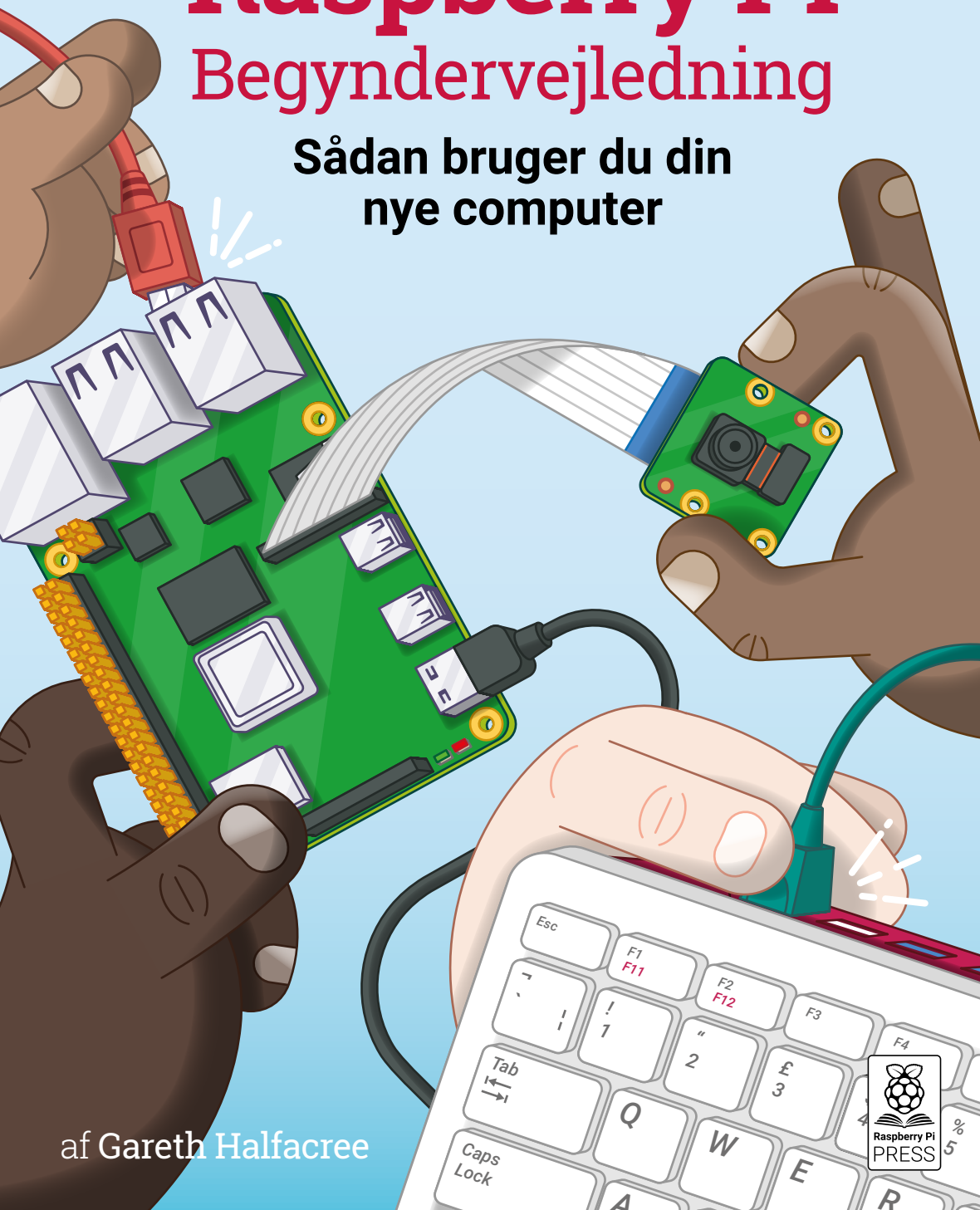
4. udgave

FULDT OPDATERET TIL **RASPBERRY PI 400**

DEN OFFICIELLE Raspberry Pi Begyndervejledning

Sådan bruger du din nye computer



af Gareth Halfacree

DEN OFFICIELLE
Raspberry Pi
Begyndervejledning
Sådan bruger du din nye computer

Udgivet første gang i 2021 af Raspberry Pi Trading Ltd, Maurice Wilkes Building,
St. John's Innovation Park, Cowley Road, Cambridge, CB4 0DS

Forlagsdirektør: Russell Barnes • Redaktør: Phil King
Design: Critical Media • Illustrationer: Sam Alder
CEO: Eben Upton

ISBN: 978-1-912047-83-3

Velkommen til den officielle begyndervejledning til Raspberry Pi

Vi tror, du vil elske Raspberry Pi. Uanset hvilken model du har – en standard Raspberry Pi eller den nye Raspberry Pi 400 med integreret tastatur – kan denne overkommelige computer bruges til at lære programmering, bygge robotter og skabe alle mulige underlige og vidunderlige projekter.

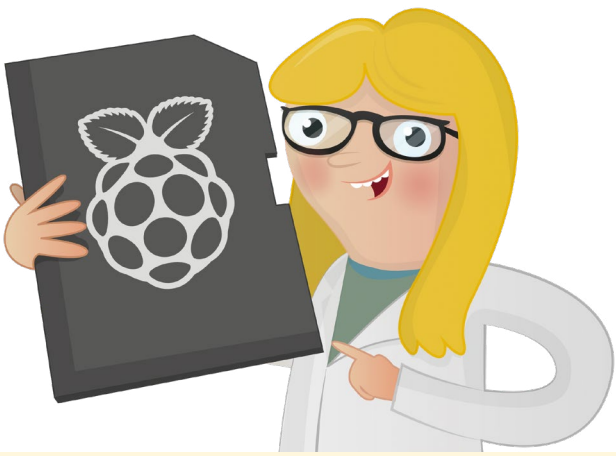
Raspberry Pi er i stand til at gøre alle de ting, du forventer af en computer – lige fra at surfe på internettet og spille spil til at se film og lytte til musik. Men Raspberry Pi er meget mere end en moderne computer.

Med en Raspberry Pi kan du komme helt ind i hjertet af en computer. Du kan konfigurere dit eget operativsystem og kan forbinde ledninger og kredsløb direkte til dens GPIO-ben. Den var oprindeligt designet til at lære unge mennesker at programmere i sprog som Scratch og Python, og alle de større programmeringssprog er inkluderet i det officielle operativsystem.

Verden har brug for programmører mere end nogensinde, og Raspberry Pi har skabt en interesse for datalogi og teknologi i en ny generation.

Folk i alle aldre bruger Raspberry Pi til at skabe spændende projekter: Alt fra retro-spilkonsoller til internetforbundne vejrstationer.

Så hvis du vil lave spil, bygge robotter eller lave en række fantastiske projekter, så vil denne bog hjælpe dig i gang.



Om forfatteren

Gareth Halfacree er freelance teknologijournalist, forfatter og tidligere systemadministrator i uddannelsessektoren. Med en passion for open source-software og hardware var han en af de første brugere af Raspberry Pi-platformen og har skrevet adskillige publikationer om dens muligheder og fleksibilitet. Han kan findes på Twitter som **@ghalfacree** eller via hans hjemmeside på **freelance.halfacree.co.uk**.

Indhold

Kapitel 1

Lær din Raspberry Pi at kende

Bliv fuldt fortrolig med din nye computer i kreditkortstørrelse ved at få en guidet beskrivelse af Raspberry Pi. Oplev dens mange komponenter, og hvad de gør

Raspberry Pi er en utrolig enhed: En fuldt funktionel computer i en lille og billig pakke. Uanset om du er på udkig efter en enhed, som du kan bruge til at surfe på nettet eller spille spil, gerne vil lære at skrive dine egne programmer eller ønsker at lave dine egne kredsløb og fysiske enheder, så vil Raspberry Pi – og de fantastiske tilknyttede grupper – støtte dig hele vejen.

Raspberry Pi er kendt som en *enkeltkortcomputer*, hvilket betyder lige præcis det: Det er en computer, ligesom en stationær, bærbar computer eller smartphone, men baseret på et enkelt *printkort*. Ligesom de fleste andre enkeltkortcomputere er Raspberry Pi lille – stort set samme størrelse som et kreditkort – men det betyder ikke, at den ikke er avanceret: En Raspberry Pi kan alt, hvad en større og kraftigere computer kan, dog ikke nødvendigvis lige så hurtigt.

Raspberry Pi-familien opstod som et ønske om at tilskynde til mere praktisk computeruddannelse over hele verden. Dens udviklere, der slog sig sammen for at danne non-profit-organisationen Raspberry Pi Foundation, havde ikke nogen idé om, at den skulle blive så populær: De få tusinder, som blev bygget i 2012 for at lodde stemningen, blev straks udsolgt, og millioner af eksemplarer er leveret over hele verden i årene siden. Disse computere har fundet vej til hjem, klasseværelser, kontorer, datacentre, fabrikker og endda selvstyrende både og balloner i rummet.

Forskellige modeller af Raspberry Pi er blevet frigivet siden den oprindelige model B, som hver især har forbedrede specifikationer eller funktioner, der er specifikke for en bestemt type brug. Raspberry Pi Zero-familien er for eksempel en lille version af Raspberry Pi i fuld størrelse, blot uden et par funktioner – især de mange USB-porte og kablet netværksport – til fordel for et betydeligt mindre layout og reduceret strømbehov.

Alle Raspberry Pi-modeller har dog en ting til fælles: De er *kompatible*, hvilket betyder, at software skrevet til en model også kører på de andre modeller. Det er endda muligt at tage den nyeste version af Raspberry Pi's operativsystem og køre det på en original pre-launch Model B prototype. Det kører langsommere, det er sandt, men det kører dog.

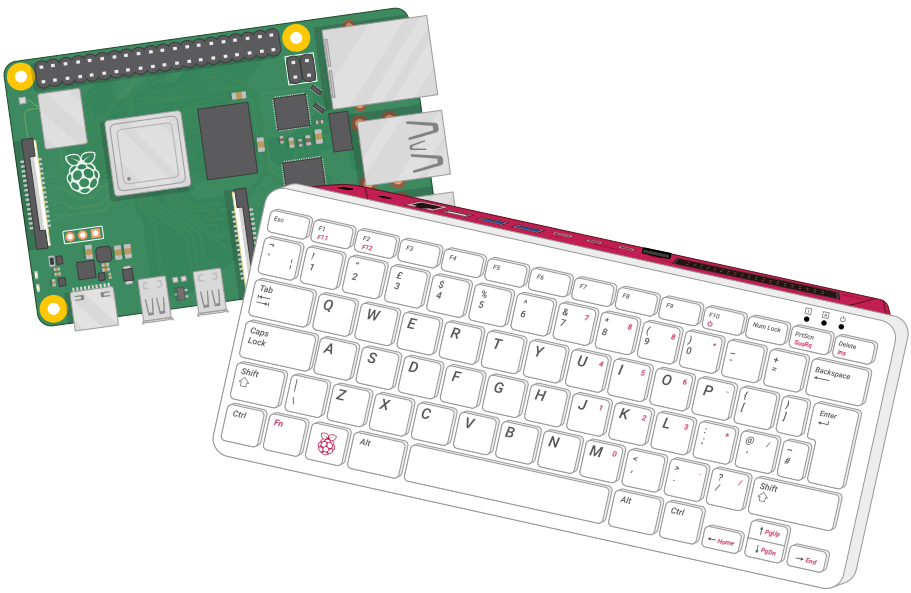
I denne bog lærer du om Raspberry Pi 4 Model B og Raspberry Pi 400, de nyeste og mest avancerede versioner af Raspberry Pi. Det, du lærer, kan dog let anvendes på andre modeller i Raspberry Pi-familien, så der er ingen grund til bekymring, hvis du bruger en anden version.

RASPBERRY PI 400

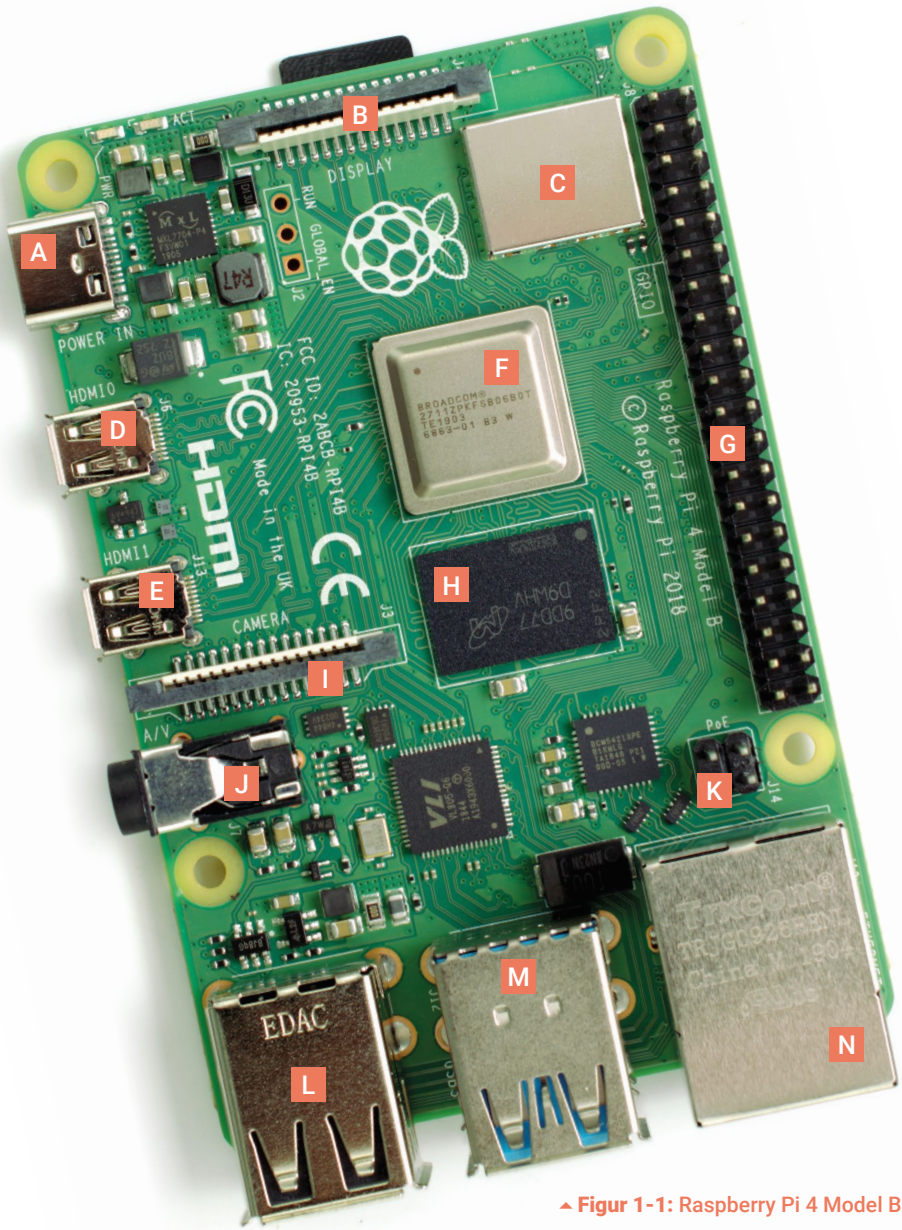
Hvis du har en Raspberry Pi 400, er printkortet indbygget i tastaturet. Læs videre for at lære om alle de komponenter, der får Raspberry Pi til at fungere, eller spring til side 20 for en rundvisning i din enhed.

En guidet beskrivelse af Raspberry Pi

I modsætning til en traditionel computer, der skjuler sin indre funktion i en kasse, har en standard Raspberry Pi alle sine komponenter, porte og funktioner til fuld skue – selvom du godt kan købe en kasse for at få ekstra beskyttelse, hvis du foretrækker det. Dette gør den til et godt værktøj til at lære om, hvad de forskellige dele af en computer gør, og det gør det også nemt at lære, hvad der sidder hvor, når det er tid til at tilslutte det forskellige ekstraudstyr – kendt som *perifert udstyr* – som du har brug for til at komme i gang.



- A USB-C strøm ind
- B DSI-skærmport
- C Trådløs/Bluetooth
- D Micro-HDMI 0
- E Micro-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



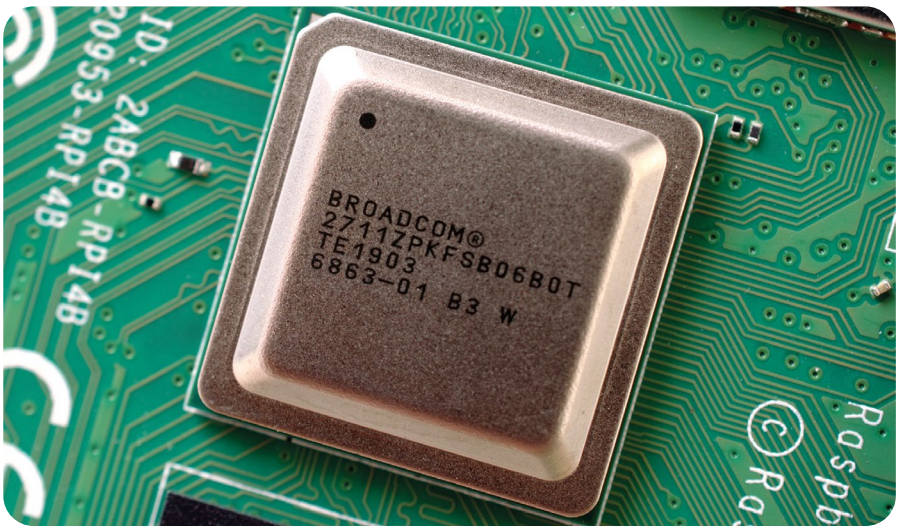
▲ **Figur 1-1:** Raspberry Pi 4 Model B

Figur 1-1 viser en Raspberry Pi 4 Model B set ovenfra. Når du bruger en Raspberry Pi med denne bog, så prøv at holde den vendt på samme måde som på billedet; hvis det er vendt rundt, kan det være forvirrende, når det f.eks. handler om at bruge ting som GPIO-headeren (nærmere beskrevet i **Kapitel 6, Fysisk programmering med Scratch og Python**).

Selvom det kan se ud, som om der er meget pakket ind i det lille kort, er en Raspberry Pi meget enkel at forstå – startende med dets *komponenter*, selve indmaden, der får enheden til at virke.

Raspberry Pi's komponenter

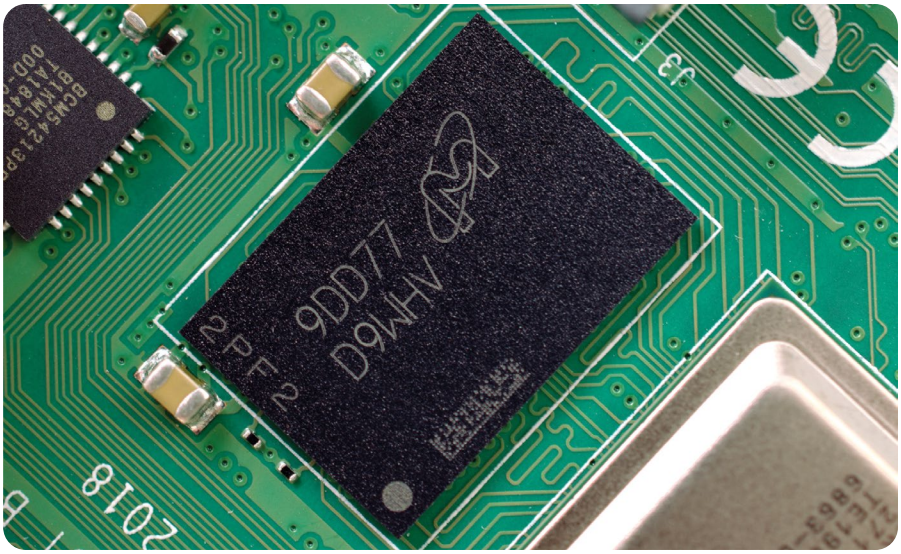
Som enhver anden computer består Raspberry Pi af forskellige komponenter, som hver især har en rolle at spille for at få det hele til at fungere. Den første og uden tvivl den vigtigste af disse findes lige over midten på oversiden af kortet (**Figur 1-2**), dækket af en metalhætte: *System-on-chip* (SoC).



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

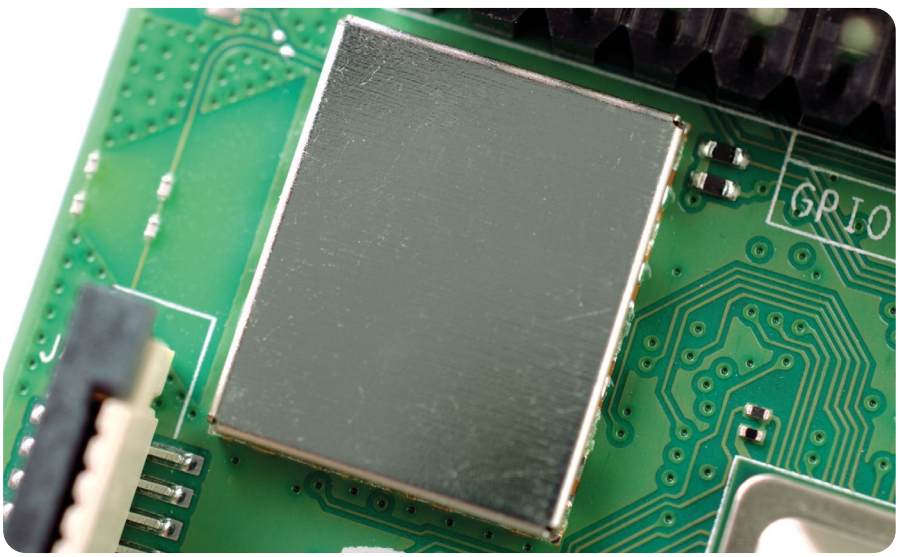
Navnet system-on-chip er en god indikator for, hvad du ville finde, hvis du åbnede metaldækslet: en siliciumchip, kendt som et *integreret kredsløb*, der indeholder hovedparten af Raspberry Pi's system. Dette inkluderer den *centrale processorenhed* (CPU), der almindeligvis betragtes som en computers "hjerne", og *grafikbehandlingsenheden* (GPU), der håndterer den visuelle side af tingene.

En hjerne er dog ikke meget værd uden hukommelse, og lige ved siden af SoC finder du lige præcis det: En anden chip, der ligner en lille, sort, plastisk firkant (**Figur 1-3**, på næste side). Dette er Raspberry Pi's *RAM (random access memory)*. Når du arbejder på Raspberry Pi, er det RAM'en, der indeholder det, du laver; kun når du gemmer dit arbejde, bliver det skrevet til microSD-kortet. Tilsammen udgør disse komponenter Raspberry Pi's flygtige og ikke-flygtige hukommelser: Den flygtige RAM mister sit indhold, når Raspberry Pi slukkes, mens det ikke-flygtige microSD-kort beholder dets indhold.



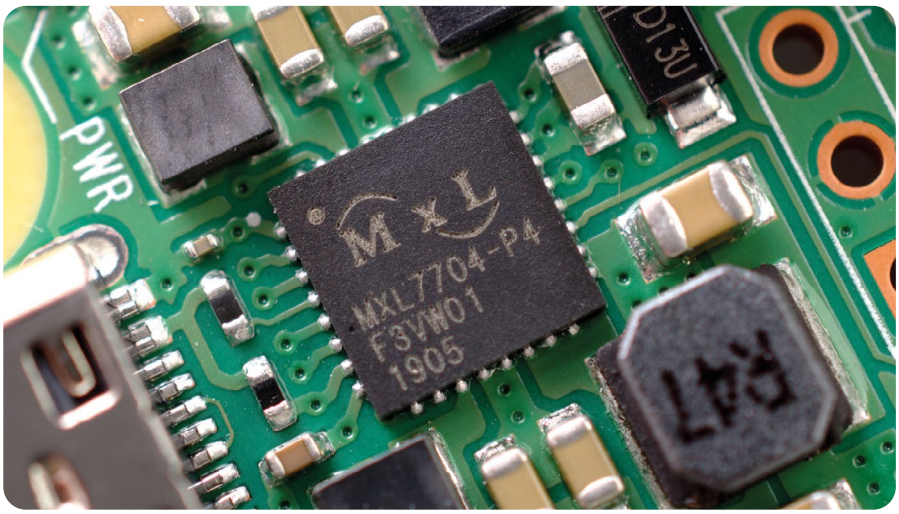
▲ **Figur 1-3:** Raspberry Pi's RAM (random access memory)

Øverst til højre på tavlen finder du et andet metallåg (**Figur 1-4**), der dækker *radioen*, den komponent, der giver Raspberry Pi mulighed for at kommunikere med enheder trådløst. Selve radioen fungerer som to hovedkomponenter, dvs. en *WiFi-radio* til tilslutning til computernetværk; og en *Bluetooth-radio* til tilslutning til perifert udstyr som mus og til afsendelse af data til eller modtagelse af data fra nærliggende smarte enheder som sensorer eller smartphones.



▲ **Figur 1-4:** Raspberry Pi's radiomodul

En anden sort, plastbelagt chip kan ses ved nederste kant af kortet lige bag det midterste sæt USB-porte. Dette er *USB-controlleren*, som er ansvarlig for at køre de fire USB-porte. Ved siden af den er der en endnu mindre chip, *netværkscontrolleren*, som håndterer Raspberry Pi's Ethernet-netværksport. Og endelig er der en sort chip, mindre end de øvrige, som kan ses lidt over USB-C-strømstikket øverst til venstre på kortet (**Figur 1-5**); dette er kendt som et *strømstyringsintegreret kredsløb (PMIC)*, der tager sig af at vende strømmen, som kommer ind fra micro-USB-porten, til den strøm, Raspberry Pi har brug for til at køre.



▲ **Figur 1-5:** Raspberry Pi's strømstyringsintegrerede kredsløb (PMIC)

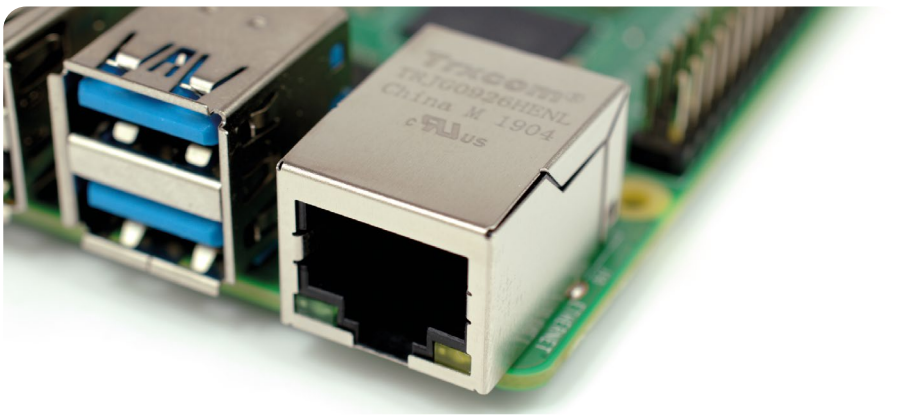
Bare rolig, hvis det hele virker en smule overvældende: Du behøver ikke vide, hvad hver enkelt komponent er, eller hvor du finder den på tavlen for at kunne bruge Raspberry Pi.

Raspberry Pi's porte

Raspberry Pi har en række porte, der starter med fire *USB-porte (Universal Serial Bus)* (**Figur 1-6**) mod midten og i højre side af den nederste kant. Disse porte giver dig mulighed for at forbinde alle USB-kompatible enheder til Raspberry Pi, lige fra tastaturer og mus til digitale kameraer og flashdrev. Teknisk set er der to typer USB-porte: Dem med sorte dele indeni er USB 2.0-porte, der er baseret på version 2 af Universal Serial Bus-standarden; dem med blå dele er hurtigere USB 3.0-porte, baseret på den nyere version 3.

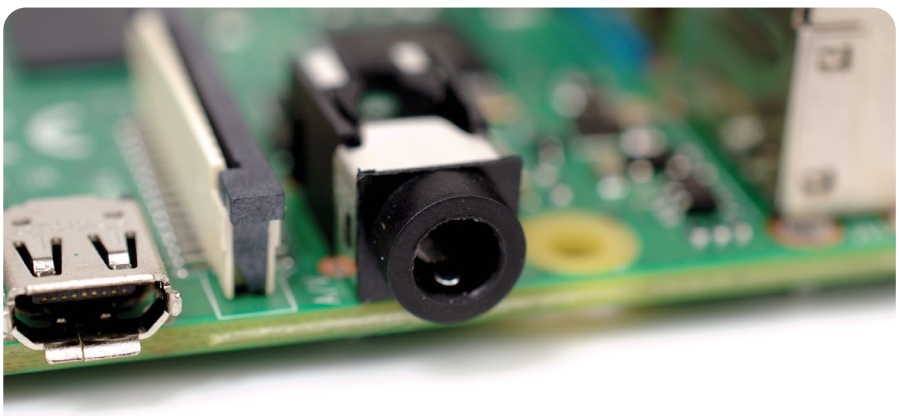
▲ **Figur 1-6:** Raspberry Pi's USB-porte

Til højre for USB-portene findes en *Ethernet-port*, også kendt som en *netværksport* (**Figur 1-7**). Du kan bruge denne port til at forbinde Raspberry Pi til et kablet computernetværk ved hjælp af et kabel med et såkaldt RJ45-stik i enden. Hvis du ser nøje på Ethernet-porten, ser du to lysdioder (LED'er) nederst; dette er status-lysdioder, som fortæller dig, at forbindelsen fungerer.



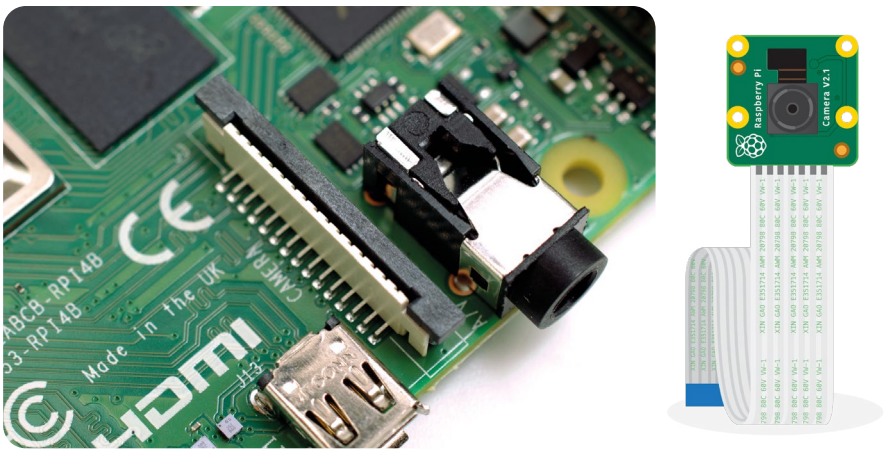
▲ **Figur 1-7:** Raspberry Pi's Ethernet-port

Lige over USB-portene, i venstre side af Raspberry Pi, er der et *3,5 mm audiovisuelt (AV) stik* (**Figur 1-8**). Dette kaldes også *hovedtelefonstikket*, og det kan bruges til lige præcis det formål – selvom du får bedre lyd ved at tilslutte til forstærkede højttalere snarere end hovedtelefoner. Det har en skjult, ekstra funktion: Ud over lyd bærer 3,5 mm AV-stikket et videosignal, der kan tilsluttes tv, projektorer og andre skærme, som understøtter et *sammensat videosignal* ved hjælp af et specielt kabel kaldet en *tip-ring-ring-sleeve (TRRS)*-adapter.



