Innhold

Innhold 1
Felix friker ut 2
Introduksjon: 2
Uke 1: Byggeblokker 2
Oppgave 1: Kollisjonsdetektorer 2
Oppgave 2: Bevegelse, gulv og fall 4
Oppgave 3: Hopp 6
Oppgave 4: Nøkler og mål 7
Steg 5: Skurker og dødelige omgivelser 8
Oppsummering 10
Uke 2: Nivåer 10
Tilleggsoppgave: Tramping 10
Tilleggsoppgave: Kraftpiller 11
Uke 3: Sett sammen spillet 11
Oppgave 1: Å vise et nytt nivå 11
Oppgave 2: Spill! 13
Tilleggsoppgave: Flere liv 13
Tilleggsoppgave: Tidsbegrensinger 14

Felix friker ut

Introduksjon:

I dette prosjektet skal du bygge et komplekst plattformspill, der katten Felix hopper rundt, unnviker skurker og samler nøkler for å slippe ut av hulen. Når han har gjort det, kommer han til neste hule der han gjør det samme om igjen.

I prosjektets første uke vil du lære hvordan du kan få Felix til å flytte seg rundt og interact med forskjellige ting. I andre uke skal du designe dine egne nivå. I den tredje uken skal du sette nivåene sammen til et komplekst spill.

Uke 1: Byggeblokker

Plattformspill, som **Manic Miner** og **Mario Bros**, handler alle om at hovedpersonen beveger seg gjennom en verden og støter på ting. Noen ting, som vegger og gulv, bare stopper deg fra å bevege deg. Noen ting, som skurker, dreper deg. Noen ting, som nøkler, ønsker du å samle på. Andre ting er kun omgivelser og påvirker ikke spillet i det hele tatt.

Dette betyr at når du treffer på kolliderer med noe (kalt **kollisjonsdeteksjon**), er det viktig. Scratch gir deg noen blokker for kollisjonsdeteksjon:
berører [ ]?, berører fargen [ ]?, farge [ ] berører [ ]?, og avstand til [ ]. Men for å utvikle spillet trenger du å vite mer enn om du berører noe, du trenger å vite hvilken **side** som berører. Hvis du beveger deg langs en vegg til venstre, kan du ikke lenger gå mot venstre. Men du kan fortsatt gå mot høyre, eller hoppe, eller falle hvis det ikke er noe gulv.

Å berøre en skurk kan skade deg, men det kan hende du skader ham om om du berører ham med føttene dine. Ingen av de innebygde Scratch-blokkene forteller deg om **retning for kontakt**, så vi må bygge våre egne kollisjonsdetektorer.

Sjekkliste Følg instruksjonene på lista. Huk av etter hvert.

Test Klikk på det grønne flagget for å teste koden.

Lagre Husk å lagre koden når du har lagt til noe nytt.

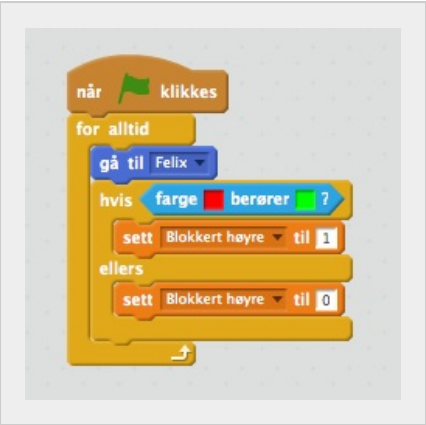
Oppgave 1: Kollisjonsdetektorer

Du vil ha fire figurer som følger Felix. Hver vil detektere kollisjoner i en retning og sette en variabel hvis den oppdager en kollisjon. Felix-figuren vil bruke disse variablene til å kontrollere hvordan

Felix kan bevege seg. Hver kollisjonsdetektor vil ha en fargelinje og bruke farge [] berører []?-blokken til å detektere ting. Vi vil bruke **sort** for gulvet og **grønt** for hindringer. Vi vil bruke **rødt** for kollisjonsdetektorene.

