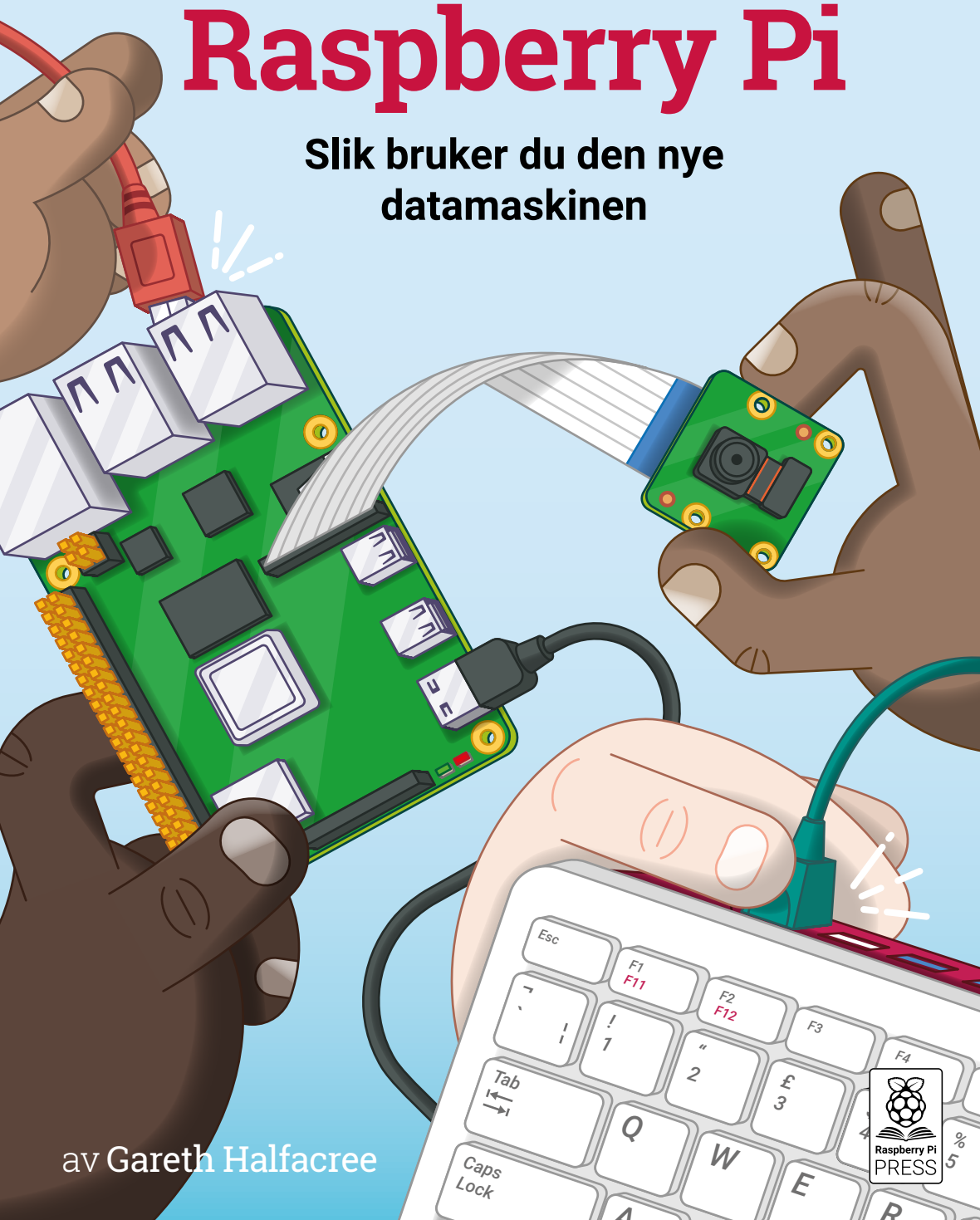
4. utgave

FULLSTENDIG OPPDATERT FOR **RASPBERRY PI 400**

DEN OFFISIELLE begynnerveiledningen for Raspberry Pi

Slik bruker du den nye datamaskinen



av Gareth Halfacree

DEN OFFISIELLE begynnervei-ledningen for Raspberry Pi

Slik bruker du den nye datamaskinen

Først publisert i 2021 av Raspberry Pi Trading Ltd, Maurice Wilkes Building, St.John's Innovation Park, Cowley Road, Cambridge, CB4 0DS, Storbritannia

Forlagsdirektør: Russell Barnes • Redaktør: Phil King
Design: Critical Media • Illustrasjoner: Sam Alder
ADMINISTRERENDE DIREKTØR: Eben Upton

ISBN: 978-1-912047-82-6

Velkommen til den offisielle begynnerveiledningen for Raspberry Pi

Vi tror du kommer til å like Raspberry Pi. Uansett hvilken modell du har – et standard Raspberry Pi-kort eller den nye Raspberry Pi 400 med integrert tastatur – denne rimelige datamaskinen kan brukes til å lære koding, bygge roboter og lage alle slags underlige og fantastiske prosjekter.

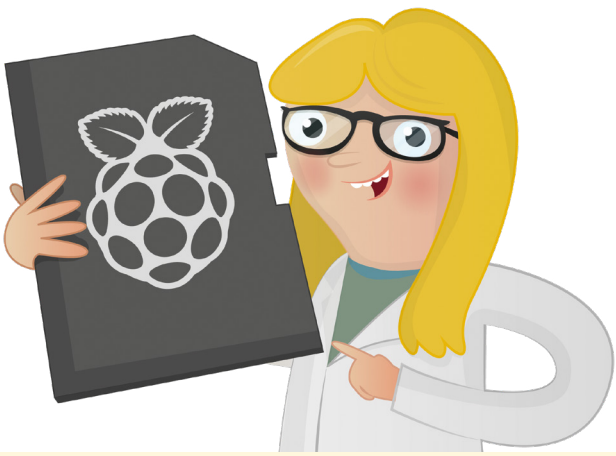
Raspberry Pi kan gjøre alt du forventer av en datamaskin – alt fra å surfe på nettet og spille dataspill til å se filmer og lytte til musikk. Men Raspberry Pi er mye mer enn en moderne datamaskin.

Med en Raspberry Pi har du tilgang til selve kjernen i en datamaskin. Du kan konfigurere ditt eget operativsystem og koble ledninger og kretser direkte til GPIO-pinnene. Den er utviklet for å lære unge å programmere på språk som Scratch og Python. Alle de vanligste programmeringsspråkene er inkludert i det offisielle operativsystemet.

Programmerere er viktigere enn noen gang, og Raspberry Pi har vekket økt interesse i informatikk og teknologi hos nyere generasjoner.

Folk i alle aldre bruker Raspberry Pi til å lage spennende prosjekter, enten det er gammeldagse spillkonsoller eller Internett-baserte værstasjoner.

Så hvis du vil lage spill, bygge roboter eller hacke en rekke fantastiske prosjekter, kan denne veiledningen hjelpe deg med å komme i gang.



Om forfatteren

Gareth Halfacree er frilans teknologijournalist, forfatter og tidligere systemadministrator i utdanningssektoren. Han har alltid vært lidenskapelig opptatt av program- og maskinvare med åpen kildekode, så han ble straks imponert av Raspberry Pi-plattformen og har skrevet flere publikasjoner om dens evner og fleksibilitet. Du kan finne ut mer om Gareth på Twitter under **@ghalfacree** eller ved å se nettsidene hans på **freelance.halfacree.co.uk**.

Innhold

VEDLEGG

Kapittel 1

Bli kjent med Raspberry Pi

Gjør deg bedre kjent med den nye kredittortstore datamaskinen din med en innføring i Raspberry Pi. Oppdag alle komponentene og finn ut hva de gjør

Raspberry Pi er utrolig artig: en fullt funksjonell datamaskin i en liten og rimelig pakke. Enten du er interessert i en enhet du kan bruke når du surfer på nettet eller spiller dataspill, vil lære å skrive dine egne programmer eller ønsker å lage dine egne kretser og fysiske enheter, vil Raspberry Pi – og produktets fantastiske fellesskap – støtte deg hvert trinn på veien.

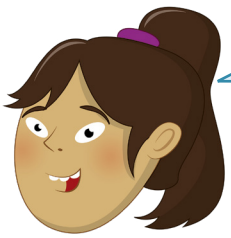
Raspberry Pi er kjent som en *ettkortsdatamaskin*, og det er akkurat det den er: en datamaskin, som en stasjonær eller bærbar datamaskin eller smarttelefon, men bygget på ett enkelt *trykt kretskort*. Som de fleste ettkortsdatamaskiner er Raspberry Pi liten – omtrent på størrelse med et kredittkort – men det betyr ikke at den ikke er kraftig. En Raspberry Pi kan gjøre det samme som en større og mer strømkrevende datamaskin, men ikke nødvendigvis like raskt.

Raspberry Pi-serien ble utviklet for å oppmuntre til mer praktisk dataundervisning rundt om i verden. Utviklerne, som gikk sammen for å danne den ideelle organisasjonen Raspberry Pi Foundation, hadde ingen idé om at de skulle bli så populære. De få tusen maskinene som ble bygd i 2012 for å teste markedet, ble utsolgt med én gang. Deretter har de solgt millioner av enheter over hele verden. Disse kretskortene har funnet veien til hjem, klasserom, kontorer, datasentre, fabrikker og til og med selvstyrende båter og romfartsballonger.

Det er utgitt forskjellige modeller av Raspberry Pi siden den opprinnelige Model B, og hver modell har enten forbedrede spesifikasjoner eller funksjoner for bestemte formål. Serien Raspberry Pi Zero er for eksempel en mye mindre versjon av fullstørrelsesversjonen – med litt færre funksjoner. Den mangler de mange USB-portene og den kablede nettverksporten, men har til gjengjeld et betydelig mindre oppsett og redusert strømbehov.

Alle Raspberry Pi-modeller har imidlertid én ting til felles. De er *kompatible*, noe som betyr at programvare som er skrevet for én modell, også kan kjøres på alle de andre modellene. Det er dessuten mulig å kjøre den nyeste versjonen av Raspberry Pi-operativsystemet på en original, forhåndslansert prototype av Model B. Det vil riktignok gå saktere, men det vil fungere.

I denne veiledningen lærer du mer om Raspberry Pi 4 Model B og Raspberry Pi 400, de nyeste og kraftigste versjonene av Raspberry Pi. Det du lærer, gjelder imidlertid også andre modeller i Raspberry Pi-serien, så det gjør ikke noe om du bruker en annen versjon.

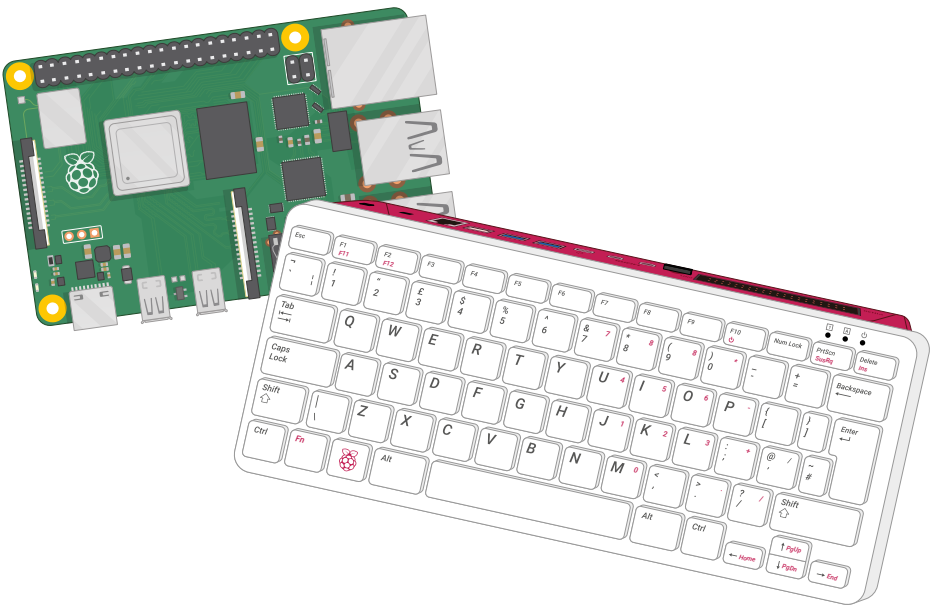


RASPBERRY PI 400

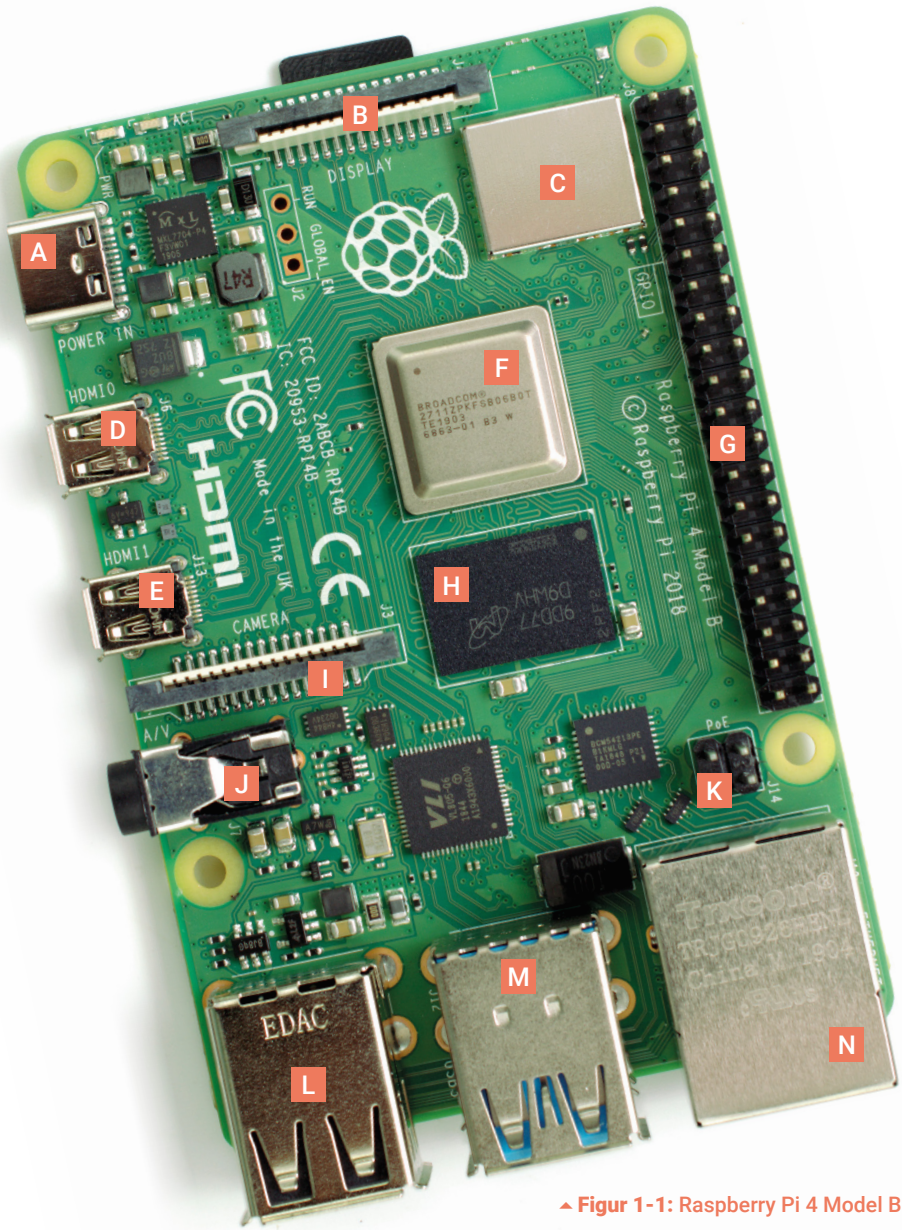
Hvis du har en Raspberry Pi 400, er kretskortet innebygd i tastaturetuiet. Les videre for å lære om alle komponentene som får Raspberry Pi til å fungere, eller hopp til side 20 for en innføring i enheten.

En innføring i Raspberry Pi

I motsetning til en tradisjonell datamaskin, der komponentene er skjult i et kabinett, kan du se alle komponentene, portene og funksjonene i en standard Raspberry Pi – selv om du kan velge å kjøpe et deksel for ekstra beskyttelse. Derfor er maskinen et flott verktøy for å lære hva de forskjellige delene av en datamaskin gjør. Det blir også enklere å se hvor ting skal være når du kobler til annet tilbehør – såkalte *eksterne enheter* – du trenger for å komme i gang.



- A Strøminngang: USB Type-C
- B DSI-skjermport
- C Trådløs/Bluetooth
- D Mikro-HDMI 0
- E Mikro-HDMI 1
- F System-on-chip
- G GPIO
- H RAM
- I CSI-kameraport
- J 3,5 mm AV
- K PoE
- L USB 2.0
- M USB 3.0
- N Ethernet-port



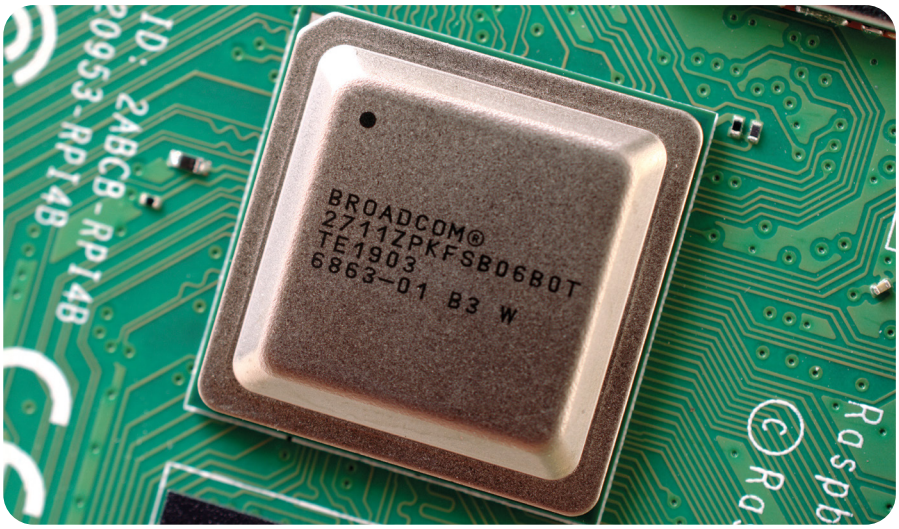
▲ **Figur 1-1:** Raspberry Pi 4 Model B

Figur 1-1 viser en Raspberry Pi 4 Model B sett ovenfra. Når du bruker en Raspberry Pi med denne veiledningen, kan det være lurt å la den ligge samme vei som på bildet. Hvis den ligger feil vei, kan det bli forvirrende når det gjelder bruk av ting som GPIO-pinneholdet (beskrevet i **Kapittel 6: Fysisk databehandling med Scratch og Python**).

Selv om det kan se ut som om det er mye pakket inn i det lille kortet, er en Raspberry Pi veldig enkel å forstå – først og fremst *komponentene*, de indre delene som får enheten til å fungere.

Komponentene til Raspberry Pi

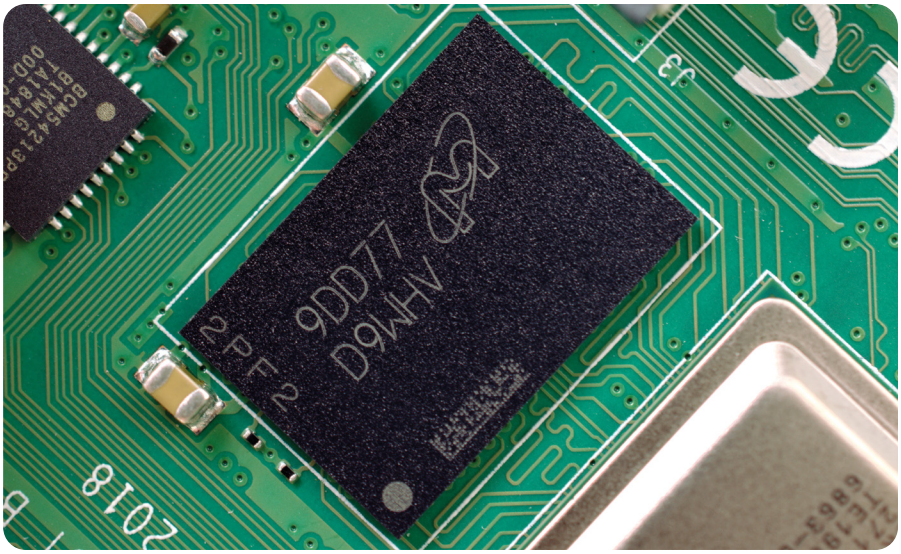
Som andre datamaskiner består Raspberry Pi av forskjellige komponenter. De har hver sin egen rolle for at alt skal fungere. Den første, og absolutt viktigste, av disse komponentene finner du omtrent midt på den øvre delen av kortet (**Figur 1-2**). Den er dekket av en metallhette og kalles *System-on-chip* (SoC), det vil si en brikke som består av flere komponenter som til sammen utgjør et komplett system.



▲ **Figur 1-2:** Raspberry Pi – SoC (system-on-chip)

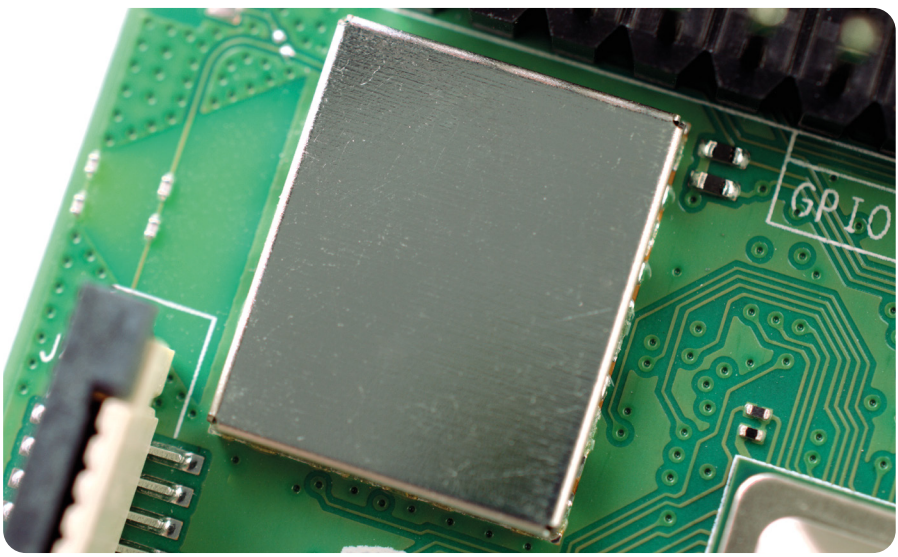
Navnet system-on-chip er en god pekepinn på hva du vil finne hvis du fjerner metalldekselet: en silisiumbrikke, kjent som en *integrert krets,* som inneholder mesteparten av Raspberry Pi-systemet. Dette omfatter den *prosessorenheten* (CPU), ofte ansett som «hjernen» til en datamaskin, og *grafikkbehandlingsenheten* (GPU), som håndterer det du ser på skjermen.

Men en hjerne fungerer ikke uten hukommelse, og like ved siden av SoC finner du akkurat det: en brikke til, som ser ut som en liten, svart plastfirkant (**Figur 1-3**, neste side). Dette er Raspberry Pis *RAM (Random Access Memory)*. Når du jobber med Raspberry Pi, er det RAM som oppbevarer det du holder på med. Først når du lagrer arbeidet, blir det skrevet til microSD-kortet. Til sammen danner disse komponentene Raspberry Pis flyktige og ikke-flyktige minner. Det flyktige RAM-kortet mister innholdet når Raspberry Pi slås av, mens det ikke-flyktige microSD-kortet lagrer innholdet.



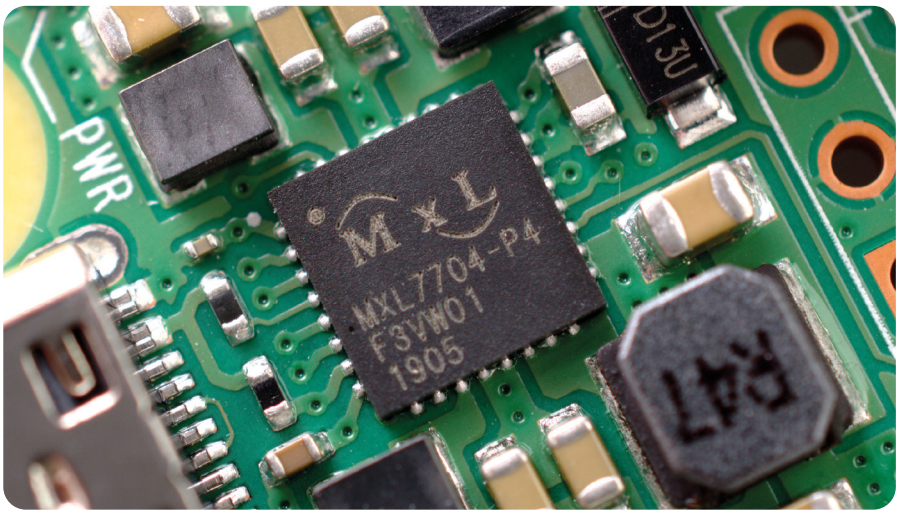
▲ **Figur 1-3:** Raspberry Pi – RAM (Random Access Memory)

Øverst til høyre på brettet finner du et annet metaldeksel (**Figur 1-4**) over *radioen*, komponenten som gjør at Raspberry Pi kan kommunisere trådløst med enheter. Selve radioen fungerer som to hovedkomponenter: en *WiFi-radio*, for tilkobling til datanettverk, og en *Bluetooth-radio*, for tilkobling til eksterne enheter som mus og for å sende eller motta data fra smartenheter i nærheten, for eksempel sensorer eller smarttelefoner.



▲ **Figur 1-4:** Raspberry Pi – radiomodul

Du finner en annen svart, plastbelagt brikke på undersiden av kortet, rett bak det midtre settet med USB-porter. Dette er *USB-kontrolleren*. Den har driftsansvaret for de fire USB-portene. Ved siden av denne ser du en enda mindre brikke, *nettverkskontrolleren*, som håndterer Raspberry Pis Ethernet-nettverksport. Den siste svarte brikken, som er mindre enn de andre, finner du litt ovenfor USB Type-C-strømkontakten øverst til venstre på kortet (**Figur 1-5**). Denne kalles en *integrert krets for strømstyring (PMIC)*, og konverterer strømmen som kommer inn fra mikro-USB-porten, til strømmen som Raspberry Pi trenger for å fungere.



▲ **Figur 1-5:** Raspberry Pi – integrert krets for strømstyring (PMIC)

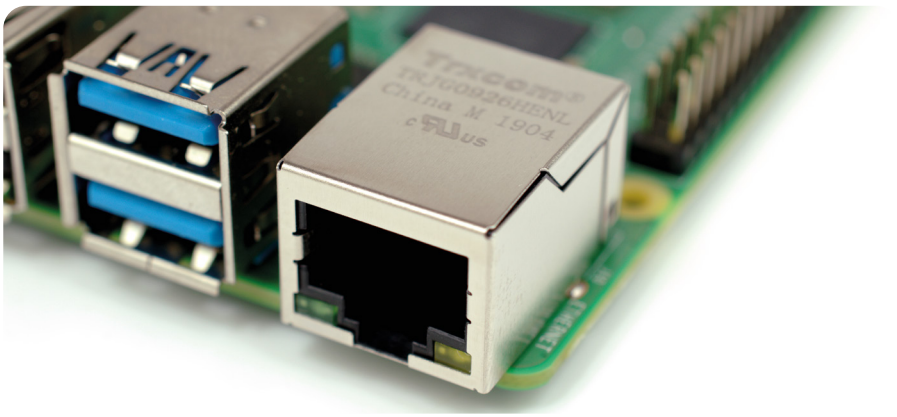
Ikke bekymre deg hvis det blir mye å huske på én gang. Du trenger ikke å vite hva hver komponent er, eller hvor du finner den på kretskortet, for å kunne bruke Raspberry Pi.

Porter på Raspberry Pi

Raspberry Pi har en rekke porter. For det første finnes det fire *USB-porter (universell seriebuss)* (**Figur 1-6**) midt på og til høyre nederst på kortet. Med disse portene kan du koble USB-kompatible eksterne enheter til Raspberry Pi, fra tastaturer og mus til digitale kameraer og minnepinner. Teknisk sett finnes det to typer USB-porter. De med sorte indre deler er USB 2.0-porter, basert på versjon to av USB-standarden. De med blå deler er raskere USB 3.0-porter, basert på den nyere versjon tre.

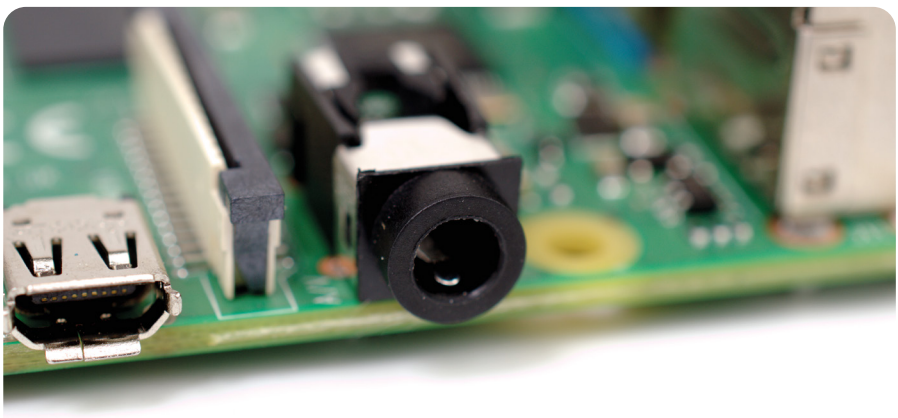
▲ **Figur 1-6:** Raspberry Pi – USB-porter

Til høyre for USB-portene finner du en *Ethernet-port*, også kalt en *nettverksport* (**Figur 1-7**). Du kan bruke denne porten når du vil koble Raspberry Pi til et kablet datanettverk med en kabel som har en såkalt RJ45-kontakt i enden. Hvis du ser nøye på Ethernet-porten, ser du to LED-lamper nederst. Dette er statuslamper som viser at tilkoblingen fungerer.



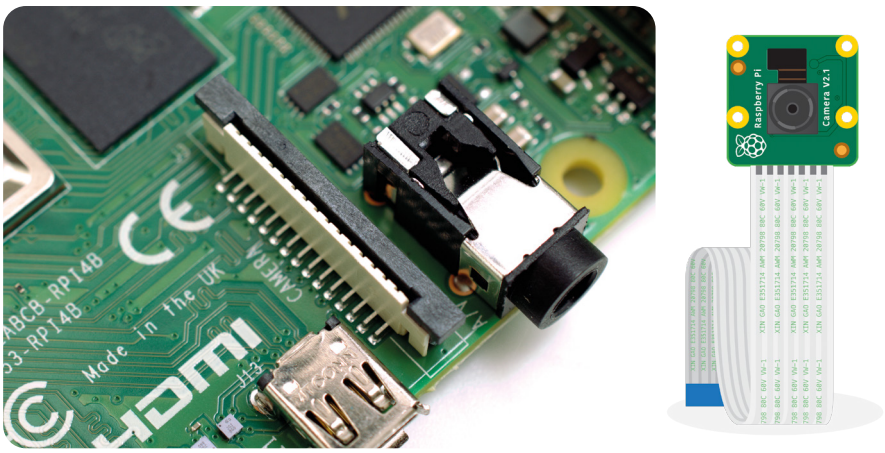
Figur 1-7: Raspberry Pi – Ethernet-port

Like over USB-portene, helt til venstre på Raspberry Pi, finner du en *3,5 mm AV-kontakt (audiovisuell)* (**Figur 1-8**). Dette kalles også en *hodetelefonkontakt*, og den kan brukes til akkurat det formålet – selv om du vil få bedre lyd hvis du kobler til aktive høyttalere i stedet for hodetelefoner. Den har imidlertid en skjult, ekstra funksjon. I tillegg til lyd har 3,5 mm AV-kontakten et videosignal som kan kobles til tv-er, projektorer og andre skjermer som støtter et *sammensatt videosignal*, med en spesiell kabel som kalles en *TRRS-adapter (tip-ring-ring-sleeve)*.



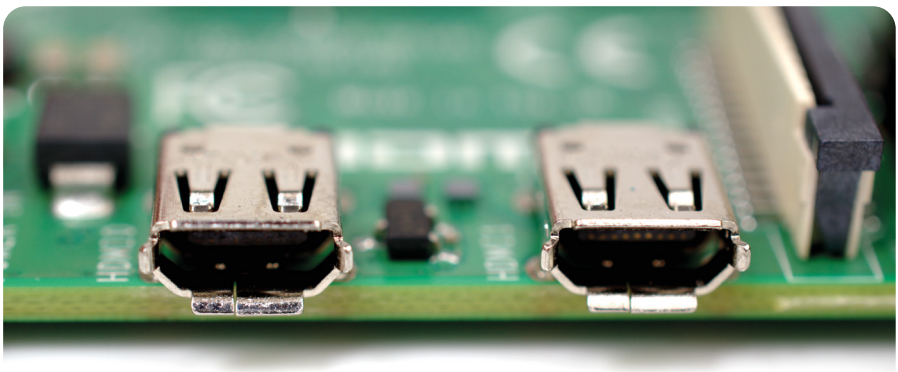
Figur 1-8: Raspberry Pi – 3,5 mm AV-kontakt

Rett over 3,5 mm AV-kontakten ser du en merkelig kontakt med plastklaff som kan trekkes opp. Dette er *kamerakontakten*, også kalt *CSI (Camera Serial Interface)* (**Figur 1-9**). Denne kontakten er beregnet på den spesialutviklede Raspberry Pi-kameramodulen (du kan lese mer om denne modulen i **Kapittel 8: Raspberry Pi Camera Module**.)



Figur 1-9: Raspberry Pi – kamerakontakt

Over denne, på venstre side av kortet, finner du *mikro-HDMI-portene (High Definition Multimedia Interface)*. Disse er mindre versjoner av kontaktene du finner på en spillkonsoll, tv-mottaker eller tv (**Figur 1-10**). Multimediedelen av navnet forteller deg at den fører både lyd- og videosignaler, mens «high-definition» forteller deg at du kan forvente topp kvalitet. Du bruker disse portene når du vil koble Raspberry Pi til én eller to skjermenheter: dataskjerm, tv eller projektor.

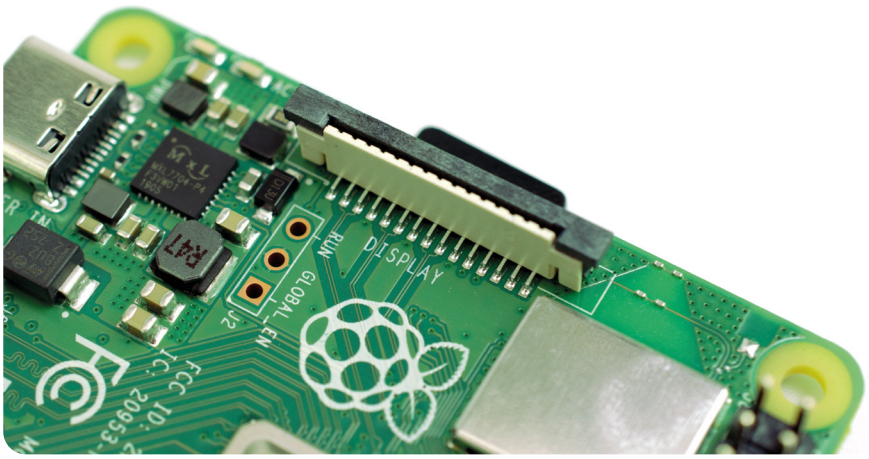


Figur 1-10: Raspberry Pi – mikro-HDMI-porter

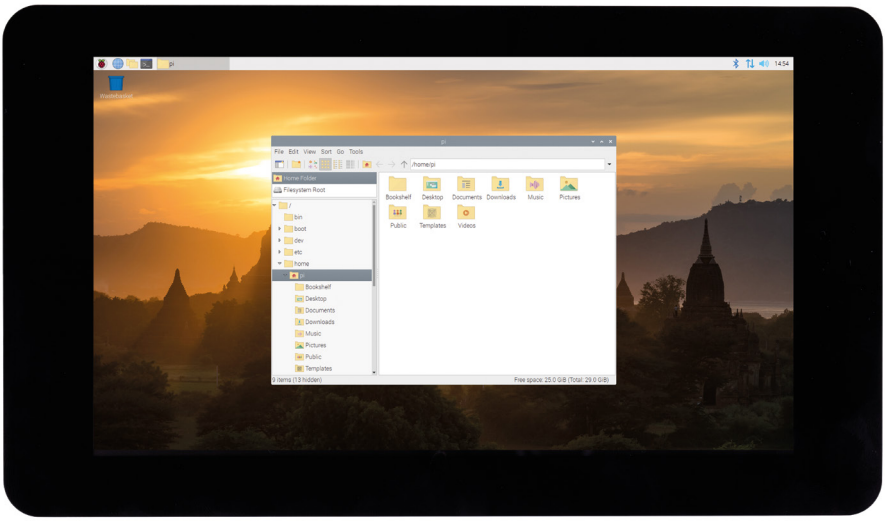
Over HDMI-portene finner du en *USB Type-C strøminngang* (**Figur 1-11**), som du bruker når du vil koble Raspberry Pi til en strømkilde. USB Type-C-porten er vanlig på smarttelefoner, nettbrett og andre bærbare enheter. Selv om du kan bruke en vanlig mobillader til å drive Raspberry Pi, er det best å bruke den offisielle USB Type-C strømadapteren for Raspberry Pi.

Figur 1-11: Raspberry Pi – USB Type-C strøminngang

Øverst på kretskortet ser du en annen merkelig kontakt **(Figur 1-12)**, som ved første øyekast ser ut til å være identisk med kamerakontakten. Den er imidlertid akkurat det motsatte: en *skjermkontakt*, eller *DSI (Display Serial Interface)*, som er beregnet for bruk med en berøringsskjerm for Raspberry Pi (**Figur 1-13**, neste side).

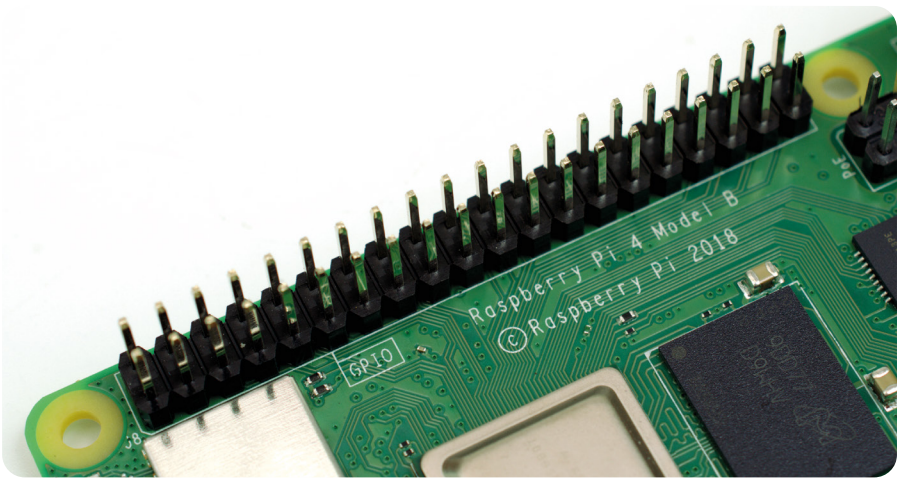


Figur 1-12: Raspberry Pi – skjermkontakt (DSI)



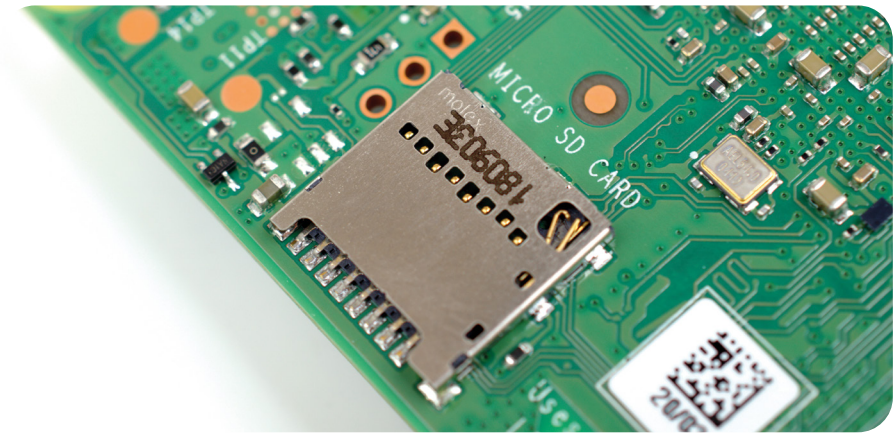
▲ **Figur 1-13:** Berøringsskjermen til Raspberry Pi

På høyre side av kretskortet finner du 40 metallpinner, fordelt på to rader med 20 pinner (**Figur 1-14**). Dette er *GPIO-pinnerekke (general-purpose input/output)*, en funksjon på Raspberry Pi som brukes til å kommunisere med annen maskinvare – fra indikatorlamper (LED) og knapper til temperaturfølere, joysticker og pulsmålere. Du kan finne ut mer om GPIO-pinnerekken i **Kapittel 6: Fysisk databehandling med Scratch og Python**. Rett under og til venstre for pinnerekken ser du et annet, mindre pinnerekke med fire pinner. Dette brukes ved tilkobling av tilleggsutstyr som PoE-HAT (Power over Ethernet), som gjør at Raspberry Pi kan få strøm fra en nettverkstilkobling i stedet for USB Type-C-porten.



▲ **Figur 1-14:** Raspberry Pi – GPIO-pinnerekke

Raspberry Pi har én port til, men du ser den ikke på oversiden. Snu kortet, så finner du en *microSD-kortkontakt* på motsatt side av kortet i forhold til skjermkontakten (**Figur 1-15**). Dette er minnet til Raspberry Pi. MicroSD-kortet som sitter her, inneholder alle filene du lagrer, all programvaren du installerer, og operativsystemet som gjør at Raspberry Pi fungerer.



Figur 1-15: Raspberry Pi – microSD-kortkontakt

▲ **Figur 1-16:** Raspberry Pi 400 har et integrert tastatur

Raspberry Pi 400

Raspberry Pi 400 har de samme komponentene som Raspberry Pi 4, men de er plassert i et tastaturhus. I tillegg til å beskytte dem tar tastaturhuset mindre plass på skrivebordet og bidrar til å holde kablene ryddigere.

Raspberry Pi 400 består av de samme kjernekomponentene som Raspberry Pi 4, inkludert SoC og minnet. Du kan ikke se dem, men de er der. Derimot ser du de eksterne delene, først og fremst tastaturet (**Figur 1-16**). Øverst til høyre på tastaturet ser du tre indikatorlamper (LED-lamper). Den første lampen lyser når du trykker på Num Lock-tasten, som får noen av tastene til å fungere som et talltastatur på tastatur i full størrelse. Den andre lampen lyser når du trykker på Caps Lock, som bytter mellom små og store bokstaver. Den siste lampen lyser når Raspberry Pi 400 er slått på.

Du finner portene på baksiden av Raspberry Pi 400 (**Figur 1-17**). Porten helt til venstre (sett bakfra) er GPIO-pinnerekken. Dette er det samme hodet som er beskrevet på side 17, men snudd opp-ned. Den første pinnen, pinne 1, er øverst til høyre, mens den siste pinnen, pinne 40, er nederst til venstre. Du kan finne ut mer om GPIO-pinnerekken i Kapittel 6: Fysisk databehandling med Scratch og Python.

Figur 1-17: Du finner portene på baksiden av Raspberry Pi 400

Ved siden av GPIO-pinnerekken finner du microSD-kortsporet. Dette inneholder microSD-kortet, som fungerer som Raspberry Pi 400s minne, der du lagrer operativsystemet, programmer og andre data. MicroSD-kortet er forhåndsinstallert på Raspberry Pi 400. Du kan fjerne det ved å skyve forsiktig på kortet til det klikker og utløses. Når du setter inn kortet igjen, må du passe på at de blanke metallkontaktene vender nedover. Kortet skal gli inn med et svakt klikk.

De to neste portene er mikro-HDMI-porter, som brukes til å koble til en dataskjerm, tv eller annen type skjerm. I likhet med Raspberry Pi 4 støtter Raspberry Pi 400 opptil to skjermer. Ved siden av disse finner du en strøminngang av typen USB Type-C, som brukes ved tilkobling av Raspberry Pis strømadapter eller kompatibel strømforsyning via USB.

De to blå portene er USB 3.0-porter, som gir høyhastighetstilkobling til for eksempel SSD-er, minnepinner, skrivere med mer. Den hvite porten til høyre for disse portene er en USB 2.0-port med lavere hastighet, hvor du kan koble til den medfølgende Raspberry Pi-musen.

Den siste porten er en Gigabit Ethernet-nettverksport. Her kan du koble Raspberry Pi 400 til nettverket med en nettverkskabel i stedet for å bruke det innebygde, trådløse Wi-Fi-nettverket. Du kan lese mer om hvordan du kobler Raspberry Pi 400 til et nettverk i Kapittel 2: Komme i gang med Raspberry Pi.

Kapittel 2

Komme i gang med Raspberry Pi

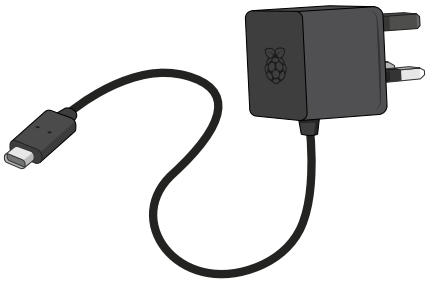
Oppdag de viktigste elementene du trenger for Raspberry Pi, og hvordan du kobler dem sammen for at alt skal fungere

Raspberry Pi er utviklet for å være så rask og enkel å konfigurere og bruke som mulig. I likhet med alle datamaskiner er den imidlertid avhengig av ulike eksterne komponenter, kalt *eksterne enheter*. Når du tar en titt på det tomme kretskortet til Raspberry Pi, er det lett å tro at det kan bli vanskelig. Kortet ser helt annerledes ut enn de innkapslede, lukkede datamaskinene du kanskje er vant til, men bare slapp av. Du kan få Raspberry Pi til å fungere på godt under ti minutter ved å følge fremgangsmåten i denne veiledningen.

Hvis denne veiledningen fulgte med da du kjøpte et Raspberry Pi Desktop Kit eller en Raspberry Pi 400, har du nesten alt du trenger for å komme i gang. I tillegg må du ha en dataskjerm eller TV med HDMI-tilkobling, samme type kontakt som brukes av TV-mottakere (STB), Blu-ray-spillere og spillkonsoller, slik at du kan se hva Raspberry Pi gjør.

Hvis du kjøpte Raspberry Pi uten tilbehør, trenger du også følgende:

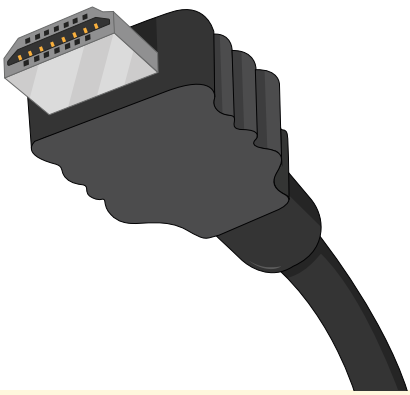
- **USB-strømadapter** – 5 V 3 A strømadapter med USB Type-C-kontakt. Vi anbefaler at du bruker den offisielle strømadapteren for Raspberry Pi, siden den tåler de raskt skiftende strømkravene til Raspberry Pi.



- **microSD-kort med NOOBS** – microSD-kortet fungerer som Raspberry Pis permanente minne. Alle filene du oppretter, programvaren du installerer, og selve operativsystemet lagres på kortet. Du kan komme i gang med et 8 GB-kort, men 16 GB gir deg rom for forbedring. Du sparer tid ved å bruke et kort med NOOBS (New Out-Of-Box Software) forhåndsinstallert. Hvis ikke ser du **Vedlegg A** for å finne ut hvordan du installerer et operativsystem (OS) på et tomt kort.

- **USB-tastatur og -mus** – tastaturet og musen brukes til å kontrollere Raspberry Pi. Nesten alle typer kablede eller trådløse tastaturer og mus med USB-kontakt fungerer med Raspberry Pi, selv om noen spilltastaturer med fargerikt belysning trekker for mye strøm til å kunne brukes med pålitelige resultater.

- **Mikro-HDMI-kabel** – denne overfører lyd og bilder fra Raspberry Pi til TV-en eller skjermen. Den ene enden av kabelen har en mikro-HDMI-kontakt for Raspberry Pi. Den andre enden har en HDMI-kontakt i full størrelse for skjermen. Du kan også bruke en mikro-HDMI-til-HDMI-adapter og en vanlig HDMI-kabel i full størrelse. Hvis du bruker en skjerm uten HDMI-kontakt, kan du kjøpe adaptere av typen mikro-HDMI-til-DVI-D, DisplayPort eller VGA. Hvis du vil koble til en eldre TV som bruker kompositttvideo eller har en SCART-kontakt, bruker du en 3,5 mm TRRS lyd-/videokabel.



Raspberry Pi kan trygt brukes uten etui, men da må du ikke plasserer den på en metalloverflate som kan lede strøm og gi kortslutning. Et valgfritt etui kan imidlertid gi ekstra beskyttelse. Desktop Kit inneholder det offisielle Raspberry Pi-etuiet, men du kan også kjøpe etuier fra tredjeparter hos alle gode forhandlere.

Hvis du skal bruke Raspberry Pi på et kablet nettverk i stedet for et trådløst nettverk (WiFi), trenger du også en nettverkskabel. Den ene enden av denne kobles til nettverkets svitsj (hub) eller ruter. Hvis du planlegger å bruke Raspberry Pis innebygde trådløse radio, trenger du ikke kabel. Du må imidlertid vite navn og nøkkel eller passfrase for det trådløse nettverket.

KONFIGURASJON AV RASPBERRY PI 400

Følgende instruksjoner gjelder konfigurasjon av Raspberry Pi 4 eller et annet tomt kort i Raspberry Pi-serien.

På side 32 finner du instruksjoner for å konfigurere Raspberry Pi 400.

Konfigurere maskinvaren

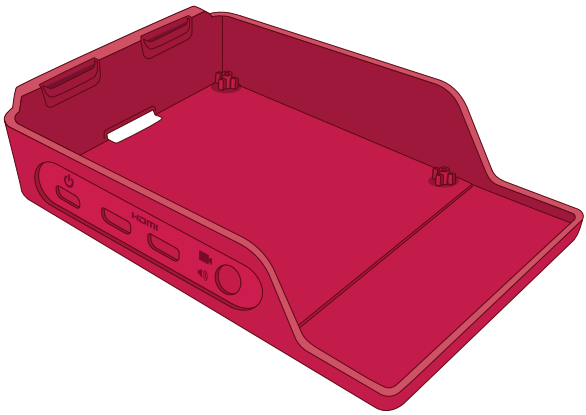
Begynn med å pakke ut Raspberry Pi fra esken. Raspberry Pi er en robust maskin, men det betyr ikke at den ikke kan skades. Venn deg til å holde kortet i kantene i stedet for på de flate sidene, og vær ekstra forsiktig med metallpinnene. Hvis pinnene blir bøyd, vil det gjøre det vanskelig å bruke tilleggskort og annen ekstra maskinvare. I verste fall kan det føre til kortslutning som skader Raspberry Pi.

Hvis du ikke allerede har gjort det, kan du lese **Kapittel 1: Bli kjent med Raspberry Pi** for å se akkurat hvor de forskjellige portene er og hva de gjør.

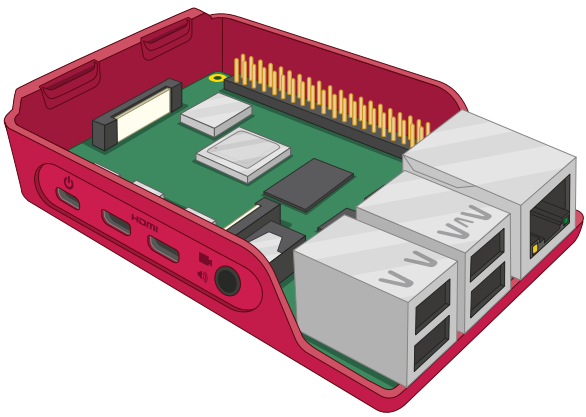
Montere etuiet

Hvis du skal bruke Raspberry Pi i et etui, bør du begynne med å installere etuiet. Hvis du bruker det offisielle Raspberry Pi-etuiet, begynner du med å skille de to delene fra hverandre: den røde basen og det hvite lokket.

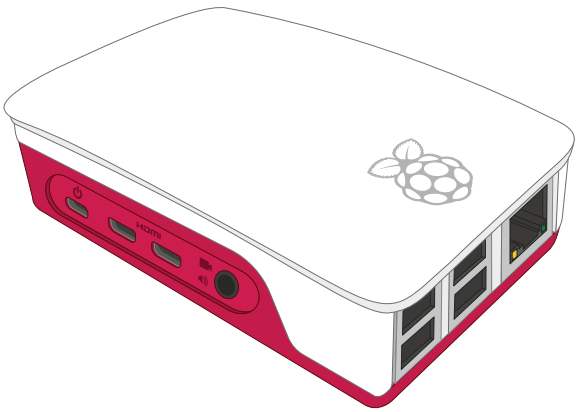
1 **Hold opp basen slik at den hevede enden er til venstre og den lave enden til høyre.**



2 **Hold Raspberry Pi (uten microSD-kort instalert) etter USB- og Ethernet-portene – i svak vinkel. Sett kontaktene (USB Type-C, 2 × mikro-HDMI og 3,5 mm) i hullene på siden av basen, og legg deretter den andre siden forsiktig ned slik at den ligger flatt.**

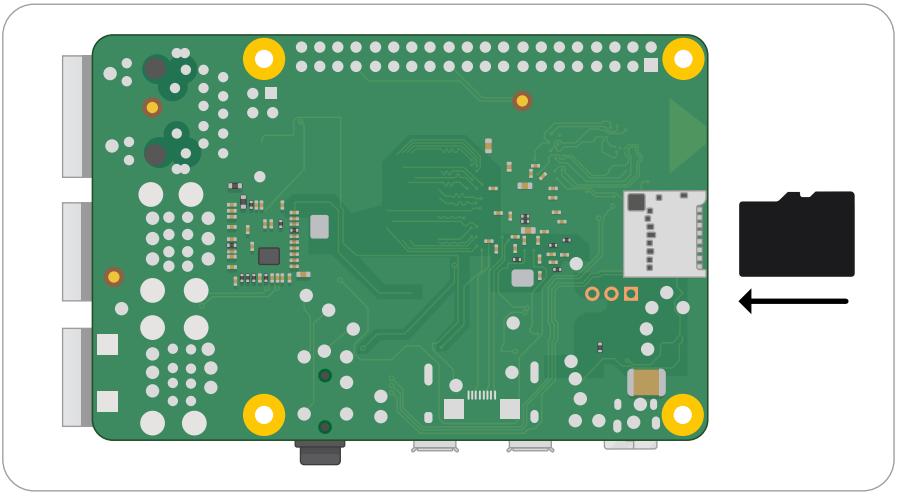


3 **Ta opp det hvite lokket og plasser de to klipsene til venstre i de tilsvarende hullene på venstre side av basen, over microSD-kortsporet. Når de er på plass, skyver du høyre side (over USB-portene) ned til du hører et klikk.**

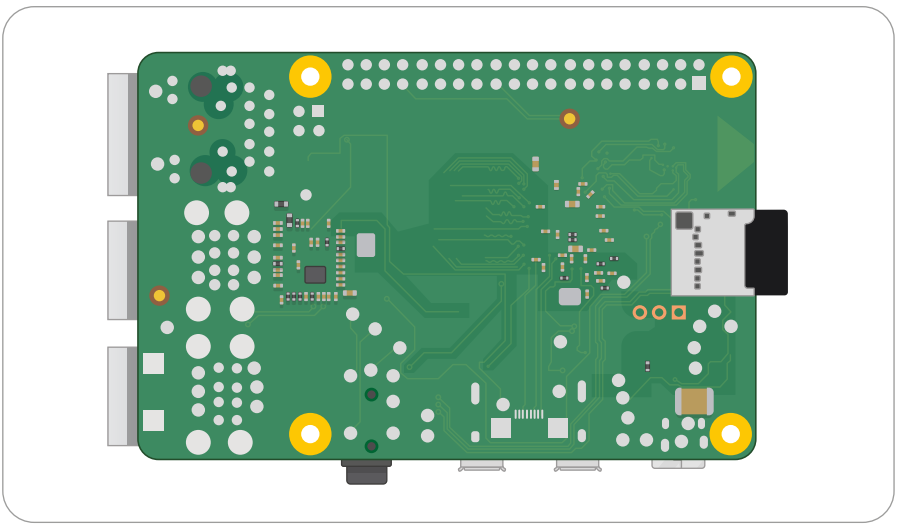


Koble til microSD-kortet

Når du skal installere microSD-kortet, som er Raspberry Pis *minne*, snur du Raspberry Pi (eventuelt i etuiet) og skyver kortet inn i microSD-sporet med etiketten vendt bort fra Raspberry Pi. Det kan bare føres inn på én måte og skal gli inn uten å bruke for mye kraft.



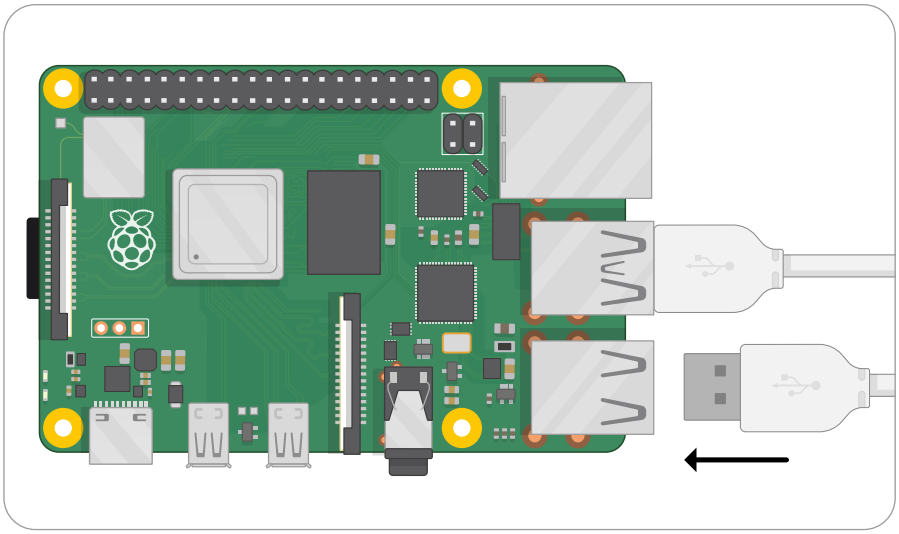
MicroSD-kortet glir inn i kontakten, men stopper uten en klikkelyd.



Hvis du vil fjerne det igjen senere, tar du bare tak i enden av kortet og trekker det forsiktig ut. Hvis du bruker en eldre modell av Raspberry Pi, må du først dytte kortet litt inn for å frigjøre det. Dette er ikke nødvendig med Raspberry Pi 3 eller 4.

Koble til tastatur og mus

Koble tastaturets USB-kabel til en av de fire USB-portene (2.0 eller 3.0) på Raspberry Pi. Hvis du bruker det offisielle Raspberry Pi-tastaturet, har det en USB-port for musen på baksiden. Hvis ikke kan du bare koble musens USB-kabel til en annen USB-port på Raspberry Pi.



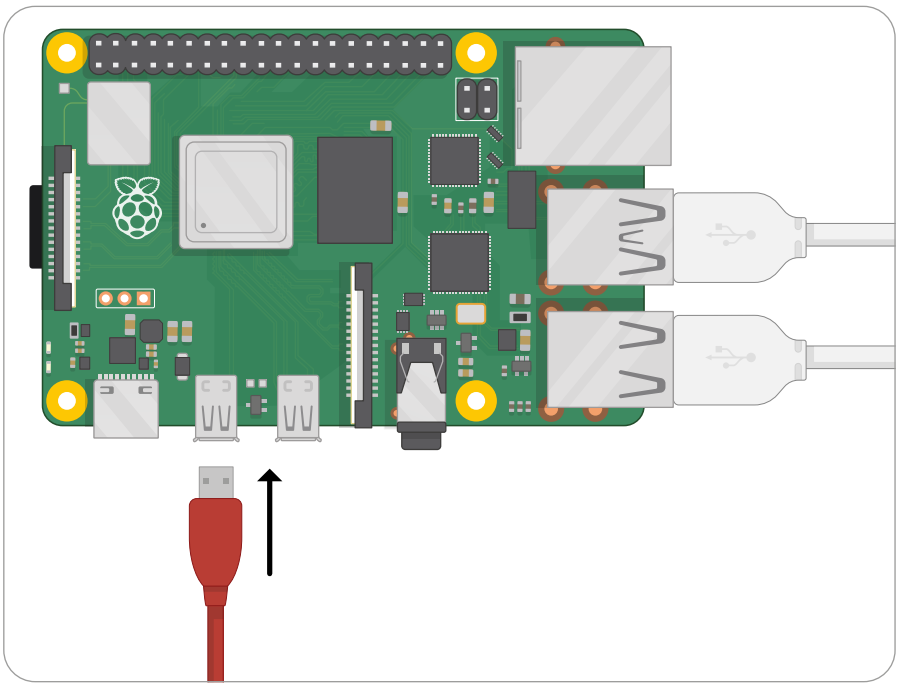
USB-kontaktene for tastatur og mus skal gli enkelt inn. Hvis du må tvinge kontakten inn, er det noe galt. Sjekk at riktig side av USB-kontakten vender opp.

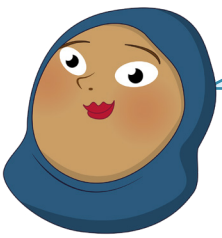
TASTATUR OG MUS

Tastaturet og musen er de viktigste hjelpemidlene for å fortelle Raspberry Pi hva den skal gjøre. Innen databehandling kalles disse *inndataenheter*, i motsetning til skjermen, som er en *utdataenhet*.

Koble til en skjerm

Ta mikro-HDMI-kabelen og koble den minste enden til mikro-HDMI-porten nærmest USB Type-C-porten på Raspberry Pi og den andre enden til skjermen. Hvis skjermen har mer enn én HDMI-port, ser du etter et portnummer ved siden av selve kontakten. Du må flytte TV-en til denne inngangen for å se skjermen til Raspberry Pi. Hvis du ikke ser et portnummer, er det ikke så farlig. Du kan bare gå gjennom hver inngang til du finner Raspberry Pi.



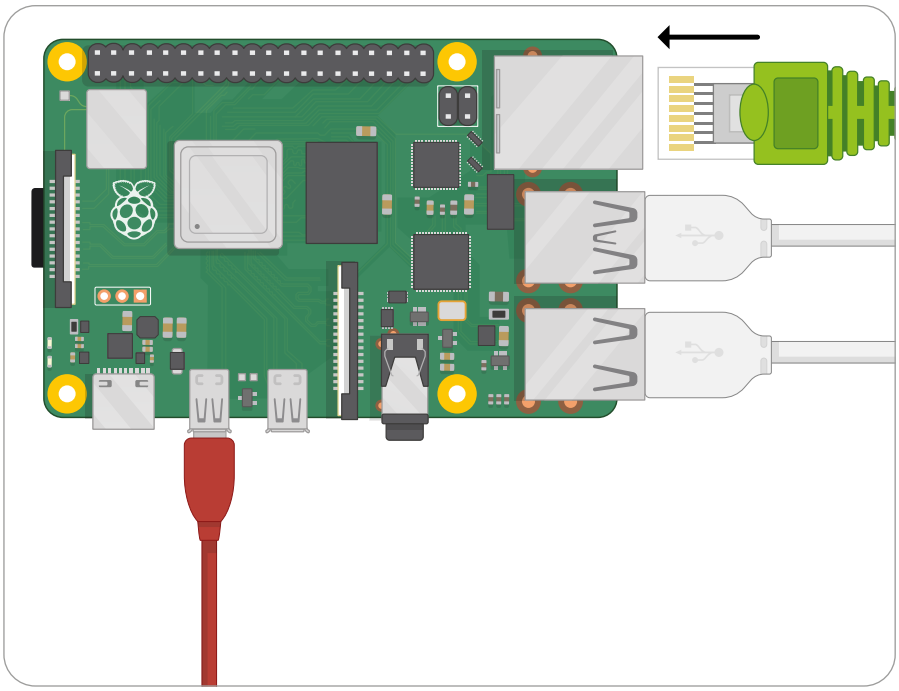


TV-TILKOBLING

Hvis TV-en eller skjermen ikke har en HDMI-kontakt, betyr det ikke at du ikke kan bruke Raspberry Pi. Med adapterkabler (selges i alle elektronikkbutikker) kan du konvertere mikro-HDMI-porten på Raspberry Pi til DVI-D, DisplayPort eller VGA for bruk med eldre dataskjermer. Du kan ganske enkelt koble disse til Raspberry Pis micro-HDMI-port. Deretter bruker du en egnet kabel til å koble adapterkabelen til skjermen. Hvis TV-en bare har en inngang for kompositttvideo eller SCART, kan du kjøpe 3,5 mm TRRS-adapterkabler og kompositt-til-SCART-adaptere som kobles til den 3,5 mm AV-kontakten.

Koble til en nettverkskabel (valgfritt)

Hvis du vil koble Raspberry Pi til et kablet nettverk, bruker du en nettverkskabel, kalt Ethernet-kabel. Skyv den inn i Ethernet-porten på Raspberry Pi, med plastklemmen nedover, til du hører et klikk. Når du skal fjerne kabelen, klemmer du bare plastklemmen innover mot pluggen og løsner kabelen forsiktig.

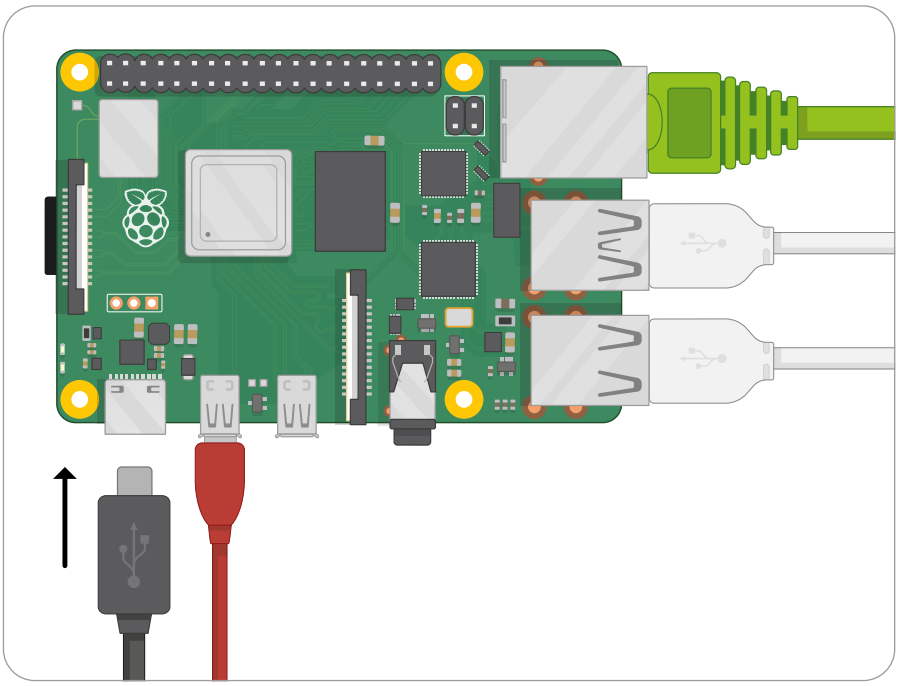


Den andre enden av nettverkskabelen kobles til en av de ledige portene på nettverkshuben, svitsjen eller ruteren på samme måte.

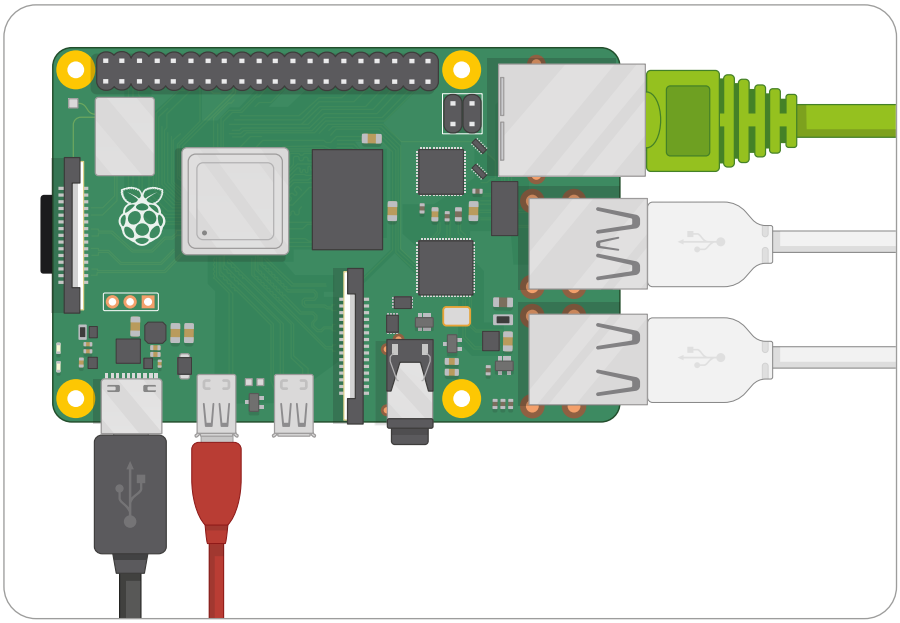
Koble til en strømforsyning

Det siste trinnet i maskinvarekonfigurasjonen er å koble Raspberry Pi til strømforsyningen. Derfor bør du vente med dette til du er klar til å konfigurere programvaren: Raspberry Pi har ingen strømbryter og slås på så snart den kobles til en strømforsyning.

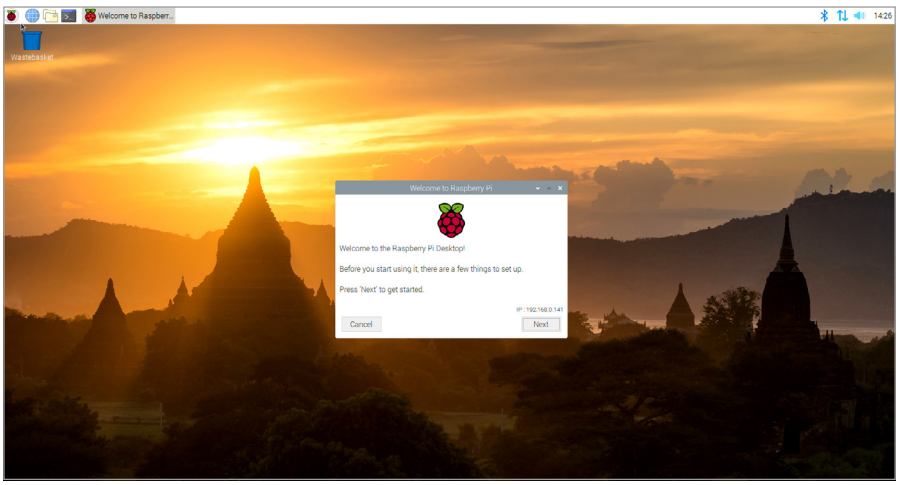
Først kobler du USB Type-C-enden av strømkabelen til USB Type-C-kontakten på Raspberry Pi. Den kan settes inn begge veier og skal gli lett inn. Hvis strømadapteren har en avtakbar kabel, må du sørge for at den andre enden er koblet til selve adapteren.



Til slutt kobler du strømadapteren til en stikkontakt. Raspberry Pi starter med det samme. Supert! Du har satt sammen Raspberry Pi.



Et kort øyeblikk vil du se fire Raspberry Pi-logoer øverst til venstre på en svart skjerm. Det kan også hende at en blå skjerm vises når programvaren endrer størrelse, for å utnytte microSD-kortet fullt ut. Hvis du ser en svart skjerm, venter du noen minutter. Den første gangen du starter opp Raspberry Pi, må den foreta litt opprydding i bakgrunnen. Etter en stund vil du se skrivebordet og installasjonsveiviseren for Raspberry Pi OS, som i **Figur 2-1**. Operativsystemet er nå klart for konfigurering. Du lærer hvordan du gjør det i **Kapittel 3: Bruke Raspberry Pi**.



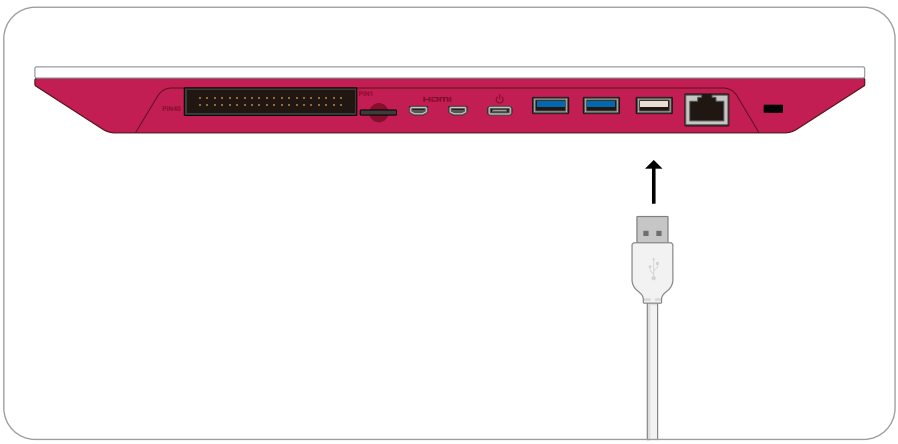
▲ **Figur 2-1:** Skrivebordet og installasjonsveiviseren for Raspberry Pi OS

Sette opp Raspberry Pi 400

I motsetning til Raspberry Pi 4 leveres Raspberry Pi 400 med et innebygd tastatur, og microSD-kortet er allerede installert. Du må likevel koble til et par kabler for å komme i gang, men det gjør du på et blunk.

Koble til en mus

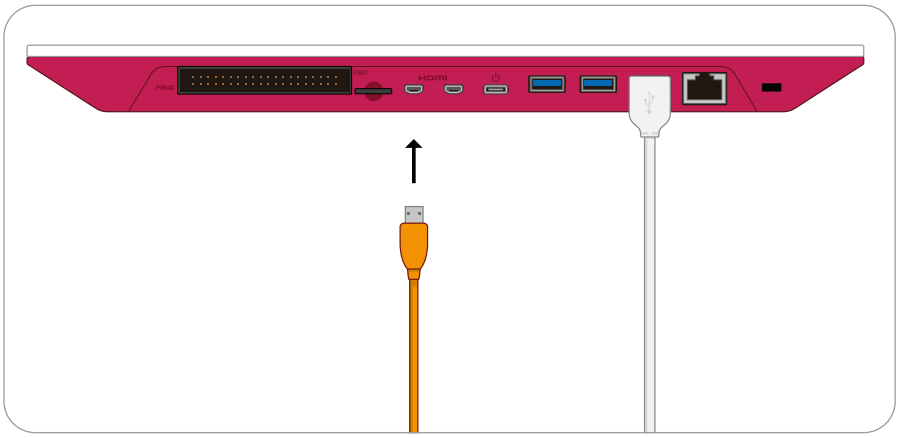
På Raspberry Pi 400 er tastaturet allerede tilkoblet, så du trenger bare å koble til musen. Sett USB-kabelen som er tilkoblet musen, inn i en av de tre USB-portene (2.0 eller 3.0) på baksiden av Raspberry Pi 400. Hvis du vil spare de to høyhastighets USB 3.0-portene til annet tilbehør, bruker du den hvite porten.



USB-kontakten skal være enkelt å sette inn. Hvis du må bruke makt, er det noe galt. Sjekk at riktig side av USB-kontakten vender opp.

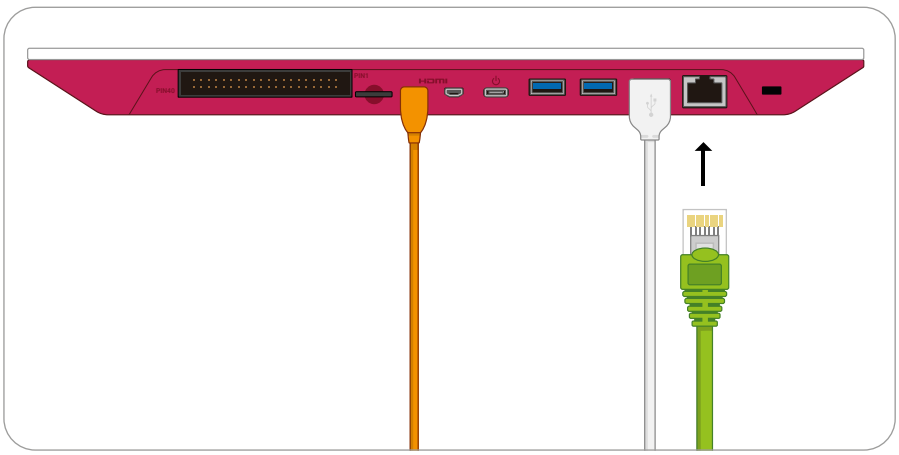
Koble til en skjerm

Koble den minste enden av mikro-HDMI-kabelen til mikro-HDMI-porten som er nærmest microSD-sporet på Raspberry Pi 400, og koble den andre enden til skjermen. Hvis skjermen har mer enn én HDMI-port, ser du etter et portnummer ved siden av selve kontakten. Du må flytte TV-en til denne inngangen for å se skjermen til Raspberry Pi. Hvis du ikke ser et portnummer, er det ikke så farlig. Du kan bare gå gjennom hver inngang til du finner Raspberry Pi.



Koble til en nettverkskabel (valgfritt)

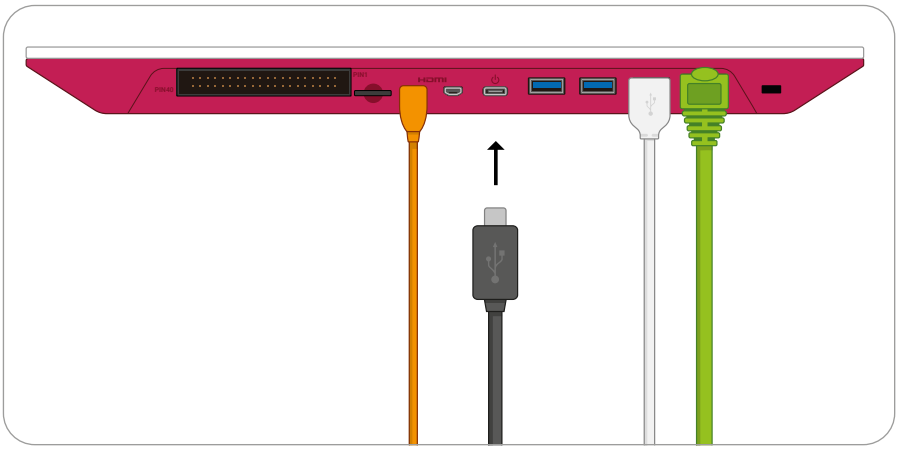
Hvis du vil koble Raspberry Pi 400 til et kablet nettverk, bruker du en nettverkskabel (Ethernet-kabel). Skyv den inn i Ethernet-porten på Raspberry Pi 400, med plastklemmen vendt opp, til du hører et klikk. Når du skal fjerne kabelen, klemmer du bare plastklemmen innover mot pluggen og løsner kabelen forsiktig.



