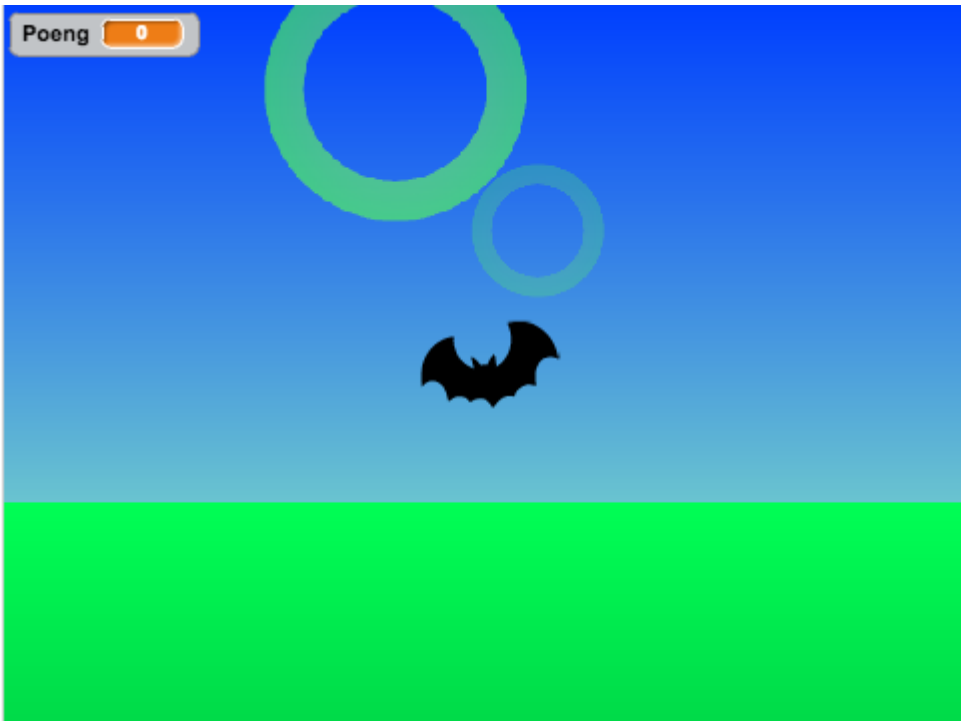
◆ 3D-Flakser, Del 2

Skrevet av: Gudbrand Tandberg og Geir Arne Hjelle

Kurs: Scratch
Tema: Blokkbasert, Spill, Animasjon
Fag: Kunst og håndverk, Matematikk, Naturfag
Klassetrinn: 5.-7. klasse, 8.-10. klasse
Språk: Norsk bokmål

Introduksjon

Velkommen til andre og siste del av **3D-Flakser**! I denne delen skal vi få Flakse til å flakse som en fugl, og snu seg i luften når vi svinger med piltastene. Til slutt skal vi gjøre det slik at man får poeng når man flyr igjennom en ring og taper hvis man treffer en ring. Etter det er det opp til deg; lag en meny, lag flere vanskelighetsgrader eller lag noe helt annet!



Steg 1: Lag bakken og få den til følge med

Vi begynner med et enkelt steg som gjør spillet litt mer realistisk. Siden figuren flyr så kan vi forvente at hvis man ikke flakser, så treffer man før eller siden bakken. Dette gjør vi med en ny figur som vi kaller `bakken`.

Sjekkliste

- [ ] Lag en ny figur som heter `bakken`. Tegn en drakt til den. Det enkleste er å bare fylle den nederste tredjeparten av tegneområdet med grønt. Vi starter med å gi den følgende skript

```
når jeg mottar [Nytt spill v]
gå til x: (0) y: (-300)
sett størrelse til (200) %
```

Nå bør bakken ligge nederst i scenen når spillet begynner. Hvis den ikke gjør det kan du endre litt på tallene.

- [ ] Nå vil vi at bakken skal følge med Flakse, det vil si: når Flakse er høyt oppe (`y` er stor) så skal bakken gå nedover, og når Flakse er langt nede så er bakken tilsvarende høyt oppe. Hvis Flakse berører bakken skal spilleren tape. Vi legger til følgende `for alltid`-løkke i skriptet til bakken,

```
for alltid
    sett y til ((20) - (y))
    hvis <berører [Flakse v]>
        si [du tapte!] i (2) sekunder
        stopp [alle v] :: control
    slutt
slutt
```

- [ ] Til slutt så vil vi helst at bakken skal forsvinne når Flakse flyr veldig høyt. Det kan vi gjøre med `skjul` og `vis` kommandoene, slik

```
hvis <(y) < [150]>
    vis
ellers
    skjul
slutt
```

Steg 2: Få Flakse til å flakse

Hvis du har gjort del en av 3D-Flakser riktig så kan du nå styre flakse-figuren gjennom ringene med piltastene. Det er to ulemper med dette: det er et veldig lett spill, og det er ikke sånn fugler flyr. Vi vil at Flakse faktisk må flakse for å holde seg i luften. Derfor endrer vi litt på skriptene til flakse-figuren slik at den flakser med vingene når vi trykker mellomromtasten.