Sjekkliste

1. **Åpne et nytt prosjekt.** Legg til bakgrunnen frantic-felix/level1 og slett den hvite background1.
2. Gi Sprite1 navnet *Felix*. Pass på at drakten hans bare kan snu seg mot høyre eller venstre.
3. Lag fire nye figurer fra bildene *frantic-felix/top*, *frantic-felix/bottom*, *frantic-felix/left* og *frantic-felix/right*. Disse figurene er kollisjonsdetektorene. Kall de nye figurene *Topp*, *Bunn*, *Venstre* og *Høyre*. Pass på at draktene deres ikke kan rotere eller snu seg.
4. Lag fire **variabler**, for alle figurene: *blokkert topp*, *blokkert bunn*, *blokkert høyre* og *blokkert venstre*.
5. Gi hver av kollisjonsdetektor-figurene dette skriptet:



6. Endre variablene for hver detektor. Bunndetektoren trenger en [ ] eller [ ]-blokk slik at den setter *blokkert bunn* hvis den berører grønn eller sort.

Hint: Det er enklere å først velge fargene i en figur, og deretter dra skriptet til de andre figurene og endre variablene som blir satt. Da slipper du å måtte finne de rette fargene fire ganger.

I starten trenger Felix bare et skript for å følge muspekeren for alltid. Se skriptet over hvis du trenger et hint.

Test prosjektet

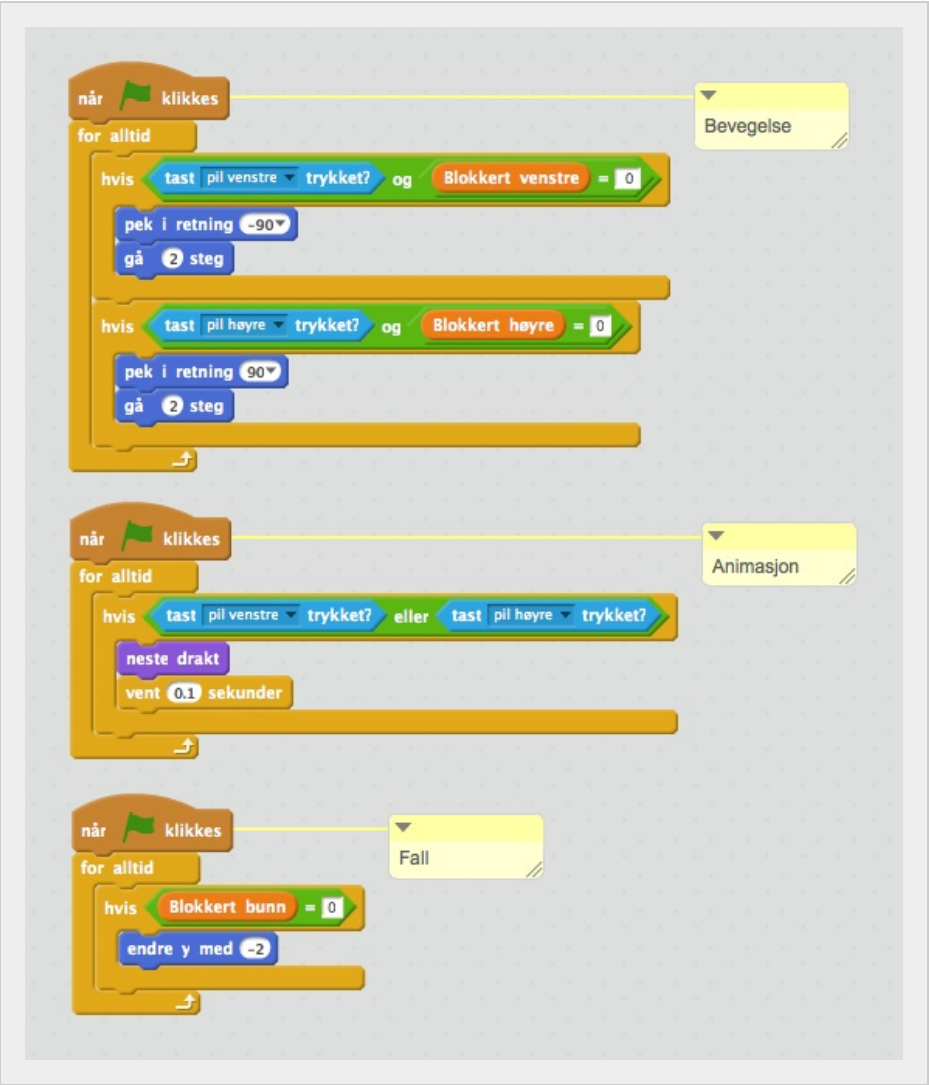
7. **Klikk på det grønne flagget.** Du bør se Felix følge muspekeren rundt, omringet av et rødt rektangel. Rektangelet er kollisjonsdetektoren. Hvis du ser på variablene, bør du se dem endre seg når du drar Felix rundt og han breører ulike deler av skjermen. For øyeblikket beveger Felix seg gjennom plattformene og de grønne hindringene. Vi vil fikse på dette i de neste stegene.