Den andre enden av nettverkskabelen kobles til en av de ledige portene på nettverkshuben, svitsjen eller ruteren på samme måte.

Koble til en strømforsyning

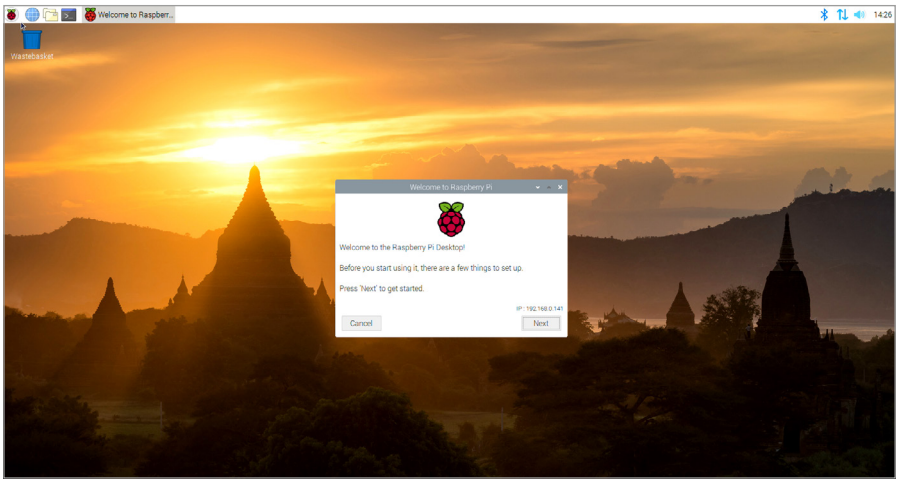
Det siste trinnet i maskinvarekonfigurasjonen er å koble Raspberry Pi 400 til strømforsyningen. Derfor bør du vente med dette til du er klar til å konfigurere programvaren: Raspberry Pi 400 har ingen strømbryter og slås på så snart den kobles til en strømforsyning. Først kobler du USB Type-C-enden av strømkabelen til USB Type-C-kontakten på Raspberry Pi. Den kan settes inn begge veier og skal gli lett inn. Hvis strømadapteren har en avtakbar kabel, må du sørge for at den andre enden er koblet til selve adapteren.



Til slutt kobler du strømadapteren til en stikkontakt. Raspberry Pi 400 starter med det samme. Supert! Du kar satt sammen Raspberry Pi 400.

Et kort øyeblikk vil du se fire Raspberry Pi-logoer øverst til venstre på en svart skjerm. Det kan også hende at en blå skjerm vises når programvaren endrer størrelse, for å utnytte

microSD-kortet fullt ut. Hvis du ser en svart skjerm, venter du noen minutter. Den første gangen du starter opp Raspberry Pi, må den foreta litt opprydding i bakgrunnen. Etter en stund vil du se skrivebordet og installasjonsveiviseren for Raspberry Pi OS, som i **Figur 2-2**. Operativsystemet er nå klart for konfigurering. Du lærer hvordan du gjør det i **Kapittel 3: Bruke Raspberry Pi**.

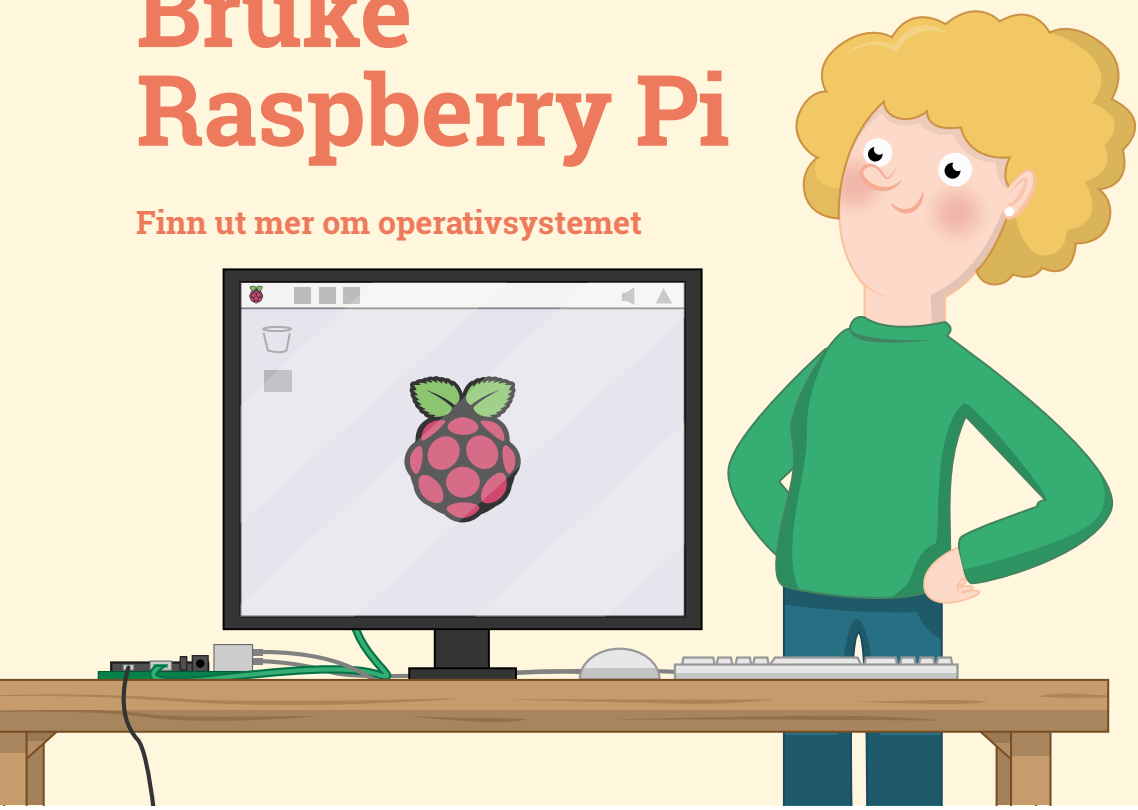


▲ **Figur 2-2:** Skrivebordet og installasjonsveiviseren for Raspberry Pi OS

Kapittel 3

Bruke Raspberry Pi

Finn ut mer om operativsystemet

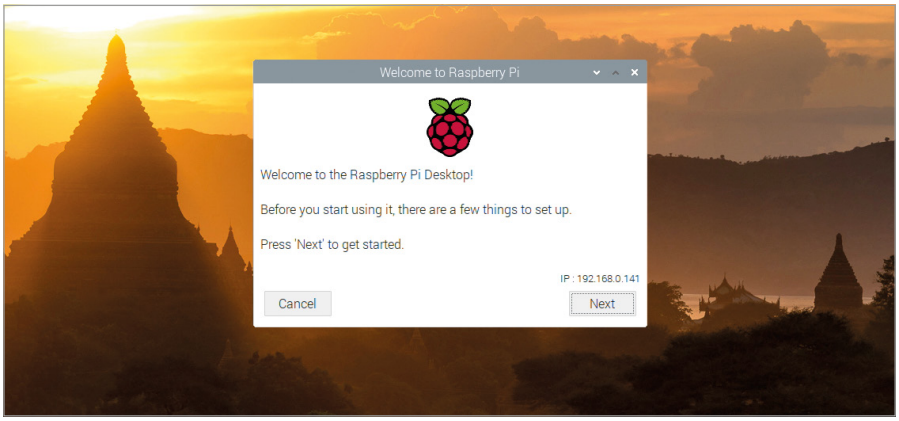


Raspberry Pi kan kjøre et bredt spekter av programvare, deriblant en rekke forskjellige operativsystemer – kjerneprogramvaren som får en datamaskin til å fungere. Det mest populære av disse, og det offisielle operativsystemet til Raspberry Pi Foundation, er Raspberry Pi OS. Det er basert på Debian Linux og er skreddersydd for Raspberry Pi. Operativsystemet leveres med en rekke ekstrautstyr – forhåndsinstallert og klart til bruk.

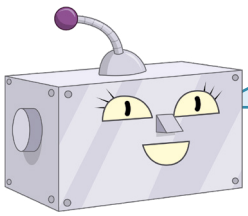
Hvis du bare har brukt Microsoft Windows eller Apple macOS, trenger du ikke å bekymre deg. Raspberry Pi OS er basert på de samme prinsippene for vinduer, ikoner, menyer og pekere (WIMP), så du vil raskt kjenne deg igjen. I dette kapittelet forklarer vi hvordan du kommer i gang, og gir deg en kort innføring i medfølgende programvare.

Velkomstveiviseren

Den første gangen du kjører Raspberry Pi OS, ser du velkomstveiviseren (**Figur 3-1**). Dette praktiske verktøyet viser deg hvordan du endrer innstillinger i Raspberry Pi OS, kjent som *konfigurasjon*, avhengig av hvordan og hvor du skal bruke Raspberry Pi.



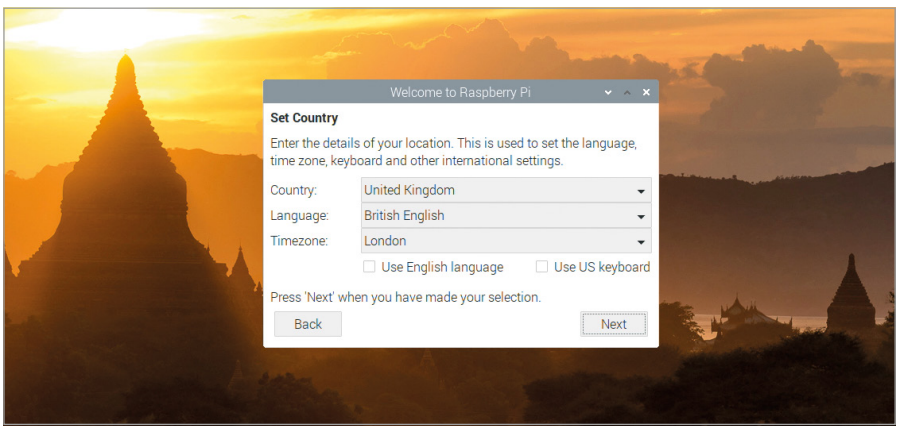
▲ **Figur 3-1:** Velkomstveiviseren



LUKKE VEIVISEREN

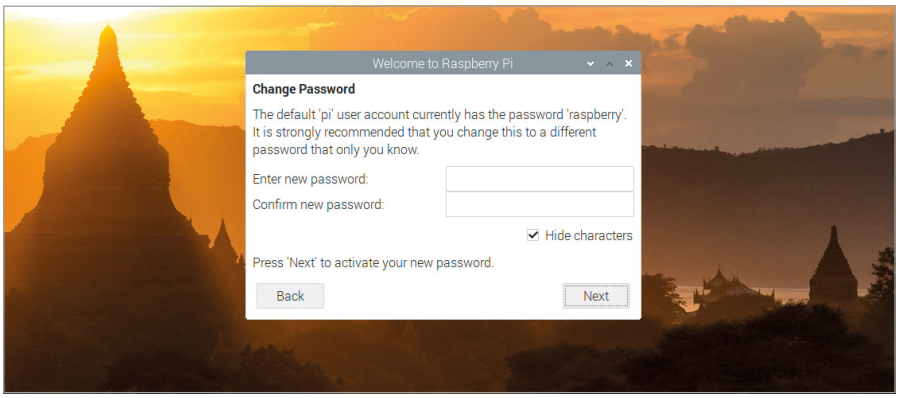
Du kan lukke velkomstveiviseren ved å klikke på Avbryt-knappen. Noen Raspberry Pi-funksjoner, for eksempel det trådløse nettverket, fungerer ikke før du svarer på minst det første settet med spørsmål.

Klikk på Next-knappen, og velg deretter land, språk og tidssone ved å klikke på hver rullegardinliste etter tur for å velge svaret fra listen (**Figur 3-2**). Hvis du bruker et tastatur med amerikansk oppsett, klikker du i avmerkingsboksen for å sikre at Raspberry Pi OS bruker riktig tastaturoppsett. Hvis du vil at skrivebordet og programmene skal vises på engelsk, uavhengig av det offisielle språket i landet ditt, klikker du i avmerkingsboksen «Use English language». Når du er ferdig, klikker du på Next.



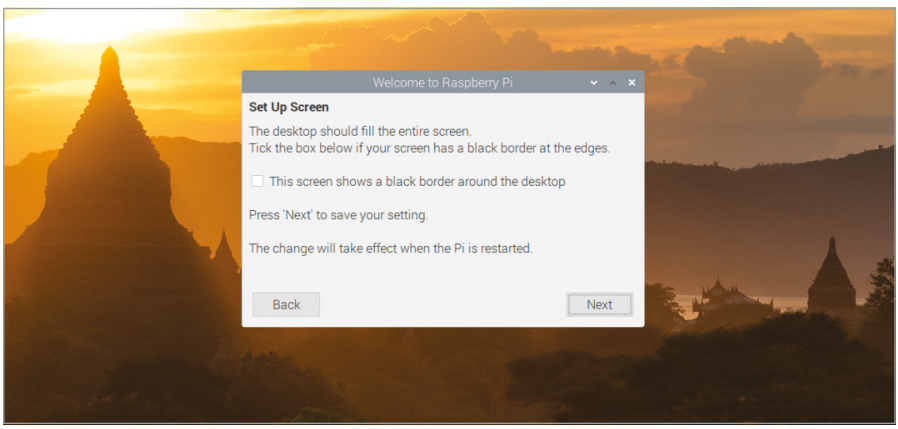
▲ **Figur 3-2:** Velge språk og andre alternativer

Den neste skjermen ber deg om å endre passordet til «pi»-brukeren (fra standardverdien «raspberry»). Av sikkerhetshensyn er det veldig lurt å opprette et nytt passord. Skriv inn passordet i feltene (**Figur 3-3**). Når du er fornøyd, klikker du på Next.



▲ **Figur 3-3:** Angi et nytt passord

Den neste skjermen spør deg om du ser en svart kantlinje rundt skjermen (**Figur 3-4**). Hvis Raspberry Pi-skrivebordet fyller hele TV-skjermen eller dataskjermen, lar du avmerkingsboksen være tom. Hvis skrivebordet har svarte kantlinjer og er mindre enn TV-en eller dataskjermen, merker du av i boksen. Når du er klar til å fortsette, klikker du på Next.



▲ **Figur 3-4:** Sjekke om du ser svarte kantlinjer

Denne skjermen lar deg velge Wi-Fi-nettverk fra en liste (**Figur 3-5**). Bla gjennom listen over nettverk med musen eller tastaturet, finn navnet til nettverket ditt og klikk på det. Deretter klikker du på Next. Hvis det trådløse nettverket er sikkert (det burde det være), blir du bedt om å oppgi passordet (også kjent som en forhåndsdelt nøkkel) til nettverket. Du finner vanligvis denne nøkkelen på et kort som leveres med ruteren, eller på bunnen av selve ruteren. Klikk på

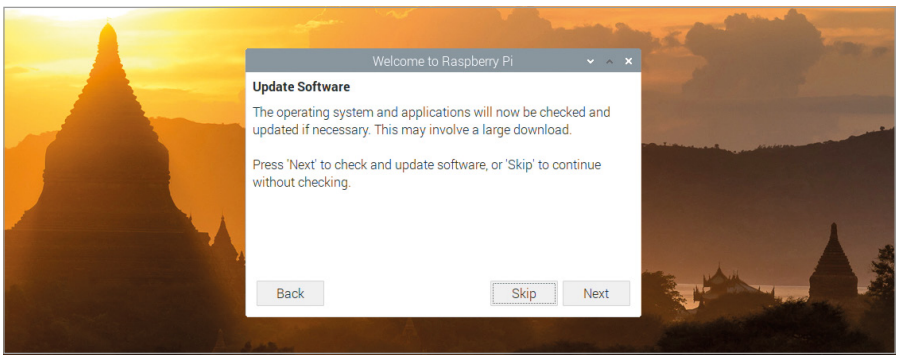
TRÅDLØSE NETTVERK

Bare seriene Raspberry Pi 3, Pi 4 og Pi Zero W kommer med et innebygd trådløst nettverk. Hvis du vil bruke en annen modell av Raspberry Pi med et trådløst nettverk, trenger du en USB Wi-Fi-adapter.

Next for å koble til nettverket. Hvis du ikke vil koble til et trådløst nettverk, kan du hoppe over dette trinne ved å klikke på Skip.

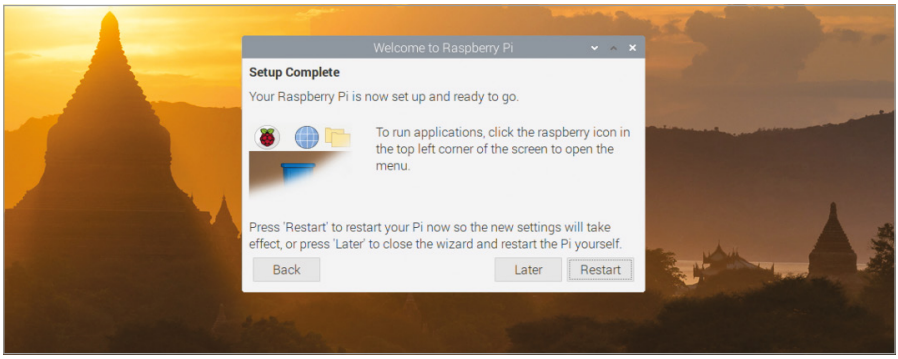
Figur 3-5: Velge et trådløst nettverk

På den neste skjermen kan du søke etter og installere oppdateringer for Raspberry Pi OS og annen programvare som er installert på Raspberry Pi (**Figur 3-6**). Raspberry Pi OS oppdateres regelmessig for å rette opp feil, legge til nye funksjoner og forbedre ytelsen. Hvis du vil installere disse oppdateringene, klikker du på Next. Hvis ikke klikker du på Skip. Det kan ta flere minutter å laste ned oppdateringene, så vær tålmodig. Når oppdateringene er installert, åpnes et vindu med teksten «System is up to date» (Systemet er oppdatert). Klikk på OK-knappen.



Figur 3-6: Se etter oppdateringer

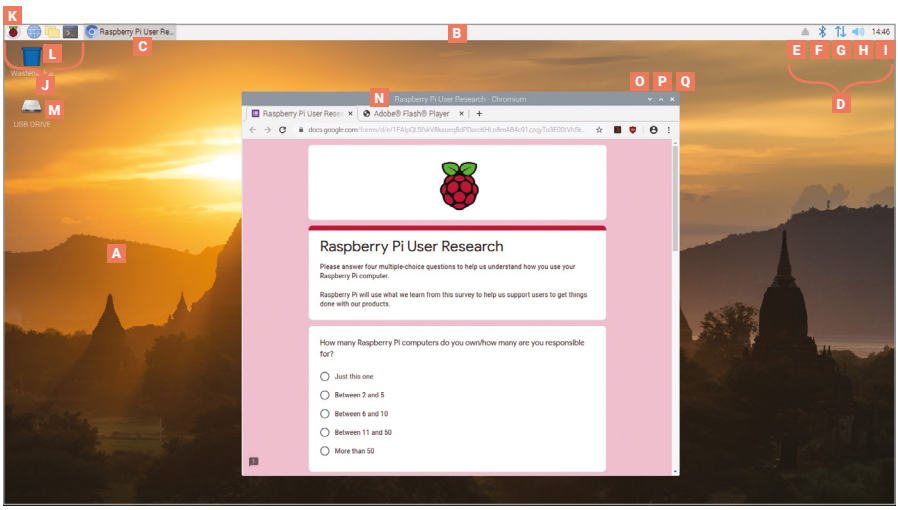
Den siste skjermen i velkomstveiviseren (**Figur 3-7**) har et enkelt formål. Noen endringer trer ikke i kraft før du starter Raspberry Pi på nytt (ogsjå kjent som en omstart). Hvis du blir bedt om å gjøre det, klikker du på Restart (Start på nytt). Deretter startes Raspberry Pi på nytt. Denne gangen vises ikke velkomstveiviseren. Den er fullført, og Raspberry Pi er klar til bruk.



▲ **Figur 3-7:** Starte Raspberry Pi på nytt

Navigere på skrivebordet

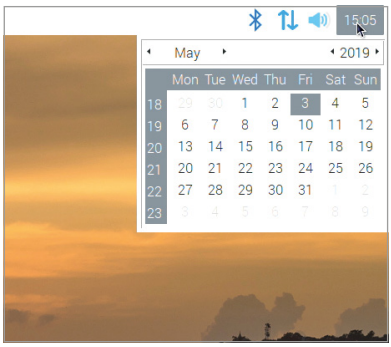
Versjonen av Raspberry Pi OS som er installert på de fleste Raspberry Pi-kort, er kjent som «Raspberry Pi OS med skrivebord». Dette viser til det viktigste grafiske brukergrensesnittet (**Figur 3-8**). Hoveddelen av dette skrivebordet består av et bilde, kjent som bakgrunnen **A** i **Figur 3-8**). Programmene du kjører, vises over bakgrunnen. Øverst på skrivebordet ser du en oppgavelinje (**B**), som lar deg laste inn hvert enkelt program. Disse angis deretter som oppgaver (**C**) på oppgavelinjen.



▲ **Figur 3-8:** Skrivebordet i Raspberry Pi OS

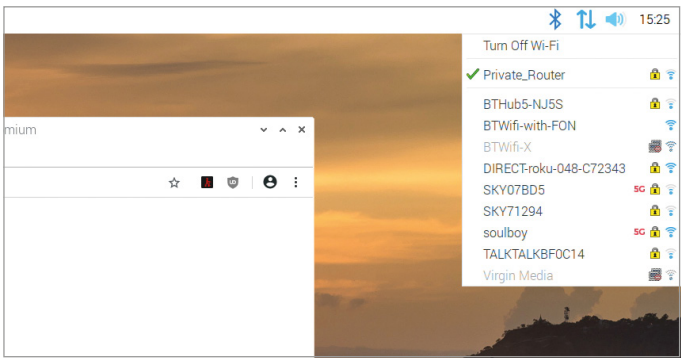
- **A** Bakgrunn
- **B** Oppgavelinje
- **C** Oppgave
- **D** Systemstatusfelt
- **E** Medieutløser
- **F** Bluetooth-ikon
- **G** Nettverksikon
- **H** Volumikon
- **I** Klokke
- **J** Velgerfelt
- **K** Menyikon (eller bringebærikon)
- **L** Papirkurvikon
- **M** Ikon for flyttbar stasjon
- **N** Vinduets tittellinje
- **O** Minimer
- **P** Maksimer
- **Q** Lukk

Til høyre på menylinjen finner du *systemstatusfeltet* (**D**). Hvis du har koblet *flyttbare lagringsmedier*, for eksempel USB-minnepinner, til Raspberry Pi, ser du et utløsersymbol (**E**). Når du klikker på dette symbolet, kan du trygt løse dem ut og fjerne dem. Helt til høyre på menyen ser du klokken (**I**). Klikk på den for å åpne en digital kalender (**Figur 3-9**).



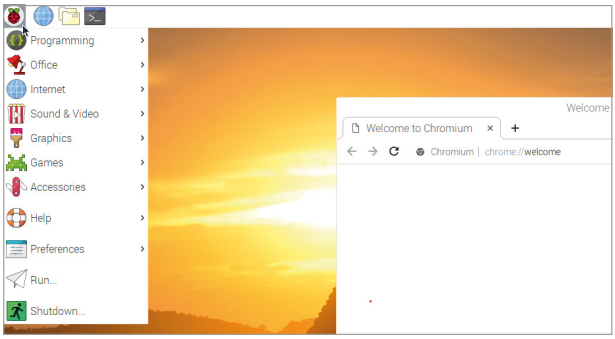
Figur 3-9: Den digitale kalenderen

Ved siden av klokken ser du et høyttalerikon (**H**). Klikk på dette ikonet med venstre museknapp for å justere lydvolumet til Raspberry Pi. Du kan høyreklikke på ikonet for å velge hvilken utgang Raspberry Pi skal bruke. Ved siden av dette ikonet ser du et nettverksikon (**G**). Hvis du er tilkoblet et trådløst nettverk, vises signalstyrken som flere stolper. Hvis du er tilkoblet et kablet nettverk, ser du bare to piler. Når du klikker på nettverksikonet, vises en liste over trådløse nettverk i nærheten (**Figur 3-10**). Når du klikker på Bluetooth-ikonet (**F**) ved siden av dette igjen, kan du koble til en Bluetooth-enhet i nærheten.



Figur 3-10: Viser en liste over trådløse nettverk i nærheten

Den venstre delen av menylinjen består av et *velgerfelt* (**J**), der du finner installerte programmer i tillegg til Raspberry Pi OS. Noen av disse programmene vises som snarveisikoner, mens andre er skjult i menyen. Du åpner menyen ved å klikke på bringebærikonet (**K**) helt til venstre på linjen (**Figur 3-11**, på neste side).

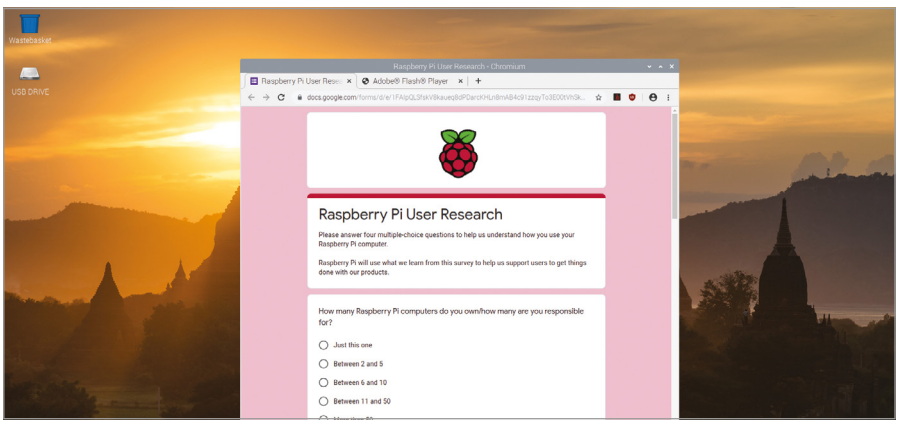


▲ **Figur 3-11:** Raspberry Pi OS-menyen

Programmene i menyen er delt inn i kategorier, med navn som forteller deg hva du kan forvente. Kategorien Utvikling inneholder for eksempel programvare som er utviklet for å hjelpe deg med å skrive dine egne programmer – som forklart i **Kapittel 4: Programmere med Scratch**. Og med kategorien Spill kan du glemme alt annet mens du spiller. Ikke alle programmene beskrives i denne veiledningen. Du kan gjerne eksperimentere med dem for å lære mer.

Nettleseren Chromium

Når du skal øve deg på å bruke Raspberry Pi, kan du begynne med å laste inn Chromium-nettleseren. Klikk på bringebærikonet øverst til venstre på skjermen for å åpne menyen, pek på Internett-kategori med musepekeren, og klikk deretter på Chromium-nettleseren for å laste den inn (**Figur 3-12**).



▲ **Figur 3-12:** Nettleseren Chromium

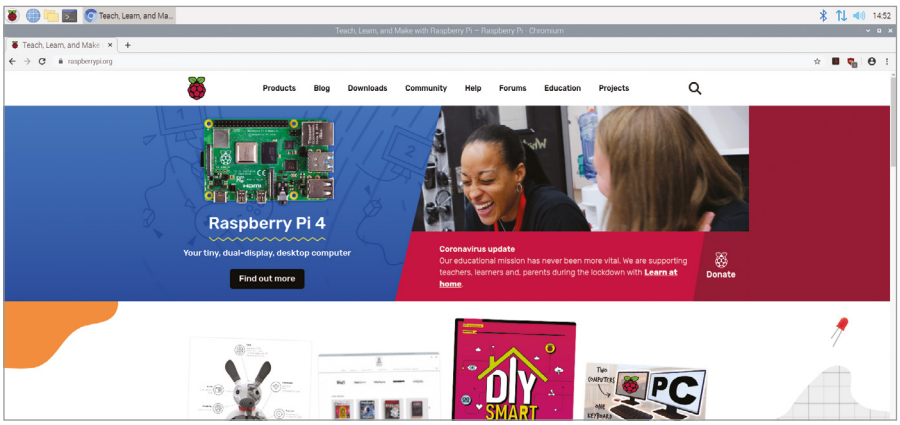
Hvis du har brukt nettleseren Google Chrome på en annen datamaskin, virker Chromium straks kjent. Med Chromium kan du besøke nettsteder, spille av videoer, prøve nye spill og til og med kommunisere med andre verden rundt i forumer og på chattesider.

Når du starter Chromium, bør du maksimere vinduet så det tar opp mer av skjermen. Finn de tre ikonene øverst til høyre på tittellinjen i Chromium-vinduet (**N**) og klikk på det midtre opp-pilikonet (**P**). Dette er *Maksimer*-knappen, som får et vindu til å fylle hele skjermen. Til venstre for Maksimer finner du *Mnimer*-knappen (**O**), som skjuler et vindu til du klikker på det i oppgavelinjen øverst på skjermen. Krysset til høyre for Maksimer står for *Lukk* (**Q**) og gjør akkurat det du forventer, dvs. lukker vinduet.

LUKKE OG LAGRE

Det er en dårlig idé å lukke et vindu før du har lagret arbeidet ditt. Noen programmer advarer deg og be deg om å lagre når du klikker på lukkeknappen. Andre programmer gjør det ikke.

Klikk i adresselinjen øverst i Chromium-vinduet – den store hvite linjen med et forstørrelsesglass på venstre side – og skriv **www.raspberrypi.org**. Deretter trykker du på **ENTER**-tasten på tastaturet. Nettstedet Raspberry Pi lastes inn (**Figur 3-13**). Du kan også skrive inn søk i adressefeltet. Du kan søke etter «Raspberry Pi», «Raspberry Pi OS» eller «Databehandling innen utdanning»..



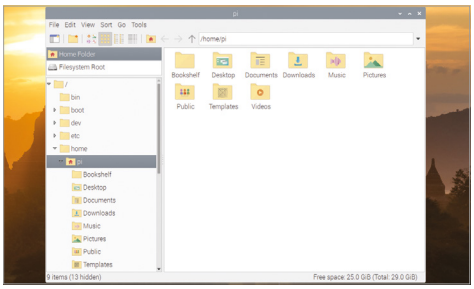
▲ **Figur 3-13:** Laste inn Raspberry Pi-nettstedet i Chromium

Den første gangen du laster inn Chromium, åpnes eventuelt flere *faner* øverst i vinduet. Du kan bytte til en annen fane ved å klikke på den. Hvis du vil lukke en fane uten å lukke Chromium, klikker du på krysset til høyre på fanen du vil lukke. Det kan være praktisk å ha flere nettsteder åpne uten å måtte veksle mellom flere Chromium-vinduer. Da kan du åpne en ny fane ved å klikke på faneknappen til høyre for den siste fanen i listen, eller ved å holde inne **CTRL**-tasten på tastaturet og trykke på **T**-tasten før du slipper opp **CTRL**.

Når du er ferdig med Chromium, klikker du på lukkeknappen øverst til høyre i vinduet.

File Manager

Filer du lagrer, f.eks. programmer, videoer og bilder, plasseres alle i *hjemmekatalogen*. Hvis du vil se hjemmekatalogen, klikker du på bringebærikonet igjen for å åpne menyen. Pek på Tilbehør, og klikk deretter på File Manager for å laste den inn (**Figur 3-14**).



Figur 3-14: File Manager

Med File Manager kan du bla gjennom filene og mappene, også kjent som *kataloger*, på microSD-kortet til Raspberry Pi. Du får også tilgang til eventuelle flyttbare lagringsmedier, f.eks. USB-flash-enheter, som du kobler til USB-portene på Raspberry Pi. Den første gangen du åpner den, går den automatisk til hjemmekatalogen. Her finner du en rekke andre mapper, kjent som *underkataloger*, som er ordnet i kategorier (på samme måte som menyen). De viktigste underkatalogene er:

- **Bookshelf (Bokhylle):** Denne «bokyllen» inneholder digitale kopier av bøker, magasiner og andre publikasjoner fra Raspberry Pi Press, inkludert en kopi av denne *Begynnerveiledningen*. Du kan lese disse, og laste ned flere, med Bookshelf-programmet. Du finner det i Help-delen på menyen.

- **Desktop (Skrivebord):** Du ser denne mappen den første gangen du laster inn Raspberry Pi OS. Hvis du lagrer en fil her, vises den på skrivebordet. Der kan du enkelt finne den og laste den inn.

- **Documents:** Her i lagres mesteparten av filene du oppretter, fra noveller til oppskrifter.

- **Downloads (Nedlastinger):** Når du laster ned en fil fra Internett ved hjelp av Chromium, lagres den automatisk i Downloads.

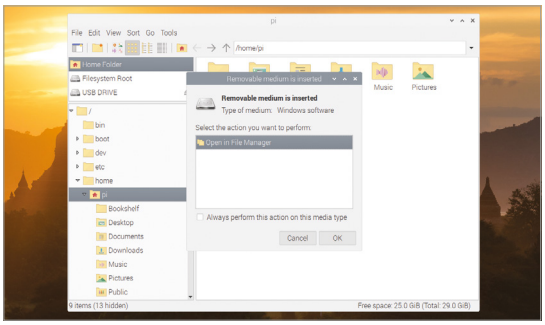
- **Music (Musikk):** All musikk du lager eller oppbevarer på Raspberry Pi, kan lagres her.

- **Pictures (Bilder):** Denne mappen er spesielt beregnet på bilder, mer teknisk kjent som *bildefiler*.

- **Public (Offentlig):** Mens de fleste av filene er private, vil alt du legger i Public være tilgjengelig for andre brukere av Raspberry Pi, selv om de har egne brukernavn og passord.

- **Templates (Maler):** Denne mappen inneholder maler, det vil si tomme dokumenter med forhåndsdefinerte oppsett eller strukturer. Du kan ha opprettet malene selv, eller de er installert av programmene dine.
- **Videos (Videoer):** En mappe for videoer. Det er det første stedet som sjekkes av programmer som spiller av videoer.

File Manager-vinduet er delt inn i to ruter. Den venstre ruten viser katalogene på Raspberry Pi. Den høyre truten viser filene og underkatalogene til katalogen som er valgt i venstre rute. Hvis du kobler et flyttbart lagringsmedium til USB-porten på Raspberry Pi, åpnes et vindu som spør om du vil åpne det i File Manager **Figur 3-15**). Klikk på OK for å se filene og katalogene på mediet.

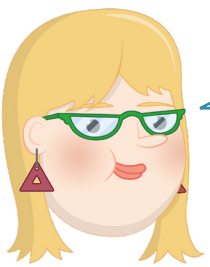


Figur 3-15: Sette inn en flyttbar lagringsenhet

Du kan enkelt kopiere filer mellom microSD-kortet på Raspberry Pi og en flyttbar enhet. Åpne både hjemmekatalogen og den flyttbare enheten i egne vinduer i File Manager. Pek på filen du vil kopiere, og klikk og hold inne venstre museknapp mens du drar musepekeren til det andre vinduet. Deretter slipper du opp museknappen (**Figur 3-16**, på neste side). Denne prosessen kalles å *dra og slippe*.

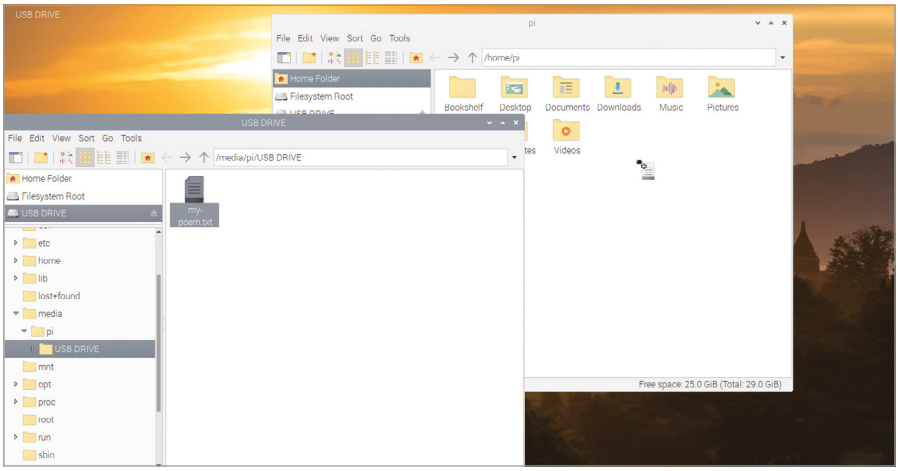
Du kan eventuelt også klikke en gang på filen, klikke på Edit-menyen (Rediger-menyen) og klikke på Copy (Kopier). Deretter klikker du i det andre vinduet, klikker på Edit-menyen og til slutt på Paste (Lim inn).

Alternativet Cut (Klipp ut) er også tilgjengelig i Edit-menyen. Det fungerer på same måte, bortsett fra at det sletter filen fra den opprinnelige plasseringen når kopien er laget. Begge alternativene kan også utføres med hurtigtastene **CTRL+C** (kopier) eller **CTRL+X** (klipp ut), og deretter **CTRL+V** (lim inn).



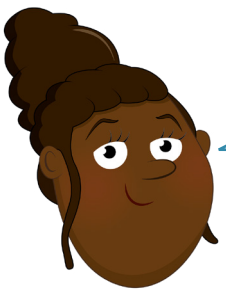
HURTIGTASTER

Når du ser en hurtigtast som **CTRL+C**, betyr det at du må holde inne den første tasten på tastaturet (**CTRL**), trykke på den andre tasten (**C**) og deretter slippe opp begge tastene.



▲ **Figur 3-16:** Dra og slippe en fil

Når du er ferdig med å eksperimentere, lukker du File Manager ved å klikke på Lukk øverst til venstre i vinduet. Hvis flere vinduer er åpne, kan du lukke alle vinduene. Hvis du har koblet en flyttbar lagringsenhet til Raspberry Pi, må du løse den ut ved å klikke på utløserknappen øverst til høyre på skjermen. Deretter finner du enheten i listen og klikke på den før du kobler den fra.



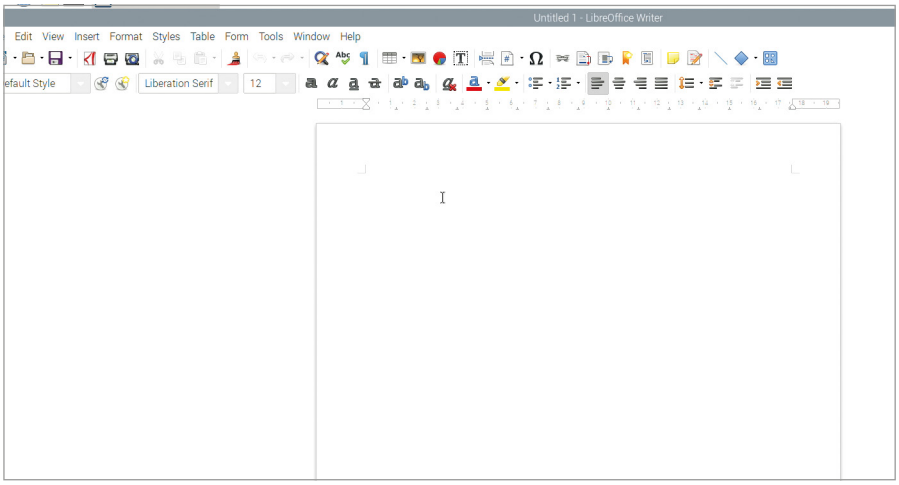
LØSE UT ENHETER

Bruk alltid utløserknappen før du kobler fra en ekstern lagringsenhet. Hvis du ikke gjør det, kan filene skades og gjøres ubrukelige.

Produktivitetspakken LibreOffice

Hvis du vil se mer om hva Raspberry Pi kan gjøre, klikker du på bringebærikonet, peker du på Kontorstøtte og klikker på LibreOffice Writer. Dette laster inn tekstbehandleren i LibreOffice (**Figur 3-17**), en populær *produktivitetspakke*. Hvis du har brukt Microsoft Office eller Google Docs, har du brukt en produktivitetspakke.

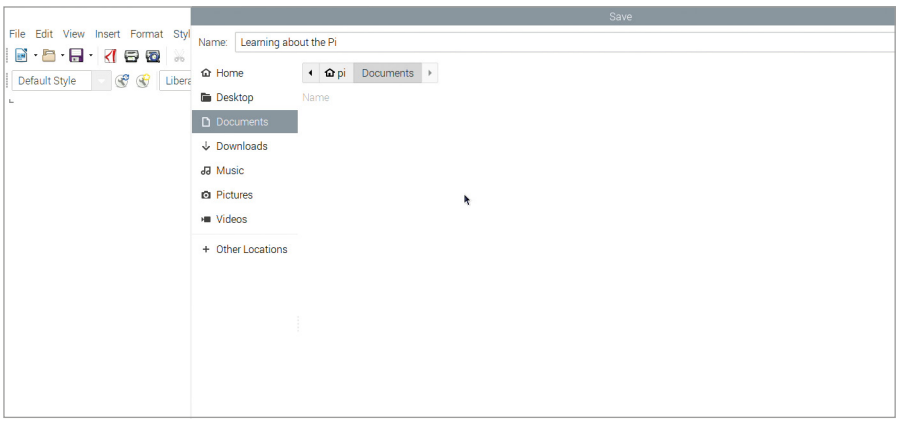
Merk: LibreOffice er kanskje ikke installert som standard på alle Raspberry Pi OS-bilder. Hvis ikke bruker du det anbefalte programvareverktøyet (se side 49) for å installere det.



Figur 3-17: Programmet LibreOffice Writer

Med et tekstbehandlingsprogram kan du ikke bare skrive dokumenter. Du kan også formatere dem på smarte måter. Du kan endre skriftstil, farge, størrelse, legge til effekter og til og med sette inn bilder, diagrammer, tabeller og annet innhold. Et tekstbehandlingsprogram lar deg også sjekke om arbeidet ditt inneholder feil. Det fremhever stave- og grammatikkfeil i henholdsvis rødt og grønt mens du skriver.

Begynn med å skrive et avsnitt om det du har lært om Raspberry Pi og den medfølgende programvaren så langt. Eksperimenter med de forskjellige ikonene øverst i vinduet for å se hva de gjør. Se om du kan skrive med større skrift og endre skriftfarge. Hvis du ikke er sikker på hvordan du gjør dette, kan du bare holde musepekeren over et ikon for å se et verktøytips som forteller deg hva ikonet gjør. Når du er fornøyd, klikker du på File-menyen og Save for å lagre arbeidet (**Figur 3-18**). Gi dokumentet et navn og klikk på Save-knappen.



Figur 3-18: Lagre et dokument

LAGRE ARBEIDET

Gjør det til en vane å lagre arbeidet, selv om du ikke er ferdig med det ennå. Det vil spare deg mange problemer hvis det blir strømbrudd eller hvis du avbrytes midt i arbeidet!

LibreOffice Writer er bare en del av den samlede produktivitetspakken til LibreOffice. Her er de andre delene, som du finner i samme Office-menykategori som LibreOffice Writer:

- **LibreOffice Base:** En database – et verktøy som brukes til å lagre informasjon og raskt søke etter og analysere denne informasjonen.
- **LibreOffice Calc**: Et regneark – et verktøy som brukes til å håndtere tall og opprette diagrammer og grafer.
- **LibreOffice Draw**: Et illustrasjonsprogram – et verktøy som brukes til å opprette bilder og diagrammer.
- **LibreOffice Impress**: Et presentasjonsprogram som brukes til å opprette lysbilder og kjøre lysbildefremvisninger.
- **LibreOffice Math**: Et formelredigeringsprogram – et verktøy som brukes til å lage matematiske formler i riktig format, som deretter kan brukes i andre dokumenter.

LibreOffice er også tilgjengelig for andre datamaskiner og operativsystemer. Hvis du liker å bruke denne pakken på Raspberry Pi, kan du laste den ned kostnadsfritt fra **libreoffice.org** og installere den på alle datamaskiner som kjører Microsoft Windows, Apple macOS eller Linux.

Hvis du vil vite mer om bruk av LibreOffice, klikker du på Help-menyen. Du kan lukke LibreOffice Writer ved å klikke på lukkeknappen øverst til høyre i vinduet.

FÅ HJELP

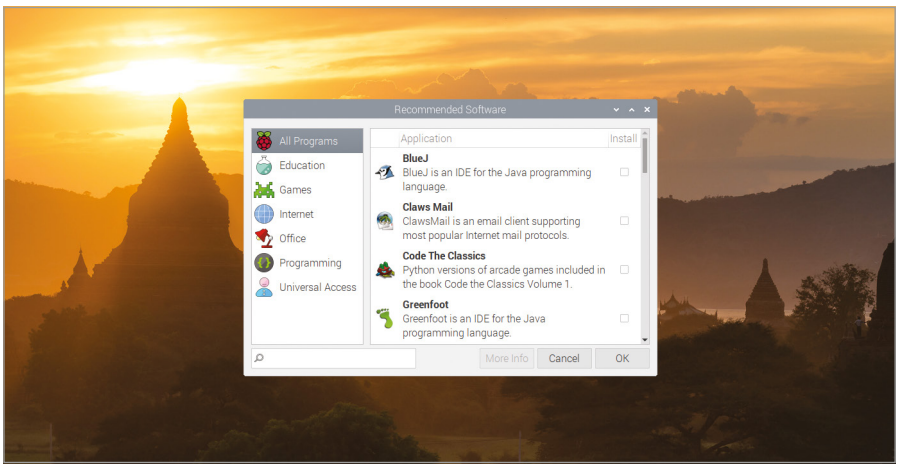
De fleste programmene har en hjelpemeny som inneholder alt fra informasjon om hva programmet gjør, til brukerveiledninger. Hvis du noen gang står fast eller føler deg overveldet av et program, kan du se Hjelp-menyen for å få tips.

Verktøyet Anbefalt programvare

Selv om Raspberry Pi OS er forhåndslastet med et bredt spekter av programvare, er den kompatibel med enda flere. Du finner et utvalg av det beste av denne programvaren i verktøyet Anbefalt programvare.

Merk: Anbefalt programvare krever tilkobling til Internett. Hvis Raspberry Pi er tilkoblet, klikker du på bringebærikonet, peker på Brukervalg og klikker på Anbefalt programvare. Verktøyet lastes inn, og deretter begynner nedlastingen av informasjon om tilgjengelig programvare.

Etter et par sekunder vises en liste over kompatible programvarepakker (**Figur 3-19**). På samme måte som programvaren i Raspberry-menyen er de ordnet i forskjellige kategorier. Klikk på en kategori i ruten til venstre for å se programvare fra den kategorien, eller klikk på Alle programmer for å se alt.

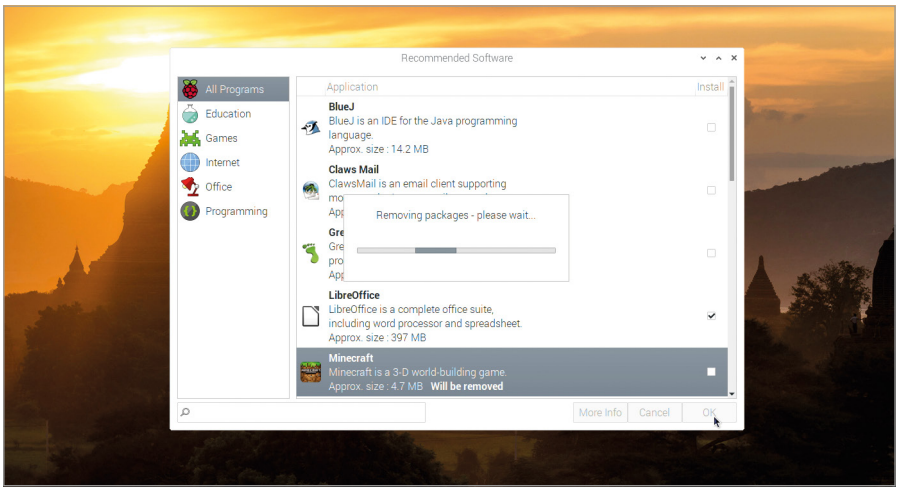


Figur 3-19: Verktøyet Anbefalt programvare

Hvis du ser en hake ved siden av en programvare, er den allerede installert på Raspberry Pi. Hvis den ikke har en hake, kan du klikke i avmerkingsboksen ved siden av programvaren for å merke den for installasjon. Du kan merke så mange programmer du vil, for å installere alle på en gang. Hvis du bruker et microSD-kort som er mindre enn det som er anbefalt, har du kanskje ikke plass til alle.

Du kan avinstallere programvare på samme måte. Finn et program som allerede har en hake i avmerkingsboksen, og klikk deretter på haken for å fjerne den. Hvis du har gjort en feil eller ombestemt deg, kan du bare å klikke på nytt for å merke programmet på nytt.

Når du er fornøyd med programvarevalget, klikker du på Bruk-knappen for å starte installasjons- eller avinstallasjonsprosessen (**Figur 3-20**, på neste side). Etter at du har lastet ned og installert ny programvare du har valgt, åpnes en dialogboks. Klikk på OK for å lukke verktøyet Anbefalt programvare.

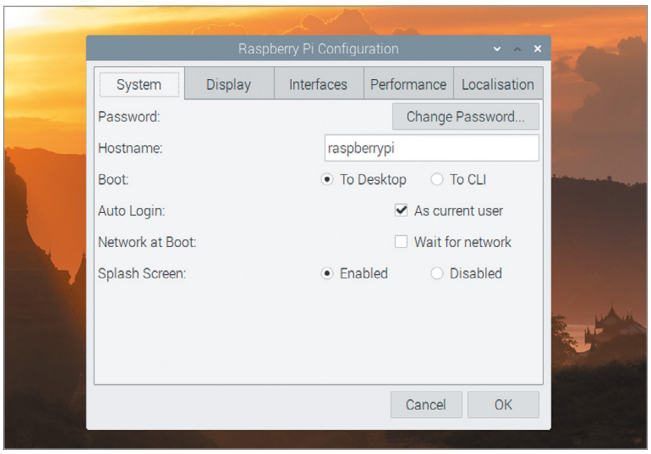


▲ **Figur 3-20:** Avinstallere programvare

Du finner Add/Remove Software, et annet verktøy for å installere eller avinstallere programvare, i kategorien Brukervalg på Raspberry Pi OS-menyen. Dette gir et bredere utvalg av programvare, men det er ikke undersøkt av Raspberry Pi Foundation.

Verktøyet Raspberry Pi Configuration

Det siste programmet du vil lære om i dette kapittelet, er verktøyet Raspberry Pi Configuration. Det ligner mye på velkomstveiviseren du brukte til å begynne med. Det brukes til å endre forskjellige innstillinger i Raspberry Pi OS. Klikk på bringebærikonet, pek på kategorien Brukervalg, og klikk deretter på Raspberry Pi Configuration for å laste inn verktøyet (**Figur 3-21**).



◀ **Figur 3-21:** Verktøyet Raspberry Pi Configuration

Verktøyet er delt inn i fem faner. Den første fanen er System. Du kan bruke den til å endre passordet til kontoen, angi et vertsnavn – navnet Raspberry Pi brukes på det lokale trådløse eller kablede nettverket – og endre en rekke andre innstillinger. De fleste av disse burde imidlertid ikke endres. Klikk på Display-fanen for å åpne neste kategori. Her kan du endre innstillingene for skjermvisning om nødvendig, slik at de passer til TV-en eller dataskjermen.

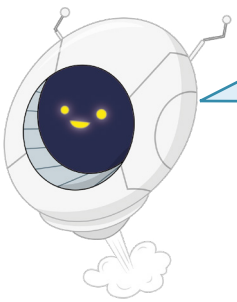
MER INFORMASJON

Du kan bruke denne korte oversikten til å bli bedre kjent med verktøyet. Du finner mer detaljert informasjon om hver enkelt innstilling i **Vedlegg E: Verktøyet Raspberry Pi Configuration**.

Fanen Interface har flere innstillinger, som alle er deaktivert som standard. Disse innstillingene bør bare endres hvis du legger til ny maskinvare, for eksempel kameramodulen for Raspberry Pi. Du bør bare endre dem hvis du blir bedt om å gjøre det av maskinvareprodusenten. Unntak fra denne regelen: SSH, som aktiverer et «Secure Shell» og lar deg logge på Raspberry Pi fra en annen datamaskin i nettverket ved hjelp av en SSH-klient. VNC, som aktiverer en «Virtual Network Computer» og lar deg se og kontrollere skrivebordet i Raspberry Pi OS fra en annen datamaskin i nettverket ved hjelp av en VNC-klient. Remote GPIO, som lar deg bruke Raspberry Pis GPIO-pinner (som du vil lære mer om i **Kapittel 6: Fysisk databehandling med Scratch og Python**) fra en annen datamaskin i nettverket.

Klikk på fanen Performance for å se den fjerde kategorien. Her kan du angi hvor mye minne som brukes av Raspberry Pis grafikkprosessor (GPU) og, for enkelte modeller, øke ytelsen til Raspberry Pi gjennom en prosess som kalles *overklokking*. Som nevnt tidligere er det best å la disse innstillingene være med mindre du vet at du må endre dem.

Til slutt klikker du på kategorien Localisation for å se den siste kategorien. Her kan du endre nasjonal innstilling, der du kan angi hvilket språk som brukes i Raspberry Pi OS og hvordan tall vises, endre tidssonen, endre tastaturoppsettet og angi landet ditt for Wi-Fi-formål. Foreløpig kan du bare klikke på Cancel (Avbryt) for å lukke verktøyet uten å gjøre endringer.



ADVARSEL!

Ulike land har forskjellige regler for hvilke frekvenser en Wi-Fi-radio kan bruke. Hvis du stiller inn Wi-Fi-landet i verktøyet Raspberry Pi Configuration til et annet land enn der du er, får det sannsynligvis problemer med å koble til nettverkene. Det kan til og med være ulovlig i henhold til lover om radiolisenser. Derfor bør du ikke gjøre det!

Slå av

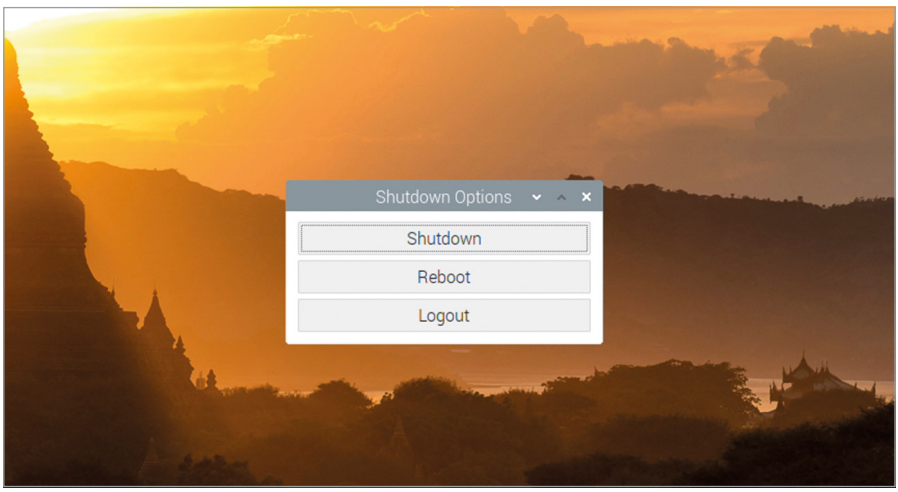
Nå som du har utforsket skrivebordet i Raspberry Pi OS, er det på tide å lære noe svært viktig: Hvordan du slår av Raspberry Pi på en sikker måte. Raspberry Pi, som alle andre datamaskiner, lagrer filene du jobber med, i et *flyktig minne*, det vil si et minne som tømmes når systemet slås av. Når du oppretter dokumenter, er det nok å lagre filene etter tur. Da overføres filen fra det flyktige minnet til et *ikke-flyktig minne*, det vil si microSD-kortet. På denne måten kan du være sikker på at ingenting går tapt.

Dokumentene du jobber med, er imidlertid ikke de eneste filene som er åpne. Når Raspberry Pi OS kjører, holdes automatisk en rekke filer åpne. Så hvis du trekker strømkabelen ut av Raspberry Pi mens disse filene er åpne, kan det føre til at operativsystemet blir skadet og må installeres på nytt.

For å forhindre dette må du huske å be Raspberry Pi OS om å lagre alle filene og gjøre seg klar til å slås av. Denne prosessen er kjent som å *skru av* (slå av) operativsystemet.

Klikk på bringebærikonet øverst til venstre på skrivebordet, og klikk deretter på Shutdown. Et vindu med tre alternativer vises (**Figur 3-22**): Skru av, Omstart og Logg av. Skru av er alternativet du vil bruke mest. Når du klikker på dette, lukker Raspberry Pi OS alle åpne programmer og filer og slår av Raspberry Pi. Når skjermen er svart, venter du et par sekunder til den blinkende grønne lampen på Raspberry Pi slukkes. Da er det trygt å slå av strømforsyningen.

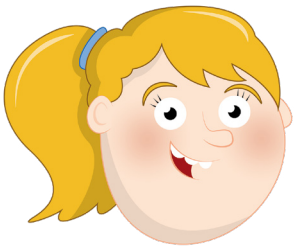
Når du skal slå på Raspberry Pi igjen, kobler du bare strømkablene fra og til igjen (i noen land kan du også slå strømmen av og på ved stikkontakten).



▲ **Figur 3-22:** Slå av Raspberry Pi

Omstart følger en lignende prosess som Skru av. Alt lukkes, men i stedet for å slå av Raspberry Pi, startes maskinen på nytt, nesten som om du velger Skru av og deretter kobler strømkabelen fra og til igjen. Du må bruke Omstart hvis du gjør visse endringer som krever at du starter operativsystemet på nytt, for eksempel når du installerer oppdateringer for kjerneprogramvaren, eller hvis et program har sluttet å fungerer, kjent som å *krasje*, og har ført til at Raspberry Pi OS ikke kan brukes.

Logg av er bare nyttig hvis du har mer enn én brukerkonto på Raspberry Pi. Alternativet lukker alle åpne programmer og tar deg til et påloggingsvindu der du blir bedt om å angi brukernavn og passord. Hvis klikker på Logg av ved en feiltagelse og ønsker å logge på igjen, skriver du bare 'pi' som brukernavn og passordet du valgte i velkomstveiviseren på begynnelsen av dette kapitlet.



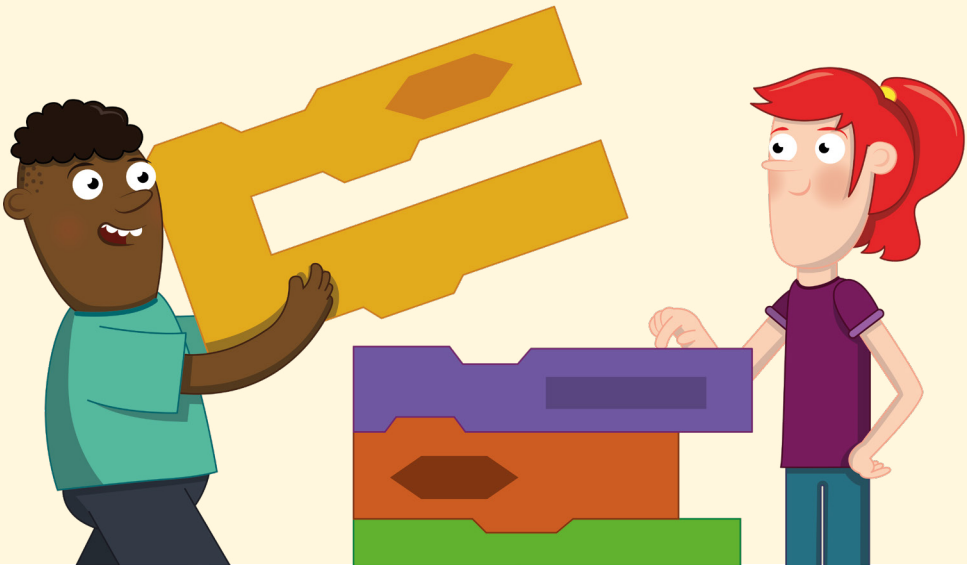
ADVARSEL!

Koble aldri strømkabelen fra en Raspberry Pi uten å slå den av først. Hvis du gjør det, kan du skade operativsystemet og miste alle filene du har opprettet eller lastet ned.

Kapittel 4

Programmering med Scratch 3

Finn ut hvordan du begynner å kode ved hjelp av Scratch, et klossbasert programmeringsspråk

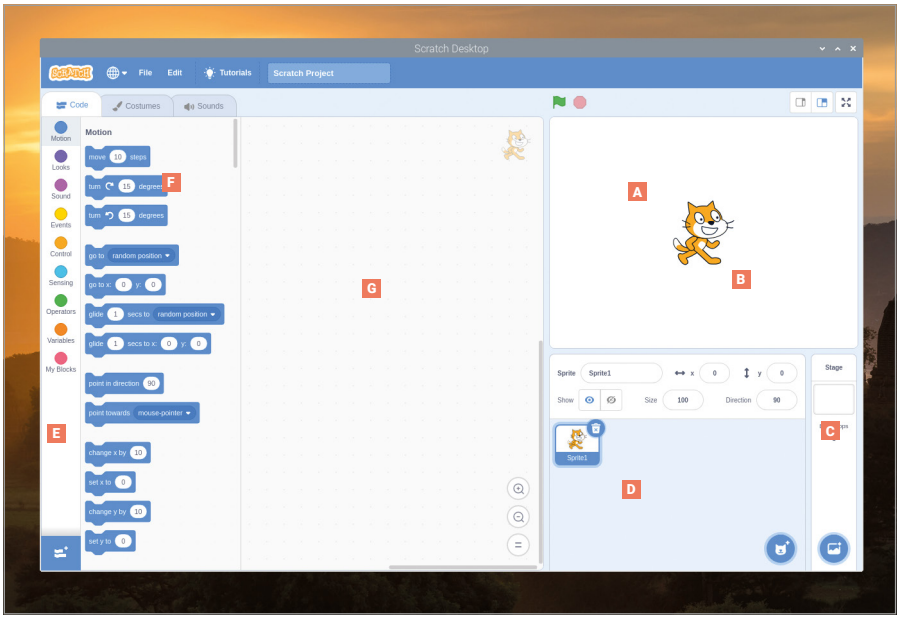


Når du bruker Raspberry Pi, bruker du ikke bare programvare som andre har laget. Du lager også din egen programvare basert på nesten alt fantasien kan trylle frem. Enten du har tidligere erfaring med å lage dine egne programmer, en prosess som kalles programmering eller koding, eller ikke, er Raspberry Pi en flott plattform for å utvikle og eksperimentere.

På Raspberry Pi er koding basert på Scratch, et visuelt programmeringsspråk som er utviklet av MIT (Massachusetts Institute of Technology). I tradisjonelle programmeringsspråk skriver du tekstbaserte instruksjoner som datamaskinen skal utføre, på samme måte som du skriver en oppskrift for å bake en kake. Med Scratch kan du bygge programmet steg for steg ved hjelp av klosser – forhåndsdefinerte deler av kode som er gjemt bak fargekodede puslespillbiter.

Scratch er et flott første språk for spirende kodere, både store og små, men la deg ikke lure av det enkle utseendet. Det er et kraftig og fullt funksjonelt programmeringsmiljø som kan brukes til å lage alt fra enkle spill og animasjoner til komplekse interaktive robotprosjekter.

Presentasjon av Scratch 3-grensesnittet



A **Sceneområde**: På samme måte som skuespillere i et skuespill beveger figurene seg rundt på scenen mens de kontrolleres av programmet ditt.

B **Figur**: Karakterene eller objektene du kontrollerer i et Scratch-program, kalles figurer og vises på scenen.

C **Scenekontroller**Du kan bruke scenekontrollene til å endre scenen, for eksempel ved å legge til dine egne bilder som bakgrunn.

D **Figurlister**: Alle figurer du har opprettet eller lastet inn i Scratch, vises i denne delen av vinduet.

E **Klosspalett**: Alle klosser som er tilgjengelige i programmet, vises i klosspaletten og vises i fargekodede kategorier.

SCRATCH-VERSJONER

Da veiledningen ble skrevet, var Raspberry Pi OS tilgjengelig med tre versjoner av Scratch: Scratch 1, 2 og 3, alt inkludert i programmeringsdelen av Raspberry Pi OS-menyen. Dette kapitlet er skrevet for Scratch 3. Legg merke til at Scratch 3 bare kan kjøres på Raspberry Pi 4. Hvis du vil bruke Scratch 2 i stedet, kan du ikke kjøre denne versjonen på Raspberry Pi Zero, Modell A, A+, B eller B+.

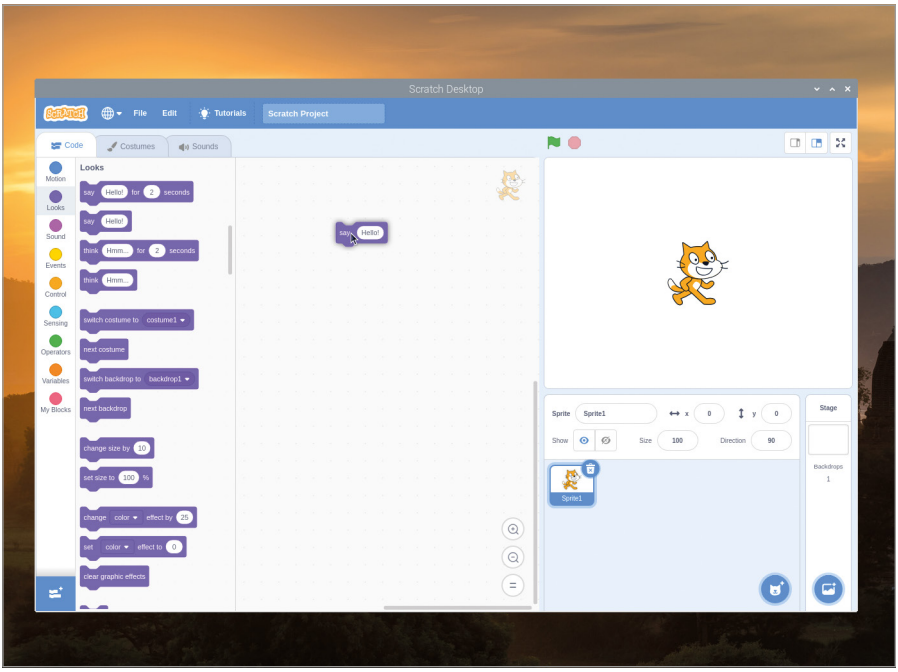
F **Klosser**: Klosser er forhåndsskrevne deler av programkode, som brukes til å bygge programmet trinnvis.

G **Kodeområde**: Her bygger du programmet ved å dra og slippe klosser fra klosspaletten for å danne skripter.

Ditt første Scratch-program: Hei, verden!

Scratch 3 lastes inn som alle andre programmer på Raspberry Pi: Klikk på bringebærikonet for å laste inn Raspberry Pi OS-menyen, flytt markøren til delen Programmering, og klikk deretter på Scratch 3. Etter et par sekunder lastes brukergrensesnittet Scratch 3.

I de fleste programmeringsspråk må du fortelle datamaskinen hva den skal gjøre ved å gi skriftlige instruksjoner, men Scratch er annerledes. Start med å klikke på Utseende-kategorien i klosspaletten. Du finner den øverst til venstre i Scratch-vinduet. Dette åpner klossene i den lilla kategorien. Finn klossen si Hei! og klikk og hold venstre museknapp over klossen. Dra den over til kodeområdet midt i Scratch-vinduet og slipp museknappen (**Figur 4-1**).

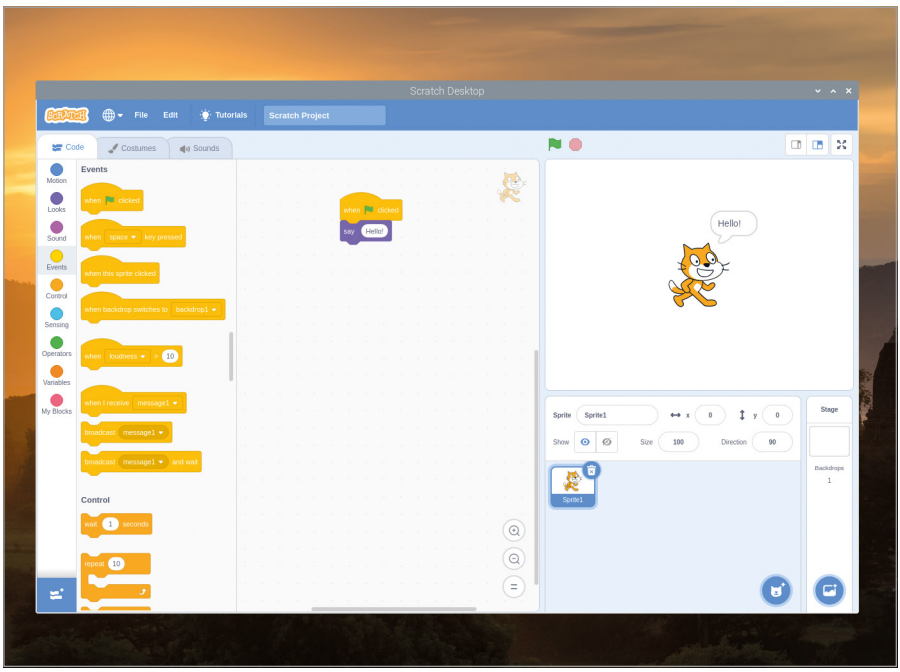


▲ **Figur 4-1:** Dra og slippe klossen i kodeområdet

Se på formen på klossen du nettopp har sluppet der. Den har en åpning øverst og takk nederst. I likhet med en puslespillbit angir dette at klossen skal ha noe over seg og under seg. I dette programmet kalles det som skal være over klossen, en *utløser*.

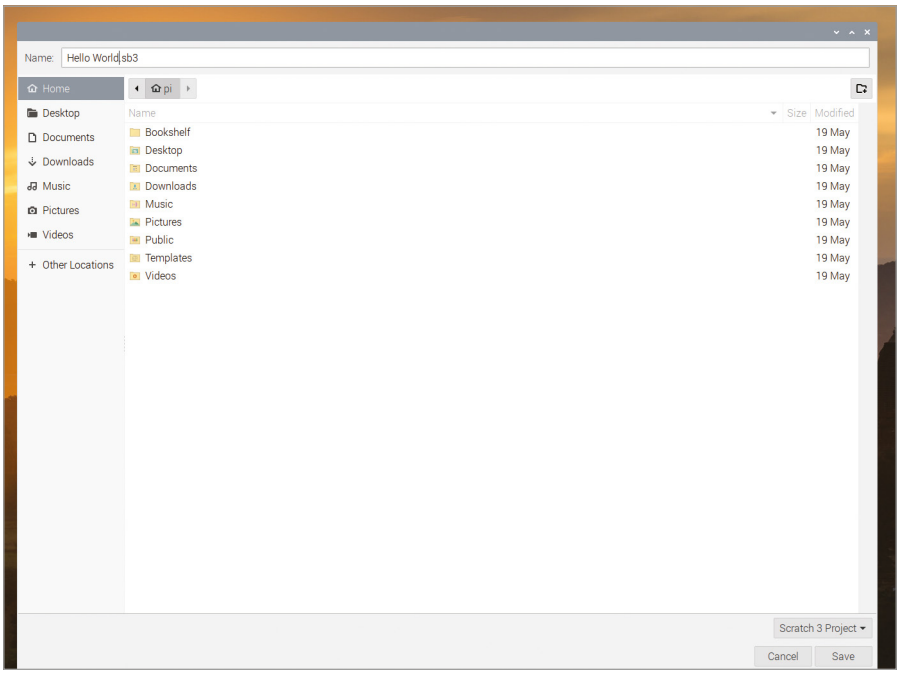
Klikk på den gullfargede kategorien Hendelser i klosspaletten. Deretter klikker du og drar når klikkes-klossen, som kalles en *hatt*-kloss, til kodeområdet. Plasser den slik at takken nederst på klossen står i åpningen øverst på klossen si Hei! Når du ser et hvitt omriss, slipper du museknappen. Du trenger ikke å være presis. Hvis den er nær nok, klikker klossen raskt på plass, som i et puslespill. Hvis den ikke gjør det, klikker du og holder klossen og justerer den til den klikker på plass.

Programmet er nå komplett. Når du vil starte programmet, kjent som å *kjøre* det, klikker du på det grønne flaggikonet øverst til venstre i sceneområdet. Hvis alt har gått bra, vil kattefiguren på scenen hilse deg med et muntert «Hei!» (**Figur 4-2**) – det første programmet ditt er vellykket.

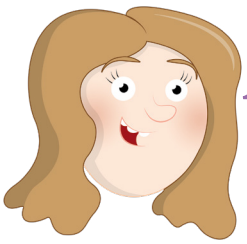


Figur 4-2: Klikk på det grønne flagget over scenen, og katten sier «Hei»

Gi programmet et navn og lagre det før du går videre. Klikk på Fil-menyen, og deretter på «Lagre på datamaskinen». Skriv inn et navn og klikk på Lagre-knappen (**Figur 4-3**)..



Figur 4-3: Lagre programmet med et navn som er lett å huske



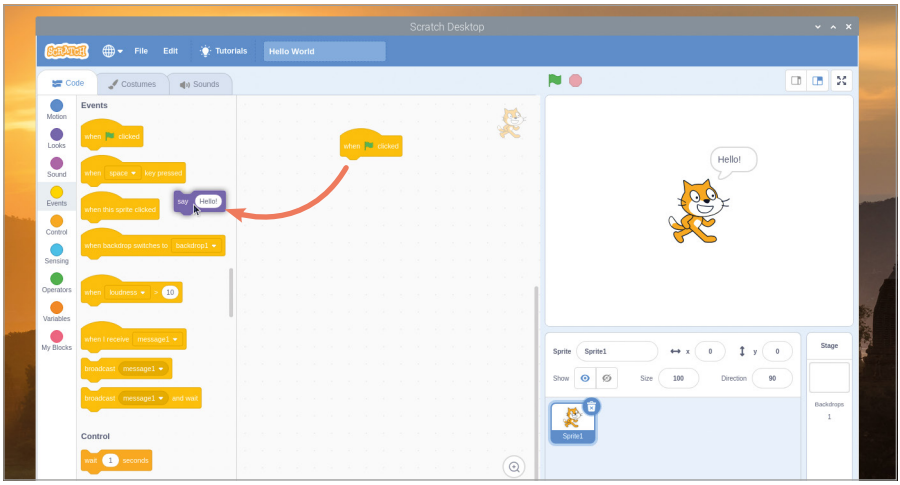
HVA KAN DEN SI?

Noen av klossene i Scratch kan endres. Prøv å klikke på ordet Hei! og skrive inn noe annet. Deretter klikker du på det grønne flagget igjen. Hva skjer på scenen?

Neste trinn: sekvensering

Programmet har to klosser, men bare en virkelig instruksjon: Si «Hei!» hver gang noen klikker på flagget, og programmet begynner å kjøre. Hvis du vil gjøre mer, må du gjøre deg kjent med *sekvensering*. Dataprogrammer er ganske enkelt en liste over instruksjoner, akkurat som en oppskrift. Hver instruksjon er en fortsettelse av den siste i en logisk progresjon som kalles en *lineær sekvens*.

Begynn med å klikke og dra si Hei! klossen fra kodeområdet tilbake til klosspaletten (**Figur 4-4**). Dette sletter klossen og fjerner den fra programmet slik at bare utløserklossen gjenstår, når klikkes.



Figur 4-4: Når du vil slette en kloss, bare drar du den ut av kodeområdet

Klikk på kategorien Bevegelse i klosspaletten. Deretter klikker du og drar klossen **gå 10 steg** til den klikker på plass under utløserklossen i kodeområdet. Som navnet antyder instruerer dette figuren, i dette tilfellet katten, om å bevege seg fremover i angitt antall trinn.

Legg til flere instruksjoner i programmet for å lage en sekvens. Klikk på Lyd-palletten, som har rosa fargekode. Deretter klikker du og drar klossen **spill lyden Mjau til den er ferdig** til den klikker på plass under klossen **gå 10 steg**. Fortsett: Klikk på kategorien Bevegelse igjen, og dra en annen kloss med teksten **gå 10 steg** til den ligger under lydklossen. Denne gangen klikker du imidlertid på 10 for å merke tallet. Deretter skriver du -10 for å lage en kloss med instruksen **gå -10 steg**.

Klikk på det grønne flagget over scenen for å kjøre programmet. Du ser at katten beveger seg mot høyre og lager en mjauelyd. Sørg for at du har koblet til høyttalere eller hodetelefoner for å høre lyden. Deretter går du tilbake til begynnelsen igjen. Klikk på flagget en gang til, og katten gjentar handlingene.

Gratulerer! Du har laget en sekvens av instruksjoner som Scratch kjører etter tur – fra øverst til nederst. Scratch kjører bare en instruksjon om gangen fra sekvensen, men det skjer veldig raskt. Prøv å slette klossen spill lyden Mjau til den er ferdig ved å klikke på og dra den nederste klossen, gå -10 steg, for å løsne den og dra klossen spill lyden Mjau til den er ferdig til klosspaletten. Deretter erstatter du den med den enklere klossen, start lyden Mjau, og drar klossen gå -10 steg tilbake til bunnen av programmet.

Klikk på det grønne flagget for å kjøre programmet igjen. Nå ser det ut som om kattefiguren ikke beveger på seg. Figuren beveger faktisk på seg, men den beveger seg så raskt tilbake igjen at den ser ut til å stå stille. Dette skjer fordi klossen start lyden Mjau ikke venter til lyden er ferdig avspilt før den fortsetter til neste trinn. Raspberry Pi «tenker» så raskt at neste instruksjon kjøres før du kan se at kattefiguren beveger seg. Du kan løse dette problemet ved å bruke klossen spill lyden Mjau til den er ferdig, men du kan også bruke denne metoden: Klikk på den lyseoransje kategorien Styring i klosspaletten. Deretter klikker du og drar klossen vent 1 sekund til den står mellom klossen start lyden Mjau og den nederste klossen gå -10 steg.

Klikk på det grønne flagget for å kjøre programmet en siste gang. Nå vil du se at kattefiguren venter et sekund etter at den har beveget seg til høyre, før den beveger seg til venstre igjen. Dette er kjent som en *forsinkelse* og er nøkkelen til å kontrollere hvor lang tid det tar å kjøre instruksjonssekvensen.

UTFORDRING: LEGG TIL FLERE STEG

Prøv å legge til flere steg i sekvensen og endre verdiene i de eksisterende stegene. Hva skjer når antall steg i en bevegelseskloss ikke samsvarer med antall steg i en annen? Hva skjer hvis du prøver å spille av lyd mens en annen lyd fremdeles spilles av?

Fullføre løkken

Sekvensen du har laget så langt, kjøres bare én gang. Du klikker på det grønne flagget og kattefiguren beveger seg og mjauer. Deretter stopper programmet til du klikker på det grønne flagget igjen. Det trenger ikke å stoppe fordi Scratch inkluderer en styringskloss som kalles en *løkke*.

