom att ändra x- och y-värdena för en sprite kan vi flytta på den. Det är precis det vi ska göra nu, men vi behöver skapa lite struktur för vår kod först, på samma sätt som vi gjort tidigare. Börja med att gå till scriptkategorin ”händelser” dra in ett startblock, ”När grön flagga klickas på” till bläckfiskens script-yta. Gå sen till kategorin ”kontroll” och koppla på ett loop-block, nämligen ”för alltid”. Inuti loopen lägger vi ett villkorsblock, nämligen ”om – då”.

Nu har vi starten på ett program, men vi ska bygga vidare. Till att börja med ska vi använda ett nytt block för att känna av om en piltangent är nedtryckt. Gå till kategorin ”känna av” och dra in blocket ”tangent [mellanslag] nedtryckt?” in i ”om – då”-blocket. Klicka sen på den lilla svarta pilen bredvid ordet ”mellanslag” i blocket och välj ”högerpil” istället.

Nu har vi skapat ett program som körs ”för alltid” och som känner av ifall tangenten ”högerpil” på tangentbordet är nedtryckt, och använder ett villkor för att i sådana fall göra någonting. Nu är det dags att flytta på bläckfisken!

Flytta på en sprite genom att ändra koordinater

Först och främst vill vi att bläckfisken ska vända sig åt rätt håll när vi flyttar på honom, så vi behöver ett block som pekar spriten. Under kategorin ”rörelse” finns blocket ”peka i [90] riktning”. Dra in det och lägg det inuti villkorsblocket. Om du klickar på den lilla svarta pilen i det här blocket så ser du att du kan välja olika riktningar som anges i grader, men eftersom höger (90 grader) är förvalt kan vi låta det vara just nu. Det här blocket vänder ju på bläckfisken i den riktning vi vill, och vi använde det förut för att vända katten åt rätt håll innan vi använde ”gå [10] steg” för att flytta på katten. Nu ska vi ju använda x-koordinater istället, så egentligen behöver vi inte vända på bläckfisken för att flytta den åt rätt håll, men det ser konstigt ut om bläckfisken inte vänder sig, för då simmar han baklänges ibland!

Precis som förra gången kommer dock spriten att hamna uppochned om vi vänder honom utan att ändra rotationsstil. Tidigare gick vi in i spritens inställningar för att ändra rotationsstil, men det finns ett sätt att göra det med block också, och det ska vi göra nu. Under kategorin ”rörelse” finns blocket ”sätt rotationsstil [vänster–höger]” som gör precis det vi vill. Det här blocket räcker det om vi kör en gång, så dra in det och släpp det precis under startblocket, ovanför loopen.

Under kategorin ”rörelse” finns också blocket ”ändra x med [10]”. Dra in det och lägg det inuti villkorsblocket, men ändra sen siffran tio till en trea. Det här flyttar alltså på bläckfisken genom att öka x-koordinaten med tre i loopen, vilket gör att bläckfiskens x-värde blir högre och alltså rör sig till höger i koordinatsystemet.

Nu kan du testa genom att klicka på den gröna flaggan. Inget händer genast, men om du trycker ner högerpilen på ditt tangentbord så börjar bläckfisken simma till höger. För att han ska kunna simma åt andra hållet behöver vi skapa ett till villkor och lägga in samma block en gång till, men ändra lite saker.

Börja med att ta villkorsblocket ”om – då” från kategorin ”kontroll” och släpp det under vårt första villkorsblock, men fortfarande inne i loopen. Gå sen till ”känna

av” och ta blocket ”tangent [mellanslag] nedtryckt” igen, släpp det i villkoret men ändra sen ”mellanslag” till ”vänsterpil”.

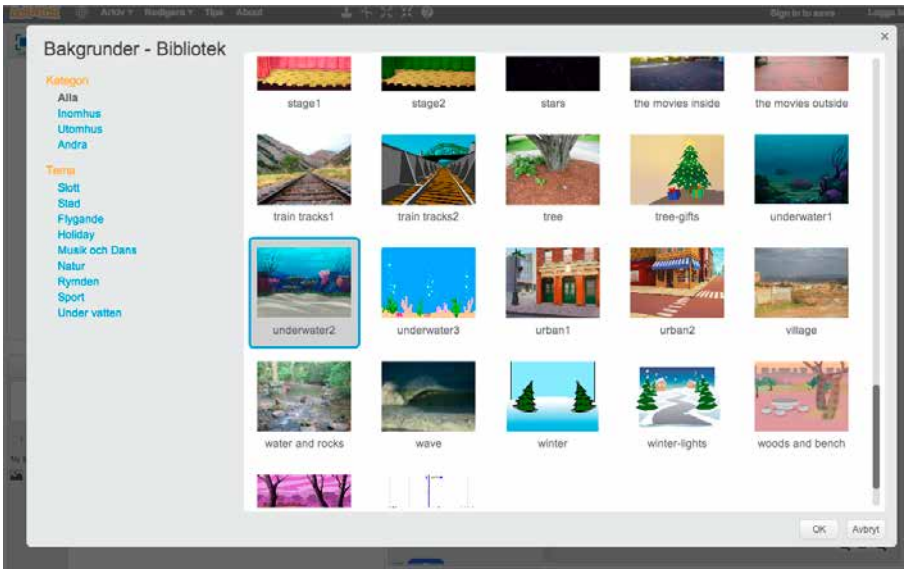
Gå sen till kategorin ”rörelse” och ta blocket ”peka i [90] riktning” och släpp det inne i vårt nya villkor, men ändra nu ”90” till ”-90” (vänster). Ta sen blocket ”ändra x med [10]” och släpp det under peka-blocket. Nu blir det lite klurigt, för om vi ökade värdet på x-koordinaten i förra villkoret för att flytta bläckfisken till höger måste vi nu istället minska värdet på x-koordinaten för att flytta bläckfisken till vänster. Ändra siffran tio till ”-3”, alltså minus tre. Det här kommer att dra tre ifrån x-värdet och alltså flytta spriten till vänster i koordinatsystemet.