Lagre prosjektet

Oppgave 2: Bevegelse, gulv og fall

Sjekkliste

1. **Nå skal vi få Felix til å bevege seg.** Vi skal bruke **venstre** og **høyre piltast** for å bevege ham mot venstre og høyre. Han skal falle hvis han ikke står på fast grunn. Hvis man trykker på venstre piltast, ønsker vi at Felix skal snu seg mot venstre og flytte seg litt mot venstre. Men vi vil ikke at han skal flytte seg mot venstre hvis det er en hindring der.
2. Vi kunne brukt når [ ] trykkes-hatter for å bevege Felix, men det gir en hakkete bevegelse. Ting beveger seg jevnere hvis du putter hvis tast [pil venstre] trykket? -blokker i for alltid-løkker. Kollisjonssdetektering gjør at vi må inkludere testen av **blokkert venstre**-variablen i hvis-blokken, ved å bruke en [ ] og [ ]-blokk slik at Felix kun beveger seg til venstre hvis venstretasten trykkes inn og **blokkert venstre** er **null**. Og vi må gjøre det samme for bevegelser mot høyre.
3. Vi kunne animert beina til Felix i den samme blokken, men de vil da bevege seg for fort. Lag animasjonen i en egen for alltid-blokk under et annet **når grønt flagg klikkes**-hatt.
4. Det siste vi skal gjøre er **fall**. Vi vil at Felix skal falle hvis det ikke er noen er noen fast grunn under ham. Det blir en egen for alltid-løkke under en annen **når grønt flagg klikkes**-hatt.



Test prosjektet

5. Dra Felix med musen til et sted på scenen og trykk deretter på det grønne flagget. Hvis Felix fortsetter å følge muspekeren, må du fjerne det skriptet! Du skal kunne bruke venstre- og høyretastene for å få Felix til å gå fra side til side. Hvis han ikke ikke står på gulvet, skal han falle forsiktig.

6. Vi vil også gjemme kollisjonsdetektorene. Vi kan ikke bruke en skjul-blokk, for de vil ikke detektere noen kollisjoner. Sett i stedet en sett [gjennomsiktig] effekt til 100 rett under **når grønt flagg klikkes**-hatten i hver kollosjonsdetektor. Det gjør figuren usynlig uten å gjemme den.

Test prosjektet

7. Når du klikker på det grønne flagget, skal kollisjonsdetektorene forsvinne. De vil vises

igjen når du klikker på det røde stoppknappen.