Klikk på kategorien Styring i klosspaletten og finn klossen gjenta for alltid. Klikk og dra den inn i kodeområdet, og slipp den detter under klossen når klikkes og over den første klossen med instruksjonen gå 10 steg.

Legg merke til hvordan den C-formede klossen med teksten «gjenta for alltid» automatisk vokser til den omgir de andre klossene i sekvensen. Klikk på det grønne flagget, så ser du raskt hva klossen gjenta for alltid gjør: I stedet for at programmet kjøres én gang og fullføres, vil det kjøre igjen og igjen – bokstavelig talt for alltid. Innen programmering er dette kjent som en *uendelig løkke* – det vil si en løkke som aldri slutter.

Hvis lyden av konstant mjauing blir litt for irriterende, klikker du på den røde åttekanten ved siden av det grønne flagget over sceneområdet for å stoppe programmet. Hvis du vil

endre løkketypen, klikker og drar du den første klossen av typen gå 10 steg for å dra den og klossene under den ut av klossen gjenta for alltid. Deretter slipper du dem under klossen når klikkes. Klikk og dra klossen gjenta for alltid til klosspaletten for å slette den. Deretter klikker og drar du klossen gjenta 10 under klossen når klikkes slik at den går rundt de andre klossene.

Klikk på det grønne flagget for å kjøre det nye programmet. Først ser det ut til å gjøre det samme som den opprinnelige versjonen: Instruksjonssekvensen gjentas om og om igjen. I stedet for å fortsette uendelig fullføres derimot løkken etter ti repetisjoner. Dette er kjent som en *endelig løkke*: Du definerer når den skal fullføres. Løkker er kraftige verktøy, og de fleste programmer – spesielt spill og registreringsprogrammer – bruker både uendelige og endelige løkker.

HVA SKJER NÅ?

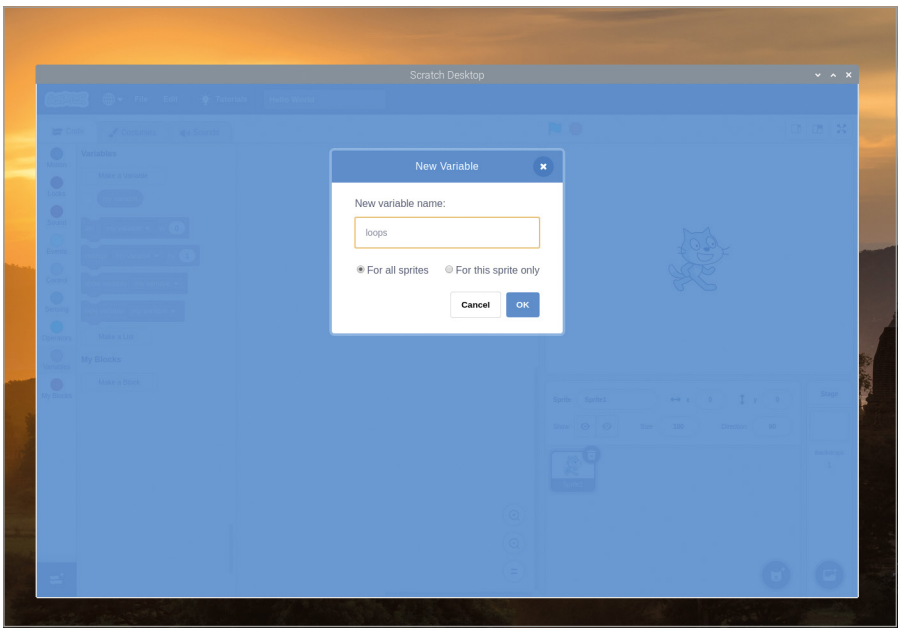
Hva skjer hvis du endrer tallet i løkkeklossen for å gjøre det større? Hva skjer hvis det er mindre? Hva skjer hvis du setter tallet 0 i løkkeklossen?

Variabler og betingelser

De siste begrepene du må forstå før du begynner å kode Scratch-programmer på alvor, er nært beslektede: *variabler* og *betingelser*. En variabel er, som navnet antyder, en verdi som kan variere – med andre ord endres – over tid og styres av programmet. En variabel har to hovedegenskaper, det vil si navnet og verdien den lagrer. Verdien trenger ikke å være et tall. Den kan være tall, tekst, sann eller usann eller helt tom – kjent som en *nullverdi*.

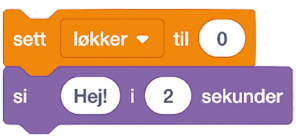
Variabler er kraftige verktøy. Tenk på tingene du må spore i et spill: helsetilstanden til en karakter, hastigheten til et bevegelig objekt, nivået som spilles for øyeblikket, og poengsummen. Alle disse spores som variabler.

Først klikker du på Fil-menyen og lagrer det eksisterende programmet ved å klikke på «Lagre på datamaskinen». Hvis du har lagret programmet tidligere, blir du spurt om du vil overskrive det. Det erstatter den gamle lagrede kopien med den nye oppdaterte versjonen. Deretter klikker du på Fil og så på Nytt for å starte et nytt, tomt prosjekt (klikk på OK når du blir spurt om du vil erstatte innholdet i det gjeldende prosjektet). Klikk på den mørkeoransje kategorien Variabler i klosspaletten, og deretter på knappen «Lag en variabel». Skriv inn «løkker» som variabelnavn (**Figur 4-5**), og klikk deretter på OK for å vise en serie av klosser i klosspaletten.



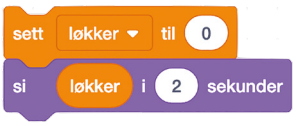
▲ **Figur 4-5:** Gi den nye variabelen et navn

Klikk og dra klossen sett løkker til 0 til kodeområdet. Dette forteller programmet at det må *initialisere* variabelen med verdien 0. Deretter klikker du på kategorien Utseende i klosspaletten og drar klossen si Hei! i 2 sekunder til den ligger under klossen sett løkker til 0.

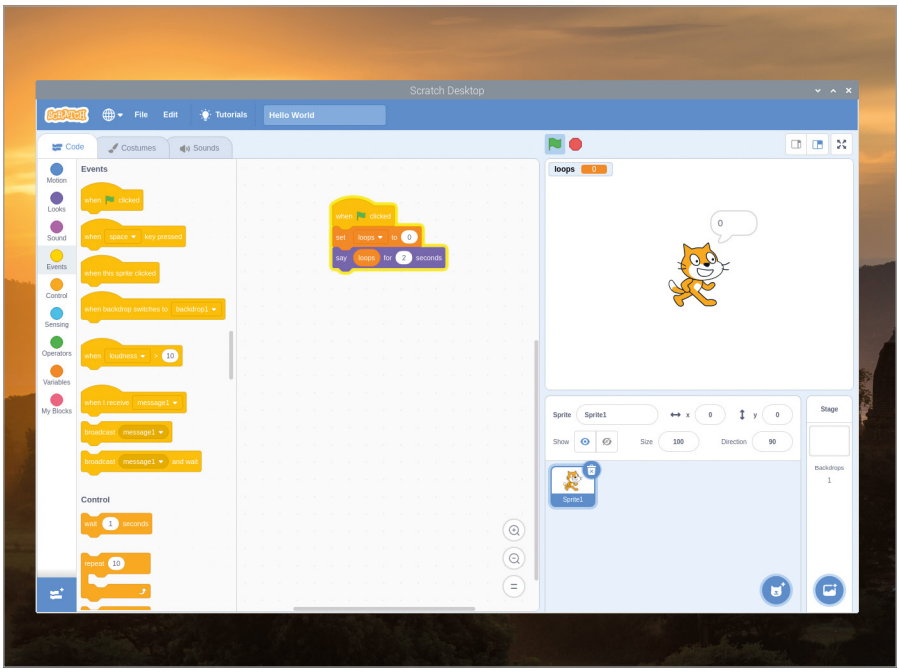


Som du så tidligere, instruerer klosser av typen si Hei! kattefiguren til å si det som står skrevet i dem. I stedet for å skrive meldingen i klossen selv, kan du heller bruke en variabel. Klikk på kategorien Variabler i klosspaletten igjen. Deretter klikker og drar du den avrundede løkker -klossen, som kalles en *rapporteringskloss* og står øverst på listen ved siden av en

avmerkingsboks, til den ligger over ordet «Hei!» i [si Hei! i 2 sekunder]-klossen. Dette lager en ny, kombinert kloss: [si løkker i 2 sekunder].



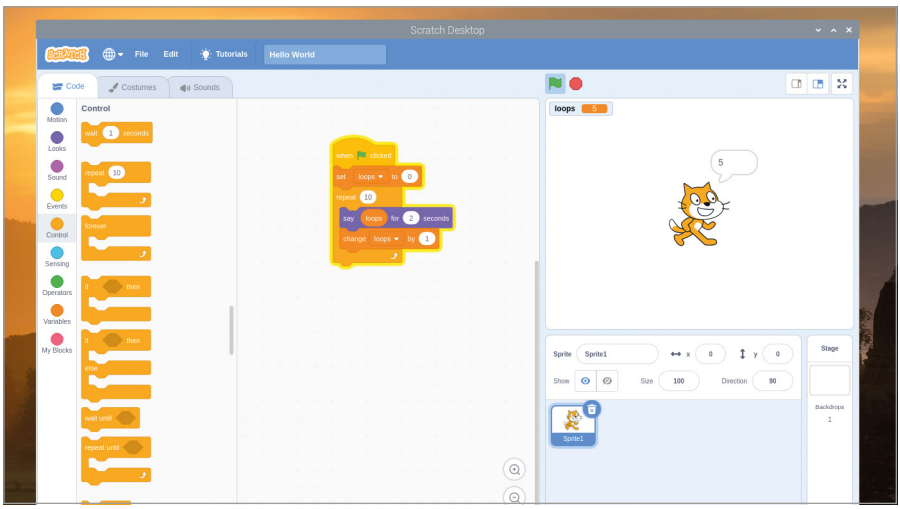
Klikk på kategorien Hendelser i klosspaletten. Deretter klikker og drar du klossen [når klikkes] for å plassere den øverst i klossekvensen. Klikk på det grønne flagget over sceneområdet, og du vil se kattefiguren sier «0» (**Figur 4-6**) – verdien du har gitt variabelen «løkker».



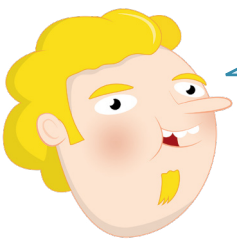
▲ **Figur 4-6:** Denne gangen vil katten si verdien i variabelen

Variabler kan imidlertid endres. Klikk på kategorien Variabler i klosspaletten. Deretter klikker og drar du klossen [endre løkker med 1] til den ligger nederst i sekvensen. Klikk på kategorien Styring. Deretter klikker og drar du en kloss av typen [gjenta 10] og slipper den slik at den starter rett under klossen [sett løkker til 0] og somslutter de gjenværende klossene i sekvensen.

Klikk på det grønne flagget igjen. Denne gangen ser du katten telle oppover fra 0 til 9. Dette fungerer fordi programmet nå endrer eller *modifiserer* selve variabelen. Hver gang løkken kjøres, legger programmet til 1 i verdien i «løkke»-variabelen(**Figur 4-7**).



▲ **Figur 4-7:** Takket være løkken teller katten nå oppover

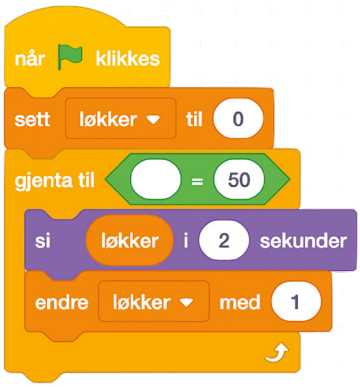


TELLE FRA NULL

Selv om løkken du har opprettet, kjøres ti ganger, teller kattefiguren bare opptil ni. Det skjer fordi vi har startet variabelen vår med verdien null. Hvis du teller fra null til ni, utgjør det ti tall, så programmet stopper før katten sier 10. Du kan endre dette ved å sette variabelens opprinnelige verdi til 1 i stedet for 0.

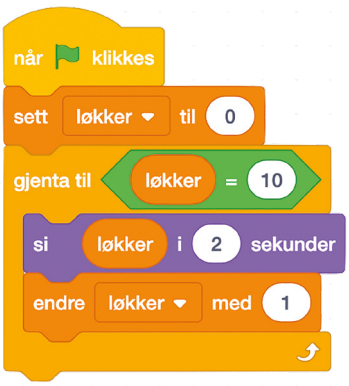
Du kan gjøre mer med en variabel enn å endre den. Klikk og dra klossen si løkker i 2 sekunder for å løsne den fra klossen gjenta 10 og slippe den under klossen gjenta 10. Klikk og dra klossen gjenta 10 til klosspaletten for å slette den. Deretter erstatter du den med en kloss av typen gjenta til. Kontroller at klossen er klikket på plass under klossen sett løkker til 0 og omgir begge de andre klossene i sekvensen. Klikk på den grønne kategorien Operatorer i klosspaletten. Deretter klikker og drar du den diamantformede 

-klossen og slipper den i det tilsvarende diamantformede hullet i klossen gjenta til.



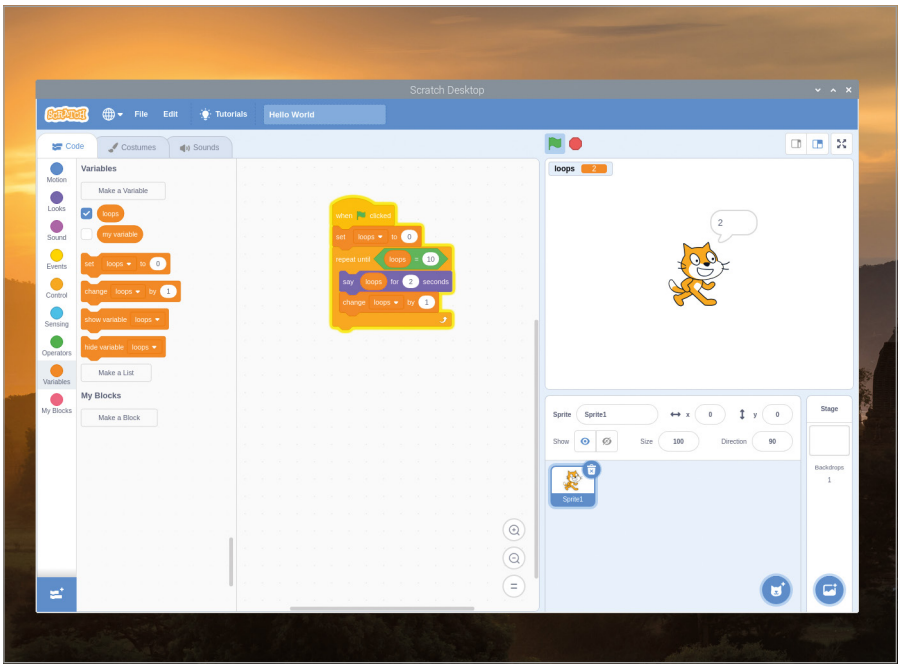
Denne Operator-klossen lar deg sammenligne to verdier, inkludert variabler. Klikk på kategorien Variabler og dra løkker rapporteringsklossen inn i det tomme rommet i 

Operator-klossen. Deretter klikker du i området med tallet 50 og skriver inn 10 i stedet.



Klikk på det grønne flagget over sceneområdet, og du vil se at programmet fungerer på samme måte som før. Kattefiguren teller fra 0 til 9 (**Figur 4-8**) og programmet stopper. Det er

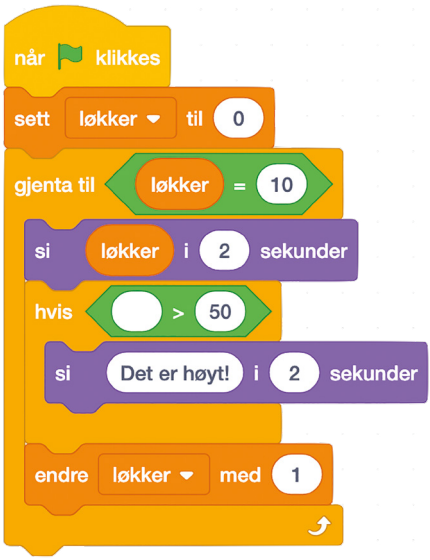
fordi klossen gjenta til fungerer på nøyaktig samme måte som klossen gjenta 10, men i stedet for å telle antall løkker i seg selv, sammenligner den verdien i «løkker»-variabelen med verdien du har skrevet til høyre for klossen. Når «løkker»-variabelen når 10, stopper programmet.



▲ **Figur 4-8:** Bruke klossen «gjenta til» med en komparativ operator

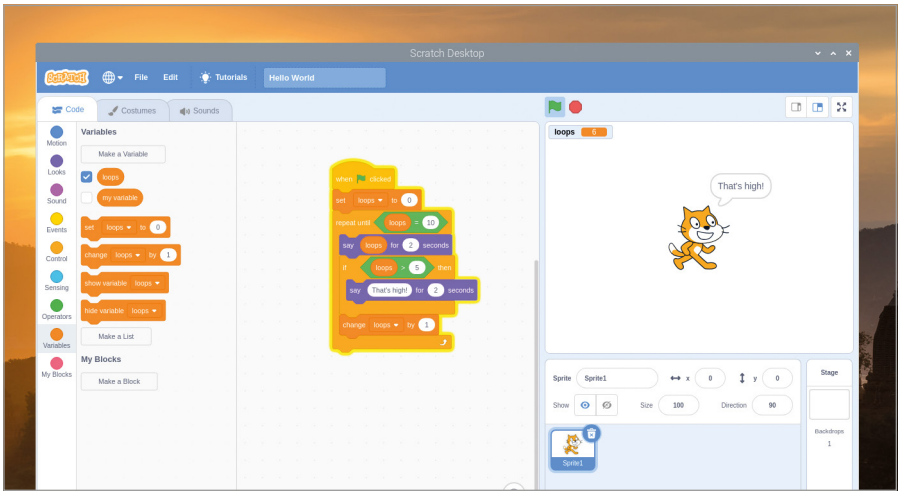
Dette er kjent som en *komparativ operator*. Den sammenligner to verdier. Klikk på kategorien Operatører i klosspaletten, og finn de to andre diamantformede klossene over og under klossen med likhetstegnet: =. Dette er også komparative operatorer. Tegnet «<» sammenligner to verdier og utløses når verdien til venstre er mindre enn den til høyre. På samme måte utløses tegnet «>» når verdien til venstre er større enn den til høyre.

Klikk på kategorien Styring i klosspaletten, finn klossen hvis, og dra den deretter til kodeområdet før du slipper den rett under klossen si løkker i 2 sekunder. Den omgir automatisk klossen endre løkker med 1, så du kan klikke og dra den for å flytte den til den ligger under klossen hvis i stedet. Klikk på kategorien Utseende i klosspaletten. Deretter klikker og drar du en kloss av typen si Hei! i 2 sekunder og slippe den inne i klossen hvis. Klikk på kategorien Operatører i klosspaletten. Deretter klikker og drar du klossen ( ) > ( ) inn i det diamantformede hullet i klossen hvis.

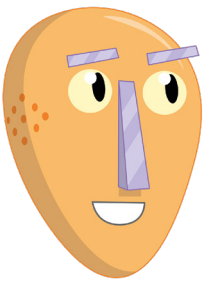


Klossen hvis er en betinget kloss. Det betyr at klossene inne i denne klossen bare kjøres hvis en viss betingelse er oppfylt. Klikk på kategorien Variabler og dra og slipp rapporteringsklossen løkker inn i det tomme rommet i operatorklossen > . Deretter klikker du i området med tallet 50 og skriver inn 5 i stedet. Til slutt klikker du på ordet «Hei!» i klossen si Hei! i 2 sekunder og skriver inn «Det er høyt!».

Klikk på det grønne flagget. Først fungerer programmet som tidligere, med en kattefigur som teller oppover fra null. Når tallet når 6, det første tallet som er større enn 5, utløses klossen hvis, og kattefiguren kommenterer at tallene blir høye (**Figur 4-9**). Gratulerer! Nå kan du jobbe med variabler og betingelser.



▲ **Figur 4-9:** Katten kommenterer når tallet når 6



UTFORDRING: HØYT OG LAVT

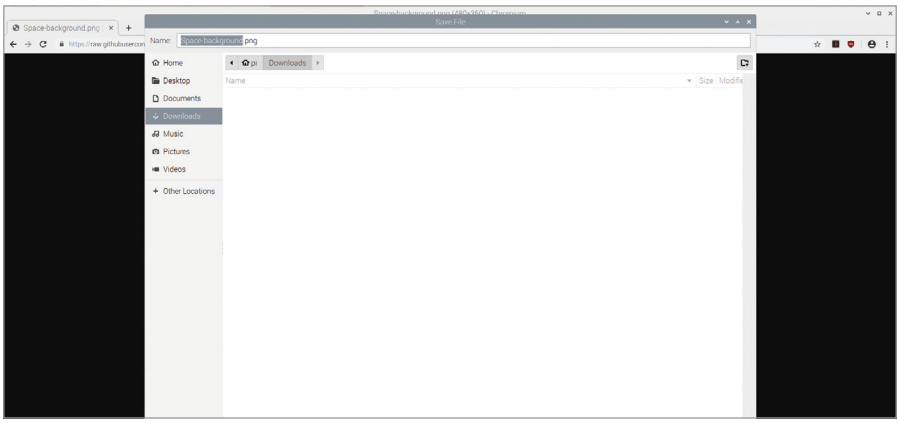
Hvordan kan du endre programmet slik at kattefiguren i stedet kommenterer hvor lave tallene under 5 er? Kan du endre det slik at katten kommenterer både høye og lave tall? Eksperimenter med klossen hvis ellers for å gjøre dette enklere!

Prosjekt 1: Reaksjonstidtaker for astronauter

Nå som du forstår hvordan Scratch fungerer, er det på tide å gjøre noe litt mer interaktivt. Du kan programmere en reaksjonstidtaker, som er utviklet for å hedre den britiske ESA-astronauten Tim Peake og hans tid ombord den internasjonale romstasjonen.

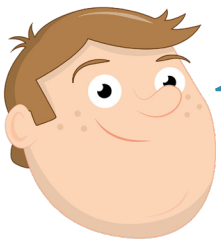
Lagre det eksisterende programmet, hvis du vil beholde det, og åpne et nytt prosjekt ved å klikke på Fil og Nytt. Før du begynner gir du programmet et navn ved å klikke på Fil og Lagre på datamaskinen. Kall det Reaksjonstidtaker for astronauter.

Dette prosjektet avhenger av to bilder, en scenebakgrunn og en figur, som ikke er inkludert i Scratchs innebygde ressurser. Slik laster du ned bildene: Klikk på bringebærikonet for å laste inn Raspberry Pi OS-menyen, flytt musepekeren til Internett og klikk på nettleseren Chromium. Når du har lastet inn nettleseren, skriver du **rpf.io/astronaut-backdrop** på adresselinjen. Deretter trykker du på **ENTER**-tasten. Høyreklikk på bildet av verdensrommet og klikk på «Lagre bilde som…». Deretter klikker du på Lagre-knappen (**Figur 4-10**). Klikk på adresselinjen igjen, og skriv **rpf.io/astronaut-sprite** etterfulgt av **ENTER**-tasten.



▲ **Figur 4-10:** Lagre bakgrunnsbildet

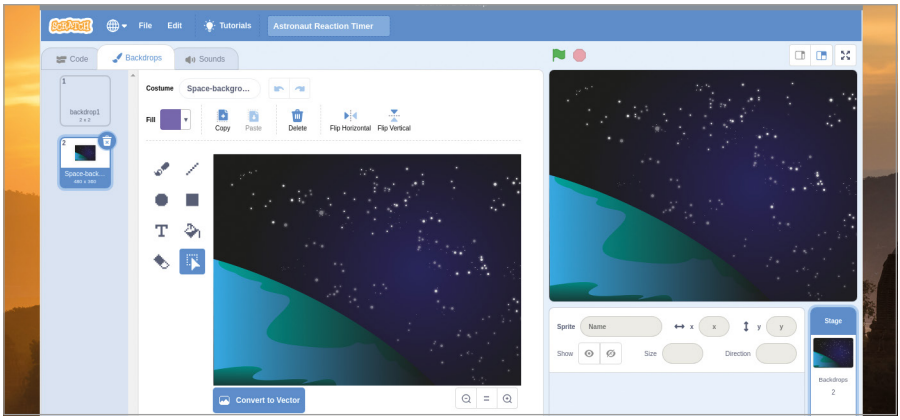
Høyreklikk på bildet av Tim Peake og klikk på «Lagre bilde som…». Deretter velger du mappen Nedlastinger og klikker på Lagre-knappen. Når du har lagret de to bildene, kan du lukke Chromium eller la den være åpen og bruke oppgavelinjen til å gå tilbake til Scratch 3.



BRUKERGRENSESNITT

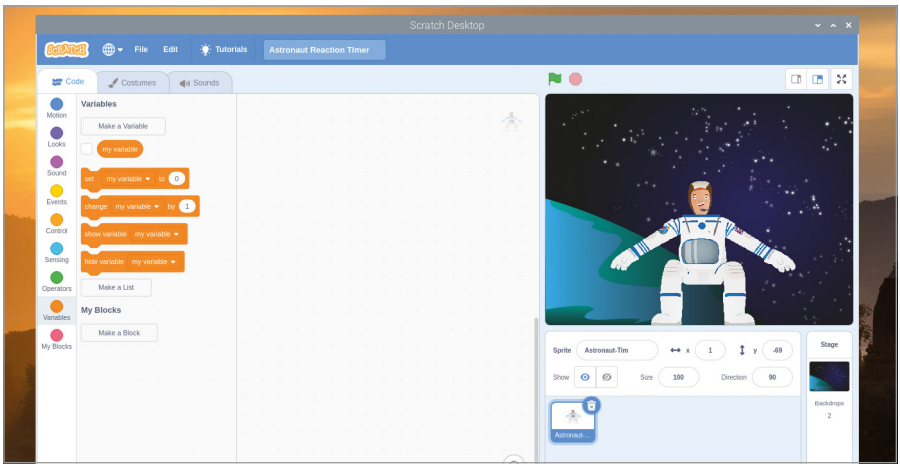
Hvis du har fulgt dette kapittelet fra starten, er du nå kjent med brukergrensesnittet til Scratch 3. Følgende prosjektinstruksjoner er basert på at du vet hvor ting er. Hvis du glemmer hvor du finner noe, kan du se på bildet av brukergrensesnittet i begynnelsen av dette kapittelet for å friske opp hukommelsen.

Høyreklikk på kattefiguren i listen og klikk på «Slett». Hold musepekeren over ikonet Velg bakgrunn, og klikk deretter på ikonet Last opp bakgrunn fra listen som vises. Du finner filen **Space-background.png** i mappen Nedlastinger. Klikk på den for å velge den, og klikk deretter på OK. Den enkle, hvite scenebakgrunnen endres til bildet av verdensrommet, og kodeområdet erstattes av bakgrunnsområdet **Figur 4-11**). Her kan du tegne over bakgrunnen, men foreløpig klikker du bare på fanen Kode øverst i Scratch 3-vinduet.



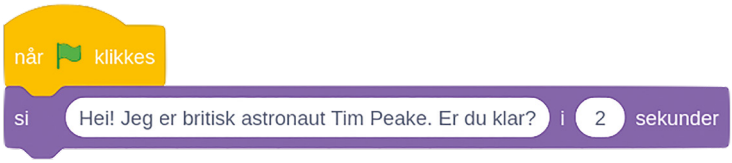
Figur 4-11: Verdensrombakgrunnen vises på scenen

Last opp den nye figuren ved å holde musepekeren over ikonet Velg figur . Deretter klikker du på ikonet Last opp figur øverst på listen som vises. Finn filen **Astronaut-Tim.png** i mappen Nedlastinger, klikk for å velge den, og klikk deretter på OK. Figuren vises automatisk på scenen, men er den er kanskje ikke i midten. Klikk og dra den med musen og slipp den midt i den nedre delen av scenen (**Figur 4-12**).

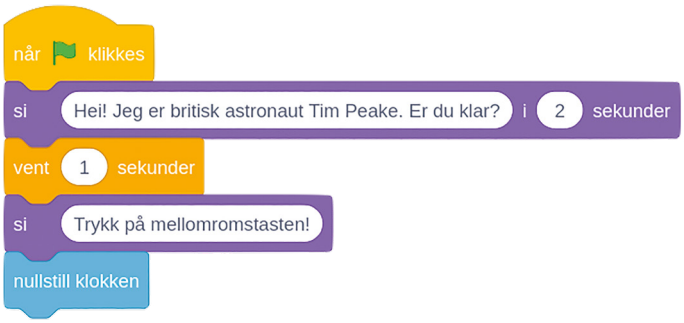


Figur 4-12: Dra astronautfiguren til den står midt i den nedre delen av scenen

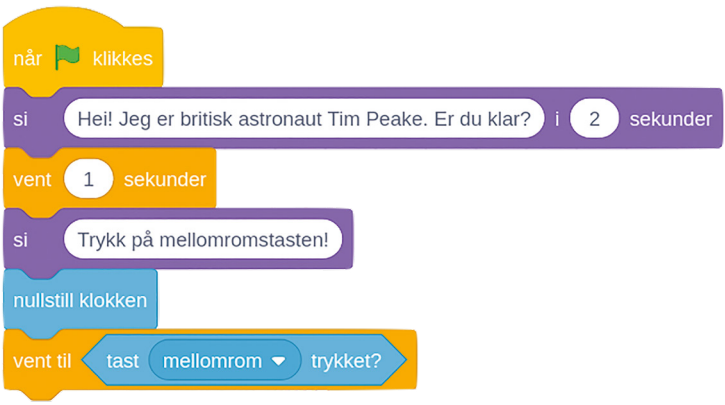
Nå er den nye bakgrunnen og figuren på plass, og du klar til å lage programmet. Start med å opprette en ny variabel kalt «tid», og pass på at «For alle figurer» er valgt før du klikker på OK. Klikk på figuren, enten på scenen eller i figurpanelet, for å velge den. Deretter legger du til en når klikkes -kloss fra kategorien Hendelser i kodeområdet. Deretter legger du til en kloss av typen si Hei! i 2 sekunder fra kategorien Utseende. Deretter klikker du på klossen for å endre den til å si «Hei! Jeg er britisk astronaut Tim Peake. Er du klar?»



Legg til en kloss av typen **vent 1 sekunder** fra kategorien Styring og deretter en kloss med teksten **si Hei!**. Endre teksten i denne klossen til «Dra til verdensrommet». Deretter legger du til en kloss av typen **nullstill klokken** fra kategorien Sansing. Denne klossen styrer en spesiell variabel som er innebygd i Scratch. Den brukes til å beregne tid, og kan beregne hvor raskt du kan reagere i spillet.



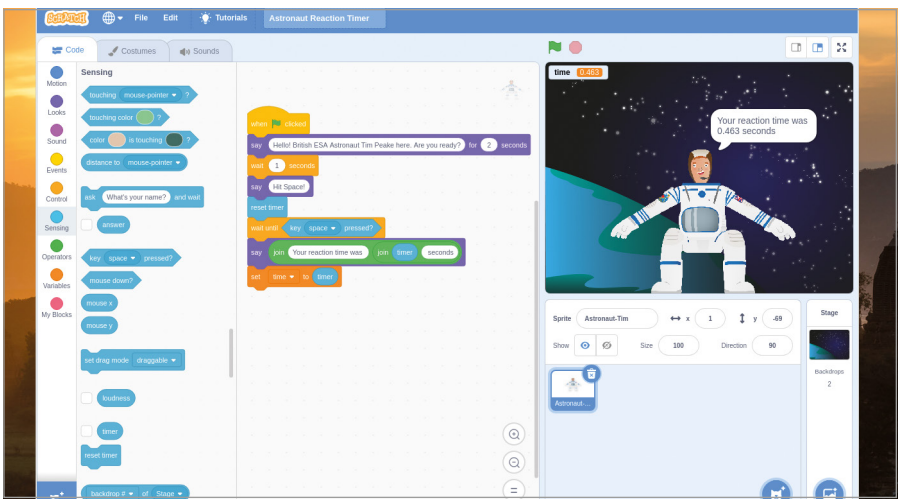
Legg til styringsklossen **vent til** og dra en sensingskloss av typen **tast mellomrom trykket?** inn i det hvite området. Dette stanser programmet til du trykker på **mellomromstasten** på tastaturet, men tidtakeren vil fortsette å gå. Den teller nøyaktig hvor lang tid det går mellom meldingen «Dra til verdensrommet» og når du faktisk trykker på **mellomromstasten**.



Nå vil du at Tim skal fortelle deg hvor lang tid det tok før du trykket på **mellomromstasten**, men på en måte som er lett å lese. Da trenger du operatorklossen **sett sammen** . Du må ha to verdier, inkludert variabler, og koble dem sammen etter hverandre – kjent som en *sammenkobling*.

Start med klossen si Hei!, og dra og slipp deretter operatorklossen sett sammen over ordet «Hei!». Klikk på «eple» og skriv «Reaksjonstiden din var » i stedet. Husk å legge til et mellomrom på slutten av strengen. Deretter drar du en annen sammensettingskloss over «banan» i det andre feltet. Dra rapporteringsklossen sekundklokke fra kategorien Sansing til det som nå er det midtre feltet, og skriv inn « sekunder» i det siste feltet. Husk å legge inn et mellomrom på begynnelsen.

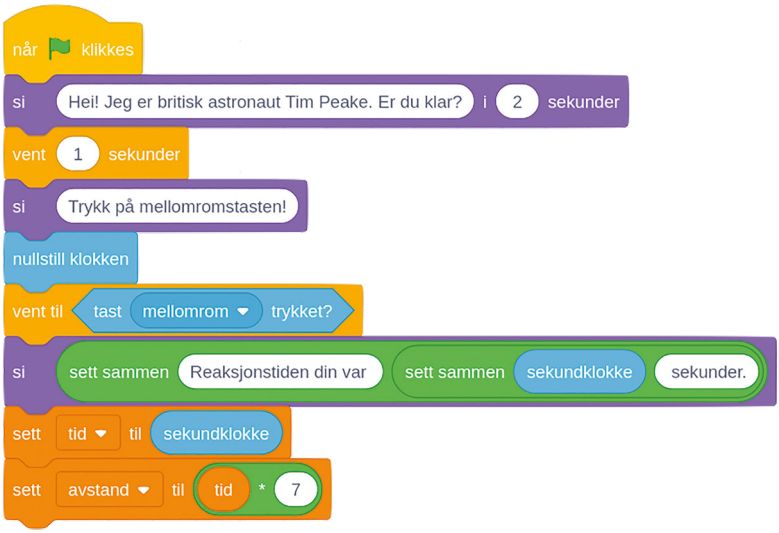
Til slutt drar du variabelklossen sett min variabel til 0 til slutten av sekvensen. Klikk på rullegardinpilen ved siden av «min variabel» og klikk på «tid» i listen. Deretter erstatter du 0 med rapporteringsklossen sekundklokke fra Sansing-kategorien. Spillet er nå klart til testing. Du klikker bare på det grønne flagget over scenen. Gjør deg klar. Straks du ser meldingen «Dra til verdensrommet!», trykker du på **mellomromstasten** så raskt du kan (**Figur 4-13**) – se om du kan slå den høyeste poengsummen vår!



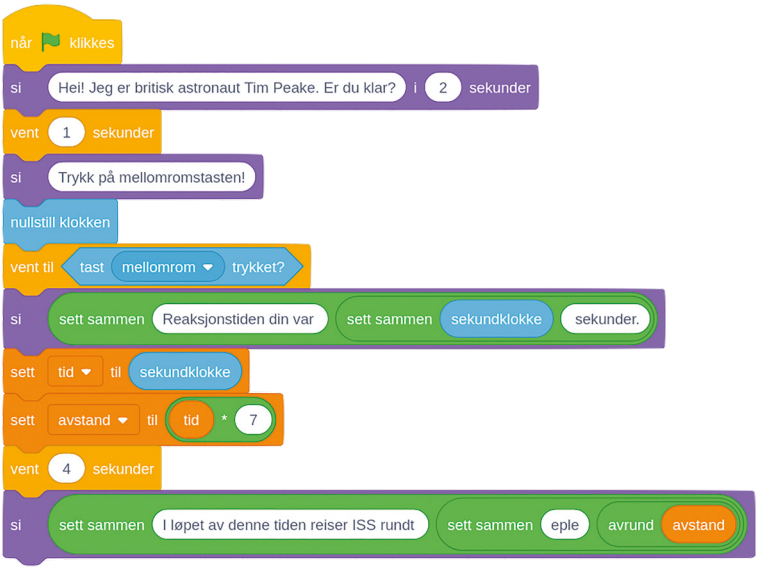
▲ **Figur 4-13:** På tide å spille spillet!

Du kan utvide dette prosjektet ytterligere ved å få det til å beregne omtrent hvor langt den internasjonale romstasjonen har reist innen du trykket på **mellomromstasten**, basert på stasjonens publiserte hastighet på syv kilometer per sekund. Først oppretter du en ny variabel som heter «avstand». Legg merke til hvordan klossene i kategorien Variabler automatisk endres for å vise den nye variabelen. De eksisterende variabelklossene av typen tid forblir de samme.

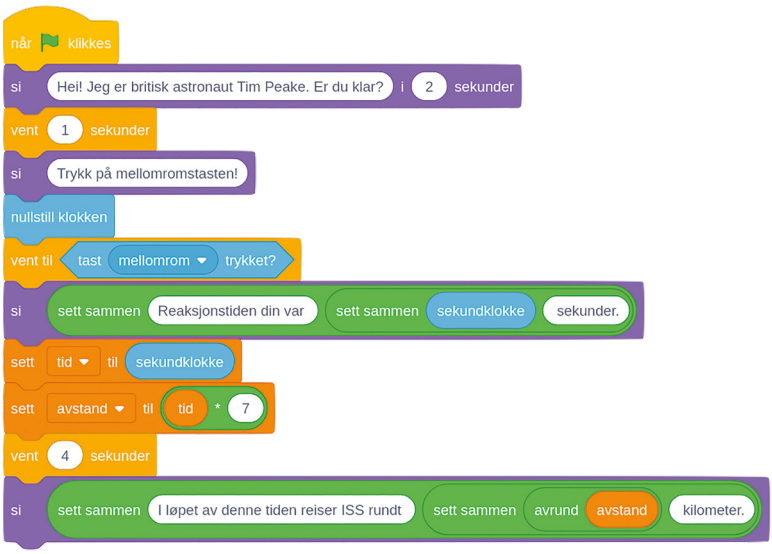
Legg til klossen sett avstand til 0, og dra deretter operatorklossen (*), som angir multiplikasjon, over 0. Dra rapporteringsklossen tid over det første tomme området, og skriv deretter inn tallet 7 i det andre området. Når du er ferdig, ser den kombinerte klossen slik ut: angi avstand til tid * 7. Det vil si at tiden det tok deg å trykke på **mellomromstasten**, multipliseres med syv for å beregne avstanden ISS har tilbakelagt, i kilometer.



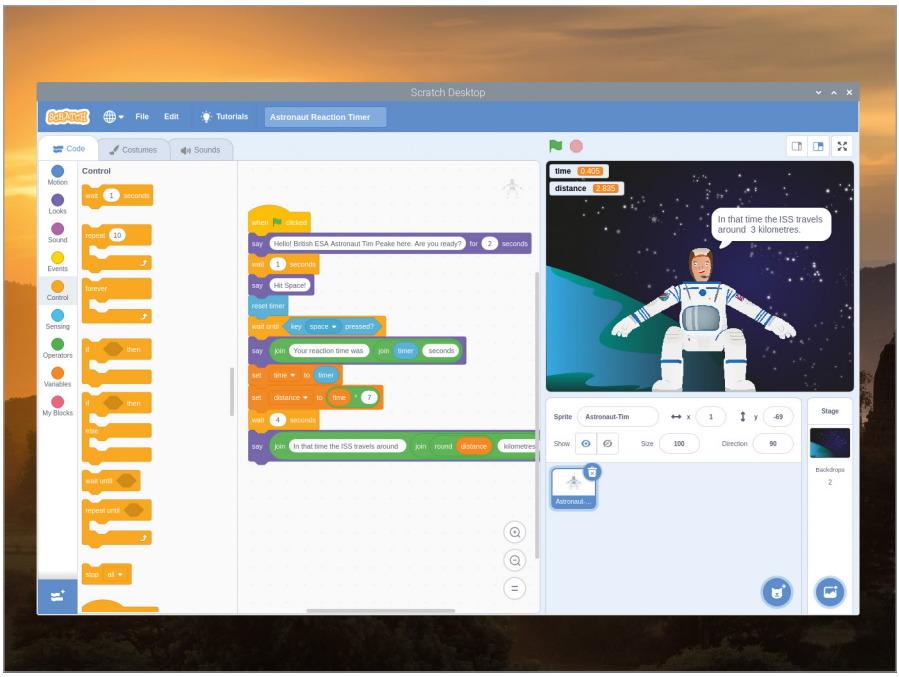
Legg til klossen vent 1 sekund og endre verdien til 4. Til slutt drar du en annen si Hei! -kloss til slutten av sekvensen og legger til to sett sammen -klosser, på samme måte som tidligere. I det første området, der det står «eple», skriver du «I løpet av denne tiden reiser ISS rundt» i stedet. Husk å legge til et mellomrom på slutten av strengen. I området der det står «banan», erstatter du teksten med «kilometer». Her må du huske å legge inn et mellomrom før strengen.



Til slutt drar du operatorklossen **avrund** inn i det midterste tomme området. Deretter drar du rapporteringsklossen **avstand** inn i det nye tomme området som ble opprettet. Klossen **avrund** runder tall opp eller ned til nærmeste hele tall, så i stedet for et supernøyaktig kilometertall som er vanskelig å lese, ser du et helt tall som er enkelt å lese.



Klikk på det grønne flagget for å kjøre programmet og se hvor langt ISS reiser innen du trykker på **mellomromstasten**. Husk å lagre programmet når du er ferdig, slik at du enkelt kan laste det inn igjen i fremtiden uten å måtte starte fra begynnelsen igjen.



▲ **Figur 4-14:** Tim forteller deg hvor langt ISS har reist

UTFORDRING: HVEM ER RASK?

Bortsett fra astronauter – hvilke andre yrker krever lynraske reflekser? Kan du tegne dine egne figurer og bakgrunner for å vise en av disse yrkene?

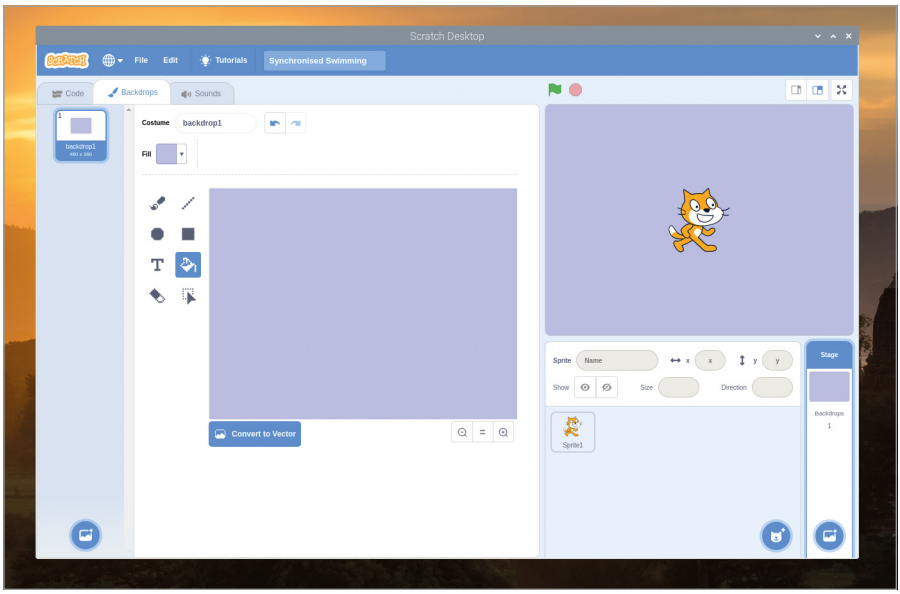
Prosjekt 2: Synkronsvømming

De fleste spill bruker mer enn en enkelt knapp. Dette prosjektet demonstrerer det ved å tilby styring med to knapper ved å bruke tastene ← og → på tastaturet.

NETTPROSJEKT

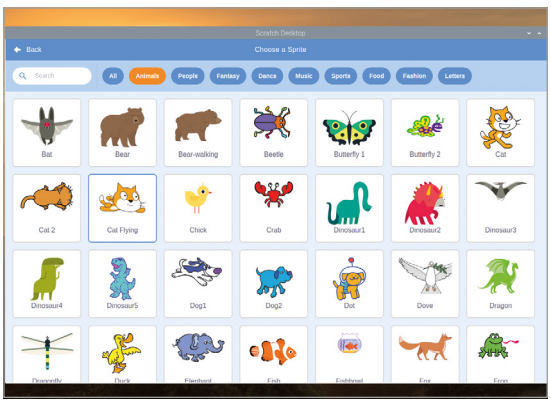
Dette prosjektet er også tilgjengelig på nettet under **rpf.io/synchro-swimming**

Lag et nytt prosjekt og lagre det som Synkronsvømming. Klikk på Scene i delen for scenekontroll, og klikk deretter på kategorien Bakgrunner øverst til venstre på skjermen. Klikk på knappen Konverter til bitmap under bakgrunnen. Velg en vannlignende blåfarge fra Fyll-paletten. Klikk på Fyll-ikonet og deretter på den rutete bakgrunnen for å fylle den med blåfargen (**Figur 4-15**).



Figur 4-15: Fylle bakgrunnen med en blåfarge

Høyreklikk på kattefiguren i listen og klikk på «Slett». Klikk på ikonet «Velg figur» for å se en liste over innebygde figurer. Klikk på kategorien Dyr, klikk på «Cat Flying» (**Figur 4-16**) og deretter på OK. Denne figuren fungerer også bra for svømmeprosjekter.



Figur 4-16: Velge en figur fra biblioteket

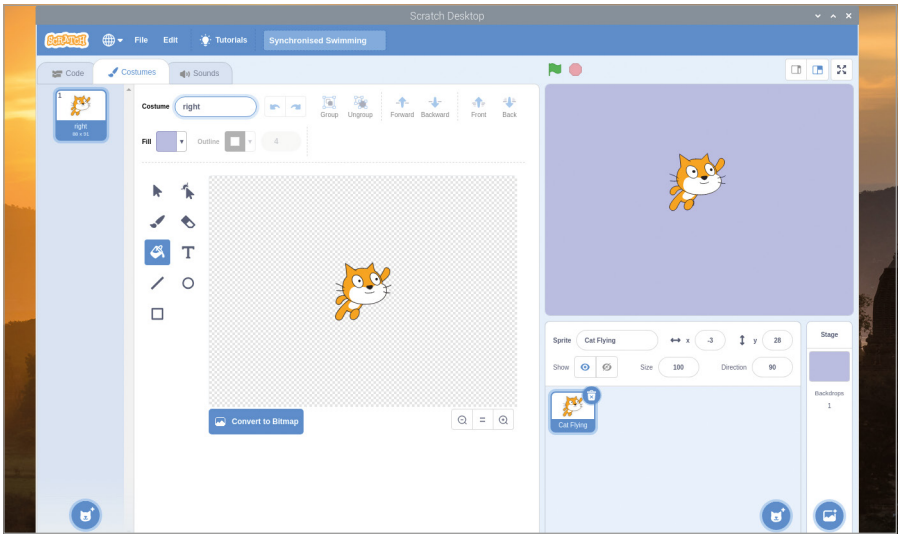
Klikk på den nye figuren, og dra deretter to hendelsesklosser av typen når mellomrom trykkes inn i kodeområdet. Klikk på den lille nedoverpilen ved siden av ordet «mellomrom» i den første klossen, og velg «pil venstre» fra listen over mulige alternativer. Dra bevegelsesklossen snu 15 grader under klossen når pil venstre trykkes, og

gjør det samme med den andre hendelsesklossen, bortsett fra at du velger «pil høyre» fra listen og bruker snu 15 grader bevegelsesklossen.

Trykk på tasten ← eller → for å teste programmet. Du ser at kattefiguren snur seg på samme måte som deg, avhengig av retningen du velger på tastaturet. Merk: Du behøvde ikke å klikke på det grønne flagget denne gangen fordi utløserklossene du har brukt fra Hendelser, er aktive hele tiden, selv når programmet ikke «kjører» i vanlig forstand.

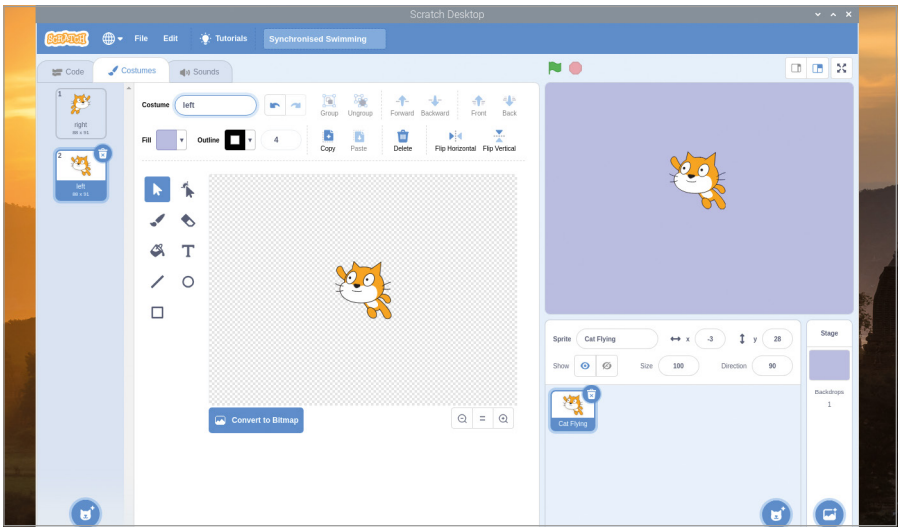
Gjenta trinnene to ganger, men denne gangen velger du « pil opp» og « pil ned» for utløserklossen fra Hendelser, og deretter gå 10 steg og gå -10 steg for bevegelsesklossene. Når du trykker på piltastene, ser du at katten kan snu seg og svømme både fremover og bakover.

For å gjøre kattefigurens bevegelser mer realistiske, kan du endre hvordan den ser ut. I Scratch kalles dette en *drakt*. Klikk på kattefiguren, og klikk deretter på fanen Drakter over klosspaletten. Klikk på drakten «cat flying-a» og klikk på ikonet med en X i en papirkurv, som vises øverst til høyre på skjermen for å slette drakten. Deretter klikker du på drakten «cat flying-b» og endrer navnet i navnefeltet øverst på skjermen til «høyre» (**Figur 4-17**).



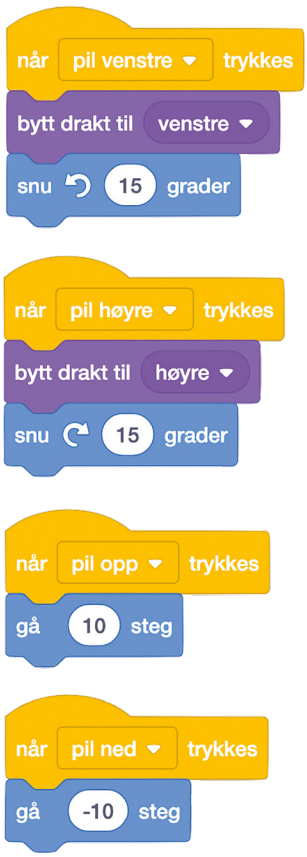
Figur 4-17: Endre draktens navn til «høyre»

Høyreklikk på den nye «høyre»-drakten, og klikk på «dupliser» for å lage en kopi. Klikk på denne kopien for å velge den, klikk på Velg-ikonet , klikk på Speilvend . Deretter endrer du navnet til «venstre» (**Figur 4-18**). Til slutt vil du ha to «drakter» for figuren din, som er nøyaktige speilbilder. Den ene kalles «høyre» der katten vender mot høyre, og den andre kalles «venstre» der katten vender mot venstre.



Figur 4-18: Duplisere drakten, snu den og endre navn til «venstre»

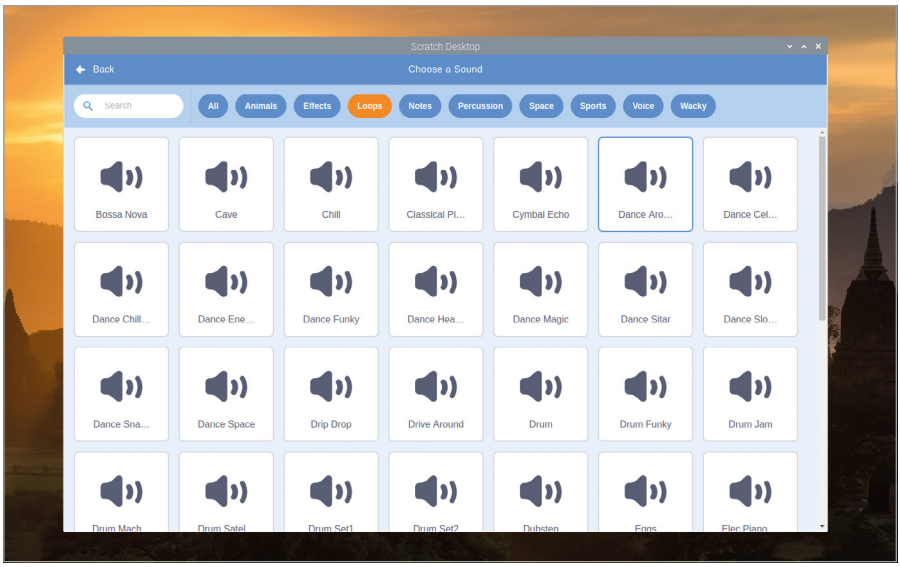
Klikk på Kode-fanen over draktområdet, og dra deretter to utseendeklosser av typen **bytt drakt til venstre** under hendelsesklossene med venstre pil og høyre pil. Endre teksten i klossen under hendelsesklossen med høyre pil til **bytt drakt til høyre** . Prøv piltastene igjen. Nå snur katten seg mot retningen den svømmer i.



For synkronsvømming på olympisk nivå trenger vi imidlertid flere svømmere. Vi trenger også en måte å tilbakestille kattefigurens posisjon på. Legg til hendelsesklossen når klikkes . Under den legger du deretter til bevegelsesklossen gå til x: 0 y: 0, endre verdiene om nødvendig, og bevegelsesklossen pek i retning 90. Når du klikker på det grønne flagget, flyttes katten til midten av scenen og vender mot høyre.

Når du vil opprette flere svømmere, legger du til klossen gjenta 6, og endrer standardverdien 10 til 6. Deretter legger du til styringsklossen lag klon av meg inne i denne klossen. For å sikre at ikke alle svømmerne svømmer i samme retning, legger du til klossen snu 60 grader over klossen lag klon, fremdeles inne i klossen gjenta 6 . Klikk på det grønne flagget, og prøv piltastene nå for å se at svømmerne bli levende.

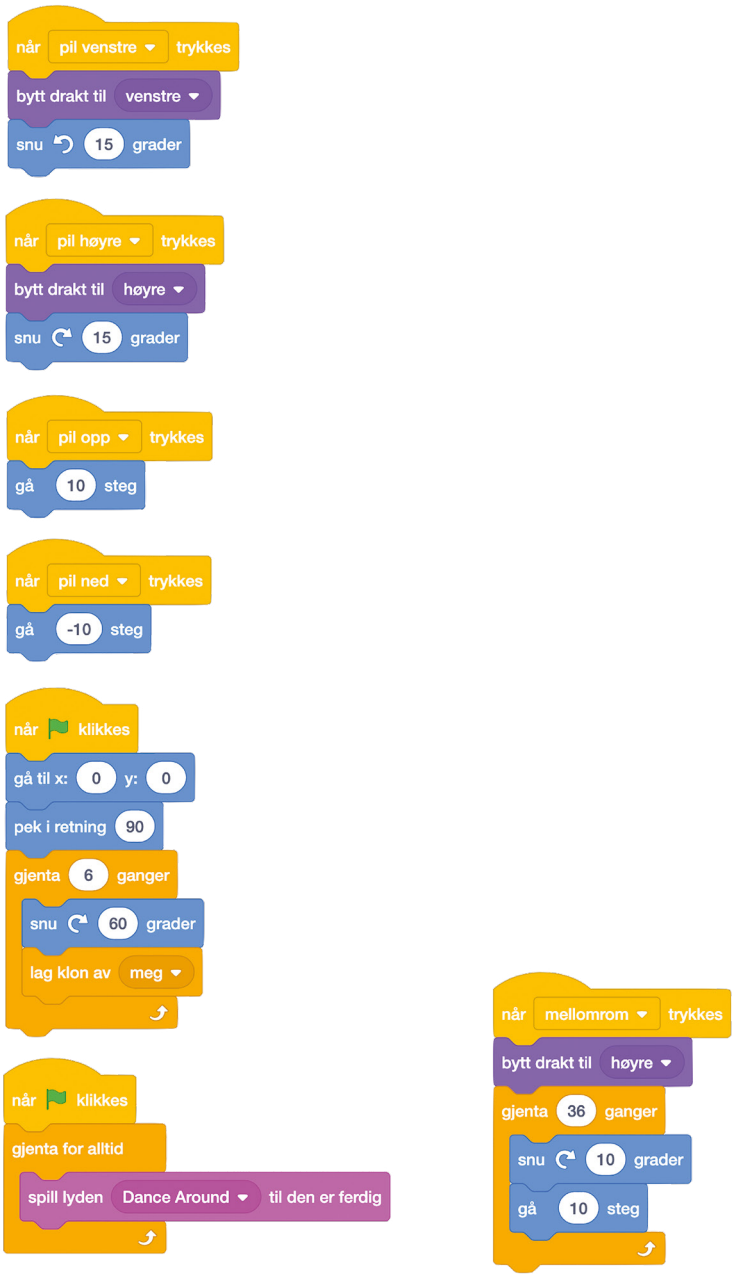
Legg til litt musikk for å få en olympisk stemning. Klikk på fanen Lyder over klosspaletten. Deretter klikker du på ikonet «Velg lyd»  . Klikk på kategorien Løkker, og bla gjennom listen (**Figur 4-19**) til du finner musikk du liker. Vi har valgt «Dance Around». Klikk på OK-knappen for å velge musikken. Deretter klikker du på Kode-fanen for å åpne kodeområdet igjen.



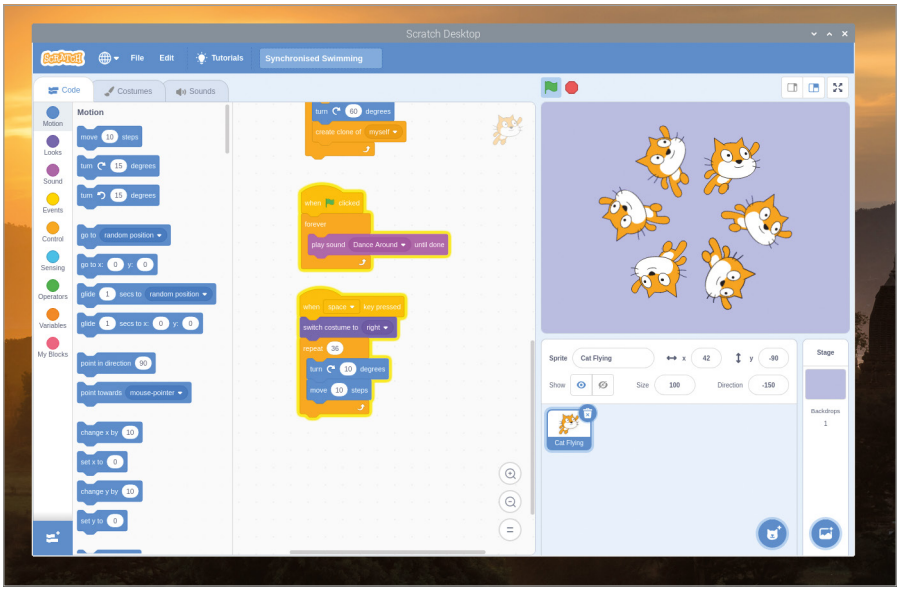
Figur 4-19: Velge en musikkløkke fra lydbiblioteket

Legg til en ny hendelseskloss av typen når klikkes i kodeområdet. Deretter legger du til styringsklossen gjenta for alltid. Inne i denne styringsklossen legger du til klossen spill «Dance around» til den er ferdig. Husk å se etter navnet på musikksnutten du har valgt. Klikk på det grønne flagget for å teste det nye programmet. Hvis du vil stoppe musikken, klikker du på den røde åttekanten for å stoppe programmet og dempe lyden.

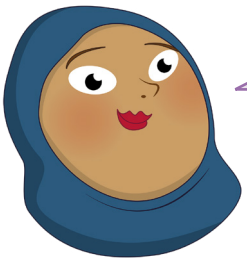
Til slutt kan du simulere en full danserutine ved å legge til en ny hendelsesutløser i programmet. Legg til hendelsesklossen **når mellomrom trykkes** og deretter klossen **bytt drakt til høyre**. Under denne klossen legger du til klossen **gjenta 36**. Husk å endre verdien fra standardverdien. Inne i denne klossen legger du til klossene **snu 10 grader** og **gå 10 steg**.



Klikk på det grønne flagget for å starte programmet, og trykk deretter på **mellomromstasten** for å prøve den nye rutinen (**Figur 4-20** på neste side)! Ikke glem å lagre programmet når du er ferdig.



▲ **Figur 4-20:** Den ferdige synkronsvømmingsrutinen



UTFORDRING: TILPASSET RUTINE

Kan du lage din egen synkronsvømmingsrutine ved å bruke løkker? Hva må du endre hvis du vil ha flere eller færre svømmere? Kan du legge til flere svømmerutiner som kan utløses ved hjelp av forskjellige taster på tastaturet?

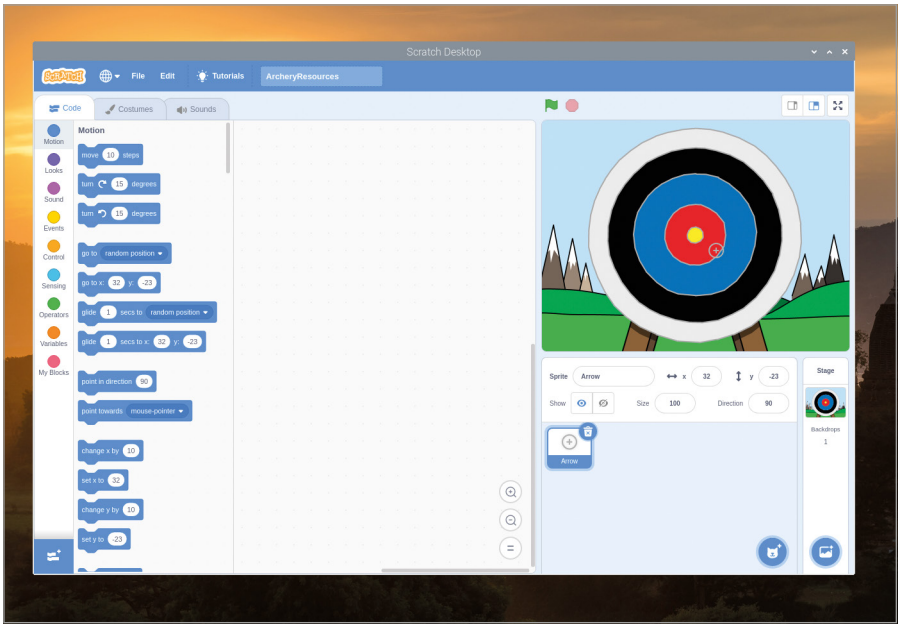
Prosjekt 3: Bueskyting

Nå som du er litt av en ekspert på Scratch, er det på tide å jobbe med noe litt mer utfordrende: et bueskytingsspill, der spilleren må treffe et mål med pil og bue når buen svaier vilkårlig.

NETTPROSJEKT

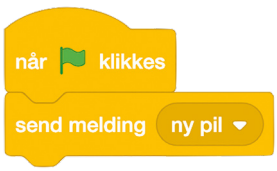
Dette prosjektet er også tilgjengelig på nettet under **rpf.io/archery**

Start med å åpne nettleseren Chromium og skrive **rpf.io/p/en/archery-go** etterfulgt av **ENTER**-tasten. Ressursene for spillet lastes ned som en zip-fil, så du må pakke den ut (høyreklikk på filen og velg Pakk ut her). Gå tilbake til Scratch 3 og klikk på Fil-menyen etterfulgt av Last inn fra datamaskinen. Klikk på **ArcheryResources.sb3** etterfulgt av Åpne-knappen. Du blir spurt om du vil erstatte innholdet i prosjektet. Hvis du ikke har lagret endringene, klikker du på Avbryt og lagrer dem. Ellers klikker du på OK.



▲ **Figur 4-21:** Ressursprosjekt lastet inn for bueskytingspillet

Prosjektet du nettopp har lastet inn, inneholder en bakgrunn og en figur (**Figur 4-21**), men ingen faktisk kode for å lage et spill. Det er din jobb å gjøre det. Begynn med å legge til klossen når klikkes og deretter klossen send melding melding1. Klikk på nedpilen på slutten av klossen og deretter på «Ny melding». Skriv inn «ny pil» og klikk på OK-knappen. Klossen har nå teksten send melding ny pil.

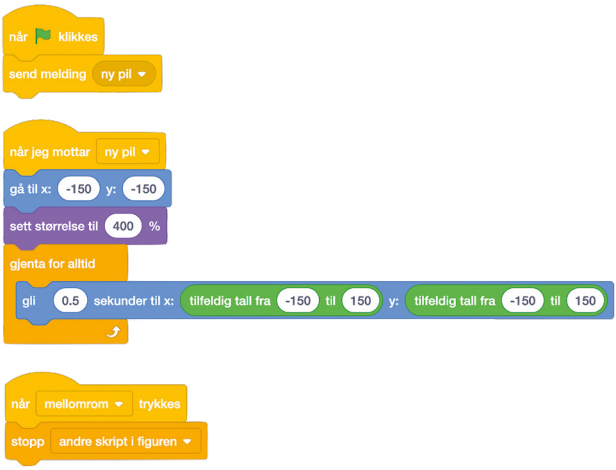


En kringkasting er en melding fra en del av programmet, som kan mottas av andre deler av programmet. For å få det til å gjøre noe legger du til klossen når jeg mottar melding1 og endrer teksten til når jeg mottar ny pil. Denne gangen kan du bare klikke på nedpilen og velge «ny pil» fra listen. Du trenger ikke å opprette meldingen igjen.

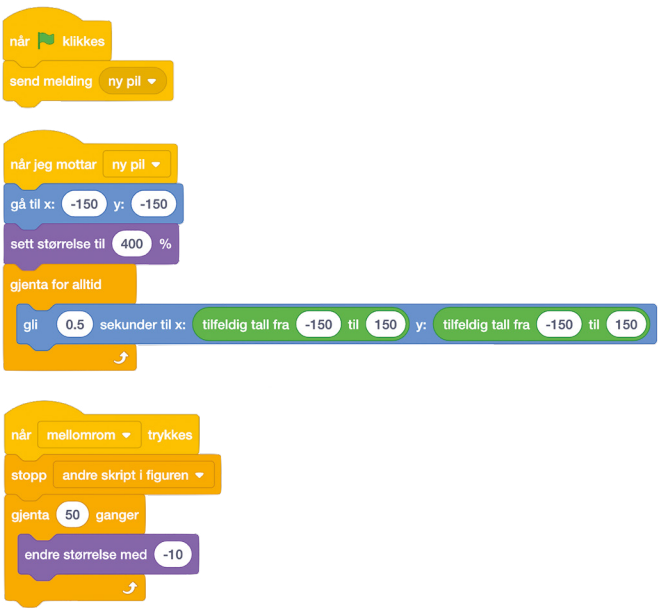
Under klossen når jeg mottar ny pil legger du til klossene gå til x: –150 y: -150 og sett størrelse til 400 %. Husk at dette ikke er standardverdiene for disse klossene, så du må endre teksten når du har dratt dem til kodeområdet. Klikk på det grønne flagget for å se hva du har gjort så langt. Pilfiguren, som spilleren bruker for å sikte på målet, hopper til nedre venstre hjørne på scenen og firedobles i størrelse.

Hvis du vil gi spilleren en utfordring, legger du til en bevegelse som simulerer svaiing når buen trekkes og bueskytteren sikter. Dra klossen **gjenta for alltid**, etterfulgt av klossen **gli 1 sekunder til x: –150 y: –150**. Rediger det første hvite feltet så det sier 0.5 i stedet for 1. Deretter plasserer du en operatorkloss av typen **tilfeldig tall fra –150 til 150** i hver av de to andre hvite feltene. Dette betyr at pilen vil bevege seg på scenen i en tilfeldig retning og en tilfeldig avstand, så det blir mye vanskeligere å treffe målet.

Klikk på det grønne flagget igjen for å se hva klossen gjør. Pilfiguren beveger seg nå rundt på scenen og dekker forskjellige deler av målet. For øyeblikket kan du ikke miste pilen ved målet. Dra klossen **når mellomrom trykkes** inn i kodeområdet, etterfulgt av styringsklossen **stopp alle**. Klikk på nedpilen på slutten av klossen og endre den til klossen **stopp andre skript i figuren**.

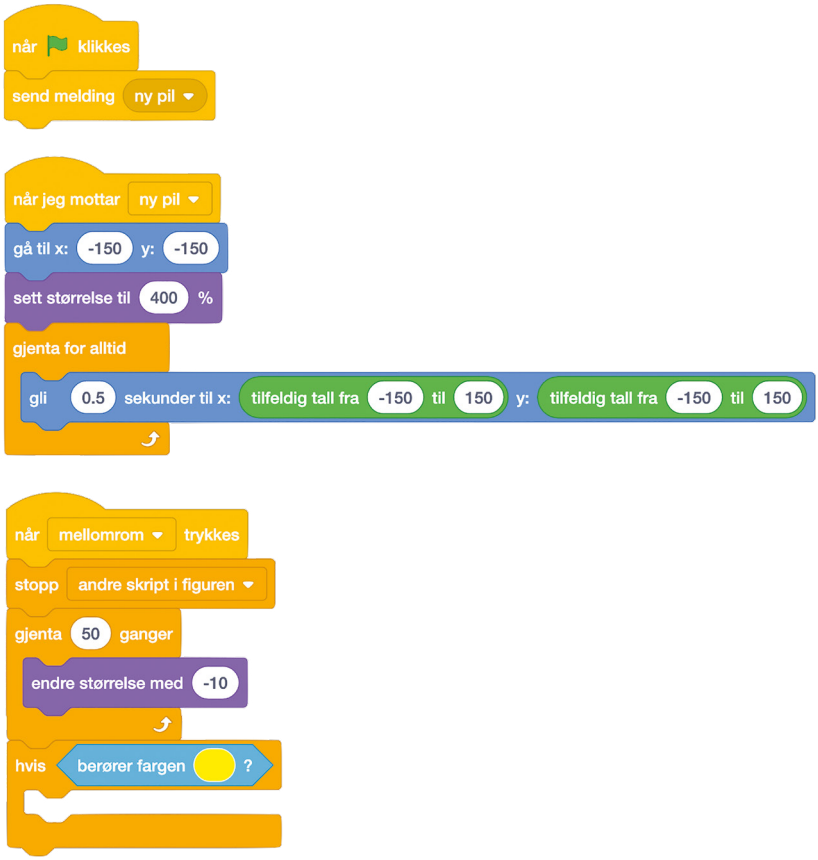


Hvis du stoppet programmet for å legge til de nye klossene, klikker du på det grønne flagget for å starte det igjen. Deretter trykker du på **mellomromstasten**. Du vil nå se at pilfiguren slutter å bevege seg. Det er bra, men du vil at det skal se ut som om pilen beveger seg mot målet. Legg til klossen gjenta 50 etterfulgt av klossen endre størrelse med -10. Deretter klikker du på det grønne flagget for å teste spillet igjen. Nå ser det ut som om pilen beveger seg bort fra deg og mot målet.

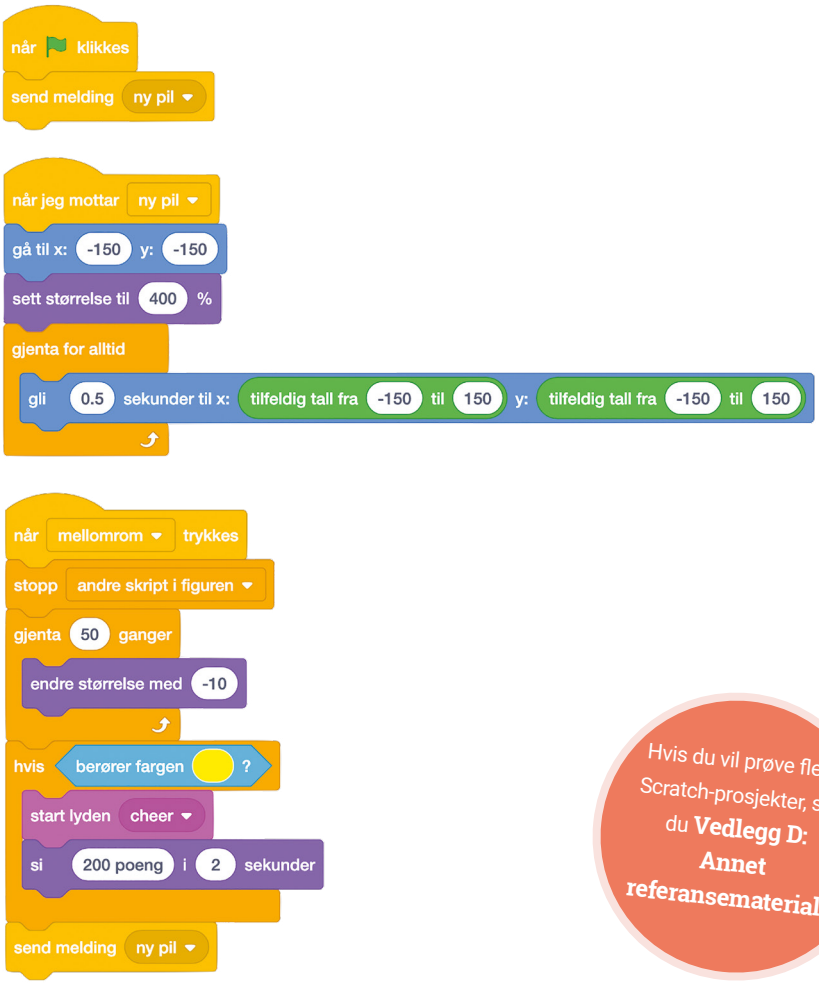


Du kan gjøre spillet morsommere ved å legge til en metode for å holde oversikt over poengsummene til brukeren. I den samme klosstabelen legger du til klossen hvis. Pass på at du plasserer den under klossen gjenta 50 og ikke inni den. Det diamantformede området skal

inneholde sansingsklossen berører farge? . Når du skal velge farge, klikker du på det fargede feltet på slutten av sansingsklossen. Deretter klikker du på dråpetellerikonet og så på den gule blinken midt i målskiven på scenen.



Hvis du vil at spillerne skal se at de har skåret poeng, legger du til klossene start lyden cheer og si 200 poeng i 2 sekunder inni klossen hvis. Til slutt legger du til klossen send melding ny pil helt nederst i stabelen, under klossen hvis, for å gi spillerne en ny pil hver gang de har skutt. Klikk på det grønne flagget for å starte spillet, og prøv å treffe den gule blinken. Når du treffer blinken, blir du belønnet med jubel fra publikum og 200 poeng!



Hvis du vil prøve flere Scratch-prosjekter, ser du **Vedlegg D: Annet referansemateriale**

Spillet fungerer, men det er litt mer utfordrende. Bruk det du har lært i dette kapittelet, og prøv å forbedre spillet ved å legge til poengsummer for treff i andre områder på målet enn selve midtblinken: 100 poeng for rødt, 50 poeng for blått og så videre.

UTFORDRING: KAN DU FORBEDRE DET?

Hvordan gjøre spillet enklere? Hvordan gjøre det vanskeligere? Kan du bruke variabler for å øke spillerens poengsum når de skyter flere piler? Kan du legge til en nedtellingstidtaker for å legge mer press på spilleren?

Kapittel 5

Programmering med Python

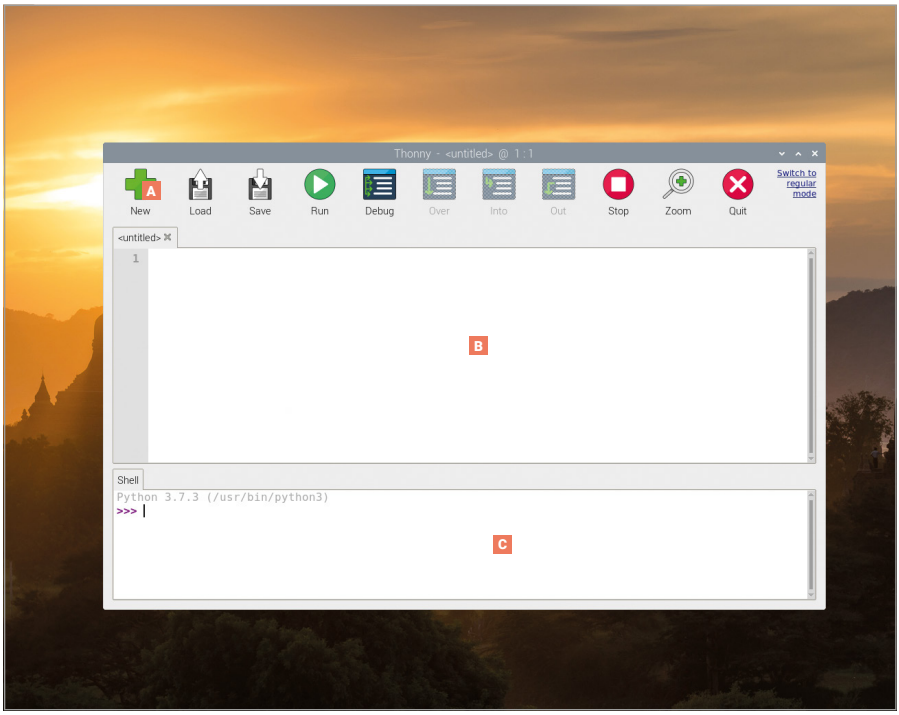
Nå som du har fått grepet på Scratch, skal vi vise deg hvordan du utfører tekstbasert koding med Python

Guido van Rossums Python er oppkalt etter humorgruppen Monty Python og startet som et hobbyprosjekt. Det ble først utgitt for publikum i 1991 og har vokst til et populært programmeringsspråk som kan brukes til mange ulike prosjekter. Scratch har et mer visuelt miljø, mens Python er tekstbasert. Du skriver instruksjoner i et enkelt språk og bestemt format, som deretter blir utført av datamaskinen.

Python er det naturlige neste trinn for de som allerede har brukt Scratch. Det er veldig fleksibelt og gir et mer «tradisjonelt» programmeringsmiljø. Det er ikke dermed sagt at det er vanskelig å lære. Med litt øvelse kan alle skrive Python-programmer for alt fra enkle beregninger til overraskende kompliserte spill.