Testa genom att klicka på den gröna flaggan igen. Nu kan du låta bläckfisken simma fram och tillbaka genom att trycka på vänster- och högerpilarna på ditt tangentbord.

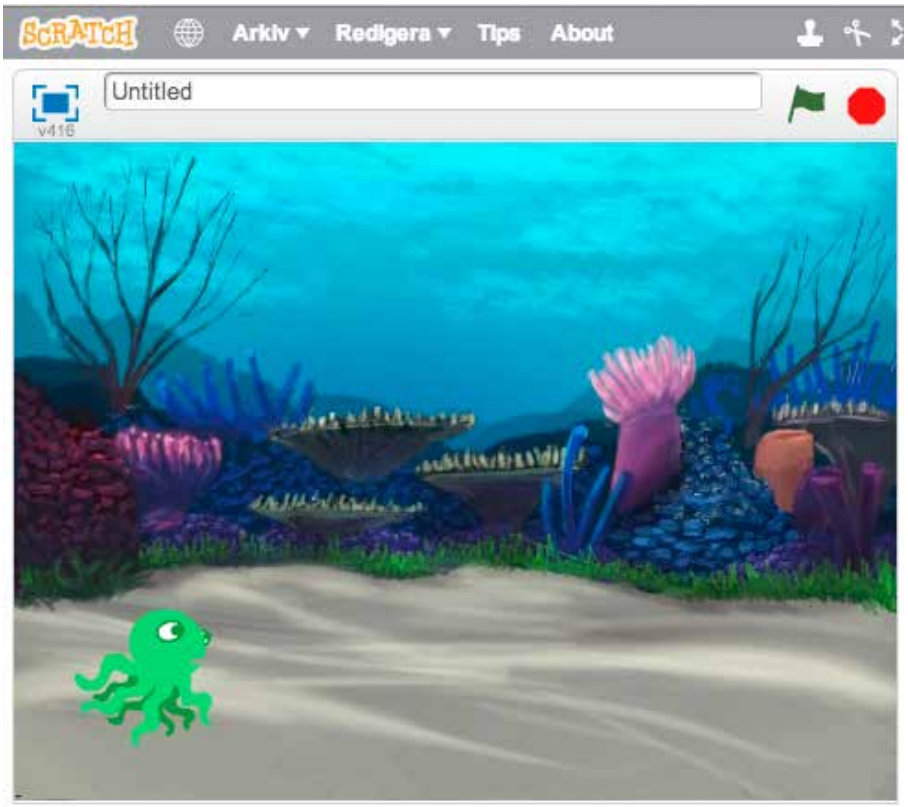
Ladda bakgrund

Vi gör ett undervattensspel, så låt oss hämta en passande bakgrund. Till vänster om sprite-listan, under ikonen som visar scenen, finns en liten ikon som ser ut som en tavla. Klicka på den för att öppna det inbyggda biblioteket med bakgrunder.

I den långa listan med inbyggda bakgrundsbilder finns, en bra bit ner, en som heter ”underwater2” som är perfekt för det här spelet. Dubbelklicka på den för att ladda in den och lägga den som bakgrund till scenen.



Nu är bakgrunden laddad, och vår bläckfisk ligger ovanpå den. Se till att han ligger rätt i höjd, så att det ser ut som att han simmar på havsbotten.



Animera bläckfisken

När bläckfisken simmar fram och tillbaka så glider han liksom över havsbotten. Just denna sprite kommer med två ”kostymer”. Vi pratade om klädslar och kostymer redan i förra kursen, och även denna gång kan vi växla mellan de två kostymerna genom att skapa ett separat script för det.

Klicka på bläckfisken i sprite-listan för att välja den, så ser du scriptet för att röra bläckfisken igen. Gå nu till kategorin ”händelser” och dra in ett nytt ”när grön flagga klickas på”-block och lägg det bredvid vårt förra program i scriptytan. Gå sen till ”kontroll” och ta en loop – ”för alltid” och lägg den under startblocket. Gå sen till ”utseende” och ta blocket ”nästa klädsel” och lägg det inne i loopen. Till slut går du tillbaka till ”kontroll” och väljer ”vänta [1] sekunder” och lägger det under ”nästa klädsel”, inne i loopen. Ändra ettan så att det står ”vänta 0.2 sekunder” istället.

Testa genom att klicka på den gröna flaggan. Nu ser det ut som att bläckfisken simmar i vattnet eftersom den växlar mellan de två klädslarna. Du kan fortfarande styra bläckfisken genom att trycka på höger- eller vänsterpiltangenterna.