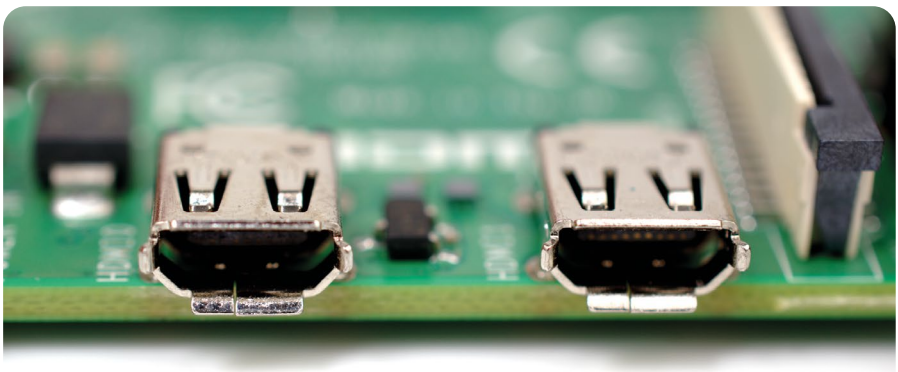
▲ **Figur 1-8:**Raspberry Pi's 3,5 mm AV-stik

Lige over 3,5 mm AV-stikket er der et mærkeligt stik med en plastikklap, der kan trækkes op; dette er *kamerastikket*, også kendt som *Camera Serial Interface (CSI)* (**Figur 1-9**). Dette giver dig mulighed for at bruge det specialdesignede Raspberry Pi-Camera Module (om hvilket du lærer mere i **Kapitel 8, Raspberry Pi's Camera Module**.)



Figur 1-9: Raspberry Pi's kamerastik

Ovenover, stadig i venstre side af kortet findes *micro High Definition Multimedia Interface (micro-HDMI) porte*, som er en mindre version af de stik, du finder på en spilkonsol, set-top-boks eller tv (**Figur 1-10**). Multimediedelen af navnet fortæller dig, at den bærer både lyd- og videosignaler, mens high-definition fortæller dig, at du kan forvente fremragende kvalitet. Du bruger disse til at forbinde Raspberry Pi til en eller to skærmenheder: En computerskærm, et tv eller en projektor.

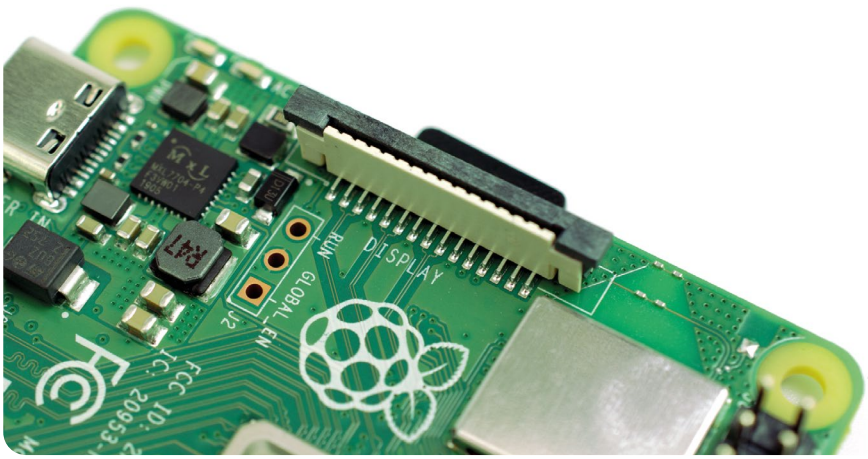


Figur 1-10: Raspberry Pi's micro-HDMI-porte

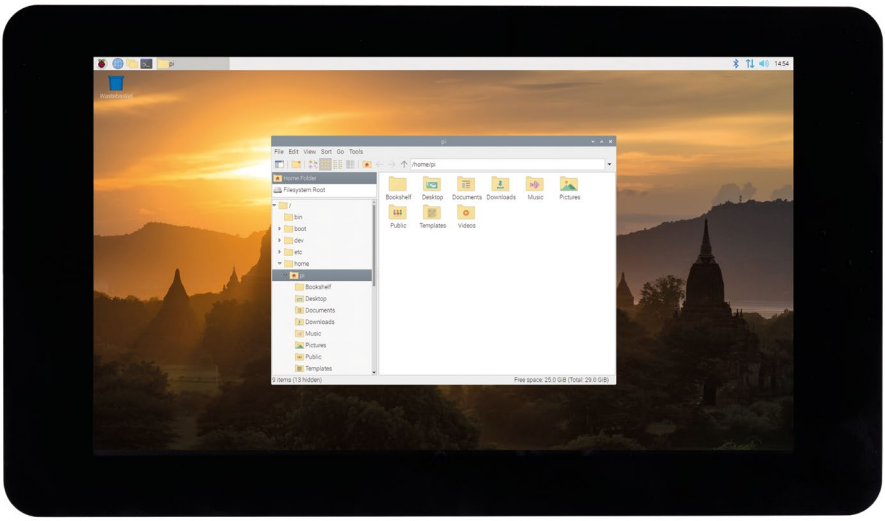
Over HDMI-portene er der en *USB-C-strømport* (**Figur 1-11**), som du bruger til at forbinde Raspberry Pi til en strømkilde. USB-C-porten er et almindeligt syn på smartphones, tablets og andre bærbare enheder. Du kan godt bruge en almindelig mobiloplader til at levere strøm til Raspberry Pi, men for at få de bedste resultater, skal du bruge den officielle Raspberry Pi USB-C-strømforsyning.

Figur 1-11: Raspberry Pi's USB-C-strømstik

Ved den øverste kant af kortet er der et andet mærkeligt stik (**Figur 1-12**), som ved første øjekast ser ud til at være identisk med kamerastikket. Dette er dog nøjagtigt det modsatte: et *skærmstik* eller *DSI (Display Serial Interface)*, der er designet til brug med en Raspberry Pi-touchskærm (**Figur 1-13**, på næste side).

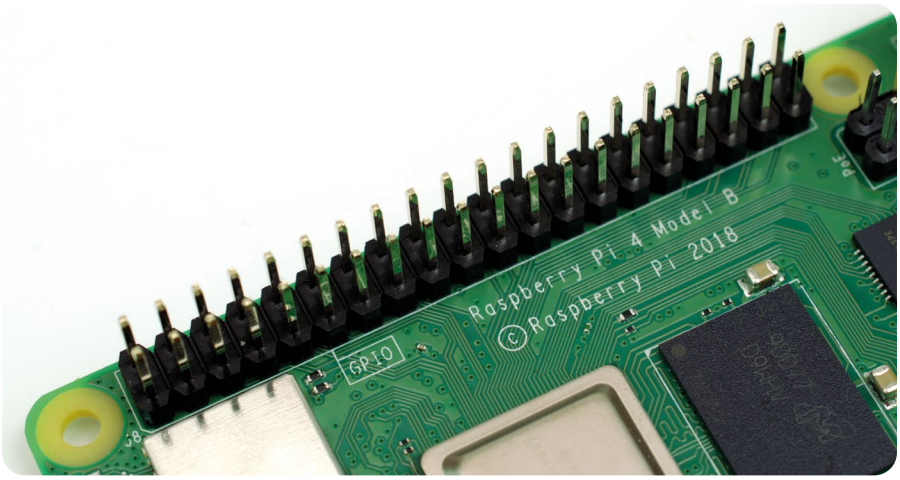


Figur 1-12: Raspberry Pi's skærmstik (DSI)



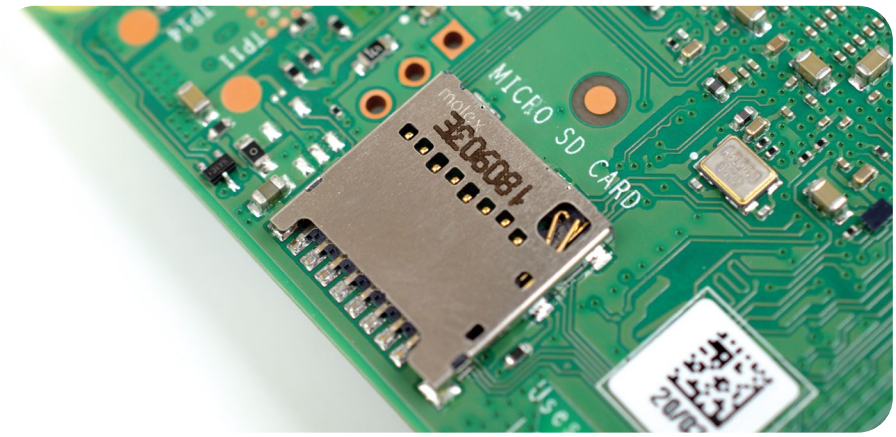
▲ **Figur 1-13:** Raspberry Pi's touchskærm

Ved den højre kant af tavlen finder du 40 metalben, opdelt i to rækker med 20 ben (**Figur 1-14**). Dette er *GPIO-headeren (almindeligt anvendt input/output)*, en funktion i Raspberry Pi, der bruges til at tale med ekstra hardware fra lysdioder og knapper hele vejen til temperatursensorer, joysticks og pulsfrekvensmonitorer. Du lærer mere om GPIO-headeren i **Kapitel 6, Fysisk programmering med Scratch og Python**. Lige under og til venstre for denne header er der en anden, mindre header med fire ben: denne bruges til at forbinde Power over Ethernet (PoE) HAT, en valgfri tilføjelse, der lader Raspberry Pi modtage strøm fra en netværksforbindelse snarere end USB-C-port.



▲ **Figur 1-14:** Raspberry Pi's GPIO-header

Der er en sidste port på Raspberry Pi, men du kan ikke se den øverst. Vend kortet om, så finder du et *microSD-kortstik* på den modsatte side af kortet til skærmstikket (**Figur 1-15**). Dette er Raspberry Pi's lagerplads: microSD-kortet, der er indsat her, indeholder alle de filer, du gemmer, al den software, du installerer, og det operativsystem, der får Raspberry Pi til at køre.



▲ **Figur 1-15:** Raspberry Pi's microSD-kortstik

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

Raspberry Pi 400

Raspberry Pi 400 tager de samme komponenter som Raspberry Pi 4 og placerer dem inde i en tastaturindkapsling. Ud over at beskytte dem optager tastaturindkapslingen mindre plads på dit skrivebord og hjælper med at holde kablerne ryddelige.

Raspberry Pi 400 består af de samme kernekomponenter som Raspberry Pi 4, inklusive system-on-chip og hukommelse. Du kan ikke se dem, men de er der. Hvad du kan se er de eksterne dele, der starter med tastaturet (**Figur 1-16**). Mod det højre hjørne er der tre lysdioder (LED'er): Den første lyser, når du trykker på Num Lock-tasten, som skifter nogle af tasterne til at fungere som et numerisk tastatur på et tastatur i fuld størrelse; den anden lyser, når du trykker på Caps Lock, hvilket gør bogstavtasterne til enten små eller store bogstaver; og den sidste lyser, når Raspberry Pi 400 er tændt.

På bagsiden af Raspberry Pi 400 (**Figur 1-17**) er portene. Den yderste venstre port, set bagfra, er GPIO-headeren til generelle formål (input/output). Dette er den samme header som beskrevet på side 17, men vendt: den første pin, Pin 1, er øverst til højre, mens den sidste pin, Pin 40, er nederst til venstre. Du kan få mere at vide om GPIO-headeren i Kapitel 6, Fysisk programmering med Scratch og Python.

Figur 1-17: Portene findes bag på Raspberry Pi 400

Ved siden af GPIO-headeren er stikket til microSD-kortet. Dette gemmer microSD-kortet, der fungerer som Raspberry Pi 400's lager til operativsystemet, programmer og andre data. MicroSD-kortet leveres forudinstalleret i Raspberry Pi 400; du kan fjerne det ved at skubbe forsigtigt på kortet, indtil det klikker og springer ud og derefter trække det ud. Når du sætter kortet i igen, skal du sørge for, at de skinnende metalkontakter vender nedad; kortet skal glide ind med et let klik.

De næste to porte er micro-HDMI-porte til tilslutning til en computerskærm, tv eller anden skærm. Ligesom Raspberry Pi 4 understøtter Raspberry Pi 400 op til to skærme. Ved siden af disse er USB-C-strømporten til tilslutning til Raspberry Pi-strømforsyning eller kompatibel USB-strømforsyning.

De to blå porte er USB 3.0-porte, der giver en højhastighedsforbindelse til enheder inklusive SSD-drev (SSD'er), hukommelsessticks, printere m.m. Den hvide port til højre for disse er en USB 2.0-port med lavere hastighed, som du kan forbinde den medfølgende Raspberry Pi-mus til.

Den sidste port er en Gigabit Ethernet-netværksport, der giver dig mulighed for at tilslutte Raspberry Pi 400 til dit netværk ved hjælp af et netværkskabel som alternativ til at bruge det indbyggede trådløse wi-fi-netværk. Du kan læse mere om tilslutning af Raspberry Pi 400 til et netværk i kapitel 2, Kom godt i gang med din Raspberry Pi.

Kapitel 2

Kom godt i gang med din Raspberry Pi

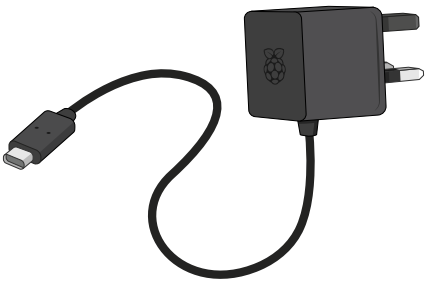
Se de væsentligste ting, som du har brug for til din Raspberry Pi, og hvordan du forbinder dem alle for at få det konfigureret og til at fungere

Raspberry Pi er designet til at være så hurtig og nem at opsætte og bruge som muligt, men – som med enhver computer – er den afhængig af forskellige eksterne komponenter, kaldet *perifert udstyr*. Selvom det er let at kigge på det bare kredsløb på Raspberry Pi – som ser væsentligt anderledes ud end de indkapslede, lukkede computere, du måske er vant til – og begynde at bekymre sig om, at det hele nu er ved at blive kompliceret, er det ikke tilfældet. Du kan være i gang med Raspberry Pi på under ti minutter ved blot at følge trinene i denne vejledning.

Hvis du har fået denne bog i et Raspberry Pi Desktop Kit eller med en Raspberry Pi 400, har du allerede næsten alt, hvad du behøver for at komme i gang: Det eneste du behøver er en computerskærm eller et tv med et HDMI-stik – det samme type stik, der bruges af set-top-bokse, Blu-ray-afspillere og spilkonsoller – så du kan se, hvad din Raspberry Pi laver.

Hvis du fik din Raspberry Pi uden tilbehør, har du også brug for:

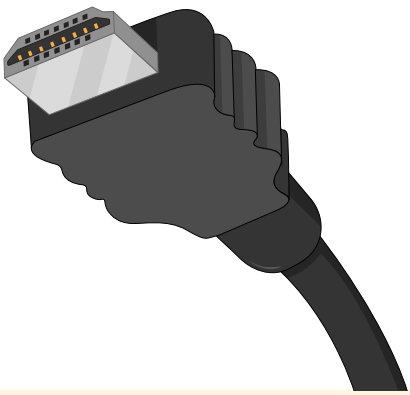
- **USB-strømforsyning** – En 5 V-strømforsyning med en effekt på 3 ampere (3A) og med et USB-C-stik. Vi anbefaler at bruge den officielle Raspberry Pi-strømforsyning, da den kan klare Raspberry Pi's hurtigt skiftende strømbehov.



- **microSD-kort med NOOBS** – microSD-kortet fungerer som Raspberry Pi's permanente lagerplads; alle de filer, du opretter, og software, som du installerer, inklusive selve operativsystemet, gemmes på kortet. Et 8 GB-kort får dig i gang, selvom et 16 GB-kort giver mere plads til at vokse. Brug af et kort med NOOBS (New Out-of-Box Software) forudinstalleret sparer dig tid. Se ellers **Bilag A** for instruktioner om installation af et operativsystem (OS) på et tomt kort.

- **USB-tastatur og -mus** – Tastaturet og musen giver dig mulighed for at styre din Raspberry Pi. Næsten ethvert kablet eller trådløst tastatur og mus med et USB-stik fungerer sammen med Raspberry Pi, selvom nogle "gaming"-tastaturer med farverige lys måske trækker for meget strøm til at være helt pålidelige.

- **Micro-HDMI-kabel** – Dette overfører lyd og billeder fra Raspberry Pi til tv'et eller skærmen. Den ene ende af kablet har et micro-HDMI-stik til Raspberry Pi; den anden, et HDMI-stik i fuld størrelse til din skærm. Eller du kan bruge en micro-HDMI til HDMI-adapter og et standard HDMI-kabel i fuld størrelse. Hvis du bruger en skærm uden et HDMI-stik, kan du købe micro-HDMI til DVI-D-, DisplayPort- eller VGA-adaptere. For at oprette forbindelse til et ældre tv, der bruger kompositvideo eller har et SCART-stik, skal du bruge et 3,5 mm tip-ring-ring-sleeve (TRRS) lyd-/videokabel.



Raspberry Pi er sikker at bruge uden etui, forudsat at du ikke placerer den på en metaloverflade, der kan lede strøm og forårsage kortslutning. Et valgfrit kabinet kan dog give yderligere beskyttelse; Desktop Kit inkluderer det officielle Raspberry Pi-kabinet, mens tredjepartskabinetter fås hos alle gode forhandlere.

Hvis du vil bruge Raspberry Pi på et kablet netværk i stedet for et trådløst netværk (wi-fi), skal du også bruge et netværkskabel. Dette skal tilsluttes i den ene ende til dit netværks switch eller router. Hvis du planlægger at bruge Raspberry Pi's indbyggede trådløse radio, behøver du ikke et kabel; du skal dog kende navnet og nøglen eller adgangssætningen til dit trådløse netværk.

RASPBERRY PI 400-OPSÆTNING

Følgende instruktioner er til opsætning af Raspberry Pi 4 eller et andet kort-medlem af Raspberry Pi-familien. Instruktioner til opsætning af Raspberry Pi 400 findes på side 32.

Opsætning af hardware

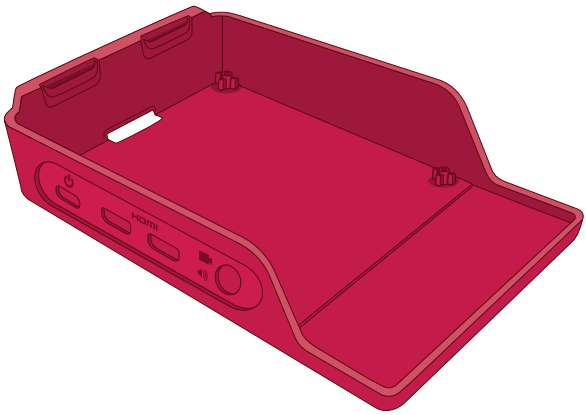
Start med at pakke din Raspberry Pi ud af kassen. Raspberry Pi er et robust stykke hardware, men det betyder ikke, at det ikke kan gå i stykker: Gør det til en vane at holde brættet ved kanterne snarere end på dets flade sider, og vær ekstra forsigtig omkring de hævede metalnåle. Hvis disse ben bøjes, vil det i bedste fald gøre det vanskeligt at bruge tilføjelseskort og anden ekstra hardware og i værste fald kan det forårsage en kortslutning, der vil beskadige din Raspberry Pi.

Hvis du ikke allerede har gjort det, skal du kigge på **Kapitel 1 Lær din Raspberry Pi at kende** for detaljer om præcis, hvor de forskellige porte er, og hvad de gør.

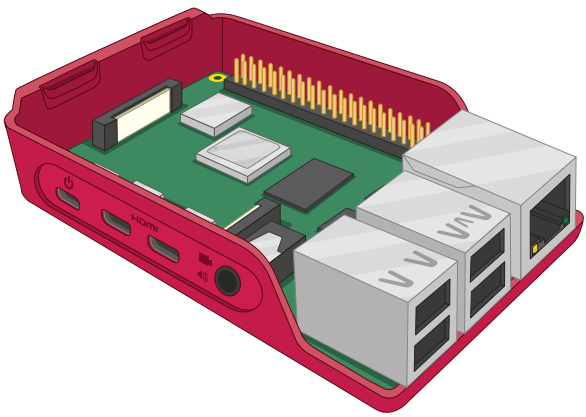
Samling af kabinettet

Hvis du installerer Raspberry Pi i et kabinet, skal dette være det første trin. Hvis du bruger det officielle Raspberry Pi-kabinet, skal du starte med at opdele det i de to individuelle dele: Den røde bund og det hvide låg.

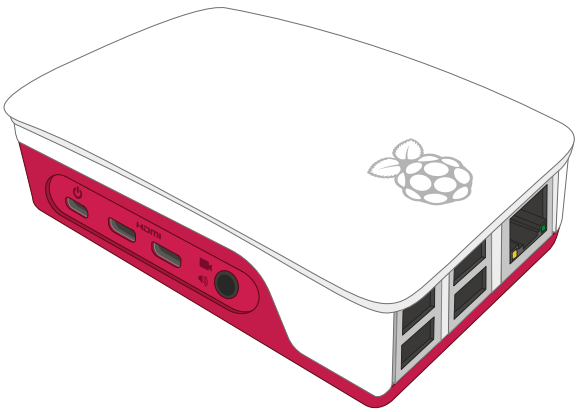
1 **Tag basen, og hold den, så den høje ende er til venstre og den lave ende er til højre.**



2 **Hold din Raspberry Pi (uden isat microSD-kort) ved dens USB- og Ethernet-porte, i en lille vinkel, og sæt dens stik (USB-C, 2 × micro-HDMI og 3,5 mm) i hullerne i siden af basen, og sænk den anden side forsigtigt ned, så den sidder fladt.**

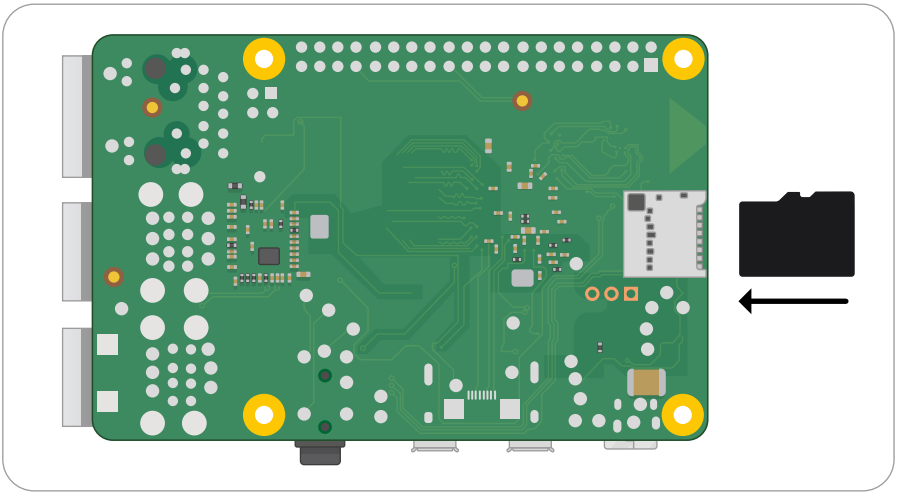


3 **Tag det hvide låg og placer de to clips til venstre i de matchende huller til venstre på basen over microSD-kortstikket. Når de er på plads, skal du skubbe højre side (over USB-porte) ned, indtil du hører et klik.**

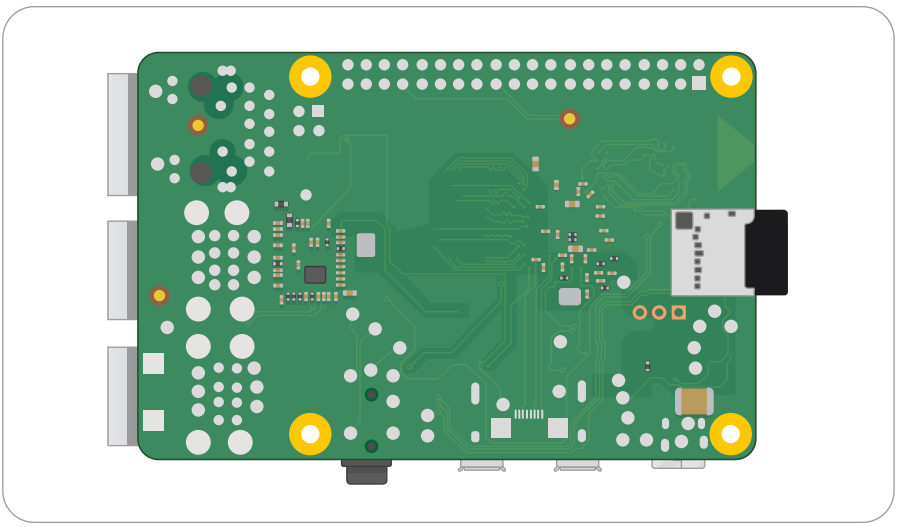


Tilslutning af microSD-kortet

For at installere microSD-kortet, som er Raspberry Pi's *lagerplads*, skal du dreje Raspberry Pi (og kabinettet, hvis du bruger et), og skubbe kortet ind i microSD-stikket med etiketten vendt væk fra Raspberry Pi. Det kan kun sættes i på én måde og skal glide i uden at trykke for meget.



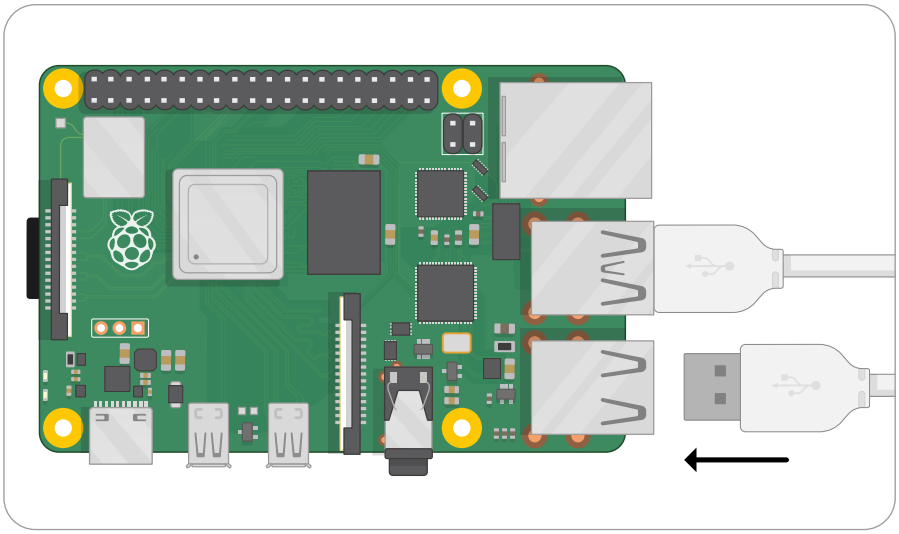
MicroSD-kortet glider ind i stikket og stopper så uden et klik.



Hvis du vil fjerne det igen på et tidspunkt, skal du bare tage fat i enden af kortet og trække det forsigtigt ud. Hvis du bruger en ældre model af Raspberry Pi, skal du først give kortet et forsigtigt skub for at låse det op; dette er ikke nødvendigt med en Raspberry Pi 3 eller 4.

Tilslutning af et tastatur og en mus

Tilslut tastaturets USB-kabel til en af de fire USB-porte (2.0 eller 3.0) på Raspberry Pi. Hvis du bruger det officielle Raspberry Pi-tastatur, er der en USB-port på bagsiden til musen; hvis ikke, skal du bare slutte USB-kablet fra din mus til en anden USB-port på Raspberry Pi.



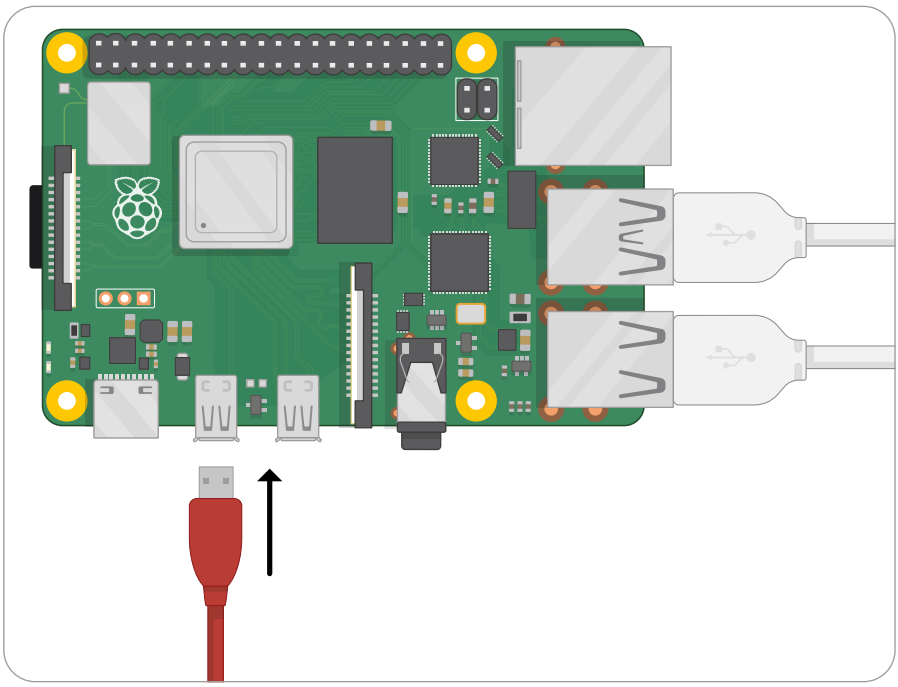
USB-stikkene til tastaturet og musen skal glide ind uden for meget pres; hvis du skal tvinge stikket ind, er der noget galt. Kontroller, at USB-stikket vender den rigtige side op!

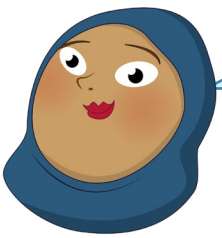
TASTATUR OG MUS

Tastaturet og musen fungerer er dine vigtigste værktøjer til at fortælle Raspberry Pi, hvad den skal gøre; i forbindelse med computere er disse kendt som *inputenheder* i modsætning til skærmen, som er en *output-enhed*.