Sjekkliste

- [ ] For å holde styr på hvor mange ganger spilleren har trykket på mellomrom så lager vi en variabel `flaks` som gjelder kun for flakse-figuren.

- [ ] Vi må slette testene som sjekker om `pil opp` eller `pil ned` tastene trykkes og erstatte dem med

```
hvis <tast [mellomrom v] trykket?>
    endre [flaks v] med (1)
    vent (0.01) sekunder
slutt
```

  Alt som skjer nå er at `flaks` økes med én hver gang mellomrom trykkes. Vi lager et nytt skript hos Flakse som tar seg av flaksingen.

- [ ] Lag en ny variabel. Kall den `løft` , og la den gjelde kun for denne figuren. Denne variabelen skal fortelle oss hvor fort flakse skal flyttes opp eller ned.

- [ ] Legg inn en `sett løft til 0` -kloss et sted før spillet starter.

- [ ] Sett inn disse klossene først i hovedløkken til Flakse:

```
endre [y v] med (løft)
hvis <(løft) > [-5]>
    endre [løft v] med (-0.5)
slutt
```

- [ ] Til slutt lager vi et nytt skript hos Flakse slik:

```
når jeg mottar [Nytt spill v]
for alltid
    gjenta til <(flaks) = [0]>
        endre [flaks v] med (-1)
        hvis <(løft) < [5]>
            endre [løft v] med (2)
        slutt
        hvis <(løft) < [0]>
            sett [løft v] til [0]
        slutt
    slutt
slutt
```

Nå kan du justere litt på tallene i skriptene over for at Flakse flyr slik DU vil!

Steg 3: Få Flakse til å snu seg i luften

For at spillet skal se best mulig ut så vil vi at flakse skal rotere i luften når vi holder piltastene inne. Litt som et fly som går inn for landing. Oppførselen vi ønsker når (for eksempel) høyre piltast trykkes er denne: når piltasten først trykkes skal figuren peke mot høyre, og der skal den holde seg så lenge piltasten holdes inne. Når piltast slippes skal figuren rotere sakte tilbake til sin vanlige posisjon.

Dette får vi til ved å endre litt på hovedskriptet til flakse.

Sjekkliste

- [ ] Legg til klossene `pek i retning 135` og `pek i retning 45` i testene som sjekker om henholdsvis høyre og venstre piltast trykkes. Prøv spillet. Peker figuren i riktig retning?

- [ ] Nå vil vi at figuren skal rotere tilbake til vannrett når piltastene ikke trykkes lenger. Det kan vi enkelt få til ved å legge til disse klossene under testene som sjekker om piltastene trykkes.

```
hvis <(retning) < [90]>
    vend høyre (1) grader
slutt
hvis <(retning) > [90]>
    vend venstre (1) grader
slutt
```

Disse klossene sørger enkelt og greit for at figuren alltid prøver å peke mot retning 90 (som er vannrett for figuren). Når bør flyvningen til Flakse se ganske bra ut!

Steg 4: Sjekk om Flakse treffer ringene

Nå ønsker vi at Flakse skal få poeng hver gang han flyr igjennom ringene, og taper om han treffer en ring. Denne oppførselen skal vi kode i `når jeg starter som klon` -skriptet til ring-figuren.

Sjekkliste

- [ ] Vi begynner med å sjekke om Flakse berører den ringen som ligger nærmest. Husk at det er `distanse` variablen til ringen som forteller oss hvor nærme ringen er. Så vi må hele tiden sjekke om Flakse berører ringen, *og* om ringen faktisk er nærme. Du må legge til denne klossen et sted i skriptet til ring-klonene.

```
hvis <<berører [Flakse v]> og <(distanse) < [1.2]>>
    si [du tapte!] i (1) sekunder
    stopp [alle v] :: control
slutt
```

Nå stopper spillet om man treffer en ring som er nærme. Hvis du vil kan du kode noe annen oppførsel når man treffer en ring (kanskje man bare mister et liv, eller det spilles en lyd?)

- [ ] Så ønsker vi å få poeng når vi fly gjennom ringene. Lag først en variabel `poeng` som gjelder for alle figurene. Nå må vi sjekke at variablene `x` og `y`, som sier hvor flakse-figuren er, ikke er altfor langt unna `ringX` og `ringY`. Vi legger til følgende klosser rett under hovedløkken til ringene, slik at det siste ringene gjør før de slettes er å sjekke om Flakse er inni.

```
hvis < <(x) < ((ringX) + (160))> og <(x) > ((ringX) – (160))> >
    hvis < <(y) < ((ringY) + (160))> og <(y) > ((ringY) – (160))> >
        endre [poeng v] med (1)
    slutt
slutt
```

Fungerer skriptet som det skal? Hva er det vi egentlig sjekker i den siste `hvis` -testen?

Nå er vi igrunn ferdig med det viktigste i spillet. Men det er fremdeles masse spennende igjen du kan prøve:

Ting å prøve

- [ ] Lag en meny.
- [ ] Få ringene til å komme fortere mot deg etterhvert som du får flere poeng.
- [ ] Gi Flakse flere drakter å det ser ut som han flyr når han skifter drakt.