Kloner – ett sätt att skapa många likadana figurer på scenen

Nu ska vi se till att det finns något att göra i spelet utöver att styra bläckfisken. Vi ska ha stjärnor som dyker upp på havsbotten som vi ska försöka fånga för att få poäng, och vi ska ha farliga hajar som kommer simmande och elaka krabbor som dyker upp och försöker knipsa oss. Alla dessa objekt ska vi bara programmera en gång, och sen låta Scratch skapa ett antal av dem åt oss genom att använda oss av ”kloner”.

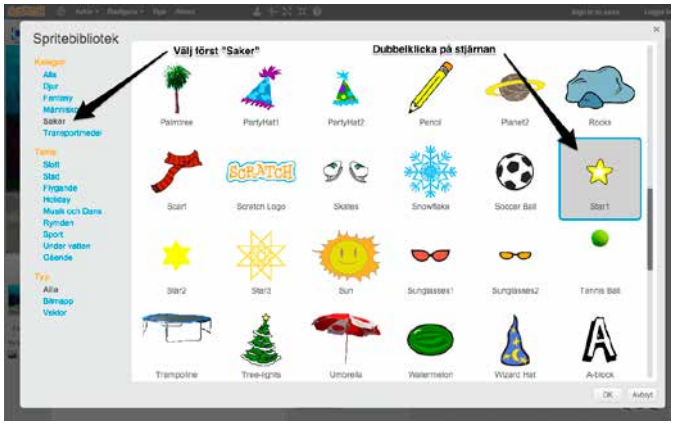
Kloner kan sägas vara Scratch introduktion till objektorienterad programmering. Poängen med OOP, som det förkortas, är att du bara behöver skriva kod en gång, men kan återanvända den i olika ”instanser”, eller kloner som det kallas i Scratch. I det här fallet ska vi alltså programmera beteendet för en stjärna, en haj och en krabba, och sen ska vi använda vårt scenscript för att dra igång olika antal av dessa objekt. Eftersom Scratch har särskilda block för att hantera kloner är det inte så komplicerat, men det kan vara lite klurigt att förstå hur det hänger ihop från början.

Tänk dig att vi skulle vilja göra ett spel med en cowboy som skjuter en pistol. Vi skulle då ha en sprite för själva cowboyen, skapa script för den som gör så att han kan gå fram och tillbaka och kanske skjuta. Men själva kulorna från pistolen, hur skulle vi programmera dem om vi vill att man ska kunna skjuta många skott i taget? Istället för att ladda in flera likadana sprites och lägga exakt samma script i allihopa kan vi med hjälp av kloner ladda en skott-sprite, programmera den en gång och sen skapa hur många sådana skott vi vill, genom att säga åt Scratch att sätta igång nya kloner av den.

Varje klon kan bara göra precis det som vi programmerat in i den enda sprite vi laddat. Om vi vill att olika skott ska bete sig på olika sätt får vi antingen lägga in programmeringen för att hantera det, eller ladda fler sprites. I vårt undervattenspel ska vi bara använda en stjärna, en haj och en krabba. Kloningen sköter resten.

Ladda sprite som ska bli klonad

Vi börjar med stjärnan. I spelet ska vi se till att det dyker upp stjärnor på olika ställen på havsbotten. Målet med spelet är att fånga så många som möjligt av dem utan att bli uppätna av hajen eller knipsade av krabban. Ladda in en ny sprite genom att klicka på den lilla gubben ovanför sprite-listan.



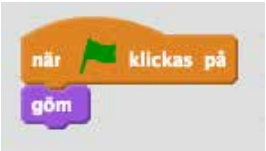
I det stora sprite-biblioteket kan du snabbare hitta rätt genom att välja en kategori till vänster. Stjärnan hittar vi i kategorin ”saker”, så klicka på ”saker” till vänster och scrolla sen ner för att hitta ”Star1”. Dubbelklicka på den för att ladda in den.

Stjärnan laddas in och lägger sig mitt på scenen. Där vill vi ju inte ha den, för bläckfisken simmar bara på botten. Vi behöver alltså lägga den någonstans på botten, där vi kan nå den sen. Dra ner stjärnan med musen så att den ligger någonstans på botten. Det spelar ingen roll var i sidled den ligger, för vi ska snart använda programmering för att flytta på den. Först av allt behöver vi se till att den är osynlig när spelet startar, och sen behöver vi skapa själva programmet som ska köras när stjärnan startar som en klon.

Göm en sprite

Ibland i spel är det praktiskt att kunna bestämma när saker ska visas på scenen och inte. I Scratch är det väldigt enkelt, för vi har två block under ”utseende” som heter ”visa” och ”göm” och de gör precis vad de säger. De flyttar alltså faktiskt inte på spriten, utan gör den bara osynlig eller synlig.

I vårt spel ska stjärnan inte visas från början, så vi tar ett startblock, ”när grön flagga klickas på” under ”händelser” och drar ut det till stjärnans script-yta. Klicka sen på ”utseende” och dra in blocket ”göm” och lägg det under startblocket.



Klicka på den gröna flaggan nu, så ser du att stjärnan genast blir osynlig. Om du av någon anledning behöver visa en osynlig sprite igen så kan du alltid gå in i inställningarna för den spriten och klicka på rutan ”visa”. På samma sätt kan du gömma en sprite manuellt.