Dette kapitlet bygger videre på termer og begreper som ble introdusert i **Kapittel 4: Programmering med Scratch 3**. Hvis du ikke har jobbet deg gjennom øvelsene i det kapitlet ennå, er det lurt å gå tilbake og gjøre det først. Da blir dette kapitlet enklere å forstå.

Vi presenterer Thonny Python IDE



A **Verktøylinje** – Grensesnittet «Simple Mode» i Thonny viser en menylinje med kjente ikoner, som du kan bruke til å opprette, lagre, laste inn og kjøre Python-programmene, samt teste dem på forskjellige måter.

B **Skriptområde** – Her skriver du Python-programmene. Området er delt inn i et hovedområde for programmet og en liten sidemarg som viser linjenumre.

C **Python Shell** – Du bruker Python-skallet til å skrive individuelle instruksjoner som kjøres så snart du trykker på **ENTER**-tasten. Det viser også informasjon om programmer som kjøres.

THONNY-VERSJONER

Thonny har to grensesnittversjoner: «Regular Mode» (vanlig modus) og «Simple Mode» (enkel modus), som er bedre for nybegynnere. Dette kapittelet bruker enkel modus, som lastes inn som standard når du åpner Thonny fra programmeringsdelen i Raspberry-menyen.

Det første Python-programmet ditt: Hei, verden!

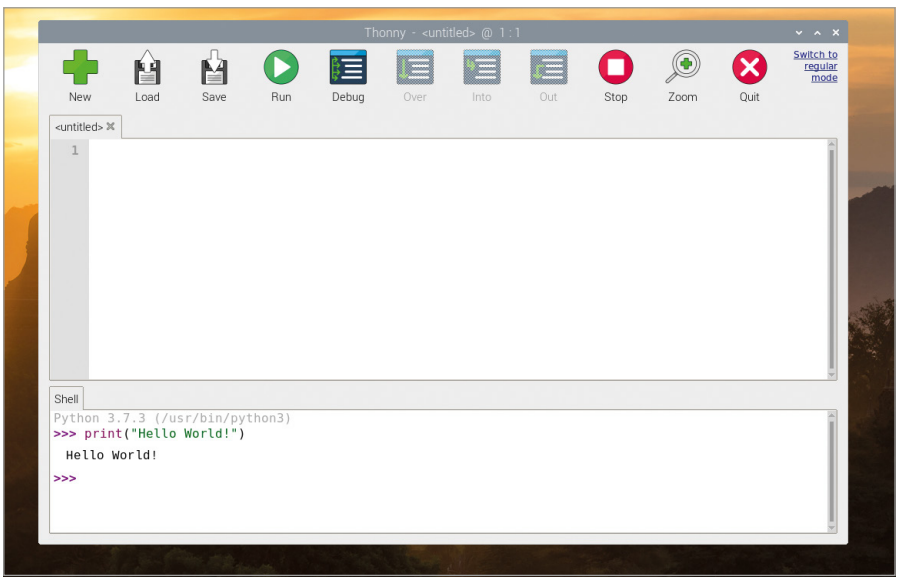
Som alle de andre forhåndsinstallerte programmene på Raspberry Pi er Thonny oppført i menyen. Klikk på bringebærikonet, flytt markøren til delen Programmering, og klikk deretter på Thonny Python IDE. Etter noen sekunder lastes brukergrensesnittet Thonny (enkel modus er standard).

Thonny er en pakke som kalles et *integrert utviklingsmiljø (IDE)*, et komplisert navn med en enkel forklaring. Det samler sammen, eller *integrerer*, alle de forskjellige verktøyene du trenger for å skrive, eller *utvikle*, programvare til et enkelt grensesnitt, eller *miljø*. Det finnes mange IDE-er, og noen av dem støtter mange forskjellige programmeringsspråk. Thonny fokuserer derimot på å støtte ett enkelt språk.

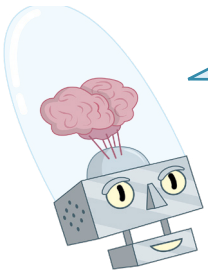
I motsetning til Scratch, som leverer visuelle byggesteiner (klosser) til programmet ditt, er Python et mer tradisjonelt programmeringsspråk der alt skrives ned. Start det første programmet ditt ved å klikke i skallområdet (Shell) for Python nederst i Thonny-vinduet. Deretter skriver du inn følgende instruksjon og trykker på **ENTER**-tasten:

```
print("Hei, verden!")
```

Når du trykker på **ENTER**, ser du at programmet begynner å kjøre umiddelbart: Python svarer med meldingen «Hello, World!» i det samme skallområdet (**Figur 5-1**), akkurat slik du programmerte det. Det skjer fordi skallet er en direktelinje til Pythons *tolk*, som har som oppgave å se på instruksjonene dine og *tolke* hva de betyr. Dette kalles *interaktiv modus*. Du kan se for deg en samtale ansikt-til-ansikt med noen. Så snart du er ferdig med å snakke, vil den andre svare og vente på hva du sier videre.



▲ **Figur 5-1:** Python skriver «Hello, World!» i skallområdet



SYNTAKSFEIL

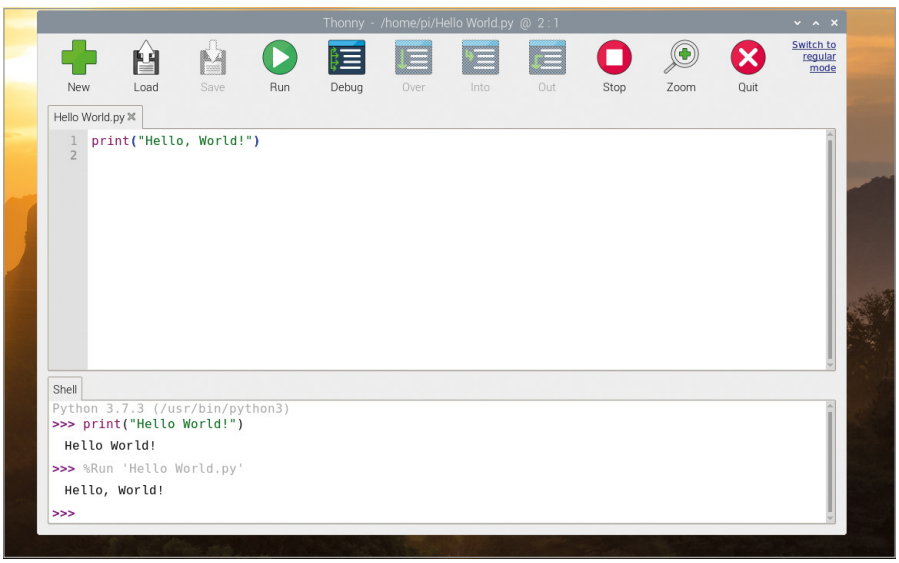
Hvis programmet ikke kjører og i stedet skriver ut en melding om «syntaksfeil» i skallområdet, er det en feil i det du har skrevet. Python krever at instruksjonene skrives på en veldig spesifikk måte: Hvis det mangler en parentes eller et anførselstegn, «print» er skrevet feil eller har stor forbokstav, eller det finnes ekstra symboler et sted i instruksjonen, kjøres ikke programmet. Skriv instruksjonen på nytt, og sørg for at den stemmer med versjonen i denne veiledningen før du trykker på **ENTER**!

Du trenger imidlertid ikke å bruke Python i interaktiv modus. Klikk på skriptområdet midt i Thonny-vinduet, og skriv inn programmet på nytt:

```
print("Hei, verden!")
```

Når du trykker på **ENTER**-tasten denne gangen, skjer det ingenting. Du ser bare en ny, tom linje i skriptområdet. Klikk på Run-ikonet på verktøylinjen i Thonny for å få denne versjonen av programmet til å fungere. Da blir du først bedt om å lagre programmet. Skriv inn et beskrivende navn, for eksempel «Hello, World» og klikk på lagreknappen (Save). Når programmet ditt er lagret, ser du to meldinger i skallområdet for Python (**Figur 5-2**):

```
>>> %Run 'Hei, verden.py'
    Hei, verden!
```



Figur 5-2: Kjøre det enkle programmet

Den første av disse linjene er en instruksjon fra Thonny som ber Python-tolken om å kjøre programmet du nettopp lagret. Den andre er resultatet av programmet – meldingen du ba Python om å skrive ut. Gratulerer! Du har nå skrevet og kjørt det første Python-programmet ditt i både interaktiv modus og skriptmodus!

UTFORDRING: NY MELDING

Kan du endre meldingen Python-programmet skriver ut som utdata? Når du skal legge til flere meldinger – vil du bruke interaktiv modus eller skriptmodus? Hva skjer hvis du fjerner parentesene eller anførselstegnene fra programmet og deretter prøver å kjøre det igjen?

Neste trinn: løkker og kodeinnrykk

På samme måte som Scratch bruker stabler med puslespillignende klosser for å kontrollere hvilke biter som hører til andre biter av programmet, har Python sin egen metode for å kontrollere sekvensen programmene kjører i: *innrykk*. Opprett et nytt program ved å klikke på New-ikonet på verktøylinjen i Thonny. Du mister ikke det eksisterende programmet. Thonny oppretter i stedet en ny fane over skriptområdet. Start med å skrive inn følgende:

```
print("Løkke starter!")
for i in range(10):
```

Den første linjen skriver ut en enkel melding til skallet, akkurat som i programmet «Hello, World!» (Hei, verden!). Den andre starter en *endelig* løkke, som fungerer på samme måte som i Scratch: En teller, `i`, tilordnes løkken og gis en rekke tall – `range`-instruksjonen. Denne får beskjed om å starte med tallet 0 og jobbe seg oppover mot, men ikke helt fram til, tallet 10 – for å telle. Kolonsymbolet (`:`) forteller Python at neste instruksjon skal være en del av løkken.

I Scratch er instruksjonene som skal inkluderes i løkken, bokstavelig talt inkludert i den C-formede klossen. Python bruker en annen metode: en innrykkskode. Neste linje starter med fire mellomrom, som Thonny skal ha lagt til da du trykket på **ENTER** etter linje 2:

```
    print("Løkkenummer", i)
```

Mellomrommene skyver linjen inn i forhold til de andre linjene. Innrykket forteller Python hvilke instruksjoner som er utenfor løkken og hvilke som er inne i løkken. Den innrykkede koden er *nestet*.

Når du trykker på **ENTER** på slutten av tredje linje, vil du se at Thonny automatisk setter inn et innrykk på neste linje. Programmet går ut fra at det er en del av løkken. Hvis du vil fjerne innrykket, trykker du på **TILBAKE**-tasten én gang før du skriver fjerde linje:

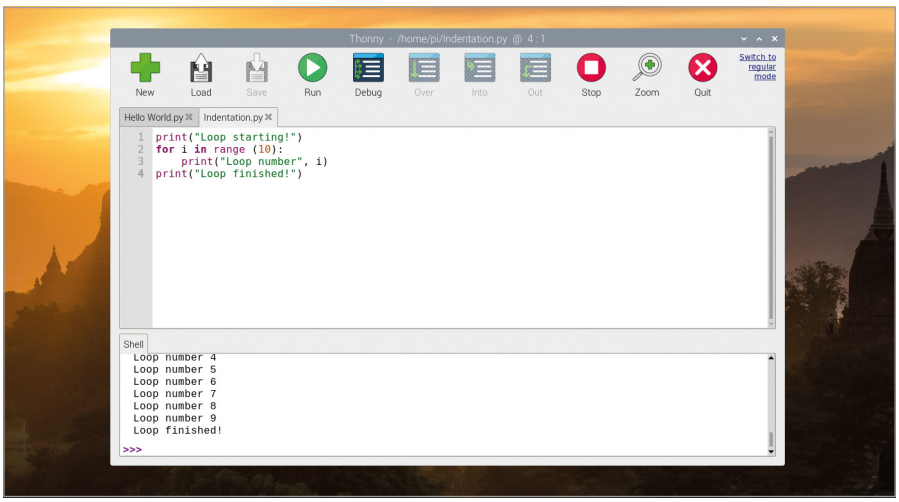
```
print("Løkke slutter!")
```

Programmet på fire linjer er nå ferdig. Den første linjen ligger utenfor løkken og blir bare kjørt én gang. Den andre linjen setter opp løkken, den tredje ligger inne i løkken og kjøres én gang hver gang løkken starter, og den fjerde linjen ligger utenfor løkken.

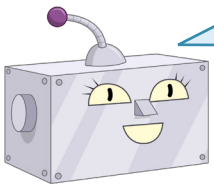
```
print("Løkke starter!")
for i in range(10):
    print("Løkkenummer", i)
print("Løkke slutter!")
```

Klikk på Run-ikonet, lagre programmet som **Innrykk** og vis resultatet for skallområdet (**Figur 5-3**):

```
Løkke starter!
Løkkenummer 0
Løkkenummer 1
Løkkenummer 2
Løkkenummer 3
Løkkenummer 4
Løkkenummer 5
Løkkenummer 6
Løkkenummer 7
Løkkenummer 8
Løkkenummer 9
Løkke slutter!
```



Figur 5-3: Utføre en løkke



TELLING FRA NULL

Python er et nullindeksert språk, noe som betyr at det begynner å telle fra 0, ikke fra 1. Derfor skriver programmet ut tallene 0 til 9 i stedet for 1 til 10. Hvis du vil, kan du endre denne atferden ved å bytte ut instruksjonen `range(10)` med `range(1, 11)` – eller andre tall.

Innrykk er en viktig del av Python – og en av de vanligste årsakene til at et program ikke fungerer slik det skal. Når du leter etter problemer i et program, en prosess kjent som *feilsøking*, må du alltid dobbeltsjekke innrykk – særlig når du begynner å neste løkker inni andre løkker.

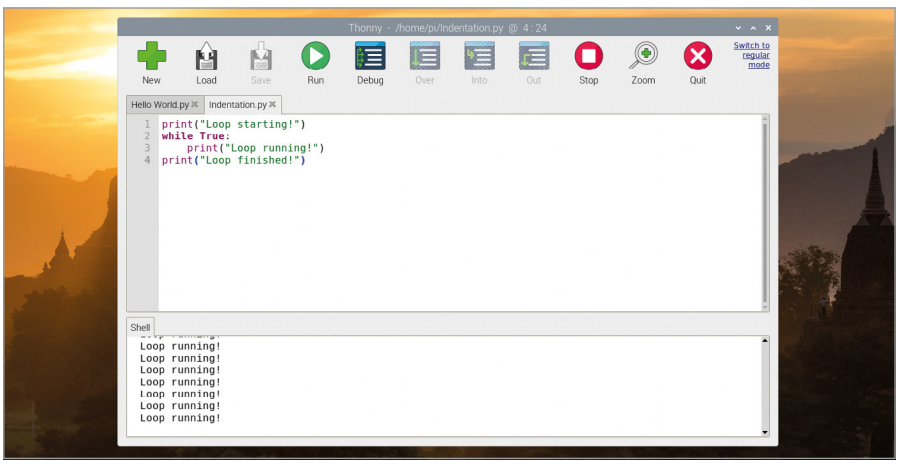
Python støtter også *uendelige* løkker, som kjører i det uendelige. Hvis du vil endre programmet fra en endelig løkke til en uendelig løkke endrer du linje 2 til:

```
while True:
```

Hvis du klikker på Run-ikonet nå, ser du denne feilmeldingen: **`name 'i' is not defined`**. Det er fordi du har slettet linjen som opprettet og tilordnet en verdi til variabelen **`i`**. Du kan løse dette problemet ved å redigere linje 3 slik at den ikke lenger bruker variabelen:

```
    print("Løkke kjører!")
```

Klikk på Run-ikonet. Hvis du er rask, vil du se meldingen «Løkke starter!» etterfulgt av en uendelig streng med meldinger av typen «Løkke starter» (**Figur 5-4**). Meldingen «Løkke slutter!» skrives aldri ut siden løkken ikke har noen slutt. Hver gang Python er ferdig med å skrive ut meldingen «Løkke kjører!», går den tilbake til begynnelsen av løkken og skriver den ut igjen.



▲ **Figur 5-4:** En uendelig løkke fortsetter til du stopper programmet

Klikk på Stop-ikonet på Thonny-verktøylinjen for å be programmet om å stoppe det det gjør – det vil si avbryte programmet. Du ser en melding i Python-skallområdet, og programmet stopper – uten å nå linje 4.

UTFORDRING: LØKKE I LØKKE

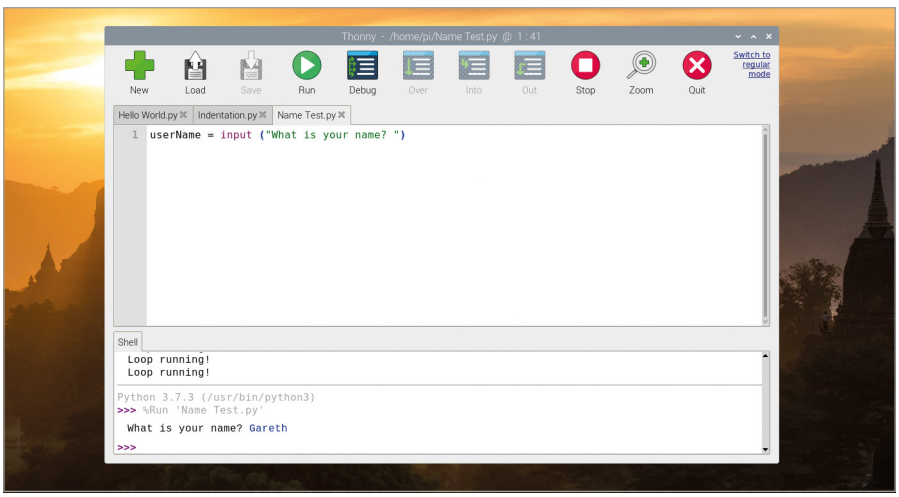
Kan du endre løkken tilbake til en endelig løkke? Kan du legge til en ny endelig løkke i programmet? Hvordan legger du til en løkke i en løkke, og hvordan tror du den vil fungere?

Betingelser og variabler

Som i alle programmeringsspråk finnes det variabler for mer enn bare å kontrollere løkker. Start et nytt program ved å klikke på New-ikonet på verktøylinjen i Thonny. Deretter skriver du inn følgende i skriptområdet:

```
userName = input ("Hva heter du?")
```

Klikk på Run-ikonet, lagre programmet som **Navnetest**, og se hva som skjer i skallområdet. Du blir bedt om å oppgi navnet ditt. Skriv inn navnet i skallområdet, og trykk deretter på **ENTER**. Fordi det er den eneste instruksjonen i programmet, vil det ikke skje noe mer (**Figur 5-5**). Hvis du faktisk vil gjøre noe med dataene du har plassert i variabelen, må du ha flere linjer i programmet.



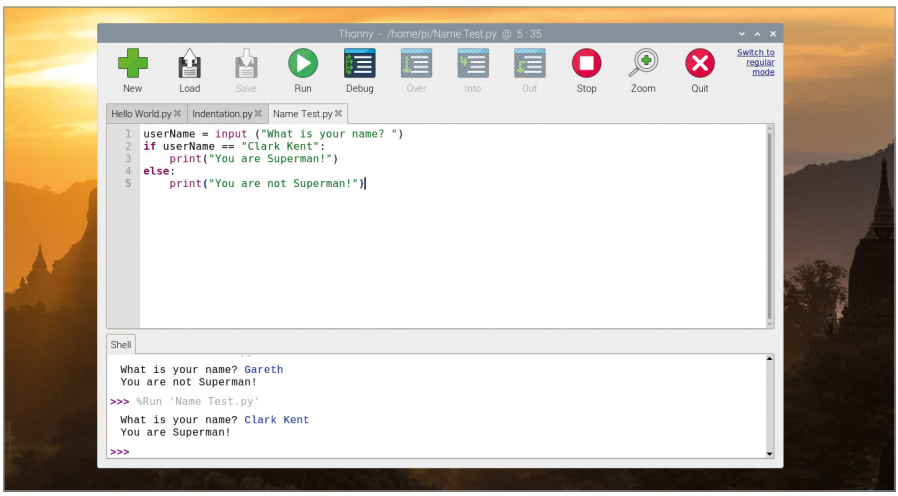
Figur 5-5: Med funksjonen `input` kan du be en brukerom å legge inn tekst

Hvis du vil at programmet skal gjøre noe nyttig med navnet, legger du til en *betinget setning* ved å skrive:

```
if userName == "Clark Kent":
    print("Du er Supermann!")
else:
    print("Du er ikke Supermann!")
```

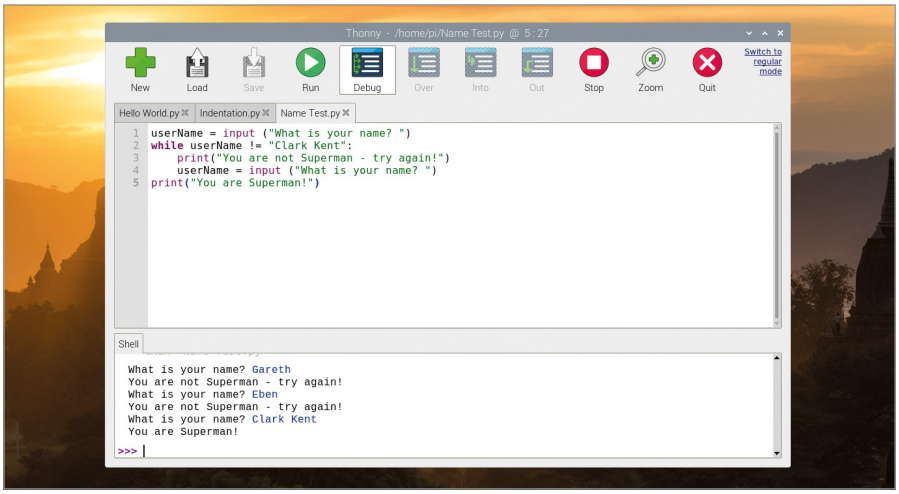
Merk: Når Thonny ser at koden din skal innrykkes, vil den gjøre det automatisk. Derimot vet ikke programmet når koden ikke skal innrykkes lenger, så du må slette mellomrommene selv.

Klikk på Run-ikonet og skriv navnet ditt i skallområdet. Med mindre navnet ditt tilfeldigvis er Clark Kent, vil du se meldingen «Du er ikke Supermann!». Klikk på Run igjen, og denne gangen skriver du «Clark Kent». Pass på at du skriver det nøyaktig som i programmet, med stor C og K. Denne gangen anerkjenner programmet at du faktisk er Supermann (**Figur 5-6**).

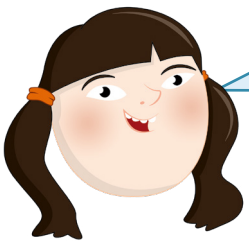


▲ **Figur 5-6:** Burde ikke du vært ute og reddet verden?

Symbolene **==** forteller Python at det skal foreta en direkte sammenligning for å se om variabelen **userName** passer med teksten – kalt en *streng* – i programmet. Hvis du jobber med tall, kan du utføre andre sammenligninger: **>** for å se om et tall er større enn et annet tall, **<** for å se om det er mindre, **=>** for å se om det er lik eller større, **=<** for å se om det er lik eller mindre enn det andre tallet. Du kan også bruke **!=**, som betyr at det ikke er lik – det er det stikk motsatte av **==**. Disse symbolene er teknisk kjent som *sammenligningsoperatorer*.



Figur 5-7: Det vil fortsette å be om navnet ditt til du sier at det er «Clark Kent»



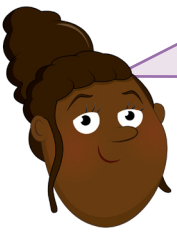
BRUKE = OG ==

Når du bruker variabler, er det viktig å vite forskjellen mellom = og ==. Husk: = betyr «la denne variabelen være lik denne verdien», mens == betyr «sjekk om denne variabelen er lik denne verdien». Hvis du blander dem sammen, vil du få et program som ikke fungerer!

Sammenligningsoperatorer kan også brukes i løkker. Slett linje 2 til 5, og skriv inn følgende i stedet:

```
while userName != "Clark Kent":
    print("Du er ikke Supermann!")
    userName = input ("Hva heter du?")
print("Du er Supermann!")
```

Klikk på Run-ikonet igjen. Denne gangen, i stedet for å slutte, vil programmet fortsette å spørre deg hva du heter til det bekrefter at du er Supermann (**Figur 5-7**) – nesten som et enkelt passord. For å komme deg ut av løkken skriver du enten «Clark Kent» eller klikker på Stop-ikonet på Thonny-verktøylinjen. Gratulerer! Nå vet du hvordan du bruker betingelser og sammenligningsoperatorer!



UTFORDRING: LEGGE TIL FLERE SPØRSMÅL

Kan du endre programmet til å stille flere enn ett spørsmål og lagre svarene i flere variabler? Kan du lage et program som bruker betingelses- og sammenligningsoperatorer til å skrive ut om et tall som angis av brukeren, skal være høyere eller lavere enn 5, på samme måte som programmet du opprettet i **Kapittel 4: Programmering med Scratch**?

Prosjekt 1: Turtle-snøfnugg

Nå som du vet hvordan Python fungerer, er det på tide å leke seg med litt grafikk. Vi skal prøve å lage et snøfnugg ved hjelp av et verktøy som kalles en *turtle* (skillpadde).

NETTPROSJEKT

Dette prosjektet er også tilgjengelig på nettet under **rpf.io/turtle-snowflakes**

Skilpadder var opprinnelige fysiske roboter som har samme form som selve dyret de er oppkalt etter. De er utviklet for å bevege seg i en rett linje, snu og heve og senke en penn. I den digitale versjonen betyr det bare å starte eller slutte å tegne en linje når den beveger seg. I motsetning til visse andre språk, for eksempel Logo og dens mange varianter, er ikke turtle-verktøyet innebygd i Python. I stedet kommer programmet med et *bibliotek* av tilleggskoder. Biblioteker er kodebunter som legger til nye instruksjoner for å utvide funksjonene i Python. De legges inn i programmene dine ved hjelp av en importkommando.

Opprett et nytt program ved å klikke på New-ikonet, og skriv deretter følgende:

```
import turtle
```

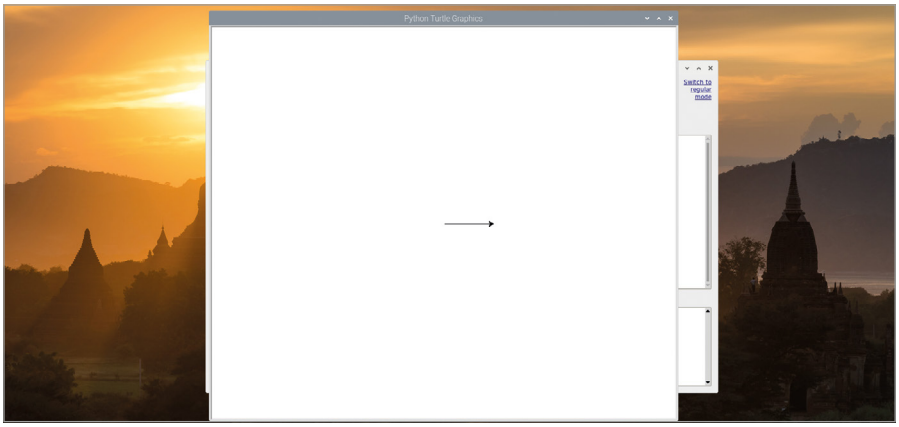
Når du bruker instruksjoner som er inkludert i et bibliotek, må du skrive biblioteknavnet etterfulgt av et punktum og deretter instruksjonsnavnet. Det kan være irriterende å skrive hele ordet hver gang, så du kan tilordne det et kortere variabelnavn i stedet. Du kan bruke bare én bokstav, men vi tenkte det kunne være en god idé å bruke det engelske kjæledyrnavnet for skilpadde, «Pat». Skriv følgende:

```
pat = turtle.Turtle()
```

For å teste programmet må du gi skilpadden noe å gjøre. Skriv følgende:

```
pat.forward(100)
```

Klikk på Run-ikonet og lagre programmet som **Turtle-snøfnugg**. Når programmet er lagret, åpnes vinduet «Turtle Graphics». Der kan du se resultatet av programmet ditt: skilpadden, Pat, beveger seg fremover 100 enheter og tegner en rett linje (**Figur 5-8**).



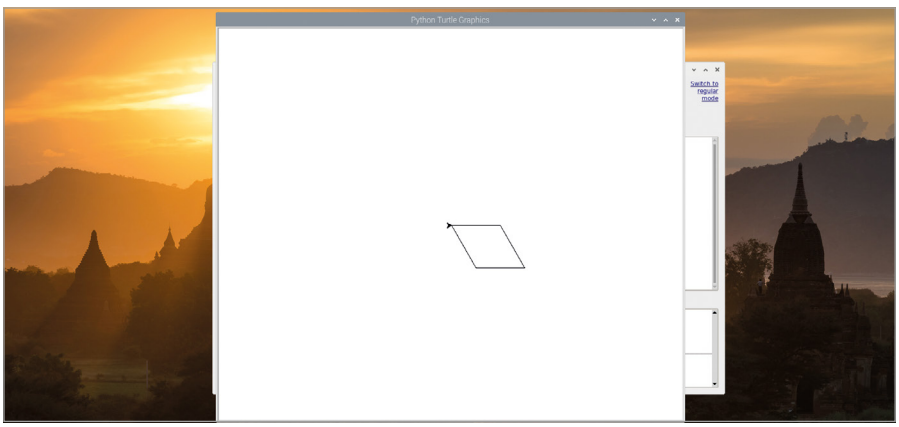
Figur 5-8: Skilpadden beveger seg fremover for å tegne en rett linje

Gå tilbake til hovedvinduet i Thonny – hvis det er skjult bak vinduet Turtle Graphics, kan du enten klikke på minimeringsknappen i Turtle Graphics-vinduet eller klikke på Thonny-oppføringen på oppgavelinjen øverst på skjermen. Klikk på Stop-knappen for å lukke Turtle Graphics-vinduet.

Det ville være kjedelig å skrive inn hver bevegelsesinstruksjon for hånd, så du kan slett linje 3 og lage en løkke som fjerner den slitsomme jobben med å lage figurer:

```
for i in range(2):
    pat.forward(100)
    pat.right(60)
    pat.forward(100)
    pat.right(120)
```

Kjør programmet, så tegner Pat et enkelt parallellogram (**Figur 5-9**).



Figur 5-9: Du kan tegne figurer ved å kombinere vendinger og bevegelser

Hvis du vil gjøre det om til en snøfnugglignende figur, klikker du på Stop-ikonet i hovedvinduet i Thonny og lager en løkke rundt løkken ved å legge til følgende linje som linje 3:

```
for i in range(10):
```

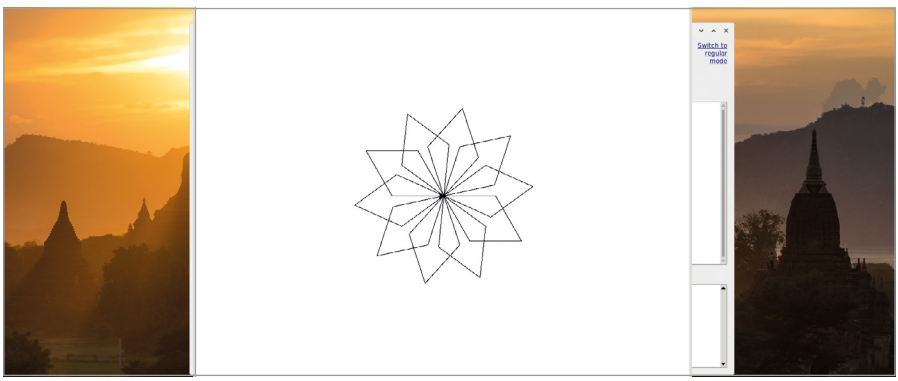
… og følgende nederst i programmet:

```
    pat.right(36)
```

Programmet kjører ikke slik det er, fordi den eksisterende løkken ikke er riktig innrykket. Du løser problemet ved å klikke på starten av hver linje i den eksisterende løkken – linje 4 til 8 – og trykke fire ganger på **MELLOMROMSTASTEN** for å korrigere innrykket. Programmet ditt skal nå se slik ut:

```
import turtle
pat = turtle.Turtle()
for i in range(10):
    for i in range(2):
        pat.forward(100)
        pat.right(60)
        pat.forward(100)
        pat.right(120)
    pat.right(36)
```

Klikk på Run-ikonet og se hva skilpadden (turtle) gjør. Verktøyet tegner et parallellogram som før. Deretter snur det imidlertid 36 grader og tegner ett til, deretter ett til og så videre til det finnes ti overlappende parallellogrammer på skjermen. Resultatet kan minne om et snøfnugg (**Figur 5-10**).

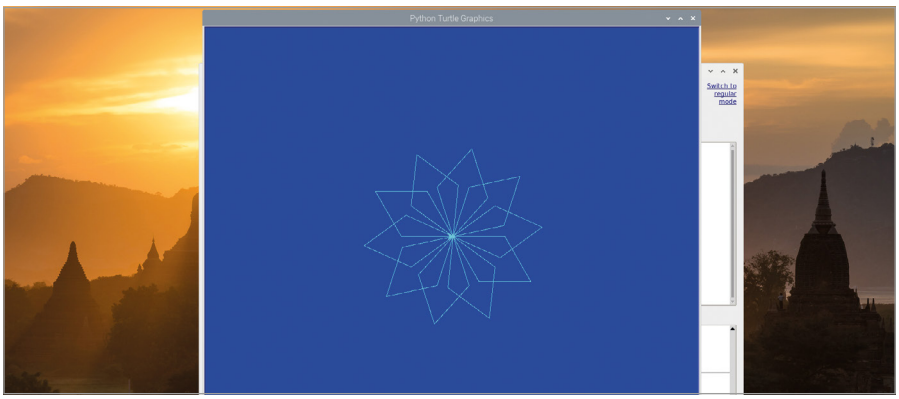


▲ **Figur 5-10:** Gjenta figuren for å lage en mer kompleks figur

Mens en robotskilpadde tegner i én farge på et stort papirark, kan Pythons simulerte skilpadde (turtle) bruke mange farger. Legg til en ny linje 3 og 4, noe som skyver de eksisterende linjene nedover:

```
turtle.Screen().bgcolor("blue")
pat.color("cyan")
```

Kjør programmet på nytt for å se resultatet av den nye koden. Bakgrunnsfargen i Turtle Graphics-vinduet er endret til blått, og snøfnugget er nå i cyan (**Figur 5-11**).



Figur 5-11: Endre farge på bakgrunnen og snøfnugget

Du kan også angi at fargene skal velges tilfeldig fra en liste ved hjelp av `random`-biblioteket. Gå tilbake til starten av programmet og sett inn følgende som linje 2:

```
import random
```

Endre bakgrunnsfargen i det som nå er linje 4 fra blå til grå, og lag deretter en ny variabel kalt «colours» ved å sette inn en ny linje 5:

```
colours = ["cyan", "purple", "white", "blue"]
```

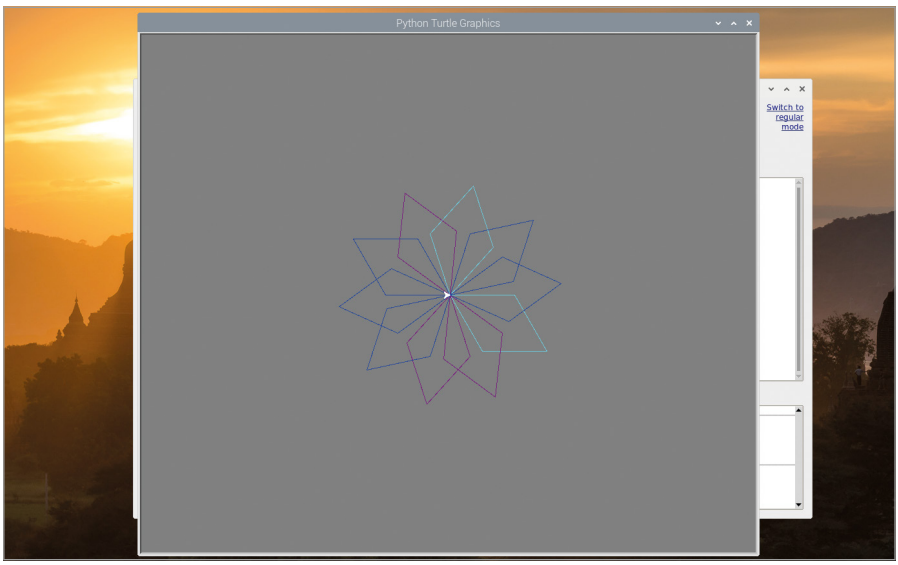
AMERIKANSK-ENGELSK STAVEMÅTE

Mange programmeringsspråk bruker amerikansk-engelsk stavemåte, og Python er ikke noe unntak. Kommandoen for å endre fargen på skilpaddens penn staves *color*, og hvis du staver på britisk engelsk som *colour*, vil det ikke fungere. Variabler kan imidlertid staves slik du vil. Dermed kan du kalle den nye variabelen *colours* og få Python til å forstå den.

Denne typen variabler er kjent som en liste og merkes med hakeparenteser. I dette tilfellet er listen fylt med mulige farger for snøfnuggsegmentene – men du må fortsatt be Python om å velge farge hver gang løkken gjentas. Helt på slutten av programmet skriver du inn følgende – husk innrykket med fire mellomrom slik at det inngår i den ytre løkken, akkurat som linjen over den:

```
    pat.color(random.choice(colours))
```

Klikk på Run-ikonet, så blir snøfnugg-/ninjastjernen tegnes på nytt. Men denne gangen velger Python en tilfeldig farge fra listen når den tegner hvert «kronblad» – noe som gir snøfnuggene et flott, flerfarget utseende (**Figur 5-12**).



▲ **Figur 5-12:** Bruke tilfeldige farger på «kronbladene»

Hvis du vil at snøfnugget skal ligne mindre på en ninjastjerne og mer på et ordentlig snøfnugg, legger du til en ny linje 6, rett under listen `colours`. Deretter skriver du følgende:

```
pat.penup()
pat.forward(90)
pat.left(45)
pat.pendown()
```

Instruksjonene **penup** og **pendown** fører en fysisk penn utenfor og på papiret hvis du bruker en skilpadderobot. I den virtuelle verden ber du bare turtle om å slutte og starte tegning av streker. I stedet for å bruke en løkke skal du nå opprette en *funksjon* – et kodesegment som du når som helst kan hente opp – som å lage din egen Python-instruksjon.

Start med å slette koden for å tegne de parallellogrambaserte snøfnuggene. Det vil si alt fra og med instruksjonen **`pat.color("cyan")`** på linje 10 til og med **`pat.right(36)`** på linje 17. La instruksjonen **`pat.color(random.choice(colours))`** stå, men legg til en hash-kode (**#**) på begynnelsen av linjen. Dette kalles å *kommentere ut* en instruksjon, noe som betyr at Python overser den. Du kan bruke kommentarer til å legge til forklaringer i koden. Da blir det enklere å forstå den flere måneder senere eller når du skal sende den til andre!

Opprett funksjonen, som kalles «branch», ved å skrive følgende instruksjon på linje 10, under **`pat.pendown():`**

```
def branch():
```

Dette *definerer* funksjonen **`branch`**. Når du trykker på **ENTER**-tasten, legger Thonny automatisk til innrykk for funksjonsinstruksjonene. Skriv følgende og husk å være nøye med innrykk – på ett tidspunkt skal du nemlig neste koder med tre innrykknivåer!

```
    for i in range(3):
        for i in range(3):
            pat.forward(30)
            pat.backward(30)
            pat.right(45)
        pat.left(90)
        pat.backward(30)
        pat.left(45)
    pat.right(90)
    pat.forward(90)
```

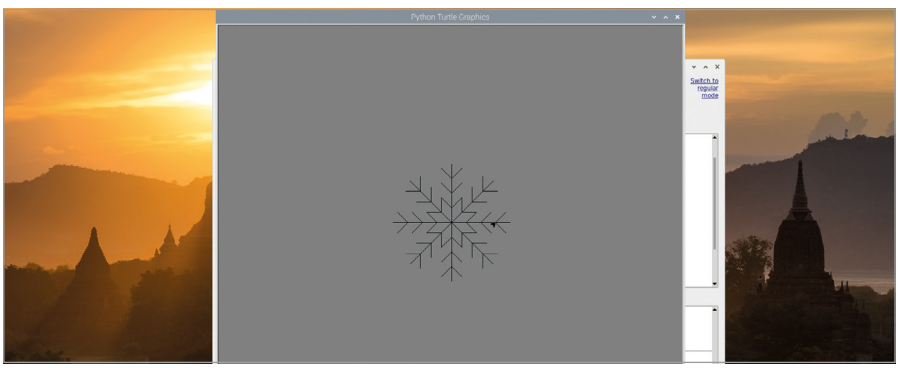
Til slutt oppretter du en ny løkke nederst i programmet, men over den utkommenterte fargelinjen – for å kjøre eller *hente* den nye funksjonen:

```
for i in range(8):
    branch()
    pat.left(45)
```

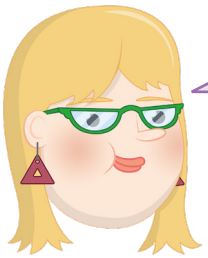
Det ferdige programmet skal nå se slik ut:

```
import turtle
import random
pat = turtle.Turtle()
turtle.Screen().bgcolor("grey")
colours = ["cyan", "purple", "white", "blue"]
pat.penup()
pat.forward(90)
pat.left(45)
pat.pendown()
def branch():
    for i in range(3):
        for i in range(3):
            pat.forward(30)
            pat.backward(30)
            pat.right(45)
        pat.left(90)
        pat.backward(30)
        pat.left(45)
    pat.right(90)
    pat.forward(90)
for i in range(8):
    branch()
    pat.left(45)
#    pat.color(random.choice(colours))
```

Klikk på Run og observer grafikkvinduet mens Pat tegner etter dine instruksjoner. Gratulerer! Nå ser snøfnugget mye mer ut som et snøfnugg (**Figur 5-13**)!



▲ **Figur 5-13:** Bruk av ekstra «branches» (grener) får det til å se ut som et snøfnugg



UTFORDRING: HVA NÅ?

Kan du bruke den utkommenterte instruksjonen til å tegne grenene av snøfnugget i forskjellige farger? Kan du lage en «snøfnugg»-funksjon, og bruke den til å tegne mange snøfnugg på skjermen? Kan du få programmet til å endre størrelsen og fargen på snøfnuggene vilkårlig?

Prosjekt 2: Skummel Finn feilen

I tillegg til turtle-basert grafikk kan Python også håndtere bilder og lyder. Det kan være flott hvis du vil lure vennene dine – spill Finn feilen med en skummel overraskelse innebygd, perfekt for Halloween!

NETTPROSJEKT

Dette prosjektet er også tilgjengelig på nettet under **rpf.io/scary-spot**

Til dette prosjektet trenger du to bilder – Finn feilen-bildet pluss et skummelt overraskelsesbilde – og en lydfil. Klikk på Raspberry-ikonet for å laste inn Raspberry Pi OS-menyen. Velg Internett og klikk på nettleseren Chromium. Når nettleseren er lastet inn, skriver du **rpf.io/astronaut-backdrop** i adressefeltet og trykker på **ENTER**. Høyreklikk på bildet og klikk på «Lagre bilde som…». Velg Hjem-mappen fra listen til høyre og klikk på Lagre. Klikk på Tilbake på adresselinjen i Chromium, skriv **rpf.io/scary-pic**, og trykk deretter på **ENTER**. Som før høyreklikker du på bildet og klikker på «Lagre bilde som…». Velg hjemmemappen og klikk på Lagre.

Klikk på Tilbake på adresselinjen for å finne lydfilen du vil bruke, skriv **rpf.io/scream**, og trykk deretter på **ENTER**. Denne filen spiller av et skrik som vil gi spilleren et skikkelig støkk. Filen spilles av automatisk, men må lagres før du kan bruke den. Høyreklikk på den lille lydspilleren. Klikk på «Lagre som…», velg hjemmemappen og klikk på Lagre. Deretter kan du lukke Chromium-vinduet.

Klikk på New-ikonet på verktøylinjen i Thonny for å starte et nytt prosjekt. Som før bruker du et bibliotek for å utvide mulighetene i Python: Pygame-biblioteket, som er opprettet med tanke på spill. Skriv følgende:

```
import pygame
```

Du trenger noen deler fra andre biblioteker, men også fra en underinndeling i Pygame-biblioteket. Importer disse ved å skrive følgende:

```
from pygame.locals import *
from time import sleep
from random import randrange
```

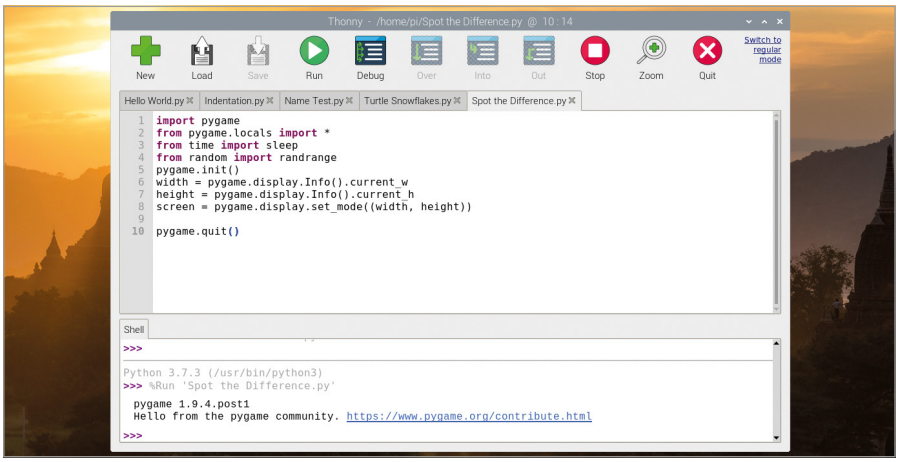
Instruksjonen **`from`** fungerer annerledes enn **`import`**. Du kan importere bare deler av et bibliotek du trenger, i stedet for hele biblioteket. Deretter må du konfigurere Pygame. Dette kalles *initialisering*. Pygame må vite bredden og høyden på spillerens skjerm eller TV, kjent som *oppløsningen*. Skriv følgende:

```
pygame.init()
width = pygame.display.Info().current_w
height = pygame.display.Info().current_h
```

Det siste trinnet når du skal sette opp et Pygame, er å opprette vinduet, som Pygame kaller screen. Skriv følgende:

```
screen = pygame.display.set_mode((width, height))

pygame.quit()
```

Legg merke til den tomme linjen i midten. Det er her programmet ditt skal legges inn. Men foreløpig klikker du bare på Run-ikonet, lagrer programmet som **Finn feilen** og ser hva som skjer: Pygame oppretter et vindu og fyller det med en svart bakgrunn, som nesten forsvinner umiddelbart når det kommer til instruksjonen om å avslutte (quit). Bortsett fra en kort melding i skallet (**Figur 5-14**), har ikke programmet utrettet særlig mye.



▲ **Figur 5-14:** Programmet fungerer, men gjør ikke så mye ennå

Hvis du vil vise Finn feilen-bildet skriver du inn følgende linje i området over **`pygame.quit()`**:

```
difference = pygame.image.load('spot_the_diff.png')
```

Hvis du vil at bildet skal fylle skjermen, må du skalere det i forhold til skjermens eller tv-ens oppløsning. Skriv følgende:

```
difference = pygame.transform.scale(difference, (width, height))
```

Nå som bildet ligger i minnet, må du fortelle Pygame at det skal vises på skjermen – en prosess som kalles *blitting* (gjennomsiktig punktgrafikk), eller *bitblokkoverføring*. Skriv følgende:

```
screen.blit(difference, (0, 0))
pygame.display.update()
```

Den første av disse linjene kopierer bildet til skjermen og begynner øverst til venstre på skjermen. Den andre ber Pygame om å tegne skjermen på nytt. Uten denne andre linjen ville bildet ligge på riktig sted i minnet, men du ville aldri se det!

Klikk på Run-ikonet, så vises bildet kort på skjermen (**Figur 5-15**).

▲ **Figur 5-15:** Ditt Finn feilen-bilde

Hvis du vil se det lengre, legger du til følgende linje like over **`pygame.quit()`**:

```
sleep(3)
```

Klikk på Run en gang til, så blir bildet værende på skjermen lengre. Legg til overraskelsesbildet ved å skrive følgende rett under linjen **`pygame.display.update()`:**

```
zombie = pygame.image.load('scary_face.png')
zombie = pygame.transform.scale(zombie, (width, height))
```

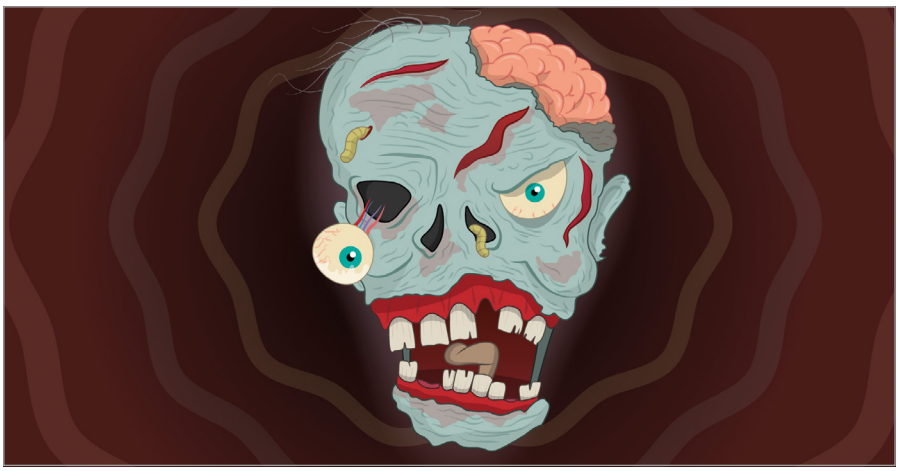
Legg til en forsinkelse, slik at zombiebildet ikke vises med én gang:

```
sleep(3)
```

Deretter blitter du bildet til skjermen og oppdaterer den slik at spilleren kan se det:

```
screen.blit(zombie, (0,0))
pygame.display.update()
```

Klikk på Run-ikonet og se hva som skjer: Pygame laster inn Finn feilen-bildet, men etter tre sekunder erstattes det av den skumle zombien (**Figur 5-16**).



▲ **Figur 5-16:** Noen kommer til å bli skikkelig skremt

En forsinkelse satt til tre sekunder, gjør det mer forutsigbart. Endre linjen **`sleep(3)`** over **`screen.blit(zombie, (0,0))`** til:

```
sleep(randrange(5, 15))
```

Da velges et tilfeldig tall mellom 5 og 15, og programmet forsinkes tilsvarende. Deretter legger du til følgende linje rett over **`sleep`**-instruksjonen for å laste inn lydfilen med skriket:

```
scream = pygame.mixer.Sound('scream.wav')
```

Nedenfor **`sleep`**-instruksjonen skriver du følgende på en ny linje for å starte lydavspillingen,

slik at den starter like før det skumle bildet vises for spilleren:

```
scream.play()
```

Til slutt ber du Pygame om å slutte å spille av lyden ved å skrive følgende linje like over **`pygame.quit()`**:

```
scream.stop()
```

Klikk på Run-ikonet og beundre verket ditt. Etter noen sekunder med uskyldig spillmoro med «Spot the difference» dukker den skumle zombien opp sammen med et hårreisende skrik – som garantert vil skremme vennene dine! Hvis zombiebildet vises før lyden spilles av, kan du kompensere ved å legge inn en liten forsinkelse like etter **`scream.play()`**-instruksjonen og før **`screen.blit`**-instruksjonen:

```
sleep(0.4)
```

Det ferdige programmet skal nå se slik ut:

```
import pygame
from pygame.locals import *
from time import sleep
from random import randrange
pygame.init()
width = pygame.display.Info().current_w
height = pygame.display.Info().current_h
screen = pygame.display.set_mode((width, height))
difference = pygame.image.load('spot_the_diff.png')
difference = pygame.transform.scale(difference, (width, height))
screen.blit(difference, (0, 0))
pygame.display.update()
zombie = pygame.image.load('scary_face.png')
zombie = pygame.transform.scale(zombie, (width, height))
scream = pygame.mixer.Sound('scream.wav')
sleep(randrange(5, 15))
scream.play()
screen.blit(zombie, (0,0))
pygame.display.update()
sleep(3)
scream.stop()
pygame.quit()
```

Nå er det bare å invitere vennene dine til å spille Finn feilen. Du må selvsagt huske å skru opp volumet på høyttalerne!

UTFORDRING: ENDRE UTSEENDET

Kan du endre bildene slik at spøken passer bedre til andre anledninger, for eksempel i julen? Kan du tegne ditt eget Finn feilen-bilde og skumle bilder (ved hjelp av et grafisk redigeringsprogram som GIMP)? Kan du spore brukere som klikker på en feil – for å gjøre det mer overbevisende?

Prosjekt 3: RPG-labyrint

Nå som du har blitt kjent med Python, er det på tide å lage noe mer komplisert med Pygame – et tekstbasert labyrintspill med full funksjonalitet, basert på klassiske rollespill. Disse spillene kalles teksteventyr, eller interaktiv fiksjon, og skriver seg fra da datamaskiner ikke klarte å håndtere grafikk. De har fremdeles sine tilhengere, som hevder at ingen grafikk kan måle seg med bildene i fantasien din.

NETTPROSJEKT

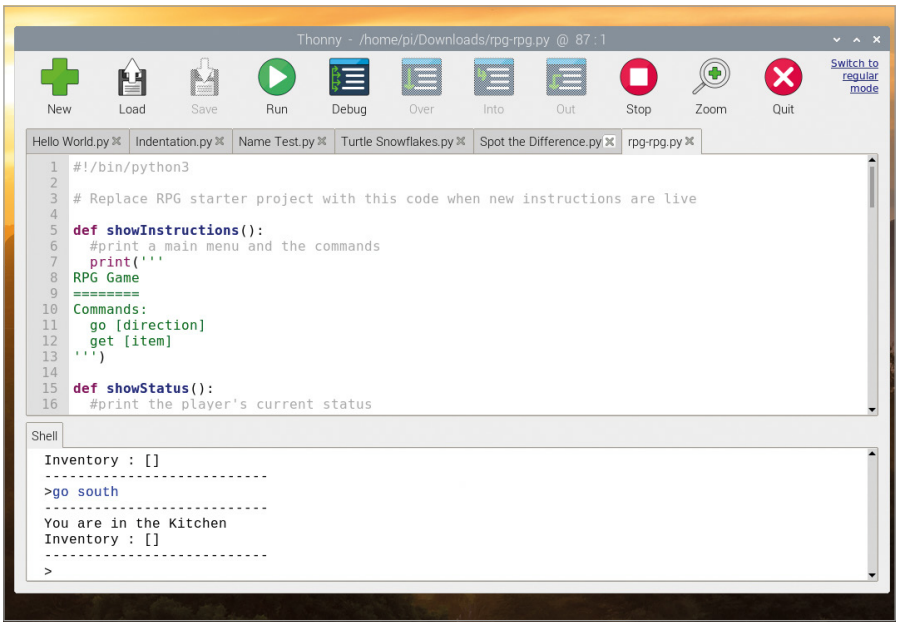
Dette prosjektet er også tilgjengelig på nettet under **rpf.io/python-rpg**

Dette programmet er ganske komplisert i forhold til andre i dette kapitlet. For å gjøre det enklere starter vi derfor med en versjon som allerede er delvis skrevet. Åpne Chromium og gå til følgende adresse: **rpf.io/rpg-code**.

Chromium laster automatisk ned koden for programmet til mappen Nedlastinger, men du advares om at filtypen – et Python-program – kan skade datamaskinen. Du har lastet ned filen fra Raspberry Pi Foundation, som er en sikker kilde. Derfor kan du trykke på Keep-knappen i advarselsmeldingen som vises nederst på skjermen. Gå tilbake til Thonny, og klikk deretter på Load-ikonet. Finn filen, **rpg-rpg.py**, i mappen Nedlastinger og klikk på Load-knappen.

Start med å klikke på Run-ikonet for å gjøre deg kjent med hvordan et teksteventyr fungerer. Resultatet av spillet vises i skallområdet nederst i Thonny-vinduet. Du kan utvide Thonny-vinduet ved å klikke på maksimeringsknappen for å gjøre det lettere å lese.

Slik spillet ser ut nå, er det veldig enkelt. Det er to rom og ingen gjenstander. Spilleren starter i Hall (hallen), det første av de to rommene. Hvis du vil gå til Kitchen (kjøkkenet), skriver du bare «go south» og klikker på **ENTER** (**Figur 5-17**). Når du er i Kitchen, kan du skrive «go north» for å gå tilbake til Hall. Du kan også prøve å skrive «go west» og «go east», men siden det ikke er noen rom i disse retningene, vil du se en feilmelding.



Figur 5-17: Det finnes bare to rom så langt

Bla ned til linje 29 i programmet i skriptområdet for å finne en variabel som heter **`rooms`**. Denne typen variabler kalles en *ordliste* og forteller spillet om rom, utganger og hvilket rom en bestemt utgang fører til.

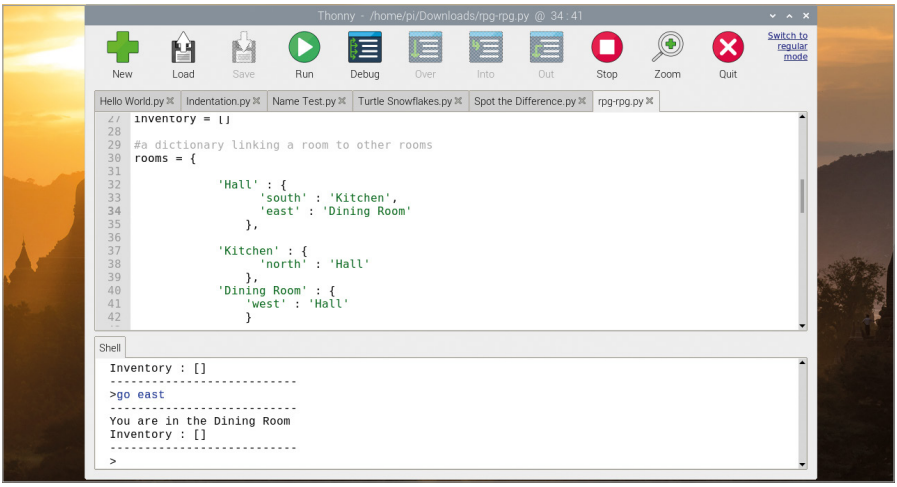
For å gjøre spillet mer interessant kan du legge til et annet rom: et Dining Room (spisestue), øst for hallen. Finn variabelen **`rooms`** i skriptområdet, og utvid den ved å legge inn et komma (**`,`**) etter **`}`**på linje 38. Deretter skriver du følgende (nøyaktige innrykk er ikke så viktig i en ordliste).

```
'Dining Room' : {
      'west' : 'Hall'
   }
```

Du trenger også en ny utgang i Hall, siden den ikke automatisk opprettes for deg. Gå til slutten av linje 33, legg til et komma og legg til følgende linje:

```
'east' : 'Dining Room'
```

Klikk på Run-ikonet, og prøv det nye rommet: skriv «go east» når du er i Hall, for å gå til Dining Room (**Figur 5-18**, på neste side), og skriv «go west» når du er i Dining Room, for å gå til Hall. Gratulerer! Du har laget et rom på egen hånd!

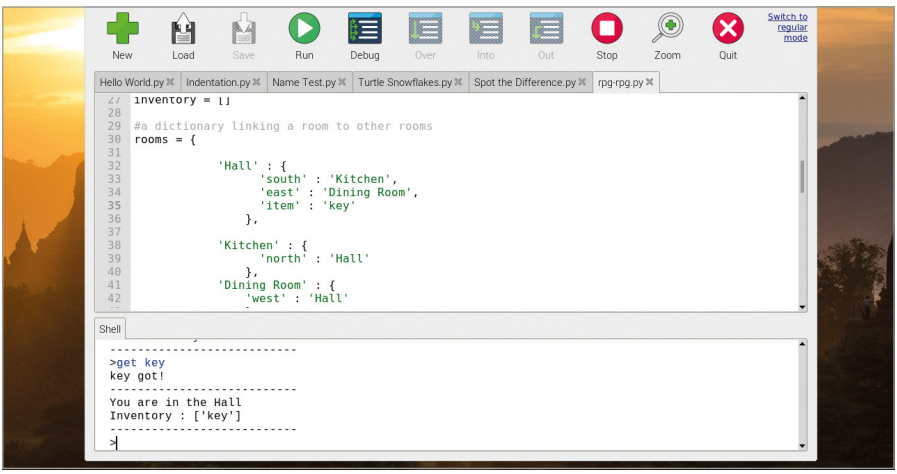


▲ **Figur 5-18:** Du har lagt til et nytt rom

Men det er ikke så morsomt med tomme rom. Hvis du vil legge til et element (item) i et rom, må du endre ordlisten for det rommet. Stopp programmet ved å klikke på Stop-ikonet. Finn **`Hall`**-ordlisten i skriptområdet, og legg til et komma på slutten av linjen **`'east' : 'Dining Room'`**. Deretter trykker du på **ENTER** og skriver følgende linje:

```
'item' : 'key'
```

Klikk på Run en gang til. Denne gangen vil spillet fortelle deg at du kan se det nye elementet: en key (nøkkel). Skriv «get key» (**Figur 5-19**), så kan du hente den og legge den til i listen over elementer du har med deg – det vil si ditt *lager*. Lageret blir med deg når du går fra rom til rom.



▲ **Figur 5-19:** Den innhentede nøkkelen legges til inventaret

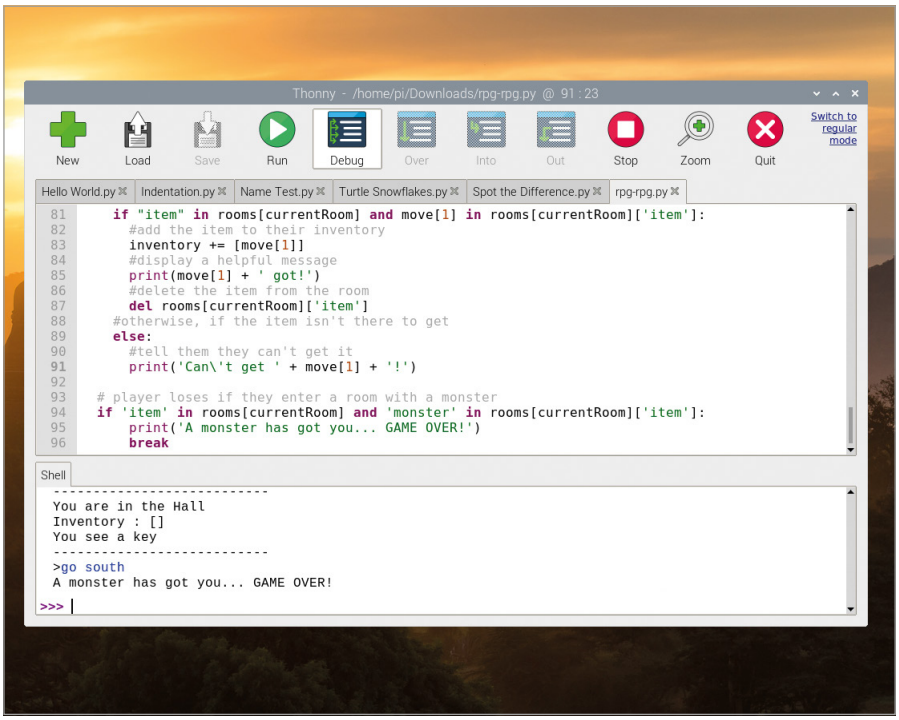
Klikk på Stop-ikonet , og gjør spillet mer interessant ved å legge til et monster som må unngås. Finn **`Kitchen`**-ordlisten, og legg til et «monster»-element på samme måte som du la til «key»-elementet. Husk å legge til et komma på slutten av linjen ovenfor:

```
'item' : 'monster'
```

Hvis du vil at monsteret skal kunne angripe spilleren, må du legge inn litt logikk i spillet. Rull helt til bunnen av programmet i skriptområdet, og legg til følgende linjer – inkludert kommentaren, merket med en hash-kode. Dette gjør det enklere å forstå programmet hvis du kommer tilbake til det en annen gang – husk å rykke inn linjene:

```
    # player loses if they enter a room with a monster
    if 'item' in rooms[currentRoom] and 'monster' in
rooms[currentRoom]['item']:
        print('A monster has got you... GAME OVER!')
        break
```

Klikk på Run, og prøv å gå inn i Kitchen (**Figur 5-20**) – monster blir ikke særlig fornøyd når du gjør det!



▲ **Figur 5-20:** Blås i rottene, det er et monster på kjøkkenet

For å gjøre dette eventyret til et skikkelig spill trenger du flere ting, et rom til og muligheten til å «vinne» ved å forlate huset med alle elementene trygt bevart i lageret. Start med å legge til et nytt rom, akkurat som du gjorde for Dining Room – bare at det denne gangen er det en Garden (hage). Legg til en utgang fra ordlisten Dining Room, og husk å legge til et komma på slutten av linjen ovenfor:

```
                'south' : 'Garden'
```

Legg deretter til det nye rommet i hovedordlisten for **`rooms`** – husk å legge til et komma etter **`}`** på linjen over som før:

```
            'Garden' : {
                'north' : 'Dining Room'
            }
```

Legg til objektet «potion» (trylledrikk) i ordlisten Dining Room, og husk å legge til det nødvendige kommaet på linjen over:

```
                'item' : 'potion'
```

Til slutt ruller du ned til bunnen av programmet. Legg til logikken som kreves for å sjekke om spilleren har alle elementene. Hvis det er tilfellet, sier du fra at spilleren har vunnet spillet:

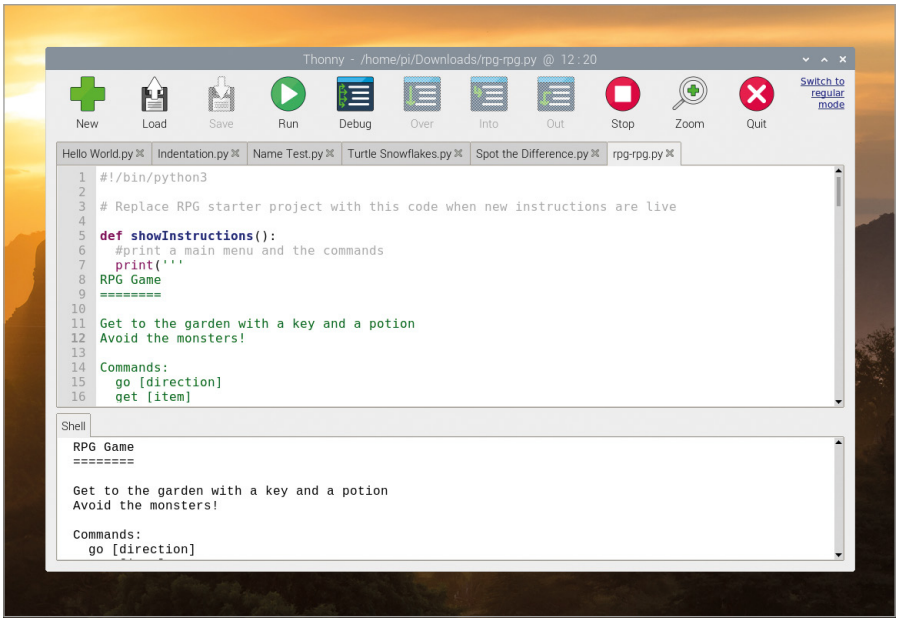
```
    # player wins if they get to the garden with a key and a potion
        if currentRoom == 'Garden' and 'key' in inventory and
'potion' in inventory:
            print('You escaped the house... YOU WIN!')
            break
```

Klikk på Run, og prøv å fullføre spillet ved å plukke opp «key» og «potion» før du går til Garden. Husk å ikke gå inn i Kitchen, for der er monster!

Som en siste finpuss kan du legge til noen instruksjoner som forteller spilleren hvordan spillet skal spilles. Rull til toppen av programmet, der funksjonen **`showInstructions()`** er definert, og legg til følgende:

```
Get to the Garden with a key and a potion
Avoid the monsters!
```

Kjør spillet en siste gang, så ser du at de nye instruksjonene vises helt fra starten av (**Figur 5-21**). Gratulerer! Du har laget et interaktivt tekstbasert labyrintspill!



Figur 5-21: Nå vet spilleren hva han/hun må gjøre

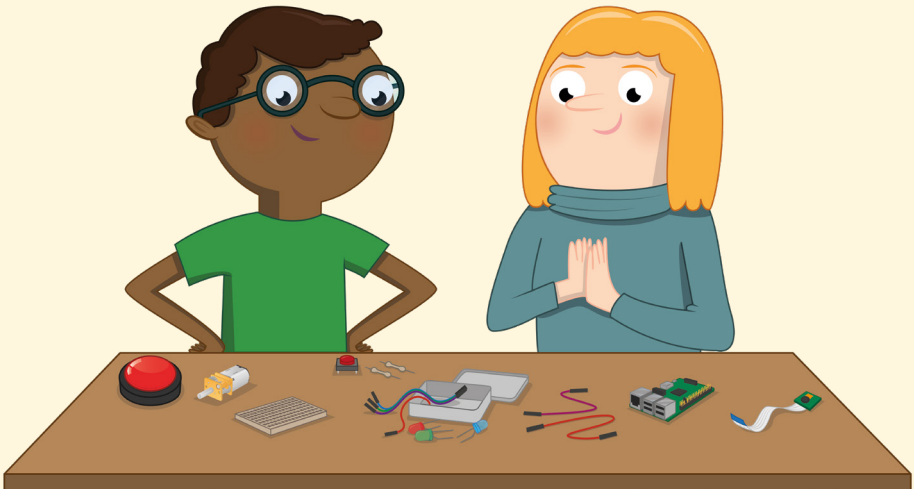
UTFORDRING: UTVIDE SPILLET

Kan du legge til flere rom for å få spillet til å vare lengre? Kan du legge til et element for å beskytte deg mot monster? Hvordan vil du legge til et våpen for å drepe monster? Kan du legge til rom over og under eksisterende rom, som kan nås via trapper?

Kapittel 6

Fysisk databehandling med Scratch og Python

Koding er mer enn å utføre ting på skjermen – du kan også kontrollere elektroniske komponenter som er koblet til GPIO-pinnene på Raspberry Pi



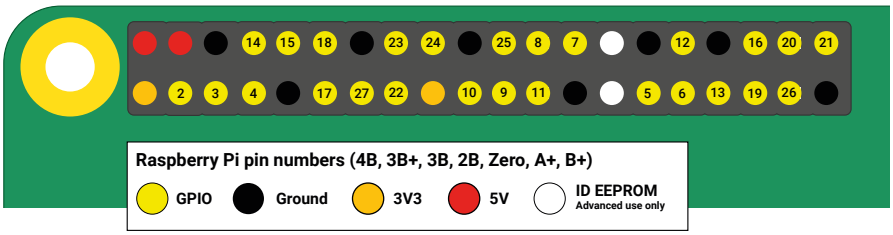
Når folk tenker på «programmering» eller «koding», tenker de vanligvis – og naturlig nok – på programvare. Koding kan imidlertid handle om mer enn bare programvare: Det kan påvirke den virkelige verden gjennom maskinvare. Denne prosessen kalles *fysisk databehandling*.

Som navnet antyder, handler fysisk databehandling om å kontrollere ting i den virkelige verden med programmene: maskinvaren og ikke programvaren. Når du stiller inn programmet på vaskemaskinen, endrer temperaturen på den programmerbare termostaten eller trykker på en knapp ved trafikklys for å krysse veien, bruker du fysisk databehandling.

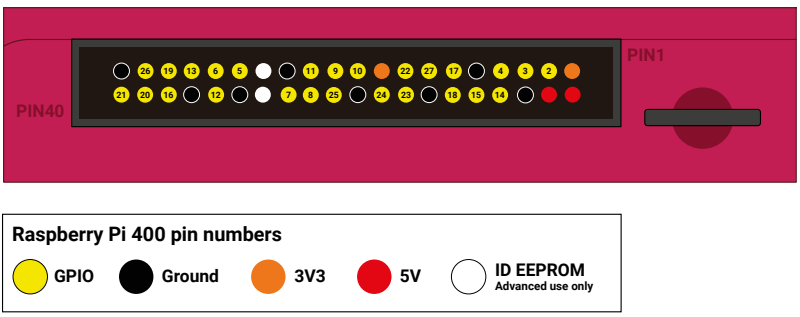
Raspberry Pi er en flott enhet for å lære mer om fysisk databehandling takket være én nøkkelfunksjon: *GPIO-pinnerekken* (*General-Purpose Input/Output*).

Vi presenterer GPIO-pinnerekken

Du finner GPIO-pinnerekken øverst på Raspberry Pi-kretskortet, eller på baksiden av Raspberry Pi 400. Det ser ut som to lange rader med metallpinner. Med disse kan du koble maskinvare som LED-lamper og bryter til Raspberry Pi for å kontrollere programmene du lager. Pinnene kan brukes som både innganger og utganger.



Raspberry Pis GPIO-pinnerekke består av 40 hannpinner. Noen pinner kan brukes til fysisk databehandling, noen pinner gir strøm, mens andre pinner er reservert for kommunikasjon med tilleggsmaskinvare som Sense HAT (se **kapittel 7**).



Raspberry Pi 400 har samme GPIO-pinnerekke, med alle de samme pinnene, men det står opp-ned i forhold til andre Raspberry Pi-modeller. Dette diagrammet viser GPIO-pinnerekken på Raspberry Pi 400 sett bakfra. Du må alltid sjekke ledningene når du kobler noe til Raspberry Pi 400s GPIO-pinnerekke – det er lett å glemme, selv om «Pin 40» og «Pin 1» er merket!

GPIO-UTVIDELSER

Det er fullt mulig å bruke GPIO-pinnerekken i Raspberry Pi 400 som det er, men det er kanskje lettere å bruke en utvidelse. Ved bruk av utvidelser plasseres pinnene på siden av Raspberry Pi 400, noe som betyr at du kan sjekke og justere kablingen uten å måtte gå til baksiden hele tiden.

Kompatible utvidelser omfatter Black HAT Hack3r-serien fra **pimoroni.com** og Pi T-Cobbler Plus fra **adafruit.com**.

Hvis du kjøper en utvidelse, må du alltid sjekke hvordan den er kablet – noen har et annet oppsett av GPIO-pinnene, for eksempel Pi T-Cobbler Plus. Følg alltid produsentens instruksjoner hvis du er i tvil.

Det finnes flere kategorier pinnetyper, hver med en bestemt funksjon:

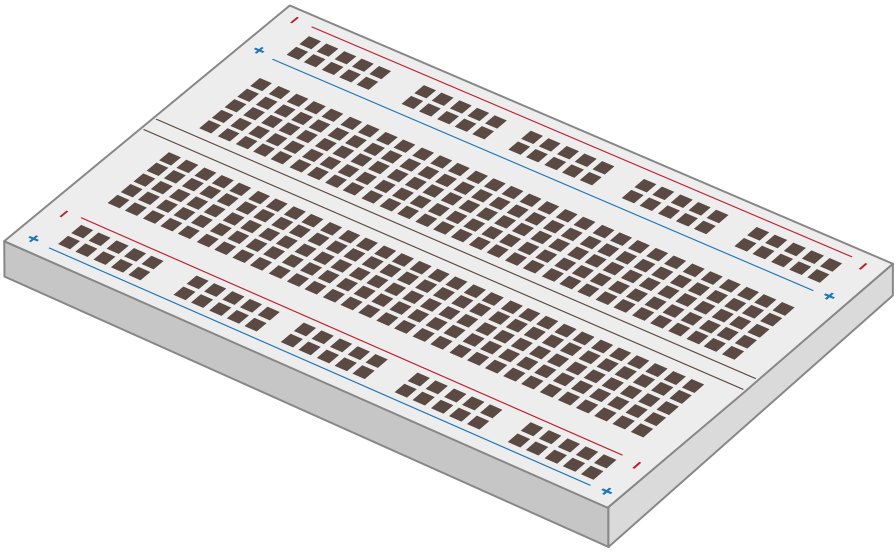
3V3	**3,3 volt strøm**	En kilde med 3,3 V-strøm som er kontinuerlig på, den samme spenningen Raspberry Pi bruker internt
5 V	**5 volt strøm**	En kilde med 5 V-strøm som er kontinuerlig på, den samme spenningen som Raspberry Pi mottar via en USB-kontakt
Jord (GND)	**0 volts jord**	En jordingstilkobling, som brukes til å fullføre en krets som er tilkoblet strømkilden
GPIO XX	**Pinnenummer for General-purpose input/output: «XX»**	GPIO-pinnene som er tilgjengelige for programmene, identifiseres ved hjelp av tall fra 2 til 27
ID EEPROM	**Reserverte spesialpinner**	Pinner som er reservert for bruk med HAT (Hardware Attached on Top) og annet tilbehør

ADVARSEL!

GPIO-pinnerekken i Raspberry Pi er en morsom og trygg måte å eksperimentere med fysisk databehandling på, men du må være forsiktig. Pass på at du ikke bøyer pinnene når du kobler til og fra maskinvaren. Du må aldri sammenkoble to pinner direkte, utilsiktet eller med vilje, med mindre du uttrykkelig får beskjed om det i instruksjonene til et prosjekt. Dette kalles kortslutning, og avhengig av pinnene kan det skade Raspberry Pi permanent.

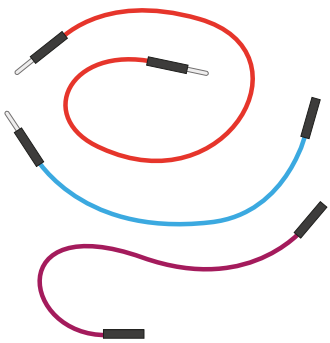
Elektroniske komponenter

GPIO-pinnerekken er bare én del av det du trenger for å begynne å jobbe med fysisk databehandling. Den andre halvdelen består av elektriske komponenter, enhetene du skal kontrollere fra GPIO-pinnerekken. Det finnes tusenvis av ulike komponenter, men de fleste GPIO-prosjekter er laget ved hjelp av følgende vanlige deler.

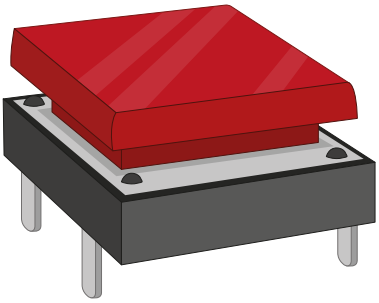


Med et *koblingsbrett*, også kalt *loddefritt koblingsbrett* (eksperimentkort), blir fysisk databehandling betydelig enklere. I stedet for å ha en haug med separate komponenter som må kobles til via ledninger, kan du bruke koblingsbrettet til å sette inn og koble til komponenter gjennom metallspor som er skjult under overflaten. Mange koblingsbrett har også seksjoner for strømfordeling, noe som gjør det enklere å bygge kretser. Du trenger ikke et koblingsbrett for å komme i gang med fysisk databehandling, men det er til god hjelp.

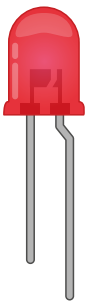
Jumperkabler, også kjent som *jumpers*, brukes for å koble komponenter til Raspberry Pi og til hverandre (hvis du ikke bruker koblingsbrett). De finnes i tre versjoner: hann/hunn (M2F), som du trenger for å koble et koblingsbrett til GPIO-pinnene; hunn/hunn (F2F), som kan brukes til å koble sammen enkeltkomponenter hvis du ikke bruker et koblingsbrett og hann/hann (M2M), som brukes til å lage tilkoblinger fra én del av et koblingsbrett til en annen. Avhengig av prosjektet kan det hende du trenger alle de tre kabeltypene. Hvis du bruker koblingsbrett, går det vanligvis bra med bare M2F- og M2M-jumperkabler.