Tilslutning af en skærm

Tag micro-HDMI-kablet, og tilslut den mindre ende til micro-HDMI-porten tættest på USB-C-porten på din Raspberry Pi, og den anden ende til din skærm. Hvis din skærm har mere end én HDMI-port, skal du kigge efter et portnummer ved siden af selve stikket; du skal skifte tv'et til denne indgang for at kunne se Raspberry Pi's skærm. Hvis du ikke kan se et portnummer, skal du ikke bekymre dig: Du kan simpelthen skifte gennem hvert input, indtil du finder Raspberry Pi.



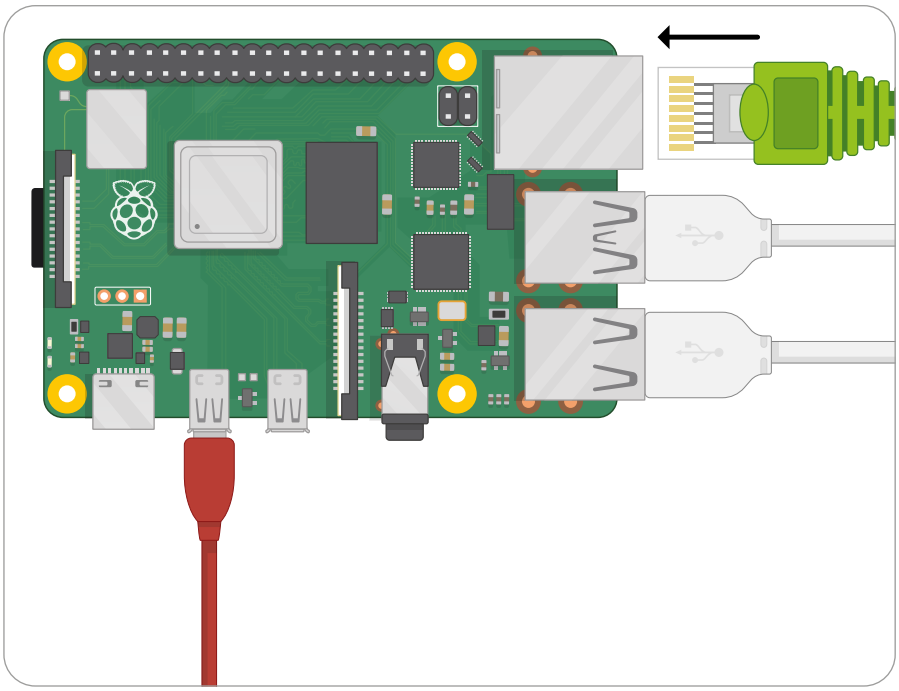


TV-TILSLUTNING

Hvis dit tv eller din skærm ikke har et HDMI-stik, betyder det ikke, at du ikke kan bruge Raspberry Pi. Adapterkabler, der er tilgængelige fra enhver elektronikforhandler, giver dig mulighed for at konvertere micro-HDMI-porten på Raspberry Pi til DVI-D, DisplayPort eller VGA til brug med ældre computerskærme; disse sluttes blot til Raspberry Pi's micro-HDMI-port, derefter bruges et passende kabel til at forbinde adapterkablet til skærmen. Hvis dit tv kun har en composite video eller SCART-indgang, kan du købe 3,5 mm tip-ring-ring-sleeve (TRRS) adapterkabler og composite-til-SCART-adaptere, der sluttes til 3,5 mm AV-stikket.

Tilslutning af et netværkskabel (valgfrit)

For at forbinde din Raspberry Pi til et kablet netværk skal du tage et netværkskabel – kaldet et Ethernet-kabel – og sætte det i Raspberry Pi's Ethernet-port med plastikclipsen nedad, indtil du hører et klik. Hvis du har brug for at fjerne kablet, skal du blot presse plastikklemmen indad mod stikket og forsigtigt trække kablet ud igen.

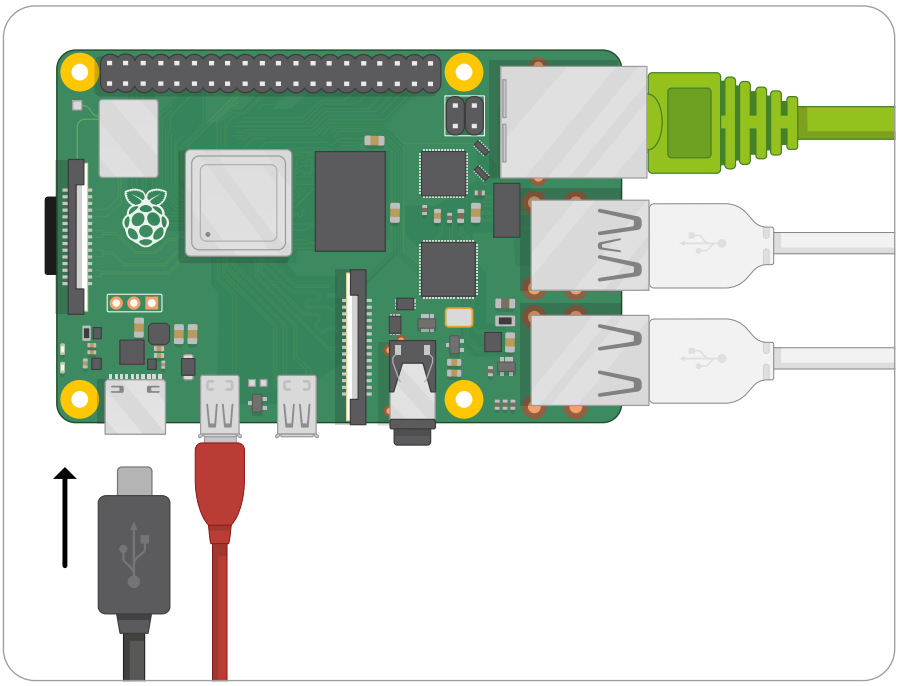


Den anden ende af dit netværkskabel skal tilsluttes en hvilken som helst ledig port på din netværkshub, switch eller router på samme måde.

Tilslutning af en strømforsyning

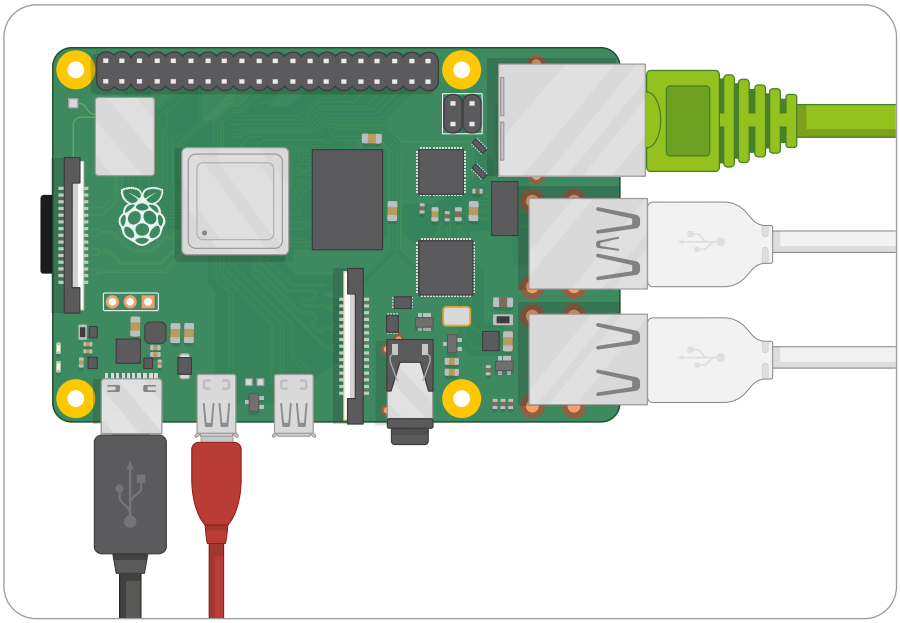
Tilslutning af Raspberry Pi til en strømforsyning er sidste trin i hardwareopsætningsprocessen, og det skal du først gøre, når du er klar til at konfigurere softwaren: Raspberry Pi har ikke en strømafbryder og tændes, så snart den er tilsluttet en strømforsyning.

Tilslut først USB-C-enden af strømkablet til USB-C-strømstikket på Raspberry Pi. Stikket kan vende begge veje og bør glide forsigtigt i. Hvis din strømforsyning har et aftageligt kabel, skal du sørge for, at den anden ende er tilsluttet selve strømforsyningen.

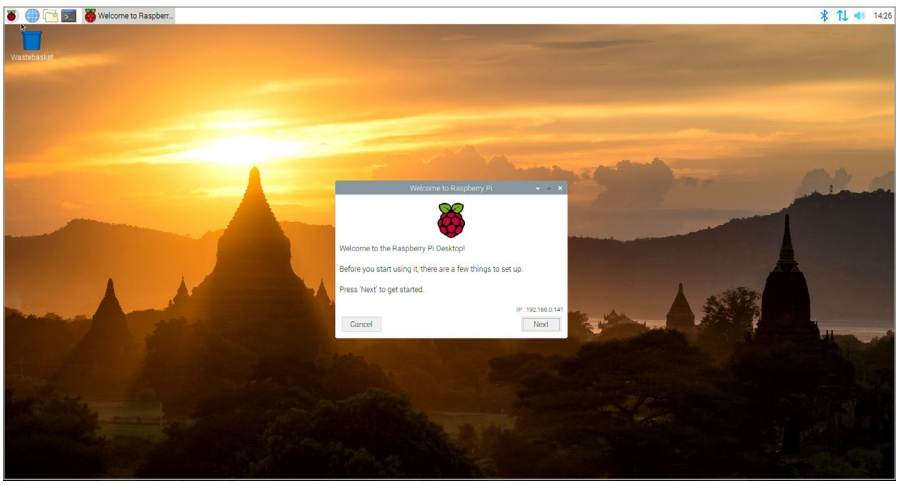


Til slut skal du sætte strømforsyningen i en stikkontakt og tænde på stikkontakten. Din Raspberry Pi begynder straks at køre.

Tillykke: Du har nu samlet din Raspberry Pi!



Du vil kort se fire Raspberry Pi-logoer øverst til venstre på en sort skærm og muligvis se en blå skærm, når softwaren ændrer størrelse for at udnytte dit microSD-kort fuldt ud. Hvis du ser en sort skærm, skal du vente et par minutter. Første gang Raspberry Pi starter op, skal den lige gøre nogle ting i baggrunden. Efter et stykke tid vil du se Raspberry Pi OS-skrivebordet og installationsguiden, som i **Figur 2-1**. Dit operativsystem er nu klar til at blive konfigureret, hvilket du lærer at gøre i **Kapitel 3, Sådan bruger du din Raspberry Pi**.



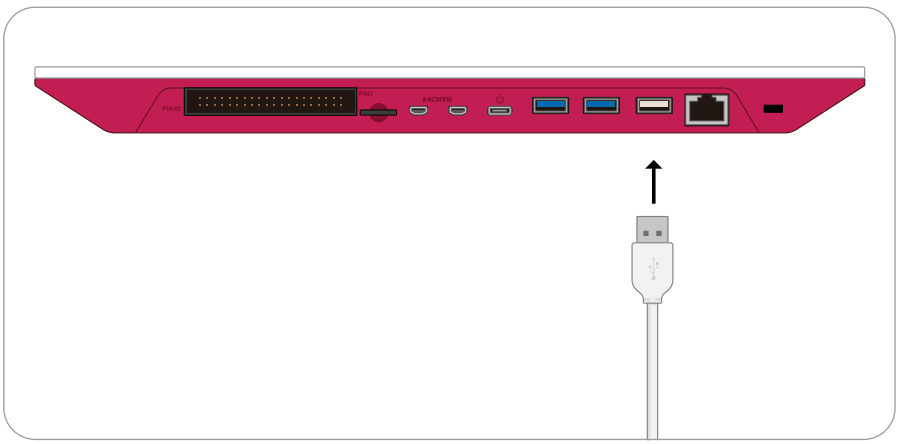
▲ **Figur 2-1:** Raspberry Pi OSskrivebordet og installationsguiden

Opsætning af Raspberry Pi 400

I modsætning til Raspberry Pi 4 leveres Raspberry Pi 400 med et indbygget tastatur og allerede installeret microSD-kort. Du skal stadig tilslutte et par kabler for at komme i gang, men det tager kun et par minutter.

Tilslutning af en mus

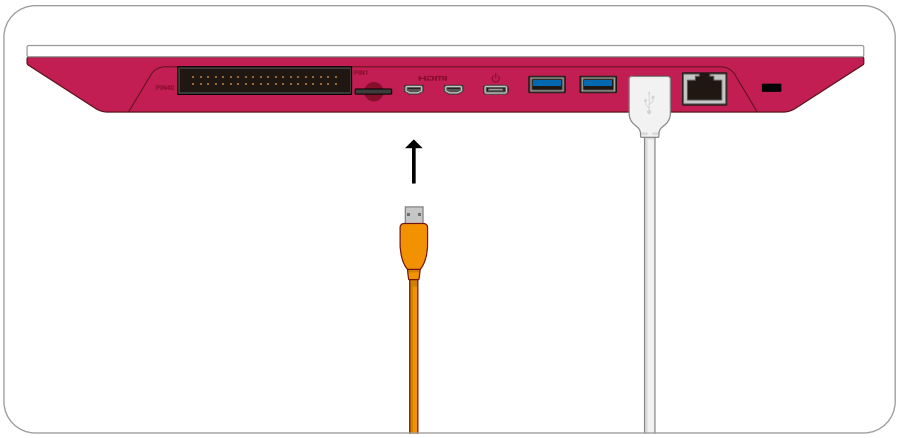
Raspberry Pi 400's tastatur er allerede tilsluttet, så du kun har musen at tilføje. Tag USB-kablet fra enden af musen, og sæt det i en af de tre USB-porte (2.0 eller 3.0) på bagsiden af Raspberry Pi 400. Hvis du vil gemme de to hurtige USB 3.0-porte til andet tilbehør, skal du bruge den hvide port.



USB-stikket skal kunne sættes i for meget pres; hvis du skal tvinge stikket ind, er der noget galt. Kontroller, at USB-stikket vender den rigtige side op!

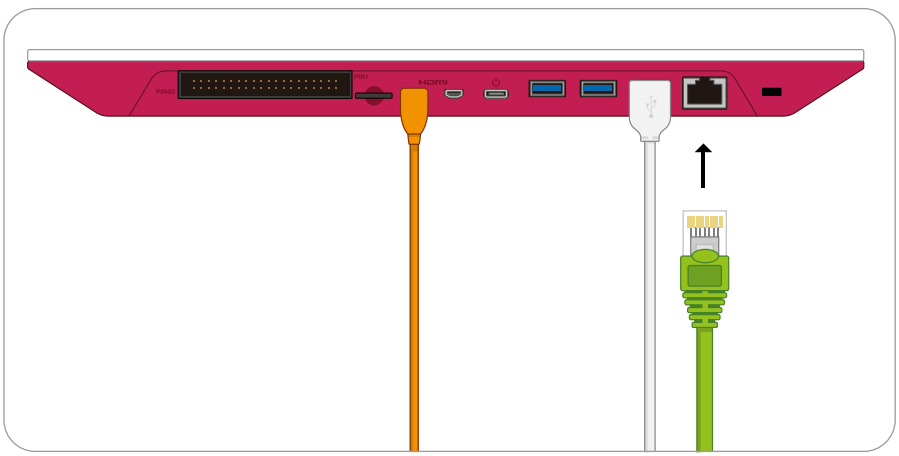
Tilslutning af en skærm

Tag micro-HDMI-kablet, og tilslut den mindre ende til micro-HDMI-porten tættest på microSD-porten på din Raspberry Pi 400, og den anden ende til skærmen. Hvis din skærm har mere end én HDMI-port, skal du kigge efter et portnummer ved siden af selve stikket; du skal skifte tv'et til denne indgang for at kunne se Raspberry Pi's skærm. Hvis du ikke kan se et portnummer, skal du ikke bekymre dig: Du kan simpelthen skifte gennem hvert input, indtil du finder Raspberry Pi.



Tilslutning af et netværkskabel (valgfrit)

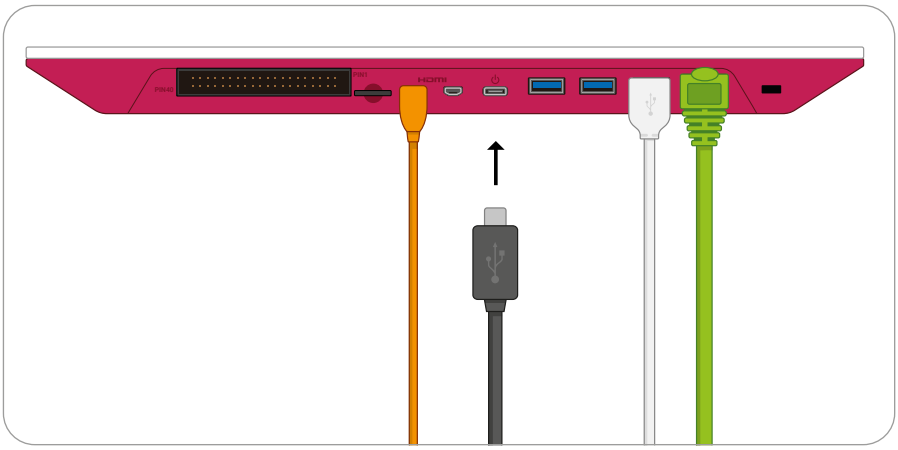
For at forbinde din Raspberry Pi 400 til et kablet netværk skal du tage et netværkskabel – kaldet et Ethernet-kabel – og sætte det i din Raspberry Pi 400's Ethernet-port med plastikclipsen nedad, indtil du hører et klik. Hvis du har brug for at fjerne kablet, skal du blot presse plastikklemmen indad mod stikket og forsigtigt trække kablet ud igen.



Den anden ende af dit netværkskabel skal tilsluttes en hvilken som helst ledig port på din netværkshub, switch eller router på samme måde.

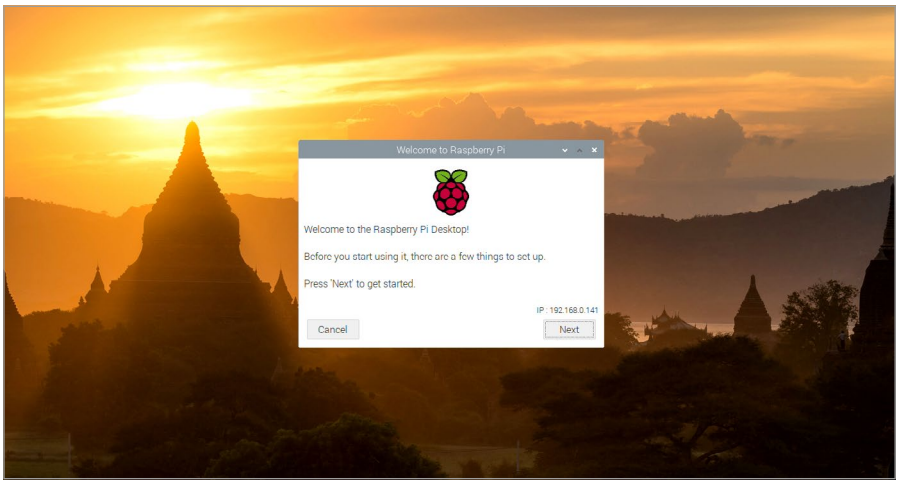
Tilslutning af en strømforsyning

Tilslutning af Raspberry Pi 400 til en strømforsyning er sidste trin i hardwareopsætningsprocessen, og det skal du først gøre, når du er klar til at konfigurere softwaren: Raspberry Pi 400 har ikke en strømafbryder og tændes, så snart den er tilsluttet en strømforsyning. Tilslut først USB-C-enden af strømkablet til USB-C-strømstikket på Raspberry Pi. Stikket kan vende begge veje og bør glide forsigtigt i. Hvis din strømforsyning har et aftageligt kabel, skal du sørge for, at den anden ende er tilsluttet selve strømforsyningen.



Til slut skal du sætte strømforsyningen i en stikkontakt og tænde på stikkontakten. Din Raspberry Pi 400 begynder straks at køre. Tillykke: Du har nu samlet din Raspberry Pi 400!

Du vil kort se fire Raspberry Pi-logoer øverst til venstre på en sort skærm og muligvis se en blå skærm, når softwaren ændrer størrelse for at udnytte dit microSD-kort fuldt ud. Hvis du ser en sort skærm, skal du vente et par minutter. Første gang Raspberry Pi starter op, skal den lige gøre nogle ting i baggrunden. Efter et stykke tid vil du se Raspberry Pi OS-skrivebordet og installationsguiden, som i **Figur 2-2**. Dit operativsystem er nu klar til at blive konfigureret, hvilket du lærer at gøre i **Kapitel 3, Sådan bruger du din Raspberry Pi**.

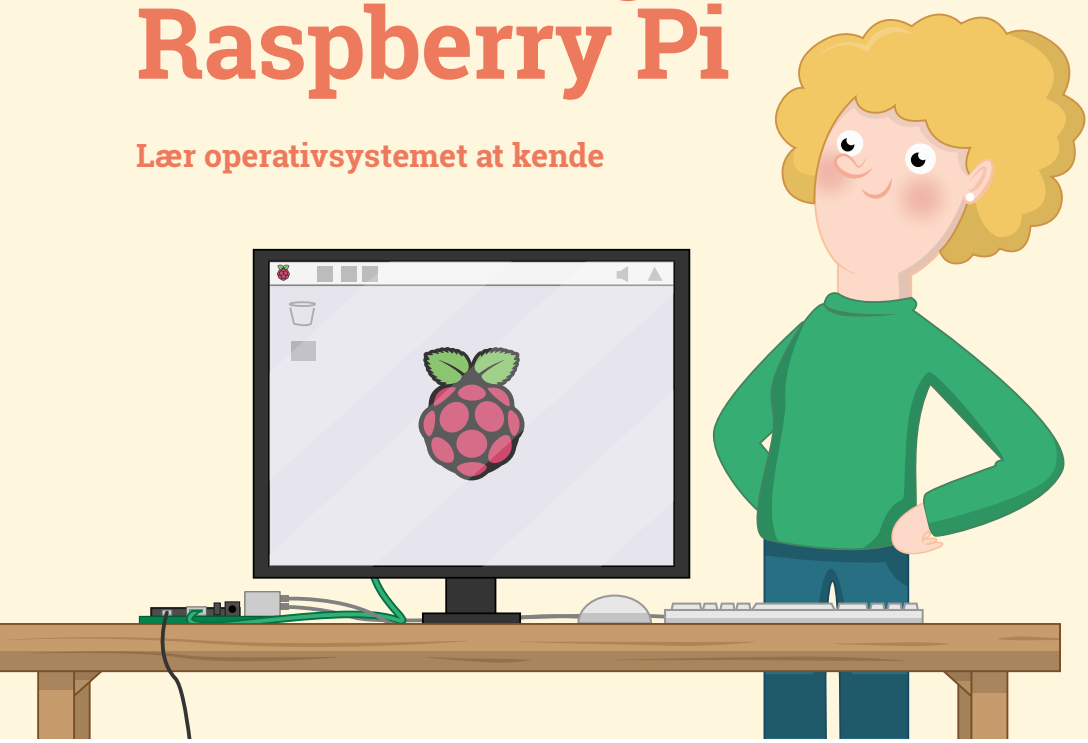


▲ **Figur 2-2:** Raspberry Pi OSskrivebordet og installationsguiden

Kapitel 3

Sådan bruger du din Raspberry Pi

Lær operativsystemet at kende

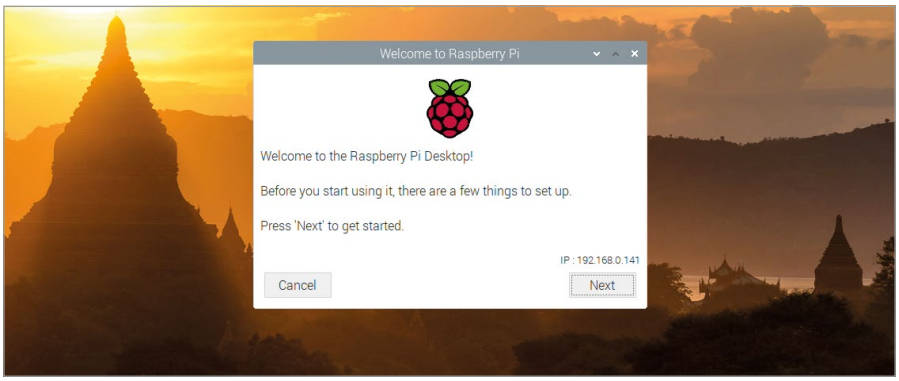


Raspberry Pi kan køre en bred vifte af software, inklusive en række forskellige operativsystemer, dvs. den software, der får computeren til at køre. Det mest populære og også det officielle Raspberry Pi Foundation-operativsystem er Raspberry Pi OS. Operativsystemet er baseret på Debian Linux, er skræddersyet til Raspberry Pi og leveres med en række præinstallerede værktøjer, så det er helt klart til brug.

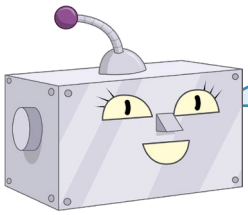
Hvis du indtil nu kun har brugt Microsoft Windows eller Apple macOS, skal du ikke bekymre dig: Raspberry Pi OS er baseret på de samme grundlæggende principper med vinduer, ikoner og menuer, så du vil hurtigt føle dig hjemme. I det følgende kapitel kan du læse om, hvordan du kommer i gang, og du kan også læse om den software, der følger med.

Velkomstguiden

Første gang, du starter Raspberry Pi OS, vises der en velkomstguide (**Figur 3-1**). Dette nyttige værktøj vil hjælpe dig med at ændre de grundlæggende indstillinger i Raspberry Pi OS (*konfigurationen*), så operativsystemet bliver tilpasset til det, du skal bruge det til.



▲ **Figur 3-1:** Velkomstguiden



SÅDAN LUKKER DU GUIDEN

Du kan lukke velkomstguiden ved at klikke på knappen Cancel (annuller). Bemærk dog, at visse funktioner i din Raspberry Pi, f.eks. det trådløse netværk, ikke vil fungere, før du har besvaret mindst det første sæt spørgsmål.

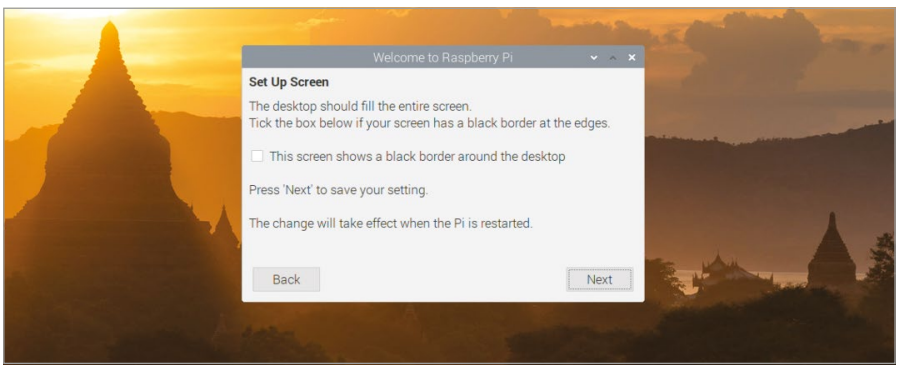
Klik på knappen Next (næste), og vælg dit land, sprog og tidszone ved at klikke på de tilsvarende rullelister og vælge den ønskede indstilling på listen (**Figur 3-2**). Hvis du f.eks. bruger et tastatur med amerikansk layout, skal du klikke på afkrydsningsfeltet for at sikre, at Raspberry Pi OS bruger det korrekte tastaturlayout. Hvis du gerne vil have, at skrivebordet og programmerne vises på engelsk, uanset hvilket sprog der tales i dit land, skal du markere afkrydsningsfeltet "Use English Language" (brug engelsk som sprog). Når du er færdig, skal du klikke på Next (næste).

▲ **Figur 3-2:** Valg af sprog

På den næste skærm bliver du bedt om at ændre adgangskoden for brugeren "pi" (standardadgangskoden er "raspberry"). Af sikkerhedsmæssige grunde er det en meget god ide at oprette en ny adgangskode. Brug de relevante felter (**Figur 3-3**). Klik på Next (næste), når du er færdig.

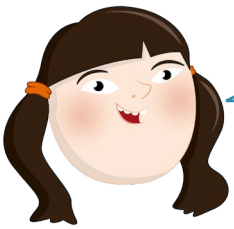
▲ **Figur 3-3:** Valg af en ny adgangskode

På den næste skærm skal du angive, om der er en sort kant omkring skærmbilledet (**Figur 3-4**). Hvis Raspberry Pi-skrivebordet fylder hele tv- eller computerskærm, skal du ikke markere afkrydsningsfeltet. Hvis der er en sort kant omkring skærmbilledet, og skærmbilledet ikke fylder hele tv- eller computerskærmen, skal du markere afkrydsningsfeltet. Klik på Next (næste), når du er klar til at fortsætte.



▲ **Figur 3-4:** Kontrol af sort kant

På næste skærm kan du vælge dit wi-fi-netværk på en liste (**Figur 3-5**). Scrol gennem listen over tilgængelige netværk med musen eller tastaturet, find dit netværk, klik på det, og klik derefter på Next (næste). Forudsat, at dit trådløse netværk er sikkert (og det burde det jo være), skal du indtaste adgangskoden til netværket (netværksnøglen). Adgangskoden står normalt på et kort, der blev leveret sammen med routeren, eller står på selve routeren. Klik på



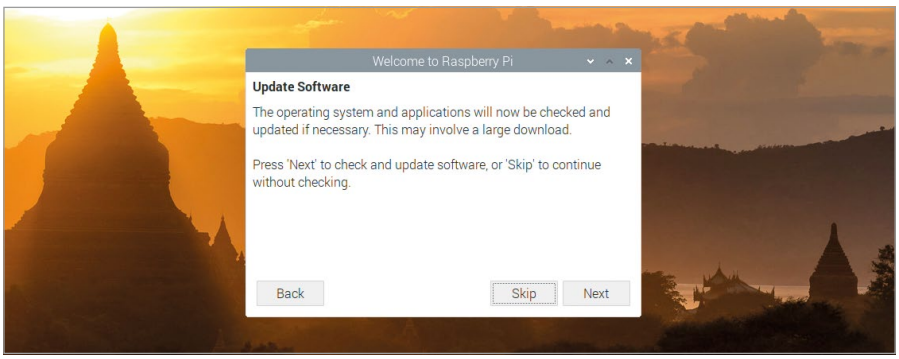
TRÅDLØST NETVÆRK

Det indbyggede trådløse netværk findes kun i Raspberry Pi 3, Pi 4 og Pi Zero W. Hvis du vil slutte en anden Raspberry Pi-model til et trådløst netværk, skal du bruge en USB-wi-fi-adapter.