Nu när vi skapat det lilla programmet som gömmer stjärnan när spelet börjar är det dags att sätta igång med scriptet som slumpmässigt ska bestämma var stjärnan ska hamna, visar den i ett antal sekunder och sen gömmer den igen. Eftersom vi vill använda kloner måste vi använda ett alldeles särskilt startblock, nämligen ”när jag startar som klon”, och det ligger märkligt nog inte under ”händelser”, utan under ”kontroll”. Leta upp det och dra in det till stjärnans scriptyta, bredvid vårt andra program.

Det här startblocket körs alltså bara när vi säger åt Scratch att vi vill ha en ny klon av denna sprite, vilket vi kommer att göra lite senare, när vi programmerat klart stjärnans beteende.

Slumptal

För att spelet ska bli omväxlande kan vi ju inte låta stjärnan ligga på samma ställe varje gång den dyker upp, då skulle det bli alldeles för lätt för vår bläckfisk att hänga där och vänta på att den kommer. Därför behöver vi slumpa positionen som stjärnan ligger på innan den visas. I Scratch finns en väldigt enkel funktion för att skapa ett slumptal, och den kan vi använda för att slumpa fram stjärnans x-värde, det vill säga var i sidled som stjärnan ska dyka upp. Den måste ju alltid ligga på botten, annars kan inte vår bläckfisk nå den – så y-värdet behöver vi aldrig ändra på.

I Scratch är scenen exakt 480 pixlar bred, och eftersom mitten av scenen har x-värdet noll är högra kanten av scenen x 240 och vänstra kanten av scenen x -240. För att stjärnan ska hamna på olika ställen i sidled kan vi alltså slumpa fram ett x-värde som ligger mellan -240 och 240, men då skulle vi riskera att en del av stjärnan hamnar utanför scenen, så vi väljer att använda -220 och 220 istället.

Under kategorin ”rörelse” finns ett annat block som hanterar x-värde för en sprite, nämligen ”sätt x till [0]”. Till skillnad från det förra x-blocket vi använde så bestämmer det här blocket exakt x-värde och struntar i vilket x-värde spriten för tillfället har. Det funkar på samma sätt som våra variabelblock vi använde i förra kursen. Ta tag i blocket och dra in det under ”när jag startar som klon”.

Nu ska vi så ta fram vårt slumptal, och det gör vi med ett block som finns under ”operatorer”. Blocket heter ”slumptal [1] till [10]”, och det kan vi dra in och släppa i rutan i blocket där det nu står ”sätt x till [0]” så att det istället står ”sätt x till slumptal 1 till 10”. Se till att släppa det på rätt ställe så att det verkligen hamnar i rutan och inte utanför.

Istället för ett och tio ska vi ju ha -220 och 220, så klicka i rutorna och skriv in våra värden, så att det står ”sätt x till slumptal -220 till 220”.

Nu har vi med hjälp av det här blocket flyttat på stjärnan, fast det kommer inte att synas direkt eftersom stjärnan är gömd när vi börjar. Vi behöver alltså visa stjärnan efter att vi flyttat på den, så gå till kategorin ”utseende” och dra in blocket ”visa”.

Nu vill vi att stjärnan ska ligga där i några sekunder och sen försvinna igen. Vi kan använda slumpen igen för att bestämma om stjärnan ska ligga där i två sekunder eller fem. Gå till ”kontroll” och dra in blocket ”vänta [1] sekunder”. Gå sen tillbaka till operatorer och dra in blocket ”slumptal [1] till [10]” och släpp det i rutan där det nu står en etta. Ändra de två slumptalen till två och fem, så att stjärnan väntar mellan två och fem sekunder varje gång.

Efter att stjärnan legat där i några sekunder ska den försvinna igen, så gå till ”utseende” och dra in blocket ”göm”.

Sist av allt behöver vi radera klonen, för efter att stjärnan visats och gömts är den färdig och ska tas bort. Att radera en klon innebär bara att just den instans av spriten som vi startat försvinner, inte att själva spriten och all dess kod försvinner. Gå till ”kontroll” och dra in blocket ”radera klonen” och lägg det sist.

Vårt program väntar alltså på att vi ska säga till att vi vill starta en klon av stjärnan, och då flyttar den på stjärnan, visar den, väntar en stund och gömmer den igen. Sist av allt raderar klonen sig själv, eftersom den är färdig.

Skapa en klon

Nu ska vi använda vår stjärna genom att skapa kloner av den. Jag tycker att det är prydligt att lägga de program som är övergripande för hela spelet i scenens scriptyta, så klicka på scenen till vänster om sprite-listan. Scenen har inga program än, så det är tomt i scriptytan till att börja med.

Vi vill att stjärnorna ska dyka upp vid olika tillfällen, så det är dags för slumpen igen. Börja med att dra in startblocket ”När grön flagga klickas på” och koppla på loopen ”för alltid”. Gå sen till ”kontroll” och lägg blocket ”vänta [1] sekunder” inne i loopen. Gå till ”operatorer” och dra in ”slumptal [1] till [10] sekunder” i rutan där det står en etta. Ändra värdena för slumptalen till en etta och en femma, så att det står ”slumptal 1 till 5 sekunder”.