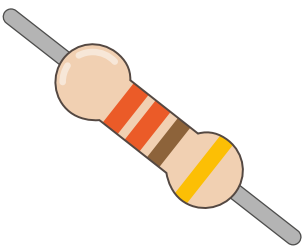
En *trykkbryter*, også kjent som en *taktbryter*, er en brytertype som brukes til å kontrollere en spillkonsoll. Trykkbrytere finnes med to eller fire ben – begge typer fungerer med Raspberry Pi – og er en inndataenhet. Du kan be programmet om å registrerer når det trykkes på bryteren, og deretter utføre en oppgave. En annen vanlig brytertype er en *låsebryter*. Mens en trykkbryter bare er aktiv mens du holder den inne, aktiveres en låsebryter (som finnes i lysbrytere) når du vipper den én gang og forblir aktiv til du vipper den igjen.



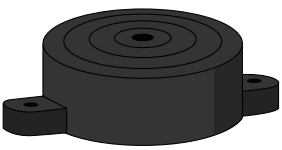
En *lysdiode* (*LED*) er en *utdataenhet*. Du styrer den direkte fra programmet. En LED-lampe lyser når den er på. Du finner dem overalt hjemme, alt fra de små som gir deg beskjed når du har latt vaskemaskinen være påslått, til de store du kanskje har for å lyse opp rommene. Det finnes LED-lamper i mange forskjellige former, farger og størrelser, men ikke alle er egnet for bruk med Raspberry Pi. Unngå versjoner som er beregnet på 5 V eller 12 V strømforsyning.



Motstander er komponenter som styrer den *elektriske strømmen* og finnes i ulike verdier målt i en enhet som kalles *ohm* (*Ω*). Jo høyere ohm-verdi, desto sterkere motstand. For fysiske dataprosjekter med Raspberry Pi er den vanligste funksjonen deres å hindre at LED-lampene trekker for mye strøm og skader seg selv eller Raspberry Pi. Til dette trenger du resistorer på rundt 330 Ω, men mange elektroforhandlere selger praktiske pakker med flere ulike vanlige verdier for mer fleksibilitet.



En *piezoelektrisk summer*, som vanligvis bare kalles en summer, er en utdataenhet. Mens en LED-lampe produserer lys, produserer en summer lyd – en summende lyd selvsagt. Inne i summerens plasthus finnes det et par metallplater. Når de er aktive, vibrerer disse platene mot hverandre for å produsere en summende lyd. Det finnes to typer summere: *aktive summere* og *passive summere*. Velg en aktiv summer – disse er de enkleste å bruke.



Andre vanlige elektriske komponenter er motorer som trenger et spesielt kontrollkort før de kan kobles til Raspberry Pi, infrarøde sensorer som oppdager bevegelse, temperatur- og fuktighetssensorer som kan brukes til å forutsi været, og lysavhengige resistorer (LDR) – inndataenheter som fungerer som en omvendt LED-lampe ved å oppdage lys.

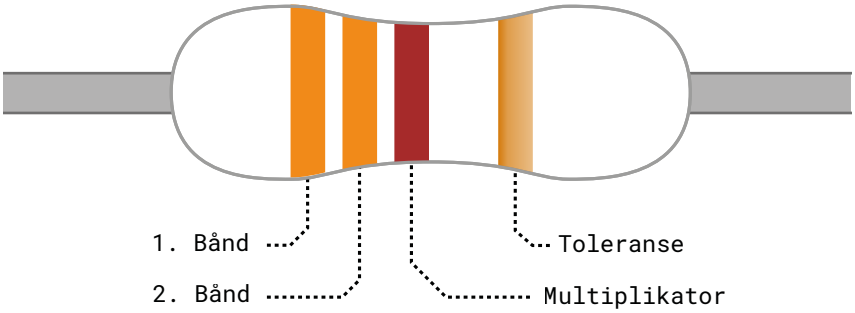
Forhandlere over hele verden leverer komponenter for fysisk databehandling med Raspberry Pi, enten som enkeltdeler eller i sett – med alt du trenger for å komme i gang. Gå inn på **rpf.io/products** for å finne forhandlere. Der klikker du på Raspberry Pi 4, og du vil se en liste over nettbutikkene til Raspberry Pis partnere (godkjente forhandlere) for landet eller området ditt.

For å fullføre prosjektene i dette kapittelet trenger du minst dette:

- 3 × LED-lamper: rød, grønn og gul eller oransje
- 2 × trykkbrytere
- 1 × aktiv summer
- jumperkabler av typen hann/hunn (M2F) og hunn/hunn (F2F)
- eventuelt et koblingsbrett og jumperkabler av typen hann/hann (M2M)

Tolke resistorfargekoder

Resistorer finnes i mange ulike verdier, fra nullmotstandsversjoner som stort sett bare er ledningsstykker, til versjoner med høy motstand på størrelse med benet ditt. Svært få av disse resistorene har imidlertid verdiene påtrykt i tall. De bruker i stedet en spesiell kode trykt som fargede striper eller bånd rundt selve resistorene.



	1./2. Bånd	Multiplikator	Toleranse
Svart	0	$\times 10^0$	-
Brun	1	$\times 10^1$	±1%
Rød	2	$\times 10^2$	±2%
Oransje	3	$\times 10^3$	-
Gul	4	$\times 10^4$	-
Grønn	5	$\times 10^5$	±0.5%
Blå	6	$\times 10^6$	±0.25%
Fiolett	7	$\times 10^7$	±0.1%
Grå	8	$\times 10^8$	±0.05%
Hvit	9	$\times 10^9$	-
Gull	-	$\times 10^{-1}$	±5%
Sølv	-	$\times 10^{-2}$	±10%
Ingen	-	-	±20%

Når du skal tolke verdien til en resistor, plasserer du den slik at gruppen av bånd befinner seg til venstre og det enslige båndet til høyre. Begynn med det første båndet, slå opp fargen i tabellkolonnen «1./2. Bånd» for å finne de første og andre sifrene. Dette eksemplet har to oransje bånd, som begge har en verdi på «3», dvs. i alt «33». Hvis resistoren har fire grupperte bånd i stedet for tre, noterer du verdien til det tredje båndet også (se **rpf.io/5-6band** for fem-/seksbånds resistorer).

Gå videre til det siste gruppebåndet – det tredje eller fjerde – og slå opp fargen i kolonnen Multiplikator. Dette gir deg tallet du skal multiplisere det gjeldende tallet med for å finne

resistorens faktiske verdi. Dette eksemplet har et brunt bånd, som betyr «×10^1»'. Dette kan virke forvirrende, men det er ganske enkelt en *vitenskapelig notasjon*: «×10^1» betyr bare at du skal «legge til en null på slutten av tallet». Hvis det var blått, ville «×10^6» bety å «legge til seks nuller på slutten av tallet».

Tallet 33, fra de oransje båndene, pluss den ekstra nullen fra det brune båndet gir 330 – som er verdien til resistoren, målt i ohm. Det endelige båndet, til høyre, er resistorens *toleranse*. Dette betyr ganske enkelt hvor nær den nominelle verdien den sannsynligvis vil være. Billigere resistorer kan ha et sølvbånd, noe som indikerer at verdien kan være 10 % høyere eller lavere enn klassifiseringen. De kan også mangle det siste båndet fullstendig, noe som indikerer at verdien kan være 20 % høyere eller lavere. De dyreste resistorene har et grått bånd, noe som betyr at verdien ligger innenfor 0,05 % av klassifiseringen. For hobbyprosjekter er ikke presisjon like viktig. All toleranse vil vanligvis fungere bra.

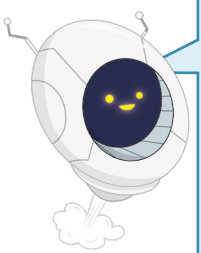
Hvis resistorverdien overstiger 1000 ohm (1000 Ω), blir den vanligvis målt i kiloohm (kΩ). Hvis den overstiger en million ohm, heter det megaohm (MΩ). En resistor med en motstand på 2200 Ω angis som 2,2 kΩ; en resistor med en motstand på 2 200 000 Ω angis som 2,2 MΩ.

VET DU SVARET?

Hvilke fargebånd har en 100 Ω resistor? Hvilke fargebånd har en 2,2 MΩ resistor? Hvis du ønsker å finne de billigste resistorene – hvilket fargetoleransebånd skal du se etter?

Ditt første fysiske dataprogram: Hello, LED!

Akkurat som å skrive ut «Hello, World» på skjermen er et fantastisk første skritt i å lære et programmeringsspråk, er det å få en LED-lampe til å lyse den tradisjonelle innføringen i fysisk databehandling. Til dette prosjektet trenger du en LED-lampe og en resistor på 330 ohm (330 Ω), eller så nær 330 Ω som mulig, pluss jumperkabler av typen hunn/hunn (F2F).

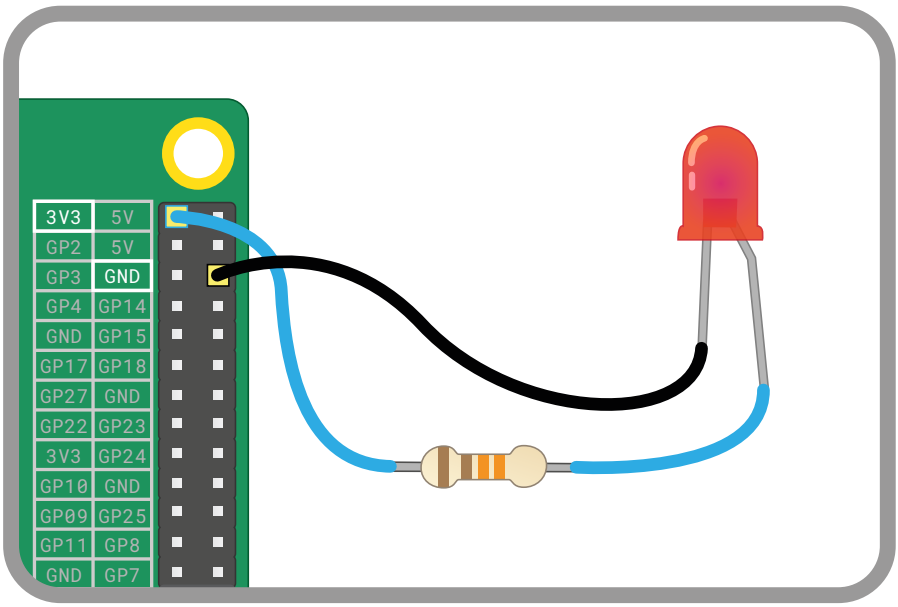


MOTSTANDEN ER VIKTIG

Resistoren er en viktig komponent i denne kretsen. Den beskytter Raspberry Pi og LED-lampen ved å begrense mengden elektrisk strøm LED-lampen kan trekke. Uten den kan LED-lampen trekke for mye strøm og brenne ut seg selv – eller Raspberry Pi. Når den brukes slik, kalles resistoren en *strømbegrensende resistor*. Den nøyaktige resistorstyrken du trenger, avhenger av LED-lampen du bruker, men 330 Ω fungerer for de vanligste LED-lampene. Jo høyere verdi, desto svakere blir LED-lampen. Jo lavere verdi, desto sterkere blir LED-lampen.

Koble aldri en LED-lampe til Raspberry Pi uten å bruke en strømbegrensende resistor, med mindre du vet at LED-lampen har en innebygd resistor med egnet verdi.

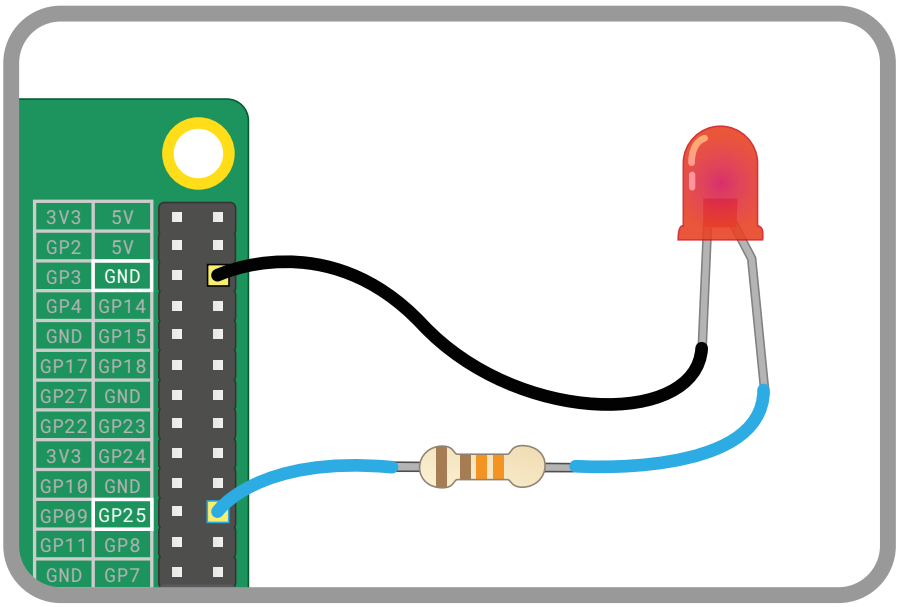
Start med å sjekke at LED-lampen fungerer. Snu Raspberry Pi slik at GPIO-pinnerekken vises med to vertikale striper til høyre. Koble den ene enden av 330 Ω-resistoren til den første 3,3 V-pinnen (merket 3V3 i **Figur 6-1**) ved hjelp av en jumperkabel av typen hunn/hunn. Deretter kobler du den andre enden til det lange benet – positiv pol eller anode – på LED-lampen med en annen jumperkabel av samme type. Bruk en siste jumperkabel av typen hunn/hunn til å koble det korte benet – negativ pol eller katode – på LED-lampen til den første jordingspinnen (merket GND i **Figur 6-1**).



▲ **Figur 6-1:** Koble LED-lampen til disse pinnene – ikke glem resistoren!

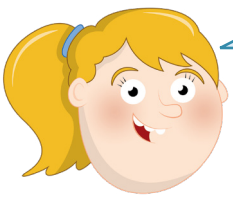
Så lenge Raspberry Pi er på, skal LED-lampen lyse. Hvis ikke, må du dobbeltsjekke kretsen. Pass på at du ikke har brukt for høy resistorverdi, at alle ledningene er riktig tilkoblet og særlig at du har valgt de riktige GPIO-pinnene i forhold til diagrammet. Sjekk også bena på LED-lampen. De fungerer nemlig bare én vei: med det lange benet koblet til den positive siden av kretsen og det korte benet til den negative.

Når LED-lampen fungerer, er det på tide å programmere den. Koble jumperkabelen fra 3,3 V-pinnen (merket 3V3 i **Figur 6-2**), og koble den til GPIO 25-pinnen (merket GP25 i **Figur 6-2**). LED-lampen slukkes, men det er helt normalt.



Figur 6-2: Koble kabelen fra 3V3 og koble den til GPIO 25-pinnen

Du er nå klar til å lage et Scratch- eller Python-program for å slå LED-lampen av og på.

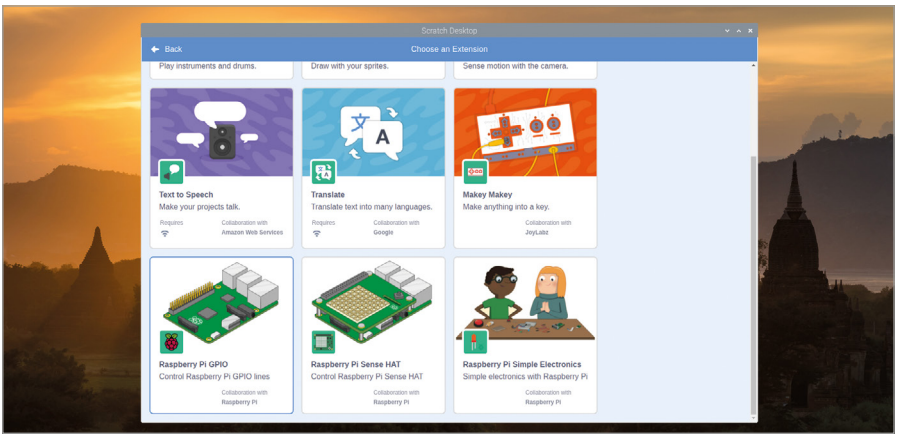


KODINGSKUNNSKAPER

Prosjektene i dette kapittelet avhenger av at du er kjent med å bruke Scratch 3 og det integrerte utviklingsmiljøet (IDE) Thonny Python. Hvis du ikke allerede har gjort det, går du til **Kapittel 4: Programmering med Scratch 3** og **Kapittel 5: Programmering med Python** og går gjennom disse prosjektene først.

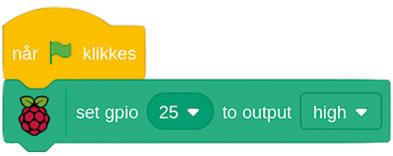
LED-kontroll i Scratch

Last ned Scratch 3, og klikk på ikonet for Legg til utvidelse. Rull nedover til du finner utvidelsen «Raspberry Pi GPIO» (**Figur 6-3**, på neste side), og klikk på den. Dette laster klossene du trenger for å kontrollere Raspberry Pis GPIO-pinnerekken fra Scratch 3. De nye klossene viser i klosspaletten. Når du trenger dem, finner du dem i kategorien Raspberry Pi GPIO.

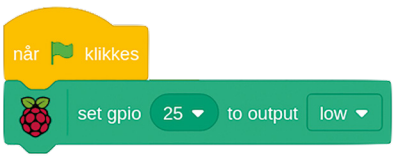


▲ **Figur 6-3:** Legge til Raspberry Pi GPIO-utvidelsen i Scratch 3

Start med å dra hendelsesklossen når klikkes til kodeområdet. Deretter plasserer du klossen set gpio to output high under den. Du må velge nummeret på pinnen du bruker. Klikk på den lille pilen for å åpne rullegardinmenyen, og klikk på «25» for å fortelle Scratch at du kontrollerer GPIO 25-pinnen.

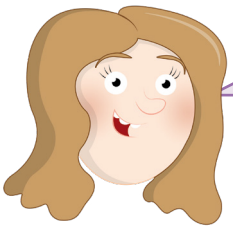


Klikk på det grønne flagget for å kjøre programmet. LED-lampen tennes. Du har programmert ditt første fysiske databehandlingsprosjekt! Klikk på den røde åttekanten for å stoppe programmet. Legger du merke til at LED-lampen fortsatt lyser? Det skyldes at programmet bare ba Raspberry Pi om å slå på LED-lampen – dette er betydningen av «output high»-delen i klossen set gpio 25 to output high. Klikk på nedpilen på slutten av klossen og velg verdien «low» på listen for å slå av lampen igjen.



Klikk på det grønne flagget igjen. Denne gangen vil programmet slå av LED-lampen. For å gjøre det mer interessant kan du legge til kontrolklossen gjenta for alltid og et par klosser av typen vent 1 sekunder for å lage et program som får LED-lampen til å blinke av og på hvert sekund.

Klikk på det grønne flagget og følg med på LED-lampen. Den slås på et sekund, slås av et sekund, slås på et sekund og gjentar dette mønsteret til du klikker på den røde åttekanten for å stoppe det. Se hva som skjer når du klikker på åttekanten mens LED-lampen er i på- eller av-tilstand.



UTFORDRING: KAN DU ENDRE DET?

Hvordan ville du endre programmet slik at LED-lampen lyser lengre? Hva med å være slukket lengre? Hva er den minste forsinkelsen du kan bruke, og fremdeles se at LED-lampen slås av og på?

LED-kontroll i Python

Last ned Thonny fra programmeringsdelen av Raspberry-menyen, og klikk deretter på Ny-knappen for å starte et nytt prosjekt og Lagre for å lagre det som **Hello LED**. For å bruke GPIO-pinnene fra Python trenger du et bibliotek som heter GPIO Zero. Til dette prosjektet trenger du bare den delen av biblioteket som kreves for å jobbe med LED-lamper. Importer bare denne delen av biblioteket ved å skrive følgende i Python-skallområdet:

```
from gpiozero import LED
```

Deretter må du fortelle GPIO Zero hvilken GPIO-pinne LED-lampen er koblet til. Skriv følgende:

```
led = LED(25)
```

Sammen gir disse to linjene Python muligheten til å kontrollere LED-lamper som er koblet til Raspberry Pis GPIO-pinner, og fortelle programmet hvilken pinne – eller pinner hvis du har mer enn én LED-lampe i kretsen – som skal kontrolleres. Skriv følgende for å kontrollere LED-lampen:

```
led.on()
```

Hvis du vil slå av LED-lampen igjen, skriver du:

```
led.off()
```

Gratulerer! Nå kan du styre Raspberry Pis GPIO-pinner i Python! Prøv å skrive disse to instruksjonene på nytt. Hvis LED-lampen allerede er av, vil ikke **`led.off()`** få noe til å skje. Det samme gjelder hvis LED-lampen allerede er på og du skriver **`led.on()`**.

Når du skal lage et ekte program, skriver du følgende i skriptområdet:

```
from gpiozero import LED
from time import sleep

led = LED(25)

while True:
    led.on()
    sleep(1)
    led.off()
    sleep(1)
```

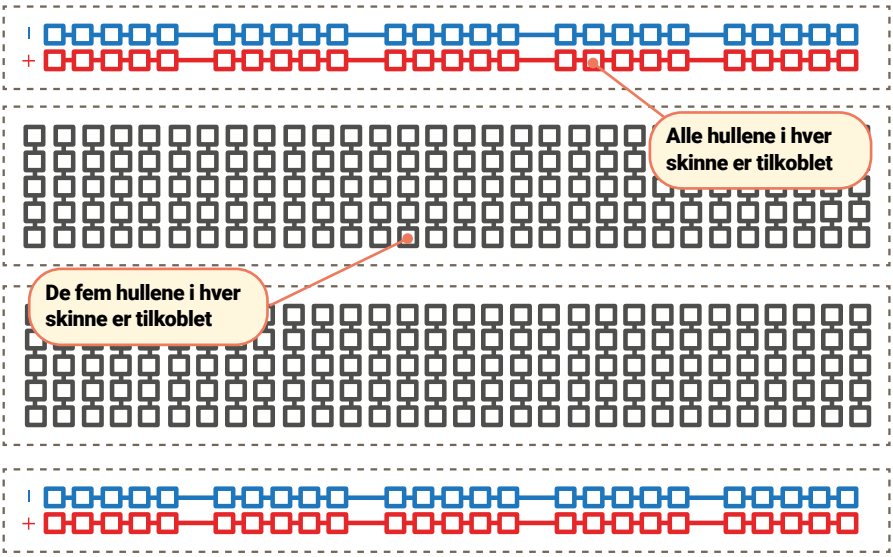
Dette programmet importerer LED-funksjonen fra biblioteket **`gpiozero`** (GPIO Zero) og funksjonen **`sleep`** fra biblioteket **`time`** og lager deretter en uendelig løkke for å slå på LED-lampen et sekund, slå den av et sekund og gjenta prosessen. Klikk på Run-knappen for å se hvordan det fungerer i virkeligheten: LED-lampen begynner å blinke. Som med Scratch-programmet observerer du hva som skjer når du klikker på Stopp-knappen mens LED-lampen er på og når den er av.

UTFORDRING: TENT LENGRE

Hvordan ville du endre programmet slik at LED-lampen lyser lengre? Hva med å være slukket lengre? Hva er den minste forsinkelsen du kan bruke, og fremdeles se at LED-lampen slås av og på?

Bruke et koblingsbrett

De neste prosjektene i dette kapittelet blir mye enklere å utføre hvis du bruker et koblingsbrett til å feste komponentene og lage de elektriske tilkoblingene.



Et koblingsbrett er dekket av hull som er plassert i en avstand på 2,54 mm, slik at de samsvarer med komponentene. Under disse hullene ser du metallstrimler, som fungerer som jumperkablene du har brukt hittil. Disse er plassert i rader over hele brettet, og de fleste brett har et tomrom på midten som deler dem inn i to halvdeler. Mange koblingsbrett har også bokstaver langs kanten øverst og tall nedover på begge sider. Du kan bruke disse for å finne et bestemt hull: A1 er øverst til venstre, B1 er hullet til rett til høyre for dette, mens B2 er neste hull nedenfor dette. A1 er koblet til B1 via de skjulte metallstrimlene, men ingen hull er på noe tidspunkt koblet til 2 hull, med mindre du selv legger til en jumperkabel.

Større koblingsbrett har også hullstrimler nedover sidene, vanligvis merket med røde og svarte eller røde og blå striper. Dette er *strømskinnene*. De er ment å gjøre installasjonen enklere. Du kan koble en enkel kabel fra Raspberry Pis jordingspinne til en av strømskinnene – vanligvis merket med en blå eller sort stripe og et minustegn – for å få en *felles jord* for flere av komponentene på brettet. Du kan gjøre det samme hvis kretsen må ha 3,3 V- eller 5 V-strøm.

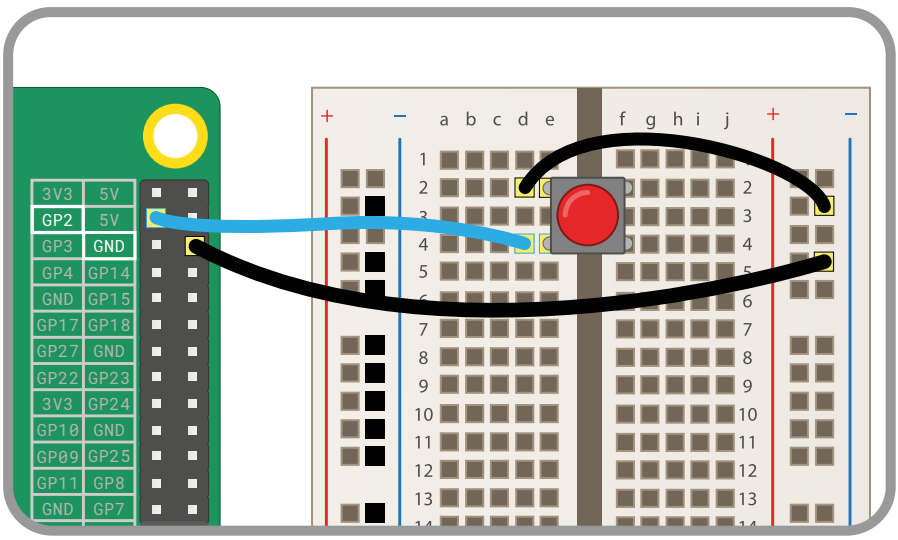
Det er enkelt å legge til elektroniske komponenter på et koblingsbrett. Du bare innretter trådene (metalldelene som stikker ut) etter hullene og skyver dem forsiktig inn til komponenten er på plass. Hvis du skal foreta flere tilkoblinger enn koblingsbrettet har plass til, kan du bruke jumperkabler av typen hann/hann (M2M). For tilkoblinger fra koblingsbrettet til Raspberry Pi bruker du kabler av typen hann/hunn (M2F).

Du må aldre feste mer enn én komponentledning eller jumperkabel i samme hull på koblingsbrettet. Husk: Hullene er koblet i kolonner, bortsett fra tomrommet i midten, så en komponentledning i A1 er elektrisk tilkoblet alt du legger til i B1, C1, D1 og E1.

Neste trinn: Lese en knapp

Utdataenheter som for eksempel LED-lamper er én ting, men «input/output»-delen i GPIO betyr at du også kan bruke pinner som innganger. Til dette prosjektet trenger du et koblingsbrett, jumperkabler av typen hann/hann (M2M) og hann/hunn (M2F) samt en trykkbryter. Hvis du ikke har et koblingsbrett, kan du bruke jumperkabler av typen hunn/hunn (F2F), men det vil være mye vanskeligere å trykke på knappen uten å bryte kretsen.

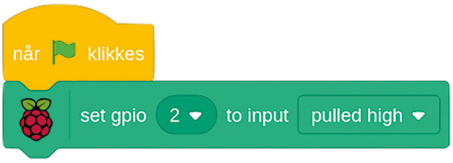
Start med å legge til trykkbryteren på koblingsbrettet. Hvis trykkbryteren bare har to ben, må du passe på at de er i forskjellige nummererte rader på koblingsbrettet. Hvis den har fire ben, snur du den slik at sidene som bena stikker ut fra, befinner seg langs koblingsbrettets rader, og at de flate sidene uten ben befinner seg øverst og nederst. Koble jordingsskinnen på koblingsbrettet til en jordingspinne på Raspberry Pi (merket GND på **Figur 6-4**) ved hjelp av en jumperkabel av typen hann/hunn. Deretter kobler du det ene benet på trykkbryteren til jordingsskinnen med en kabel av typen hann/hann. Til slutt kobler du det andre benet – på samme side som benet du nettopp koblet til, hvis du bruker en firbensbryter – til GPIO 2-pinnen (merket GP2 på **Figur 6-4**) på Raspberry Pi ved å bruke en jumperkabel av typen hann/hunn.



▲ **Figur 6-4:** Koble en trykkbryter til GPIO-pinnene

Lese en knapp i Scratch

Start et nytt Scratch-program og dra klossen når klikkes til kodeområdet. Koble til klossen set gpio to input pulled high, og velg nummer 2 fra rullegardinmenyen, som passer til GPIO-pinnen du brukte for trykkbryteren.



Hvis du klikker på det grønne flagget nå, skjer det ingenting. Det er fordi du har fortalt Scratch at pinnen skal brukes som en inngang, men ikke hva du skal gjøre med denne inngangen. Dra klossen **gjenta for alltid** til slutten av sekvensen, og dra deretter klossen **hvis ellers** inn i denne klossen. Finn klossen **gpio is high?** og dra den inn i det diamantformede feltet i **hvis** -delen av klossen. Deretter velger du tallet 2 fra rullegardinmenyen for å fortelle Scratch hvilken GPIO-pinne du vil sjekke. Dra klossen **si Hei! i 2 sekunder** inn i **ellers**-delen av klossen, og rediger den for å si «Det trykkes på knappen!». La «hvis»-delen (if then) av klossen være tom foreløpig.

Det er mye som skjer der, men start med å teste det. Klikk på det grønne flagget, og trykk deretter på knappen på koblingsbrettet. Figuren burde vise at det er trykket på knappen: Du har lest inndata fra GPIO-pinnen!

Du har kanskje lagt merke til at delen **hvis gpio 2 is high?** i klossen er tom. Koden som kjøres når det trykkes på knappen, er derimot i **ellers**-delen av klossen. Det kan være forvirrende. Skulle ikke et trykk på knappen få den til å gå høyt? Det er faktisk motsatt: GPIO-pinnene i Raspberry Pi er vanligvis satt til høy spenning (high) eller på når de er angitt som en inngang, og trykk på knappen tar dem ned til lav spenning (low).

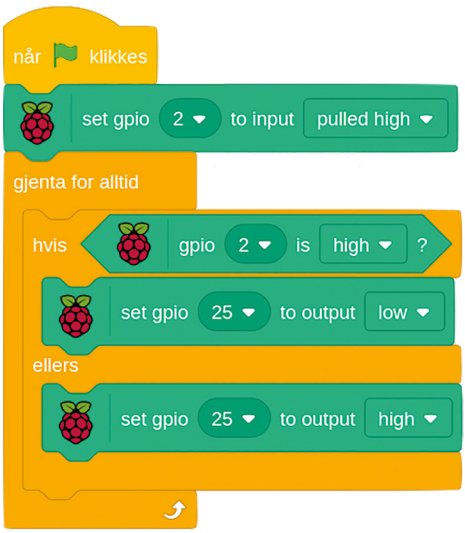
Se på kretsen igjen. Observer at knappen er koblet til GPIO 2-pinnen, som utgjør den positive delen av kretsen, og jordingspinnen. Når det trykkes på knappen, trekkes spenningen på GPIO-

pinnen lavt gjennom jordingspinnen, og Scratch-programmet slutter å kjøre koden i klossen hvis gpio 2 is high?. I stedet kjøres koden i ellers -delen i klossen.

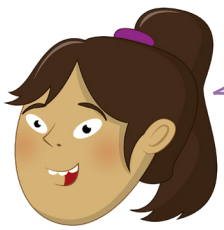
Hvis det høres forvirrende ut, bør du huske dette: En knapp på en Raspberry Pi GPIO-pinne trykkes når pinnen blir lav, ikke når den blir høy!

Hvis du vil utvide programmet ytterligere, legger du til LED-lampen og resistoren i kretsen igjen. Husk å koble resistoren til GPIO 25-pinnen og til det lange benet på LED-lampen. Koble det korte benet på LED-lampen til jordingsskinnen på koblingsbrettet.

Trekk klossen si Knapp trykket! i 2 sekunder fra kodeområdet til klosspaletten for å slette den og erstatte den med blokken set gpio 25 to output high. Husk at du må endre GPIO-nummeret ved hjelp av rullegardinpilen. Legg til klossen set gpio 25 to output low (husk å endre verdiene) i delen hvis gpio 2 is high? i klossen, som for øyeblikket er tom.



Klikk på det grønne flagget og trykk på knappen. LED-lampen lyser så lenge du holder knappen inne. Når du slipper den, slukkes lampen igjen. Gratulerer! Nå kan du kontrollere én GPIO-pinne basert på inndata fra en annen!



UTFORDRING: FÅ DEN TIL Å FORBLI TENT

Hvordan ville du endre programmet slik at LED-lampen lyser et par sekunder, selv etter at du har sluppet opp knappen? Hva ville du måtte endre for å ha LED-lampen på når du ikke trykker på knappen og av når du gjør det?

Lese en knapp i Python

Klikk på New-knappen i Thonny for å starte et nytt prosjekt, og klikk på Save for å lagre det som **Knappeinndata**. Når du bruker en GPIO-pinne som inngang for en knapp, fungerer det nesten på samme måte som å bruke en pinne som utgang for en LED-lampe. Du må derimot importere en annen del av GPIO Zero-biblioteket. Skriv følgende i skriptområdet:

```
from gpiozero import Button
button = Button(2)
```

Hvis du vil at koden skal kjøres når det trykkes på knappen, kan du bruke GPIO Zero-funksjonen **`wait_for_press`**. Skriv følgende:

```
button.wait_for_press()
print("Du trykket på meg!")
```

Klikk på Run-knappen, og trykk deretter på trykkbryteren. Meldingen din skrives ut til Python-skallet nederst i Thonny-vinduet. Du har lest inndata fra GPIO-pinnen! Hvis du vil teste programmet igjen, klikker du på Run-knappen igjen. Siden det ikke finnes noen løkker i programmet, avsluttes det så snart det er ferdig med å skrive meldingen til skallet.

Hvis du vil utvide programmet ytterligere, legger du til LED-lampen og resistoren i kretsen igjen, hvis du ikke allerede har gjort det. Husk å koble resistoren til GPIO 25-pinnen og til det lange benet på LED-lampen. Koble det korte benet på LED-lampen til jordingsskinnen på koblingsbrettet.

Når du vil kontrollere en LED-lampe og lese en knapp, må du importere både **`Button`**- og **`LED`**-funksjonene fra GPIO Zero-biblioteket. Du trenger også **`sleep`**-funksjonen fra **`time`**-biblioteket. Gå tilbake til toppen av programmet og sett inn følgende som de to nye første linjene:

```
from gpiozero import LED
from time import sleep
```

Under linjen **`button = Button(2)`** skriver du:

```
led = LED(25)
```

Slett linjen **`print("Du trykket på meg!")`** og erstatt den med:

```
led.on()
sleep(3)
led.off()
```

Det ferdige programmet skal nå se slik ut:

```
from gpiozero import LED
from time import sleep
from gpiozero import Button

button = Button(2)
led = LED(25)
button.wait_for_press()
led.on()
sleep(3)
led.off()
```

Klikk på Run-knappen, og trykk deretter på trykkbryteren. LED-lampen lyser i tre sekunder, slukkes igjen, og programmet avsluttes. Gratulerer! Nå kan du styre en LED ved å bruke knappeinndata i Python!

UTFORDRING: LEGGE TIL EN LØKKE

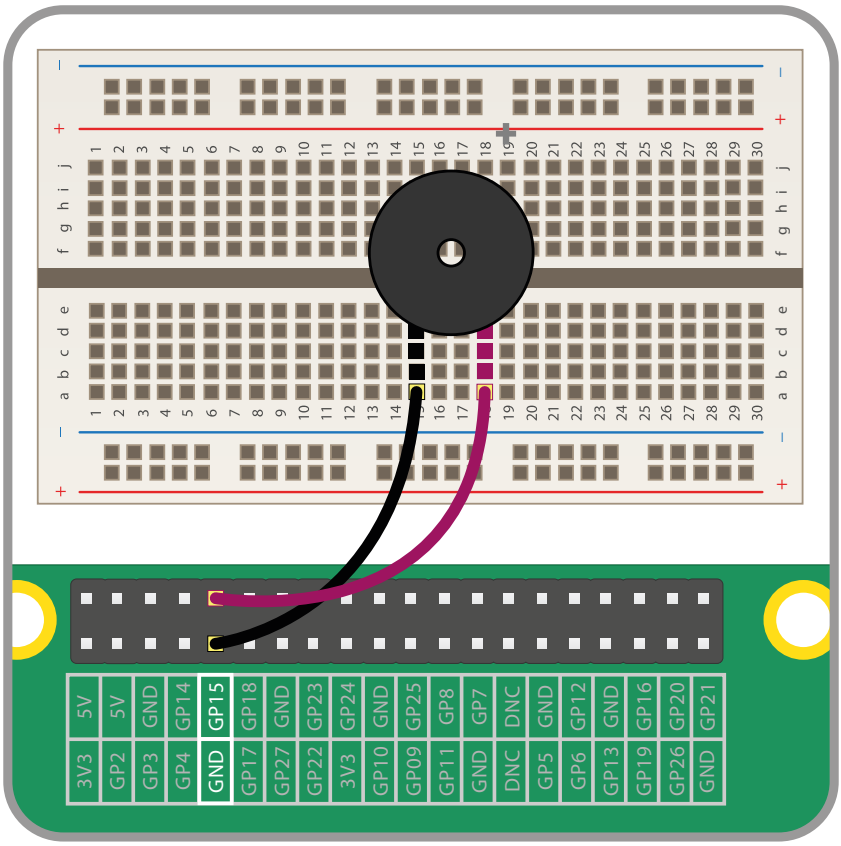
Hvordan legger du til en løkke for å få programmet til å gjenta seg i stedet for å avslutte etter ett tastetrykk? Hva ville du måtte endre for å ha LED-lampen på når du ikke trykker på knappen og av når du gjør det?

Lag litt støy: Kontrollere en summer

LED-lamper er flotte utdataenheter, men de er ikke så nyttige hvis du ser den andre veien. Løsningen: Summere, som lager en lyd som kan høres over hele rommet. Til dette prosjektet trenger du et koblingsbrett, jumperkabel av typen hann/hunn (M2F) samt en trykkbryter. Hvis du ikke har et koblingsbrett, kan du koble til summeren ved hjelp av jumperkabel av typen hunn/hunn (F2F) i stedet.

En aktiv summer kan behandles akkurat som en LED-lampe når det gjelder kretser og programmering. Gjenta kretsen du laget for LED-lampen, men bytt ut LED-lampen med den aktive summeren og fjern resistoren. Summeren trenger nemlig mer strøm for å fungere. Koble det ene benet på summeren til GPIO 15-pinnen (merket GP15 i **Figur 6-5**) og den andre til jordingspinnen (merket GND i diagrammet) ved hjelp av koblingsbrettet og jumperkabler av typen hann/hunn.

Hvis summeren har tre ben, må du passe på at benet som er merket med et minustegn (–), er koblet til jordingspinnen og at benet som er merket med «S» eller «SIGNAL», er koblet til GPIO 15. Koble til det gjenværende benet – vanligvis det midtre benet – til 3,3 V-pinnen (merket 3V3.)



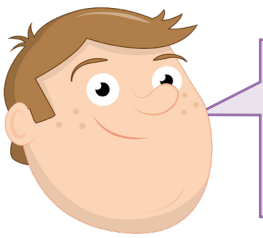
▲ **Figur 6-5:** Koble en summer til GPIO-pinnene

Kontrollere en summer i Scratch

Gjenopprett det samme programmet som får LED-lampen til å blinke – eller last det inn hvis du lagret det før du opprettet knappeprosjektet. Bruk rullegardinmenyen i klossene av typen **set gpio to output high** for å velge tallet 15, slik at Scratch kontrollerer riktig GPIO-pinne.

Klikk på det grønne flagget, så utløses summeren: et sekund på og et sekund av. Hvis du bare hører summeren klikke én gang i sekundet, bruker du en passiv summer og ikke en aktiv summer. En aktiv summer genererer selv det raskt skiftende signalet, kjent som en *svingning*, som får metallplatene til å vibrere, mens en passiv summer treger et svingningssignal. Hvis du bare slår den på ved å bruke Scratch, beveges platene bare én gang før de stopper. Det er dette som lager «klikkelyden» til neste gang programmet slår pinnen på eller av.

Klikk på den røde åttekanten for å stoppe summeren, men sørg for å gjøre det når den ikke avgir noen lyd. Ellers vil summeren fortsette til du kjører programmet igjen!



UTFORDRING: ENDRE SUMMINGEN

Hvordan kan du endre programmet slik at summeren høres i kortere tid? Kan du bygge en krets slik at summeren styres av en knapp?

Kontrollere en summer i Python

Når du skal kontrollere en aktiv summer gjennom GPIO Zero-biblioteket, fungerer det neste på samme måte som å kontrollere en LED-lampe, da den har på- og av-tilstander. Men du trenger også en annen funksjon: en **`buzzer`**. Start et nytt prosjekt i Thonny og lagre det som **Buzzer**. Deretter skriver du følgende:

```
from gpiozero import Buzzer
from time import sleep
```

Som med LED-lamper må GPIO Zero vite hvilken pinne summeren er koblet til, for å kontrollere den. Skriv følgende:

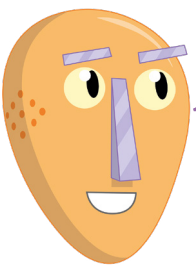
```
buzzer = Buzzer(15)
```

Herfra er programmet nesten identisk med det du skrev for å kontrollere LED-lampen. Den eneste forskjellen (bortsett fra et annet GPIO-pinnenummer) er at du bruker **`buzzer`** i stedet for **`led`**. Skriv følgende:

```
while True:
    buzzer.on()
    sleep(1)
    buzzer.off()
    sleep(1)
```

Klikk på det grønne flagget, så utløses summeren: ett sekund på og ett sekund av. Hvis du bruker en passiv summer i stedet for en aktiv summer, hører du bare et kort klikk hvert sekund i stedet for en kontinuerlig lyd. Det skyldes at en passiv summer mangler en *oscillator*, som lager det raskt skiftende signalet som får platene inne i summeren til å vibrere.

Klikk på Stop-knappen for å avslutte programmet, men sørg for at summeren ikke avgir en lyd denne gangen. Ellers vil den fortsette å summe til du kjører programmet igjen!



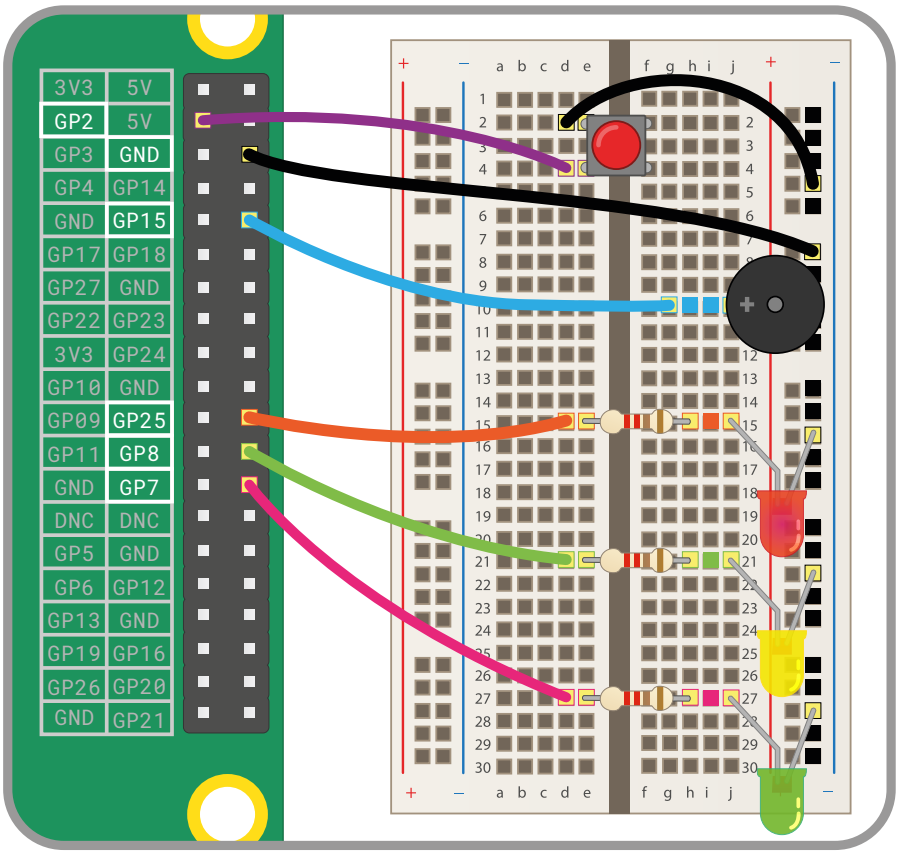
UTFORDRING: BEDRE SUMMING

Hvordan kan du endre programmet slik at summeren høres i kortere tid? Kan du bygge en krets slik at summeren styres av en knapp?

Scratch-prosjekt: Trafikklys

Nå som du vet hvordan du bruker knapper, summere og LED-lamper som inn- og utdataenheter, er du klar til å bygge et dataprogram for dagliglivet: trafikklys og en knapp du kan trykke på for å krysse veien. Til dette prosjektet trenger du et koblingsbrett, en rød, gul og grønn LED-lampe, tre 330 Ω-resistorer, en summer, en trykkbryter og et utvalg av jumperkabler av typen hann/hann (M2M) og hann/hunn (M2F).

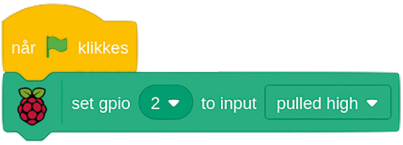
Start med å bygge kretsen (**Figur 6-6**), koble summeren til GPIO 15-pinnen (merket GP15 i **Figur 6-6**), den røde LED-lampen til GPIO 25-pinnen (merket GP25), den gule LED-lampen til GPIO 8 (GP8), den grønne LED-lampen til GPIO 7 (GP7) og bryteren til GPIO 2 (GP2). Husk å koble 330 Ω-resistorene mellom GPIO-pinnene og de lange bena på LED-lampene, og koble de andre bena på alle komponentene til jordingsskinnen på koblingsbrettet. Til slutt kobler du jordingsskinnen til en jordingspinne (merket GND) på Raspberry Pi for å fullføre kretsen.



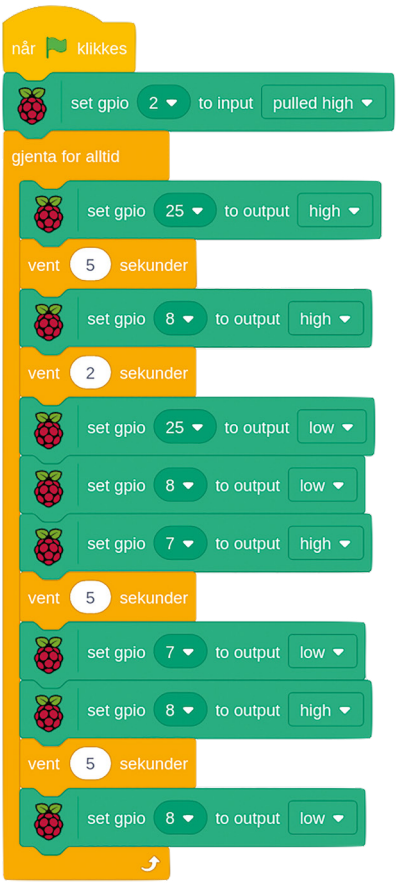
▲ **Figur 6-6:** Koblingsskjema for trafikklysprosjektet

Start et nytt Scratch 3-program og dra klossen når klikkes til kodeområdet. Deretter må du fortelle Scratch at GPIO 2-pinnen, som er koblet til trykkbryteren i kretsen, er en inngang

i stedet for en utgang. Dra klossen **set gpio to input pulled high** fra Raspberry Pi GPIO-kategorien i klosspaletten under klossen **når klikkes**. Klikk på nedpilen ved siden av «0» og velg tallet 2 fra rullegardinlisten.



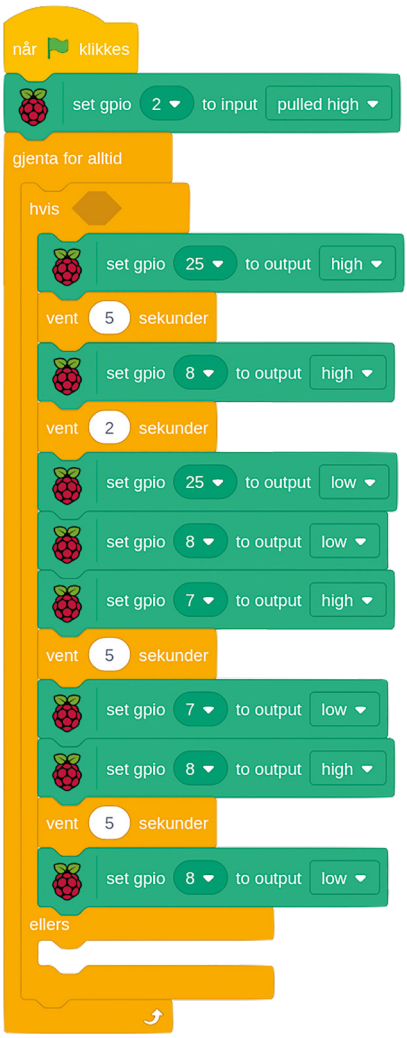
Deretter lager du trafikklyssekvensen. Dra klossen **gjenta for alltid** inn i programmet, og fyll den deretter med klosser for å slå trafikklyslampene av og på i et mønster. Husk hvilke GPIO-pinner som er tilknyttet hvilken komponent. Når du bruker pinne 25, bruker du den røde LED-lampen, pinne 8 den gule LED-lampen og pinne 7 den grønne LED-lampen.



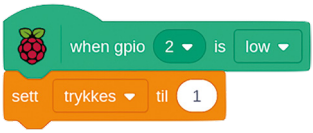
Klikk på det grønne flagget, og følg med på LED-lampene. Først lyser den røde, deretter både den røde og gule, så den grønne, så den gule og til slutt gjentas sekvensen fra den

røde lampen én gang til. Dette mønsteret samsvarer med det som brukes av trafikklys i Storbritannia. Hvis du vil, kan du redigere sekvensen slik at den følger mønstre i andre land.

Hvis du vil simulere en fotgjengerfelt, må programmet kunne registrere at det trykker på knappen . Klikk på den røde åttekanten for å stoppe programmet hvis det kjører. Dra klossen **hvis ellers** til skriptområdet og plasser den slik at den er rett under **gjenta for alltid** -klossen, med trafikklyssekvensen i «hvis»-delen (if then). La det diamantformede feltet være tomt inntil videre.

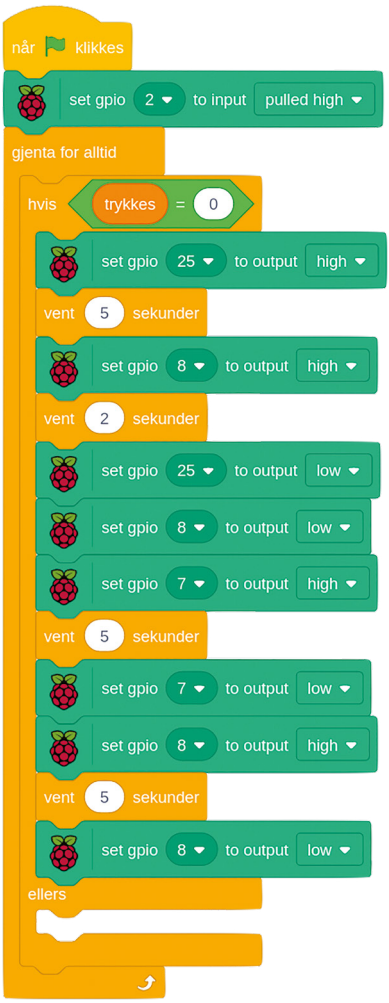


Ved en virkelig fotgjengerovergang endres ikke trafikklyset til rødt straks noen trykker på knappen. Det venter i stedet på neste røde lys i sekvensen. Hvis du vil bygge dette inn i programmet ditt, drar du klossen **when gpio is low** til kodeområdet og velger «2» fra rullegardinlisten. Dra deretter klossen **sett trykket til 1** under denne klossen.



Denne klosstabelen registrerer om det trykkes på knappen, og setter deretter variabelen «trykket» (pushed) til 1. Ved å angi en variabel på denne måten kan du lagre at det ble trykket på knappen, selv om du ikke vil reagere på det med én gang.

Gå tilbake til den opprinnelige klosstabelen og finn klossen **hvis**. Dra operatorklossen **= **
inn i det tomme diamantformede feltet i klossen **hvis**, og dra deretter rapporteringsklossen **trykket** inn i det første tomme feltet. Skriv «0» over «50» til høyre i klossen.



Klikk på det grønne flagget, og se trafikklysene gå gjennom sekvensen. Trykk på trykkbryteren. Først ser det ut som om ingenting skjer, men når sekvensen har nådd slutten – med bare den gule LED-lampen tent – vil trafikklysene slukkes og forbli slukket på grunn av variabelen «trykket» (pushed).

Alt som står igjen nå, er å få gangfeltknappen til å gjøre noe annet enn å slå av lysene. I hovedstablen med klosser går du til klossen **ellers** og drar klossen **set gpio 25 to output high** inn i den. Husk å endre standard GPIO-pinnenummer slik at det samsvarer med pinnen den røde LED-lampen er tilkoblet.

Under denne klossen, fremdeles i klossen **ellers**, lager du et mønster for summeren: Dra blokken **gjenta 10** hit, fyll den med blokkene **set gpio 15 to output high**, **vent 0.2 sekunder** **set gpio 15 to output low** og **vent 0.2 sekunder**. Da endres GPIO-pinneverdiene slik at de samsvarer med pinnen for smmerkomponenten.

Til slutt, under klossen **gjenta 10**, men fremdeles i **ellers** -klossen, legger du til klossen **set gpio 25 to output low** og klossen **sett trykket til 0**. Den siste klossen nullstiller variabelen som lagrer knappetrykket, slik at summersekvensen ikke gjentas i det uendelige.

Klikk på det grønne flagget, og trykk deretter på bryteren på koblingsbrettet. Når sekvensen er fullført, ser du at det røde trafikklyset tennes og summeren utløses for å varsle fotgjengere om at det er trygt å krysse. Etter et par sekunder stopper summeren og trafikklyssekvensen starter igjen. Den fortsetter til neste gang du trykker på knappen.

Gratulerer! Du har programmert ditt eget fullt funksjonelle sett med trafikklys, med gangfelt og det hele!

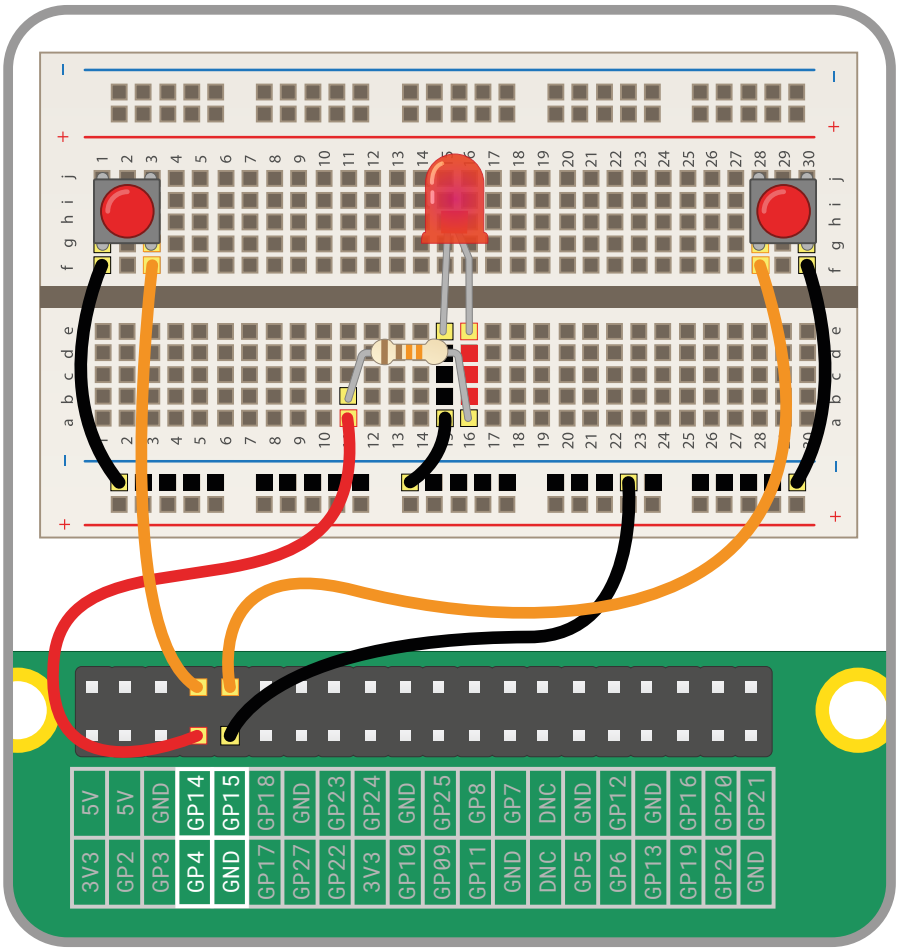
UTFORDRING: KAN DU FORBEDRE DET?

Kan du endre programmet slik at fotgjengeren får lengre tid å krysse? Kan du finne informasjon om andre lands trafikklysmønstre og omprogrammere lampene dine etter disse? Hvordan kan du gjøre LED-lampene svakere?

Python-prosjekt: Hurtigreaksjonsspill

Nå som du vet hvordan du bruker knapper og LED-lamper som inn- og utdataenheter, er du klar til å bygge et dataprogram for dagliglivet: et hurtigreaksjonsspill for to deltakere for å se hvem som reagerer raskest! Til dette prosjektet trenger du et koblingsbrett, en LED-lampe og en 330 Ω-resistor, to trykkbrytere, flere jumperkabler av typen hann/hunn (M2F) og hann/hann (M2M).

Start med å bygge kretsen (**Figur 6-7**). Koble den første bryteren til venstre på koblingsbrettet til GPIO 14-pinnen (merket GP14 i **Figur 6-7**), koble den andre bryteren til høyre på koblingsbrettet til GPIO 15-pinnen (merket GP15) og koble det lange benet på LED-lampen til 330 Ω-resistoren. Deretter kobler du resistoren til GPIO 4-pinnen (merket GP4) på Raspberry Pi og de andre bena på alle komponentene til jordingsskinnen på koblingsbrettet. Til slutt kobler du jordingsskinnen til en jordingspinne (merket GND) på Raspberry Pi.



▲ **Figur 6-7:** Koblingsskjema for hurtigreaksjonsspillet

Start et nytt prosjekt i Thonny og lagre det som **Reaksjonsspill**. Du kommer til å bruke funksjonene **LED** og **button** fra GPIO Zero-biblioteket, samt **sleep** fra tidsbiblioteket. I stedet for å importere de to GPIO Zero-funksjonene på to separate linjer, kan du spare tid og importere dem sammen ved å atskille dem med et kommategn (**,**). Skriv følgende i skriptområdet:

```
from gpiozero import LED, Button
from time import sleep
```

Som før må du fortelle GPIO Zero hvilke pinner de to knappene og LED-lampene er koblet til. Skriv følgende:

```
led = LED(4)
right_button = Button(15)
left_button = Button(14)
```

Legg til instruksjoner for å slå LED-lampen på og av, slik at du kan kontrollere at den fungerer som den skal:

```
led.on()
sleep(5)
led.off()
```

Klikk på Run-knappen. LED-lampen lyser i fem sekunder, slår seg av igjen og programmet avsluttes. Når det gjelder reaksjonsspill, er det imidlertid litt forutsigbart å la LED-lampen slukkes etter nøyaktig 5 sekunder hver gang. Legg til følgende under linjen **from time import sleep**:

```
from random import uniform
```

Som navnet tilsier, kan du generere tilfeldige tall med random-biblioteket (her med en ensartet fordeling – se **rpf.io/uniform**). Finn linjen **sleep(5)** og endre den til:

```
sleep(uniform(5, 10))
```

Klikk på Run-knappen igjen. Denne gangen lyser LED-lampen i et tilfeldig antall sekunder mellom 5 og 10. Tell for å se hvor lang tid det tar før LED-lampen slukkes. Deretter klikker du på Run-knappen noen ganger til. Du vil se at tiden er forskjellig hver gang, noe som gjør programmet mindre forutsigbart.

Hvis du vil gjøre om knappene til utløsere for hver spiller, må du legge til en funksjon. Gå helt nederst i programmet og skriv inn følgende:

```
def pressed(button):
    print(str(button.pin.number) + " vant spillet")
```

Husk at Python bruker innrykk for å vite hvilke linjer som er en del av funksjonen. Thonny vil automatisk rykke inn den andre linjen for deg. Til slutt legger du til følgende to linjer for å registrere at spillerne trykker på knappene. Merk: De skal ikke være innrykket, ellers vil Python behandle dem som en del av funksjonen.

```
right_button.when_pressed = pressed
left_button.when_pressed = pressed
```

Kjør programmet igjen. Denne gangen prøver du å trykke på en av de to knappene så snart LED-lampen slukkes. Du ser en melding for den første knappen som skal trykkes, i Python-skallet nederst i Thonny-vinduet. Dessverre ser du også meldinger hver gang det trykkes på en av knappene, og de bruker pinnenummeret i stedet for et kallenavn for knappen.

Du kan endre dette ved å be om spillernes navn. Under linjen **`from random import uniform`** skriver du følgende:

```
left_name = input("Venstre spillers navn er ")
right_name = input("Høyre spillers navn er ")
```

Gå tilbake til funksjonen og erstatt linjen **`print(str(button.pin.number) + " vant spillet")`** med:

```
    if button.pin.number == 14:
        print (left_name + " vant spillet")
    else:
        print(right_name + " vant spillet")
```

Klikk på Run-knappen, og skriv deretter inn navnene på begge spillerne i Python-skallområdet. Når du trykker på knappen denne gangen (husk å gjøre det så fort du kan etter at LED-lampen slukkes), ser du at spillerens navn skrives ut i stedet for pinnenummeret.

Hvis du vil løse problemet med at alle knappetrykk registreres som vinner, må du legge til en ny funksjon fra sys-biblioteket (sys er en kortform av *system*): **`exit`**. Under den siste **`import`**-linjen skriver du følgende:

```
from os import _exit
```

Deretter går du til slutten av funksjonen, under linjen **`print(right_name + " vant spillet")`**, og skriver følgende:

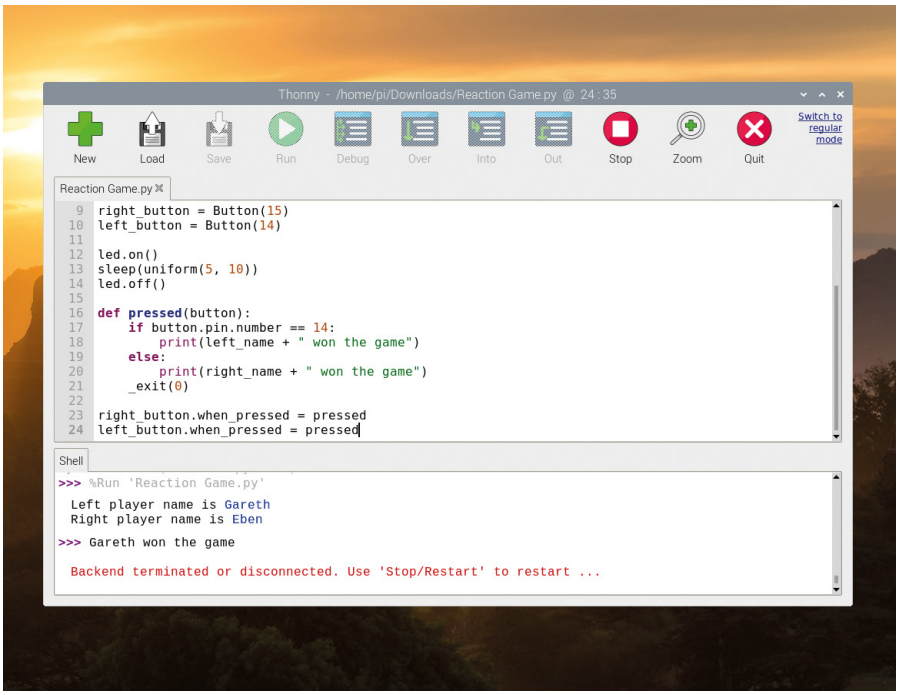
```
    _exit(0)
```

Her er innrykket viktig: **`_exit(0)`** skal innrykkes med fire mellomrom, på linje med **`else:`** to linjer over oppføringen og **`if`** to linjer over den igjen. Denne instruksjonen ber Python om å stoppe programmet etter at det er trykket på den første knappen. Det betyr at den siste spilleren som trykker på knappen, ikke får noen belønning for å tape!

Det ferdige programmet skal nå se slik ut:

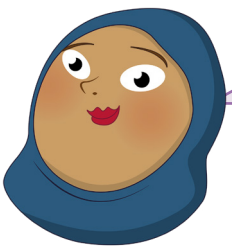
```
from gpiozero import LED, Button
from time import sleep
from random import uniform
from os import _exit

left_name = input("Venstre spillers navn er ")
right_name = input ("Høyre spillers navn er ")
led = LED(4)
right_button = Button(15)
left_button = Button(14)

led.on()
sleep(uniform(5, 10))
led.off()

def pressed(button):
    if button.pin.number == 14:
        print(left_name + " vant spillet")
    else:
        print(right_name + "vant spillet")
    _exit(0)

right_button.when_pressed = pressed
left_button.when_pressed = pressed
```

Klikk på Run-knappen, skriv inn spillernes navn, vent til LED-lampen slukkes, og du vil se navnet på spilleren som vant. Du vil også se en melding fra selve Python: **`Backend terminated or disconnected . Use 'Stop/Restart' to restart ...`** Dette betyr bare at Python har mottatt **`_exit(0)`**-kommandoen og har stanset programmet. Du må imidlertid klikke på Stop-ikonet for å avslutte det helt og klargjøre programmet for en ny runde (**Figur 6-8**).



▲ **Figur 6-8:** Når vinneren er utropt, må du stoppe programmet

Gratulerer! Du har laget ditt eget fysiske spill!



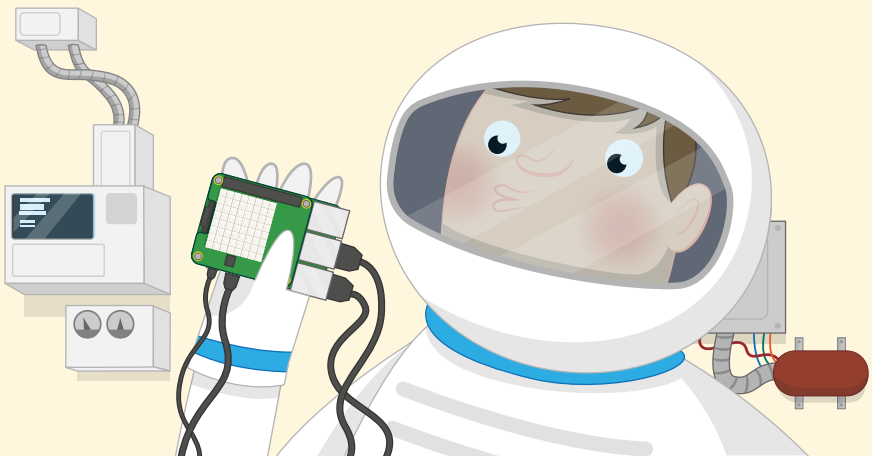
UTFORDRING: FORBEDRE SPILLET

Kan du legge til en løkke slik at spillet kjører kontinuerlig? Husk å fjerne `_exit(0)`-instruksjonen først! Kan du legge til en poengteller, slik at du kan se hvem som vinner over flere runder? Hva med en timer så du kan se hvor lang tid det tok deg å reagere da lampen ble slukket?

Kapittel 7

Fysisk databehandling med Sense HAT

Sense HAT er et multifunksjonelt tilleggsbrett for Raspberry Pi, som er utstyrt med sensorer og en LED-matriseskjerm – akkurat somdet samme brettet som brukes på den internasjonale romstasjonen



Raspberry Pi leveres med støtte for en spesiell type tilleggsbrett som heter *HAT (Hardware Attached on Top)*. HAT-er kan legge til alt fra mikrofoner og lamper til elektroniske releer og skjermer til Raspberry Pi. En bestemt HAT er imidlertid veldig spesiell: Sense HAT.

Sense HAT ble utviklet spesielt for romoppdraget Astro Pi. Astro Pi var et fellesprosjekt mellom Raspberry Pi Foundation, UK Space Agency og Det europeiske romfartsbyrået (ESA). Raspberry Pi-brett og Sense HAT-er var med på ferden til den internasjonale romstasjonen ombord et lastefartøy av typen Orbital Science Cygnus. Etter at Sense HAT-er, som astronautene ga kallenavnene Ed og Izzy, kom trygt i bane høyt over jorden, har skoleelever i hele Europa brukt dem til å kjøre kode og utføre vitenskapelige eksperimenter.

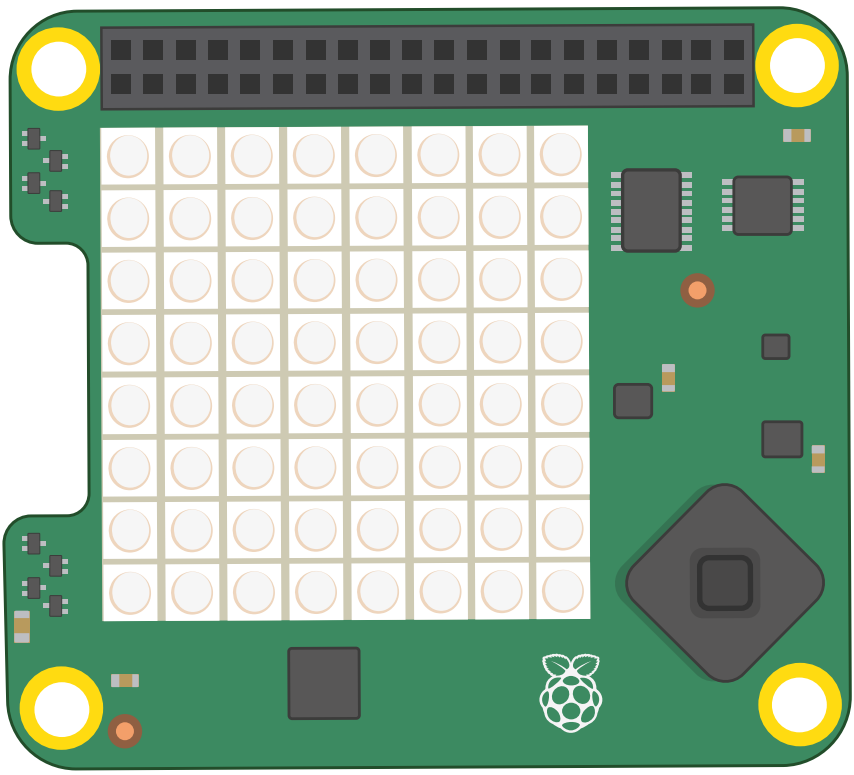
Ed og Izzy befinner seg riktignok litt langt unna, så det blir nok vanskelig å bruke dem selv. Derimot finner du også den samme Sense HAT-maskinvaren her på jorden – hos alle Raspberry Pi-forhandlere. Hvis du ikke vil kjøpe en Sense HAT akkurat nå, kan du simulere en ved å bruke programvare.

EKTE ELLER SIMULERT

Det er best å lese dette kapitlet når du har en Sense HAT festet til en GPIO-pinnerekke på Raspberry Pipinnerekke. Hvis du ikke har en slik rekke, kan du bare hoppe over avsnittet med overskriften «Installere Sense HAT» og prøve prosjektene i Sense HAT Emulator i stedet. Det fungerer like godt.

Vi presenterer Sense HAT

Sense HAT er et kraftig, multifunksjonelt tillegg for Raspberry Pi. Sense HAT har en 8 × 8 matrise med 64 røde, grønne og blå (RGB) programmerbare LED-lamper som kan produsere mange millioner farger. I tillegg har den en femveis joystick og seks innebygde sensorer.



Gyroskopsensor: Brukes til å registrere vinkelendringer over tid, teknisk kjent som *vinkelhastighet*, ved å holde rede på retningen til jordens tyngdefelt – kraften som trekker ting mot sentrum av planeten. Enkelt sagt kan den gyroskopiske sensoren fortelle når du roterer Sense HAT i forhold til jordoverflaten og hvor raskt den roterer.

Akselerometer: Den ligner på gyroskopsensoren, men i stedet for å overvåke en vinkel i forhold til jordens tyngdekraft, måler den akselerasjonskraften i flere retninger. Avlesninger (data) fra de to sensorene kan brukes kombinert til å spore hvilken vei en Sense HAT peker og hvordan den beveger seg.

Magnetometer: Dette er en sensor som måler styrken på et magnetfelt og kan brukes til å spore Sense HATs bevegelser. Magnetometeren finner retningen på magnetisk nord ved å måle jordens naturlige magnetfelt. Denne sensoren kan også brukes til å registrere metallgjenstander og til og med elektriske felt. Alle disse tre sensorene er innebygd i én enkelt brikke, merket «ACCEL/GYRO/MAG» på kretskortet til Sense HAT.

Fuktighetssensor: Denne måler mengden vanndamp i luften, kalt *relativ fuktighet*. Relativ fuktighet kan variere fra 0 %, det vil si ikke noe vann i det hele tatt, til 100 %, hvor luften er helt mettet. Fuktighetsdata kan brukes til å finne ut når det vil begynne å regne.

Barometrisk trykkføler: også kalt *barometer*, måler lufttrykket. Selv om folk flest kjenner barometertrykk fra værmeldingen, har barometeret en hemmelig anvendelse. Det kan registrere når du klatrer opp eller ned en bakke eller et fjell, ettersom luften blir tynnere og har lavere trykk jo lenger du fjerner deg fra jordens havnivå.

Temperaturføler: Måler hvor varmt eller kaldt det omgivende miljøet er. Det kan også påvirkes av hvor varm eller kald Sense HAT er: Hvis du bruker etui, kan avlesningene være høyere enn forventet. Sense HAT har ingen egen temperatursensor. Den bruker i stedet temperatursensorer som er innebygd i fuktighetssensoren og barometeret. Et program kan bruke én eller begge av disse sensorene; det er opp til deg.

SENSE HAT PÅ RASPBERRY PI 400

Sense HAT er fullt kompatibel med Raspberry Pi 400 og kan settes rett inn i GPIO-pinnerekken på baksiden. Hvis du gjør det, betyr det imidlertid at LED-lampene vender bort fra deg, og brettet vil være opp-ned.

Du kan løse dette problemet ved å bruke en GPIO-forlengelseskabel eller et -brett. Kompatible tilleggsfunksjoner inkluderer Black HAT Hack3r fra Pimoroni. Du kan bruke Sense HAT med selve Black HAT Hack3r-brettet eller bare bruke den medfølgende 40-pinners båndkabelen som en tilleggsfunksjon. Sjekk alltid produsentens instruksjoner for å være sikker på at du kobler kabelen og Sense HAT riktig vei!

Vi presenterer Sense HAT

Hvis du har en fysisk Sense HAT, begynner du med å pakke den ut. Sjekk om du har alle delene: selve Sense HAT, fire søyler av metall eller plast, såkalte *avstandsstykker*, og åtte skruer. Det kan også følge med noen metallpinner i en svart plaststrimmel, som GPIO-pinnene på Raspberry Pi. I så fall skyver du strimmelen med pinnene opp gjennom bunnen av Sense HAT til du hører et klikk.

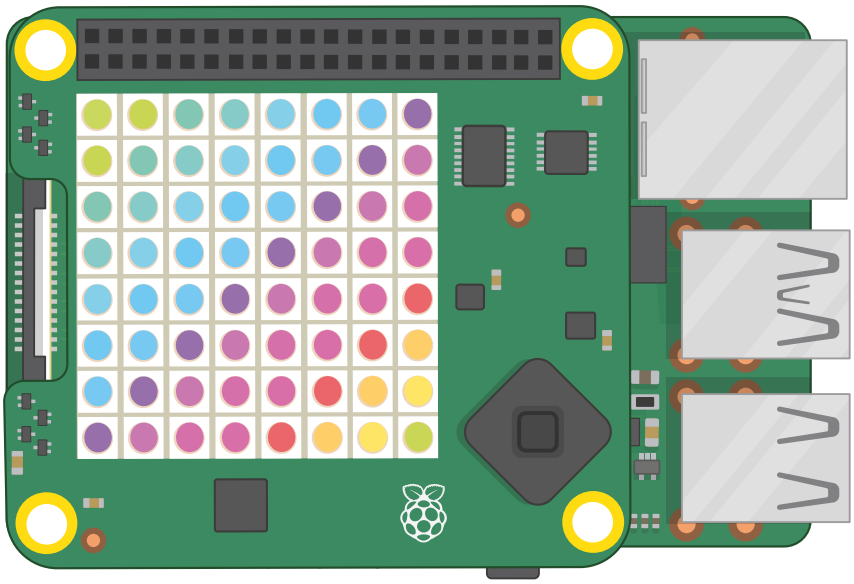
Avstandsstykkene skal forhindre at Sense HAT bøyes og strekkes mens du bruker joysticken. Selv om Sense HAT fungerer uten at de er installert, kan de brukes til å beskytte Sense HAT, Raspberry Pi og GPIO-pinnerekken.

ADVARSEL!

HAT-moduler (Hardware Attached on Top) skal bare kobles til og fra GPIO-pinnerekken når Raspberry Pi er slått av og frakoblet strømforsyningen. Pass på at HAT-en ligger flatt når du installerer den, og dobbeltsjekk at den er innrettet etter pinnene på GPIO-pinnerekken før du skyver den ned.

Installer avstandsstykkene ved å skyve opp fire av skruene fra bunnen av Raspberry Pi gjennom de fire monteringshullene i hvert hjørne. Deretter dreier du avstandsstykkene inn på skruene. Skyv Sense HAT ned på Raspberry Pis GPIO-pinnerekke. Pass på at den er riktig plassert med pinnene under og så flatt som mulig. Til slutt skrur du de fire siste skruene gjennom monteringshullene på Sense HAT og inn i avstandsstykkene du installerte tidligere. Hvis den er riktig installert, skal Sense HAT ligge flatt og plant og ikke bøye seg eller vingle når du trykker på joysticken.