Next (næste) for at oprette forbindelse til netværket. Hvis du ikke vil oprette forbindelse til et trådløst netværk, kan du springe dette trin over.

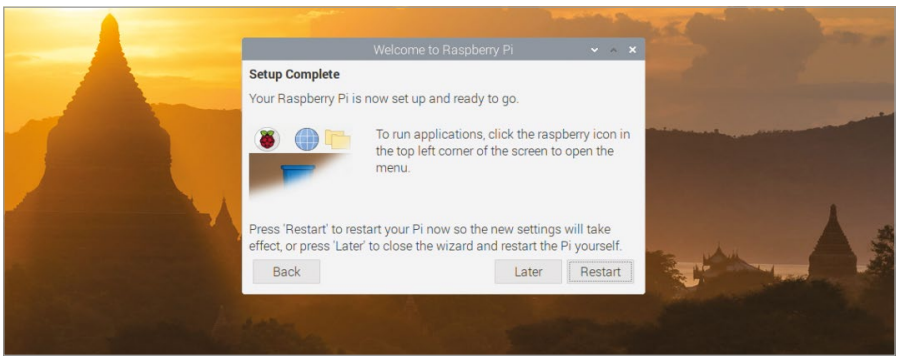
Figur 3-5: Valg af et trådløst netværk

På den næste skærm kan du kontrollere, om der er opdateringer til Raspberry Pi OS og anden software på din Raspberry Pi, og installere disse opdateringer (**Figur 3-6**). Raspberry Pi OS opdateres regelmæssigt for at rette fejl, tilføje nye funktioner og forbedre ydeevnen. Klik på Næste for at installere disse opdateringer, eller klik på Skip (spring over). Det kan tage nogle minutter at downloade opdateringerne, så vent et øjeblik. Når opdateringerne er installeret, vises der en meddelelse om, at systemet er opdateret. Klik på OK.



Figur 3-6: Tjek efter opdateringer

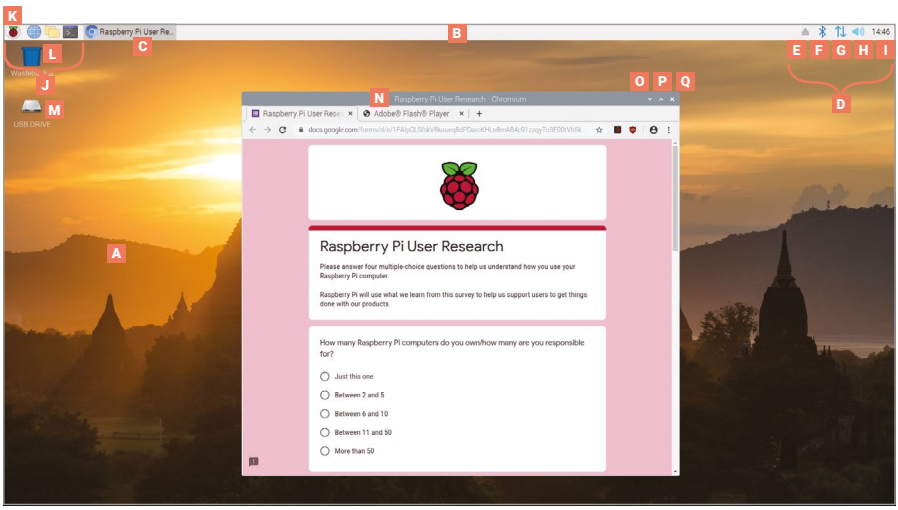
På den sidste skærm i velkomstguiden (**Figur 3-7**) bliver du spurgt, om din Raspberry Pi-skal genstartes. En genstart er nødvendig for, at visse ændringer kan træde i kraft). Hvis du bliver spurgt, om din Raspberry Pi skal genstartes, skal du klikke på Restart (genstart). Efter genstart vises velkomstguiden ikke igen, fordi du allerede har gennemført de nødvendige trin, og nu er din Raspberry Pi klar til brug.



▲ **Figur 3-7:** Genstart af Raspberry Pi

Navigation på skrivebordet

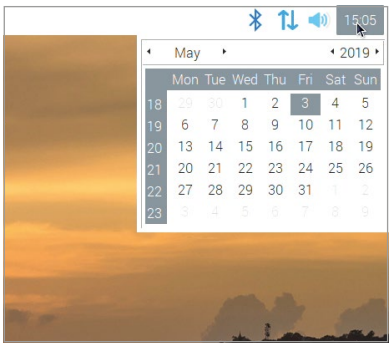
Det officielle navn på den version af Raspberry Pi OS, der installeres på de fleste Raspberry Pi-kort, er "Raspberry Pi OS with desktop", med henvisning til dens vigtigste grafiske brugergrænseflade (**Figur 3-8**). Hovedparten af skrivebordet er dækket af en billedbaggrund (**A** i **Figur 3-8**). Oven på denne baggrund vises de programmer, du kører på din Pi. Øverst på skrivebordet er der en proceslinje (**B**), som du kan bruge til at starte programmerne. De programmer, der kører på enheden, vises som opgaver (**C**) på proceslinjen.



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

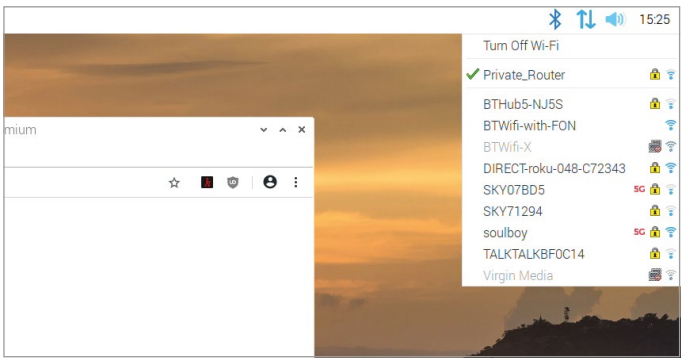
- **A** Baggrund
- **B** Proceslinje
- **C** Opgave
- **D** Systembakke
- **E** Skub medie ud
- **F** Bluetooth-ikon
- **G** Netværksikon
- **H** Lydstyrkeikon
- **I** Ur
- **J** Startmenu
- **K** Startknap (hindbærikon)
- **L** Papirkurv
- **M** Ikon for flytbart drev
- **N** Vindues titellinje
- **O** Minimer
- **P** Maksimer
- **Q** Luk

I højre side af menulinjen findes der en *systembakke* (**D**). Hvis der er nogen *flytbare drev*, f.eks. USB-nøgler, tilsluttet din Raspberry Pi, vises der et skubbe ud-ikon (**E**). Ved at klikke på dette ikon kan forbindelsen til drevet afsluttes på en sikker måde, så du kan fjerne drevet. Yderst til højre er der et ur (**I**). Ved at klikke på det får du vist kalenderen (**Figur 3-9**).



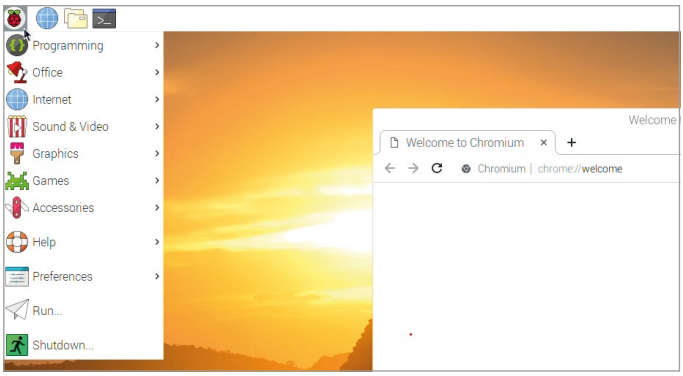
Figur 3-9: Den digitale kalender

Ved siden af den er der et højttalerikon (**H**). Klik på ikonet med venstre museknap for at justere Raspberry Pi-computerens lydstyrke, eller klik med højre museknap for at vælge, hvilken output der skal bruges. Næste ikon i rækken er netværksikonet (**G**). Når du har forbindelse til et trådløst netværk, vises signalstyrken som en række søjler, og når du har netværksforbindelse via kabel, vises der to pile. Ved at klikke på netværksikonet vises der en liste over trådløse netværk i nærheden (**Figur 3-10**), og når du klikker på Bluetooth-ikonet (**F**) ud for det, kan du oprette forbindelse til en Bluetooth-enhed i nærheden.



Figur 3-10: Oversigt over trådløse netværk i nærheden

I venstre side på menulinjen er der en *startmenu* (**J**). Her finder du de programmer, der er installeret sammen med Raspberry Pi OS. Nogle af disse er vises som genvejsikoner, mens andre vises ikke i startmenuen og skal startes ved at klikke på hindbærikonet (**K**) yderst til venstre (**Figur 3-11**).

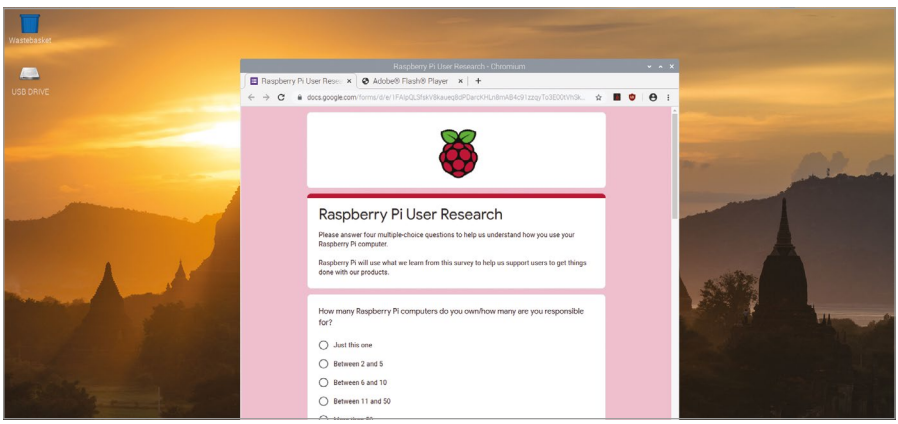


▲ **Figur 3-11:** Startmenuen i Raspberry Pi OS

Programmerne i startmenuen er opdelt i kategorier med beskrivende navne: F.eks. indeholder kategorien Programmering software, der kan hjælpe dig med at kode dine egne programmer (se **Kapitel 4 – Programmering med Scratch**), mens i kategorien Spil finder du lidt underholdende tidsfordriv. Ikke alle programmerne vil blive beskrevet i denne vejledning, så du er velkommen til selv at prøve dig frem.

Chromium-browseren

Som en lille øvelse i at bruge din Raspberry Pi kan du starte med at indlæse Chromium-browseren: Klik på hindbærikonet øverst til venstre for at åbne startmenuen, vælg kategorien Internet ved at flytte musen, og klik på Chromium-browseren for at indlæse den (**Figur 3-12**).



▲ **Figur 3-12:** Chromium-browseren

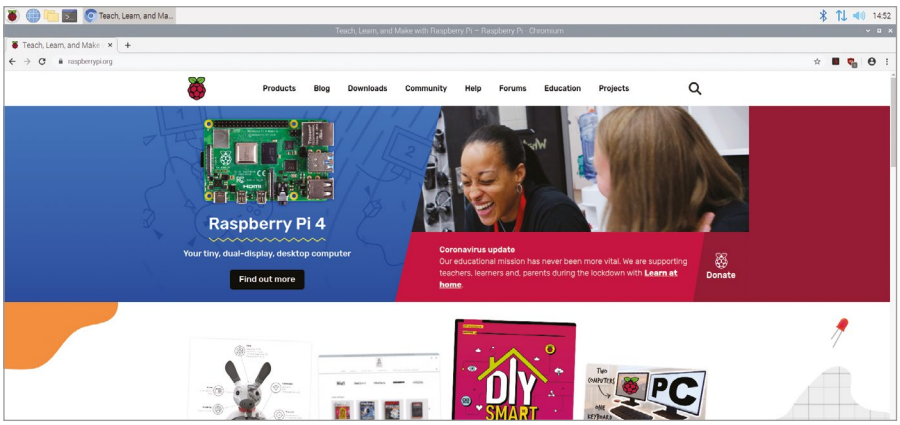
Hvis du nogensinde har brugt Google Chrome, vil du straks føle dig hjemme i Chromium. Chromium er en webbrowser, som du kan bruge til at besøge websteder, afspille videoer, spille spil og endda kommunikere med mennesker over hele verden på forummer og chatwebsteder.

For at komme i gang med at bruge Chromium kan du maksimere vinduet, så det fylder hele skærmen. Der er tre ikoner øverst til højre på titellinjen i Chromium-vinduet (**N**). Klik på det midterste pileikon (**P**). Denne knap bruges til at *maksimere* vinduet, så det fylder hele skærmen. Til venstre for maksimer-knappen har vi *minimer-knappen* (**O**). Når du klikker på denne knap, skjules vinduet, indtil du klikker på det på proceslinjen øverst på skærmen. Krydset til højre for maksimer-knappen er *luk-knappen* (**Q**), og gør præcist, som beskrevet: det lukker vinduet.

LUK OG GEM

At lukke et vindue, før du har gemt dit arbejde, er som regel en dårlig ide. Mange programmer vil spørge dig, om du vil gemme dit arbejde, når du klikker på luk-knappen, men nogle gør ikke.

Klik på adresselinjen øverst i Chromium-vinduet (den store hvide bjælke med et forstørrelsesglas i venstre side), skriv **www.raspberrypi.org**, og tryk derefter på **ENTER** på tastaturet. Raspberry Pi-webstedet indlæses (**Figur 3-13**). Du kan også skrive søgninger i adresselinjen: prøv at søge efter "Raspberry Pi", "Raspberry Pi OS" eller "computer til uddannelse".



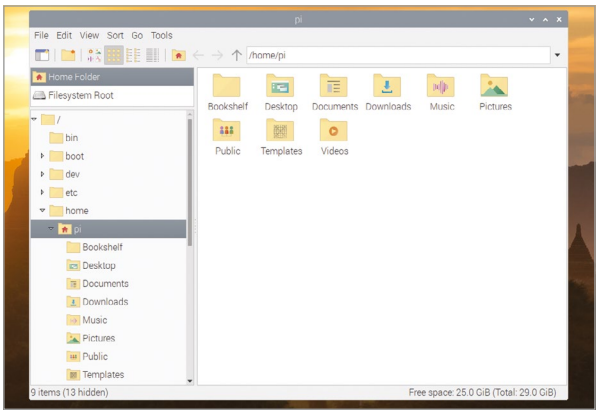
▲ **Figur 3-13:** Indlæsning af Raspberry Pi-webstedet i Chromium

Første gang, du åbner Chromium, åbnes der muligvis flere *faner* øverst i vinduet. Klik på en fane for at skifte til den. Hvis du vil lukke en fane uden at lukke Chromium skal du klikke på krydset i højre side af fanebladet. Faner er en praktisk måde at have flere websteder åbne uden at skulle jonglere med flere Chromium-vinduer. Du kan åbne en ny fane ved enten at klikke på faneknappen til højre for den sidste fane på listen eller holde **CTRL**-tasten nede, mens du trykker på **T**-tasten. Derefter kan du slippe **CTRL**.

Når du er færdig med at bruge Chromium, skal du klikke på knappen Luk øverst til højre i vinduet.

Filhåndtering

Filer, som du gemmer – f.eks. programmer, videoer, billeder – gemmes i din *hjemmemappe*. For at åbne hjemmemappen skal du klikke på hindbærikonet for at åbne startmenuen, vælge Tilbehør og derefter klikke på Filhåndtering (**Figur 3-14**).



Figur 3-14: Filhåndtering

Du kan bruge Filhåndtering til at gennemse filer og *mapper* på Raspberry Pi-enhedens microSD-kort eller flytbare drev som USB-drev, som du tilslutter til USB-porte på din Raspberry Pi. Når du åbner Filhåndtering, vises din hjemmemappe automatisk. Her findes der en række andre mapper, såkaldte *undermapper*, der – ligesom menuen – er arrangeret i kategorier. De vigtigste undermapper er:

- **Bookshelf (bogreol):** Denne mappe indeholder digitale kopier af bøger, magasiner og andre publikationer fra Raspberry Pi Press, inklusive en kopi af denne *Begyndervejledning*. Du kan læse disse og downloade flere bøger med Bookshelf-programmet, der findes i kategorien Help (hjælp) i startmenuen.

- **Desktop (skrivebord):** Denne mappe er, hvad du ser, når du starter Raspberry Pi OS. Hvis du gemmer en fil herinde, vises den på skrivebordet, hvilket gør det let at finde.

- **Documents (dokumenter):** Her gemmes de fleste af de filer, du opretter, f.eks. noveller og opskrifter.

- **Downloads:** Når du downloader en fil fra internettet med Chromium, gemmes den automatisk i Downloads.

- **Music (musik):** Her kan du gemme den musik, du laver eller lægger på Raspberry Pi.

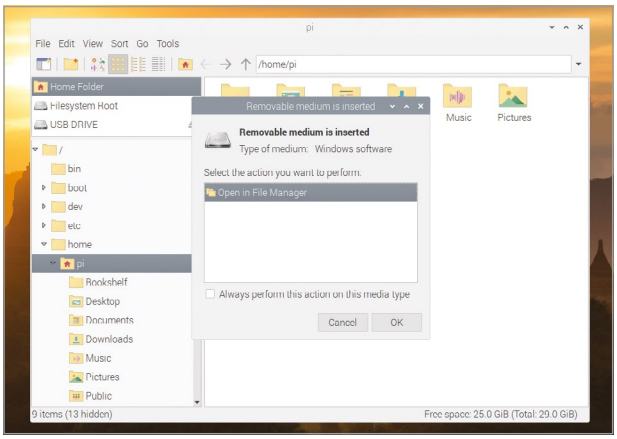
- **Pictures (billeder):** Denne mappe er specielt beregnet til billeder – eller som de kaldes i computerterminologi: *billedfiler*.

- **Public (til deling med andre):** Mens de fleste af dine filer er private, vil alt, hvad du lægger i denne mappe, kunne ses af andre brugere af Raspberry Pi, selvom de har deres eget brugernavn og adgangskode.

- **Templates (skabeloner):** Denne mappe indeholder skabeloner, blanke dokumenter med et grundlæggende layout eller struktur, der allerede er på plads, som du har oprettet eller er blevet installeret af dine programmer.

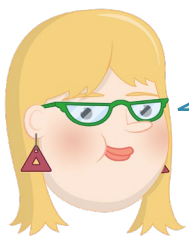
- **Videos (videoer):**En mappe til videoer. Det er også den mappe, som de fleste programmer til videoafspilning vil kigge i først.

Selve filhåndteringsvinduet er opdelt i to ruder: i den venstre rude vises mapperne på din Raspberry Pi, og i den højre rude vises filerne og undermapperne til den mappe, der er valgt i den venstre rude. Hvis du tilslutter et flytbar drev til USB-porten i din Raspberry Pi, bliver du spurgt, om du vil åbne den i Filhåndtering (**Figur 3-15**). Når du klikker på OK, vises de filer og mapper, der ligger på drevet, i Filhåndtering.



Figur 3-15: Isætning af et flytbart drev

Det er let at kopiere filer mellem microSD-kortet på Raspberry Pi og et flytbar drev: Åbn både din hjemmemappe og det flytbare i separate File Manager-vinduer, pege med musemarkøren på den fil, du vil kopiere, klik på filen med venstre museknap, og mens du stadig holder venstre museknap nede, skal du flytte musemarkøren hen til det andet vindue og slippe museknappen (se **Figur 3-16** på næste side). Denne manøvre kaldes ofte at *trække og slippe*.

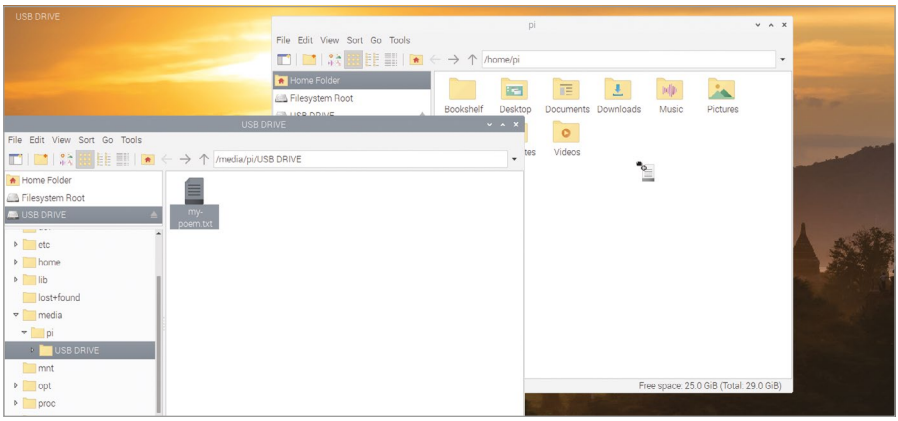


TASTATURGENVEJE

En tastaturgenvej som **CTRL+C** betyder, at du skal holde den første tast nede (**CTRL**), trykke på den anden tast (**C**) og derefter slippe begge taster.

En anden metode er at klikke på filen, klikke på menuen Rediger, vælge Kopiér, klikke på det andet vindue, klikke på menuen Rediger og vælge Indsæt.

Funktionen Klip, der ligeledes findes menuen Rediger, gør stort set det samme, bortset fra, at den fil, der kopieres, slettes fra sin oprindelige placering, efter at den er blevet kopieret til den nye placering. Begge funktioner kan også bruges vha. tastaturgenveje: tryk på **CTRL+C** for at kopiere eller **CTRL+X** for at klippe, og indsæt med **CTRL+V.**.



▲ **Figur 3-16:**Træk og slip af en fil

Når du er færdig med at prøve de forskellige funktioner, skal du lukke Filhåndtering ved at klikke på knappen Luk øverst til venstre i vinduet. Hvis du har mere end et vindue åbent, skal du lukke dem alle. Hvis du har tilsluttet et flytbart drev til din Raspberry Pi, skal du skubbe drevet ud ved at klikke på skub ud-knappen øverst til højre på skærmen, finde drevet på listen og klikke på det, før du frakobler det fysisk.

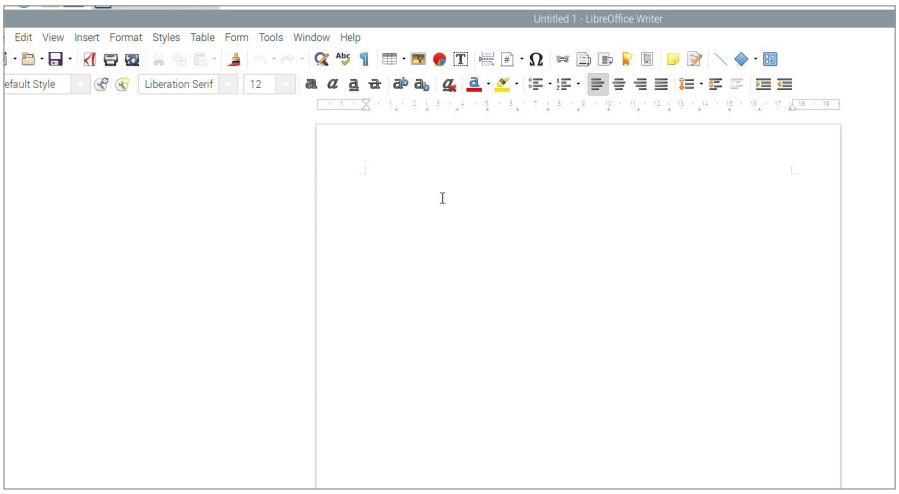
SKUB EKSTERNE DREV UD

Brug altid skub ud-knappen, inden du frakobler et eksternt drev. Hvis du ikke gør det, kan filerne på drevet blive ødelagt.

LibreOffice-pakken

For at prøve en anden ting, du kan gøre med Raspberry Pi, kan du klikke på hindbærmenuen, vælge Kontor og klikke på LibreOffice Writer. Dette indlæser tekstbehandlingsdelen af LibreOffice (**Figur 3-17**), der er en populær *kontorpakke* i stil med Microsoft Office eller Google Docs.

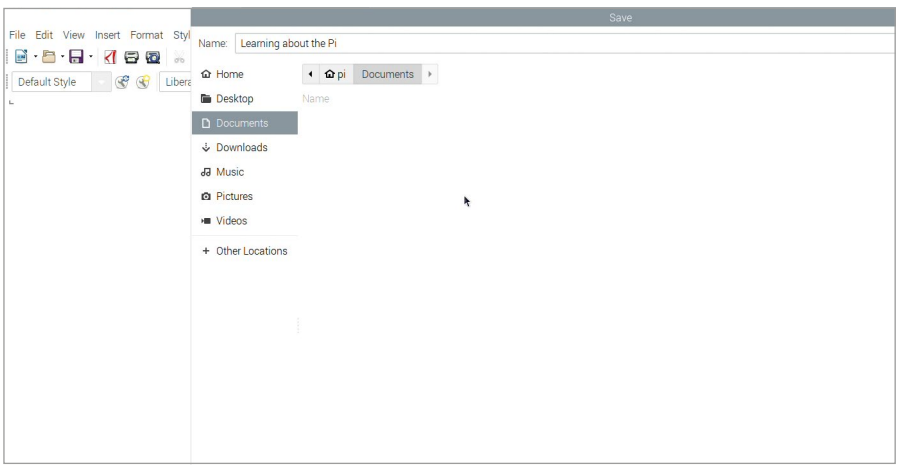
Bemærk: LibreOffice er muligvis ikke installeret som standard på alle enheder med Raspberry Pi OS. Hvis du ikke har den, kan du nemt installere den med værktøjet Recommended Software (anbefalet software) – se side 49.



Figur 3-17: LibreOffice Writer

Et tekstbehandlingsprogram kan bruges til at forfatte dokumenter og formatere dem, så de ser godt ud: du kan bl.a. ændre skrifttype, -farve, -størrelse, tilføje effekter og endda indsætte billeder, diagrammer, tabeller og andet indhold. Et tekstbehandlingsprogram kan også tjekke teksten for slåfejl og fremhæve stavefejl og grammatiske fejl med henholdsvis rødt og grønt, mens du skriver.

Begynd med at skrive et afsnit om, hvad du indtil nu har lært om Raspberry Pi og dens software. Eksperimenter med de forskellige ikoner øverst i vinduet for at se, hvad de gør. Se, om du kan gøre teksten større og ændre dens farve. Hvis du ikke er sikker på, hvordan du gør det, skal du blot flytte musemarkøren hen over ikonerne for at se et "værktøjstip", der beskriver, hvad ikonet gør. Når du er tilfreds med resultatet, skal du åbne menuen File (filer) og vælge Save (gem) for at gemme dit arbejde(**Figur 3-18**). Giv dokumentet et navn, og klik på knappen Save (gem).



Figur 3-18: Sådan gemmer du et dokument

GEM DIT ARBEJDE

Gør det til en vane at gemme dit arbejde løbende, mens du er i gang med det. Du vil takke os senere, hvis der f.eks. er strømafbrydelse, mens du arbejder på noget vigtigt!

LibreOffice Writer er kun en del af LibreOffice-pakken. De andre dele, som findes i samme Kontor-kategori som LibreOffice Writer, er:

- **LibreOffice Base:**En database er et værktøj til lagring af information, der hurtigt kan slås op, analyseres og behandles.
- **LibreOffice Calc**: Et regneark et værktøj til håndtering af tal og oprettelse af diagrammer og grafer.
- **LibreOffice Draw**: Et illustrationsprogram, der kan bruges til oprettelse af billeder og diagrammer.
- **LibreOffice Impress**: Et præsentationsprogram til oprettelse af dias ogvisning af slideshows.
- **LibreOffice Math**: Et værktøj til oprettelse af korrekt formaterede matematiske formler, som derefter kan bruges i andre dokumenter.

LibreOffice findes også til andre computere og operativsystemer. Hvis du kan lide at bruge den på din Raspberry Pi, kan du downloade den gratis fra **libreoffice.org** og installere på en computer med Microsoft Windows, Apple macOS eller Linux.

Klik på menuen Help (hjælp), hvis du vil vide mere om brug af LibreOffice. Ellers kan du lukke LibreOffice Writer ved at klikke på luk-knappen øverst til højre i vinduet.

HVIS DU HAR BRUG FOR HJÆLP

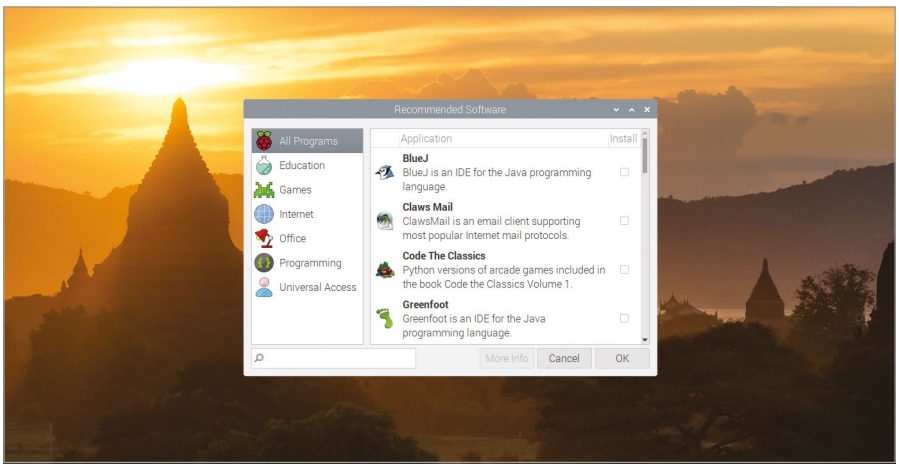
De fleste programmer har en Hjælp-menu, hvor du kan finde generelle oplysninger om programmet og typisk en vejledning om, hvordan du bruger det. Hvis du kommer i tvivl om, hvordan du bruger et program, er Hjælp-menuen altid et godt sted at starte.

Værktøjet Recommended Software

Selvom Raspberry Pi OS leveres med en bred vifte af præinstalleret software, er den kompatibel med endnu mere af slagsen. Et udpluk af nogle af de bedste programmer findes i værktøjet Recommended Software.

Bemærk, at værktøjet Recommended Software skal have forbindelse til internettet. Hvis din Raspberry Pi har forbindelse til internettet, kan du klikke på hindbærikonet, vælge Indstillinger og klikke på Recommended Software. Værktøjet indlæses og går i gang med at downloade oplysninger om tilgængelig software.