Nu ska vi skapa själva klonen. Under kategorin ”kontroll” finns blocket ”skapa klon av”. Dra in det och lägg det under vänta-blocket. Klicka på den lilla svarta pilen för att välja ”Star1” så att det står ”starta klon av Star1”. Nu kommer vi att under hela spelets gång skapa nya kloner av stjärnan efter ett slumpmässigt intervall.

Testa nu genom att klicka på den gröna flaggan. Bläckfisken simmar och då och då dyker en stjärna upp på botten, men försvinner igen efter ett tag. Om du tittar en stund kommer du att få se att det kan bli flera stjärnor samtidigt! Vi har ju sagt att en ny klon av stjärnan ska skapas varje 1–5 sekunder, så med lite tur kan vi få flera stjärnor att dyka upp innan de gamla hunnit försvinna.

Kollisioner – rör de vid varandra?

Nu när vi har stjärnor som dyker upp på havsbotten är det dags att börja räkna poäng. Varje gång bläckfisken lyckas ta en stjärna ska stjärnan försvinna, man ska få en poäng och så kan vi spela upp ett litet ljud så att man hör att man lyckas.
Redan i förra kursen räknade vi poäng med variabler, så vi gör på liknande sätt denna gång. Gå till kategorin ”data” och klicka på ”skapa en variabel”. Döp den till ”Poäng” och tryck på ”OK”. Nu dyker vår poängtavla upp på scenen och vi får variabelblock att använda oss av.

Klicka på stjärnan i sprite-listan så att du ser stjärnans program igen. Nu behöver vi skapa ett program till här som håller koll på om stjärnan och bläckfisken rör vid varandra, och även detta program körs ju när vi klonat stjärnan, så gå till ”kontroll” och dra in ett nytt startblock ”när jag startar som klon” och lägg den bredvid dina andra program för stjärnan. Dra sen in en för alltid-loop och ett om-då-villkor i loopen.

Scratch kan hålla koll på om sprites rör vid varandra, vilket är praktiskt i väldigt många olika slags spel. I det här fallet vill vi använda den funktionen för att se om stjärnan rör vid bläckfisken, så gå till ”känna av” och dra in det diamantformade blocket ”rör” och släpp det i om-då-blocket. Klicka på den lilla svarta pilen och välj ”octopus” så att vi kollar om ”jag” (alltså klonen) rör vid bläckfisken och inget annat.

Vi ska få en poäng när bläckfisken rör vid stjärnan, så gå till ”data” och dra in blocket ”ändra [poäng] med 1”. Gå sen till kategorin ”ljud” och dra in blocket ”spela

ljudet [pop]", så att det låter lite när vi lyckas ta en stjärna. När vi fått poäng och spelat ljud så ska stjärnan försvinna, så gå till "kontroll" och dra in blocket "radera klonen".

Vårt nya program körs alltså när stjärnan startas som en klon, det loopar och kollar om klonen rör vid bläckfisken, och om det händer får vi en poäng, spelar ett ljud och raderar klonen.

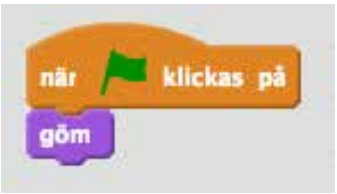
Testa nu spelet och se om du hinner simma ifatt stjärnorna och att du får poäng när du lyckas. Ibland hör du ett "pop" och får ett poäng utan att du hinner se någon stjärna, och det är när det slumpar sig så att en stjärna dyker upp på precis den plats där bläckfisken råkar befinna sig för tillfället. Gratispoäng!

MERA KLONER – HAJEN

Nu kan vi ju spela spelet, men för att det ska bli lite svårare ska vi lägga till en farlig haj och en krabba. Även dessa ska vi göra som kloner, men de ska bete sig på lite olika sätt. Vi börjar med hajen.

Ladda in en ny sprite genom att klicka på den lilla gubben ovanför sprite-listan igen. Hajen som vi ska använda heter "Shark" och ligger nästa längst ner i kategorin "Djur".

Nu ska vi bygga programmet för hajen. Vi vill ju inte att hajen ska synas när spelet börjar, så ta en "när grön flagga klickas på" och ett "göm".



Gå sen till "kontroll" och dra in ett "när jag startar som klon", för även hajens program ska främst köras när vi klonar den. Hajen ska få simma snett över skärmen, från det övre vänstra hörnet och ner till höger, fast lite olika varje gång. Det kan vi uppnå genom att slumpmässigt rotera hajen lite innan den visas och börjar åka.

Först av allt när klonen skapas ska vi placera hajen på sin startposition i övre vänstra hörnet. Hajen är ganska stor, så den behöver programmeras att flytta en bit utanför scenen för att dyka upp på ett snyggt sätt. Gå till ”rörelse” och dra in ett block som heter ”gå till x: y:”. Jag mätte upp hajen och kom fram till att en bra startposition är x -270, y 235. Det här kan du ändra senare om du vill, men skriv in ”-270” i rutan för x-värde och ”235” i rutan för y-värde nu.

För att hajen ska simma på lite olika ställen varje gång ska vi nu rotera den med lite slumptal. Under ”utseende” finns blocket ”peka i [90] riktning”. Dra in det och släpp det under ”gå till”-blocket. Gå sen till ”operatorer” och dra in slumptals-blocket i rutan för riktningsvärde. Mellan 110 och 180 grader verkar vara lagom för hajen, så skriv ”110” i den första rutan och ”180” i den andra.