Koble Raspberry Pi til strømforsyningen igjen, så vil du se at LED-lampene på Sense HAT lyser i et regnbuemønster (**Figur 7-1**) og deretter slukkes igjen. Sense HAT er nå installert.

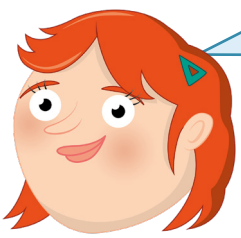


▲ **Figur 7-1:** Et regnbuemønster vises når du slår den på for første gang

Hvis du vil fjerne Sense HAT igjen, løsner du bare de øverste skruene og løfter HAT-en av. Vær forsiktig så du ikke bøyer pinnene på GPIO-pinnerekken. HAT-en sitter ganske godt, så du må kanskje lirke den av med en liten skrutrekker. Deretter fjerner du avstandsstykkene fra Raspberry Pi.

Hei, Sense HAT!

Når du begynner å bruke Sense HAT, starter du på samme måte som med andre programmeringsprosjekter. Lag et program som ruller en velkomstmelding over LED-skjermen. Hvis du bruker Sense HAT-emulatoren, kan du laste den inn nå ved å klikke på menyikonet for Raspberry Pi OS, velge programmeringskategorien og klikke på Sense HAT Emulator.

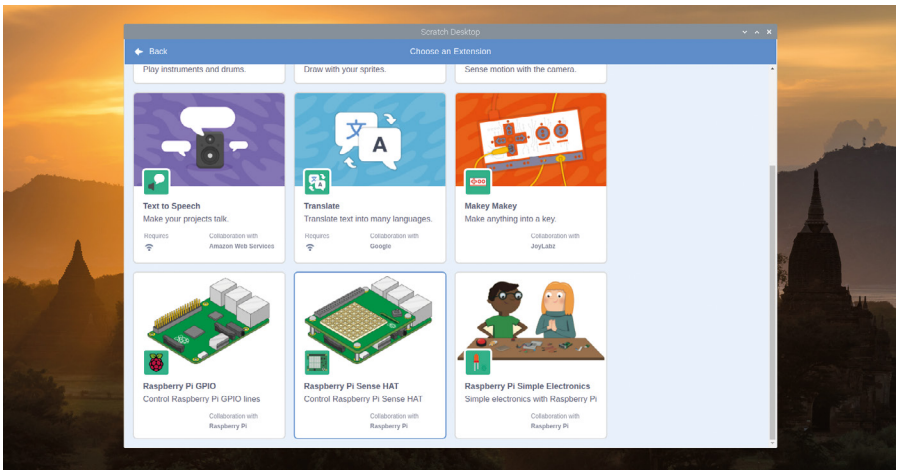


ERFARING MED PROGRAMMERING

Dette kapitlet forutsetter at du har erfaring med Scratch 3 eller Python og Thonnys integrerte utviklingsmiljø (IDE), avhengig av om du jobber deg gjennom eksemplene på Scratch- eller Python-koding – eller begge deler! Hvis du ikke allerede har gjort det, går du til **Kapittel 4: Programmering med Scratch** eller **Kapittel 5: Programmering med Python** og gjennomfører prosjektene i det kapitlet først.

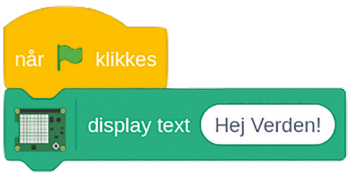
Hilsen fra Scratch

Last opp Scratch 3 fra Raspberry Pi OS-menyen. Klikk på Hent tilleggsfunksjon nederst til venstre i Scratch-vinduet. Klikk på tilleggsfunksjonen Raspberrypå Pi Sense HAT (**Figur 7-2**). Dette laster inn klossene du trenger for å kontrollere de ulike funksjonene til Sense HAT, inkludert LED-skjermen. Når du trenger dem, finner du dem i kategorien Raspberry Pi Sense HAT.

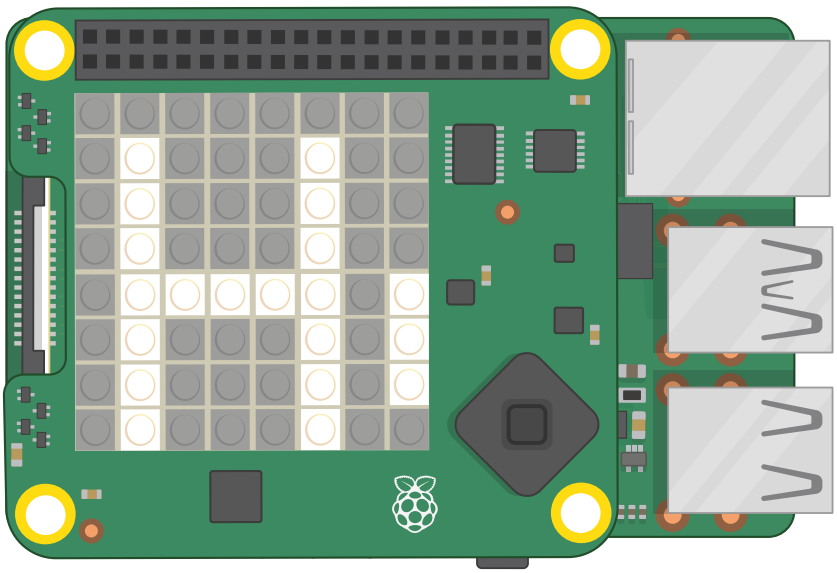


Figur 7-2: Legge til tilleggsfunksjonen Raspberry Pi Sense HAT i Scratch 3

Begynn med å dra hendelsesklossen når klikkes til skriptområdet, og plasser deretter blokken klossen display text Hello! rett under den. Endre teksten på klossen til display text Hei Verden!.

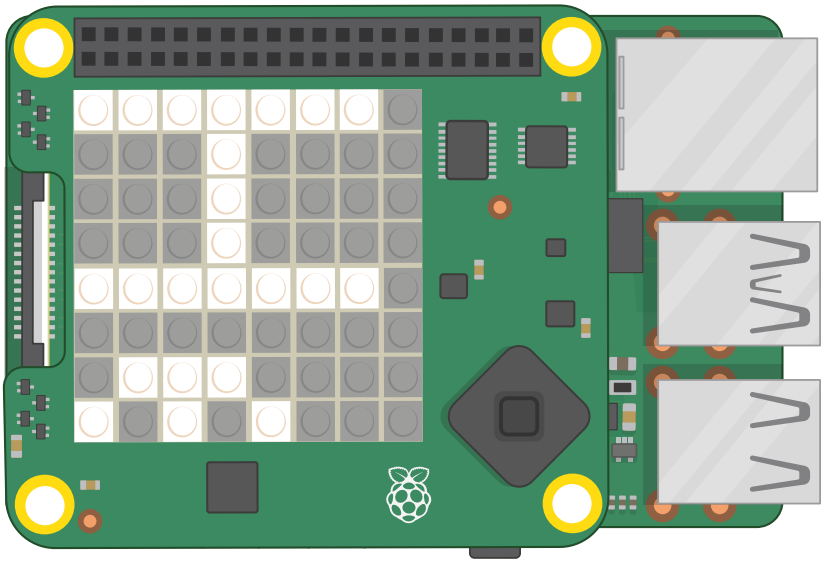


Klikk på det grønne flagget i sceneområdet, og følg med på Sense HAT eller Sense HAT-emulatoren. Meldingen ruller sakte over Sense HATs LED-matrise og tenner LED-pikslene for å forme hver bokstav i tur og orden (**Figur 7-3**, på neste side). Gratulerer, programmet fungerer!



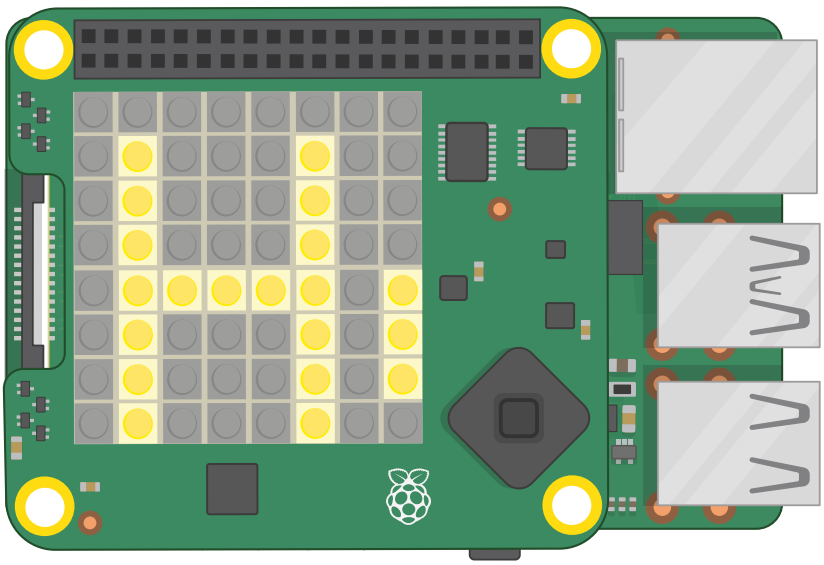
Figur 7-3: Meldingen ruller over LED-matrisen

Nå som du kan få en enkel melding til å rulle over matrisen, er det på tide å kontrollere hvordan meldingen skal vises. Du kan endre både meldingen og meldingens retning – hvilken vei meldingen vises på Sense HAT. Dra klossen set rotation to 0 degrees fra klosspaletten, og sett den inn under når klikkes og over display text Hei Verden!. Deretter klikker du på nedpilen ved siden av 0 og endrer verdien til 90. Klikk på det grønne flagget, så ser du den samme meldingen som før, men i stedet for å rulle fra venstre mot høyre, vil den rulle nedenfra og opp (**Figur 7-4**) – du må snu hodet eller dreie på Sense HAT for å lese den!



Figur 7-4: Denne gangen ruller meldingen loddrett

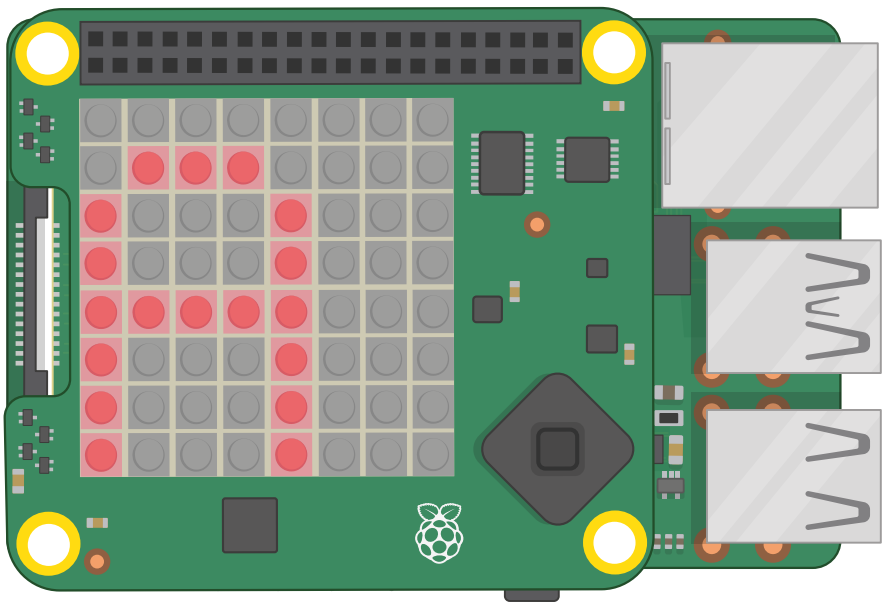
Nå kan du endre retningen tilbake til 0. Deretter drar du klossen set colour mellom set rotation to 0 degrees og display text Hei Verden!. Klikk på fargen på slutten av klossen for å åpne fargevelgeren i Scratch og finne en flott gulfarge. Deretter klikker du på det grønne flagget for å se hvordan programmet oppfører seg nå (**Figur 7-5**).



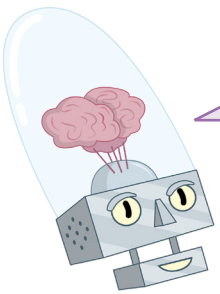
Figur 7-5: Endre fargen på teksten

Til slutt drar du klossen set background mellom set colour to gul og display text Hei Verden!, og klikker på fargen for å få åpne fargevelgeren igjen. Denne gangen påvirker ikke fargevalget LED-lampene som utgjør meldingen, men LED-lampene som ikke gjør det – det vil si bakgrunnen. Finn en fin blåfarge, og klikk på det grønne flagget igjen. Denne gangen vil meldingen ha en sterk gulfarge mot blå bakgrunn. Prøv deg fram med fargene for å finne favorittkombinasjonen din – ikke alle farger fungerer godt sammen.

I tillegg til å rulle hele meldinger over LED-matrisen kan du vise enkeltbokstaver. Dra klossen display text vekk fra skriptområdet for å slette den, og dra deretter klossen display character A til skriptområdet i stedet. Klikk på det grønne flagget, så ser du forskjellen. Denne klossen viser bare én bokstav om gangen, og bokstaven blir værende på Sense HAT uten å rulle eller forsvinne til du sier fra. De samme fargekontrollklossene gjelder for denne klossen som for klossen display text. Prøv å endre fargen på bokstaven til rødt (**Figur 7-6**).



▲ **Figur 7-6:** Vise en enkeltbokstav



UTFORDRING: GJENTA MELDINGEN

Kan du bruke dine kunnskaper om løkker til å få en rullemelding til å gjenta seg selv? Kan du lage et program som staver et ord bokstav for bokstav i forskjellige farger?

Hilsen fra Python

Last inn Thonny ved å klikke på ikonet for Raspberry-menyen, velg Utvikling og klikk på Thonny. Hvis du bruker Sense HAT-emulatoren og den dekkes av Thonny-vinduet, klikker og holder du nede museknappen på tittellinjen i ett av vinduene – øverst, i blått – og drar den rundt på skrivebordet til du kan se begge vinduene.

ENDRING AV PYTHON-LINJE

Python-kode som er skrevet for en fysisk Sense HAT, kjører på Sense HAT-emulatoren, og omvendt, med bare én endring. Hvis du bruker Sense HAT-emulatoren med Python, må du endre linjen fra `sense_hat import SenseHat` i alle programmene fra dette kapitlet til `from sense_emu import SenseHat`. Hvis du senere ønsker å kjøre dem på en fysisk Sense HAT igjen, er det bare å endre linjen igjen.

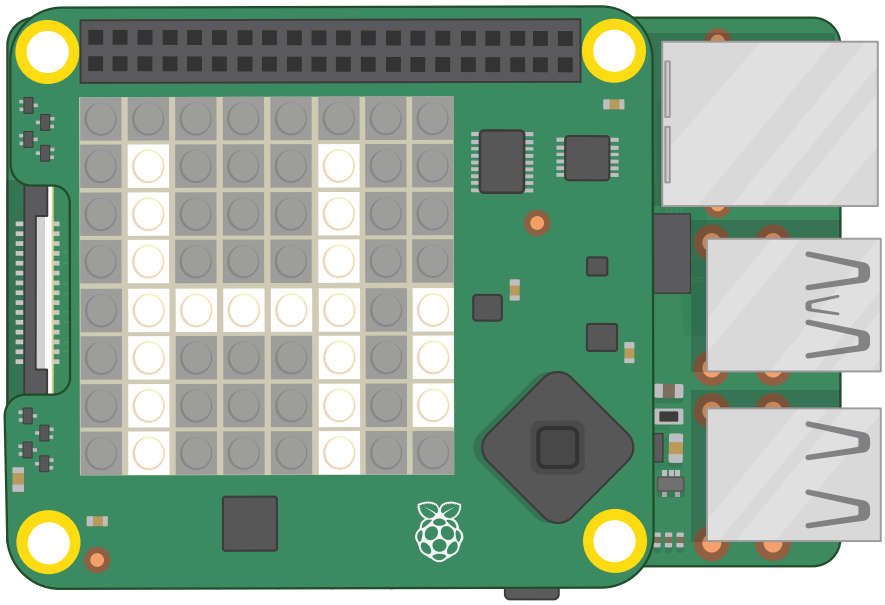
Hvis du vil bruke Sense HAT, eller Sense HAT-emulatoren, i et Python-program må du importere Sense HAT-biblioteket. Skriv følgende i skriptområdet. Husk å bruke **`sense_emu`** (i stedet for **`sense_hat`**) hvis du bruker Sense HAT-emulatoren:

```
from sense_hat import SenseHat
sense = SenseHat()
```

Sense HAT-biblioteket har en enkel funksjon som henter en melding, formaterer den slik at den kan vises på LED-skjermen og lar den rulle jevnt. Skriv følgende:

```
sense.show_message("Hei, verden!")
```

Lagre programmet som **Hei Sense HAT**, og klikk på Run-knappen. Du ser en melding som ruller langsomt over LED-matrisen på Sense HAT og tenner LED-pikslene for å forme hver bokstav i tur og orden (**Figur 7-7**, på neste side). Gratulerer, programmet fungerer!



Figur 7-7: Få en melding til å rulle over LED-matrisen

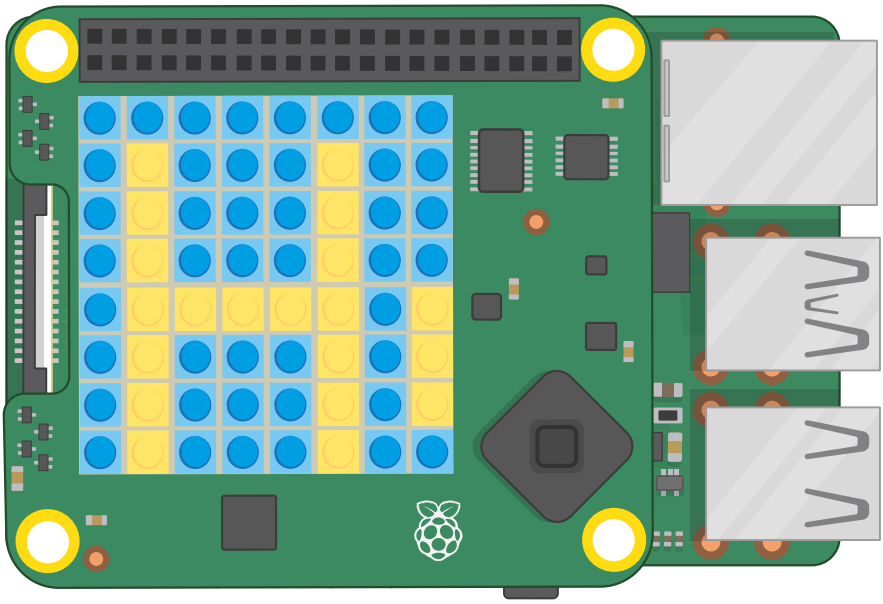
Funksjonen **`show_message()`** har imidlertid flere triks på lur. Gå tilbake til programmet og rediger den siste linjen så den sier følgende:

```
sense.show_message("Hei, verden!", text_colour=(255, 255, 0),
back_colour=(0, 0, 255), scroll_speed=(0.05))
```

Disse tilleggsinstruksjonene, atskilt med komma, kalles *parametere*. De kontrollerer ulike aspekter av funksjonen **`show_message()`**. Den enkleste er **`scroll_speed=()`**, som regulerer hvor raskt meldingen ruller over skjermen. Hvis verdien er 0,05, ruller den omtrent dobbelt så fort som vanlig hastighet. Jo høyere tall, desto lavere hastighet.

Parametrene **`text_colour=()`** og **`back_colour=()`** – skrevet på britisk engelsk, i motsetning til de fleste Python-instruksjonene – angir fargen på henholdsvis skriften og bakgrunnen. De godtar ikke fargenavn. Du må oppgi ønsket farge som et tresifret tall. Det første tallet står for mengden rødt i fargen. 0 angir ikke noe rødt i det hele tatt til og 255 angir så mye rødt som mulig. Det andre tallet er mengden grønt i fargen og det tredje tallet mengden blått. Samlebetegnelsen er *RGB* – rødt, grønt og blått.

Klikk på Run-ikonet og følg med på Sense HAT. Denne gangen ruller meldingen betydelig raskere og har en sterk gulfarge mot en blå bakgrunn (**Figur 7-8**). Prøv å endre parametrene for å finne en hastighet og fargekombinasjon du liker.



Figur 7-8: Endre fargen på meldingen og bakgrunnen

Hvis du vil bruke kallenavn i stedet for RGB-verdier for å angi fargene, må du opprette variabler. Over linjen **`sense.show_message()`** legger du til følgende:

```
yellow = (255, 255, 0)
blue = (0, 0, 255)
```

Gå tilbake til linjen **`sense.show_message()`** og endre den til:

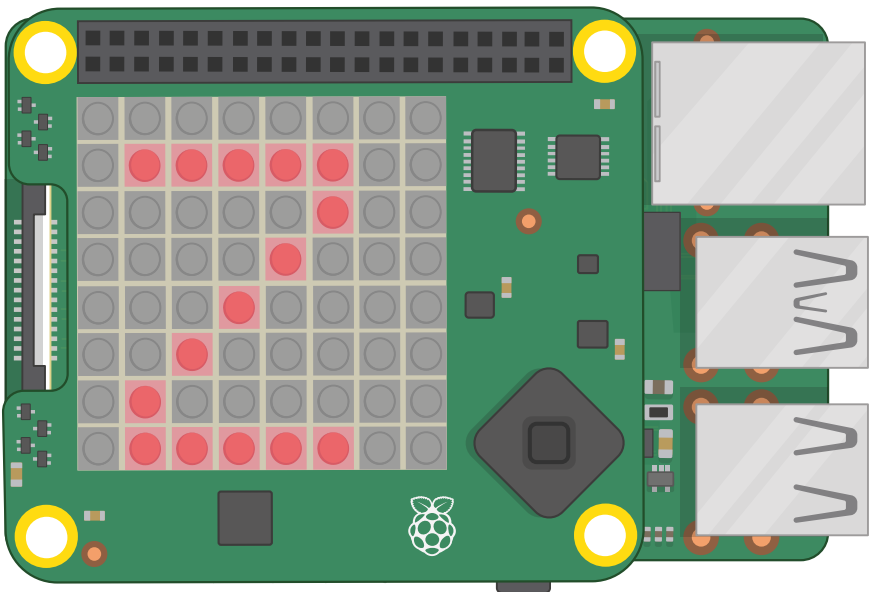
```
sense.show_message("Hei, verden!", text_colour=(yellow), back_
colour=(blue), scroll_speed=(0.05))
```

Klikk på Run-ikonet igjen, så ser du at ingenting er endret. Meldingen din er fortsatt gul mot blå bakgrunn. Men denne gangen har du brukt variabelnavnene for å gjøre koden mer lesbar. I stedet for en streng med tall forklarer koden hvilken farge den stiller inn. Du kan definere så mange farger du vil. Prøv å legge til en variabel kalt «rød» med verdiene 255, 0 og 0, en variabel kalt «hvit» med verdiene 255, 255, 255 og en variabel kalt «svart» med verdiene 0, 0 og 0.

Nå vet du både hvordan du ruller hele meldinger over skjermen, og hvordan du viser enkeltbokstaver. Slett hele linjen **`sense.show_message()`**, og skriv følgende i stedet:

```
sense.show_letter("Z")
```

Klikk på Run, så ser du at bokstaven «Z» vises på Sense HAT-skjermen. Denne gangen blir den stående. I motsetning til meldinger ruller ikke enkeltbokstaver automatisk. Du kan også kontrollere `sense.show_letter()` med de samme fargeparametrene som `sense.show_message()`. Prøv å endre fargen på bokstaven til rødt (**Figur 7-9**).



▲ **Figur 7-9:** Vise en enkeltbokstav

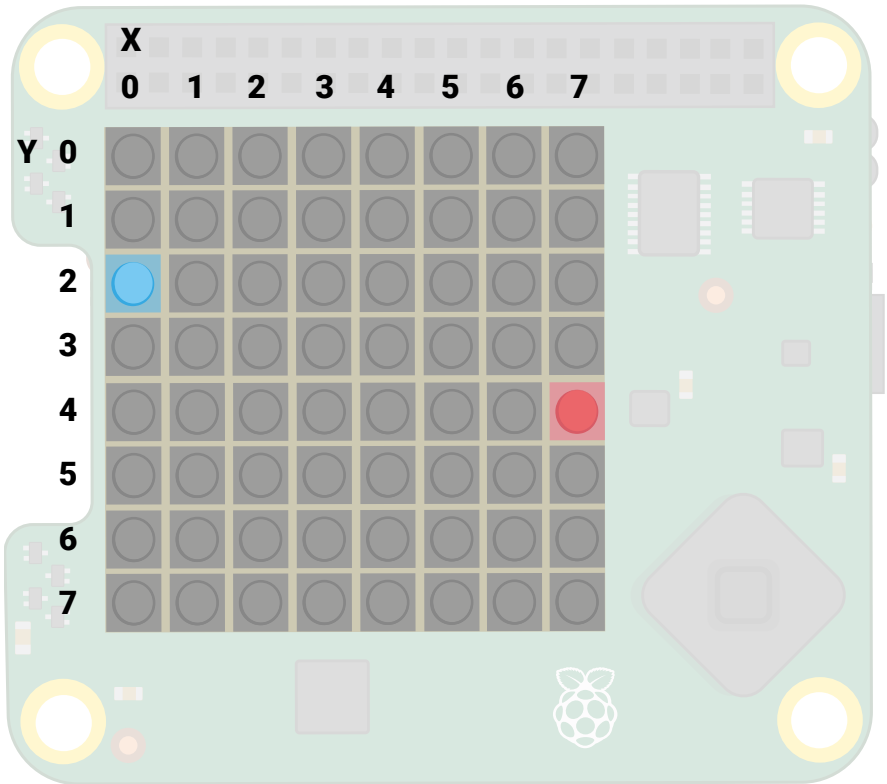
UTFORDRING: GJENTA MELDINGEN

Kan du bruke dine kunnskaper om løkker for å få en rullemelding til å gjenta seg selv? Kan du lage et program som staver et ord bokstav for bokstav i forskjellige farger? Hvor raskt kan du få en melding til å rulle?

Neste trinn: Tegne med lampe

LED-skjermen på Sense HAT er ikke bare beregnet på meldinger – du kan også vise bilder. Hver LED-lampe kan behandles som en enkelt piksel – kort for *bildeelement* – i et bilde du velger. På den måten kan du live opp programmene med bilder og til og med animasjon.

Hvis du vil lage tegninger, må du imidlertid vite hvordan du endrer LED-lamper enkeltvis. Du må forstå hvordan LED-matrisen på Sense HAT er satt opp, før du kan skrive et program som slår de riktige LED-lampene av eller på.



Figur 7-10: LED-matrisens koordinatsystem

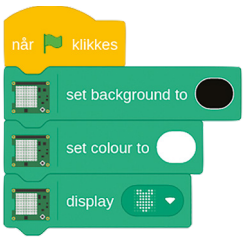
finnesHver rad og hver kolonne på skjermen har åtte LED-lamper på (**Figur 7-10**). Når du teller LED-lampene, starter du med 0 og slutter på 7 – som med de fleste programmeringsspråk. Den første LED-lampen er øverst til venstre, den siste er nederst til høyre. Du kan bruke tallene fra radene og kolonnene til å finne *koordinatene* for alle LED-lamper på matrisen. Den blå LED-lampen i matrisen på bildet har koordinatene 0, 2. Den røde LED-lampen har koordinatene 7, 4. X-aksen, over matrisen, kommer først, etterfulgt av Y-aksen, nedover matrisen.

Når du planlegger bilder som skal tegnes på Sense HAT, kan det være lurt å tegne dem for hånd først (på rutepapir). Du kan også planlegge ting i et regneark som LibreOffice Calc.

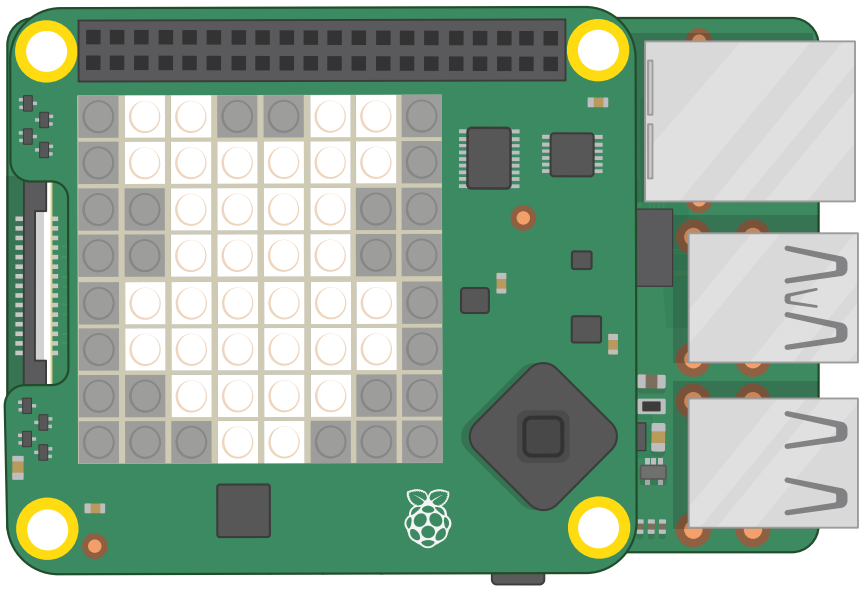
Bilder i Scratch

Start et nytt prosjekt i Scratch, og lagre det eksisterende programmet hvis du vil beholde det. Hvis du har jobbet deg gjennom prosjektene i dette kapitlet, holdefortsetter tilleggsfunksjonen Raspberry Pi Sense HAT å være innlastet i Scratch 3. Hvis du har lukket og åpnet Scratch 3 igjen siden forrige prosjekt, laster du inn tilleggsfunksjonen ved hjelp av knappen Hent tilleggsfunksjon. Dra hendelsesklossen når klikkes til kodeområdet. Deretter drar du klossene set background og set colour under den igjen. Rediger begge for å sette

bakgrunnsfargen til svart og fargen til hvit. Du endrer den til svart ved å skyve glidebryterne Lysstyrke og Metning til 0, og du endrer den til hvit ved å sette Lysstyrke til 100 og Metning til 0. Du må gjøre dette på begynnelsen av hvert Sense HAT-program, ellers vil Scratch ganske enkelt bruke de siste fargene du valgte – selv om du valgte dem i et annet program. Til slutt drar du klossen **display bringebær** til bunnen av programmet.



Klikk på det grønne flagget, så ser du at LED-lampene på Sense HAT får ettenne til å lyses i form av et bringebær (**Figur 7-11**).



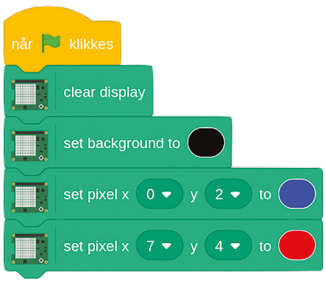
Figur 7-11: Ikke se direkte på LED-lampene når de lyser sterkt hvitt

Du trenger ikke å bruke den forhåndsinnstilte bringebærformen heller. Klikk på nedpilen ved siden av bringebæret for å aktivere tegnemodus. Du kan klikke på en hvilken som helst LED-lampe i mønsteret for å slå den av eller på. De to knappene nederst slår alle LED-lamper av eller på. Prøv å tegne ditt eget mønster. Deretter klikker du på den grønne pilen for å se mønsteret på Sense HAT. Prøv også å endre farge og bakgrunnsfarge ved å bruke klossene ovenfor.

Når du er ferdig, drar du de tre klossene inn i klosspaletten for å slette dem. Deretter

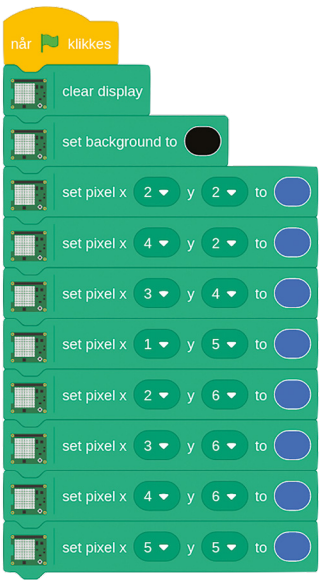
plasserer du klossen **clear display** under **når klikkes**. Klikk på det grønne flagget, og alle LED-lampene slukkes.

Når du skal lage et bilde, må du vite hvordan du kontrollerer enkeltpiksler og gir dem forskjellige farger. Det kan du gjøre ved å koble redigerte klosser av typen **display bringebær** til klosser av typen **set colour**. Du kan også redigere en piksel om gangen. Hvis du vil lage din egen versjon av LED-matriseeksemplet som er avbildet i starten av denne delen, med to spesielt utvalgte LED-lamper som tennes i rødt og blått, beholder du klossen **clear display** øverst i programmet og drar klossen **set background** under den. Endre klossen **set background** til svart, og dra deretter to klosser av typen **set pixel x 0 y 0** under den. Til slutt redigerer du disse klossene slik:

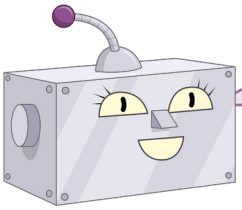


Klikk på det grønne flagget, så ser du at LED-lampene tennes på samme måte som på matrisebildet (**Figur 7-10**) på side 165. Gratulerer, nå kan du kontrollere individuelle LED-lamper.

Rediger de eksisterende pikselklossene på følgende måte, og dra flere til bunnen til du har følgende program:



Før du klikker på det grønne flagget kan du prøve å gjette hvilket bilde som du vil se, basert på LED-matrisekoordinatene du har brukt. Deretter kjører du programmet og ser om du har rett.



UTFORDRING: NYE DESIGN

Kan du designe flere bilder? Prøv å få tak i graf- eller rutepapir, og bruk det til å skissere bildet for hånd først. Kan du tegne et bilde og få fargene til å endre seg?

Bilder i Python

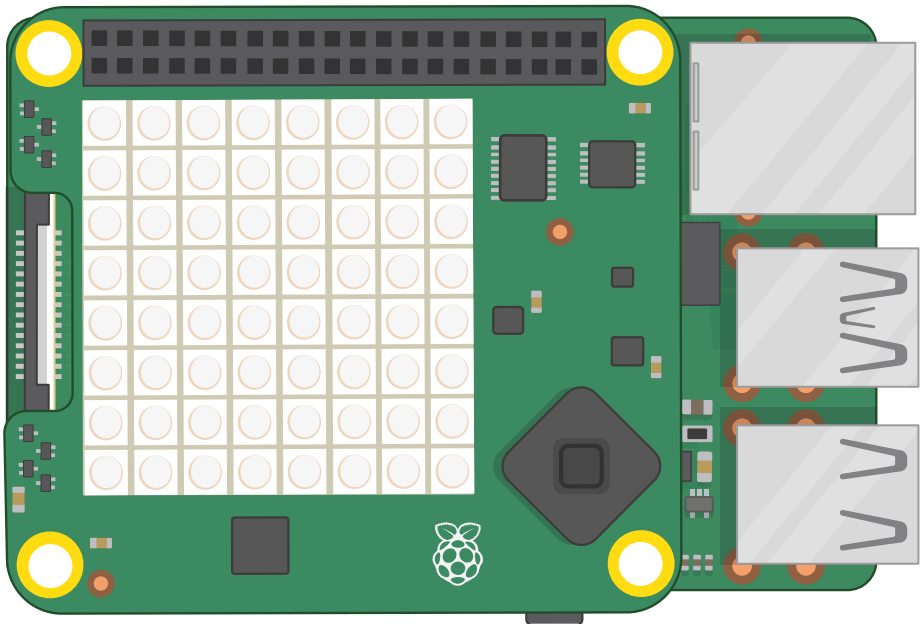
Start et nytt program i Thonny og lagre det som Sense HAT-tegning. Deretter skriver du følgende – husk å bruke **`sense_emu`** (i stedet for **`sense_hat`**) hvis du bruker emulatoren:

```
from sense_hat import SenseHat
sense = SenseHat()
```

Husk at du trenger begge disse linjene i programmet for å kunne bruke Sense HAT. Deretter skriver du:

```
sense.clear(255, 255, 255)
```

Mens du unngår å se direkte på LED-lampene på Sense HAT, klikker du på Run-ikonet. De skal alle lyse sterkt hvitt (**Figur 7-12**) – derfor må du ikke se direkte på dem når du kjører programmet!

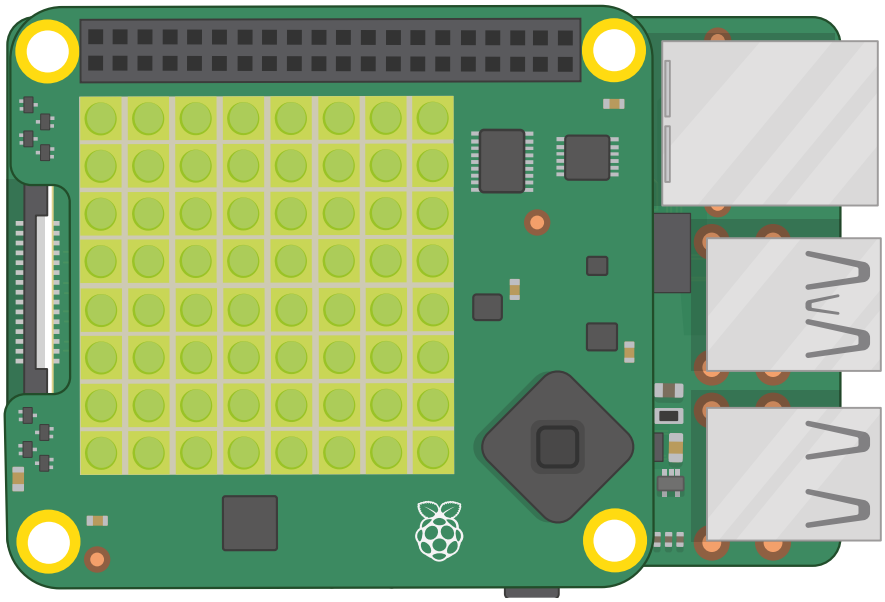


Figur 7-12: Ikke se direkte på LED-lampene når de lyser sterkt hvitt

`sense.clear()` sletter LED-lampene fra all tidligere programmering, men godtar RGB-fargeparametere. Det betyr at du kan endre skjermen til hvilken som helst farge. Forsøk å endre linjen til følgende:

```
sense.clear(0, 255, 0)
```

Klikk på Run, og Sense HAT blir lysegrønn (**Figur 7-13** på neste side). Eksperimenter med forskjellige farger, eller legg til fargenavnvariablene du opprettet for Hei verden-programmet for å gjøre ting enklere å lese.

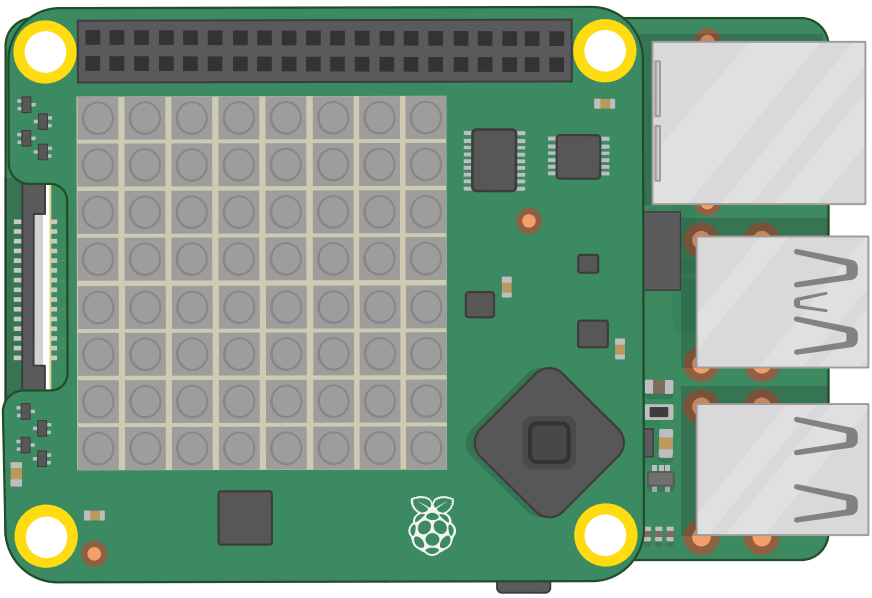


Figur 7-13: LED-matrisen lyser lysegrønt

Hvis du vil slette LED-lampene, bruker du RGB-verdiene for svart: 0 rød, 0 blå og 0 grønn. Det finnes imidlertid en enklere måte. Rediger linjen i programmet til:

```
sense.clear()
```

Sense HAT blir helt svart. Hvis funksjonen `sense.clear()` ikke har noe mellom parentesene, er dette det samme som å fortelle programmet at alle LED-lampene skal endres til svart – dvs. slå dem av (**Figur 7-14**). Du bruker denne funksjonen når du vil slå av alle LED-lampene i programmene.



Figur 7-14: Bruke funksjonen `sense.clear` for å slå av alle LED-lampene

Hvis du vil lage din egen versjon av LED-matrisen som er avbildet tidligere i dette kapitlet, med to spesielt utvalgte LED-lamper tent i rødt og blått, legger du til følgende linjer i programmet etter **`sense.clear()`**:

```
sense.set_pixel(0, 2, (0, 0, 255))
sense.set_pixel(7, 4, (255, 0, 0))
```

De to første tallene angir pikselens plassering på matrisen, X-aksen (på tvers) etterfulgt av Y-aksen (nedover). De to neste tallene (i egne parenteser) er RGB-verdiene for pikselfargen. Klikk på Run-knappen for å se effekten. To LED-lamper på Sense HAT tennes, akkurat som i **Figur 7-10** på side 165.

Slett de to linjene, og skriv inn følgende:

```
sense.set_pixel(2, 2, (0, 0, 255))
sense.set_pixel(4, 2, (0, 0, 255))
sense.set_pixel(3, 4, (100, 0, 0))
sense.set_pixel(1, 5, (255, 0, 0))
sense.set_pixel(2, 6, (255, 0, 0))
sense.set_pixel(3, 6, (255, 0, 0))
sense.set_pixel(4, 6, (255, 0, 0))
sense.set_pixel(5, 5, (255, 0, 0))
```

Sjekk koordinatene og sammenligne dem med matrisen før du klikker på Run. Kan du gjette hvilket bilde som blir tegnet med disse instruksjonene? Klikk på Run for å finne ut om du har rett.

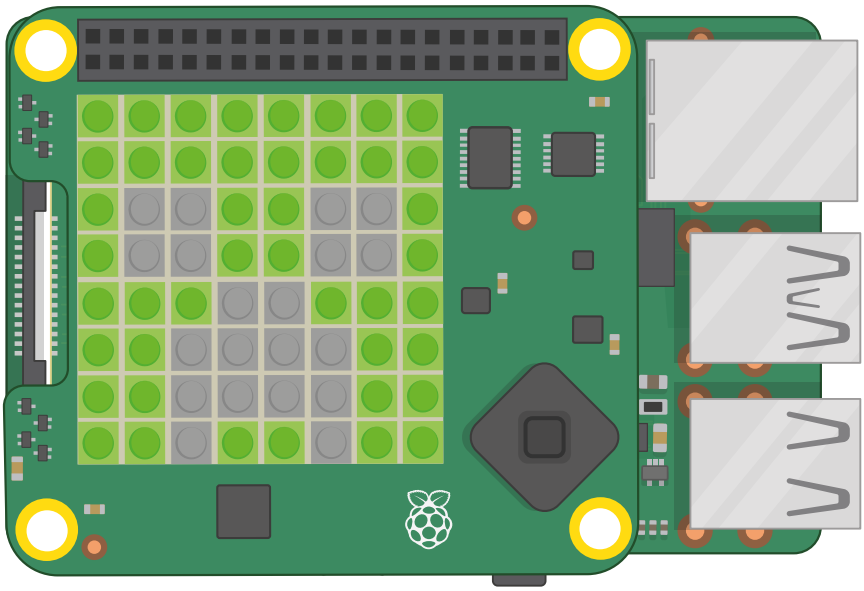
Det går imidlertid sakte å tegne et detaljert bilde ved å bruke funksjonene **`set_pixel()`** enkeltvis. Du kan endre flere piksler samtidig for å få det til å gå raskere. Slett alle **`set_pixel()`**-linjene og skriv følgende:

```
g = (0, 255, 0)
b = (0, 0, 0)

creeper_pixels = [
    g, g, g, g, g, g, g, g,
    g, g, g, g, g, g, g, g,
    g, b, b, g, g, b, b, g,
    g, b, b, g, g, b, b, g,
    g, g, g, b, b, g, g, g,
    g, g, b, b, b, b, g, g,
    g, g, b, b, b, b, g, g,
    g, g, b, g, g, b, g, g
]

sense.set_pixels(creeper_pixels)
```

Her er det mye, men start med å klikke på Run for å se om du kjenner igjen en bestemt creeper-piksel. De to første linjene lager to variabler som inneholder farger: grønt og svart. For å gjøre koden for tegningen enklere å skrive og lese er variablene enkeltbokstaver: g for grønt og b for svart (black).

Neste kodeblokk lager en variabel som inneholder fargeverdier for alle 64 piksler på LED-matrisen, atskilt med komma og omsluttet av hakeparenteser. I stedet for tall bruker det fargevariablene du opprettet tidligere. Se nøye, husk at «g» står for grønn og «b» står for svart. Du kan allerede se bildet som blir vist (**Figur 7-15**).

Til slutt tar **`sense.set_pixels(creeper_pixels)`** denne variabelen og bruker funksjonen **`sense.set_pixels()`** til å tegne hele matrisen på én gang. Det er mye enklere enn å prøve å tegne en piksel om gangen.



Figur 7-15: Vise et bilde på matrisen

Du kan også rotere og vende bilder, enten for å vise bilder den rette veien når Sense HAT er snudd, eller for å lage enkle animasjoner fra et enkelt, asymmetrisk bilde.

Start med å redigere variabelen **`creeper_pixels`** for å lukke det ene øyet ved å erstatte de fire «b»-pikslene – først de to første på tredje linje og deretter de to første på fjerde linje, med en «g»:

```
creeper_pixels = [
    g, g, g, g, g, g, g, g,
    g, g, g, g, g, g, g, g,
    g, g, g, g, g, b, b, g,
    g, g, g, g, g, b, b, g,
    g, g, g, b, b, g, g, g,
    g, g, b, b, b, b, g, g,
    g, g, b, b, b, b, g, g,
    g, g, b, g, g, b, g, g
]
```

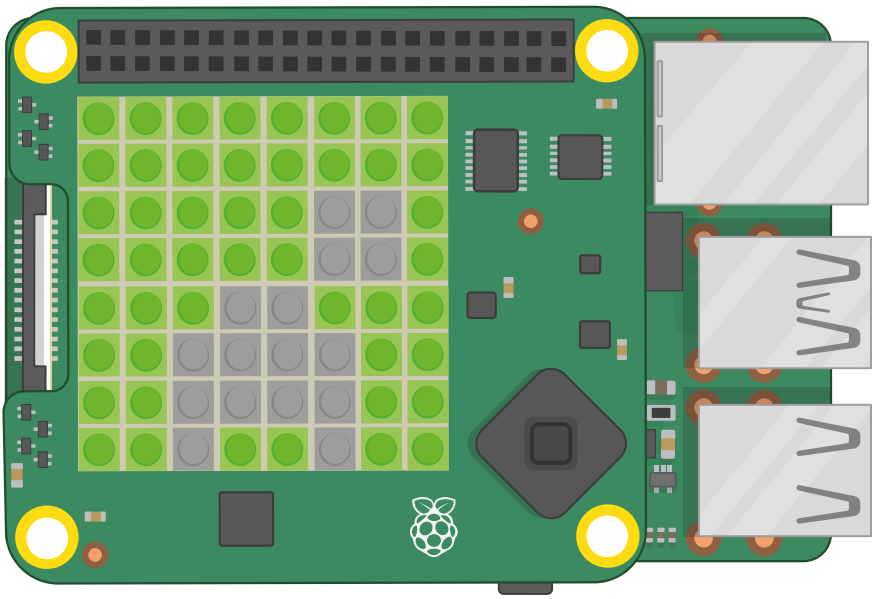
Klikk på Run, så ser du at krypets venstre øye lukkes (**Figur 7-16** på neste side). Hvis du vil lage en animasjon, går du til toppen av programmet og setter inn linjen:

```
from time import sleep
```

Deretter går du nederst og skriver:

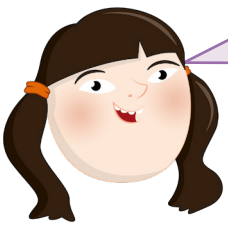
```
while True:
    sleep(1)
    sense.flip_h()
```

Klikk på Run, og du vil se at creeper-pikselen dyret lukker og åpner øynene, ett om gangen.



Figur 7-16: Vise en enkel animasjon med to bilder

Funksjonen **`flip_h()`** vender et bilde rundt den vannrette aksen, på tvers. Hvis du ønsker å vende et bilde på rundt den loddrette aksen, erstatter du **`sense.flip_h()`** med **`sense.flip_v()`**. Du kan også rotere et bilde 0, 90, 180 eller 270 grader ved hjelp av **`sense.set_rotation(90)`**. Endre tallet i forhold til hvor mange grader du ønsker å rotere bildet. Prøv å bruke dette for å få dyret til å snurre rundt i stedet for å blunke med øynene.

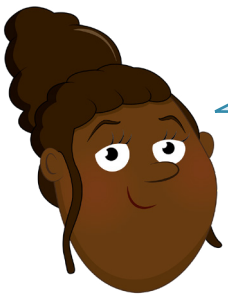


UTFORDRING: NYE DESIGN

Kan du designe flere bilder og animasjoner? Prøv å få tak i graf- eller rutepapir, og bruk det til å skissere bildet for hånd først. Da blir det enklere å skrive variabelen. Kan du tegne et bilde og få fargene til å endre seg? Tips: Du kan endre variablene etter at du har brukt dem én gang.

Måle verdenen rundt deg

Sense HATs virkelige kraft ligger i de forskjellige sensorene den har. Du kan bruke sensorene til å foreta avlesninger av alt fra temperatur til akselerasjon og bruke dem i programmene etter behov.



EMULERE SENSORENE

Hvis du bruker Sense HAT-emulatoren, må du aktivere simulering av treghets- og miljøsensor. I Emulator klikker du på Edit, deretter Preferences og merker av for disse alternativene. I den samme menyen velger du «180 °. 360 ° | 0 °. 180 °» under «Orientation Scale» for å sikre at tallene i Emulator samsvarer med tallene rapportert av Scratch og Python. Deretter klikker du på Close.

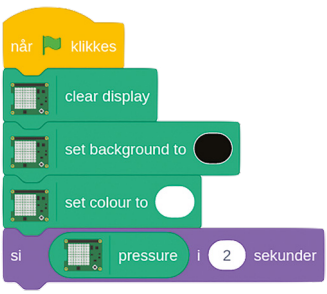
Måling av omgivelsene

Den barometriske trykksensoren (barometeret), fuktighetssensoren og temperatursensoren er alle omgivelsessensorer. De foretar målinger fra omgivelsene rundt Sense HAT.

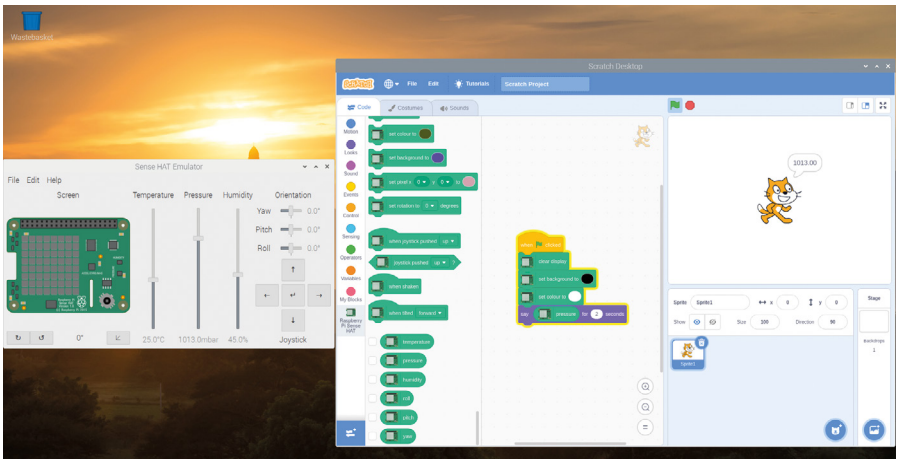
Måling av omgivelsene i Scratch

Start et nytt program i Scratch (lagre det gamle programmet hvis du vil,), og legg til tilleggsfunksjonen Raspberry Pi Sense HAT hvis den ikke allerede er lastet inn. Dra hendelsesklossen når klikkes til kodeområdet. Deretter drar du klossen clear display og set background to svart under der igjen. Deretter legger du til klossen set colour to hvit – bruk glidebryterne Lysstyrke og Metning for å velge riktig farge. Det er alltid lurt å gjøre dette på starten av programmene for å unngå at Sense HAT viser noe fra et gammelt program, samtidig som du vet hvilke farger som brukes.

Dra utseendeklossen si Hei! i 2 sekunder rett under de eksisterende klossene. Foreta en avlesning fra trykksensoren, finn klossen pressure i Raspberry Pi Sense HAT-kategorien og dra den over ordet «Hei!» i klossen si Hei! i 2 sekunder.



Klikk på det grønne flagget, så vil Scratch-katten gi deg gjeldende avlesning fra trykksensoren i *millibar*. Etter to sekunder forsvinner meldingen. Forsøk å blåse på Sense HAT (eller å flytte glidebryteren Trykk opp i emulatoren). Deretter klikker du på det grønne flagget for å kjøre programmet på nytt. Da skal du se en høyere avlesing (**Figur 7-17** på neste side).

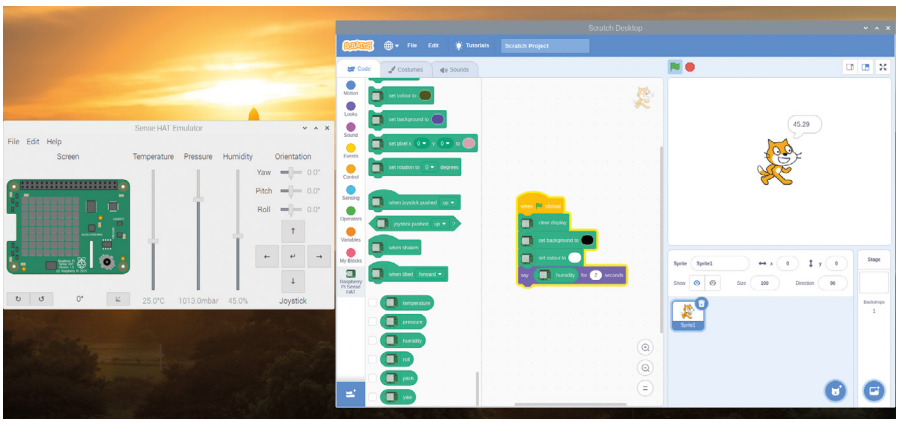


▲ **Figur 7-17:** Vise trykksensoravlesningen

ENDRE VERDIER

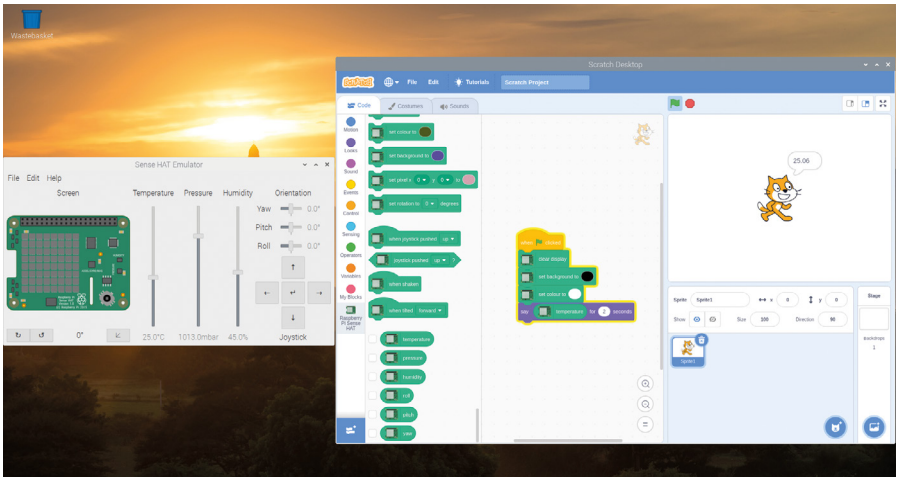
Hvis du bruker Sense HAT-emulatoren, kan du endre verdiene som rapporteres av hver av de emulerte sensorene, ved hjelp av glidebryterne og knappene. Prøv å skyve trykksensorinnstillingen ned mot bunnen, og klikk deretter på det grønne flagget igjen.

Bytt til fuktighetssensoren ved å slette klossen pressure og erstatte den med humidity. Kjør programmet på nytt, så ser du relativ fuktighet i rommet. Du kan prøve å kjøre det på nytt mens du blåser på Sense HAT (eller flytter glidebryteren Fuktighet på emulatoren opp) for å endre avlesningen (**Figur 7-18**) – pusten din er overraskende fuktig.

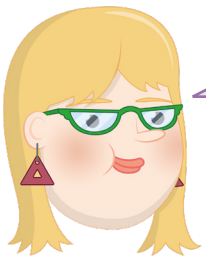


▲ **Figur 7-18:** Vise avlesningen fra fuktighetssensoren

For temperatursensoren sletter du ganske enkelt klossen `humidity` og erstatter den med `temperature`. Deretter kjører du programmet på nytt. Du vil se en temperatur i grader celsius (**Figur 7-19**). Dette er kanskje ikke den eksakte temperaturen i rommet: Raspberry Pi genererer varme hele tiden mens den kjører. Dermed blir også Sense HAT og sensorene densvarmere.



▲ **Figur 7-19:** Vise avlesningen for temperaturføleren



UTFORDRING: RULLE OG GJENTAING OG SLØYFERLØKKER

Kan du endre programmet slik at det tas en avlesning fra hver av sensorene etter tur, og deretter la dem rulle over LED-matrisen i stedet for å skrive dem ut til sceneområdet? Kan du løkkefå programmet til å kjøre i en løkke, slik at den kontinuerlig skriver ut de gjeldende omgivelsesforholdene?

Måling av omgivelsene i Python

Når du skal begynne å ta målinger fra sensorer, oppretter du et nytt program i Thonny og lagrer det som **Sense HAT-sensorer**. Skriv følgende i skriptområdet, som alltid når du bruker Sense HAT. Husk å bruke `sense_emu` hvis du bruker emulatoren:

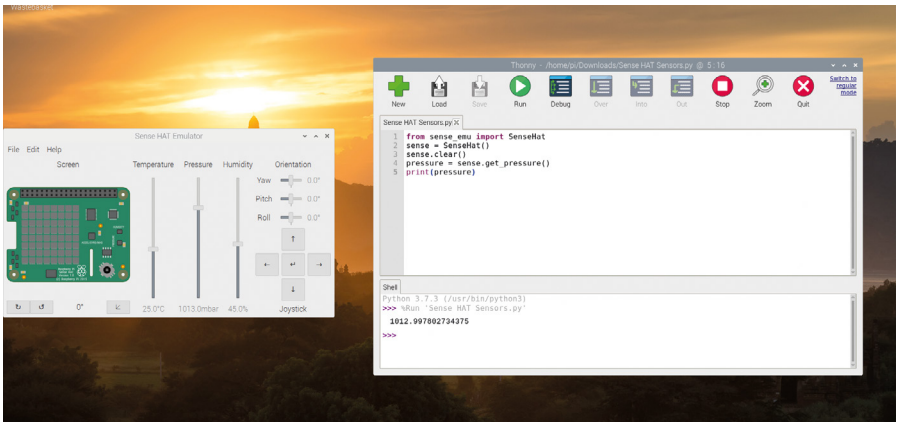
```
from sense_hat import SenseHat
sense = SenseHat()
sense.clear()
```

Det er alltid lurt å ta med `sense.clear()` på starten av programmene, i tilfelle Sense HAT-skjermen fremdeles viser noe fra det siste programmet den kjørte.

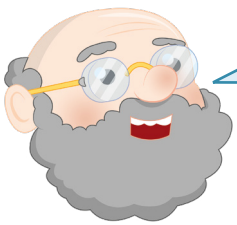
Hvis du vil ta en avlesning fra trykksensoren, skriver du:

```
pressure = sense.get_pressure()
print(pressure)
```

Klikk på Run, så ser du et tall i Python-skallet nederst i Thonny-vinduet. Dette er avlesningen for lufttrykk målt av barometeres, i *millibar* (**Figur 7-20**). Forsøk å blåse på Sense HAT (eller skyve glidebryteren Pressure opp i emulatoren) og klikk på Run -ikonet igjen. Nå skal tallet være høyere.



▲ **Figur 7-20:** Skrive ut en trykkavlesning fra Sense HAT



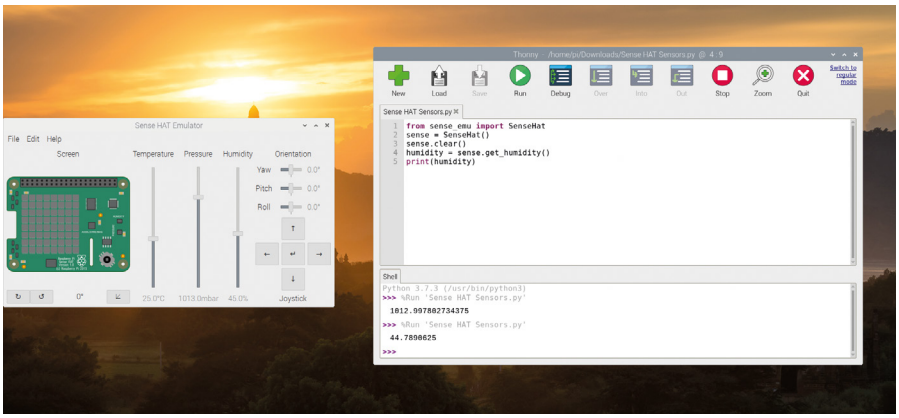
ENDRE VERDIER

Hvis du bruker Sense HAT-emulatoren, kan du endre verdiene som rapporteres av hver av de emulerte sensorene, ved hjelp av glidebryterne og knappene. Prøv å skyve trykksensorinnstillingen ned mot bunnen, og klikk deretter på Run igjen.

Hvis du vil bytte til fuktighetssensoren, sletter du de to siste kodelinjene og erstatter dem med:

```
humidity = sense.get_humidity()
print(humidity)
```

Klikk på Run, så ser du et tall i Python-skallet. Nå vises den relative fuktigheten i rommet som en prosentdel. Blås på Sense HAT igjen (eller flytt glidebryteren Fuktighet på emulatoren opp), så ser du at den går opp når du kjører programmet på nytt (**Figur 7-21**) – pusten din er overraskende fuktig.

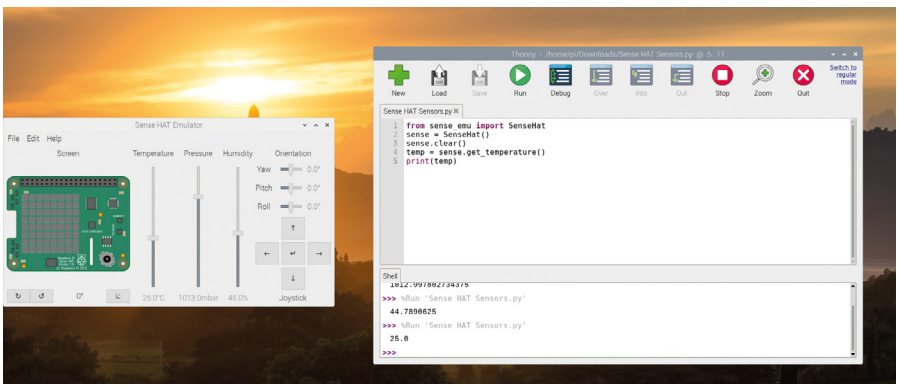


Figur 7-21: Vise avlesningen for fuktighetssensoren

For temperaturføleren sletter du de to siste linjene av programmet og erstatter dem med:

```
temp = sense.get_temperature()
print(temp)
```

Klikk på Run igjen for å se en temperatur i grader celsius (**Figur 7-22**). Dette er kanskje ikke den eksakte temperaturen i rommet: Raspberry Pi genererer varme hele tiden mens den kjører. Dermed blir også Sense HAT og sensorene densvarmere.

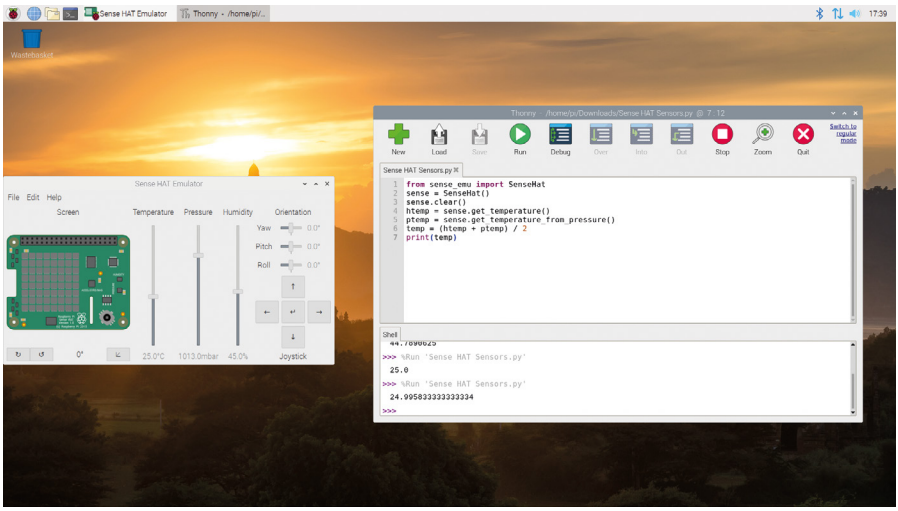


Figur 7-22: Vise gjeldende temperaturavlesning

Normalt rapporterer Sense HAT temperaturen basert på en avlesning fra temperatursensoren som er innebygd i fuktighetssensoren. Hvis du ønsker å bruke avlesningen fra trykksensoren i stedet, bruker du **`sense.get_temperature_from_pressure ()`**. Du kan også kombinere de to målingene for å få et gjennomsnitt, noe som kan være mer nøyaktig enn å bruke bare én av sensorene. Slett de to siste linjene av programmet og skriv:

```
htemp = sense.get_temperature()
ptemp = sense.get_temperature_from_pressure()
temp = (htemp + ptemp) / 2
print(temp)
```

Klikk på Run for å se et tall på Python-konsollen (**Figur 7-23**). Denne gangen er den basert på målinger fra begge sensorene. Du legger dem sammen og deler tallet på to – antall målinger – for å finne et gjennomsnitt av begge målingene. Hvis du bruker emulatoren, vil alle tre metodene – fuktighet, trykk og gjennomsnitt – vise det samme tallet.



▲ **Figur 7-23:** En temperatur basert på avlesningene fra begge sensorer

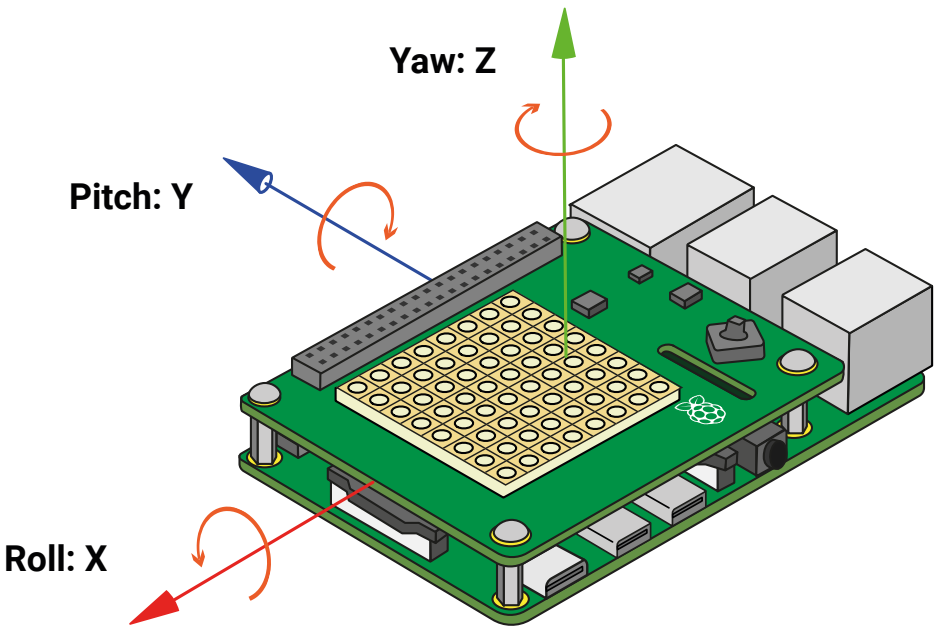
UTFORDRING: RULLING OG LØKKER

Kan du endre programmet slik at det tas en avlesning fra hver av sensorene etter tur, og deretter la dem rulle over LED-matrisen i stedet for å skrive dem ut til skallet? Kan du løkkefå programmet til å kjøre i en løkke, slik at den kontinuerlig skriver ut de gjeldende omgivelsesforholdene?

Måling av treghet

Den gyroskopiske sensoren, akselerometeret og magnetometerer danner til sammen det som kalles en *treghetsmåleenhet (IMU)*. Disse sensorene tar teknisk sett målinger fra omgivelsene, akkurat som omgivelsessensorene. Magnetometeret måler for eksempel magnetisk feltstyrke. De brukes imidlertid vanligvis til å måle data om bevegelsen til selve Sense HAT. IMU er summen av flere sensorer. Med noen programmeringsspråk kan du ta avlesninger fra hver sensor for seg, mens andre bare gir deg en kombinert avlesning.

For å forstå hvordan IMU-en fungerer må du først forstå hvordan ting beveger seg. Sense HAT og Raspberry Pi, som den er festet til, kan bevege seg langs tre romakser: fra side til side på X-aksen, forover og bakover på Y-aksen og opp og ned på Z-aksen **(Figur 7-24)**. Den kan også rotere langs disse tre aksene, men navnene endres. Rotering rundt X-aksen kalles *roll*, rotering rundt Y-aksen kalles *pitch* og rotering rundt Z-aksen kalles *yaw*. Når du roterer Sense HAT rundt kortaksen, justerer du pitch. Rotering rundt den lange aksen kalles roll, og hvis du snurrer den rundt mens den holdes flatt mot bordet, justerer du yaw. tSe for deg et fly: Når det tar av, øker det pitch-verdien for å stige. Når det ruller sideveis i en oppvisning, snurrer det bokstavelig talt langs roll-aksen. Når det bruker sideroret til å svinge som i en bil, kalles det yaw.



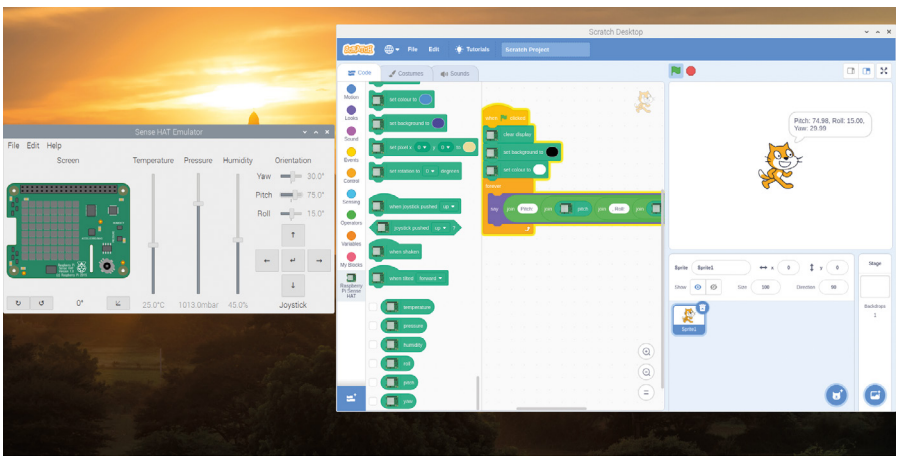
▲ **Figur 7-24:** Romaksene til Sense HATs IMU

Måling av treghet i Scratch

Start et nytt program i Scratch og last inn tilleggsfunksjonen Raspberry Pi Sense HAT, hvis du ikke allerede har gjort det. Start programmet på samme måte som før: Dra hendelsesklossen når klikkes til kodeområdet. Deretter drar du klossen clear display under den og så redigerer du klossene set background to svart og set colour to hvit.

Deretter drar du en gjenta for alltid-kloss under alle de eksisterende blokkene og fyller den med klossen si Hei!. Hvis du vil vise en avlesning for hver av de tre aksene til IMU-enheten, pitch, roll og yaw, må du legge til operatorklossene sett sammen pluss de tilsvarende Sense HAT-klossene. Husk å ta med mellomrom og komma, slik at utdataene blir enkle å lese.

Klikk på det grønne flagget for å kjøre programmet, og forsøk å flytte litt på Sense HAT og Raspberry Pi. Vær forsiktig så du ikke river løs noen ledninger. Når du beveger Sense HAT gjennom de tre aksene, ser du verdiene for henholdsvis «pitch», «roll» og «yaw», dvs. rotasjon om hver av hovedaksene (**Figur 7-25**).



▲ **Figur 7-25:** Vise verdier for pitch, roll og yaw

Måling av treghet i Python

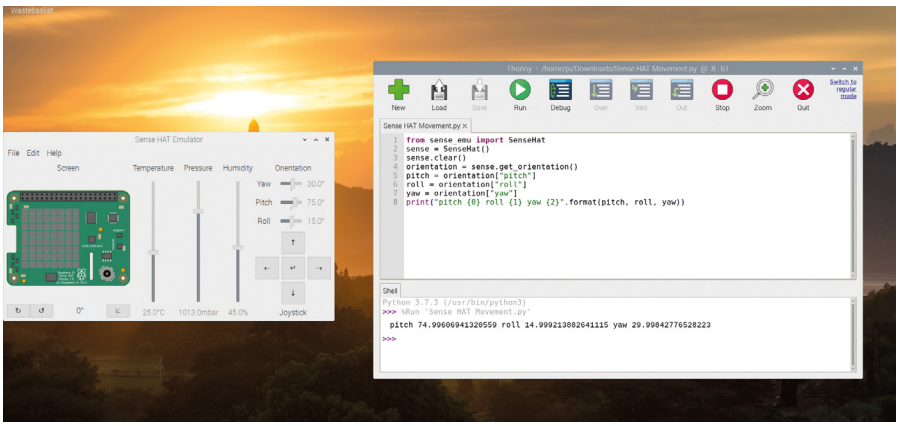
Start et nytt program i Thonny og lagre det som **Sense HAT-bevegelse**. Fyll ut de vanlige startlinjene. Husk å bruke `sense_emu` hvis du bruker Sense HAT-emulatoren:

```
from sense_hat import SenseHat
sense = SenseHat()
sense.clear()
```

Hvis du vil bruke informasjon fra IMU til å beregne den aktuelle retningen til Sense HAT på de tre aksene, skriver du inn følgende:

```
orientation = sense.get_orientation()
pitch = orientation["pitch"]
roll = orientation["roll"]
yaw = orientation["yaw"]
print("pitch {0} roll {1} yaw {2}".format(pitch, roll, yaw))
```

Klikk på Run for å vise avlesninger for Sense HATs retning fordelt over de tre aksene (**Figur 7-26**). Prøv å rotere Sense HAT og klikk på Run igjen. Da skal du se at tallene blir endret for å gjenspeile den nye retningen.



▲ **Figur 7-26:** Vise verdier for Sense HATs pitch, roll og yaw

Men IMU-enheten kan gjøre mer enn å måle retningen. Den kan også registrere bevegelse. For å få nøyaktige målinger for bevegelse må IMU avleses ofte i en løkke. I motsetning til når du jobber med retninger vil ikke én enkelt måling gi nyttig informasjon når det gjelder å registrere bevegelse. Slett alt etter `sense.clear()` og skriv følgende kode:

```
while True:
    acceleration = sense.get_accelerometer_raw()
    x = acceleration["x"]
    y = acceleration["y"]
    z = acceleration["z"]
```

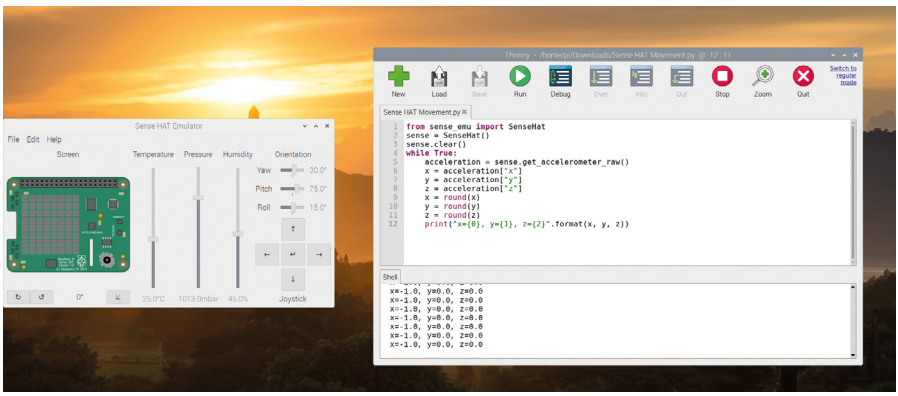
Du har nå variabler som inneholder gjeldende akselerometeravlesninger for de tre romaksene: X – venstre og høyre, Y – fremover og bakover og Z – opp eller ned. Tallene fra akselerometersensoren kan være vanskelige å lese. Derfor skriver du følgende for å gjøre dem enklere å forstå ved å avrunde dem til nærmeste hele tall:

```
    x = round(x)
    y = round(y)
    z = round(z)
```

Til slutt skriver du ut de tre verdiene ved å skrive denne linjen:

```
    print("x={0}, y={1}, z={2}".format(x, y, z))
```

Klikk på Run for å se verdier fra akselerometeret trykt i Pythons skallområde (**Figur 7-27**). I motsetning til det forrige programmet skrives disse kontinuerlig. Du kan stoppe skrivingen ved å klikke på den røde stoppknappen for å stoppe programmet.



▲ **Figur 7-27:** Akselerometeravlesninger avrundet til nærmeste hele tall

Du har kanskje lagt merke til at akselerometeret forteller deg at en av aksene (Z-aksen) har en akselerasjonsverdi på 1,0 gravitasjoner (1 G) når Raspberry Pi ligger flatt på bordet, men Sense HAT beveger seg ikke. Det er fordi den oppdager jordens tyngdekraft, dvs. kraften som trekker Sense HAT ned mot midten av jorden. Det er det som gjør at ting som faller fra bordet, havner på gulvet.

Når programmet kjører, kan du prøve å løfte Sense HAT og Raspberry Pi forsiktig opp og rotere dem rundt. Pass på at du ikke river løs noen av ledningene. Når Raspberry Pis nettverk og USB-portene peker mot gulvet, vil du se at verdiene endres slik at Z-aksen viser 0 G og X-aksen viser 1 G. Snu den igjen slik at HDMI- og strømportene peker mot gulvet. Nå viser Y-aksen 1 G. Hvis du gjør det motsatte og HDMI-porten peker mot taket, ser du –1 G på Y-aksen i stedet.