Oppgave 3: Hopp

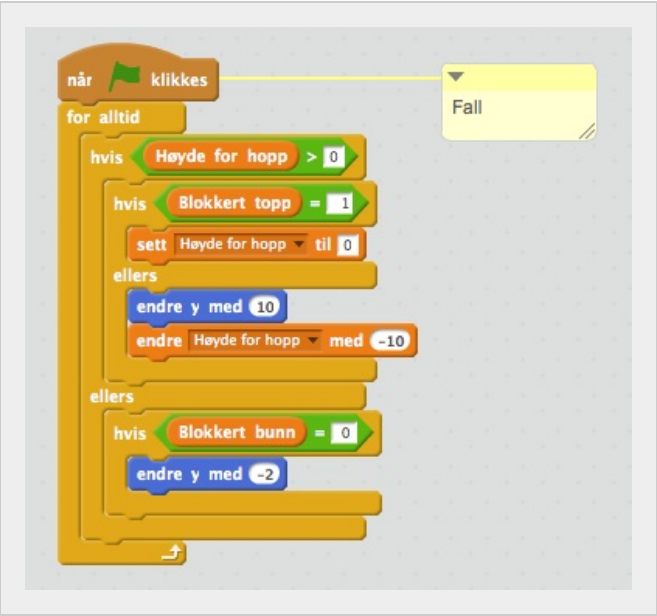
Sjekkliste

1. Siste bevegelse vi skal utvikle for Felix er hopp. La oss bruke mellomromstasten for å få Felix til å hoppe.

Noen ting vi må tenke på når det gjelder hopp:

- Vi vil ikke at Felix skal **falle** mens han er på vei **oppover**.
- Vi vil ikke at Felix skal stoppes av gulv på vei oppover, men vi vil at han skal stoppes av **gulv** når han kommer ned igjen.
- Vi vil ikke at Felix skal **hoppe** inn i en **grønn hindring**.
- Vi vil at hoppet skal være **animert**, så vi vil ikke at han skal flytte seg for langt i ett steg.
- Felix skal bare kunne hoppe når han **står på et gulv**. Spillet ville bli for enkelt hvis han kunne hoppe videre i luften.

2. Vi skal kontrollere hopp med en ny variabel, **høyde for hopp**. Hvis denne er større enn **null**, går Felix oppover. Hvis den er null, faller Felix (eller han har falt).
3. Vi skal få Felix til å hoppe opp **100 piksler**. Legg til en ny hvis-blokk inni for alltid-blokken som håndterer tastetrykk. Hvis vi trykker **mellomromstasten** og Felix står på et gulv (**blokkert bunn** er satt til **1**), sett **høyde for hopp til 100**.
4. Vi vil endre skriptet for å falle. Inni for alltid-blokken vil vi ha en hvis.. ellers..-blokk som detekterer om Felix hopper eller ikke. Betingelsen for denne hvis.. ellers..-blokken er om **høyde for hopp** er **større enn null**. Den eksisterende hvis-blokken for fall legges i ellers-delen av den nye hvis.. ellers..-blokken.
5. Når vi vet at Felix hopper oppover, må vi sjekke om han kræsjer hodet i noe. Hvis **blokkert topp** er **1**, sett **høyde for hopp** til **0**, så han ikke hopper videre inn i hindringer. **Ellers, beveg Felix opp med 10 og reduser høyde for hopp med 10.**
6. Du skal ende opp med dette (den nederste delen er den eksisterende hvis-blokken):



Test prosjektet

7. **Klikk på det grønne flagget.** Hopper Felix? Hopper han fra en plattform til en annen? Faller han fortsatt hvis han går utfor kanten på en plattform? Hva om han hopper fra kanten av en plattform? Hva om han prøver å hoppe under den grønne hindringen til høyre? Hva skjer om du trykker på mellomromstasten mens Felix faller?

Lagre prosjektet

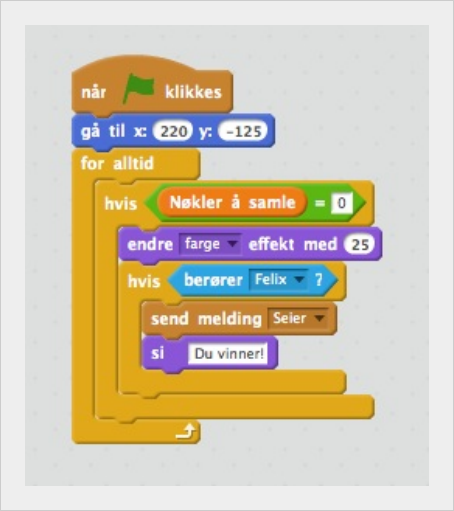
Oppgave 4: Nøkler og mål

Vi har fått Felix til å bevege seg rundt i spillet. Nå skal vi få han til å gjennomføre et nivå.