Efter att vi pekat hajen i rätt riktning är det dags att visa den och sätta igång att flytta på den. Dra in blocket ”visa” som ligger under kategorin ”utseende” och släpp det efter peka-blocket.

För att hajen ska simma fram av sig själv behöver vi nu en för alltid-loop. Gå till ”kontroll” och dra in loopen och släpp den under ”visa”. Nu kan vi använda det gamla fina blocket ”Gå [10] steg” för att flytta på hajen – det ligger under ”rörelse”. Lägg det inne i loopen och ändra siffran tio till en fyra, så rör sig hajen lagom fort.

Till sist behöver vi ta bort hajen när den simmat fram över underkanten, så nu behöver vi använda ett villkor igen för att kolla om hajen simmat färdigt. Vi kan använda hajens y-värde för att kolla var den befinner sig i höjdled, och ett villkors-block för att göra någonting när den kommit fram. Gå till ”kontroll” och ta ett ”om – då” och släpp det i loopen, under gå-blocket. Det vi vill kolla är om hajens y-läge är under scenens kant. Scenens mitt är ju nollpunkten, och när vi rör oss nedåt på scenen blir det negativa y-värden. Nederkanten av scenen är y -180, men hajen ska få en chans att simma lite till så att den helt försvinner ur bilden, så vi använder -230 istället.

För att jämföra två värden behöver vi gå till kategorin ”operatorer” och hitta det diamantformade blocket som kollar om ett värde är mindre än ett annat.

Släpp det blocket i den diamantformade rutan i villkorsblocket. Vi kan nu kontrollera om ett värde är mindre än ett annat, och vi vill ju veta om hajens y-värde är mindre än -220 för att se om den simmat över underkanten. Att hämta hajens

y-läge kan vi göra under kategorin ”rörelse”, med blocket ”y-läge”. Dra in blocket i den ena vita rutan i den gröna operatorn, klicka sen på den andra vita rutan och skriv in ”-230”.

Det vi vill ska hända om hajen simmat över kanten är att den ska försvinna helt, så gå till ”kontroll” och dra in ett ”radera klonen”-block.

Det här programmet fick många olika delar, men om vi läser det uppifrån och ned är det inte så jobbigt. När klonen startas, lägg hajen i det övre vänstra hörnet av scenen, peka den i lite slumpad riktning och visa den. Sätt sen igång en loop så att hajen simmar framåt i den riktning den blivit pekad, känn av när den simmat förbi kanten och radera klonen då.

Nu behöver vi ett script som startar haj-klonerna, och det lägger vi precis som scriptet som visar stjärnorna i scenens script. Klicka alltså på den lilla undervattensbilden till vänster om sprite-listan för att visa scenens script. Nu ska vi skapa ett script som är nästan identiskt med det vi använde för att starta kloner av stjärnan, men hajen behöver inte dyka upp lika ofta, så vi kan använda andra tal i slumpgeneratorn. Mellan fem och tio sekunders paus mellan hajarna känns lagom.

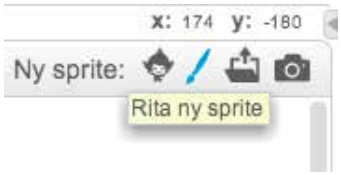
Vi har fortfarande inte sett till att något händer när hajen lyckas äta upp bläckfisken. Klicka på hajen igen i sprite-listan så sätter vi igång.

Det här är ju nästan exakt samma program som vi skapade för stjärnan, men vi vill spela ett annat ljud och dessutom att spelet ska ta slut när hajen rör bläckfisken. Först och främst behöver vi ett nytt ”När jag startar som klon”-block, som ligger under ”kontroll”. Ta sen en ”för alltid”-loop och inuti den ett ”om-då”-villkor.

Under ”kontroll” har vi blocket ”rör” igen, lägg det i villkoret och välj ”Octopus” så att vi känner av om hajen rör vid bläckfisken. Nu kan vi få saker att hända om en haj lyckas få tag i en bläckfisk, men vi behöver ju också meddela spelaren att det är game over. Det här är ett bra tillfälle att titta på hur man gör egen grafik i Scratch.

Gör en egen sprite

Vi vill göra en skylt som det står ”GAME OVER” på. Eftersom den här skylten ska dyka upp först när en haj tagit bläckfisken kan vi göra skylten som en separat sprite som vi gömmer när spelet börjar och sen visar när spelet är slut. För att rita en helt egen sprite kan vi använda Scratch inbyggda ritprogram, klicka på den lilla penseln över sprite-listan för att skapa en ny, tom sprite:



Vi får en ny, tom sprite som heter ”Sprite1” eller liknande, och vi kommer direkt in i det inbyggda ritprogrammet. Nu kan vi skapa en enkel skylt genom att använda verktyget för rektanglar och textverktyget. Rita ett par rektanglar och använd fyllningshinken för att fylla dem så att du får en vit skylt. Klicka sen på textverktyget och skriv in ”GAME OVER” på skylten.