Ved å bruke kunnskapen om romaksene og at jordens tyngdekraft er omtrent 1 G, kan du bruke målinger fra akselerometeret til å finne ut hvilken vei som er ned – og hvilken vei som er opp. Du kan også bruke den til å registrere bevegelse. Rist forsiktig på Sense HAT og Raspberry Pi, og følg med på tallene mens du gjør det. Jo hardere du rister, desto større er akselerasjonen.

Når du bruker **`sense.get_accelerometer_raw()`**, ber du Sense HAT om å slå av de to andre sensorene i IMU – gyroskopisk sensor og magnetometer – og bare returnere data fra akselerometeret. Du kan selvsagt gjøre det samme med de andre sensorene også.

Finn linjen **`acceleration = sense.get_accelerometer_raw()`** og endre den til:

```
orientation = sense.get_gyroscope_raw()
```

Endre ordet **`acceleration`** på alle tre linjer under den til **`orientation`**. Klikk på Run, så ser du retningen til Sense HAT for alle de tre aksene, avrundet til nærmeste hele tall. I motsetning til siste gang du sjekket retningen, kommer dataene denne gangen bare fra gyroskopet uten å bruke akselerometeret eller magnetometeret. Dette kan være nyttig hvis du for eksempel vil vite retningen til en bevegelig Sense HAT på baksiden av en robot, uten at bevegelsen skaper forvirring, eller hvis du bruker Sense HAT i nærheten av et sterkt magnetfelt

Stopp programmet ved å klikke på den røde stoppknappen. Når du skal bruke magnetometeret, sletter du alt fra programmet ditt, bortsett fra de første fire linjene. Deretter skriver du følgende under linjen **`while True`**:

```
north = sense.get_compass()
print(north)
```

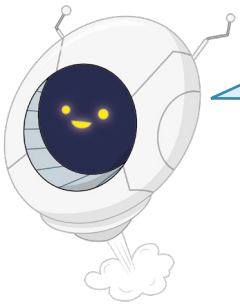
Kjør programmet, så vil du se at retningen for magnetisk nord skrives ut gjentatte ganger til Python-skallområdet. Drei forsiktig på Sense HAT, så ser du at overskriften endres etter som Sense HATs retning i forhold til nord endres. Du har bygget et kompass. Hvis du har en magnet – en kjøleskapsmagnet er ok – beveger du den rundt Sense HAT for å se hva som skjer med magnetometerets avlesninger.

UTFORDRING: AUTO-ROTERING

Ved å bruke det du har lært om LED-matrisen og sensorene til måleenheten for treghet, kan du skrive et program som roterer et bilde avhengig av plasseringen til Sense HAT?

Joystick-kontrollenStyring av joysticken

Sense HAT-joysticken, som du finner i nederste høyre hjørne, er riktignok liten, men den er overraskende kraftig. I tillegg til å kunne gjenkjenne inndata i fire retninger – opp, ned, venstre og høyre – har den også en femte inndatakilde, som nås ved å trykke den ned ovenfra som en trykkbryter.



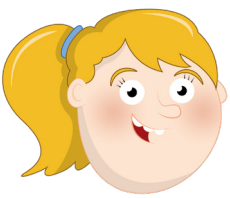
ADVARSEL!

Sense HAT-joysticken må bare brukes hvis du har montert avstandsstykkene som beskrevet på begynnelsen av dette kapitlet. Uten avstandsstykkene kan trykk på joysticken bøye Sense HAT-kortet og skade både Sense HAT og Raspberry Pis GPIO-pinnerekke.

Styre joysticken i Scratch

Start et nytt program i Scratch med tilleggsfunksjonen Raspberry Pi Sense HAT innlastet. Som tidligere drar du hendelsesklossen når klikkes til skriptområdet. Deretter drar du klossen clear display under den før du drar og redigerer klossen set background to svart og set colour to hvit.

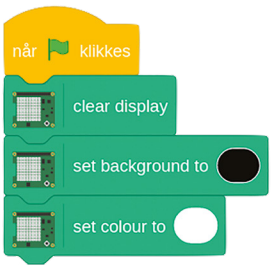
I Scratch tilordnes joystickenSense HAT-joysticken til piltastene på tastaturet. Når du skyver joysticken opp, fungerer det som å trykke på pil opp-tasten. Når du skyver den ned, fungerer det som å trykke på pil ned-tasten. Når du skyver den til venstre, fungerer det som å trykke på pil venstre-tasten. Når du skyver den til høyre, fungerer det som å trykke på pil høyre-tasten. jNår du skyver joysticken inn som en trykkbryter, fungerer det som å trykke på **ENTER**-tasten.

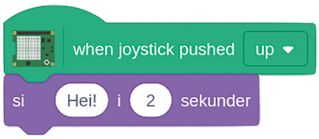


ADVARSEL!

Styring av joysticken er bare tilgjengelig på den fysiske Sense HAT. Når du bruker Sense HAT-emulatoren, må du i stedet bruke de tilsvarende tastene på tastaturet til å simulere trykk på joysticken.

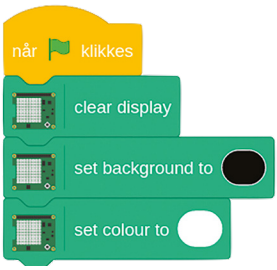
Dra klossen **when joystick pushed up** til kodeområdet. Deretter, for å gi den noe å gjøre, drar du klossen **si Hei! i 2 sekunder** under den.

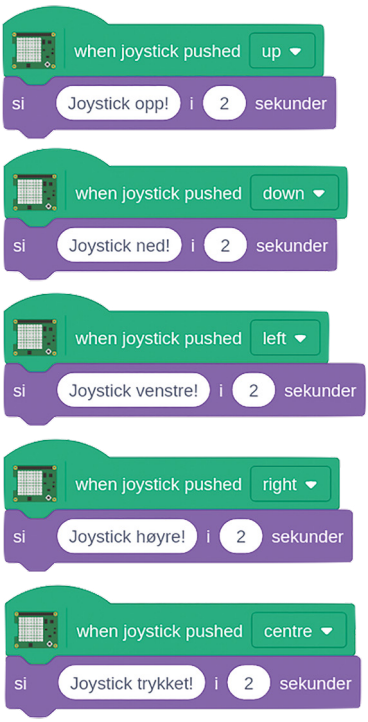




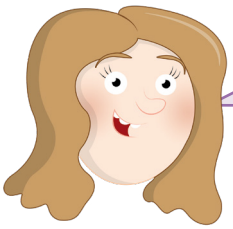
Skyv joysticken opp, så vil du se at Scratch-katten sier et muntert «Hei!»

Deretter endrer du klossen **si Hei! i 2 sekunder** til klossen **si Joystick opp! i 2 sekunder** og fortsetter å legge til hendelses- og utseendeklosser til du har noe for hver av de fem metodene du kan bruke for å trykke på joysticken.





Prøv å skyve joysticken i forskjellige retninger for å se at meldingene dine vises!



SISTE UTFORDRING

Kan du bruke Sense HATs joystick til å kontrollere en Scratch-figur i sceneområdet? Kan du gjøre det slik at hvis figuren fanger opp en annen figur (som representerer en gjenstand), viser Sense HATs LED-lamper en artig melding?

Styring av joysticken i Python

Start et nytt program i Thonny, og lagre det som Sense HAT-joystick. Begynn med de vanlige tre linjene som setter opp Sense HAT og sletter LED-matrisen:

```
from sense_hat import SenseHat
sense = SenseHat()
sense.clear()
```

Deretter setter du opp en uendelig løkke:

```
while True:
```

Be deretter Python om å se etter inndata fra Sense HAT-joysticken med følgende linje, som Thonny automatisk rykker inn for deg:

```
    for event in sense.stick.get_events():
```

Til slutt legger du til følgende linje, som Thonny igjen vil rykke inn for deg, for å faktisk gjøre noe når det registreres trykk på joysticken:

```
        print(event.direction, event.action)
```

Klikk på Run, og prøv å skyve joysticken i forskjellige retninger. Du vil se at retningen du har valgt, skrives ut i Pythons skallområde: opp, ned, venstre, høyre og i midten for når du har trykket joysticken ned som en trykkbryter.

Du vil også se at du får to hendelser hver gang du trykker én gang på joysticken: én hendelse, **`pressed`**, for første bevegelse i en retning; den andre hendelsen, **`released`**, for når joysticken går tilbake til sentrum. Du kan bruke dette i programmene dine. Tenk på en figur i et spill, som begynner å bevege seg når du skyver joysticken i en retning, og stopper når den slippes.

Du kan også bruke joysticken til å utløse funksjoner, i stedet for å være begrenset til å bruke en løkke. Slett alt nedenfor **`sense.clear()`**, og skriv følgende:

```
def red():
    sense.clear(255, 0, 0)

def blue():
    sense.clear(0, 0, 255)

def green():
    sense.clear(0, 255, 0)

def yellow():
    sense.clear(255, 255, 0)
```

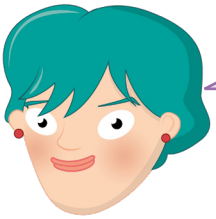
Disse funksjonene endrer hele Sense HAT LED-matrisen til én farge: rød, blå, grønn eller gul. Dette gjør det mye enklere å se om programmet fungerer. Hvis du vil utløse dem, må du fortelle Python hvilken funksjon som følger hvilke inndata fra joysticken. Skriv følgende linjer:

```
sense.stick.direction_up = red
sense.stick.direction_down = blue
sense.stick.direction_left = green
sense.stick.direction_right = yellow
sense.stick.direction_middle = sense.clear
```

Til slutt trenger programmet en uendelig løkke – kalt en *hovedløkke* – for å fortsette å kjøre og se etter inndata fra joysticken, i stedet for bare å kjøre gjennom koden du har skrevet, én gang og deretter avslutte programmet. Skriv følgende to linjer:

```
while True:
    pass
```

Klikk på Run, og prøv å bevege joysticken. Da vil du se at LED-lampene lyser i strålende farger. Hvis du vil slå av LED-lampene, skyver du inn joysticken som en trykkbryter. Den midtre retningen, **`middle`**, angis for å bruke funksjonen **`sense.clear()`** til å slå av alle lampene. Gratulerer, nå vet du hvordan du bruker inndata fra joysticken.



SISTE UTFORDRING

Kan du bruke det du har lært til å tegne et bilde til skjermen, og deretter rotere det i den retningen joysticken skyves? Kan du få den midterste inngangen til å veksle mellom flere enn ett bilde?

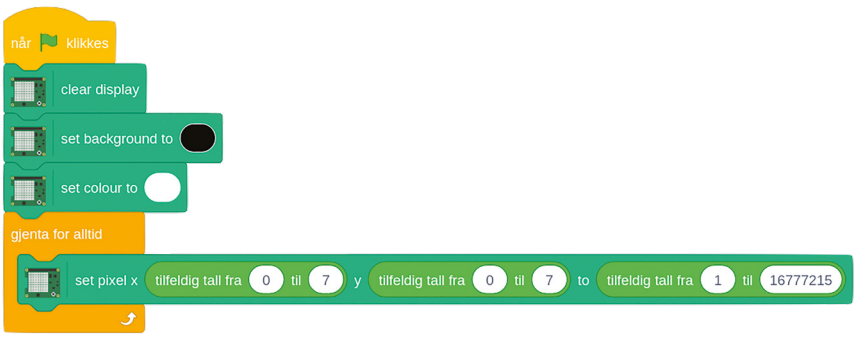
Scratch-prosjekt: Sense HAT-stjerneskudd

Nå som du vet hvordan Sense HAT fungerer, er det på tide å bruke alt du har lært, for å bygge et varmefølsomt stjerneskudd – en enhet som er mest fornøyd når den er kald og som gradvis slukker jo varmere den blir.

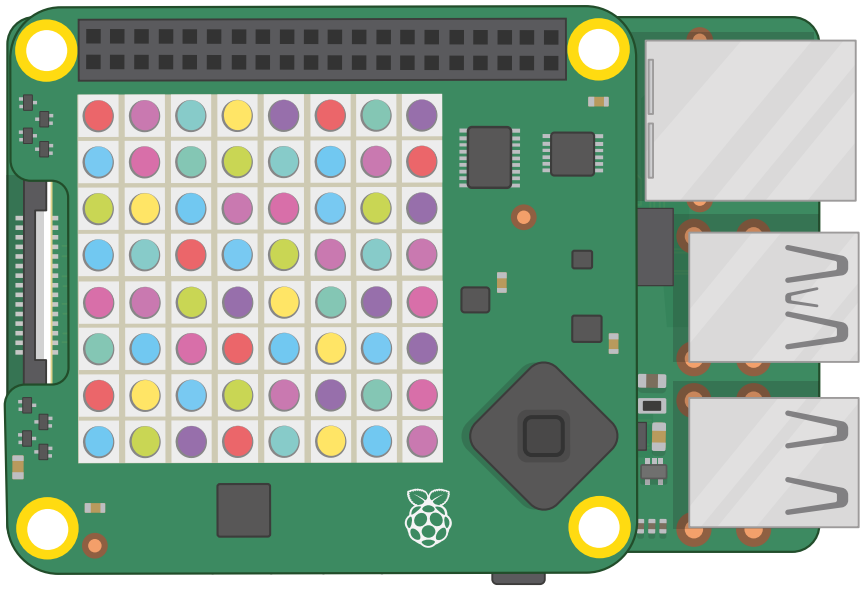
Start et nytt program i Scratch og last inn tilleggsfunksjonen Raspberry Pi Sense HAT, hvis du ikke allerede har gjort det. Som alltid starter du med fire klosser: når klikkes , clear display , set background to svart og set colour to hvit . Husk at du må endre fargene fra standardinnstillingene.

Start med å lage et enkelt, men kunstnerisk, stjerneskudd. Dra klossen gjenta for alltid til kodeområdet, og fyll den med klossen set pixel x 0 y 0 to farge . I stedet for å bruke faste tall fyller du ut hver av x-, y- og fargeseksjonene i den klossen operatorklossen tilfeldig tall fra 1 til 10 .

Verdiene 1 til 10 er ikke så nyttige her, så du må redigere litt. De to første tallene i klossen set pixel er X- og Y-koordinatene til pikselen på LED-matrisen, så de må være tall mellom 0 og 7. Derfor må du endre de to første klossene til tilfeldig tall fra 0 til 7 . Den neste delen er fargen pikselen skal settes til. Når du bruker fargevelgeren, vises valgt farge direkte i skriptområdet. Internt er fargene imidlertid representert med et tall, og du kan bruke tallet direkte. Endre den siste tilfeldige klossen til tilfeldig tall fra 0 til 16777215 .

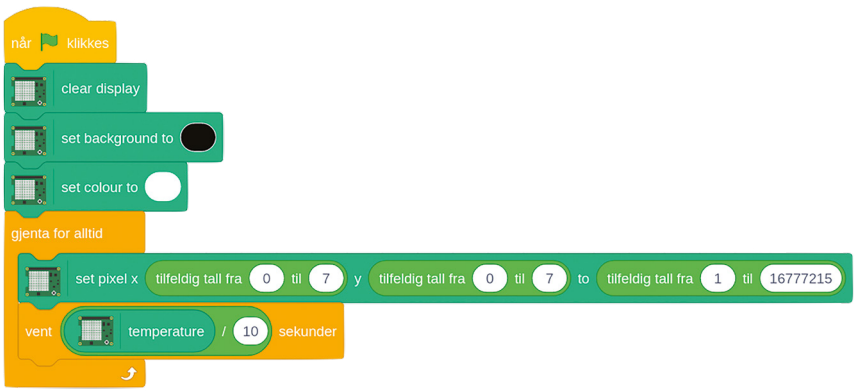


Klikk på det grønne flagget, og du ser at LED-lampene på Sense HAT begynner å tennes i tilfeldige farger (**Figur 7-28**). Gratulerer, du har laget et elektronisk stjerneskudd.



Figur 7-28: Tenne pikslene i tilfeldige farger

Stjerneskuddet er ikke særlig interaktivt. Du kan endre dette ved å dra klossen **vent 1 sekunder** under klossen **set pixel**, men inne i klossen **gjenta for alltid**. Dra operatorklossen **● / ●** over 1, og skriv deretter 10 i det andre feltet. Til slutt drar du klossen **temperature** over det første feltet operatorklossen for delignmed per-tegnet.



Klikk på det grønne flagget. Legg merke til følgende: Med mindre du bor et sted der det er veldig kaldt, vil stjerneskuddet være betydelig tregere enn før. Det er fordi du har laget en temperaturavhengig forsinkelse. Programmet venter nå *gjeldende temperatur delt på 10* antall sekunder før hver løkke. Hvis temperaturen i rommet er 20 °C, venter programmet i 2 sekunder før løkkeden kjører løkken. Hvis temperaturen er 10 °C, venter det i 1 sekund, og hvis det er under 10 °C, venter det i mindre enn 1 sekund.

Hvis Sense HAT registrerer en negativ temperatur – under 0 °C, vannets frysepunkt – vil den prøve å vente mindre enn 0 sekunder. Fordi det ikke er mulig – i hvert fall ikke uten å reise i tid – får du samme virkning som om den ventet i 0 sekunder. Gratulerer, nå vet du hvordan du integrerer de ulike funksjonene i Sense HAT i dine egne programmer!

Python-prosjekt: Sense HAT Tricorder

Nå som du vet hvordan Sense HAT fungerer, er det på tide å bruke det du har lært for å bygge en tricorder – en enhet som er godt kjent blant fansen av en viss Si-Fi-serie og som brukes til å måle ting med innebygde sensorer.

Start et nytt prosjekt i Thonny, og lagre det som **Tricorder**. Deretter begynner du med de vanlige linjene som trengs for å lage et Sense HAT-program:

```
from sense_hat import SenseHat
sense = SenseHat()
sense.clear()
```

Deretter må du definere funksjoner for hver av Sense HATs ulike sensorer. Start med måleenheten for treghet ved å skrive:

```
def orientation():
    orientation = sense.get_orientation()
    pitch = orientation["pitch"]
    roll = orientation["roll"]
    yaw = orientation["yaw"]
```

Siden resultatet fra sensoren skal rulle over LED-lampene, er det lurt å runde dem av slik at du ikke må vente på dusinvis av desimaler. I stedet for hele tall runder du dem av til en desimal ved å skrive følgende:

```
    pitch = round(pitch, 1)
    roll = round(roll, 1)
    yaw = round(yaw, 1)
```

Til slutt ber du Python om å rulle resultatene til LED-lampene, slik at tricorderen fungerer som en håndholdt enhet uten å måtte være koblet til en skjerm eller tv:

```
    sense.show_message("Pitch {0}, Roll {1}, Yaw {2}".
format(pitch, roll, yaw))
```

Nå som du har en fullstendig funksjon for avlesning og visning av retningen fra IMU, lager du lignende funksjoner for hver av de andre sensorene. Begynn med temperaturføleren:

```
def temperature():
    temp = sense.get_temperature()
    temp = round(temp, 1)
    sense.show_message("Temperatur: %s grader celsius" % temp)
```

Se nøye på linjen som skriver ut resultatet til LED-lampene: **%s** kalles en plassholder og blir erstattet med innholdet i variabelen **temp**. På den måten kan du formatere utdataene med en etikett, «Temperatur:», og en måleenhet, «grader celsius», for å gjøre programmet mer brukervennlig.

Deretter definerer du en funksjon for fuktighetssensoren:

```
def humidity():
    humidity = sense.get_humidity()
    humidity = round(humidity, 1)
    sense.show_message("Fuktighet: %s prosent" % humidity)
```

Og deretter trykksensoren:

```
def pressure():
    pressure = sense.get_pressure()
    pressure = round(pressure, 1)
    sense.show_message("Trykk: %s millibar" % pressure)
```

Og til slutt kompassavlesningen fra magnetometeret:

```
def compass():
    for i in range(0, 10):
        north = sense.get_compass()
    north = round(north, 1)
    sense.show_message("Nord: %s grader" % north)
```

Den korte **for**-løkken i denne funksjonen krever ti avlesninger fra magnetometeret for å sikre at den har nok data til å gi deg et nøyaktig resultat. Hvis du opplever at den rapporterte verdien stadig endrer seg, kan du prøve å utvide den til 20, 30 eller til og med 100 løkker for få enda bedre nøyaktighet.

Programmet har nå fem funksjoner, som hver tar en avlesning fra en av Sense HATs sensorer og lar dem rulle over LED-lampene. Men det trenger en metode for å velge hvilken sensor som skal brukes, og da er joysticken perfekt.

Skriv følgende:

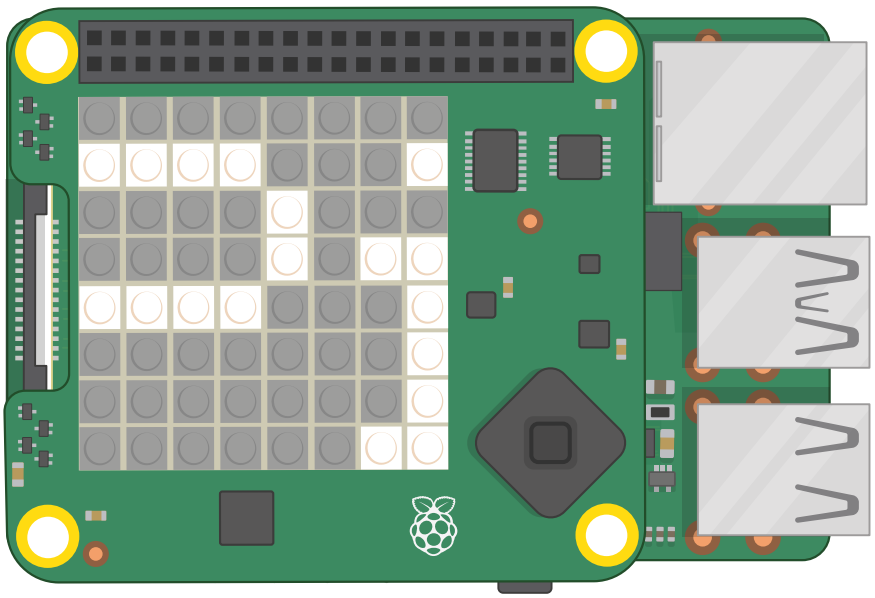
```
sense.stick.direction_up = orientation
sense.stick.direction_right = temperature
sense.stick.direction_down = compass
sense.stick.direction_left = humidity
sense.stick.direction_middle = pressure
```

Disse linjene tilordner en sensor til hver av de fem mulige retningene på joysticken: opp leser fra retningssensoren, ned leser fra magnetometeret, venstre leser fra fuktighetssensoren, høyre fra temperaturføleren og et midttrykk på joysticken leser fra trykksensoren.

Til slutt trenger du en hovedløkke, slik at programmet fortsetter å se etter trykk på joysticken og ikke bare avslutter umiddelbart. Gå helt nederst i programmet og skriv inn følgende:

```
while True:
    pass
```

Klikk på Run, og forsøk å bevege joysticken til å ta en avlesning fra én av sensorene (**Figur 7-29**). Når den er ferdig med å rulle resultatet, trykker du på en annen retning. Gratulerer, du har bygd en håndholdt tricorder som vil imponere den intergalaktiske føderasjonen.



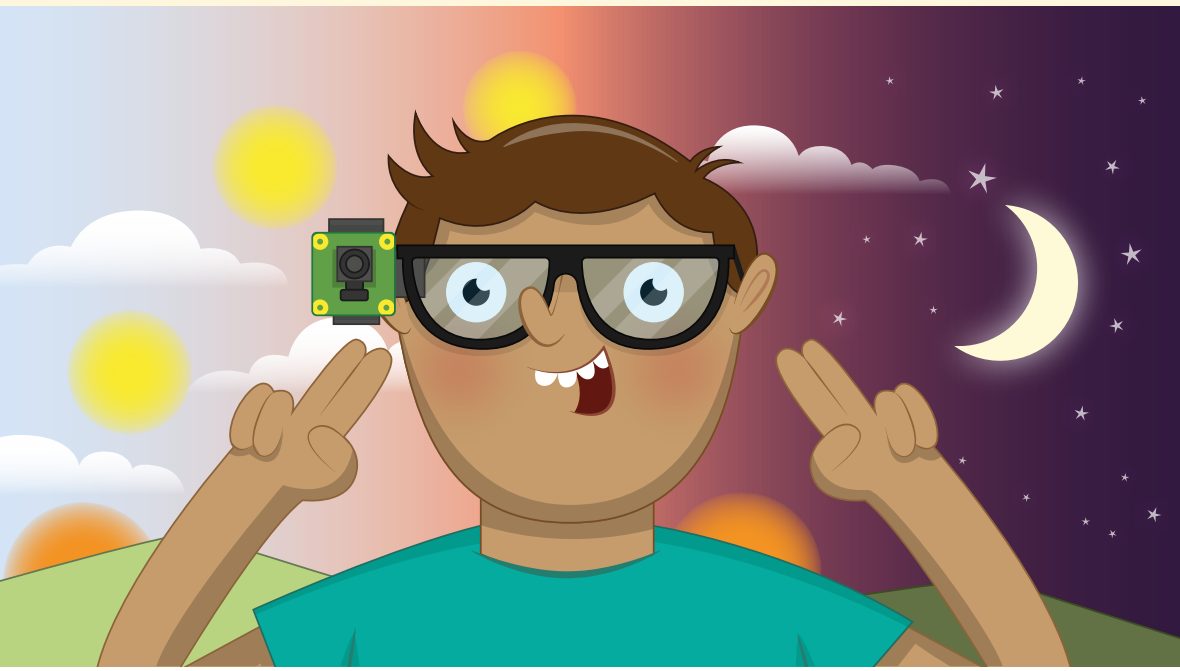
▲ **Figur 7-29:** Hver avlesning ruller over skjermen

Du finner ut mer om Sense HAT-prosjekter ved å følge koblingene i **Vedlegg D: Annet referansemateriale**.

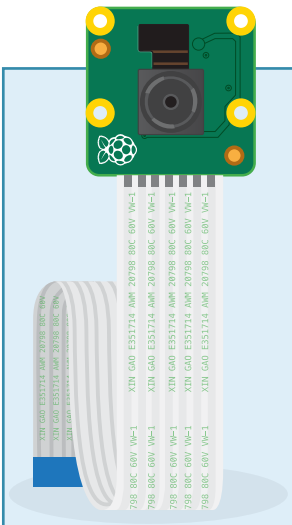
Kapittel 8

Raspberry Pi Camera Module

Når du kobler Camera Module eller High Quality Camera (HQ Camera) til Raspberry Pi, kan du ta bilder og ta opp videoer med høy oppløsning og lage fantastiske prosjekter med visuelt innhold



Hvis du alltid har hatt lyst til å bygge noe som kan se – kjent innen robotikkmiljøet som *visuelt innhold* – så er Raspberry Pis valgfrie Camera Module, eller det nye High Quality Camera, et flott utgangspunkt. Camera Module / HQ Camera er et lite firkantet kretskort med en tynn båndkabel, som kobles til CSI-porten (Camera Serial Interface) på Raspberry Pi (ikke tilgjengelig på Raspberry Pi 400). De produserer stillbilder med høy oppløsning og bevegelige videosignaler som kan brukes som de er eller integreres i dine egne programmer.



KAMERATYPER!

Det finnes tre typer Raspberry Pi-kameraer: standard Camera Module, NoIR-versjonen og High Quality (HQ) Camera. Hvis du vil ta vanlige bilder eller ta opp vanlige videoopptak i godt opplyste omgivelser, bør du bruke standard Camera Module. Hvis du vil bruke spesielle objektiver og ønsker den beste bildekvaliteten, kan du bruke HQ Camera Module. NoIR Camera Module har fått navnet sitt fordi den ikke har noe infrarødt filter eller IR-filter. Modulen er utviklet for bruk med infrarøde lyskilder så den kan ta bilder og ta opp video i totalt mørke. Hvis du bygger en fuglekasse, et sikkerhetskamera eller et annet prosjekt som involverer nattesyn, trenger du NoIR-versjonen. Du må imidlertid huske å kjøpe en infrarød lyskilde samtidig.

Standard og NoIR Raspberry Pi Camera Modules er basert på en Sony IMX219 bildesensor. Dette er en *sensor på 8 megapiksler*, noe som betyr at den kan ta bilder med opptil 8 millioner piksler. Det gjør den ved å ta bilder med en bredde på opptil 3280 piksler og en høyde på opptil 2464 piksler. I tillegg til stillbilder kan Camera Module ta opp videoopptak med full HD-oppløsning med en bildefrekvens på 30 fps. Hvis du vil ha jevnere video eller ønsker å skape en sakte film-effekt, kan du redusere oppløsningen for å få kameraet til å filme ved en høyere bildefrekvens: 60 fps for 720p videoopptak, og opptil 90 fps for 480p opptak (VGA).

High Quality Camera bruker en Sony IMX477 sensor på 12,3 megapiksler. Det er høyere enn sensoren i standard Camera Module og NoIR Camera Module – noe som betyr at den fanger opp mer lys og produserer bilder av høyere kvalitet. I motsetning til de to kameramodulene har ikke HQ Camera et objektiv. Derfor kan det ikke ta bilder eller ta opp videoer. Du kan bruke et hvilket som helst objektiv med C- eller CS-feste; andre objektivfester kan brukes med en egmet C- eller CS-adapter. Hvis du vil vite mer om hvordan du fester et objektiv, ser du **Vedlegg F: High Quality Camera**.

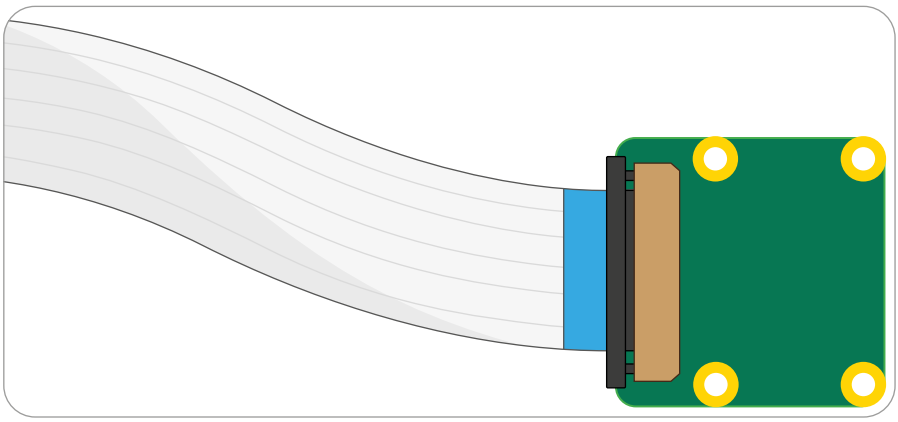
RASPBERRY PI 400

Dessverre er Raspberry Pi Camera Modules kompatible ikke med Raspberry Pi 400. Du kan bruke USB-webkameraer som et alternativ, men du kan ikke bruke spesifikke programvareverktøy for Raspberry Pi Camera Module, som er inkludert i Raspberry Pi OS.

Installere kameraet

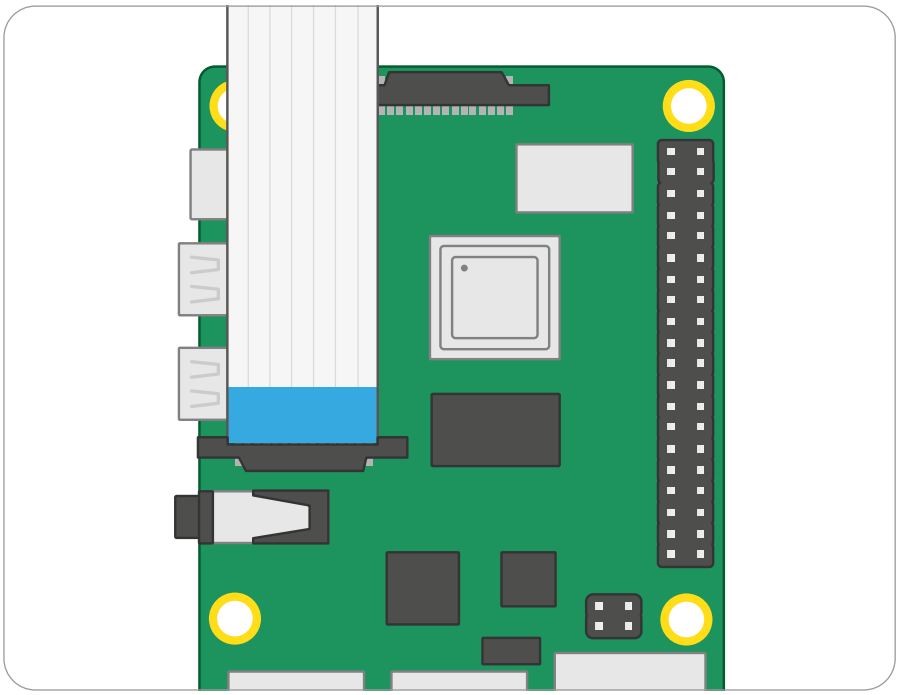
Som alle maskinvaretillegg bør Camera Module eller HQ Camera kun kobles til eller fra Raspberry Pi når strømmen er slått av og strømkabelen er koblet fra. Hvis Raspberry Pi er slått på, velger du Slå av i Raspberry-menyen. Deretter venter du til den slås av og trekker ut kontakten.

I de fleste tilfeller vil den medfølgende båndkabelen allerede være koblet til Camera Module eller HQ Camera. Hvis ikke snur du kameraet opp ned, slik at sensoren er på bunnen, og ser etter en flat plastkontakt. Sett neglene dine forsiktig under den grå fliken og trekk den ut til kontakten er delvis uttrukket. Skyv båndkabelen, med sølvkantene vendt nedover og den blå plastdelen vendt oppover, under fliken du nettopp trakk ut. Deretter skyver du fliken forsiktig inn igjen til den klikker på plass (**Figur 8-1**). Det spiller ingen rolle hvilken ende av kabelen du bruker. Hvis kabelen er riktig installert, sitter den godt og faller ikke ut hvis du drar litt i den. Hvis ikke trekker du klaffen ut og prøver en gang til.



Figur 8-1: Koble båndkabelen til Camera Module

Installer den andre enden av kabelen på samme måte. Finn kameraporten (CSI) på Raspberry Pi og trekk klaffen forsiktig oppover. Hvis Raspberry Pi er installert i et etui, kan det være lettere å fjerne det først. Hold Raspberry Pi slik at HDMI-porten vender mot deg, og skyv båndkabelen inn slik at sølvkantene er til venstre og den blå plastdelen til høyre (**Figur 8-2**). Deretter skyver du klaffen forsiktig på plass igjen. Hvis kabelen er riktig installert, sitter den godt og faller ikke ut hvis du drar litt i den. Hvis ikke trekker du klaffen ut og prøver en gang til.



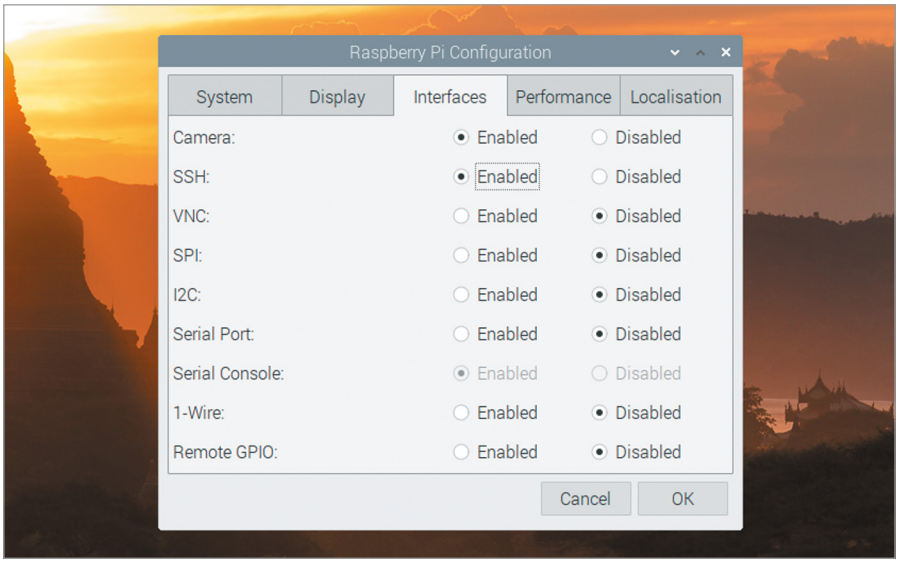
Figur 8-2: Koble båndkabelen til kamera- eller CSI-porten på Raspberry Pi

Camera Module leveres med et lite blått plastdeksel som dekker objektivet, for å beskytte det mot riper under produksjon, frakt og installasjon. Finn den lille plastklaffen og trekk den forsiktig av objektivet for å gjøre kameraet klart til bruk.

JUSTERING AV FOKUS

Camera Module leveres vanligvis med et lite plasthjul som brukes til å justere objektivets fokus. Det fabrikkinnstilte fokuset er vanligvis perfekt. Men hvis du bruker kameraet til å ta presise nærbilder, kan du skyve hjulet over objektivet og dreie forsiktig på det for å justere fokus manuelt. Du finner mer informasjon om fokusering av HQ Camera i **Vedlegg F**.

Koble strømforsyningen til Raspberry Pi igjen, og vent til den har lastet inn Raspberry Pi OS. Før du kan bruke kameraet må du fortelle Raspberry Pi at et kamera er tilkoblet. Åpne bringebærikonmenyen, velg kategorien Innstillinger og klikk på Raspberry Pi Configuration. Når verktøyet er lastet inn, klikker du på kategorien Interfaces (grensesnitt). Gå til oppføringen Camera i listen og klikk på den runde knappen til venstre for «Enabled» (aktivert)for å slå den på (**Figur 8-3** på neste side). Klikk på OK, og verktøyet ber deg om å starte Raspberry Pi på nytt. Deretter er kameraet klart til bruk.

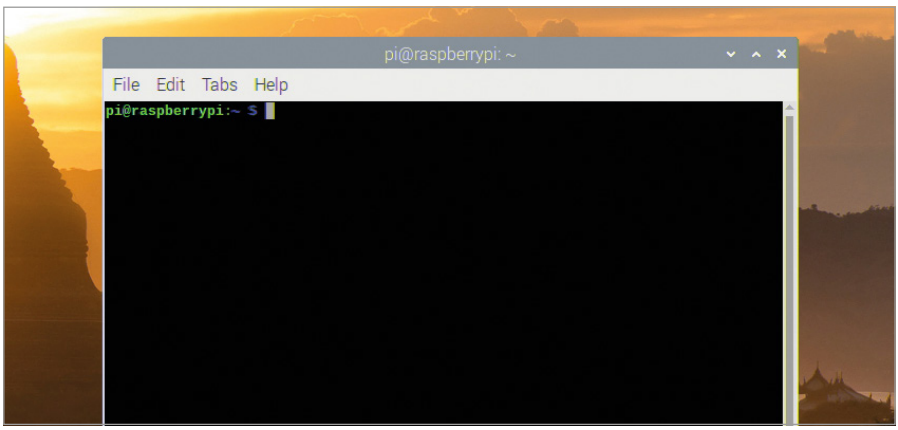


▲ **Figur 8-3:** Du må aktivere kameraet i Raspberry Pi Configuration

Teste kameraet

Når du vil bekrefte at Camera Module eller HQ Camera er riktig installert, og at du har aktivert grensesnittet i Raspberry Pi Configuration Tool, kan du bruke verktøyet **raspistill**. Dette, sammen med **raspivid** for videoer, er utviklet for å ta bilder med kameraet ved å bruke Raspberry Pis *kommandolinjegrensesnitt (CLI)*.

I motsetning til programmene du har brukt så langt, finner du ikke raspistill i menyen. I stedet klikker du på bringebærikonet for å laste inn menyen. Velg kategorien Tilbehør og klikk på LXTerminal. Et svart vindu med grønn og blå skrift vises (**Figur 8-4**): Dette er *terminalen*, som gir deg tilgang til kommandolinjegrensesnittet.

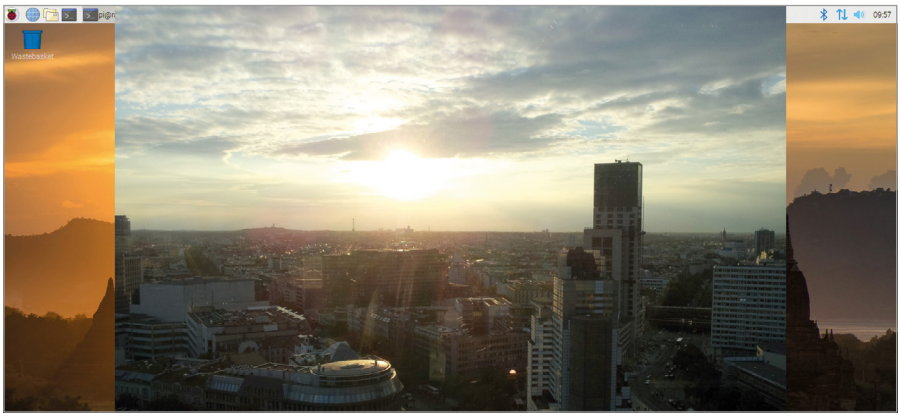


▲ **Figur 8-4:** Åpne et LXTerminal-vindu for å legge inn kommandoer

Når du skal teste kameraet, skriver du følgende i LXTerminal:

```
raspistill -o test.jpg
```

Straks du trykker på **ENTER**, ser du et stort bilde på skjermen, som viser det kameraet ser (**Figur 8-5**). Dette kalles *direkte forhåndsvisning*, og med mindre du forteller raspistill noe annet, varer det i 5 sekunder. Når disse 5 sekundene er over, tar kameraet et enkelt stillbilde og lagrer det i hjemmemappen under navnet **test.jpg**. Hvis du vil ta et nytt bilde, skriver du inn den samme kommandoen en gang til, men husk å endre navnet på utdatafilen, etter `-o`. Ellers lagrer du det nye bildet over det første bildet!



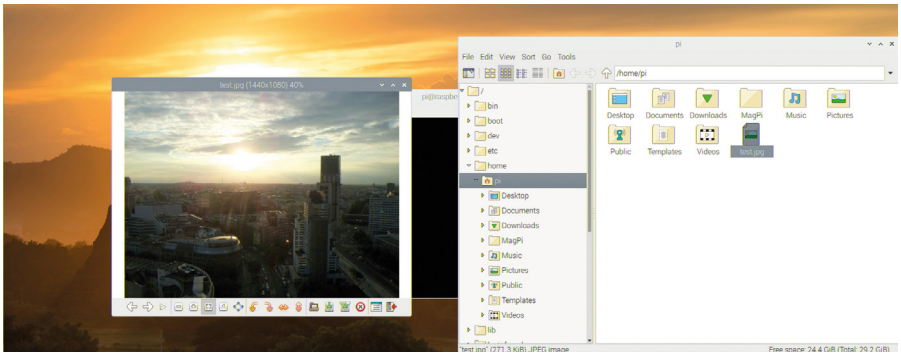
Figur 8-5: Direkte forhåndsvisning fra kameraet

Hvis direkte forhåndsvisning var opp ned, må du fortelle raspistill at kameraet er rotert. Camera Module er utformet slik at båndkabelen kommer ut av den nedre kanten. Hvis den kommer ut av en av sidene eller toppen, som noen ganger er tilfellet med tredjeparts kameratilbehør, kan du rotere bildet 90, 180 eller 270 grader ved hjelp av `-rot`-bryteren. For kameraer som er montert slik at kabelen kommer ut av toppen, bruker du følgende kommando:

```
raspistill -rot 180 -o test.jpg
```

Hvis båndkabelen kommer ut av høyre kant, bruker du en rotasjonsverdi på 90 grader; hvis den kommer ut av venstre kant, bruker du 270 grader. Hvis det opprinnelige bildet var i feil vinkel, kan du ta et nytt ved å bruke `-rot`-bryteren for å korrigere dette.

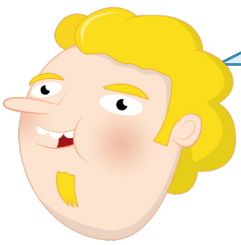
Du kan se bildet ditt ved å åpne Filbehandling fra Tilbehør-kategorien i bringebærmenyen: Bildet du har tatt, kalt **test.jpg**, vil være i **home/pi**-mappen. Finn bildet i listen over filer og dobbeltklikk på det for å laste det inn i en bildefremviser (**Figur 8-6**). Du kan også legge ved bildet i en e-postmelding, laste det opp til nettsteder via nettleseren eller dra det til en ekstern lagringsenhet.



▲ **Figur 8-6:** Åpne bildet du har tatt

Vi presenterer picamera

Den mest fleksible måten å kontrollere Camera Module eller HQ Camera på er å bruke Python via det praktiske picamera-biblioteket. Det gir deg full kontroll over kameraets funksjoner for å forhåndsvise opptak, ta bilder og ta opp video. Du kan også integrere dem i dine egne programmer, og til og med kombinere dem med programmer som bruker GPIO-modulen gjennom GPIO Zero-biblioteket.



PYTHON-PROGRAMMERING

Prosjektene i dette kapittelet forutsetter at du har erfaring med programmeringsspråket Python, Thonny IDE og GPIO-pinnene på Raspberry Pi. Hvis du ikke allerede har gjort det, kan du først gjennomgå prosjektene i **Kapittel 5: Programmering med Python** og **Kapittel 6: Fysisk databehandling med Scratch og Python**.

Lukk LXTerminal, hvis den fortsatt er åpen, ved å klikke på lukkeknappen (X) øverst til høyre i vinduet. Deretter laster du ned Thonny fra kategorien Utvikling i bringebærmenyen. Lagre det nye prosjektet som **Camera**, og begynn deretter å importere bibliotekene som programmet ditt trenger, ved å skrive inn følgende i skriptområdet:

```
from picamera import PiCamera
from time import sleep
camera = PiCamera()
```

Den siste linjen lar deg kontrollere Camera Module eller HQ Camera ved hjelp av **camera**-funksjonen. Skriv inn følgende tekst for å starte:

```
camera.start_preview()
sleep(10)
camera.stop_preview()
```

Klikk på Run, og skrivebordet forsvinner. I stedet ser du en forhåndsvisning av det kameraet kan se, i fullskjerm (**Figur 8-7**). Prøv å flytte den litt rundt, eller vink med hånden foran objektivet, så ser du at bildet på skjermen endres tilsvarende. Etter 10 sekunder lukkes forhåndsvisningen, og programmet avsluttes. I motsetning til forhåndsvisningen fra raspistill, blir imidlertid ingen bilder lagret etterpå.

Figur 8-7: En direkte forhåndsvisning av kameravisningen i fullskjerm

Hvis forhåndsvisningen er opp ned, kan du rotere bildet for justere det. Like under linjen `camera = PiCamera()` skriver du følgende:

```
camera.rotation = 180
```

Hvis forhåndsvisningen var opp ned, får denne linjen alt til å se riktig ut. På samme måte som raspistill kan du bruke **camera.rotation** for å rotere bildet 90, 180 eller 270 grader – avhengig av om kabelen kommer ut fra toppen eller fra høyre eller venstre side av Camera Module. Husk å bruke **camera.rotation** på starten av et program du skriver, for å unngå å ta bilder eller ta opp video som er opp ned.

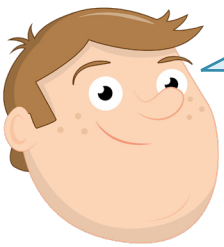
Ta stillbilder

Når du skal ta et bilde, i stedet for bare å vise en forhåndsvisning, må du endre programmet. Begynn med å redusere forsinkelsen for forhåndsvisningen: Gå til linjen **`sleep(10)`**, og endre den til:

```
sleep(5)
```

Direkte under denne linjen, legger du til følgende tekst:

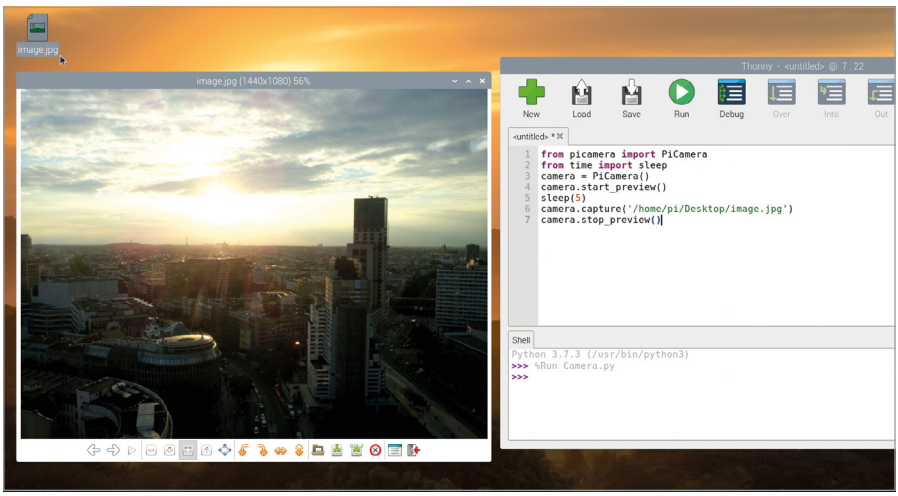
```
camera.capture('/home/pi/Desktop/image.jpg')
```



PÅ TIDE Å JUSTERE

Når kameraet er i forhåndsvisningsmodus, analyserer det videoen for å se om den trenger å justere innstillingene for å få best mulig kvalitet. Du ser dette hvis du er i et miljø med veldig dårlig eller veldig sterk belysning. Først er det umulig å se forhåndsvisningen, men deretter blir den tydeligere. For å gi kameraet tid til å justere seg bør du alltid legge inn en forhåndsvisningstid på minst 2 sekunder i programmet ditt før du tar et bilde.

Funksjonen **`camera.capture`** ber Python om å lagre et stillbilde. Programmet trenger ikke bare å vite hva bildet skal hete, men også i hvilken mappe det skal lagres. I dette eksemplet lagrer du det på skrivebordet. Du finner det rett under papirkurven. Hvis Thonny-vinduet er i veien, bare klikker og drar du på tittellinjen for å flytte det. Dobbeltklikk på filen for å se bildet du har tatt (**Figur 8-8**). Gratulerer: Du har programmert et kamera.



▲ **Figur 8-8:** Åpne bildet du har tatt

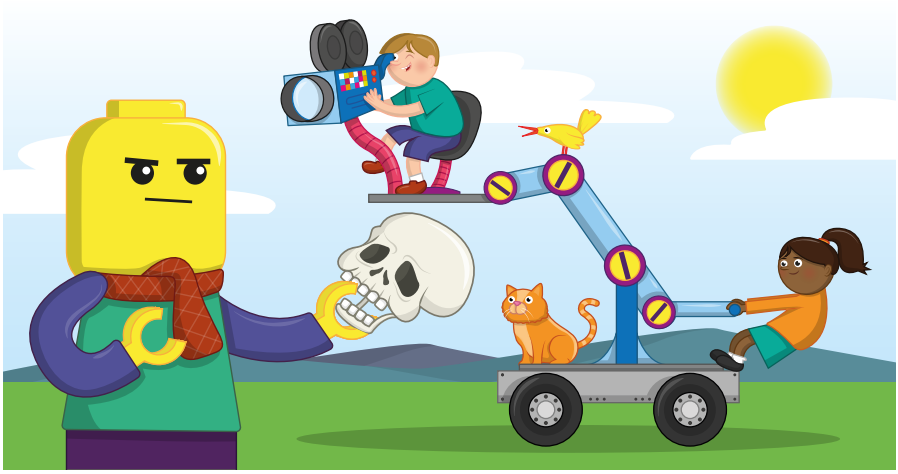
Ta opp video

I tillegg til å ta stillbilder kan du ta opp video. Slett alt mellom linjene **`camera.start_preview()`** og **`camera.stop_preview()`**, og skriv deretter inn følgende under **`camera.start_preview()`**:

```
camera.start_recording('/home/pi/Desktop/video.h264')
sleep(10)
camera.stop_recording()
```

Forhåndsvisningen av kameraet vises som før, men denne gangen blir det også tatt opp til en fil på skrivebordet. Vent til de ti sekundene du har bedt Python om å hvile, har gått. Du kan kanskje danse litt foran kameraet for å gjøre videoen interessant. Når forhåndsvisningen er avsluttet, finner du videofilen din på skrivebordet.

Hvis du vil spille av videoen, dobbeltklikker du på filen **video.h264** på skrivebordet. Videoen starter. Hvis du danset, vil du se dansingen. Når videoen er ferdig, avsluttes spillerprogramvaren med en vennlig melding i LXTerminal. Gratulerer: Nå kan du ta opp videoopptak med Raspberry Pi Camera Module eller HQ Camera.



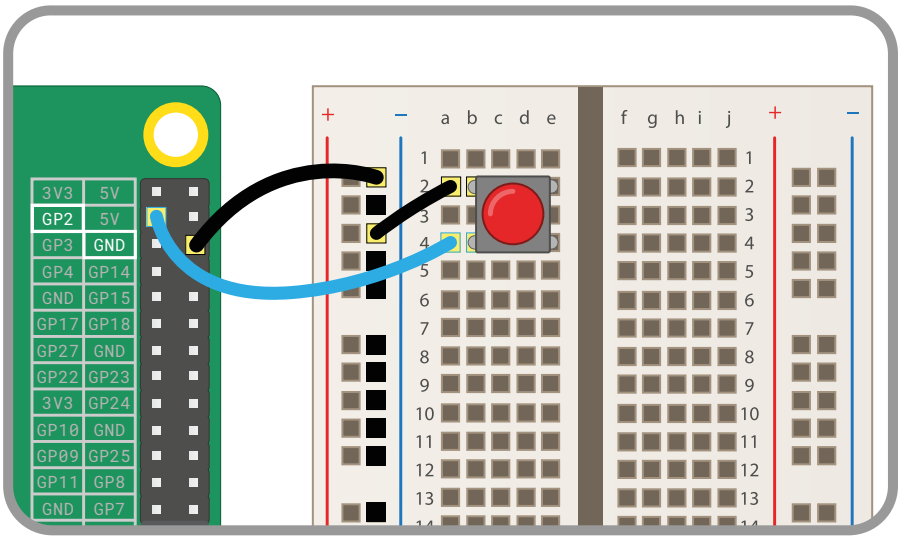
Stop-motion-animasjon med trykkbryter

Nå kan du bruke det du har lært i dette kapittelet, og det du har lært i **Kapittel 6**: **Fysisk databehandling**, som forklarer hvordan du kobler maskinvare til Raspberry Pis GPIO-pinnerekke. Det er på tide å bygge noe spesielt: ditt egne studio for å lage stop-motion-animasjon.

Stop-motion-animasjon er en prosess som går ut på å ta mange bilder av stillobjekter, for eksempel modellbiler eller actionfigurer, og deretter flytte gjenstandene litt mellom hvert bilde. Objektene beveger seg aldri i noen av bildene. Hvis du viser dem etter hverandre raskt nok, vil det likevel se ut som om de beveger seg så raskt eller så sakte som du vil.

Til dette prosjektet trenger du et trykkbryter, et koblingsbrett, jumperkabler av typen hann/hann (M2M) og hann/hunn (M2F). Hvis du ikke har et koblingsbrett, kan du koble til bryteren ved hjelp av hunn/hunn-kabler (F2F) i stedet, men da blir det vanskeligere å trykke på den. Hvis du trenger å friske opp kunnskapene om noen av disse komponentene, kan du se **Kapittel 6: Fysisk databehandling med Scratch og Python**. Du trenger også gjenstander du kan animere. Det kan være alt fra en leirklump til en lekebil eller en actionfigur.

Begynn med å lage kretsen: Legg til trykkbryteren på koblingsbrettet, og koble jordingsskinnen til en jordingspinnen på Raspberry Pi (merket GND i **Figur 8-9**) ved å bruke en jumperkabel av typen hunn/hunn. Bruk en jumperkabel av typen hann/hann for å koble det ene benet på bryteren til jordingsskinnen på koblingsbrettet. Deretter bruker du en jumperkabel av typen hann/hunn for å koble det andre benet på bryteren til GPIO-pinne 2 (merket GP2 i **Figur 8-9**).



▲ **Figur 8-9:** Koblingsskjema som viser hvordan du kobler en trykkbryter til GPIO-pinnene

Start et nytt prosjekt i Thonny og lagre det som **Stop Motion**. Begynn med å importere og sette opp bibliotekene du trenger for å bruke kameraet og GPIO-porten:

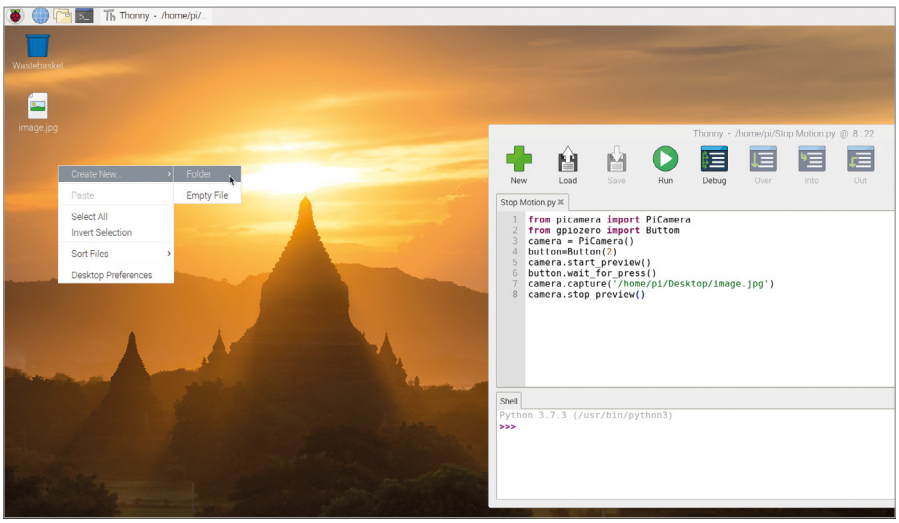
```
from picamera import PiCamera
from gpiozero import Button
camera = PiCamera()
button = Button(2)
```

Skriv deretter følgende:

```
camera.start_preview()
button.wait_for_press()
camera.capture('/home/pi/Desktop/image.jpg')
camera.stop_preview()
```

Klikk på Run, så ser du en forhåndsvisning av det som kameraet peker mot. Forhåndsvisningen forblir på skjermen til du trykker på trykkbryteren. Hvis du trykker på den nå, lukkes forhåndsvisningen etter at programmet har lagret et bilde på skrivebordet. Finn bildet med navnet **image.jpg**. Dobbeltklikk på bildet for å åpne det, så du kan sjekke om programmet fungerer.

Stop-motion-animasjon innebærer å lage mange stillbilder for å gi inntrykk av bevegelse når de settes sammen. Hvis du lagrer alle disse enkeltbildene på skrivebordet ditt, vil det føre til kaos. Du trenger en mappe der du kan lagre alle bildene. Høyreklikk hvor som helst i et tomt område på skrivebordet (uten filer eller ikoner), og velg deretter Opprett ny og Mappe (**Figur 8-10**). Kall mappen **animasjon**, med små bokstaver, og klikk deretter på OK-knappen.



▲ **Figur 8-10:** Opprett en ny mappe for bildene du har tatt

Det er ikke så bra å måtte starte programmet på nytt hver gang du tar et bilde for animasjonen. Derfor bør du endre programmet slik at det kjører i en løkke. I motsetning til de forrige løkkene du har opprettet, må du finne en elegant måte å lukke denne løkken på. Hvis du ikke gjør det, og du stopper programmet mens kameraforhåndsvisningen vises, kan du ikke se skrivebordet lenger. For å gjøre dette må du bruke to spesielle instruksjoner: `try` og `except`.

Start med å slette alt etter **camera.start_preview(),**. Deretter skriver du følgende:

```
frame = 1
```

Dette oppretter en ny variabel, **frame** som programmet bruker til å lagre gjeldende bildenummer. Du skal bruke dette snart for å sikre at du lagrer en ny fil hver gang. Hvis ikke lagrer du bare bildet over det første bildet hver gang du trykker på knappen.

Deretter konfigurerer du løkken ved å skrive følgende:

```
while True:
    try:
```

Den nye instruksjonen **try** forteller Python at programmet skal kjøre en hvilken som helst kode som er inni den. Det skal være koden for å ta bilder. Skriv følgende:

```
button.wait_for_press()
camera.capture('/home/pi/Desktop/animation/frame%03d.jpg' % frame)
frame += 1
```

Det er et par smarte triks i disse tre kodelinjene. Det første er i bildets filnavn: Når du skriver **%03d**, ber du Python om å ta et tall og deretter legge til så mange foranstilte nuller som kreves for å få et tresifret tall. Dermed endres 1 til 001, 2 til 002, og 10 til 010. Du trenger dette i programmet for å holde filene i riktig rekkefølge og for å sikre at du ikke skriver over en fil du allerede har lagret.

Teksten **% frame** på slutten av linjen ber Python om å bruke nummeret på bildevariabelen i filnavnet. For å sikre at hver fil er unik, øker den siste linjen – **frame += 1** – bildevariabelen med 1. Første gang du trykker på knappen, økes **frame** fra 1 til 2, neste gang fra 2 til 3 og så videre.

For øyeblikket gir ikke koden din klare resultater når du er ferdig med å ta bilder. For å løse problemet trenger du å legge til **except** for **try**. Skriv inn følgende tekst, men husk å fjerne ett innrykk på første linje, slik at Python vet at det ikke er en del av **try**-delen:

```
    except KeyboardInterrupt:
        camera.stop_preview()
        break
```

Det ferdige programmet vil se slik ut:

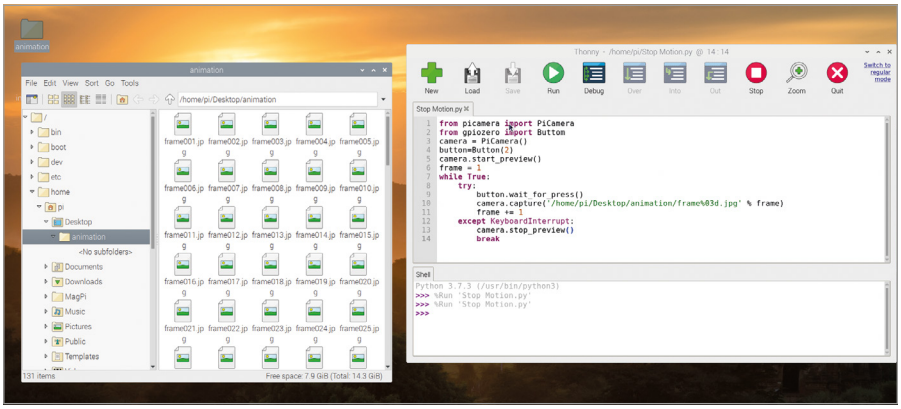
```
from picamera import PiCamera
from time import sleep
from gpiozero import Button
camera = PiCamera()
button = Button(2)
camera.start_preview()
frame = 1
while True:
    try:
        button.wait_for_press()
        camera.capture('/home/pi/Desktop/animation/frame%03d.jpg'
% frame)
        frame += 1
    except KeyboardInterrupt:
        camera.stop_preview()
        break
```

Prøv å klikke på Run, men i stedet for å trykke på knappen, trykker du på **CTRL**- og **C**-tastene på tastaturet. Du trenger ikke å trykke på begge tastene samtidig: Bare hold inne **CTRL**-tasten, trykk og slipp **C**-tasten, og slipp deretter **CTRL**. Disse to tastene fungerer som et avbrudd, og ber Python om å stoppe alt det gjør. Uten linjen `except KeyboardInterrupt:` ville Python umiddelbart avslutte og la kameraets forhåndsvisning blokkere skjermen, men med linjen på plass kjører Python koden som er inni den. I dette tilfellet er dette koden som ber programmet om å stoppe forhåndsvisningen av kameraet og avslutte på en klar måte.

Nå er du klar til å begynne å ta opp stop-motion-animasjonen. Plasser Camera Module eller HQ Camera der den/det kan se objektene du skal animere. Sørg for at den/det ikke beveger seg. Hvis modulen eller kameraet beveger seg, ødelegges effekten. Plasser objektene i startposisjonene. Deretter klikker du på Run for å starte programmet. Sjekk at alt ser bra ut i forhåndsvisningen. Trykk på trykkbryteren for å ta det første bildet.

Flytt litt på objektene. Jo mindre du flytter dem mellom hver bilde, jo jevnere blir den ferdige animasjonen. Trykk på trykkbrtyeren igjen for å ta et bilde til. Fortsett slik til animasjonen er ferdig. Jo flere bilder du tar, desto lenger blir animasjonen.

Når du er ferdig, trykker du på **CTRL+C** for å lukke programmet. Dobbeltklikk deretter på **animation**-mappen på skrivebordet for å se bildene du har tatt (**Figur 8-11** på neste side). Dobbeltklikk på et hvilket som helst bilde for å åpne det og se det mer detaljert.



▲ **Figur 8-11:** Bildene du har tatt, vist i mappen

For øyeblikket har du bare en mappe full av stillbilder. Hvis du vil lage en animasjon, må du gjøre dem om til en video. Det gjør du ved å klikke på bringebærikonet for å laste inn menyen. Velg kategorien Tilbehør og klikk på LXTerminal. Dette åpner et *kommandolinjegrensesnitt*, beskrevet mer detaljert i **Vedlegg C**, der du kan skrive kommandoer til Raspberry Pi. Når LXTerminal er lastet inn, starter du med å endre til mappen du har opprettet, ved å skrive følgende:

```
cd Desktop/animation
```

Det er viktig at Desktop skrives med stor forbokstav «D». Raspberry Pi OS *skiller mellom store og små bokstaver*, så hvis du ikke skriver inn et kommando- eller mappenavn nøyaktig slik det opprinnelig ble skrevet, vil den ikke fungere! Når du har endret mapper, skriver du følgende:

```
ffmpeg -i frame%03d.jpg -r 10 animation.h264
```

Dette bruker et program som kalles **ffmpeg**, for å ta stillbildene i mappen og konvertere dem til en video som kalles **animation.h264**. (Merk: hvis ffmpeg ikke er tilgjengelig, kan du installere det med `sudo apt-get install ffmpeg`.) Avhengig av hvor mange stillbilder du har tatt, kan denne prosessen ta noen minutter. Du vet at den er ferdig når du ser LXTerminal-meldingen igjen.

Når du vil spille av videoen, går du til filen **animation.h264** i **animation**-mappen. Dobbeltklikk på filen for å åpne den. Alternativt kan du spille den av fra LXTerminal ved å skrive følgende:

```
omxplayer animation.h264
```

Når videoen er lastet inn, ser du stop-motion-animasjonen. Gratulerer: Du har forvandlet Raspberry Pi til et kraftig animasjonsstudio.

Hvis animasjonen beveger seg for raskt eller for sakte, kan du endre **`-r 10`**-delen av **`ffmpeg`**-kommandoen til et lavere eller høyere tall. Dette er bildefrekvensen, eller bilder per sekund (fps) i videoen. Et lavere tall får animasjonen til å kjøre saktere, men den vil ikke være like jevn. Med et høyere tall ser animasjonen jevnere ut, men den kjører raskere.

Hvis du vil lagre videoen, må du dra og slippe den fra skrivebordet til mappen Videoer. Ellers kommer du til å overskrive filen neste gang du kjører programmet.

Avanserte kamerainnstillinger

Hvis du trenger mer kontroll over Raspberry Pi Camera Module eller HQ Camera, kan du bruke Python picamera-biblioteket for å få tilgang til forskjellige innstillinger. Disse innstillingene, sammen med standardverdiene, er beskrevet nedenfor. Du kan inkludere dem i dine egne programmer.

`camera.awb_mode = 'auto'`

Dette stiller inn modusen automatisk hvitbalanse på kameraet. Den kan settes til en av følgende moduser: **`off`**, **`auto`**, **`sunlight`**, **`cloudy`**, **`shade`**, **`tungsten`**, **`fluorescent`**, **`incandescent`**, **`flash`** eller **`horizon`**.. Hvis du synes bildene og videoene ser litt blå eller gule ut, kan du prøve en annen modus.

`camera.brightness = 50`

Dette stiller inn lysstyrken på kamerabildet, fra mørkest ved 0 til lysest ved 100.

`camera.color_effects = None`

Dette endrer fargeeffekten som for øyeblikket brukes av kameraet. Vanligvis skal du ikke røre denne innstillingen, men hvis du oppgir et par tall, kan du endre måten kameraet tar opp farger på: prøv **`(128, 128)`** for å lage et svart-hvitt-bilde.

`camera.contrast = 0`

Dette stiller inn kontrasten i bildet Et høyere tall får ting til å se mer dramatiske og sterkere ut, og et lavere tall får ting til å blekere ut. Du kan bruke et hvilket som helst tall mellom –100 for minimal kontrast og 100 for maksimal kontrast.

`camera.crop = (0.0, 0.0, 1.0, 1.0)`

Dette lar deg beskjære bildet, kutte av deler fra sidene og toppen, slik at du bare beholder den delen av bildet du trenger. Tallene representerer X-koordinat, Y-koordinat, bredde og høyde, og tar som standard med hele bildet. Prøv å redusere de to siste tallene – 0,5 og 0,5 er et godt utgangspunkt – for å se hvilken effekt denne innstillingen har.

`camera.exposure_compensation = 0`

Dette angir kameraets *eksponeringskompensering*, slik at du manuelt kan kontrollere hvor mye lys som fanges opp for hvert bilde. I motsetning til å endre lysstyrken styrer dette faktisk selve kameraet. Gyldige verdier varierer fra –25 for et svært mørkt bilde til 25 for et svært lyst bilde.

`camera.exposure_mode = 'auto'`
Dette angir *eksponeringsmodus*, eller logikken som Camera Module / HQ Camera bruker for å avgjøre hvordan et bilde skal eksponeres. Mulige moduser er: **`off`**, **`auto`**, **`night`**, **`backlight`**, **`spotlight`**, **`sports`**, **`snow`**, **`beach`**, **`verylong`**, **`fixedfps`**, **`antishake`** og **`fireworks`**.

`camera.framerate = 30`
Dette angir antall bilder som er tatt for å lage en video, per sekund, og kalles *bildefrekvensen*. En høyere bildefrekvens produserer en jevnere video, men tar opp mer lagringsplass. Høyere bildefrekvenser krever en lavere oppløsning, som du kan stille inn via **`camera.resolution`**.

`camera.hflip = False`
Dette vender kamerabildet rundt den horisontale aksen, eller X-aksen, når den er satt til **`True`**.

`camera.image_effect = 'none'`
Dette bruker en av mange bildeeffekter på videostrømmen, som vil være synlig i forhåndsvisningen, samt de lagrede bildene og videoene. Mulige effekter: **`blur`**, **`cartoon`**, **`colorbalance`**, **`colorpoint`**, **`colorswap`**, **`deinterlace1`**, **`deinterlace2`**, **`denoise`**, **`emboss`**, **`film`**, **`gpen`**, **`hatch`**, **`negative`**, **`none`**, **`oilpaint`**, **`pastel`**, **`posterise`**, **`saturation`**, **`sketch`**, **`solarize`**, **`washedout`** og **`watercolor`**.

`camera.ISO = 0`
Dette endrer ISO-innstillingen til kameraet, noe som påvirker hvor følsomt det er for lys. Som standard justerer kameraet dette automatisk avhengig av tilgjengelig lys. Du kan angi ISO selv ved å bruke en av følgende verdier: 100, 200, 320, 400, 500, 640, 800. Jo høyere ISO-verdi, desto bedre vil kameraet yte i omgivelser med lite lys, men jo mer kornete vil bildet eller videoen bli.

`camera.meter_mode = 'average'`
Dette styrer hvordan kameraet beregner mengden tilgjengelig lys når det stiller inn eksponeringen. Standardverdien er gjennomsnittet av mengden lys som er tilgjengelig gjennom hele bildet. Andre mulige moduser er **`backlit`**, **`matrix`** og **`spot`**.

`camera.resolution = (1920, 1080)`
Dette angir oppløsningen til bildet du har tatt, eller videoen du har tatt opp. Verdien representeres av to tall for bredde og høyde. Lavere oppløsninger tar opp mindre lagringsplass og lar deg bruke en høyere bildefrekvens. Høyere oppløsninger har bedre kvalitet, men tar opp mer lagringsplass.

`camera.rotation = 0`
Dette styrer rotasjonen av bildet, fra 0 grader til 90, 180 og 270 grader. Bruk dette hvis du ikke kan plassere kameraet slik at båndkabelen kommer ut av bunnen.

`camera.saturation = 0`
Dette styrer metningen på bildet, eller hvor vibrerende fargene skal være. Mulige verdier varierer fra –100 til 100.

`camera.sharpness = 0`
Dette styrer skarpheten i bildet. Mulige verdier varierer fra –100 til 100.

`camera.shutter_speed = 0`
Dette styrer hvor raskt lukkeren åpnes og lukkes når du tar bilder og tar opp videoer. Du kan angi lukkerhastigheten manuelt i mikrosekunder. Lengre lukkerhastigheter fungerer bedre i svakere lys og raskere lukkerhastigheter i sterkere lys. Dette skal vanligvis stå i den automatiske standardinnstillingen.

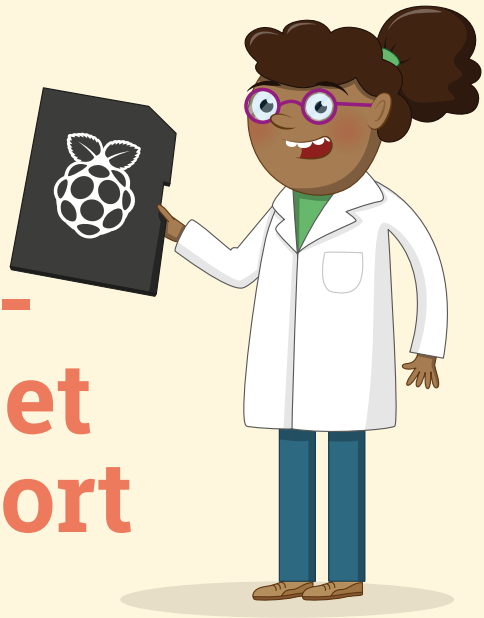
`camera.vflip = False`
Dette vender kamerabildet rundt den vertikale aksen, eller Y-aksen, når den er satt til **`True`**.

`camera.video_stabilization = False`
Når den er satt til **`True`**, slås videostabilisering på. Du trenger bare dette hvis Camera Module eller HQ Camera beveger seg mens du tar opp, for eksempel hvis de er festet til en robot eller bæres rundt. Funksjonen brukes til å redusere risting i videoen.

Du finner mer informasjon om disse innstillingene, samt tilleggsinnstillinger som ikke er dokumentert her, ved å gå til **picamera.readthedocs.io**.

Vedlegg A

Installere et operativ-system på et microSD-kort



Du kan kjøpe microSD-kort med NOOBS (New Out of the Box Software) forhåndsinstallert fra alle gode Raspberry Pi-forhandlere, slik at du enkelt kan installere Raspberry Pi OS (tidligere kjent som Raspbian) for Raspberry Pi. Alternativt kan du følge veiledningen under for å bruke Raspberry Pi Imager for å installere et operativsystem manuelt på et tomt (eller gjenbrukt) microSD-kort.

> **ADVARSEL!**
>
> Hvis du har kjøpt et microSD-kort med NOOBS forhåndsinstallert, trenger du bare å sette det inn i Raspberry Pi. Denne veiledningen er beregnet på tomme microSD-kort, eller kort du har brukt tidligere, som du vil installere et nytt operativsystem på. Hvis følger denne veiledningen for et microSD-kort som allerede inneholder filer, går disse filene tapt. Derfor må du passe på at du har sikkerhetskopiert viktige filer før du begynner.

Laste ned Raspberry Pi Imager

Raspberry Pi OS er basert på Debian og er det offisielle operativsystemet for Raspberry Pi. Verktøyet Raspberry Pi Imager er den enkleste metoden for å installere Raspberry Pi OS på et microSD-kort for Raspberry Pi. Du kan laste det ned fra **rpf.io/downloads**. Merk: Denne metoden erstatter installasjon av operativsystemet via NOOBS, selv om sistnevnte fortsatt er tilgjengelig fra samme nedlastingsside.

Programmet Raspberry Pi Imager er tilgjengelig for maskiner som kjører Windows, macOS og Ubuntu Linux, så pass på at du velger riktig versjon for systemet ditt.

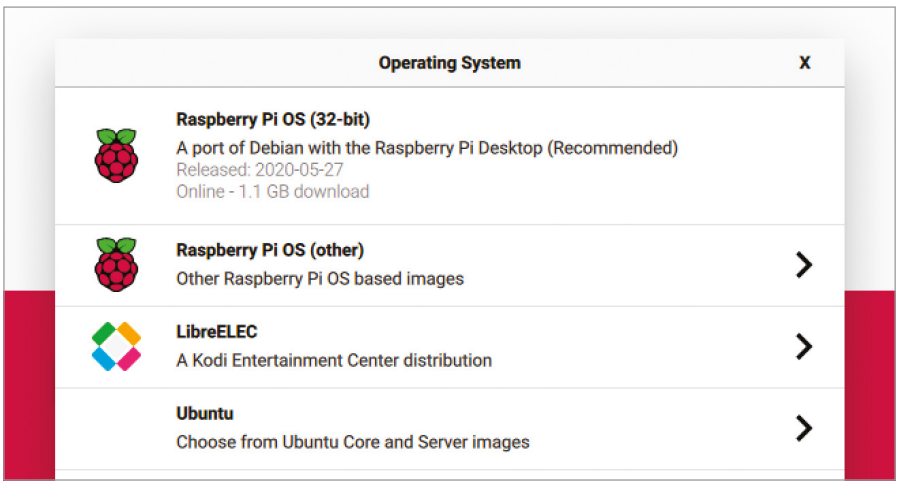
For macOS: Dobbeltklikk på den nedlastede DMG-filen. Du må kanskje endre innstillingene i Sikkerhet og personvern for å tillate apper som er lastet ned fra «App Store og identifiserte utviklere» for å kunne kjøre filen. Deretter drar du bare ikonet for Raspberry Pi Imager til mappen Programmer.

Windows-datamaskiner: Dobbeltklikk på den nedlastede EXE-filen. Når du blir bedt om det, klikker du på Ja-knappen for å kjøre filen. Deretter klikker du på Installer-knappen for å starte installasjonen.

Skriv operativsystemet til microSD-kortet

Sett microSD-kortet inn i PC-en eller Mac-maskinen: Du trenger en USB-adapter for microSD-kort med mindre den aktuelle maskinen har en innebygd kortleser. Merk: Kortet trenger ikke å være forhåndsformatert.

Start Raspberry Pi Imager. Klikk på knappen «Choose OS» for å velge hvilket operativsystem du vil installere. Det første alternativet er standard Raspberry Pi OS. Hvis du foretrekker den forenklede Lite-versjonen eller Full-versjonen (med all anbefalt programvare forhåndsinstallert), velger du «Raspberry Pi OS (other)». Du vil også se alternativene LibreELEC (velg riktig versjonen for Raspberry Pi-modellen din), Ubuntu Core eller Server.