Efter nogle sekunder vises der en liste over kompatible softwarepakker (**Figur 3-19**). Denne software er, ligesom softwaren i hindbærmenuen, arrangeret i forskellige kategorier. Klik på en kategori i ruden til venstre for at se software fra den pågældende kategori, eller klik på All programs (alle programmer) for at se hele udvalget.

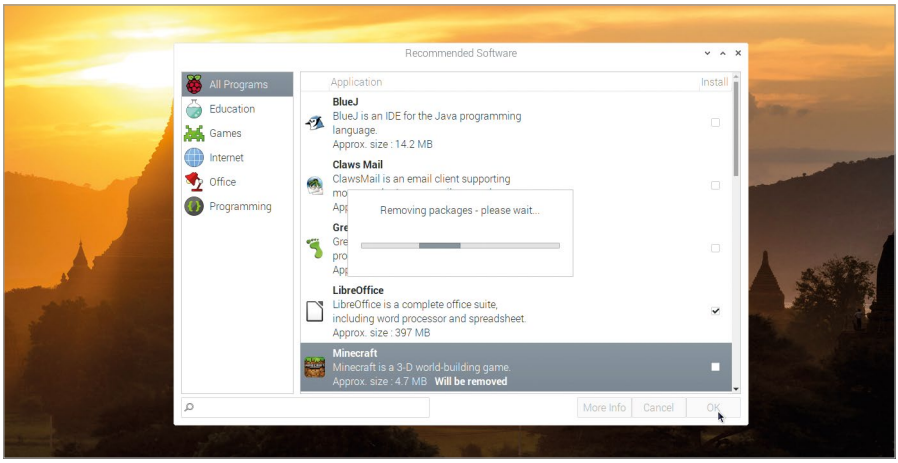


Figur 3-19: Værktøjet Recommended Software

Hvis der er et kryds ud for et program, er programmet allerede installeret på din Raspberry Pi. Hvis der ikke er noget kryds, kan du klikke på afkrydsningsfeltet ud for programmet for at markere, at programmet skal installeres. Du kan markere lige så mange programmer, som du vil, og så kan du installere dem alle på én gang. Men hvis det microSD-kort, du bruger, er for lille, har du muligvis ikke plads til dem alle.

Softwaren afinstalleres på samme måde: Find et program, der allerede har et flueben i afkrydsningsfeltet, og klik derefter på afkrydsningsfeltet for at fjerne fluebenet. Hvis du har klikket ved en fejltagelse eller fortrudt, skal du bare klikke igen for at sætte flueben igen.

Når du er færdig med at vælge softwaren, skal du klikke på OK for at starte installations- eller afinstallationsprocessen (se **Figur 3-20** på næste side). Efter download og installation af den software, du har valgt, vises der en dialogboks. Klik på OK for at lukke værktøjet Recommended Software.

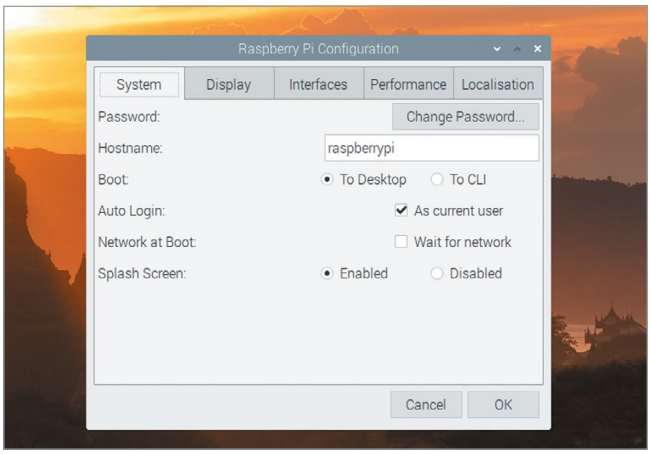


Figur 3-20: Afinstallation af software

Et yderligere værktøj til installation eller afinstallation af software, værktøjet Add/Remove Software (tilføj/fjern software), findes i den samme kategori Indstillinger i Raspberry Pi OS-menuen. Her finder du et større udvalg af software, dog uden at denne software er blevet kontrolleret af Raspberry Pi Foundation.

Værktøjet Raspberry Pi Configuration

Det sidste program, du lærer om i dette kapitel, er værktøjet Raspberry Pi Configuration, der minder meget om den velkomstguide, du brugte i starten, og giver dig mulighed for at ændre forskellige indstillinger i Raspberry Pi OS. Klik på hindbærikonet, vælg kategorien Indstillinger, og klik på Raspberry Pi Configuration (**Figur 3-21**).



Figur 3-21: Værktøjet Raspberry Pi Configuration

Værktøjet er opdelt i fem faner. Den første fane er System, hvor du kan ændre adgangskoden til din konto, angive værtsnavnet, dvs. det navn på din Raspberry Pi, der bruges på dit lokale wi-fi- eller kabelnetværk, og ændre en række andre indstillinger. De fleste af dem behøver du dog ikke at ændre. Klik på fanen Display (skærm) for at åbne den næste kategori. Her kan du ændre skærmindstillingerne, hvis det er nødvendigt, så det passer til dit tv eller din computerskærm.

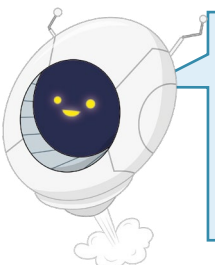
FLERE OPLYSNINGER

Formålet med denne korte oversigt er at give dig et fingerpeg om, hvordan du bruger værktøjet. Der findes mere detaljerede oplysninger om alle indstillinger i **Appendiks E – Værktøjet Raspberry Pi Configuration** .

Under fanen Interface finder du en række indstillinger, som alle er deaktiveret som standard. Du skal kun ændre disse indstillinger, hvis du tilføjer ny hardware, f.eks. et Raspberry Pi-kameramodul, og kun, hvis hardwareproducenten skriver det i deres dokumentation. Der er dog nogle undtagelser fra denne regel: SSH, der aktiverer en "sikker skal", som gør det muligt for dig at logge ind på din Raspberry Pi fra en anden computer på netværket vha. en SSH-klient, VNC, der aktiverer en "virtuel netværkscomputer" og gør det muligt or dig at se og styre Raspberry Pi OS-skrivebordet fra en anden computer på netværket vha. en VNC-klient, og til sidst og ikke mindst Remote GPIO, der gør det muligt for dig at bruge GPIO-stikkene på Raspberry Pi-kortet – som du kan læse mere om i **Kapitel 6, Fysisk programmering med Scratch og Python** – fra en anden computer på netværket.

Klik på fanen Performance (ydeevne) for at åbne den fjerde kategori. Her kan du indstille mængden af den hukommelse, der bruges af Raspberry Pi-computerens grafikkort (GPU) og for nogle modeller øge Raspberry Pi-computerens ydelse via en proces kaldet *overclocking*. Som nævnt før er det dog bedst at lade disse indstillinger være, som de er, medmindre du ved, at det er nødvendigt at ændre dem.

Til sidst kan du klikke på fanen Localisation (sprog) for at åbne den sidste kategori. Her kan du ændre din lokalitet, hvilket igen styrer ting som det sprog, der bruges i Raspberry Pi OS, og hvordan numre vises, ændre tidszonen, ændre tastaturlayoutet og vælge land, så wi-fi-netværk kan fungere korrekt. Men lige nu kan du bare klikke på Cancel (annuller) for at lukke værktøjet uden at lave nogen ændringer.



ADVARSEL!

Forskellige lande har forskellige regler for, hvilke frekvenser der kan bruges af en wi-fi-radio. Hvis du indstiller wi-fi-landet i værktøjet Raspberry Pi Configuration til et andet land end det, hvor du befinder dig i, vil du højst sandsynligt få problemer med at oprette forbindelse til netværket, og det kan endda være en overtrædelse af licenslove for radiosignaler – så gør det ikke.

Sådan slukker du

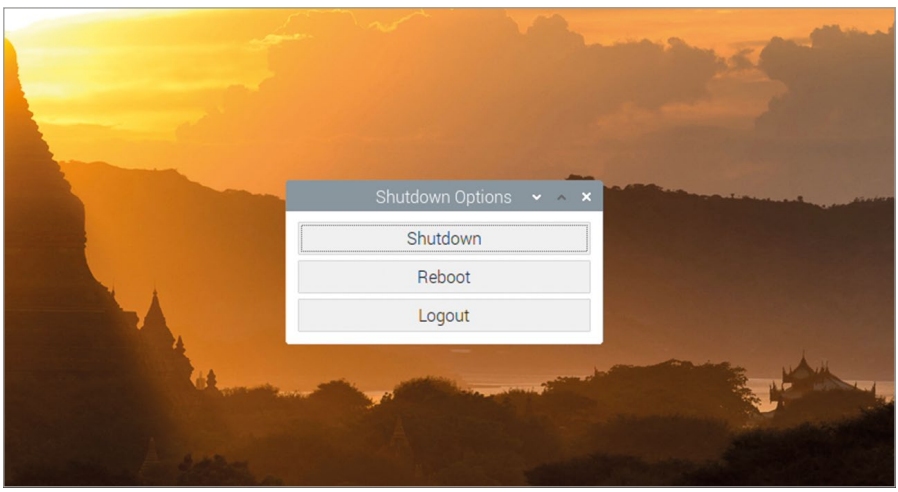
Nu har du udforsket Raspberry Pi OS-skrivebordet, og det er på tide at lære endnu en vigtig ting: Hvordan du korrekt slukker for din Raspberry Pi. Som alle andre computere opbevarer Raspberry Pi de filer, du arbejder på, i *flygtig hukommelse*, dvs. hukommelse, der tømmes, når systemet slukkes. Når du gemmer de dokumenter, du opretter, overføres de fra den flygtige hukommelse til *permanent hukommelse* i form af microSD-kortet. Når dokumentet er gemt, går det ikke tabt, når computeren slukkes.

Men de dokumenter, du arbejder på, er ikke de eneste filer, der er åbne. Raspberry Pi OS holder selv et antal filer åbne, mens det kører, og hvis du afbryder strømmen til din Raspberry Pi, mens disse filer stadig er åbne, risikerer du, at operativsystemet bliver beskadiget og skal installeres forfra.

For at forhindre, at dette sker, skal du altid sørge for at sende en kommando til Raspberry Pi OS om at gemme alle de åbne filer og gøre sig klar til at blive slukket – en proces kendt som *nedlukning* af operativsystemet.

Klik på hindbærikonet øverst til venstre på skrivebordet, og klik derefter på Shutdown (luk ned). Der vises et vindue med tre valgmuligheder (**Figur 3-22**): Shutdown (luk ned), Reboot (genstart) og Logout (log af). Shutdown er den funktion, du kommer til at bruge mest. Når du klikker på dette menupunkt, sender du en kommando til Raspberry Pi OS om at lukke alle åbne programmer og filer og derefter slukke for Raspberry Pi-computeren. Når skærmen er blevet sort, skal du vente et par sekunder, til den blinkende grønne indikator på Raspberry Pi slukkes, hvorefter det er sikkert at slukke for strømforsyningen.

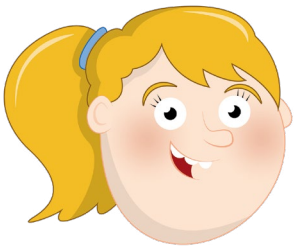
For at tænde Raspberry Pi igen skal du bare tage strømkablet ud og sætte det i igen, eller slukke og tænde for stikkontakten.



▲ **Figur 3-22:** Sådan slukker du for Raspberry Pi

Funktionen Reboot (genstart) udfører en lignende proces som Shutdown, idet alt lukkes ned, men i stedet for at slukke for Raspberry Pi-computeren, genstartes den – dvs. det svarer nogenlunde til, at du vælger Shutdown og derefter tager strømkablet ud og sætter det i igen. Funktionen Reboot bruges typisk, når du foretager visse ændringer, der kræver genstart af operativsystemet – for eksempel når du installerer visse opdateringer til kernesoftwaren – eller hvis noget software er *gået ned* og har gjort Raspberry Pi OS ustabil.

Og til sidst og ikke mindst er der Logout (log af). Denne funktion er nyttig, hvis du har flere brugerkonti på din Raspberry Pi. Når du vælger denne funktion, lukkes alle åbne programmer, og du får vist en loginskærm, hvor du skal indtaste dit brugernavn og din adgangskode. Hvis du vælger Logout ved en fejl og ønsker at komme tilbage, skal du blot skrive "pi" som brugernavn samt den adgangskode, du har valgt i velkomstguiden, der blev gennemgået i starten af dette kapitel.



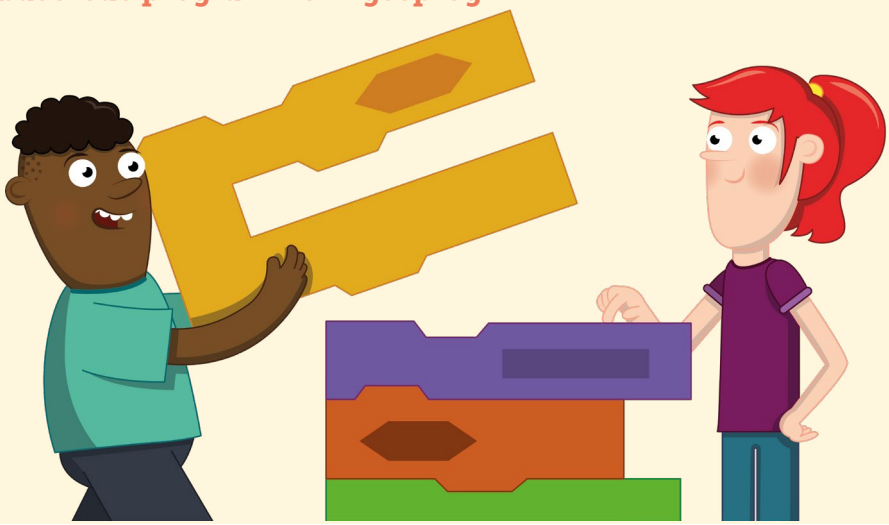
ADVARSEL!

Tag aldrig strømkablet ud af en Raspberry Pi-computer uden først at lukke computeren ned. Hvis du gør det, risikerer du at ødelægge operativsystemet, og du kan miste alle filer, du har oprettet eller downloadet.

Kapitel 4

Programmering med Scratch 3

Sådan starter du med at kode med Scratch – det brikbaserede programmeringssprog

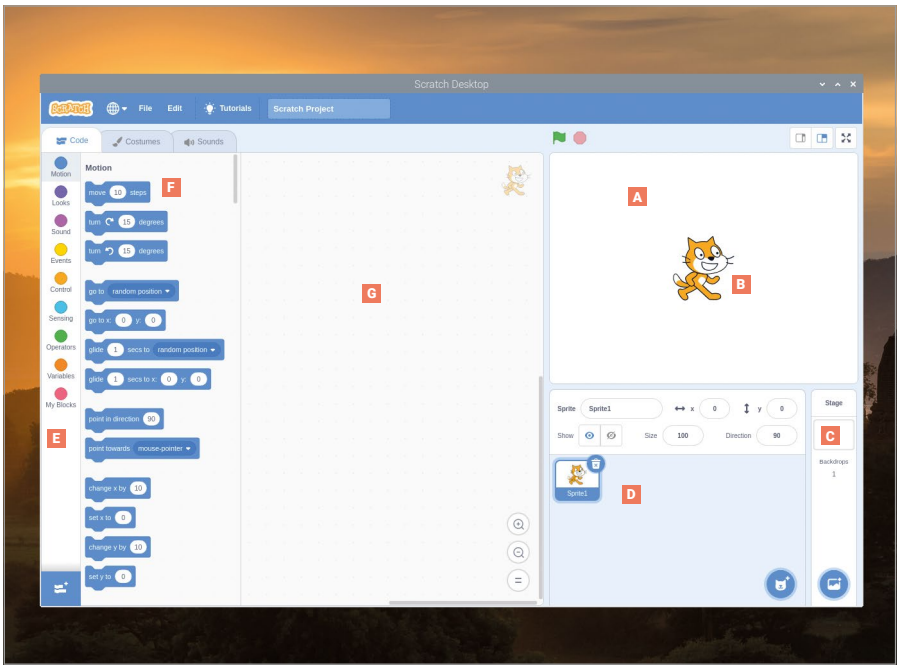


Raspberry Pi drejer sig ikke kun om at bruge software, som andre har lavet – det handler også om at lave sin egen software. Og her er det kun fantasien, der sætter grænser. Uanset, om du har tidligere erfaring med at lave dine egne programmer – en proces kendt som programmering eller kodning – eller ej, er Raspberry Pi en fantastisk platform til dette formål.

Nøglen til kodning på Raspberry Pi er Scratch, som er et visuelt programmeringssprog udviklet af MIT (Massachusetts Institute of Technology). Mens i traditionelle programmeringssprog skal du skrive de instruktioner, computeren skal udføre, i form af tekst – på samme måde, som når du skriver en kageopskrift ned – så kan du i Scratch lave et program ved at sammensætte de forskellige trin vha. såkaldte brikker, som er foruddefinerede stykker kode arrangeret i puslespilsbrikker med forskellige farver.

Scratch er et fantastisk programmeringssprog til folk i alle aldre, der gerne vil lære at programmere. Men selvom princippet lyder meget simpelt, skal du ikke undervurdere dets muligheder, for Scratch er en meget effektiv og fuldt funktionel programmeringsplatform, hvor du kan skabe alt fra simple spil og animationer til komplekse, interaktive og robotstyrede projekter.

Introduktion af grænsefladen i Scratch 3



A **Scene** – Som skuespillere i et skuespil bevæger dine sprites sig rundt på scenen, mens de styres af dit program.

B **Sprite** – De figurer eller objekter, du styrer i et Scratch-program, kaldes sprites, og de hører til på scenen.

C **Sceneværktøjer** – Disse værktøjer bruges til at ændre scenen. Du kan f.eks. bruge et af dine egne billeder som baggrund.

D **Spritelisten** – Alle de sprites, du har skabt eller indlæst i Scratch, vises i denne del af vinduet.

E **Brikpalet**– Alle de brikker, der kan bruges i dit program, vises i brikpaletten og er arrangeret i farvekategorier.

VERSIONER AF SCRATCH

I skrivende stund findes der tre versioner af Scratch til Raspberry Pi OS: Scratch 1, 2 og 3. Alle versionerne findes i afsnittet Programmering i Raspberry Pi OS-menuen. Dette kapitel omhandler Scratch 3. Bemærk, at Scratch 3 kun kan køre på Raspberry Pi 4. Hvis du vil bruge Scratch 2 i stedet, kører denne version ikke på Raspberry Pi Zero, Model A, A+, B eller B+.

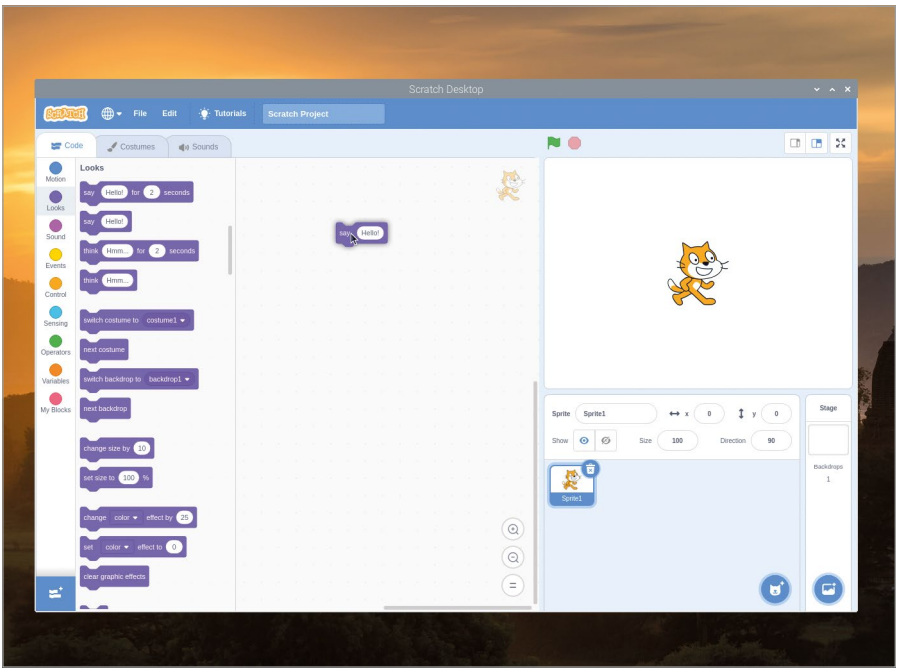
F **Brikker** – Foruddefinerede kodemoduler, som du kan bruge til at opbygge dit program trin for trin.

G **Kodeområde** – Det er her, du bygger dit program op ved at trække og slippe brikker fra brikpaletten for at oprette scripts.

Dit første Scratch-program: Hello, World!

Scratch 3 indlæses på samme måde som alle andre programmer i Raspberry Pi: Klik på hindbærikonet for at åbne Raspberry Pi OS-menuen, peg med musen på afsnittet Programmering, og klik på Scratch 3. Efter et par sekunder indlæses Scratch 3-brugergrænsefladen.

I de fleste programmeringssprog skal instruktionerne til computeren om at gøre forskellige ting gøres på skrift. Men i Scratch foregår det anderledes. Start med at klikke på kategorien Udseende i brikpaletten i venstre side af Scratch-vinduet. Dette viser de brikker, der findes i denne kategori, og de er alle lilla. Find brikken sig Hej!, klik på den, og mens du holder venstre museknap nede, skal du trække den over til kodeområdet i midten af Scratch-vinduet og slippe museknappen (**Figur 4-1**).



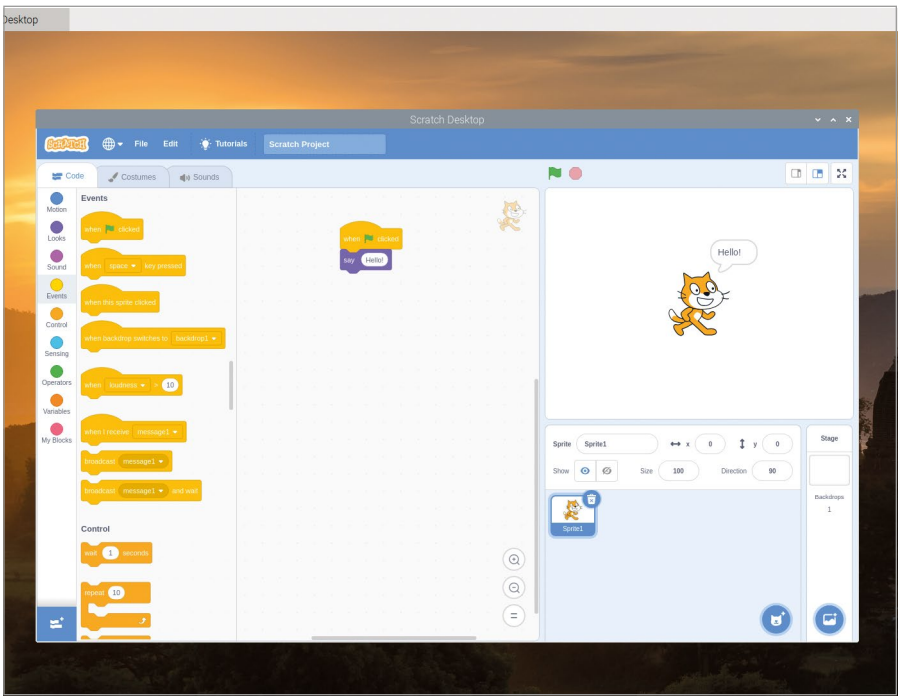
▲ **Figur 4-1:** Træk og slip brikken oven på kodeområdet

Se på formen på den brik, du lige har flyttet: den har et hul øverst og en matchende del, der stikker ud i bunden. Princippet er det samme, som når du lægger et puslespil – brikken kan forbindes med en anden brik foroven og en anden forneden. Det, der sættes foroven, kaldes en *udløser*.

Klik på den guldfarvede kategori Hændelser i brikpaletten, og træk brikken når der klikkes på – kendt som en *hat* – over på kodeområdet. Placer brikken, så den del, der stikker ud af bunden på den, passer ind i hullet øverst på brikken sig Hej!, og når

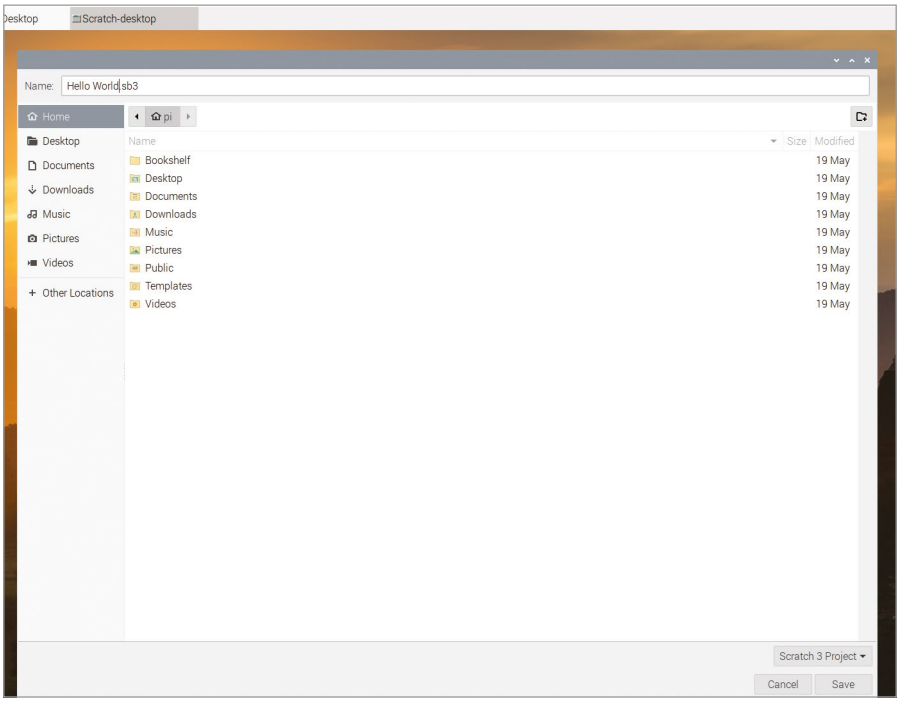
der vises en hvid kontur, skal du slippe museknappen. Du behøver ikke være præcis – når du kommer tæt nok på, klikker brikken på plads ligesom i et puslespil. Hvis den ikke gør det, skal du klikke på den igen, og mens du holder museknappen nede, skal du justere brikkens position, indtil den klikker på plads.

Programmet er nu færdigt. For at se det i aktion, dvs. *køre* programmet, skal du klikke på det grønne flag over scenevinduet. Hvis alt er godt, vil katten på scenen hilse på dig med et muntert "Hej!" (**Figur 4-2**) – dit første program er en succes!

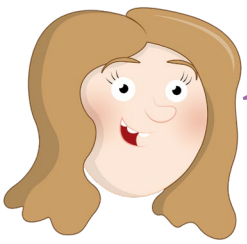


▲ **Figur 4-2:**Klik på det grønne flag over scenen, og katten vil sige "Hej"

Inden du går videre, skal du give dit program et navn og gemme det. Klik på menuen Fil, og vælg "Gem på din computer". Indtast et navn, og klik på knappen Save (gem) (**Figur 4-3**).



▲ **Figur 4-3:** Giv dit program et passende navn



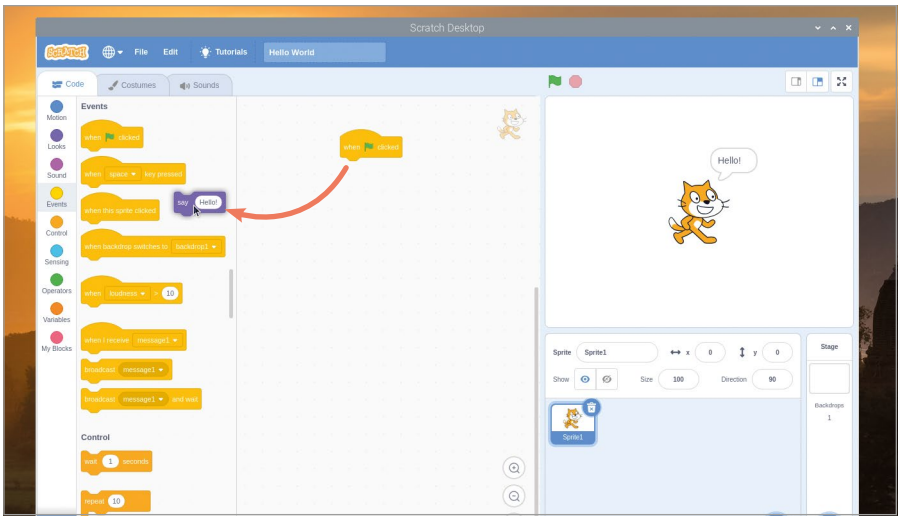
HVAD KAN DET SIGE?

Nogle brikker i Scratch kan ændres. Prøv at klikke på ordet "Hej!" og skrive noget andet, og klik derefter på det grønne flag igen. Hvad sker der på scenen?

Næste trin: sekvensering

Dit program har i øjeblikket kun to brikker, og det kun én instruktion: at sige "Hej!", hver gang programmet køres ved at klikke på flaget. For at gøre mere skal du lære lidt om *sekvensering*. Et computerprogram er i princippet en række instruktioner, lidt ligesom en madopskrift. De enkelte instruktioner følger efter hinanden i en logisk rækkefølge, der kaldes en *lineær sekvens*.

Klik på brikken sig Hej!, og træk den fra kodeområdet og tilbage til brikpaletten (**Figur 4-4**). Dette fjerner brikken den fra dit program, så kun udløserbrikken når der klikkes på er tilbage.



Figur 4-4: For at slette en brik skal du bare trække den væk fra kodeområdet

Klik på kategorien Bevægelse i brikpaletten, klik på brikken **gå 10 trin**, og træk den over på kodeområdet, så den falder i hak under udløserbrikken. Som navnet antyder, fortæller dette din sprite (katten), at den skal bevæge sig et antal trin i den retning, den i øjeblikket vender mod.

Tilføj flere instruktioner til dit program for at oprette en sekvens. Klik på den lyserøde kategori Lyd, klik på brikken **spil lyden Mijav indtil færdig**, og træk den med musen, så den falder i hak under brikken **gå 10 trin**. Fortsæt: Klik på kategorien Bevægelse igen, træk en anden brik **gå 10 trin**, og placer den under lydbrikken. Klik derefter på "10", og lav det om til "-10", så brikken ser nu sådan ud: **gå -10 trin** .

Klik på det grønne flag over scenen for at køre programmet. Katten bevæger sig til ud højre, mjaver (sørg for, at du har tilsluttet højtalere eller hovedtelefoner for at høre det) og vender tilbage til startpositionen. Klik på flaget igen, og katten gentager disse handlinger.