Den här skylten ska vi bara visa när spelet är slut, men medan vi ritar den så visas den direkt på scenen. Nu ska vi programmera den. Klicka därför på fliken ”Script” för denna sprite och dra in ett ”när grön flagga klickas på” och ett ”göm”. Nu kommer ju skylten att försvinna så fort vi startar spelet, vilket är precis vad vi vill. Men för att vi ska kunna programmera in att skylten ska visas så fort hajen tagit bläckfisken behöver vi ju kunna få information från vårt andra program. Det gör vi i Scratch med hjälp av något som kallas ”meddelanden”.

Meddelanden – låt programmen prata med varandra

Hittills har vi skapat ett antal olika program som ”bott” i en sprite eller scenen. De har skött sig själva, och inte haft något med varandra att göra mer än att känna av om en sprite rör vid en annan. Men om nu vill att något ska hända med en sprite när något annat händer i vårt spel behöver vi använda meddelanden, som är ett sätt att låta olika program prata med varandra, oavsett var de bor.

Ett meddelande är helt enkelt ett litet tillrop mellan två program. Vi kan skicka ett meddelande och vi kan lyssna på ett meddelande. Alla meddelanden skickas genom hela Scratch, så om du till exempel skickar ett meddelande från bläckfisken så behöver du inte berätta vem du skickar meddelandet till – det räcker att börja lyssna där du vill plocka upp det. Du kan skapa många olika meddelanden och använda dem till olika saker.

Nu ska vi skapa ett meddelande som skickas från hajens script när den lyckats fånga bläckfisken, och som tas emot av skyltens script. Klicka på hajen i sprite-listan för att gå tillbaka till hajens program. Här finns vårt script som vi gjorde för att känna av om hajen rör vid bläckfisken. Gå nu till kategorin ”kontroll” och hitta blocket ”skicka [message1]”, näst längst ner. Dra in det och släpp det i villkoret.

Scratch har redan skapat ett meddelande åt oss, men vi tar och gör ett eget. Klicka på den lilla svarta pilen i blocket du just drog in och välj ”nytt meddelande”. I rutan som poppar upp kan du skriva vad du vill kalla ditt nya meddelande. Vi kallar det ”game over” den här gången så skriv det i rutan. När du sparat meddelandet ser du att blocket nu valt ditt nya meddelande istället, så det står ”skicka [game over]”. När hajen fångar bläckfisken kommer nu meddelandet ”game over” att skickas över hela Scratch, och alla andra script kan ”lyssna” på det.

Efter att vi skickat meddelandet ska vi också radera klonen, så gå till ”kontroll” och dra in ”radera klonen”.

Nu ska vi se till att vår sprite med skylten lyssnar på meddelandet och visar sig när det tas emot. Klicka på skylten i sprite-listan så att du ser skyltens program igen. Under kategorin ”Händelser” finns startblocket ”När jag tar emot [game over]”.

Det senaste meddelandet som skapats är förvalt, men om det står något annat där kan du klicka på den lilla svarta pilen och välja ditt meddelande. Dra in blocket och börja på ett nytt script. Det vi ska göra när vi tar emot meddelandet är att visa skylten och sen avsluta spelet, så gå till ”utseende” och dra in ett ”visa”.

Ladda in ljud

Nu skulle jag vilja spela ett riktigt läskigt ljud om vi blir uppätna, för då är spelet slut. Klicka på fliken ”ljud” som ligger ovanför scriptlistan så ser du att vi bara har ett ljud till hajen just nu, nämligen ”pop” som vi använde när bläckfisken fångar en stjärna. Vi laddar in ett nytt ljud från de som kommer inbyggda i Scratch genom att klicka på den lilla ikonen som ser ut som en högtalare.

Även här kan man bläddra mellan ljuden genom att klicka på kategorierna till vänster. Under kategorin ”Elektronisk” ligger ett ljud som passar bra, det heter ”screech”. Dubbelklicka på det för att ladda in det. Klicka sen på ”Script”-fliken så att du ser skyltens program igen.

Det finns ett par olika block som har med ljud att göra under kategorin ”Ljud”. Det vi använt förut heter ”spela ljudet” men det finns ett block under det som heter ”spela ljudet tills färdigt”, och det ska vi använda denna gång. Dra in det och släpp det under ”visa”. Där ska ljudet ”screech” redan vara valt, men dubbelkolla för säkerhets skull.

Nu kommer det läskiga ljudet att spelas när spelet tar slut.

Stoppa spelet

För att spelet verkligen ska ta slut behöver vi också se till att stoppa allt efter att det blivit game over. Gå till ”kontroll”. Leta upp blocket ”stoppa [alla]” och dra in det efter ljudblocket.

”Stoppa alla” gör just det – samma sak som om du hade tryckt på stoppknappen högst upp. När spelet är slut måste man starta om det genom att trycka på den gröna flaggan igen.

Initscript – när spelet börjar

Nu börjar spelet ta form, men vid det här laget har du kanske upptäckt att vi inte nollställer poängen varje gång spelet startas. Det här pratade vi om även i förra kursen, men vi behöver ett ”initscript”, något som nollställer det som behövs varje gång spelet startas. Det här scriptet kan vi lägga i scenens script, så klicka på scenens ikon bredvid sprite-listan igen.