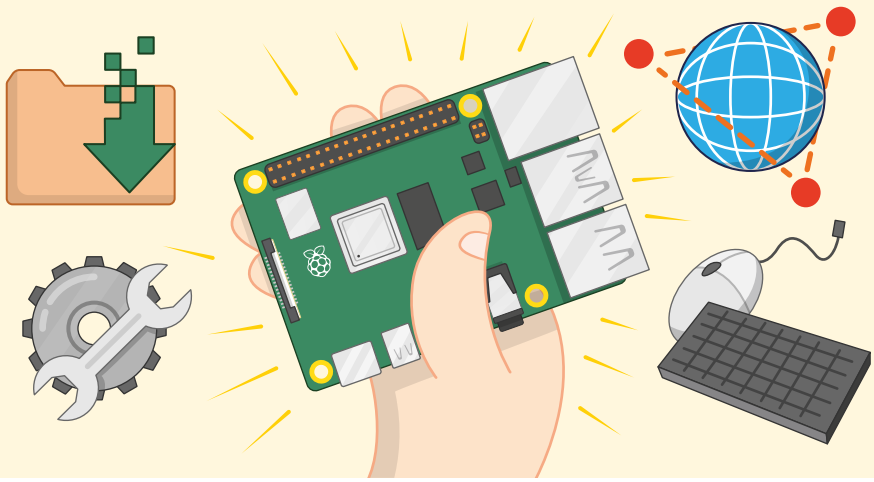
Merk: Hvis du vil installere et annet operativsystem, for eksempel Lakka, kan du bare laste ned bildefilen fra det aktuelle nettstedet. Deretter velger du alternativet «Use Custom» i Raspberry Pi Imager for å velge en tilpasset installasjon.

Når du har valgt et operativsystem, klikker du på knappen «Choose SD card» og velger microSD-kortet ditt (vanligvis ser du bare ett alternativ).

Til slutt klikker du på «Write» og venter mens verktøyet skriver det valgte operativsystemet til kortet ditt og bekrefter det. Når du er ferdig, kan du fjerne microSD-kortet. Deretter setter du det inn i Raspberry Pi og starter den i operativsystemet du nettopp har installert.

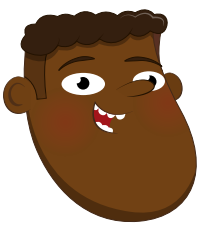
Vedlegg B

Installere og avinstallere programvare



Raspberry Pi OS leveres med en rekke populære programvarepakker som er håndplukket av Raspberry Pi Foundation. Dette er imidlertid ikke de eneste pakkene som fungerer med Raspberry Pi. Ved å følge denne fremgangsmåten kan du bla gjennom ekstra programvare, installere og avinstallere den – og dermed utvide funksjonaliteten til Raspberry Pi.

Fremgangsmåten i dette vedlegget skal brukes sammen med fremgangsmåten i kapittel **Kapittel 3: Bruke Raspberry Pi**, som forklarer hvordan du bruker verktøyet Anbefalt programvare. Hvis du ikke har lest dette, bør du gjøre det før du prøver metodene som beskrives i dette vedlegget.

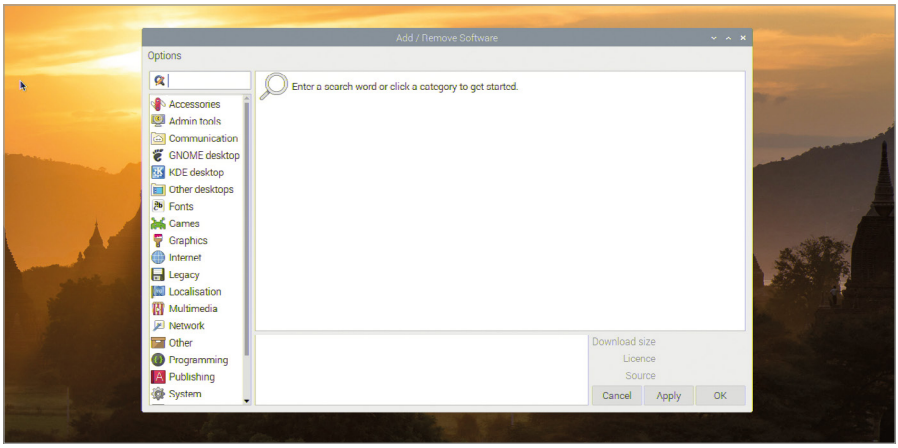


KORTKAPASITET

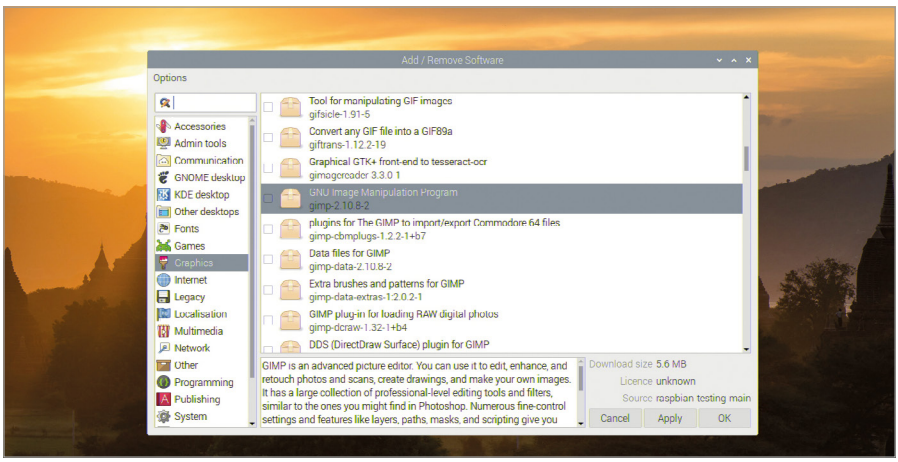
Hvis du legger til mer programvare på Raspberry Pi, vil de ta opp plass på microSD-kortet ditt. Et 16 GB kort eller større har plass til mer programvare. Hvis du vil sjekke om kortet du har tenkt å bruke, er kompatibelt med Raspberry Pi, kan du gå til **rpf.io/sdcardlist**.

Bla gjennom tilgjengelig programvare

Hvis du vil se og søke i listen over tilgjengelige programvarepakker for Raspberry Pi OS, ved å bruke våre såkalte *programvaredatabaser*, klikker du på bringebærikonet for å laste inn menyen. Deretter velger du kategorien Brukervalg og klikker på Add/Remove Software. Etter et par sekunder vises verktøyets hovedvindu.



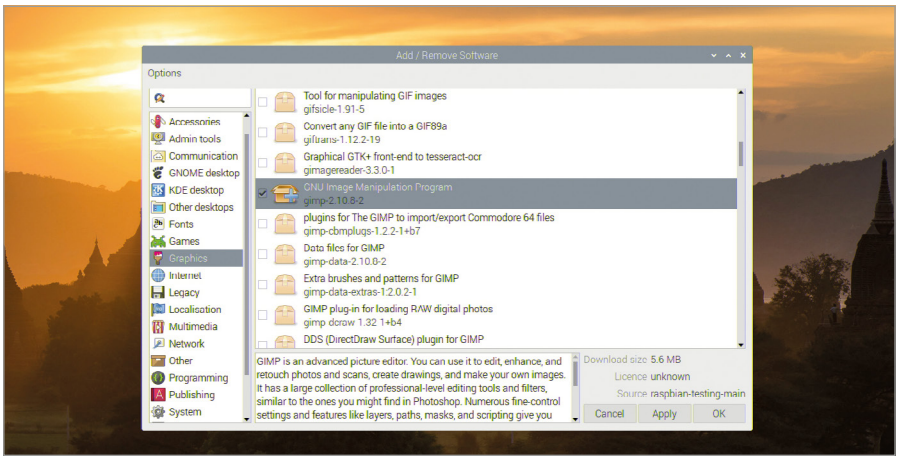
Venstre side av vinduet Add/Remove Software inneholder en liste over kategorier. Det er de samme kategoriene som du finner i hovedmenyen når du klikker på bringebærikonet. Når du klikker på en av kategoriene, vises en liste over tilgjengelig programvare i den aktuelle kategorien. Du kan også angi et søkeord i feltet øverst til venstre i vinduet, for eksempel «tekstredigeringsprogram» eller «spill». Da vil du se en liste med samsvarende programvarepakker fra en vilkårlig kategori. Hvis du klikker på en vilkårlig pakke, vil du se mer informasjon om pakken i feltet nederst i vinduet.



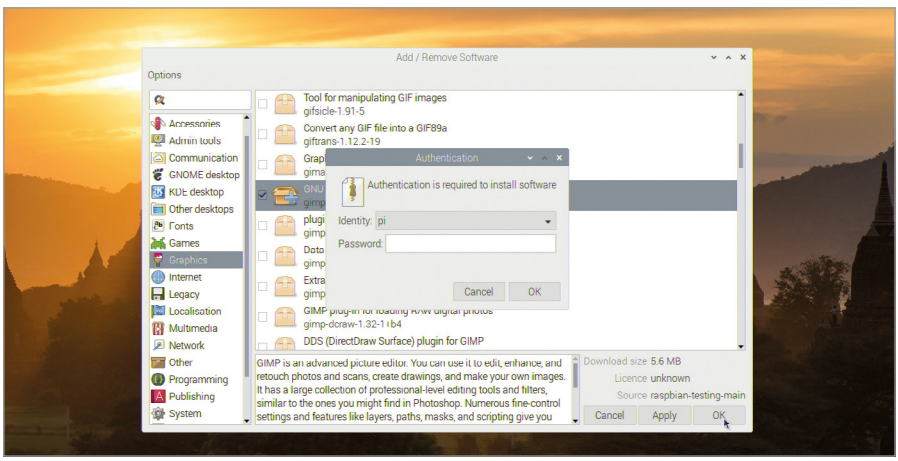
Hvis kategorien du har valgt, har mange tilgjengelige programvarepakker, kan det ta litt tid før Add/Remove Software viser den fullstendige listen.

Installere programvare

Du merker av for en pakke du vil installere, ved å klikke i boksen ved siden av den aktuelle pakken. Du kan installere mer enn én pakke om gangen ved å klikke i flere bokser for å merke av for flere pakker. Ikonet ved siden av pakken endres til en åpen boks med et plussymbol (+). Dette ikonet bekrefter at den skal installeres.

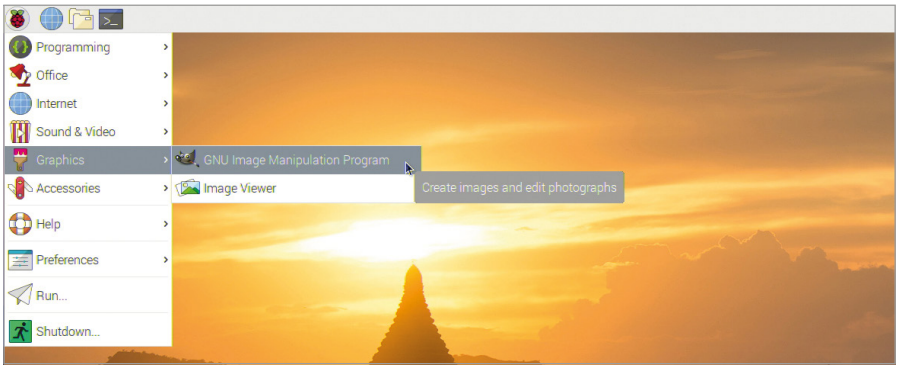


Når du er fornøyd med valgene, klikker du på OK eller Apply. Den eneste forskjellen er at OK lukker verktøyet Add/Remove Software når programvaren er installert, mens Apply lar det være åpent. Du blir bedt om å oppgi et passord for å bekrefte identiteten din. Det forhindrer at hvem som helst kan legge til eller fjerne programvare fra Raspberry Pi.



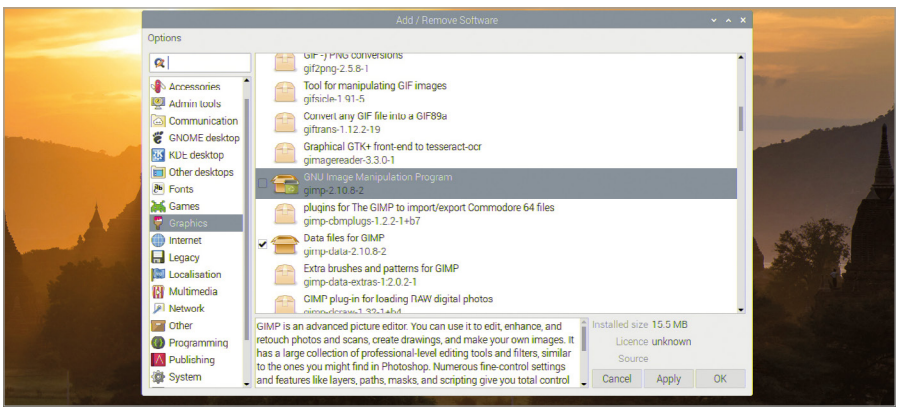
Når du installerer en enkeltpakke, vil du kanskje oppdage at andre pakker installeres samtidig. Disse pakkene kalles *avhengigheter*, det vil si pakker som programvaren du installerer, trenger for å fungere, for eksempel lydeffektpakker for et dataspill eller en database som brukes sammen med en webserver.

Når programvaren er installert, kan du finne den ved å klikke på bringebærikonet for å laste inn menyen og finne kategorien til programvarepakken. Merk: Menykategorien er ikke alltid er den samme som kategorien fra verktøyet Add/Remove Software. Enkelte typer programvare er ikke oppført i menyen i det hele tatt. Slik programvare er kjent som *kommandolinjeprogramvare* og må kjøres fra terminalen. Du finner mer informasjon om kommandolinjen og terminalen i **Vedlegg C: Kommandolinjegrensesnitt**.



Avinstallere programvare

Når du skal velge en pakke for fjerning eller *avinstallasjon*, kan du finne den i pakkelisten ved å bruke den praktiske søkefunksjonen. Deretter fjerner du avmerkingen i boksen ved siden av den aktuelle pakken ved å klikke i boksen. Du kan avinstallere mer enn én pakke om gangen. Du bare klikker i boksene for å fjerne flere pakker. Ikonet ved siden av pakken endres til en åpen boks ved siden av en liten papirkurv. Dette ikonet bekrefter at den skal avinstalleres.



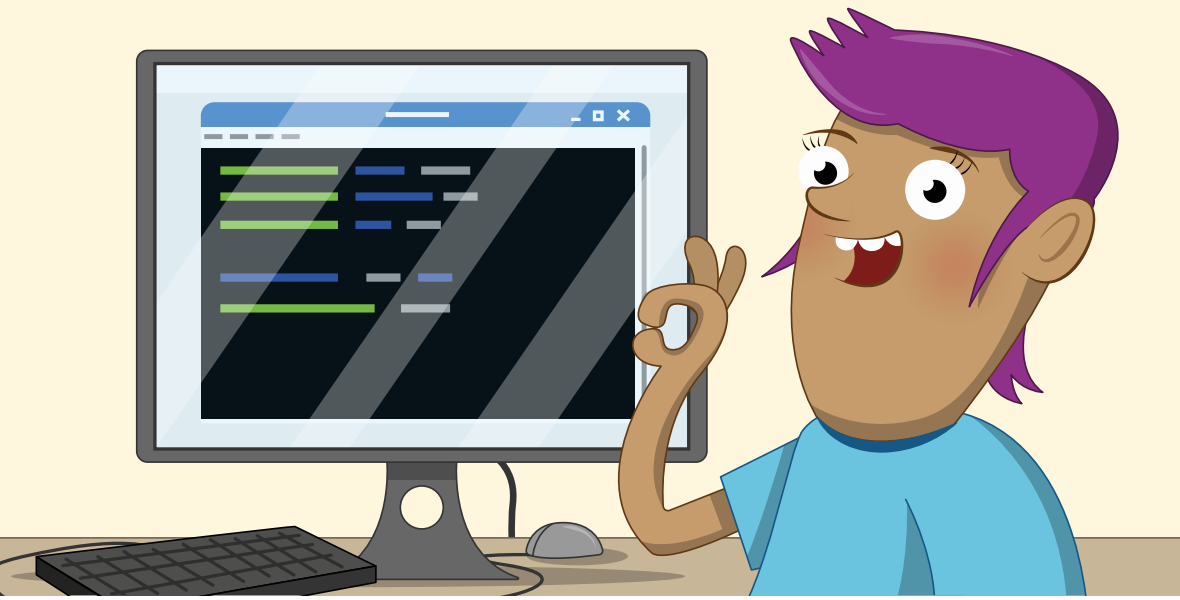
Som tidligere kan du klikke på OK eller Apply for å avinstallere de valgte programvarepakkene. Du blir bedt om å bekrefte passordet ditt med mindre du har gjort det for et par minutter siden. I tillegg kan du bli bedt om å bekrefte at du også vil fjerne eventuelle avhengigheter som er knyttet til programvarepakken. Når avinstallasjonen er fullført, forsvinner programvaren fra menyen til bringebærikonet. Filer du har opprettet ved hjelp av programvaren, for eksempel bilder for en grafikkpakke eller spillbonuser du har lagret, fjernes ikke.

ADVARSEL!

All programvare som er installert i operativsystemet til Raspberry Pi, vises i Add/Remove Software, inkludert programvare som kreves for at Raspberry Pi skal fungere. Hvis du fjerner for mange pakker, er det mulig at skrivebordet ikke lastes inn. For å unngå dette må du ikke avinstallere noe med mindre du er sikker på at du ikke lenger trenger det. Hvis det allerede har skjedd, installerer du Raspberry Pi OS på nytt ved å følge fremgangsmåten i **Kapittel 2: Komme i gang med Raspberry Pi**. Du kan eventuelt også installere operativsystemet på nytt ved å følge fremgangsmåten i **Vedlegg A**.

Vedlegg C

Kommando-linjegrensesnittet



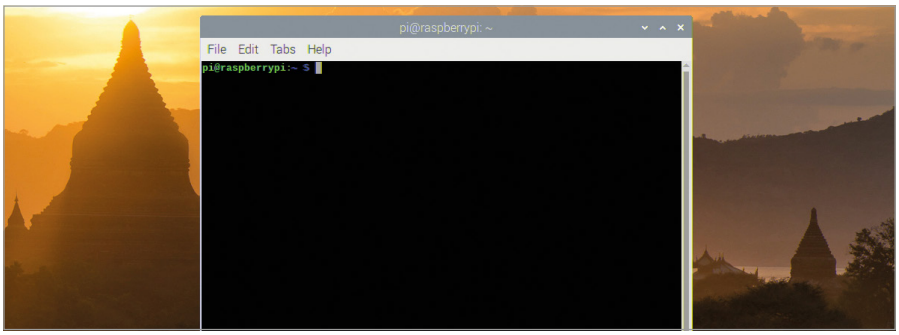
Selv om du kan behandle mesteparten av programvaren for Raspberry Pi gjennom en stasjonær datamaskin, kan enkelte programmer kun behandles ved å brukte en tekstbasert modus som kalles *kommandolinjegrensesnittet (CLI)* i et program som heter LXTerminal. De fleste brukere trenger som regel aldri å bruke CLI, men for de som ønsker å lære mer, gir denne veiledningen en enkel introduksjon.

MER INFORMASJON

Dette vedlegget er ikke ment å være en uttømmende veiledning for kommandolinjegrensesnittet Linux. Hvis du vil lære mer om bruk av CLI, kan du gå til **rpf.io/terminal** i en nettleser.

Laste inn LXTerminal

Du får tilgang til CLI gjennom LXTerminal, en programvarepakke som laster inn noe som teknisk sett er kjent som en *VTY-terminal (virtuell teleks)*. Dette er et navn som går tilbake til datamaskinens opprinnelse, da brukere utførte kommandoer ved hjelp av store elektromekaniske skrivemaskiner i stedet for å bruke et tastatur og en skjerm. Når du skal laste inn LXTerminal-pakken, klikker du på Raspberry-ikonet for å laste inn Raspberry Pi-menyen. Velg kategorien Tilbehør og klikk på LXTerminal.



Du kan dra LXTerminal-vinduet rundt på skjermen, endre størrelse på det, maksimere/minimee det akkurat som alle andre vinduer. Du kan også gjøre skriften større for å gjøre den enklere å se, eller mindre hvis du vil se mer tekst i vinduet. Klikk på Edit-menyen og velg henholdsvis Zoom In eller Zoom Out, eller trykk og hold inne **CTRL**-tasten på tastaturet samtidig som du trykker på **+** eller **–**.

Ledeteksten

Det første du ser i LXTerminal, er *ledeteksten*, som venter på instruksjoner fra deg. Ledeteksten på en Raspberry Pi som kjører Raspberry Pi OS, ser slik ut:

```
pi@raspberrypi:~ $
```

Den første delen av ledeteksten, **pi**, er brukernavnet ditt. Den andre delen, som kommer etter **@**-tegnet, er vertsnavnet til datamaskinen du bruker. Som standard er dette **raspberrypi**. Etter kolonet (**:**) ser du en tilde (**~**). Dette tegnet viser til hjemmekatalogen din og representerer *gjeldende arbeidskatalog (CWD)*. Til slutt ser du et dollarsymbol (**$**) som angir at brukeren er en *bruker uten administrative rettigheter*. Det betyr at du må oppgi et passord for å utføre oppgaver som å legge til og fjerne programvare.

Navigere

Prøv å skrive følgende tekst, og trykk deretter på **ENTER**-tasten:

```
cd Desktop
```

Ledeteksten endres til følgende:

```
pi@raspberrypi:~/Desktop $
```

Denne teksten viser at den gjeldende arbeidskatalogen er endret. Før var du i hjemmekatalogen, angitt med et tildesymbol (**~**), og nå er du i underkatalogen **Desktop**, som befinner seg nedenfor hjemmekatalogen. Du gjorde det ved å bruke **cd**-kommandoen – *endre katalog* (change directory).

STOR/LITEN BOKSTAV

Kommandolinjegrensesnittet i operativsystemet til Raspberry Pi skiller mellom store og små bokstaver. Det betyr at bruk av store og små bokstaver i kommandoer eller navn har forskjellig mening. Hvis du ser meldingen «finner ikke denne filen eller katalogen» når du prøver å endre katalog, må du sjekke om du har skrevet en stor D i ordet «Desktop».

Det finnes fire metoder for å gå tilbake til hjemmekatalogen. Du kan prøve en metode om gangen for å se hvordan de tar deg tilbake til underkatalogen **Desktop**. Første metode:

```
cd ..
```

Symbolet med to prikker (**..**) er en annen snarvei. Denne kommandoen tar deg tilbake til katalogen som ligger over gjeldende katalog, og kalles *overordnet katalog*. Da katalogen som ligger over **Desktop**, er hjemmekatalogen din, tas du tilbake dit. Endre katalogen til underkatalogen **Desktop** igjen, og prøv den andre metoden ved å skrive følgende:

```
cd ~
```

Denne metoden bruker tildesymbolet (**~**), som rett og slett betyr «endre til hjemmekatalogen». Kommandoen **cd ..** tar deg direkte til den overordnede katalogen, uansett hvilken katalog du befinner deg i. Denne kommandoen, derimot, tar deg til hjemmekatalogen uansett hvor du er. Det finnes imidlertid en enklere metode:

```
cd
```

Hvis du ikke angir navnet på en katalog, tar **cd** deg som standard tilbake til hjemmekatalogen. Den siste metoden for å gå tilbake til hjemmekatalogen går ut på å skrive inn følgende:

```
cd /home/pi
```

Denne metoden bruker noe som kalles en *absolutt bane*, og fungerer uansett hvilken arbeidskatalog du befinner deg i. På samme måte som å skrive bare **cd** eller **cd ~** tar denne kommandoen deg tilbake til hjemmekatalogen uansett hvor du er. I motsetning til de andre metodene må du imidlertid vite brukernavnet ditt.

Behandle filer

Hvis du vil øve deg på å arbeide med filer, endrer du til katalogen **Desktop** og skriver følgende tekst:

```
touch Test
```

Den fil med navnet **Test** vises på skrivebordet. Kommandoen **touch** brukes vanligvis til å oppdatere dato og klokkeslett i en fil. Hvis filen ikke finnes (som i dette tilfellet), oppretter kommandoen en fil.

Prøv å skrive følgende:

```
cp Test Test2
```

Du ser en annen fil med navnet **Test2** på skrivebordet. Dette er en *kopi* av den opprinnelige filen, helt identisk. Slett den ved å skrive følgende:

```
rm Test2
```

Dette *fjerner* filen, og du vil se at den forsvinner.

ADVARSEL!

Når du sletter filer ved å bruke Filbehandling, lagres de i Papirkurv, der du kan gjenopprette dem igjen. Filer du sletter med **rm**, er permanent slettet. Så vær nøyaktig når du skriver inn tekst.

Deretter prøver du å skrive følgende:

```
mv Test Test2
```

Denne kommandoen *flytter* filen, og du vil se at den opprinnelige**Test**-filen forsvinner og erstattes av **Test2**. Flytt-kommandoen, **mv**, kan også brukes på denne måten for å endre navn på filer.

Hvis du ikke er på skrivebordet, trenger du likevel å se filene i en katalog. Skriv følgende:

```
ls
```

Denne kommandoen *oppretter en liste* over innholdet i gjeldende katalog, eller i en annen katalog du angir. Hvis du vil vite mer, for eksempel hvordan du viser skjulte filer og rapporterer filstørrelser, kan du prøve å bruke svitsjer. Eksempel:

```
ls -larth
```

Disse svitsjene styrer **ls**-kommandoen: **l** endrer utdataene til en lang loddrett liste, **a** viser alle filer og kataloger, inkludert de som vanligvis er skjult, **r** reverserer den vanlige sorteringsrekkefølgen, **t** sorterer elementene etter endringstidspunkt, som i kombinasjon med **r** viser de eldste filene øverst og de nyeste filene nederst, og **h** bruker filstørrelser i lesbar form, som er enklere å forstå.

Kjøre programmer

Noen programmer kan bare kjøres på kommandolinjen, mens andre har både grafiske grensesnitt og kommandolinjegrensesnitt. Et eksempel på sistnevnte er Raspberry Pis Software Configuration Tool, som du vanligvis laster inn fra menyen til bringebærikonet.

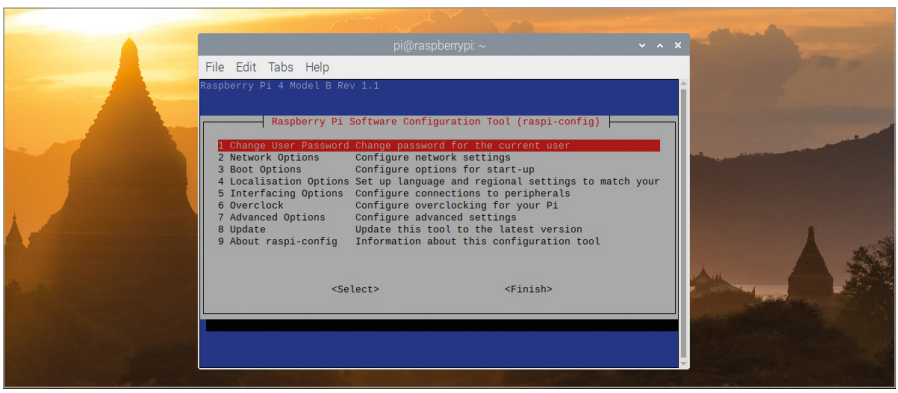
Skriv følgende:

```
raspi-config
```

Du ser en feilmelding som forteller deg at programvaren bare kan kjøres som *rot*, det vil si superbrukerkontoen på Raspberry Pi. Du kan også finne ut hvordan du gjør det, ved å skrive følgende:

```
sudo raspi-config
```

Delen **sudo** i kommandoen står for «*switch-user do*» og ber Raspberry Pi OS om å kjøre kommandoen som rotbruker.



Du trenger bare å bruke **sudo** når et program trenger avanserte *rettigheter*, for eksempel når det installerer eller avinstallerer programvare eller justerer systeminnstillinger. Du bør for eksempel aldri kjøre et spill ved å bruke **sudo**.

Trykk to ganger på **TAB**-tasten (tabulator) for å velge Finish, og trykk deretter på **ENTER** for å avslutte Raspberry Pis Software Configuration Tool og gå tilbake til kommandolinjegrensesnittet. Deretter skriver du følgende:

```
exit
```

Denne kommandoen avslutter økten i kommandolinjegrensesnittet og lukker LXTerminal-appen.

Bruke TTY-er

LXTerminal-appen er ikke den eneste metoden for å bruke kommandolinjegrensesnittet. Du kan også bytte til en av flere terminaler som allerede kjører, som kalles *telekser* eller *TTY-er*. Hold inne **CTRL**- og **ALT**-tastene på tastaturet og trykk samtidig på **F2**-tasten for å bytte til «tty2».

Du må logge på igjen med brukernavnet og passordet ditt. Deretter kan du bruke kommandolinjegrensesnittet akkurat som i LXTerminal. De kan være praktisk å bruke disse TTY-ene hvis hovedgrensesnittet på den stasjonære datamaskinen ikke fungerer.

Du går ut av TTY-en ved å trykke og holde inne **CTRL+ALT** samtidig som du trykker på **F7**: Skrivebordet vises igjen. Trykk på **CTRL+ALT+F2** en gang til for å gå tilbake til «tty2». Alt du kjørte i TTY-en, er fremdeles der.

Før du bytter igjen skriver du følgende:

```
exit
```

Deretter trykker du på **CTRL+ALT+F7** for å gå tilbake til skrivebordet. Grunnen til at du må avslutte før du bytter fra TTY-en, er at alle som har tilgang til tastaturet kan bytte til en TTY. Så hvis du fremdeles er logget på, har de tilgang til kontoen din uten å vite passordet ditt.

Gratulerer! Du har tatt de første skrittene for å bli kjent med kommandolinjegrensesnittet til Raspberry Pi OS.

Vedlegg D

Annet referansemateriale

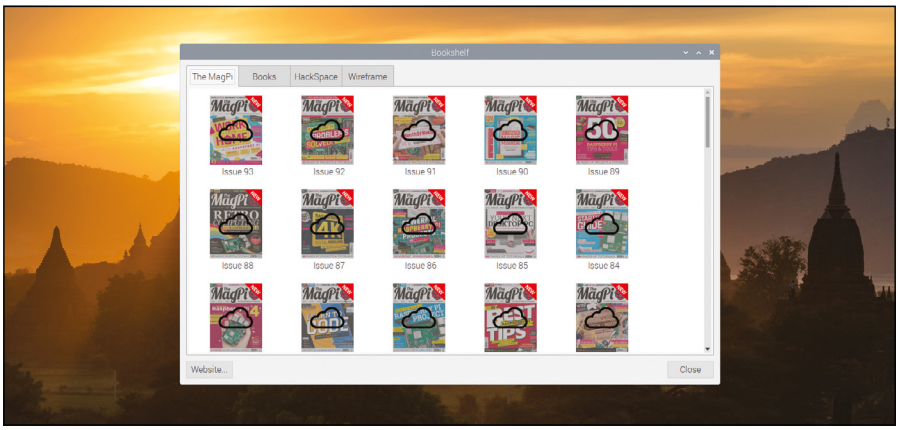
Den offisielle begynnerveiledningen for Raspberry Pi hjelper deg med å komme i gang med Raspberry Pi. Den dekker imidlertid ikke alt du kan gjøre med datamaskinen. Datamaskinene kan brukes til alt fra spill og sensing-programmer til robotikk og kunstig intelligens så Raspberry Pi-fellesskapet er stort og verdensomspennende. Det finnes mye inspirasjon verden rundt.

I dette vedlegget finner du en oversikt over flere kilder som tilbyr prosjektideer, læreplaner og annet materiale som kan være nyttig nå som du har gått gjennom *begynnerveiledningen*.

Bookshelf

Raspberry Menu > Help > Bookshelf

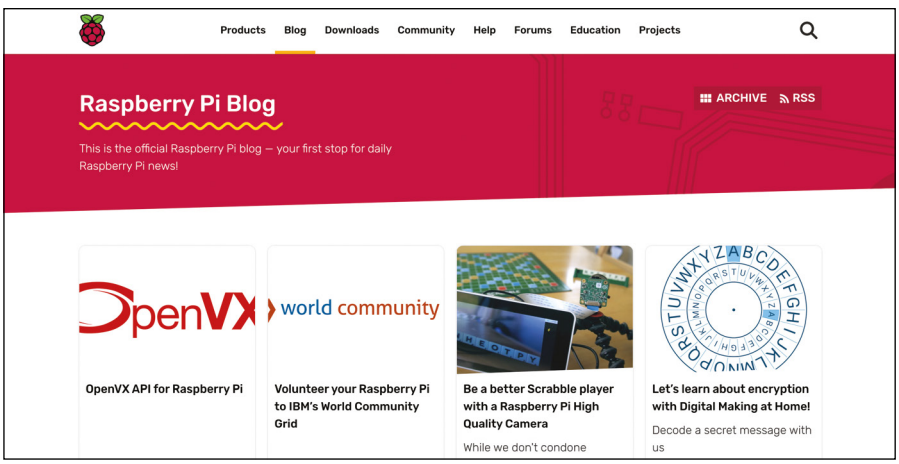
Bookshelf er et program som følger med Raspberry Pi OS. Det kan brukes til å bla gjennom, laste ned og lese digitale versjoner av presseartikler tilknyttet Raspberry Pi, deriblant denne *begynnerveiledningen for Raspberry Pi*. Du kan laste inn programmet ved å klikke på bringebærikonet, velge Help, og deretter klikke på Bookshelf. Der kan du bla gjennom en rekke magasiner og bøker som du kan laste ned helt kostnadsfritt og lese når du vil.



Raspberry Pi-bloggen

rpf.io/blog

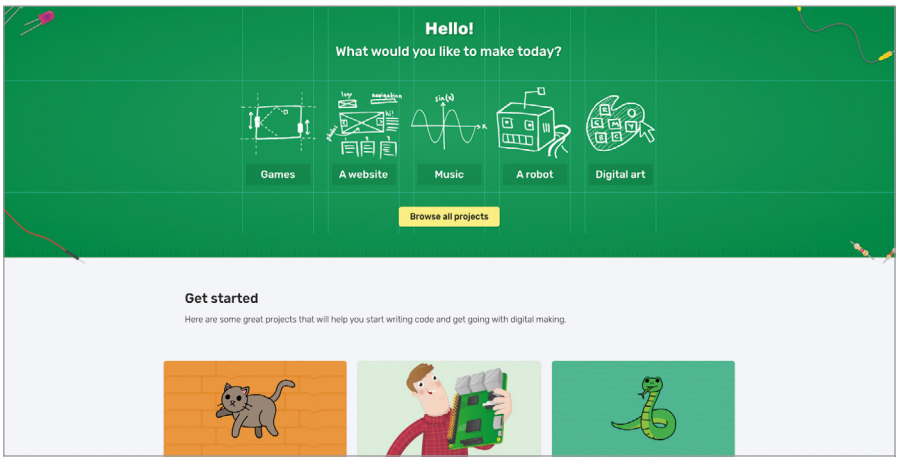
Det første stedet du bør besøke hvis du vil vite siste nytt om Raspberry Pi. Den offisielle bloggen dekker alt fra nye maskinvarelanseringer og pedagogisk materiale til en oversikt over populære samfunnsprosjekter, kampanjer og initiativer. Hvis du vil holde deg oppdatert om alt som gjelder Raspberry Pi, bør du sjekke her.



Raspberry Pi Projects

rpf.io/projects

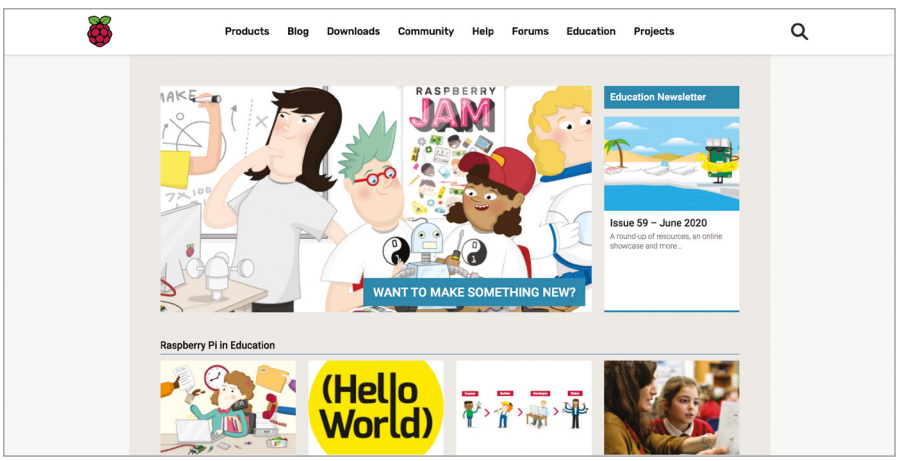
Det offisielle nettstedet Raspberry Pi Projects tilbyr trinnvise prosjektveiledninger i en rekke kategorier, fra å lage spill og musikk til å utvikle egne nettsteder eller Rasperry Pi-drevne roboter. Mesteparten av prosjektene finnes dessuten på en rekke språk og dekker flere vanskelighetsnivåer – alt fra nybegynnere til erfarne utviklere.



Raspberry Pi Education

rpf.io/education

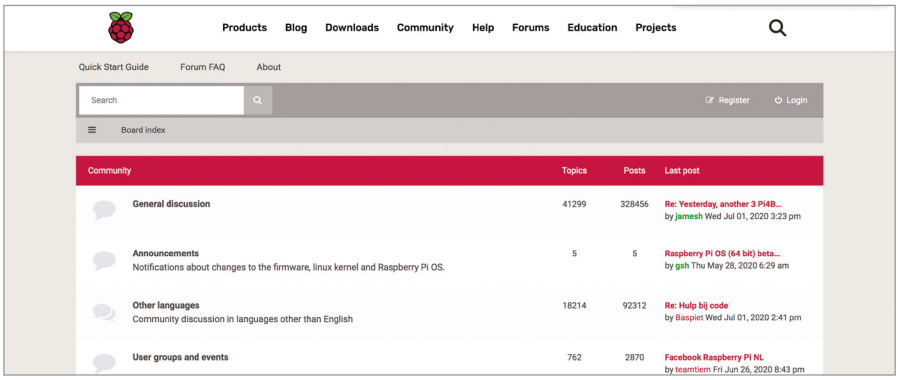
Det offisielle opplæringsnettstedet, Raspberry Pi Education, tilbyr nyhetsbrev, nettbasert opplæring og prosjekter som først og fremst er beregnet på lærere. Nettstedet har også koblinger til andre ressurser, inkludert opplæringsprogrammet Picademy, kodingsklubbene Code Club og Coder Dojo, som drives av frivillige medarbeidere, samt globale Raspberry Jam-arrangementer.



Raspberry Pi-foraer

rpf.io/forums

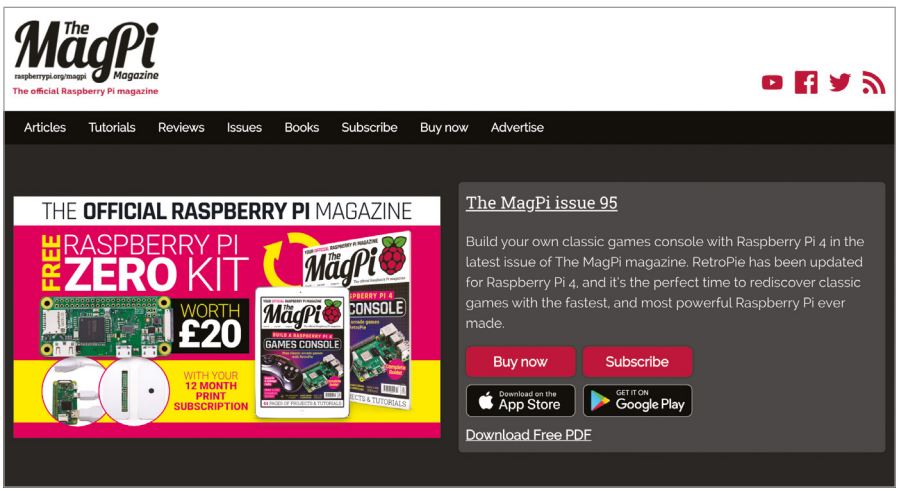
Raspberry Pi-entusiaster kan gå til Raspberry Pi-foraene for å diskutere alt fra nybegynnerproblemer til svært tekniske emner – det finnes til og med et Off-topic-område for generell nettprat som ikke er emnerelatert.



Magasinet The MagPi

magpi.cc

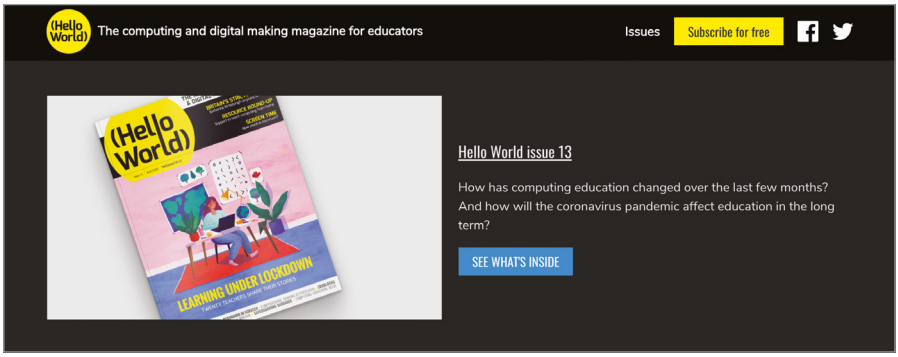
Det offisielle Raspberry Pi-magasinet, The MagPi, er en glanset månedlig publikasjon som inneholder alt fra opplæringsmoduler og veiledninger til anmeldelser og nyheter. Magasinet støttes i større grad av det verdensomspennende Raspberry Pi-fellesskapet. Du finner kopier i alle gode aviskiosker og supermarkeder, men de kan også lastes ned digitalt helt kostnadsfritt under en Creative Commons-lisens. MagPi publiserer også bøker og bokmagasiner om en rekke emner. Du kan kjøpe disse i trykt format eller laste dem ned kostnadsfritt fra nettet.



Magasinet Hello World

helloworld.cc

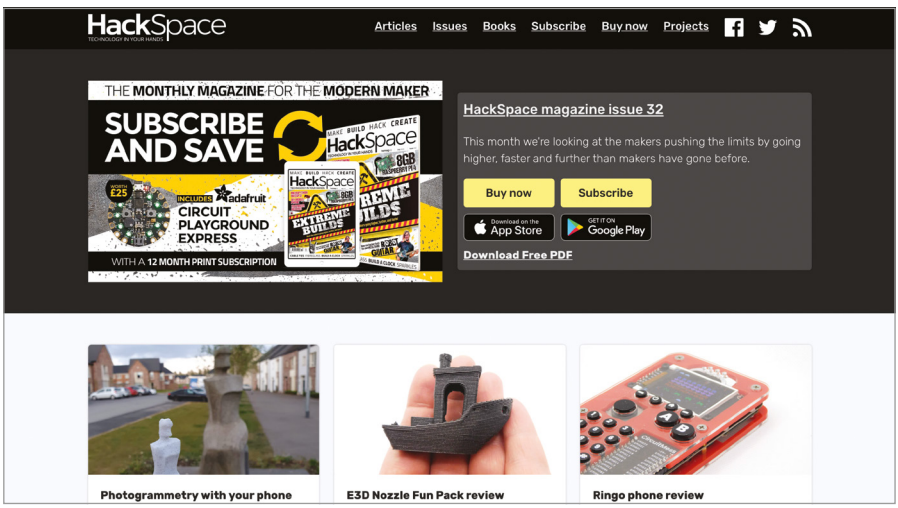
Hello World publiseres tre ganger i året og er kostnadsfritt for lærere, frivillige medarbeidere og bibliotekarer i Storbritannia. Alle andre kan laste ned kostnadsfrie digitale kopier under en Creative Commons-lisens, og abonnementer på den trykte versjonen er kommersielt tilgjengelig.



Magasinet HackSpace

hsmag.cc

HackSpace er rettet mot et bredere publikum enn The MagPi og er spesielt interessert i utviklermiljøet. Magasinet inneholder maskinvare- og programvareomtaler, opplæringsmoduler og intervjuer. Hvis du ønsker å utvide horisonten utover Raspberry Pi, er HackSpace et flott utgangspunkt. Du finner trykte kopier i supermarkeder og aviskiosker, men du kan også laste ned digitale kopier helt kostnadsfritt fra nettet.



Vedlegg E

Verktøyet Raspberry Pi Configuration

Raspberry Pi Configuration Tool er et kraftig verktøy som brukes til å justere mange innstillinger på Raspberry Pi, fra tilgjengelige grensesnitt til programmer som kontrollerer det via et nettverk. Verktøyet kan virke litt overveldende for nybegynnere. Derfor leder dette vedlegget deg gjennom alle innstillingene etter tur, og forklare hvordan de fungerer.

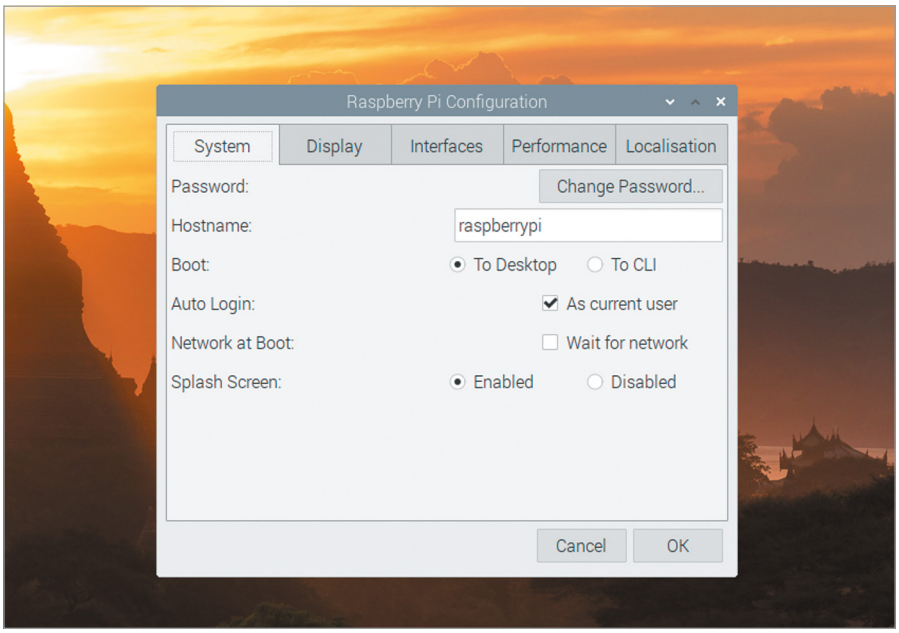
Du kan laste inn Raspberry Pi Configuration Tool fra menyen til bringebærikonet, under kategorien Brukervalg. Det kan også kjøres fra kommandolinjegrensesnittet eller LXTerminal ved å bruke kommandoen **`raspi-config`**. Oppsettene til kommandolinjeversjonen og den grafiske versjonen er forskjellige. Alternativene vises i ulike kategorier avhengig av hvilken versjon du bruker. Dette vedlegget er basert på den grafiske versjonen.

ADVARSEL!

Med mindre du vet at du må endre en bestemt innstilling, er det best å ikke endre på noen innstillinger i Raspberry Pi Configuration Tool. Hvis du legger til ny maskinvare på Raspberry Pi, for eksempel en lyd-HAT eller en Camera Module, følger du veiledningen for å se hvilke innstillinger du må endre. Ellers bør du ikke endre standardinnstillingene.

System-fanen

System-fanen inneholder alternativer som styrer forskjellige systeminnstillinger for Raspberry Pi OS.



- **Passord:** Klikk på knappen «Change password...» for å angi et nytt passord for gjeldende brukerkonto. Som standard er dette Pi-kontoen.

- **Hostname:** Navnet som identifiserer en Raspberry Pi i ulike nettverk. Hvis du har mer enn én Raspberry Pi i samme nettverk, må hver enkelt datamaskin ha et unikt navn.

- **Boot:** Hvis du merker av for «To Desktop» (standardverdien) her, lastes det kjente skrivebordet for Raspberry Pi OS inn. Hvis du merker av for «To CLI», lastes kommandolinjegrensesnittet som beskrives i **Vedlegg C: Kommandogrensesnittet**.

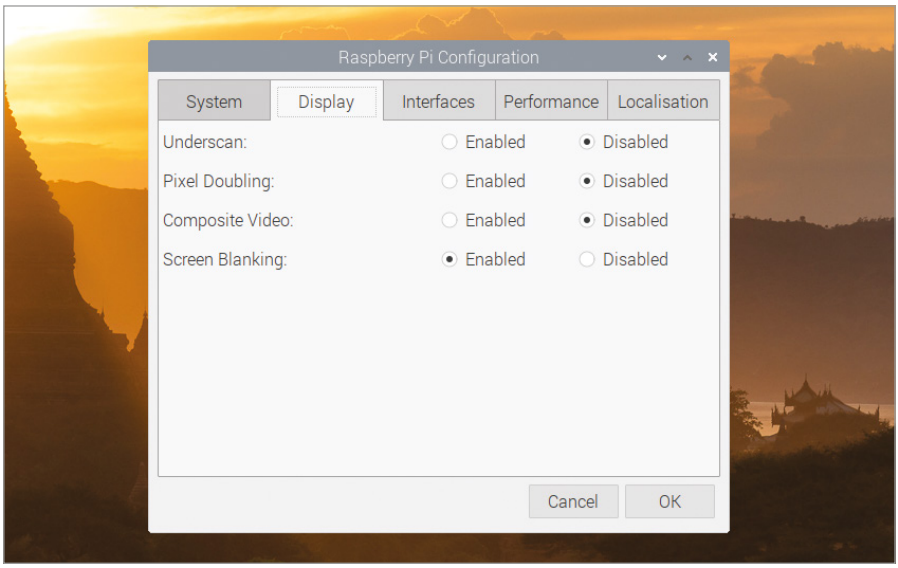
- **Auto Login:** Hvis du merker av for «Log in as user pi» (standardverdien), laster Raspberry Pi OS inn skrivebordet uten at du behøver å skrive inn brukernavn og passord.

- **Network at Boot:** Hvis du merker av for «Wait for network», lastes ikke Raspberry Pi OS inn før det har opprettet en nettverkstilkobling.

- **Splash Screen:** Hvis du merker av for «Enabled» (standardverdien), skjules oppstartsmeldingene til Raspberry Pi OS bak et grafisk velkomstbilde.

Display-fanen

Display-fanen inneholder innstillinger som styrer hvordan skjermen skal vises.



- **Overscan:** Denne innstillingen styrer om videoutdataene på Raspberry Pi skal ha svarte rammer rundt kantene for å kompensere for rammen på mange TV-er. Hvis du ser svarte rammer, merker du av for «Deaktivert». Hvis ikke beholder du innstillingen «Aktivert».

- **Pixel Doubling:** Hvis du bruker en liten skjerm med høy oppløsning, kan du slå på «Pixel Doubling» for å gjøre alle elementene på skjermen større og enklere å se.

- **Composite Video:** Dette styrer kompositttvideoutgangen som er tilgjengelig på den kombinerte lyd-video-kontakten (AV), når den brukes med en TRRS-adapter (tip-ring-ring-sleeve). Hvis du vil bruke kompositttvideoutgangen i stedet for HDMI, merker du av for alternativet «Aktivert». Ellers beholder du innstillingen «Aktivert».

- **Screen Blanking:** Med dette alternativet kan du slå skjermslukking (et tidsavbrudd som slår skjermen av etter et par minutter) av og på.

Fanen Interfaces

Fanen Interfaces inneholder innstillinger som styrer maskinvaregrensesnittene som er tilgjengelige på Raspberry Pi.

- **Camera:** Aktiverer eller deaktiverer CSI-et (Camera Serial Interface) som brukes sammen med Raspberry Pi Camera Module.

- **SSH:** Aktiverer eller deaktiverer SSH-grensesnittet (Secure Shell). Alternativet brukes til å åpne et kommandolinjegrensesnitt på Raspberry Pi fra en annen datamaskin i nettverket ditt ved hjelp av en SSH-klient.

- **VNC:** Aktiverer eller deaktiverer VNC-grensesnittet (Virtual Network Computing). Alternativet brukes til å vise skrivebordet på Raspberry Pi fra en annen datamaskin i nettverket ditt ved hjelp av en a VNC-klient.

- **SPI:** Aktiverer eller deaktiverer SPI-grensesnittet (Serial Peripheral Interface), som brukes til å styre maskinvaretillegg som kobles til GPIO-pinnene.

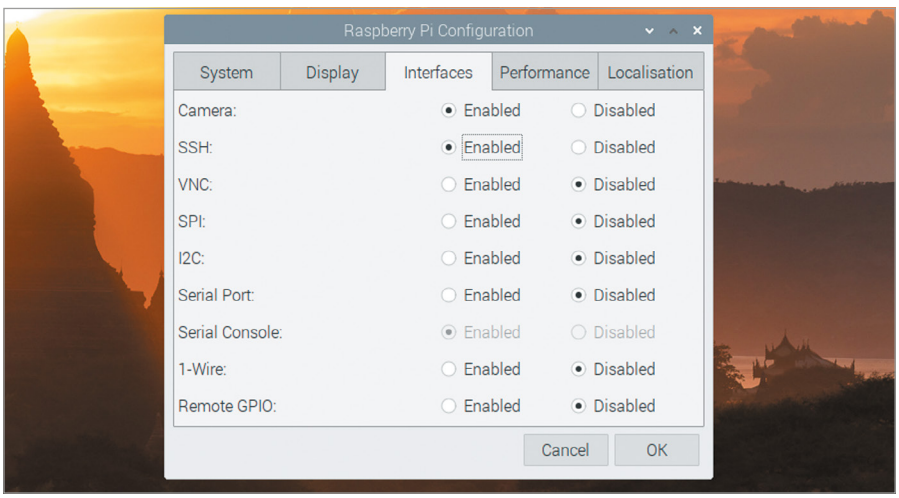
- **I2C:** Aktiverer eller deaktiverer I²C-grensesnittet (Inter-Integrated Circuit), som brukes til å styre maskinvaretillegg som kobles til GPIO-pinnene.

- **Serial Port:** Aktiverer eller deaktiverer Raspberry Pis serielle port, tilgjengelig på GPIO-pinnene.

- **Serial Console:** Aktiverer eller deaktiver seriekonsollen, et kommandolinjegrensesnitt som er tilgjengelig på den serielle porten. Dette alternativet er bare tilgjengelig hvis innstillingen for «Serial Port» ovenfor er satt til «Aktivert».

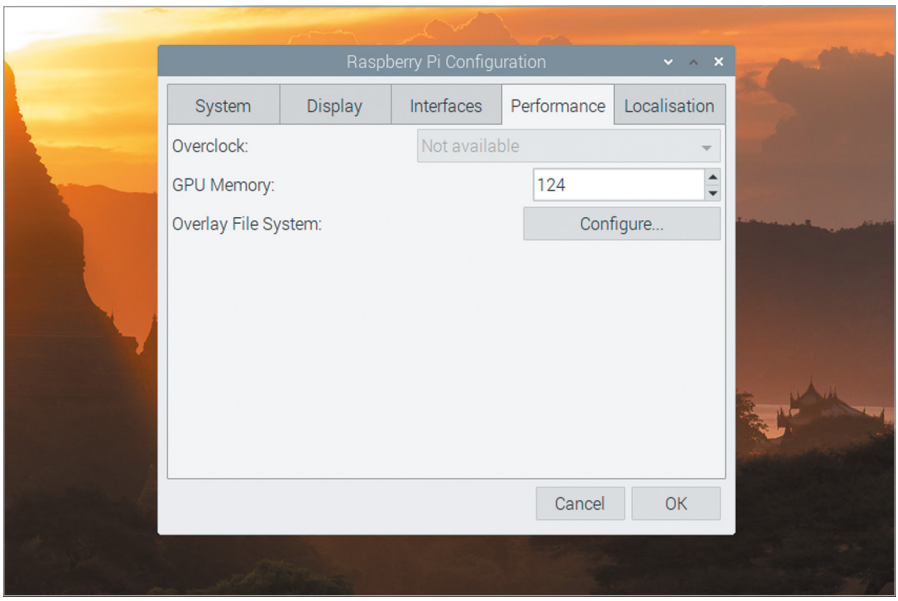
- **1-Wire:** Aktiverer eller deaktiverer 1-Wire-grensesnittet, som brukes til å styre maskinvaretillegg som kobles til GPIO-pinnene.

- **Remote GPIO:** Aktiverer eller deaktiverer en nettverkstjeneste som lar deg styre Raspberry Pis GPIO-pinner fra en annen datamaskin i nettverket ditt ved hjelp av GPIO Zero-biblioteket. Du finner mer informasjon om fjernstyrt GPIO på **gpiozero.readthedocs.io**.



Fanen Performance

Fanen Performance inneholder innstillinger som styrer hvor mye minne som er tilgjengelig, og hvor raskt prosessoren til Raspberry Pi kjører.



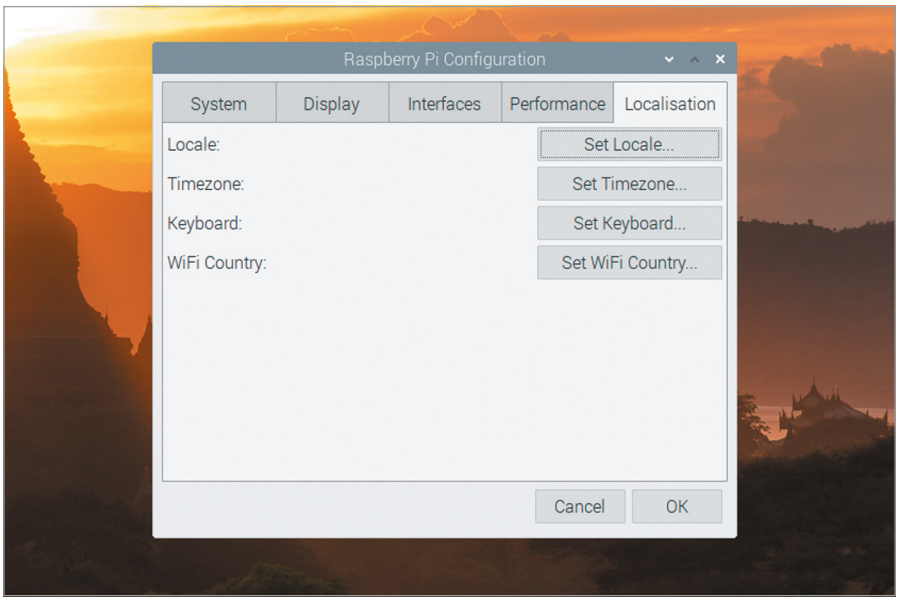
- **Overclock:** Lar deg velge mellom en rekke innstillinger som øker ytelsen til Raspberry Pi på bekostning av økt strømforbruk, varmeproduksjon og mulig redusert total levetid. Ikke tilgjengelig på alle modeller av Raspberry Pi.

- **GPU Memory:** Lar deg stille inn mengden minne som skal være reservert for bruk av Raspberry Pis grafikkprosessor (GPU). Verdier som er høyere enn standardverdien, kan forbedre ytelsen ved kompliserte 3D-gjengivelser og generelle GPU-oppgaver (GPGPU). Men dette reduserer hvor mye minne som er tilgjengelig for Raspberry Pi OS. Lavere verdier kan forbedre ytelsen ved minneintensive oppgaver på bekostning av 3D-gjengivelse. Kameraet og utvalgte videoavspillingsfunksjoner blir tregere eller utilgjengelige.

- **Overlay File System:** Lar deg låse Raspberry Pis filsystem, slik at endringer bare gjøres på en virtuell RAM-disk i stedet for at de skrives til mircoSD-kortet. Det betyr at du går tilbake til en ren tilstand når du starter på nytt.

Fanen Localisation

Fanen Localisation inneholder innstillinger som styrer hvilket område Raspberry Pi skal operere i, inkludert innstillinger for tastaturoppsett.



- **Locale:** Brukes til å velge nasjonale innstillinger, det vil si en systeminnstilling som omfatter språk, land og tegnsett. Merk: Når du endrer språk her, endres bare det viste språket i programmer som er lokalisert.

- **Timezone:** Brukes til å velge den regionale tidssonen ved å velge et område i verden angitt av den nærmeste storbyen. Hvis Raspberry Pi er tilkoblet nettverket, men klokken viser feil tid, skyldes det vanligvis at du har valgt feil tidssone.

- **Keyboard:** Brukes til å velge tastaturets type, språk og oppsett. Hvis tastaturet skriver feil bokstaver eller symboler, kan du endre innstillingene her.

- **WiFi Country:** Brukes til å angi land for radioreguleringsformål. Pass på at du velger landet der du bruker Raspberry Pi. Hvis du velger et annet land, kan det bli vanskelig å koble til nærliggende trådløse tilgangspunkter (Wi-Fi) eller utgjøre brudd på kringkastingsloven. Du må angi et land for å kunne bruke WiFi-radioen.

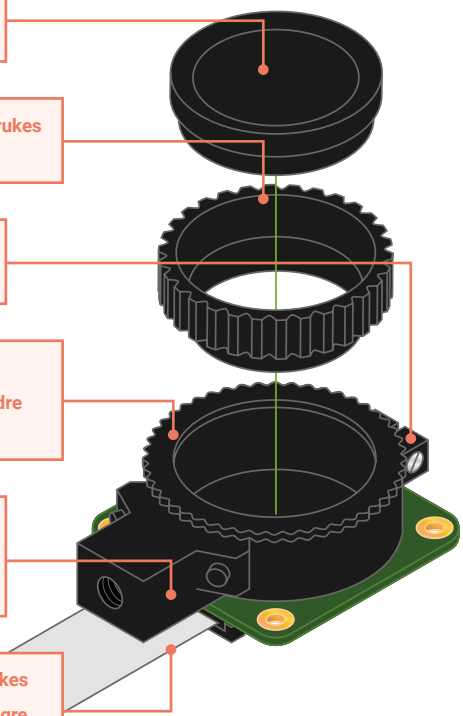
Vedlegg F

Oppsett av High Quality Camera

Komponenten High Quality Camera (forkortet til HQ-kamera) kan ta bilder med høyere oppløsning enn vår standard Camera Module. I motsetning til sistnevnte har ikke HQ-kameraet et forhåndsmontert objektiv. Du kan bruke det med alle standard C- eller CS-objektiver. Du kan kjøpe 6 mm og 16 mm objektiver sammen med kameraet for å gjøre det enklere å komme i gang.

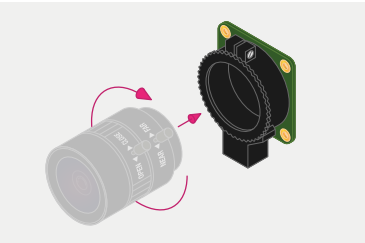
6 mm C-objektiv

Et rimelige 6 mm objektiv er tilgjengelig for HQ-kameraet. Dette objektivet er egnet for vanlig fotografering. Objektivet kan også brukes til makrofotografering da det kan fokusere på nærobjekter.



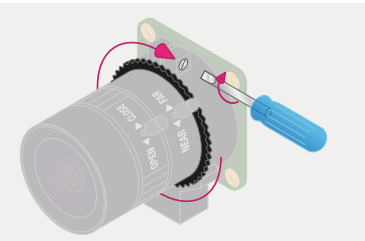
01 Montere objektivet

Det 6 mm objektivet er en CS-enhet. Derfor trenger du ikke å bruke C-CS-adapterringen (se diagrammet ovenfor). Det fokuserer ikke riktig hvis adapteren er montert, så du må fjerne den om nødvendig. Deretter dreier du objektivet med klokken til det sitter godt i justeringsringen for bakfokus.

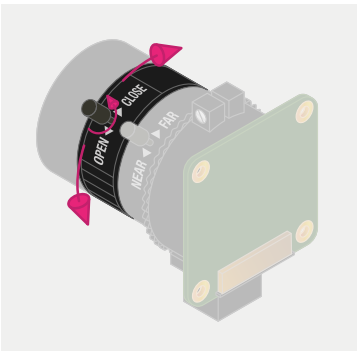


02 Justeringsring og låseskrue for bakfokus

Justeringsringen for bakfokus skal skrus helt inn for å få kortest mulig brennvidde. Bruk låseskruen for bakfokus til å sikre at den ikke endrer posisjon når du justerer blenderåpning og fokus.

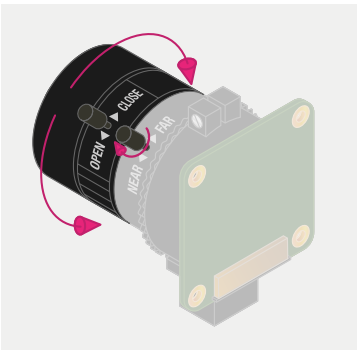


03 Blenderåpning



Når du skal justere blenderåpningen, holder du kameraet med objektivet vendt bort fra deg. Drei på den midtre ringen mens du holder godt tak i den ytre ringen, dvs. den som er lengst borte fra kameraet. Drei den med klokken for å lukke blenderåpningen og redusere lysstyrken i bildet. Drei den mot klokken for å åpne blenderåpningen. Når du er fornøyd med lysstyrken, strammer du til skruen på siden av objektivet for å låse blenderåpningen.

04 Fokus

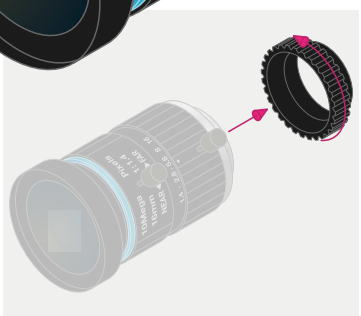


Først låser du på plass den indre fokusringen, kalt «NEAR ◀▶ FAR», ved å stramme til den relevante skruen. Deretter holder du kameraet med objektivet vendt bort fra deg. Hold de to ytre ringene på objektivet og drei begge med klokken til bildet er i fokus. Det krever kanskje fire eller fem fulle omdreininger. Når du skal justere fokus, dreier du de to ytre ringene med klokken for å fokusere på et nærobjekt. Drei dem med klokken for å fokusere på et fjernobjekt. Du må eventuelt justere blenderåpningen igjen etterpå.

16 mm C-objektiv

Det 16 mm objektivet gir deg bilder av høyere kvalitet enn det 6 mm objektivet. Det har en smal synsvinkel som er bedre egnet for å vise fjernobjekter.

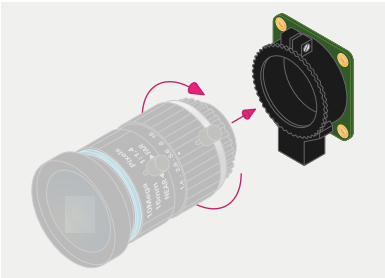
01 Montere C-CS-adapteren



Kontroller at C-CS-adapteren som følger med HQ-kameraet er montert på det 16 mm objektivet. Objektivet er en C-enhet, så det har litt lengre bakfokus enn 6 mm objektivet. Derfor må du bruke adapteren sammen med dette objektivet.

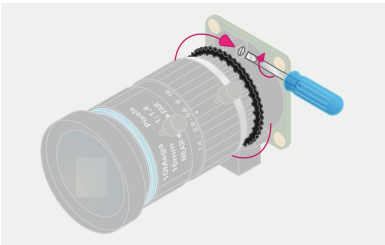
02 Montere objektivet på kameraet

Drei 16 mm objektivet og C-CS-adapteren med klokken til det sitter godt i justeringsringen for bakfokus.



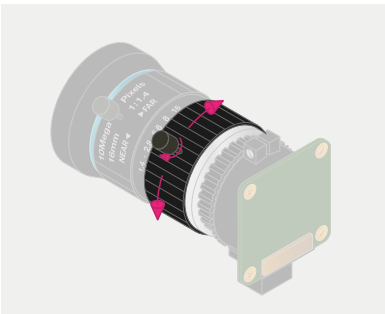
03 Justeringsring og låseskrue for bakfokus

Justeringsringen for bakfokus skal skrus helt inn. Bruk låseskruen for bakfokus til å sikre at den ikke endrer posisjon når du justerer blenderåpning og fokus.



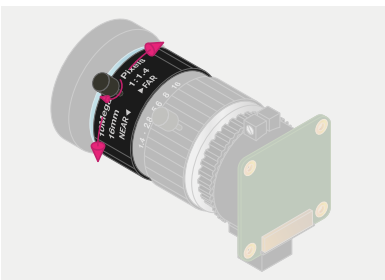
04 Blenderåpning

Når du skal justere blenderåpningen, holder du kameraet med objektivet vendt bort fra deg. Drei den indre ringen, nærmest kameraet, mens du holder kameraet stødig. Drei den med klokken for å lukke blenderåpningen og redusere lysstyrken i bildet. Drei den mot klokken for å åpne blenderåpningen. Når du er fornøyd med lysstyrken, strammer du til skruen på siden av objektivet for å låse blenderåpningen.



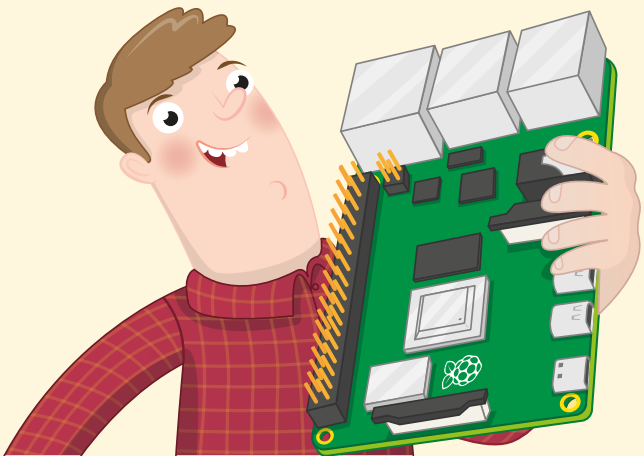
05 Fokus

Når du skal justere fokus, holder du kameraet med objektivet vendt bort fra deg. Drei fokusringer, som kalles «NEAR ◀▶ FAR» mot klokken for å fokusere på et nærobjekt. Drei den med klokken for å fokusere på et fjernt objekt. Du må eventuelt justere blenderåpningen igjen etterpå.



Vedlegg G

Spesifikasjoner for Raspberry Pi

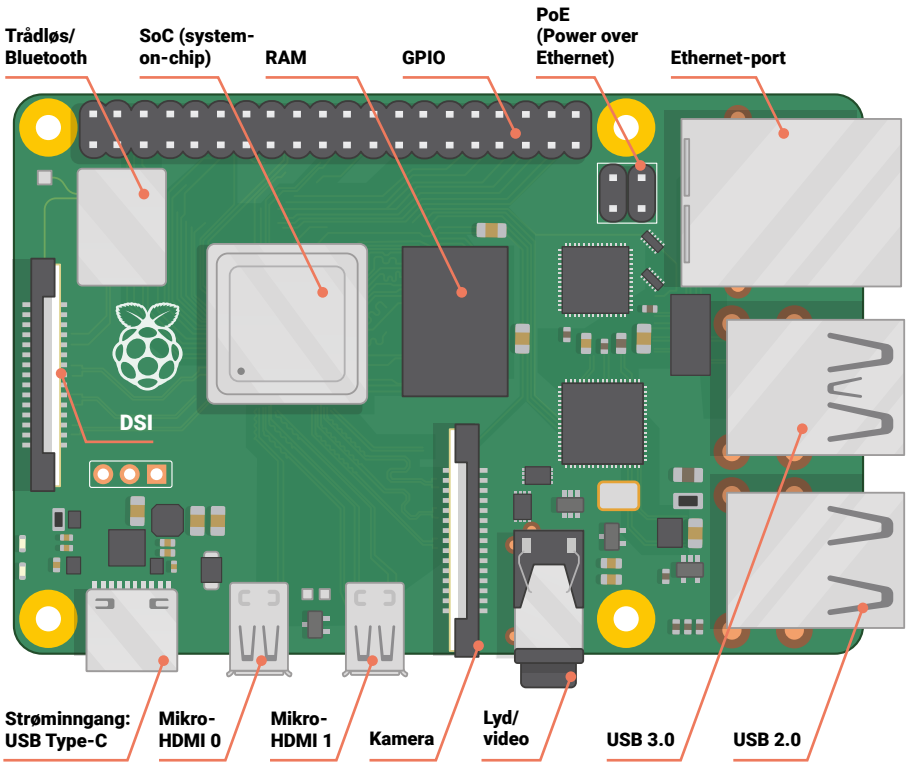


De forskjellige komponentene og egenskapene til en datamaskin er kjent som spesifikasjoner. Når du vil sammenligne to datamaskiner, sjekker du spesifikasjonene. Til å begynne med kan spesifikasjonene virke forvirrende da de er svært tekniske. Du trenger ikke å kjenne til spesifikasjonene for å kunne bruke en Raspberry Pi, men vi har inkludert dem for de som er litt mernysgjerrige.

Raspberry Pi 4 Model B og Raspberry Pi 400s SoC (system-on-chip) er en Broadcom BCM2711B0. Dette er trykt på metalldekselet (på Raspberry Pi 4). Den har fire kjerner med en 64-biters ARM Cortex-A72 prosessorenhet (CPU), som hver kjører ved 1,5 GHz eller 1,8 GHz (1,5 eller 1,8 tusen millioner sykluser per sekund), og en Broadcom VideoCore VI (Six) grafikkbehandlingsenhet (GPU), som kjører ved 500 MHz (500 millioner sykluser per sekund) for videoer og for 3D-gjengivelser som for eksempel spill.

SoC er koblet til et 2 GB, 4 GB eller 8 GB (to fire eller åtte tusen millioner byte) – 4 GB på Raspberry Pi 400 – med LPDDR4 (Low-Power Double-Data-Rate 4) RAM (Random Access Memory), som kjører ved 3200 MHz (tre tusen to hundre millioner sykluser per sekund). Dette minnet deles mellom CPU-en og GPU-en. microSD-kortsporet støtter opptil 512 GB (512 tusen millioner byte) minne.

Ethernet-port støtter opptil GB-tilkoblinger (1000 Mbps, 1000-Base-T), mens radioen støtter 802.11ac Wi-Fi-tilkoblinger som kjører på 2,4 GHz and 5 GHz frekvensbånd, Bluetooth 5.0, and BLE-tilkoblinger (Bluetooth Low Energy).



Oversikt over spesifikasjonene til Raspberry Pi 4:

- **CPU:** 64-biters firekjerners ARM Cortex-A72, 1,5 GHz
- **GPU:** VideoCore VI, 500 MHz
- **RAM:** 1 GB, 2 GB eller 4 GB LPDDR4
- **Nettverk:** Gigabit Ethernet, dual-band 802.11ac, Bluetooth 5.0, Bluetooth Low Energy
- **Lyd- og videoutganger:** 3,5 mm analog AV-kontakt, 2 × micro-HDMI 2.0
- **Ekstern tilkobling:** 2 × USB 2.0-porter, 2 × USB 3.0-porter, CSI (Camera Serial Interface), DSI (Display Serial Interface)
- **Minne:** microSD, opptil 512 GB
- **Strøm:** 5 volt ved 3 ampere via USB Type-C
- **Tilleggsutstyr:** 40-pinners GPIO-pinnerekke, PoE-kompatibilitet (med ekstra maskinvare)

Spesifikasjoner for Raspberry Pi 400:

- **CPU:** 64-biters firekjerners ARM Cortex-A72, 1,8 GHz
- **GPU:** VideoCore VI, 500 MHz
- **RAM:** 4 GB LPDDR4
- **Nettverk:** Gigabit Ethernet, dual-band 802.11ac, Bluetooth 5.0, Bluetooth Low Energy
- **Lyd- og videoutganger:** 2 × micro-HDMI 2.0
- **Ekstern tilkobling:** 1 × USB 2.0-port, 2 × USB 3.0-porter
- **Minne:** microSD, opptil 512 GB (16 GB leveres)
- **Strøm:** 5 volt ved 3 ampere via USB Type-C
- **Tilleggsutstyr:** 40-pinners GPIO-pinnerekke

Vedlegg H

Sikkerhet og brukerveiledning for Raspberry Pi

Raspberry Pi

Utviklet og distribuert av
Raspberry Pi Trading Ltd
Maurice Wilkes Building
Cowley Road
Cambridge
CB4 0DS
Storbritannia
www.raspberrypi.org

Raspberry Pi – forskriftsmessig samsvars- og sikkerhetsinformasjon

Raspberry Pi 4 Model B
FCC-ID: 2ABCB-RPI4B
IC-ID: 20953-RPI4B

Raspberry Pi 400
FCC-ID: 2ABCB-RPI400
IC-ID: 20953-RPI400

VIKTIG: Se installasjonsveiledningen før du kobler til strømforsyningen, på **www.raspberrypi.org/safety**

ADVARSEL: Kreft og reproduksjonsskade – www.P65Warnings.ca.gov.

Du finner forskriftsinformasjon og sertifikater på **www.raspberrypi.org/compliance**

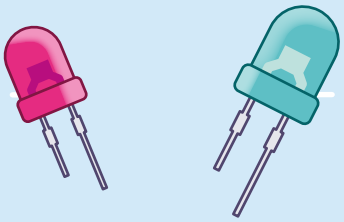
IFETEL: 2019LAB-ANCE4957
Sertifikatnummer for Raspberry Pi 4 Model B.

TRA-registreringsnr.
ER73381/19
Sertifikatnummer for Raspberry Pi 4 Model B.

CCAO19LP1120T2
Sertifikatnummer for Raspberry Pi 4 Model B.

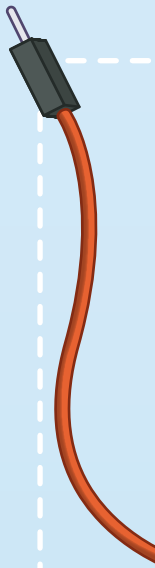
NTC
Godkjent type:
NR.: ESD-GEC-1920098C
Sertifikatnummer for Raspberry Pi 4 Model B.

TA-2019/750 GODKJENT
Sertifikatnummer for Raspberry Pi 4 Model B.



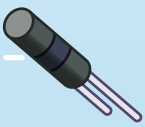
DEN OFFISIELLE
begynnerveiledningen for
Raspberry Pi

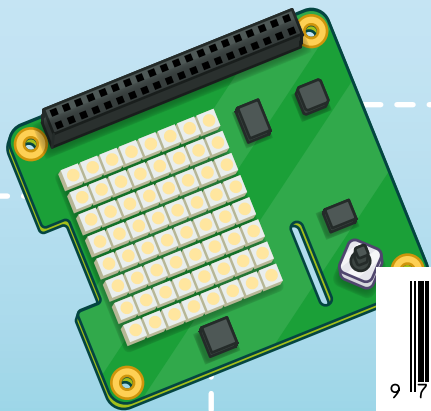
Raspberry Pi er laget i Storbritannia. Det er en liten, smart datamaskin som er fullpakket med potensiale. Raspberry Pi er basert på den samme teknologien som brukes i smarttelefoner. Den er utviklet for å gjøre det enklere å lære koding, finne ut hvordan datamaskiner fungerer, og lage dine egne fantastiske ting. Denne veiledningen viser deg hvor enkelt det er å komme igang.

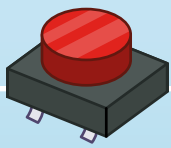


Du lærer å

- konfigurere Raspberry Pi, installere operativsystemet og komme i gang med den bruksklare datamaskinen
- starte kodingsprosjekter med trinnvise veiledninger som bruker programmeringsspråkene Scratch 3 og Python
- eksperimentere med å koble til elektroniske komponenter og ha det gøy med å lage fantastiske prosjekter







raspberrypi.org

9 781912 047826

Raspberry Pi PRESS