Tillykke: du har oprettet en instruktionssekvens, som Scratch kører igennem en ad gangen, fra toppen til bunden. Scratch kan ganske vist kun køre én instruktion ad gangen, men den gør det meget hurtigt. Prøv nu at slette brikken spil lyden Mijav indtil færdig ved at klikke på den nederste brik gå -10 trin, trække den lidt væk for at løsne den og trække brikken spil lyden Mijav indtil færdig tilbage til brikpaletten. Udskift den derefter med den enklere brik spil lyden Mijav, og træk til sidst brikken gå -10 trin tilbage til bunden af programmet.

Når du nu klikker på det grønne flag for at køre programmet, set katten slet ikke ud til at bevæge sig. I virkeligheden bevæger den sig faktisk, men den flytter sig så hurtigt tilbage, at den ser ud til at stå stille. Dette skyldes, at brikken spil lyden Mijav ikke venter på, at lyden er færdig med at spille, før det næste trin udføres, og fordi Raspberry Pi "tænker" så hurtigt, kører den næste instruktion, før du overhovedet når at se katten bevæge sig. Der er en anden måde at løse det på ud over at bruge brikken spil lyden Mijav indtil færdig: Klik på den lysorange kategori Kontrol på brikpaletten, og træk brikken vent 1 sekunder, og placer den mellem brikken spil lyden Mijav og den nederste brik gå -10 trin.

Klik på det grønne flag for at køre dit program igen. Nu venter katten et sekund, efter den har bevæget sig til højre, før den vender tilbage til start. Denne funktion kaldes en *forsinkelse*, og er en meget vigtig funktion til at kontrollere, hvor lang tid din instruktionssekvens skal vare.

UDFORDRING: TILFØJ FLERE TRIN

Prøv at tilføje flere trin til din sekvens og ændre værdierne i de eksisterende trin. Hvad sker der, hvis antallet af trin i en bevægelsesbrik ikke svarer til antallet af trin i en anden? Hvad sker der, hvis du prøver at afspille en lyd, mens en anden lyd stadig afspilles?

Løkker

Den sekvens, du har oprettet, kører kun én gang: Du klikker på det grønne flag, katten bevæger sig og miaver, og programmet stopper, indtil du klikker på det grønne flag igen. Du kan dog få programmet til at køre videre ved hjælp af den kontrolbrik i Scratch, der udfører en såkaldt *løkke*.

Klik på kategorien Kontrol i brikpaletten, og find brikken for evigt. Træk den over i kodeområdet, og placer den mellem når der klikkes på og gå 10 trin.

Bemærk at den C-formede evighedsbrik automatisk udvider sig og lægger sig omkring de andre brikker i sekvensen. Når du nu klikker på det grønne flag er det tydeligt at se, hvad brikken for evigt gør: Mens programmet før kørte én gang, kører det nu igen og igen – bogstaveligt talt uden ende. I programmering kaldes dette fænomen en *uendelig løkke* – altså en løkke, der aldrig slutter.

Hvis lyden af katten, der mjaver i det uendelige, bliver lidt meget, kan du blot klikke på den røde ottekant ved siden af det grønne flag over scenen, så stopper programmet med at køre. Du kan

ændre typen af løkken ved at trække den første brik af typen gå 10 trin sammen med brikkerne under det ud af brikken for evigt og placere dem direkte under brikken når der klikkes på . Træk brikken for evigt over til brikpaletten for at slette den, og træk derefter brikken gentag 10, og placer den under når der klikkes på, så den omslutter de andre brikker.

Klik på det grønne flag for at køre det nye program. Først ser det ud til, at programmet gør det samme som i den forrige version – dvs. instruktionssekvensen gentages igen og igen. Men denne gang afsluttes løkken efter ti gentagelser. Dette kaldes en *definitiv løkke*, hvor du definerer, hvornår løkken skal afsluttes. Løkker er avancerede værktøjer, og de fleste programmer – især spil og registreringsprogrammer – benytter sig af både uendelige og definitive løkker.

HVAD SKER DER, HVIS?

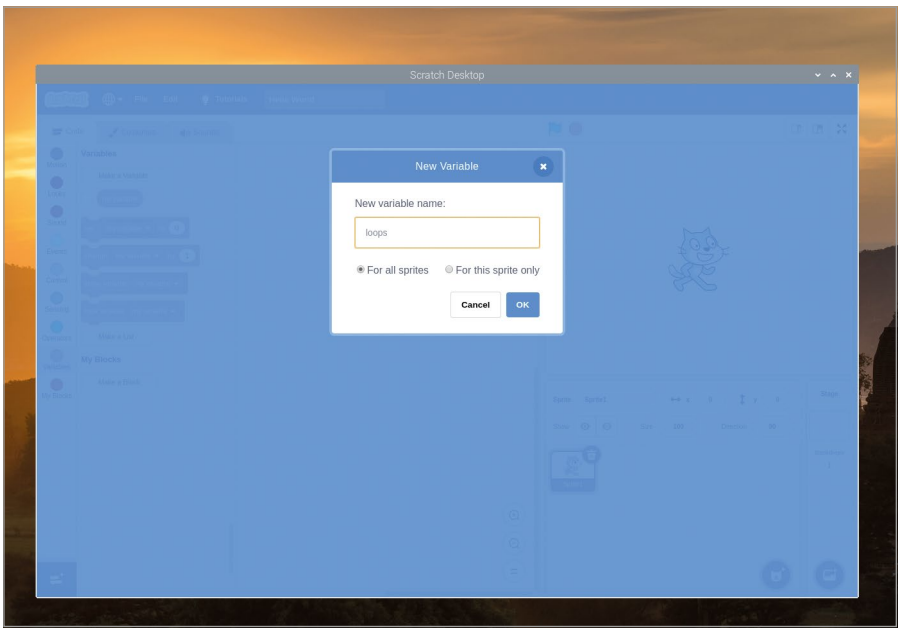
Hvad sker der, hvis du gør tallet i en løkkebrik større? Hvad sker der, hvis du gøre det mindre? Hvad sker der, hvis du bruger 0 som værdi i en løkkebrik?

Variabler og betingelser

De sidste to begreber, vi skal nævne, før du for alvor kaster dig ud i programmering med Scratch, er nært beslægtede og kaldes *variabler* og *betingelser* . En variabel er, som navnet antyder, en værdi, der kan variere – dvs. være forskellig – over tid, og mens den kontrolleres af programmet. En variabel har to hovedegenskaber: dens navn og den værdi, den repræsenterer. Værdien behøver ikke at være et tal: det kan være tal, tekst, et sandt- eller falsk-udtryk, eller den kan være tom – kendt som en *nulværdi*.

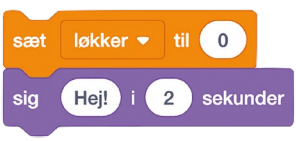
Variabler er avancerede værktøjer. Tænk på de ting, du skal have styr på i et spil: Det antal liv, du har tilbage, hastigheden på et objekt i bevægelse, det niveau, du spiller på, og pointtavlen. Alle disse ting er variabler.

Klik først på menuen Fil, og gem dit eksisterende program ved at klikke på "Gem på din computer". Hvis du allerede har gemt programmet, bliver du spurgt, om du vil overskrive det og erstatte den gamle gemte kopi med den opdaterede version. Klik så igen på Fil, og vælg Ny for at starte et nyt, tomt projekt. Når du bliver spurgt, om du vil erstatte indholdet af det aktuelle projekt, skal du klikke på OK. Klik på den mørkorange kategori Variabler i brikpaletten, og klik på knappen "Lav en variabel". Skriv "løkker" som navnet på variablen (**Figur 4-5**), og klik på OK. Der kommer en række nye brikker i brikpaletten.



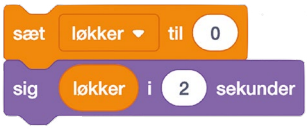
▲ **Figur 4-5:** Giv din nye variabel et navn

Træk brikken sæt løkker til 0 over til kodeområdet. Dette fortæller programmet, at det skal *initialisere* variablen med værdien 0. Klik så på kategorien Udseende i brikpaletten, og træk brikken sig Hej! i 2 sekunder over til kodeområdet, og placer den under brikken sæt løkker til 0.

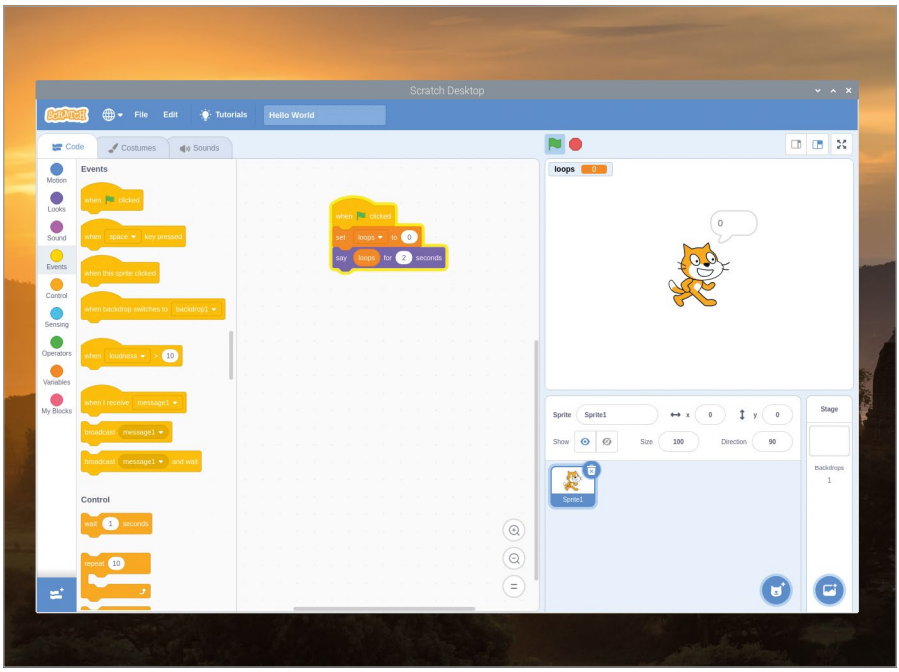


Som nævnt tidligere, får brikken sig Hej! katten til at sige, hvad der end står på brikken. I stedet for at skrive meddelelsen på selve brikken, kan du bruge en variabel i stedet. Klik igen på kategorien Variabler i brikpaletten, og træk den afrundede brik løkker – en såkaldt

rapportbrik, der ligger øverst på listen og har et afkrydsningsfelt – over ordet "Hej!" på brikken sig Hej! i 2 sekunder. Nu har du fået en ny brik: sig løkker i 2 sekunder.



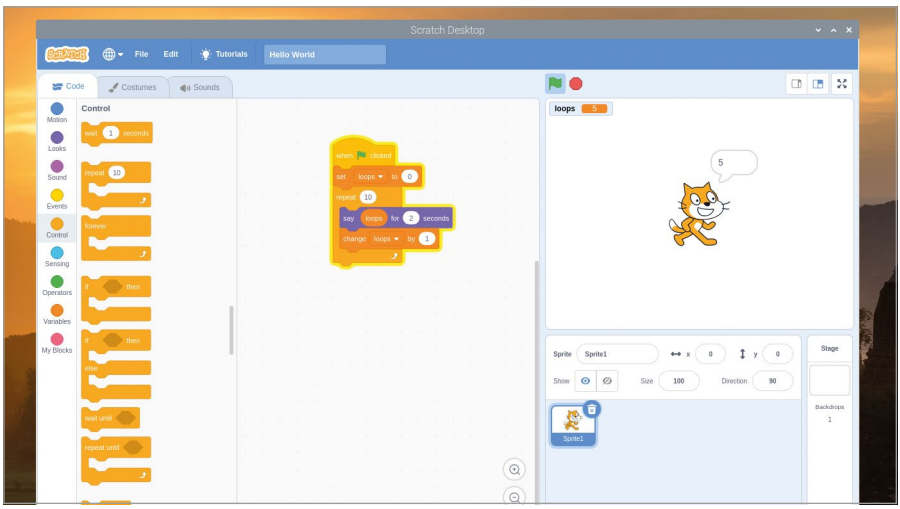
Klik på kategorien Hændelser i brikpaletten, træk brikken når der klikkes på over til kodeområdet, og placer den øverst i din briksekvens. Når du så klikker på det grønne flag over scenen, vil katten sige "0" (**Figur 4-6**) – hvilket er den værdi, du gav variablen "løkker".



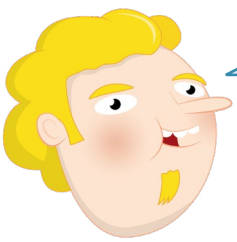
▲ **Figur 4-6:** Denne gang siger katten værdien af variablen

Men formålet med variabler er, at de skal ændres. Klik på kategorien Variabler i brikpaletten, træk brikken ændre løkker med 1 over til kodeområdet, og placer den nederst i din briksekvens. Klik derefter på kategorien Kontrol, træk brikken gentag 10 over til kodeområdet, og placer den sådan, at den starter lige under brikken sæt løkker til 0 og omslutter de resterende brikker i sekvensen.

Klik på det grønne flag igen. Denne gang tæller katten fra 0 til 9. Dette fungerer, fordi programmet ændrer, eller *tilpasser*, variablen: hver gang løkken køres, lægger programmet 1 til værdien i variablen "løkker" (**Figur 4-7**).



Figur 4-7: Løkken gør, at katten tæller opad



AT TÆLLE FRA NUL

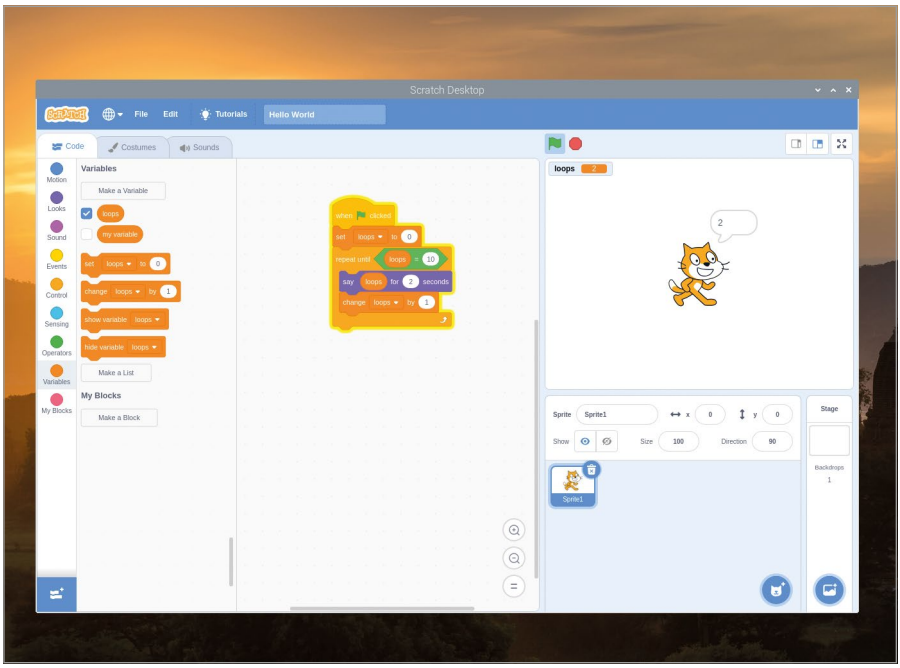
Selvom den løkke, du har lavet, kører ti gange, tæller katten kun op til ni. Det er, fordi vores variabel starter med nul. Når man medregner tallet nul, er der ti tal fra nul og ni, så programmet stopper, før katten når til "10". Hvis du vil have katten til at tælle til 10, skal du give variablen en startværdi på 1 i stedet for 0.

Du kan gøre mere med en variabel end at ændre dens værdi. Træk brikken sig løkker i 2 sekunder ud af briksekvensen gentag 10, og placer den under brikken gentag 10. Træk brikken gentag 10 til brikpaletten for at slette den, og erstat den derefter med brikken gentag indtil, og sørg for, at den er fastgjort til bunden af brikken sæt løkker til 0 og omslutter de to andre brikker i sekvensen. Klik på den grønne kategori Operatorer i brikpaletten, træk den diamantformede brik (=) over i kodeområdet, og placer den i den diamantformede åbning i brikken gentag indtil.

Brikkerne i kategorien Operatorer gør det muligt at sammenligne to værdier, inklusive variabler. Klik på kategorien Variabler, træk rapportbrikken løkker til den tomme åbning i operatorbrikken (=), og klik derefter på feltet med tallet "50", og lav det om til "10".

Når du så klikker på det grønne flag over scenen, fungerer programmet på samme måde som før: katten tæller fra 0 til 9 (**Figur 4-8**), og når den er færdig, stopper programmet. Dette

sker, fordi brikken gentag indtil fungerer nøjagtigt på samme måde som brikken gentag 10, men i stedet for at tælle antallet af løkker, sammenligner den værdien af variablen "løkker" med den værdi, som du indtastede i højre side af brikken. Når værdien af variablen "løkker" når op på 10, stopper programmet.

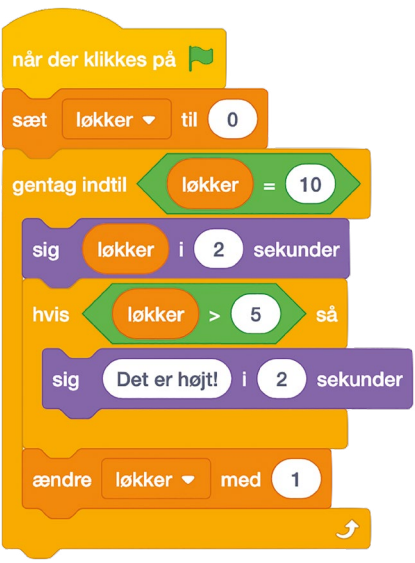


▲ **Figur 4-8:**Brug af en "gentag indtil"-brik sammen med en sammenligningsoperator

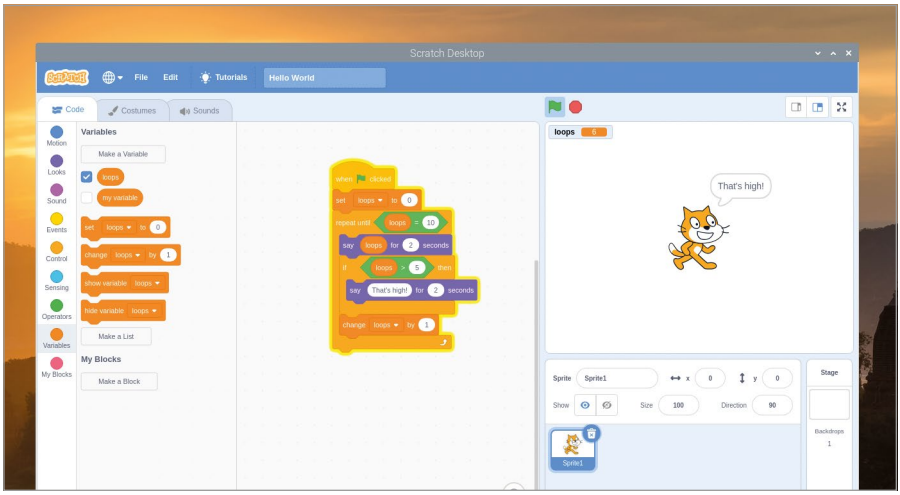
Denne funktion kaldes en *sammenligningsoperator*, fordi den sammenligner to værdier. Klik på kategorien Operatorer i brikpaletten, og find de to andre diamantformede brikker over brikken med lighedstegnet "=". Disse brikker er også sammenligningsoperatorer: "<" sammenligner to værdier og udløses, når værdien til venstre er mindre end den til højre, og ">" udløses, når værdien til venstre er større end den til højre.

Klik på kategorien Kontrol i brikpaletten, find brikken hvis så, træk den over til kodeområdet, og placer den direkte under brikken sig løkker i 2 sekunder. Den omslutter brikken ændre løkker med 1 automatisk, så klik på den, og fastgør den til bunden af brikken hvis så i stedet. Klik på kategorien Udseende i brikpaletten, og træk brikken sig Hej! i 2 sekunder over til kodeområdet, og placer den inde i brikken hvis så. Klik på kategorien Operatorer i brikpaletten, træk brikken ( ) > ( ) over i kodeområdet, og placer den i den diamantformede åbning i brikken hvis så.

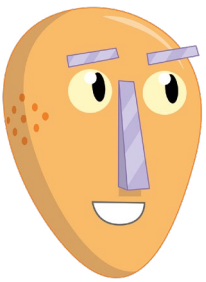
Brikken hvis så er en betinget brik, dvs. de brikker, der placeres inde i den, vil kun blive kørt, når en bestemt betingelse er opfyldt. Klik på kategorien Variabler i brikpaletten, træk rapportbrikken løkker til den tomme åbning i brikken > , og klik på feltet med tallet "50", og lav det om til "5". Til sidst skal du klikke på ordet "Hej!" i brikken sig Hej! i 2 sekunder og skrive "Det er højt!".



Klik på det grønne flag. I starten fungerer programmet som før med katten, der tæller fra nul og opad. Men når den når til 6 – hvilket er det første tal, der er større end 5 – udløses brikken hvis så, og katten kommer med en kommentar om, at tallet er blevet højt (**Figur 4-9**). Tillykke: Du kan nu bruge variabler og betingelser!



▲ **Figur 4-9:** Katten kommer med en kommentar, når tællingen når til 6



UDFORDRING: HØJT OG LAVT

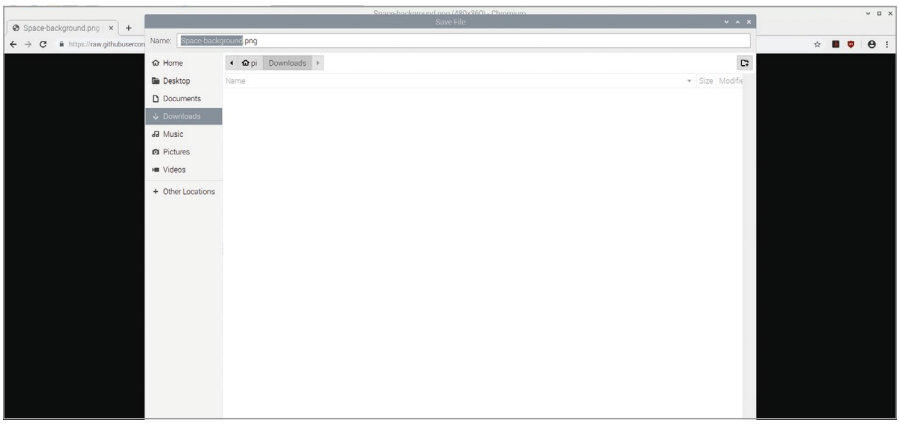
Hvordan ville du ændre programmet, så katten kommenterer lave tal (under 5) i stedet? Kan du ændre det, så katten kommenterer både høje og lave tal? Prøv dig frem med hvis så -brikker!

Projekt 1: Astronauten, der måler dine reaktioner

Nu har du fået en ide om, hvordan Scratch fungerer, og det er tid til at prøve noget mere interaktivt. Lad os lave en timer, der måler dine reaktioner. Til formålet bruger vi den britiske ESA-astronaut Tim Peake og hans tid ombord på den internationale rumstation.

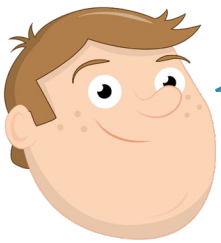
Hvis du vil beholde det program, vi lige har øvet os på, skal du bare gemme det på din computer. Åbn derefter et nyt projekt ved at klikke på Fil og Ny. Inden vi begynder, skal du give projektet et navn ved at klikke på Fil og "Gem på din computer". Kald projektet "Astronauten, der måler dine reaktioner".

Vi skal bruge to billeder til projektet: et baggrundsbillede og et billede, der skal bruges som en sprite. De findes ikke i samlingen af det materiale, der er installeret med Scratch, så vi skal lige downloade dem først. Klik på hindbærikonet for at indlæse Raspberry Pi OS-menuen, vælg internetkategorien, og klik på Chromium Web Browser. Når browseren er indlæst, skal du skrive **rpf.io/astronaut-backdrop** på adresselinjen og derefter trykke på **ENTER**-tasten. Højreklik på billedet af rummet, klik på "Save image as..." (gem billede som), og klik så på knappen Save (gem) (**Figur 4-10**). Klik igen på adresselinjen, skriv **rpf.io/astronaut-sprite**, og tryk på **ENTER**.



Figur 4-10: Gem baggrundsbilledet

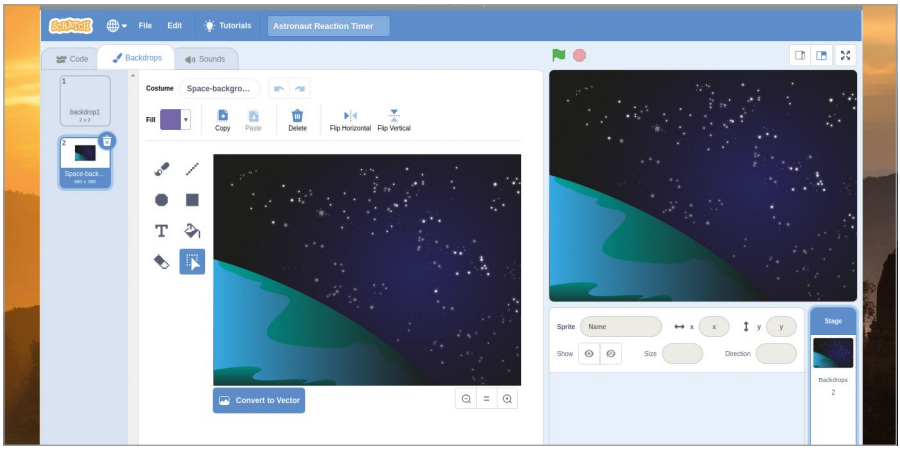
Højreklik på billedet af Tim Peake, klik på "Save image as..." (gem billede som), vælg mappen Downloads, og klik så på knappen Save (gem). Når du har gemt de to billeder, kan du lukke Chromium eller lade den være åben og bruge proceslinjen til at skifte tilbage til Scratch 3.



BRUGERGRÆNSEFLADE

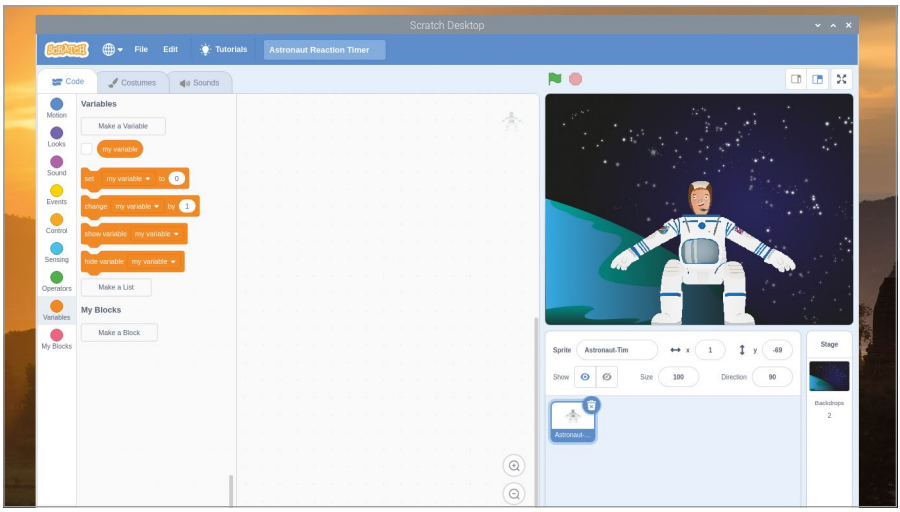
Hvis du har fulgt dette kapitel fra starten, er nu blevet fortrolig med brugergrænsefladen i Scratch 3. Følgende instruktioner er baseret på, at du ved, hvor tingene er. Hvis du glemmer der er noget, du ikke kan finde, er der et billede af brugergrænsefladen i starten af dette kapitel.

Højreklik på katten på listen, og klik på "slet". Hold musemarkøren hen over ikonet "Vælg en baggrund"  , og klik på ikonet "Hent baggrund"  i den menu, der vises. Find frem til filen **Space-background.png** i mappen Downloads, klik på den for at markere den, og klik på OK. Den hvide scenebaggrund ændres til billedet af rummet, og den nye baggrund vises også i stedet for kodeområdet (**Figur 4-11**). Her kan du tegne på baggrunden, men indtil videre skal du bare klikke på fanen Kode øverst i Scratch 3-vinduet.



Figur 4-11: Rumbaggrunden vises på scenen

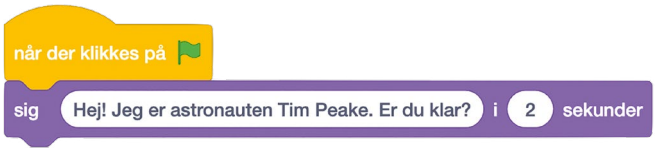
Hold musemarkøren hen over ikonet "Vælg en sprite", og klik på ikonet "Hent sprite" øverst i den menu, der vises. Find frem til filen **Astronaut-Tim.png** i mappen Downloads, klik på den for at markere den, og klik på OK. Den nye sprite vises automatisk på scenen, men er måske ikke helt i midten. Klik på den, og træk den med musen, til den er i midten af scenen, og astronautens fødder rører den nederste scenekant (**Figur 4-12**).



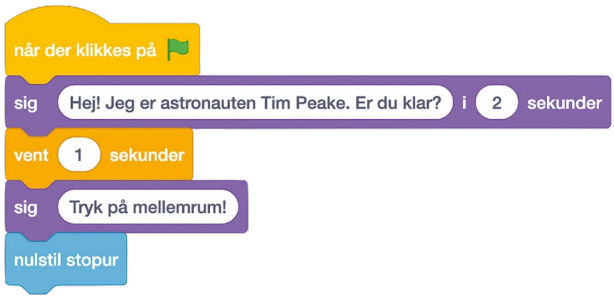
Figur 4-12: Træk astronauten til den nederste kant i midten af scenen

Og nu, når din nye baggrund og sprite er på plads, er det på tide at komme i gang med programmeringen. Start med at oprette en ny variabel kaldet "tid", og sørg for, at indstillingen "For alle sprites" er markeret, før du klikker på OK. Klik på din sprite – enten på scenen eller

i sprite-vinduet – for at vælge den, og tilføj så en [når der klikkes på]-brik fra kategorien Hændelser til kodeområdet. Tilføj derefter en [sig Hej! i 2 sekunder]-brik fra kategorien Udseende, klik på den, skriv "Hej! Jeg er astronauten Tim Peake. Er du klar?"



Tilføj en [vent 1 sekunder] -brik fra kategorien Kontrol, og så en [sig Hej!]-brik. Rediger brikken, så der står "Tryk på mellemrum!", og tilføj så en [nulstil stopur] -brik fra kategorien Registrering. Denne funktion styrer en stopursvariabel i Scratch. Her vil vi bruge den til at måle, hvor hurtigt du kan reagere i spillet.