Vi vil nå legge ut tre nøkler som Felix kan samle ved å berøre dem. Når han har samlet alle tre, kan han klatre inn i en rømningskapsel og flykte.

1. Vi vil ha en ny variabel, **nøkler å samle**, som holder greie på hvor mange nøkler han mangler. Lag et nytt skript for scenen som setter denne til 3 når man trykker på **det grønne flagget**.
2. Nøklene og rømningskapselen skal begge være figurer. (Bruk frantic-felix/key1 for nøklene og frantic-felix/escape-pod for rømningskapselen.)
3. Hver nøkkel trenger **to skript**: et plasserer nøklene på rett posisjon (sjekk andre implementasjoner!), med størrelse og vinkel, og har deretter en for alltid-løkke for å endre fargen (som gjør det enklere å se den på skjermen). Det andre skritpet er en for alltid hvis-løkke som venter til Felix berører nøkkelen. Så fort det skjer, gjemmer skriptet nøkkelen og reduserer **nøkler å samle**.
4. Rømningskapselen har et noe mer komplisert skript. Den bruker en for alltid hvis-

blokk for å vente på at **nøkler å samle** skal bli **null**. Så fort det skjer, skal rømningskapselen begynne å blinke, for å vise spilleren at de kan flykte. Da kan vi bruke en ny hvis til å detektere når Felix berører den blinkende rømningskapselen. Når han gjør det skal rømningskapselen sende melding om seier. Når Felix får seiersmeldingen skal han skjules.



Test prosjektet

5. **Trykk på det grønne flagget.**

Lagre prosjektet

Steg 5: Skurker og dødelige omgivelser

Endelig, skurker!

Vi skal ha to farlige elementer i spillet. Skurker vil bevege seg rundt og skade Felix hvis han kommer borti dem. I tillegg vil vi plassere farlige ting i omgivelsene.

Først lager vi en skurk. Den skal bevege seg langs en fast bane.

Sjekkliste

1. Lag en ny figur, med hvilken drakt du vil. Den bør være omtrent samme størrelse som Felix (velg gjerne en fra biblioteket og gjør den mindre). Skurken trenger kun ett skript som både beveger den og detekterer når den berører Felix.

2. Putt tre hvis-blokker inni en for alltid-løkke. Den første hvis-en sjekker om skurken berører Felix. Gjør han det, skal den sende melding om tap. De andre to hvis-ene sjekker om skurken har kommet til enden av banen han beveger seg på; hvis den har det, skal den snu. Til slutt skal skurken ta to steg. Ved å bruke gå i stedet for gli-blokkene blir det lettere å kontrollere hvor fort skurken beveger seg.
Vi trenger ikke bruke kollisjonsdetektor-figurer her, siden vi ikke bryr oss om hvilken side av skurken Felix berører.

3. Legg til skript for Felix og rømningskapselen for å respondere på meldingen om tap. Felix skal gjemme seg, mens rømningskapselen skal si **[Du taper!]**.

Test prosjektet

4. **Klikk på det grønne flagget**. Beveger skurken seg? Stopper den og snur ved endene av banen sin? Hva skjer om Felix går inn i skurken? Hva skjer om Felix hopper inn i den, ovenfra eller nedenfra? Forsvinner Felix? Sier rømningskapselen riktig ting? Kan du fortsatt vinne spillet?

Lagre prosjektet

5. **Nå skal vi lage farlige omgivelser!** La oss si at alt som er lyseblått er dødelig for Felix. Last bakgrunnen frantic-felix/level2, som har en blå rose på øverste nivå. Legg til et nytt skript for Felix, under en **grønt flagg**-hatt: for alltid hvis berører fargen [blå] sende melding [tap].