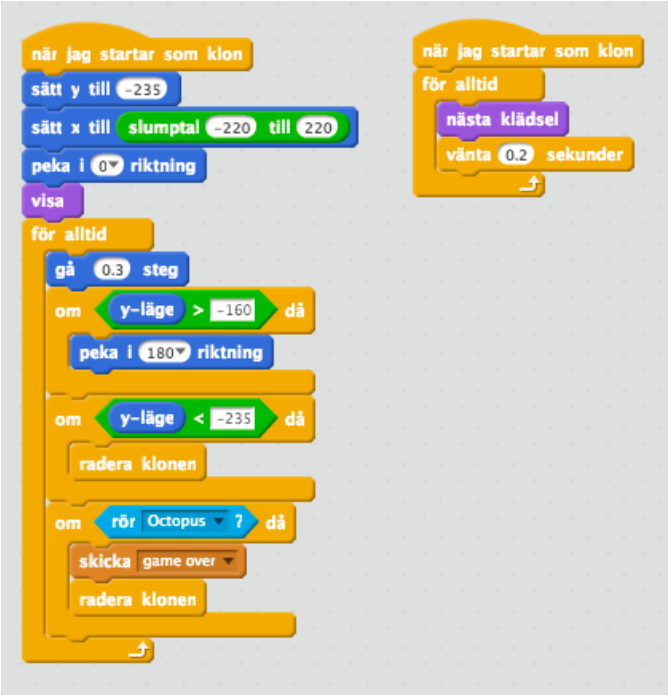
Dra in ett nytt startblock, ”när grön flagga klickas på”, och gå sen till ”data” och dra in ett ”sätt [poäng] till [0]”. Nu har vi sett till att poängen nollställs i början av varje spelomgång.

Ännu en klon – krabban

Hajen är farlig, men jag tänkte att vi skulle lägga in en krabba också som kommer upp från botten av havet då och då och försöker knipsa vår bläckfisk. Nästan all programmering för krabban kommer att vara likadan som den för hajen.

Först laddar vi in en ny sprite, krabban heter ”Crab” och ligger under ”djur”. Gå till krabbans script och dra in ett ”när grön flagga klickas på” och lägg till ett ”göm”, så att krabban inte syns när spelet börjar.

Nu ska vi göra scriptet som hanterar en klon av krabban. Titta på bilden så ser du hur jag programmerat och du kan göra likadant. Nedanför bilden förklarar jag skillnaderna mot hajens script.



Som du ser har jag två script som körs när krabban startas som klon. Det lilla scriptet animerar bara krabban – den kommer med två klädslar precis som bläckfisken och om vi växlar mellan dem så ser det ut som att krabban knipsar med klorna.

Det andra, längre scriptet börjar med att sätt y-värdet så att krabban ligger längst ner på skärmen, och sätter y-värdet med hjälp av slumpen så att krabban dyker upp på olika ställen längs botten. Sen pekas krabban rakt uppåt (noll graders riktning) men för att den inte ska snurra runt så sätter vi rotationsstilen till ”rotera inte” – då visas spriten rakt upp oavsett rotation. Sen visar vi krabban.

I loopen flyttar sig krabban långsamt, nämligen 0.3 steg i taget rakt upp (för vi satte ju riktningen till uppåt tidigare), och sen kommer tre villkor efter varandra. Det första kollar om krabban hunnit simma upp en bit, nämligen till ett y-läge som är mer än -160. Då vänder vi helt om på krabban så att han pekar neråt, och därför börja simma tillbaka mot botten istället.

Nästa villkor kollar om krabban hunnit simma tillbaka ner till botten av skärmen och raderar klonen när den är färdig. Det sista villkoret känner av om krabban rör vid bläckfisken, skickar ”game over”-meddelandet, som är samma som hajen skickar, och raderar klonen.

Jag valde att krabban låta röra sig väldigt långsamt för att det blir svårt för bläckfisken att hinna undan annars. Om du ändå tycker att det är för svårt när du testspelar kan du flytta upp bläckfisken på scenen lite, men tänk på att du kanske behöver flytta upp stjärnorna också då.

Glöm inte att du behöver ett script i scenen för att sätta igång krabbans kloner också. Jag satte tiden som slumpas till mellan 15 och 25 sekunder, annars kommer det för många krabbor, men testa gärna med olika värden!

Bakgrundsljud – att loopa ljud

Spelet börjar verkligen ta form, men vi har ett par saker kvar att göra innan det är helt klart. Först och främst skulle jag vilja att det låter som att vi är under havet när vi spelar. För att göra det kan vi lägga ett script i scenen som loopar ett bakgrundsljud.

Klicka på scenen och klicka sen på fliken ”Ljud”. Klicka på den lilla högtalaren för att ladda in ett nytt ljud. Ljudet vi ska ha heter ”bubbles” och ligger under kategorin ”Effekter”.

När du laddat in ljudet, ta och börja på ett nytt script genom att dra in ett ”när grön flagga klickas på”-block. Ta sen en för alltid-loop och i den lägger du ljudblocket ”spela ljudet [bubbles] tills färdigt”. Nu har du gjort en ljudloop. Om du väljer fel block här, och istället tar det block som bara heter ”spela ljudet [bubbles]” så kommer Scratch att försöka spela ditt bubbelljud om och om igen så snabbt det kan, och det låter inte så bra. Testa!

Spelet är färdigt! Hur många poäng kan du få? Jag fick som mest 23 när jag testade.

Det här spelet finns att spela på http://scratch.mit.edu/projects/21528024/.