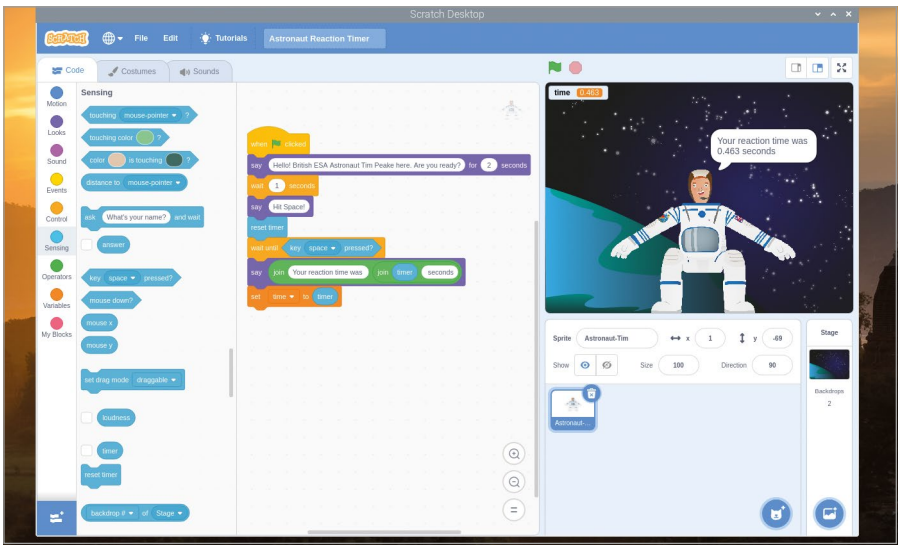


Tilføj en [gentag indtil]-kontrolbrik, og placer en [trykket på tasten mellemrum?] -registreringsbrik i den hvide åbning i kontrolbrikken. Dette får programmet til at vente på, at du trykker på **mellemrumstasten** på tastaturet, men timeren fortsætter med at køre og tæller, hvor lang tid der går, fra du ser meddelelsen "Tryk på mellemrum", til du trykker på **mellemrumstasten**.

Nu skal vi have Tim til at fortælle dig, hvor lang tid du var om at trykke på **mellemrumstasten**. Til det formål skal vi bruge en [tilkobles] -operatorbrik. Denne brik tager to værdier, f.eks. to variabler, og sætter dem sammen i en sammenkædning, der inden for datalogi kaldes *konkatenering*.

Tag en sig Hej!-brik, og placer operatorbrikken tilkobles i ordet "Hej!". Klik på "æble", og overskriv det med teksten "Din reaktionstid var på " (bemærk, at der skal være et mellemrum efter "på"). Læg derefter en anden tilkoblingsbrik i ordet "banan" i det andet felt. Træk en stopur -rapportbrik fra kategorien Registrering, og placer den i, hvad der nu er det midterste felt, og skriv " sekunder." i det sidste felt – bemærk, at der skal være et mellemrum i starten af ordet.

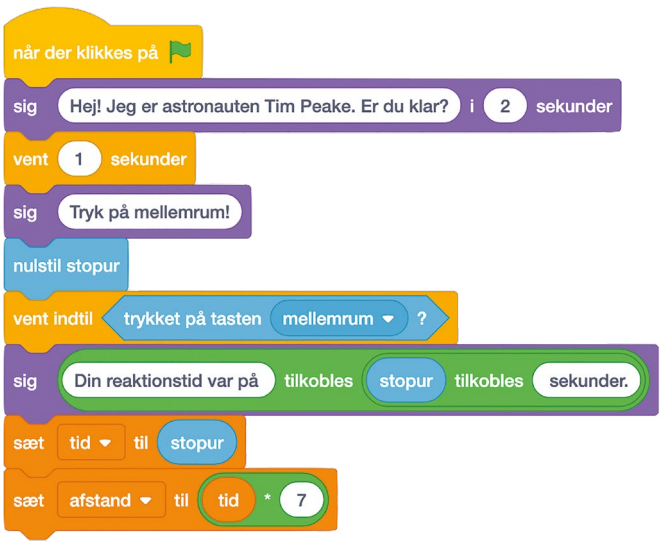
Til sidst skal du lægge en sæt min variabel til 0 -brik fra kategorien Variabler i slutningen af briksekvensen. Klik på pilen ud for "min variabel" i brikken, vælg "tid" på listen, og udskift derefter "0" med en stopur -rapportbrik fra kategorien Registrering. Du kan nu teste dit nye spil ved at klikke på det grønne flag over scenen. Gør dig klar, og så snart du ser meddelelsen "Tryk på mellemrum!", skal du skynde dig at trykke på **mellemrumstasten** (**Figur 4-13**). Se om du kan slå vores highscore!



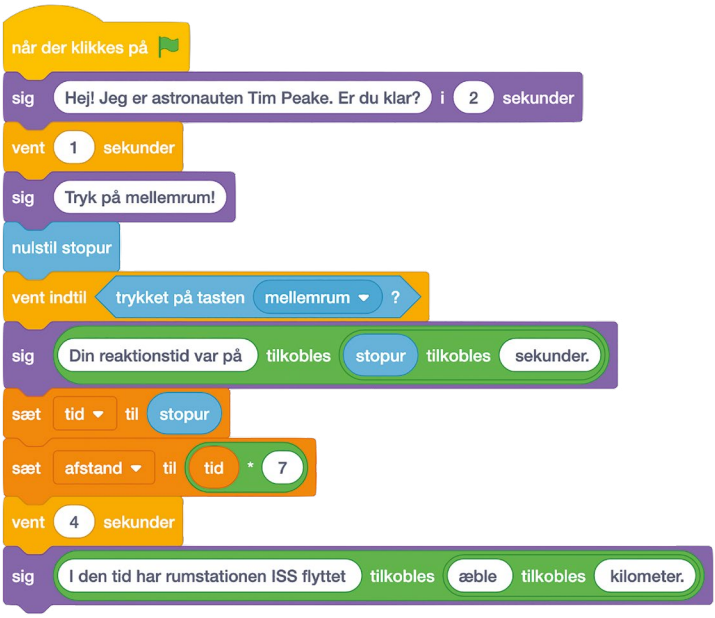
▲ **Figur 4-13:** Tid til at prøve spillet!

Du kan udvide projektet yderligere ved at lade det beregne, hvor langt den internationale rumstation har flyttet sig i den tid, det tog dig at trykke på **mellemrumstasten**, baseret på stationens oplyste hastighed på syv kilometer i sekundet. Opret først en ny variabel, og kald den "afstand". Læg mærke til, hvordan brikkerne i kategorien Variabler automatisk ændrer udseende for at vise den nye variabel, men de eksisterende tid-variabler i dit program forbliver uændret.

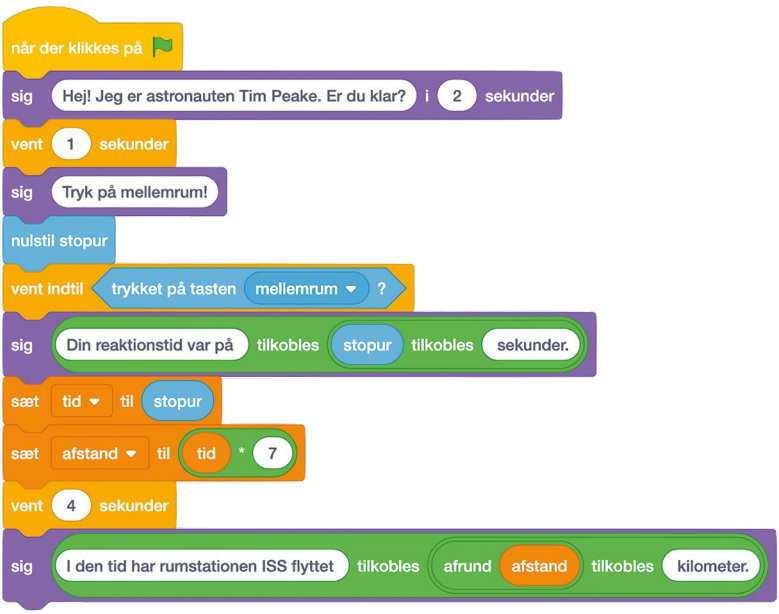
Tilføj en sæt afstand til 0 -brik til din sekvens, og læg operatorbrikken (*) (der ganger to værdier) i "0". Læg en tid-rapportbrik i det første tomme felt, og indtast tallet "7" i det andet felt. Når du er færdig, ser den færdige brik sådan ud:
sæt afstand til tid * 7. Denne funktion tager den tid, det tog dig at trykke på **mellemrumstasten**, og ganger den med syv for at nå frem til den afstand i kilometer, som den internationale rumstation ISS har tilbagelagt.



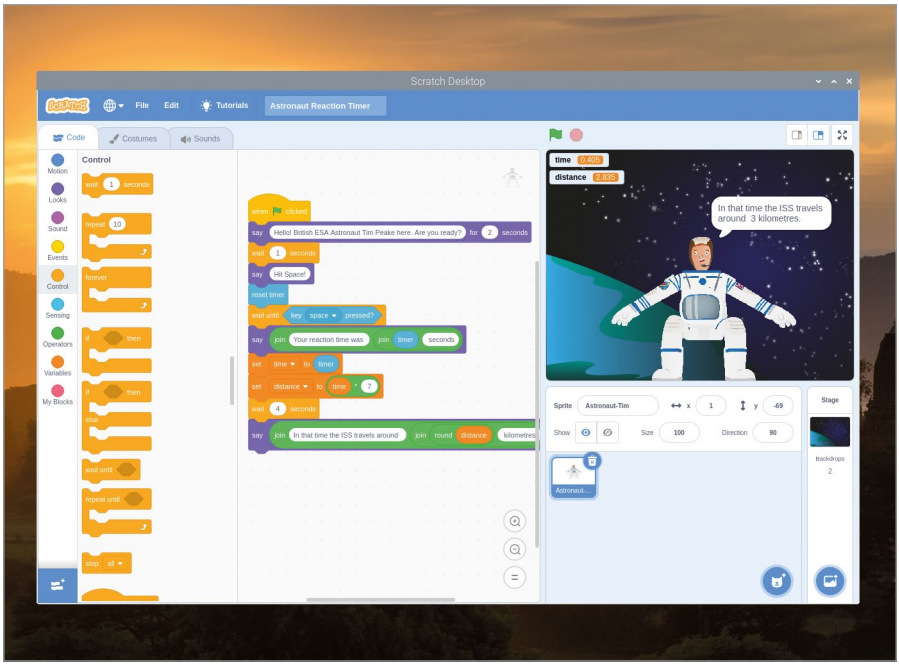
Tilføj en vent 1 sekunder -brik, og rediger dens værdi til "4". Til sidst skal du lægge en ny sig Hej! -brik i slutningen af briksekvensen og tilføje to tilkobles -brikker til den lige som før. Skriv "I den tid har rumstationen ISS flyttet sig " i det første felt (husk mellemrummet til sidst), og skriv " kilometer." i "banan"-feltet (med mellemrum foran).



Læg en afrund -operatorbrik i det midterste felt, og læg en afstand -rapportbrik i det nye blanke felt, der er kommet. Brikken afrund afrunder tal op eller ned til det nærmeste heltal, så i stedet for en supernøjagtig afstandsangivelse med mange decimaler får du et letlæseligt heltal.



Kør programmet ved at klikke på det grønne flag, og du vil se, hvor langt rumstationen ISS har fløjet, mens du var travlt beskæftiget med at trykke på **mellemrumstasten**. Husk at gemme dit program, når du er færdig, så du nemt kan indlæse det igen i fremtiden uden at skulle starte forfra.



▲ **Figur 4-14:** Tim fortæller dig, hvor langt rumstationen ISS er fløjet

UDFORDRING: HVEM ER HURTIG?

Hvilke andre jobs udover astronaut kræver, at man har gode reflekser? Prøv at lave dine egne sprites og baggrunde, så de passer til sådant et job.

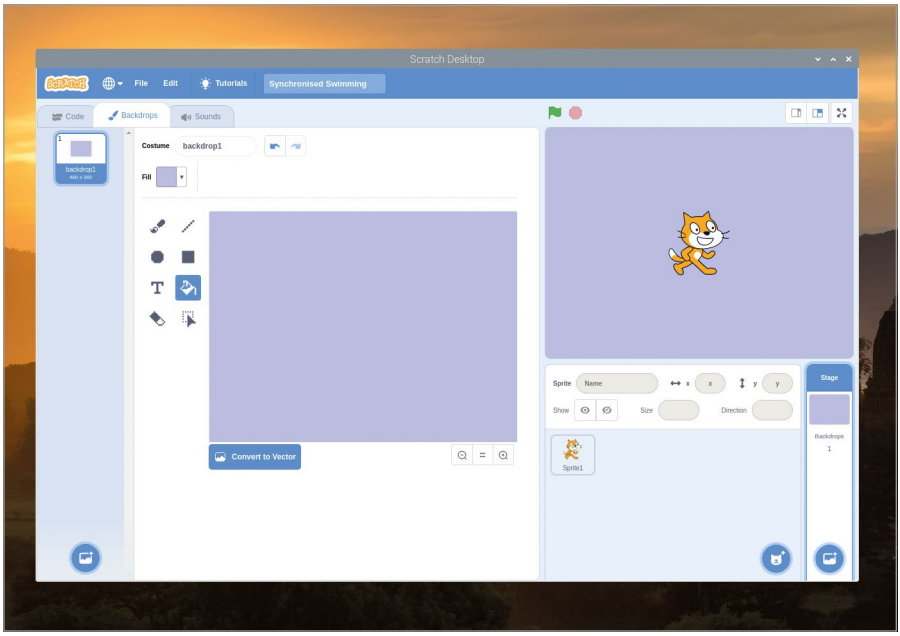
Projekt 2: Synkroniseret svømning

De fleste spil styres ved hjælp af mere end en enkelt knap, og i dette projekt kan du lære, hvordan du bruger tasterne ← og → til at styre dit spil.

ONLINEPROJEKT

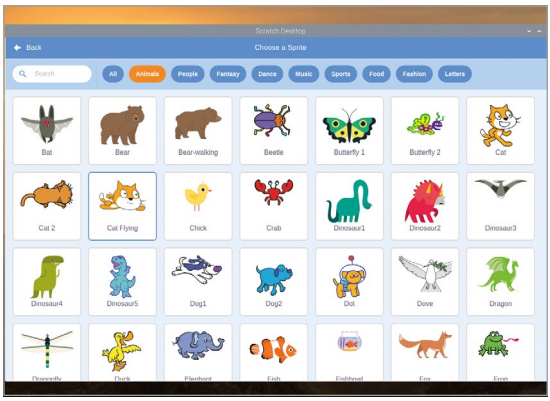
Dette projekt findes også online på **rpf.io/synchro-swimming**

Opret et nyt projekt, og gem det med navnet "Synkroniseret svømning". Klik på scenen, klik på fanen Kostumer øverst til venstre. Klik på knappen Konverter til Bitmap under baggrunden. Vælg en blå farve, der kunne ligne vand, fra paletten Fyld, klik på ikonet Fyld, og klik derefter på den ternede baggrund for at udfylde den med den blå farve (**Figur 4-15**).



Figur 4-15: Fyld baggrunden med en blå farve

Højreklik på katten på listen, og klik på "slet". Klik på ikonet "Vælg en sprite" for at se en liste over de sprites, der følger med programmet. Klik på kategorien Dyr, vælg "Cat Flying" (flyvende kat) (**Figur 4-16**), og klik på OK. Denne sprite fungerer fint i projekter med svømning.



Figur 4-16: Vælg en sprite fra biblioteket

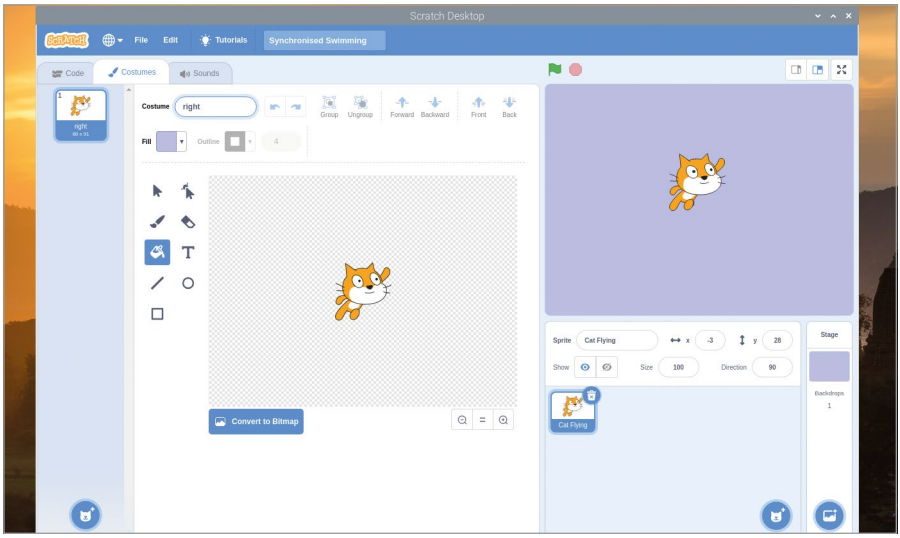
Klik på den nye sprite, og træk derefter to
når du trykker på mellemrum
-brikker over til kodeområdet. Klik på den lille pil ud for ordet "mellemrum" på den første brik, og vælg "venstre pil" på listen. Træk en
drej 15 grader
-brik fra kategorien Bevægelse, og læg den under din
når du trykker på venstre pil
-brik. Gør det samme med den

anden hændelsesbrik, men vælg denne gang "højre pil" på listen, og brug en drej 15 grader -bevægelsesbrik.

Tryk på ← og derefter på → på tastaturet for at teste programmet. Bemærk, at katten drejer rundt i den retning, pilen peger på, når du trykker på tasten. Bemærk også, at du ikke behøvede at klikke på det grønne flag denne gang. Det er, fordi de udløserbrikker, du lige har brugt, er altid aktive, selv når programmet ikke "kører" som sådan.

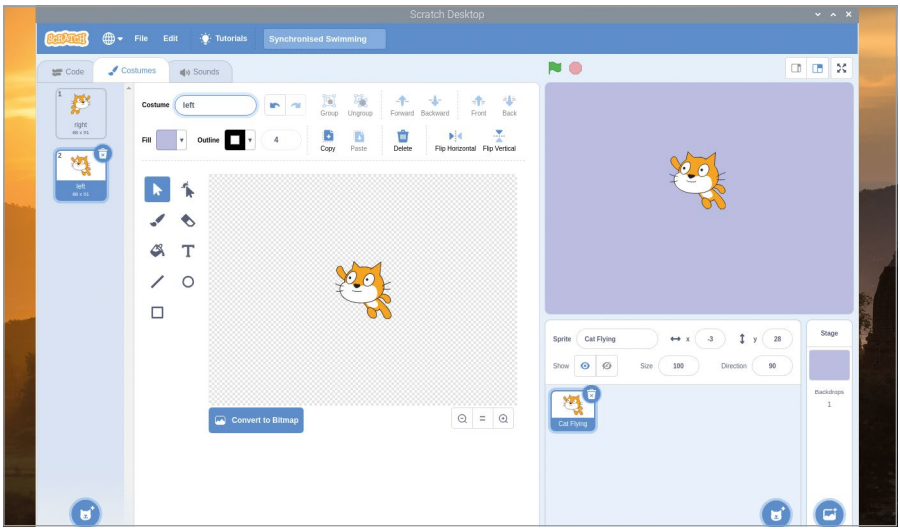
Gentag disse trin to gange mere, men denne gang skal du vælge "pil opad" og "pil nedad" i hændelsesbrikkerne og henholdsvis gå 10 trin og gå -10 trin i bevægelsesbrikkerne. Tryk på piletasterne nu, og læg mærke til, at katten nu både kan dreje og svømme forlæns og baglæns.

For at gøre kattens bevægelser lidt mere realistiske kan du ændre dens udseende, dvs. dens *kostume*, som det hedder i Scratch-terminologien. Klik på katten, og klik derefter på fanen Kostumer over brikpaletten. Klik på "cat flying-a"-kostumet, og klik på papirkurvsikonet med det lille X for at slette kostumet. Klik derefter på "cat flying-b"-kostumet, og omdøb det til "højre" ved at overskrive teksten i navnefeltet (**Figur 4-17**).



▲ **Figur 4-17:** Omdøb kostumet til "højre"

Højreklik på kostumet "højre", som du lige har omdøbt, og vælg "kopier" for at oprette en kopi. Klik på denne kopi, klik på Vælg-værktøjet , klik på Vend venstre-højre-knappen , og omdøb kostumet til "venstre" (**Figur 4-18**). Du har nu to "kostumer" til din sprite, som er spejlbilleder af hinanden: et "højre"-kostume, hvor katten vender mod højre, og et "venstre"-kostume, hvor katten vender mod venstre.



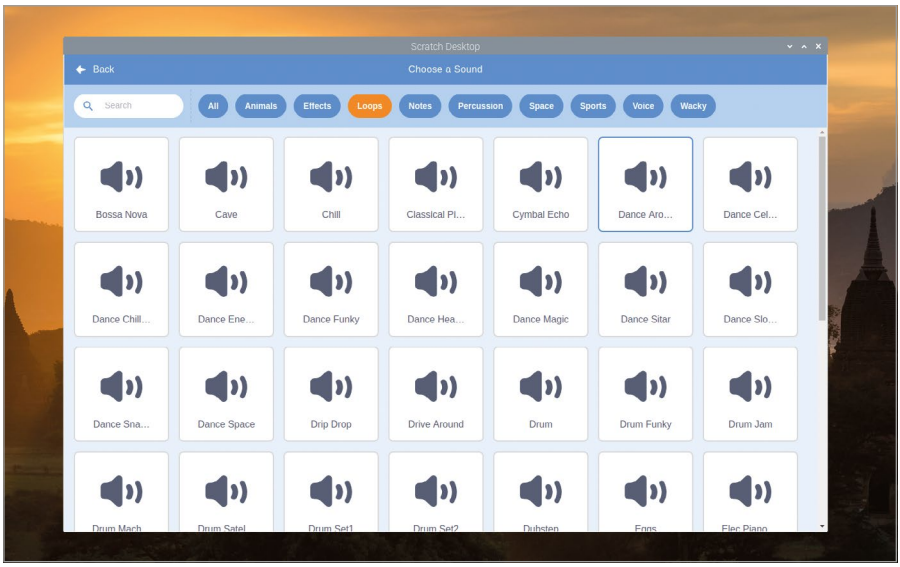
▲ **Figur 4-18:** Kopiér kostumet, spejlvend det, og omdøb det til "venstre"

Klik på fanen Kode over kostumeområdet, træk to **skift kostume til venstre** -brikker fra kategorien Udseende over til kodeområdet, placer dem under dine venstre og højre pilebrikker, og rediger brikken under den højre pilebrik, så den viser **skift kostume til højre**. Prøv piletasterne igen. Katten vender sig nu korrekt i den retning, den svømmer.

Men for at kunne deltage i synkroniseret svømning skal vi bruge flere svømmere, og vi har brug for en måde at nulstille kattens position på. Tilføj en [når der klikkes på] -hændelsesbrik, og læg en [gå til x: 0 y: 0] -bevægelsesbrik under den (rediger værdierne, hvis det er nødvendigt, så de svarer til eksemplet vist her) samt en [peg i retning 90] -bevægelsesbrik. Når du nu klikker på det grønne flag, flytter katten sig til midten af scenen og vender mod højre.

For at oprette flere svømmere skal du tilføje en **gentag 6** -brik (og ændre standardværdien "10" til "6") og placere en **opret en klon af mig selv** -kontrolbrik inde i den. For at forhindre, at alle svømmerne svømmer i samme retning, skal du tilføje en **drej 60 grader** -brik over **opret en klon**-brikken men stadig inde i **gentag 6** -brikken. Klik på det grønne flag, og prøv piletasterne for at se, hvordan svømmerne reagerer nu.

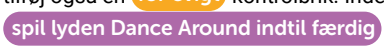
For at skabe en ægte OL-stemning skal du tilføje noget musik. Klik på fanen Lyde over brikpaletten, og klik på ikonet "Vælg en lyd"  . Klik på kategorien Gentagelser, og scrol ned på listen (**Figur 4-19**), indtil du finder noget musik, du kan lide – vi har valgt "Dance Around". Klik på knappen OK for at vælge musikken, og klik på fanen Kode for at åbne kodeområdet igen.



Figur 4-19: Vælg gentaget musik fra lydbiblioteket

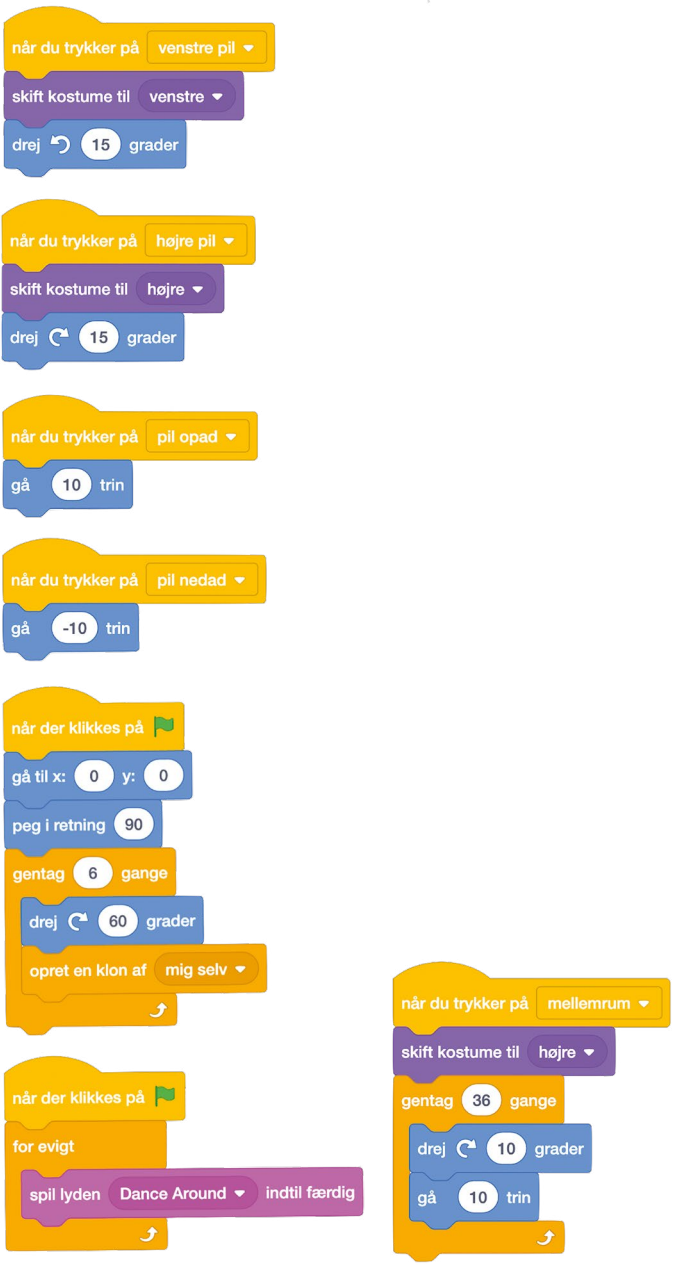
Tilføj en ny 

-hændelsesbrik til kodeområdet, og tilføj også en 

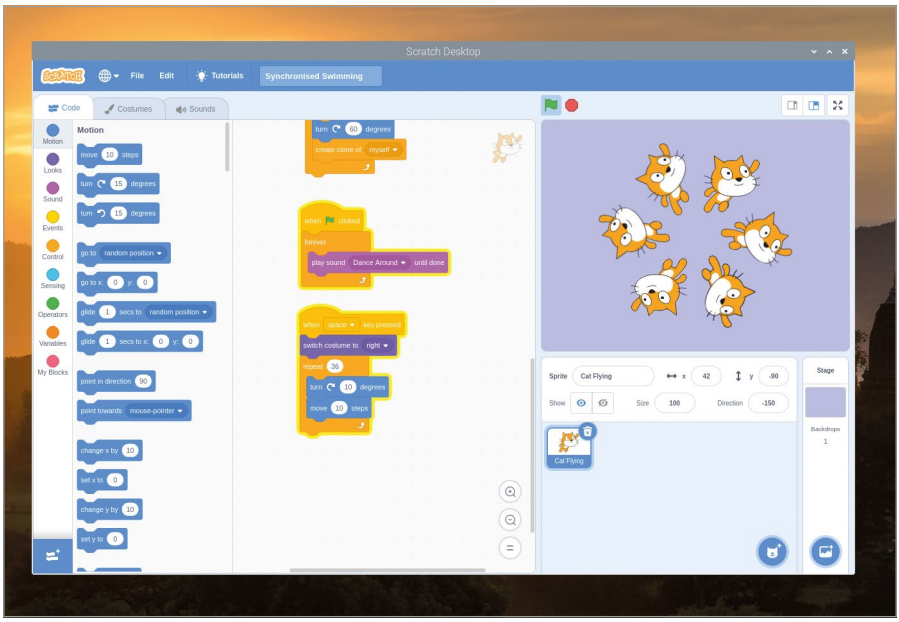
-kontrolbrik. Inde i denne kontrolbrik skal du tilføje en 

-brik – husk at kigge efter navnet på det musikstykke, du valgte. Klik derefter på det grønne flag for at teste dit nye program. For at stoppe musikken skal du bare klikke på den røde ottekant, så stopper programmet afspilningen.

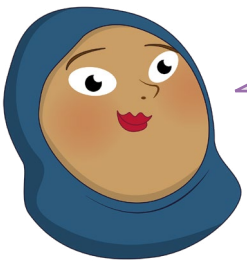
Endelig kan du lave en simulation af en fuld danserutine ved at tilføje en ny hændelsesudløser til programmet. Tilføj en **når du trykker på mellemrum** -hændelsesbrik og en **skift kostume til højre** -brik. Under den sidste brik skal du tilføje en **gentag 36**-brik (husk at ændre standardværdien) og inde i denne en **drej 10 grader**-brik og en **gå 10 trin**-brik.



Klik på det grønne flag for at starte programmet, og tryk derefter på **mellemrumstasten** for at teste den nye danserutine (**Figur 4-20** på næste side). Husk at gemme dit program, når du er færdig.



▲ **Figur 4-20:** Den færdige danserutine i synkroniseret svømning



UDFORDRING: BRUGERDEFINERET RUTINE

Kan du lave din egen rutine i synkroniseret svømning ved hjælp af løkker? Hvad skal du ændre, hvis du vil have flere svømmere eller færre svømmere? Kan du tilføje flere svømmerutiner, der kan udløses ved hjælp af forskellige taster på tastaturet?