Test prosjektet

6. **Klikk på det grønne flagget.** Dør Felix hvis han tar på den blå rosen? Hva om han tar på andre ting?

Lagre prosjektet

Oppsummering

Du har nå bygget et veldig enkelt plattformspill! Enn så lenge er det et ganske kjedelig spill, men det er et godt grunnlag for å bygge dine egne nivåer i et plattformspill. Neste uke skal du lage dine egne nivåer.

Uke 2: Nivåer

Forrige uke bygget du alle deler av et plattformspill. Denne uken skal du bruke disse delene til å lage dine egne nivåer.

For å oppsummere hva du har gjort:

- Felix kan gå til høyre og venstre og hoppe.
- Felix faller hvis han ikke står på et sort gulv.
- Felix kan ikke bevege seg gjennom grønne hindringer.
- Felix dør om han berører blå biter i omgivelsene eller skurker.
- Skurker beveger seg langs faste baner.
- Felix kan samle nøkler ved å berøre dem.
- Når Felix har alle nøklene, vil en rømningskapsel lyse opp og få ham i sikkerhet (eller til neste nivå).

1. Du kan bruke disse delene som en verktøykasse. **Bruk dem til å bygge dine egne nivåer.**
   Du skal nå lage et sett med nivåer som Felix skal gjennom, og neste uke skal du se hvordan du kan koble sammen nivåene.
2. Nivåer kan ha store eller små plattformer, og de kan ha mange eller få plattformer. Det kan være mange skurker, eller ingen. Det kan være mange eller få hindringer og dødelige ting i omgivelsene. Ikke lag nivåene for begrenset; det bør være flere veier til mål, selv om noen er enklere enn andre. Tenk på hvor lett eller vanskelig et nivå er.
3. Du kan endre spesialfargene (sort, grønn og blå), men du må da **oppdatere fargeberørings-blokkene** i alle skriptene, og du må beholde de samme fargene gjennom alle nivåene. (Du kan også ha forskjellige farger i forskjellige nivåer, men da må du puttet mange [ ] eller [ ]-blokker rundt fargeberørings-blokkene.)
4. **Test nivåene dine.** Hvis du får tid, implementer nivåene i Scratch og spill dem. Sjekk at de **ikke er for lette** eller **for vanskelige**. Hvis du implementerer nivåene, pass på å **lagre bakgrunnene du lager** og noter **startposisjonene** til Felix, nøkleme og skurker. Pass også på hvilken **retning** skurkene kommer inn og **hvor langt de går**.

Hvis du allerede har laget noen nivåer og implementert dem i Scratch, **prøv disse tilleggsoppgavene:**

Tilleggsoppgave: Tramping

5. **Hvorfor ikke la skurkene dø hvis de blir trampet på?** Kanskje du kan legge til et skript som gjør noe hvis skurken berører bunnkollisjonsdetektor-figuren?

Tilleggsoppgave: Kraftpiller

6. **Du kan lage en "kraftpille" som lar Felix ødelegge skurker.** Når Felix får tak i kraftpillen, kan han i en begrenset periode ødelegge skurker ved å berøre dem. Pillene slutter å virke etter en stund.
   Du kan selv få finne ut hvordan du kan få det til å fungere. Du kan gjerne få skurkene til å se annerledes ut mens Felix kan ødelegge dem.

Test prosjektet

7. Klikk på det grønne flagget.

Lagre prosjektet

Uke 3: Sett sammen spillet

Du har nå noen nivåer, og nok verktøy til å få hvert nivå til å fungere. Nå trenger du bare å sette dem sammen så det blir til et helt spill!
Hvis du klarer å sette sammen noen nivåer fort, kan du gjerne se på tilleggsoppgave før du begynner å spille spillet.

Oppgave 1: Å vise et nytt nivå

1. **Når Felix fullfører et nivå, må vi bevege oss til neste nivå.** Det betyr at rømningskapselen må sende en **"start nivå"-melding** (i stedet for seiersmeldingen) når Felix får alle nøklene og går inn i rømningskapselen. Vi kan bruke "start nivå"-meldingen for å sette opp neste nivå. Vi trenger også en %(styring)variabel%*, aktivt nivå, som blir satt av rømningskapselen før den sender "start nivå"-meldingen.
   Så og si alt må respondere til "start nivå"-meldingen for å sette opp neste nivå.

2. Scenen må vise riktig bakgrunn. Skurkene, nøklene og rømningskapselen må alle flytte seg til riktig posisjon. Skurkenes bevegelsesbane må oppdateres. Felix må flytte seg til sin nye startposisjon. Deretter må nivået starte.

3. De fleste figurene må respondere til "start nivå"-meldingen i stedet for det grønne flagget. Det betyr at du må endre hatt-blokker for de fleste skriptene.

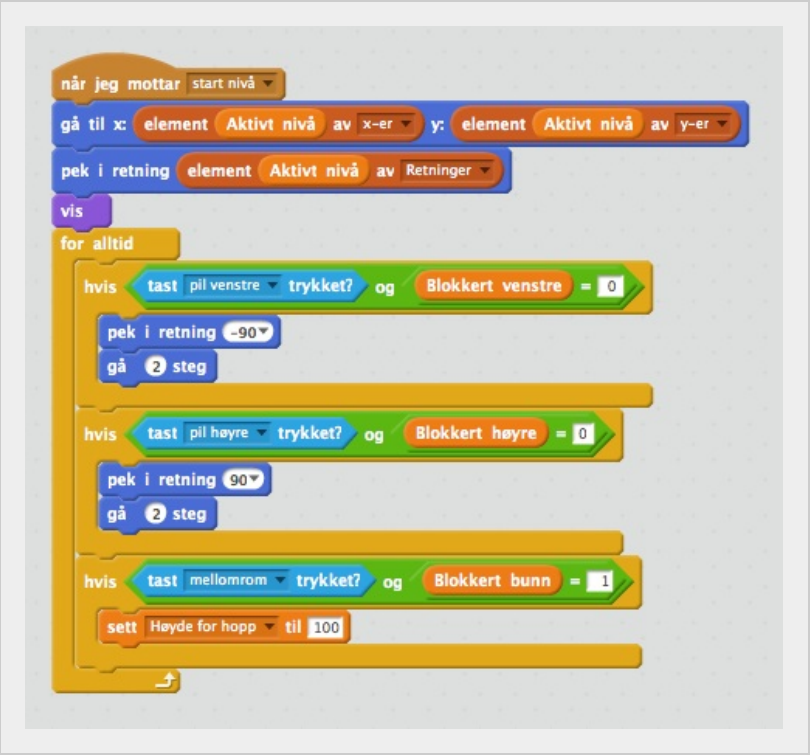
Lagre prosjektet

Før du gjør noe, lagre arbeidet ditt. Gjennom spillet vil du hele tiden gjøre endringer, så fortsett å lagre arbeidet regelmessig.

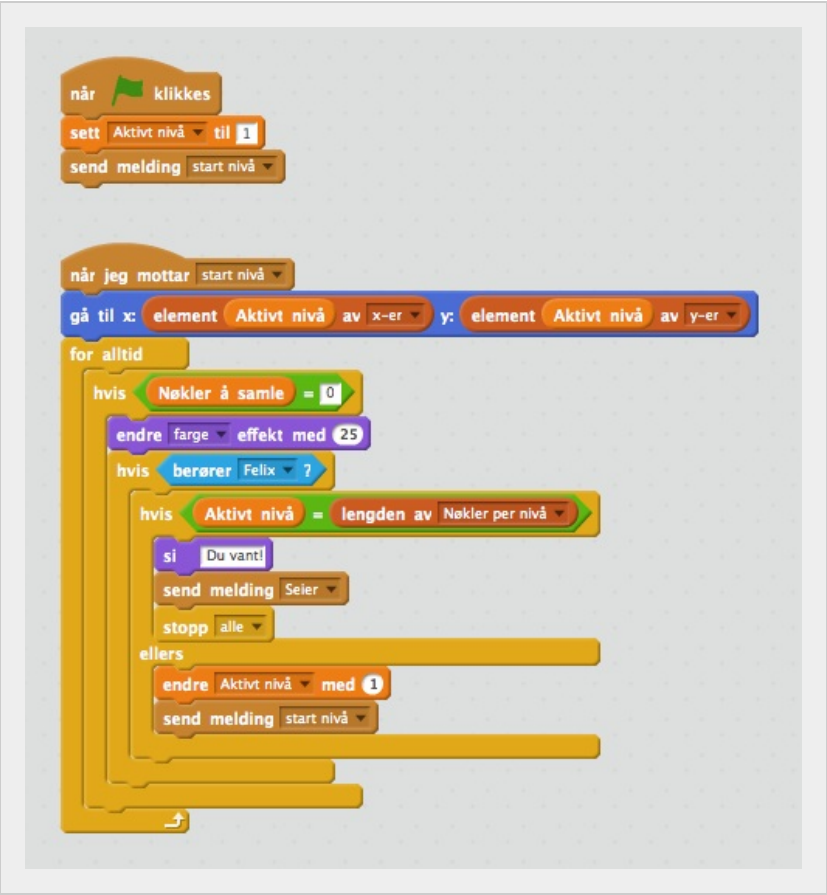
Test prosjektet

4. Fortsett å teste spillet etter hver endring. Test også delene du allerede har endret, for å forsikre deg om at de fortsatt virker.

Vi vil ikke si hvordan du gjør alle endringene. Men vi kan vise deg hvordan vi har endret Felix sitt skript så du ser hva slags ting du må gjøre:



Startverdiene for x, y og retning for Felix er satt med lister. Vi har laget noen lister for hver figur (hver liste er privat for den figuren) for å lagre alle verdiene vi trenger for den figuren. Du trenger en liste for hver ting du lagrer. Du må ikke bruke lister, du kan bruke hvis-blokker som sjekker nåværende nivå og gjør riktig ting avhengig av verdien. Og her er rømningskapselen, som håndterer alle nivåbytter:



Oppgave 2: Spill!

Nå har du et spill, og dine medelever har spill. Spill alle spillene. Klarer du å gjennomføre deres spill, og klarer de ditt?

Tilleggsoppgave: Flere liv

1. Vi kan gi Felix flere liv for å hjelpe han å komme seg gjennom alle nivåene. Lag en ny figur med 3 drakter, som viser 1, 2 og 3 hjerter. Bruk draktene *frantic-felix/1-heart* , *frantic-felix/2-heart* og *frantic-felix/3-heart* . Få hver figur til å vises i hjørnet av scenen. Når det grønne flagget klikkes, skal det vises 3 hjerter. Hver gang den mottar en melding om tap, skal den vise ett hjerte mindre. I stedet for å vise null hjerter, skal den gjemme seg og vise en beskjed om at spillet er slutt.

2. Beskjedene om seier eller at spillet er slutt skal håndteres av en ny figur, som skjules når de grønne flagget trykkes på, og viser riktig beskjed når meldingene om seier eller at spillet er slutt sendes. Figuren skal også stoppe all skript når den viser seg selv.

3. Du må også endre på hvordan rømningskapselen håndterer seier og tap, siden det nå er antall liv og sluttbeskjed-figurene som er ansvarlige for dette.

4. Det kan hende du finner ut at skurkene er litt for kjappe, og tar Felix før nivået starter skikkelig. Hvis Felix mister mer enn ett liv når han går inn i en skurk, skjul skurken så fort den sender meldingen om tap. Det gir resten av spillet (inkludert Felix) tid til å

tilbakestille seg før skurken registrerer en ny kollisjon med Felix.

Tilleggsoppgave: Tidsbegrensinger

1. La Felix få dårlig tid! Bruk Timer Scratch card for å legge til en tidsfrist. Hvis tiden løper ut, send melding om tap. Husk å nullstille tidtakeren på starten av hvert nivå.