Projekt 3: Bueskydningsspil

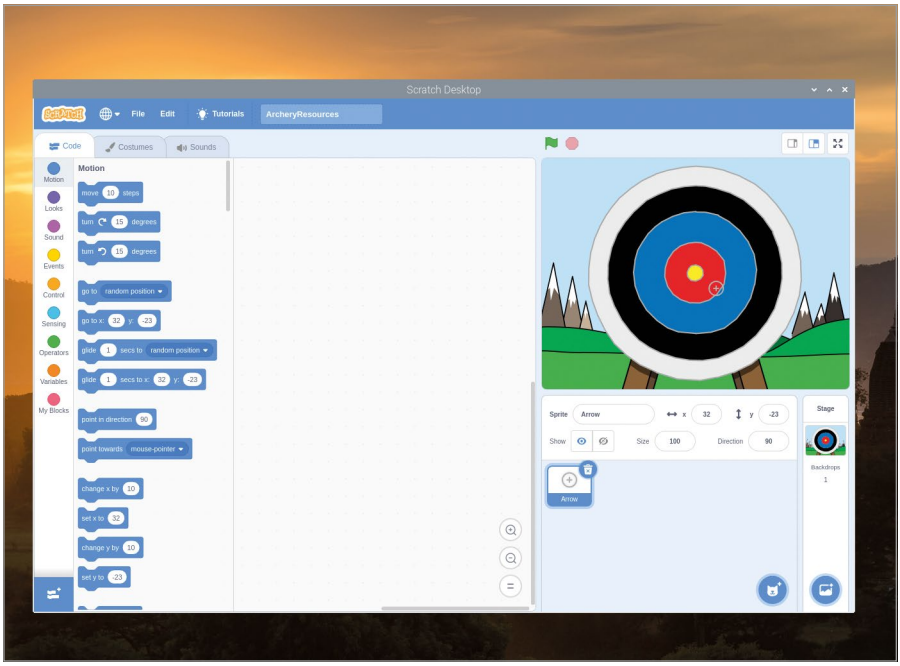
Du er ved at blive en sand Scratch-ekspert! Lad os kaste os over en ny udfordring: Et bueskydningsspil, hvor spilleren skal ramme et mål med en pil skudt ud af en bue, der svinger tilfældigt.

ONLINEPROJEKT

Dette projekt findes også online på **rpf.io/archery**

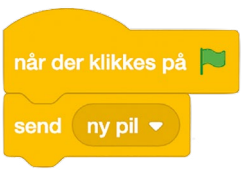
Start med at åbne Chromium-browseren og skrive **rpf.io/p/en/archery-go**. Tryk derefter på **ENTER**. Ressourcerne til spillet downloades som en zip-fil, så du skal først pakke zip-filen ud – højreklik på den, og vælg Extract Here (pak ud hertil). Skift tilbage til Scratch 3, klik på menuen Fil,

og vælg "Hent fra din computer". Klik på **ArcheryResources.sb3**, og vælg Åbn. Du bliver spurgt, om du vil erstatte indholdet af dit aktuelle projekt. Hvis du ikke har gemt dine ændringer, skal du klikke på Annuller og gemme dem, ellers skal du klikke på OK.



▲ **Figur 4-21:** Projektressourcer indlæst til bueskydningsspillet

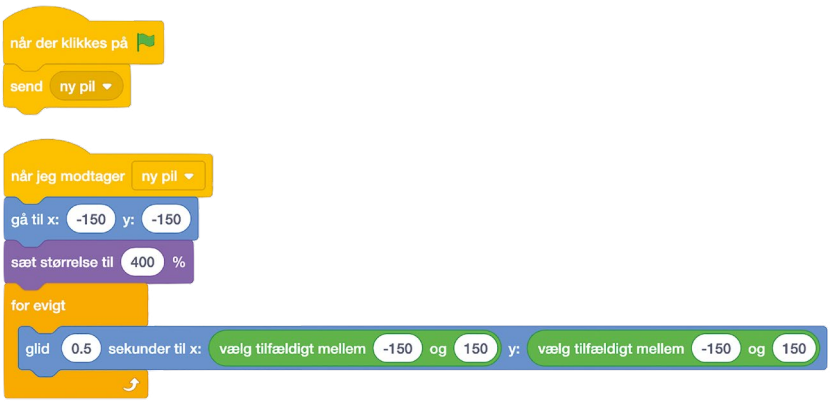
Det projekt, du lige har indlæst, indeholder en baggrund og en sprite (**Figur 4-21**), men ingen af den kode, der skal bruges til spillet – for det bliver din opgave. Start med at tilføje en når der klikkes på -brik og en send besked1 -brik. Klik på pileikonet for enden af brikken, vælg "Ny besked", skriv "ny pil", og klik på OK. Brikken ser nu sådan ud: send ny pil .



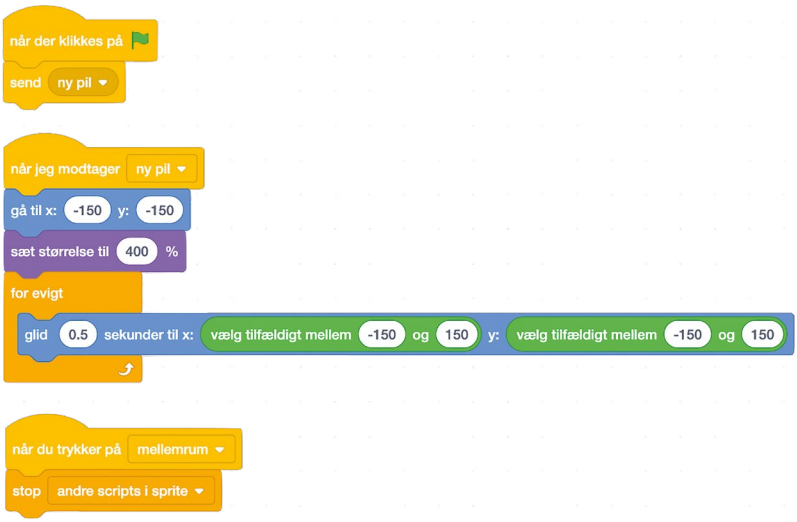
Denne funktion sender en besked fra en del af programmet til en anden del af programmet. For at få funktionen til at fungere skal du tilføje en når jeg modtager besked1 -brik. Rediger den, så den viser når jeg modtager ny pil. Du skal blot klikke på pilikonet og vælge "ny pil" på listen – du behøver ikke at indtaste teksten igen.

Under når jeg modtager ny pil-brikken skal du lægge en gå til x: -150 y: -150 -brik og en sæt størrelse til 400 %-brik. Husk, at disse ikke er standardværdierne for disse brikker, så du skal redigere dem, når du har placeret dem i kodeområdet. Klik på det grønne flag for at se, hvordan programmet fungerer indtil videre: Pilen, som spilleren bruger til at sigte på målet med, flyttes nederst til venstre på scenen og bliver fire gange så stor.

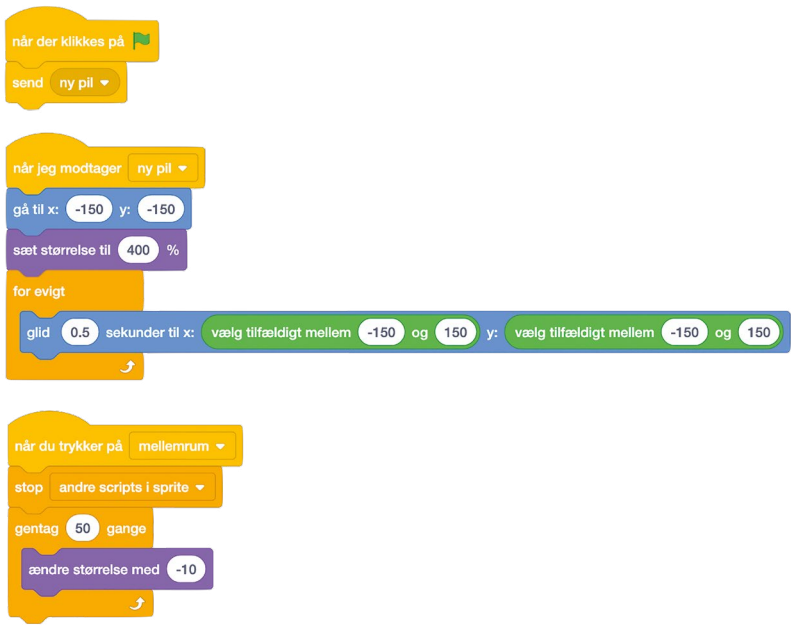
Som en ekstra udfordring skal du tilføje bevægelse, der simulerer, at buen bevæger sig lidt, når den trækkes, og der tages sigte. Tilføj en for evigt -brik og en glid 1 sekunder til x: -150 y: -150 -brik. Rediger det første hvide felt, så der står "1" i stedet for "0.5", og placer derefter en vælg tilfældigt mellem -150 og 150 -operatorbrik i de to resterende hvide felter. Dette gør, at sigten flytter sig lidt, så pilen peger i en tilfældig retning – hvilket gør det meget sværere at ramme målskiven!



Klik på det grønne flag igen, og læg mærke til, hvordan brikken fungerer: Pilen flytter sig rundt og peger på forskellige dele af måskiven. I øjeblikket har du dog ingen måde at affyre pilen på. Træk en når du trykker på mellemrum -brik og en stop alle -brik til kodeområdet. Klik på pileikonet for enden af brikken, og vælg stop andre scripts i sprite.

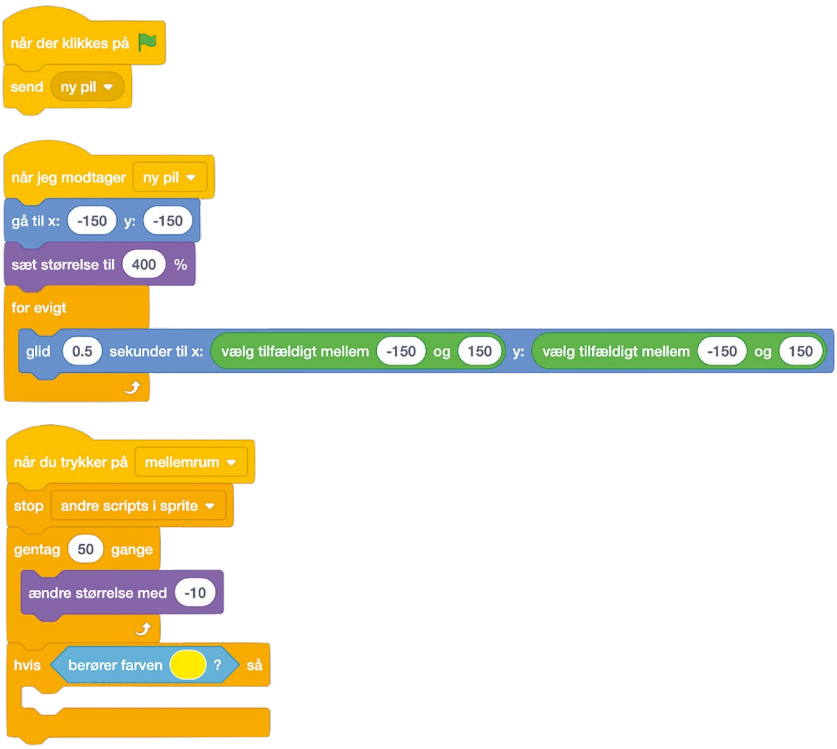


Hvis du havde stoppet programmet for at tilføje de nye brikker, skal du klikke på det grønne flag for at starte det igen og derefter trykke på **mellemrumstasten**. Pilen stopper med at bevæge sig. Så langt, så godt, men vi skal også få det til at se ud, som om pilen flyver af sted mod målet. Tilføj en gentag 50-brik og en ændre størrelse med -10-brik, og klik på det grønne flag for at teste programmet igen. Denne gang ser pilen ud til at flyve afsted fra din placering og frem mod målet.

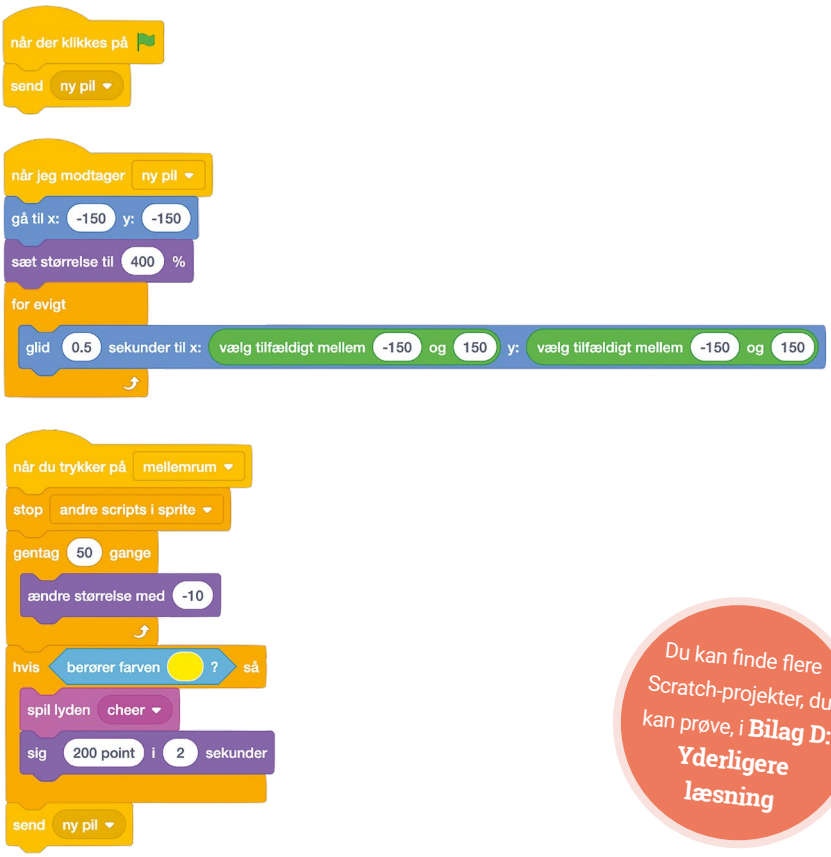


For at gøre spillet sjovt skal du tilføje en måde at holde styr på point. Tilføj en hvis så -brik fra samme kategori, læg den under gentag 50-brikken (ikke inde i den), og placer en

berører farven? -brik i dens diamantformede åbning. For at vælge den rigtige farve skal du klikke på det farvede felt på registreringsbrikken, vælge pipetteværktøjet og klikke i det gule felt midt i målskiven på scenen.



Tilføj en spil lyden cheer -brik og en sig 200 point i 2 sekunder -brik inde i hvis så -brikken, så spilleren ved, at målet er ramt. Tilføj så en send ny pil -brik helt i bunden af sekvensen, under hvis så -brokken (men ikke inde i den), så spilleren får en ny pil, når den gamle er blevet affyret. Klik på det grønne flag for at starte spillet, og prøv at ramme det gule felt midt i målskiven. Hver gang du rammer, lyder der et begejstret bifald fra tilskuerne, og du får 200 point.



Du kan finde flere Scratch-projekter, du kan prøve, i **Bilag D: Yderligere læsning**

Spillet fungerer, men lige nu er det lidt svært. Brug de ting, du har lært i dette kapitel, til at udvide spillet, så spilleren også får point for at ramme andre dele af målskiven end det gule felt i midten: 100 point for at ramme rødt, 50 point for at ramme blåt, osv.

UDFORDRING: KAN DU GØR DET ENDNU BEDRE?

Hvad ville du gøre for at gøre spillet nemmere? Hvad ville du gøre for at gøre spillet sværere? Kan du bruge variabler til at give flere point, når der skydes med flere pile? Kan du tilføje en nedtælling for at presse spilleren lidt?

Kapitel 5

Programmering med Python

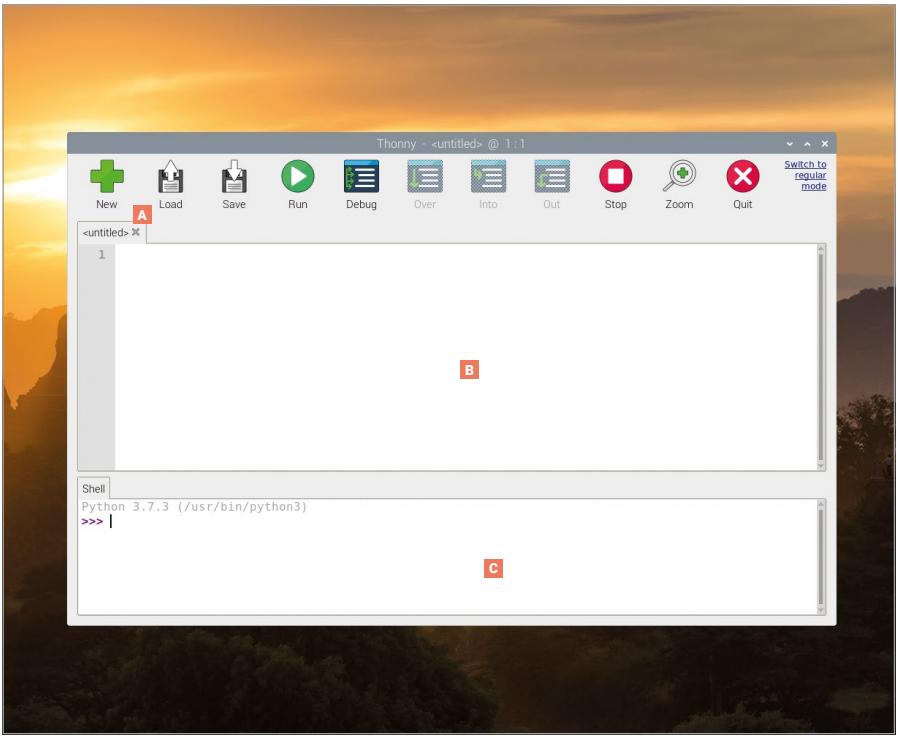
Nu hvor du har prøvet Scratch, viser vi dig, hvordan du udfører tekstbaseret kodning med Python

Guido van Rossums Python er opkaldt efter komedieserien Monty Python og er vokset fra et hobbyprojekt, der først blev udgivet til offentligheden i 1991, til et højt elsket programmeringssprog, der driver en bred vifte af projekter. I modsætning til det visuelle miljø i Scratch er Python tekstbaseret: Du skriver instruktioner ved hjælp af et forenklet sprog og specifikt format, som computeren derefter udfører.

Python er et godt næste skridt for dem, der allerede har brugt Scratch, og giver øget fleksibilitet og et mere "traditionelt" programmeringsmiljø. Det betyder ikke, at det er svært at lære: Med lidt øvelse kan alle skrive Python-programmer til alt fra enkle beregninger til overraskende komplicerede spil.

Dette kapitel bygger oven på vilkår og begreber introduceret i **Kapitel 4, Programmering med Scratch 3**. Hvis du endnu ikke har gennemgået øvelserne i dette kapitel, vil du finde dette kapitel lettere at følge, hvis du går tilbage og gør det først.

Introduktion til Thonny Python IDE



A **Værktøjslinje** – Thonnys brugergrænseflade "Simple Mode" bruger en linje med venlige ikoner som menu, så du kan oprette, gemme, indlæse og køre dine Python-programmer samt teste det på forskellige måder.

B **Scriptområde** – Scriptområdet er, hvor dine Python-programmer er skrevet, og er opdelt i et hovedområde for dit program og en lille sidemargen til visning af linjenumre.

C **Python Shell** – Shell-området i Python giver dig mulighed for at skrive individuelle instruktioner, som derefter køres, så snart du trykker på **ENTER**-tasten og giver også oplysninger om kørende programmer.

VERSIONER AF THONNY

Thonny har to versioner af brugergrænseflader: "Regular Mode" og en "Simple Mode", som er bedre for begyndere. Dette kapitel bruger Simple Mode, der indlæses som standard, når du åbner Thonny fra afsnittet Programmering i hindbærmenuen.

Dit første Python-program: Hello, World!

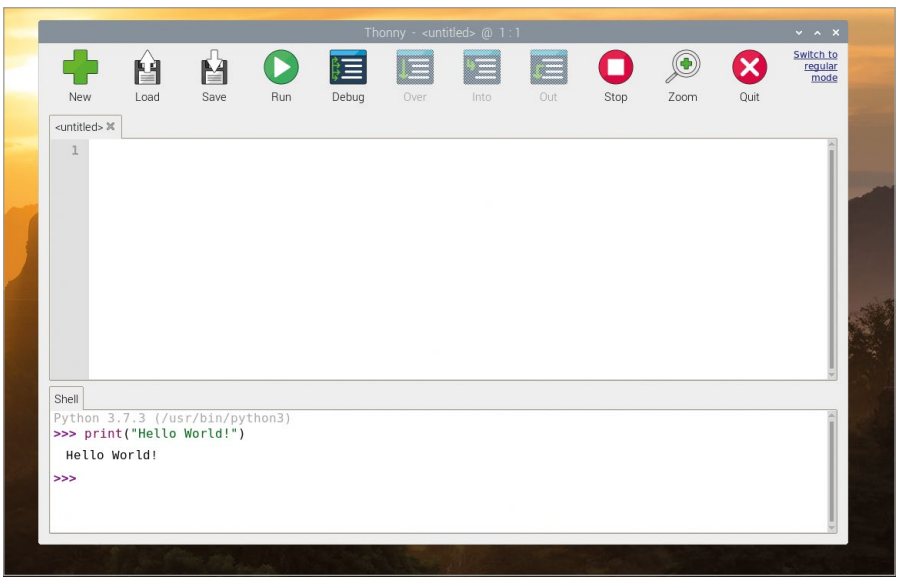
Ligesom de andre forudinstallerede programmer på Raspberry Pi er Thonny tilgængelig fra menuen: Klik på hindbærikonet, flyt markøren til afsnittet Programming (Programmering), og klik på Thonny Python IDE. Efter et par sekunder indlæses Thonny-brugergrænsefladen (Simple Mode som standard).

Thonny er en pakke kendt som et *integreret udviklingsmiljø (IDE)*, et kompliceret lydende navn med en simpel forklaring: Det samler eller *integrerer* alle de forskellige værktøjer, du har brug for til at skrive eller *udvikle*, software i en enkelt brugergrænseflade, eller *miljø*. Der er mange IDE'er til rådighed, hvoraf nogle understøtter mange forskellige programmeringssprog, mens andre, som Thonny, fokuserer på at understøtte et enkelt sprog.

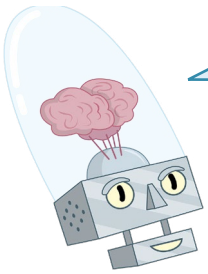
I modsætning til Scratch, som giver dig visuelle byggesten som grundlag for dit program, er Python et mere traditionelt programmeringssprog, hvor alt er nedskrevet. Start dit første program ved at klikke i Python shell-området nederst i Thonny-vinduet, og skriv derefter følgende instruktion, før du trykker på **ENTER**-tasten:

```
print("Hello, World!")
```

Når du trykker på **ENTER**, vil du se, at dit program begynder at køre med det samme: Python vil svare i samme shell-område med beskeden "Hello, World!" (**Figur 5-1**), som du bad om. Det skyldes, at denne "shell" er en direkte linje til Python-*fortolkeren*, hvis opgave det er at se på dine instruktioner og *fortolke*, hvad de betyder. Dette kaldes *interaktiv tilstand*, og du kan tænke på det som en ansigt til ansigt-samtale med nogen: Så snart du er færdig med det, du siger, vil den anden person svare og derefter vente på, hvad du siger bagefter.



▲ **Figur 5-1:** Python skriver beskeden "Hello, World!" i shell-området



SYNTAKS FEJL

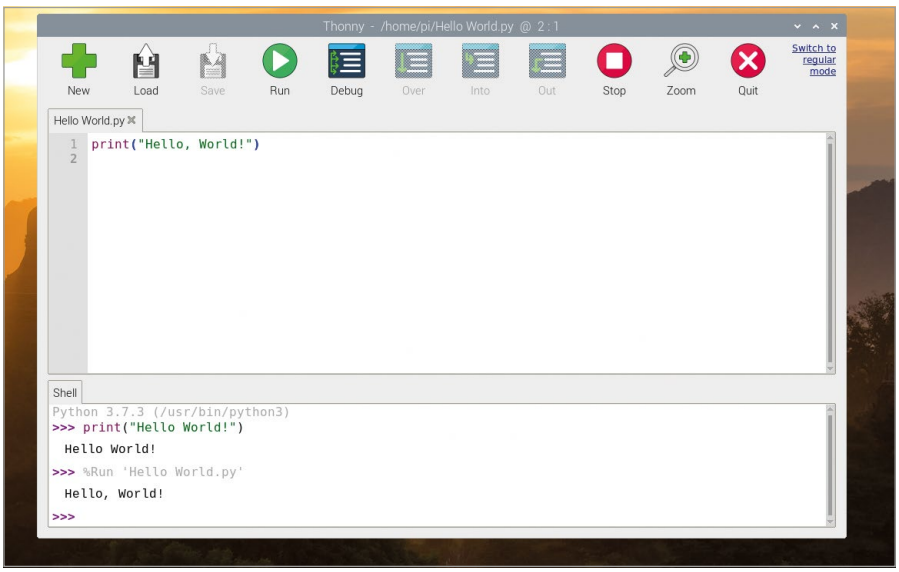
Hvis dit program ikke kører, men i stedet skriver beskeden "syntax error" i shell-området, er der en fejl et eller andet sted i det, du har skrevet. Python har brug for, at instruktionerne skrives på en meget specifik måde: Hvis der mangler en parentes eller et anførselstegn, hvis du staver "print" forkert eller giver det et stort P, eller tilføjer ekstra symboler et eller andet sted i instruktionen, så kører programmet ikke. Prøv at skrive instruktionen igen, og sørg for at den matcher versionen i denne bog, inden du trykker på **ENTER**-tasten!

Du behøver dog ikke at bruge Python i interaktiv tilstand. Klik på scriptområdet i midten af Thonny-vinduet, og skriv derefter dit program igen:

```
print("Hello, World!")
```

Når du trykker på **ENTER**-tasten denne gang, sker der intet – bortset fra at du får en ny, tom linje i scriptområdet. For at få denne version af dit program til at fungere, skal du klikke på ikonet Run (Kør) på Thonny-værktøjslinjen. Når du gør det, bliver du bedt om at gemme dit program først. Angiv et beskrivende navn som f.eks. "Hello World", og klik på knappen Save (Gem). Når dit program er gemt, vises to meddelelser i Python shell-området (**Figur 5-2**):

```
>>> %Run 'Hello World.py'
    Hello, World!
```



▲ **Figur 5-2:** Kørsel af dit enkle program

Den første af disse linjer er en instruktion fra Thonny, der beder Python-fortolkeren køre det program, du lige har gemt. Det andet er output fra programmet – den besked, du bad Python om at udskrive. Tillykke: Du har nu skrevet og kørt dit første Python-program i både interaktiv og scripttilstand!

UDFORDRING: NY MEDDELELSE

Kan du ændre den besked, som Python-programmet udskriver som output? Hvis du ville tilføje flere beskeder, ville du så bruge interaktiv tilstand eller scripttilstand? Hvad sker der, hvis du fjerner parenteserne eller anførselstegnene fra programmet og derefter prøver at køre det igen?

Næste trin: Løkker og kodeindrykning

Ligesom Scratch bruger stakke af puslespilslignende blokke til at kontrollere, hvilke bits i programmet der er forbundet til hvilke andre bits, har Python sin egen måde at kontrollere den rækkefølge, som dets programmer kører i: *indrykning*. Opret et nyt program ved at klikke på ikonet New (Nyt) på Thonny-værktøjslinjen. Du mister ikke dit eksisterende program; i stedet opretter Thonny en ny fane over scriptområdet. Start med at skrive følgende:

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

Den første linje skriver en simpel besked til din shell, ligesom dit Hello World-program. Den anden begynder en *definitiv* løkke, der fungerer på samme måde som i Scratch: en tæller, **`i`**, tildeles løkken og får en række numre – **`range`** instruktionen, som får besked om at starte med tallet 0 og arbejde opad mod, uden dog nogensinde at nå, tallet 10 – for at tælle. Kolontegnet (**`:`**) fortæller Python, at den næste instruktion skal være en del af løkken.

I Scratch er instruktionerne, der skal inkluderes i løkken, bogstaveligt talt inkluderet inde i den C-formede blok. Python bruger en anden tilgang: indrykning af kode. Den næste linje starter med fire mellemrum, som Thonny skulle have tilføjet, da du trykkede på **ENTER** efter linje 2:

```
    print("Løkkenummer", i)
```

Mellemrummene skubber denne linje indad i forhold til de andre linjer. Med denne indrykning skelner Python fortæller mellem instruktioner uden for løkken og instruktioner inde i løkken; den indrykkede kode kaldes *indlejret*.

Du vil bemærke, at da du trykkede på **ENTER** i slutningen af tredje linje, indrykkede Thonny automatisk den næste linje, idet det blev forudsat, at den ville være en del af løkken. For at fjerne dette skal du blot trykke på tasten **BACKSPACE** (Tilbage) en gang, før du skriver den fjerde linje:

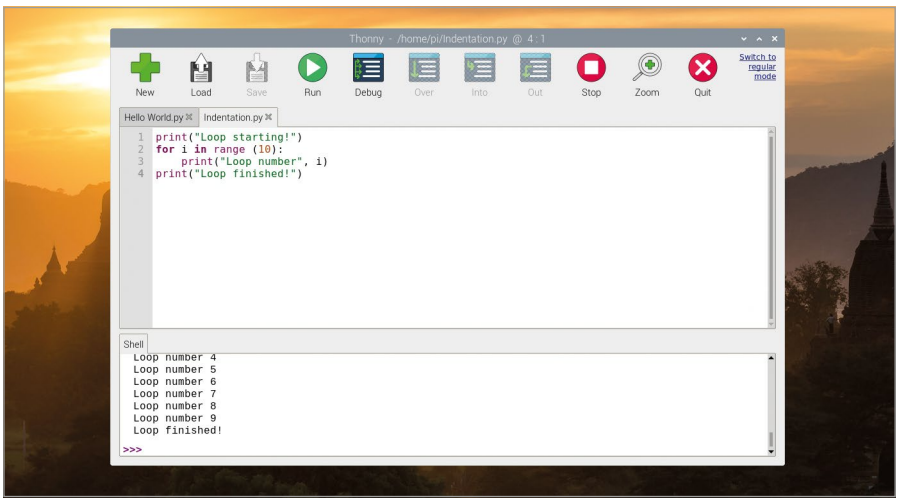
```
print("Løkke færdig!")
```

Dit program på fire linjer er nu afsluttet. Den første linje sidder uden for løkken og vil kun køre én gang; den anden linje opretter løkken; den tredje sidder inde i løkken og løber én gang for hver gang løkken kører; og den fjerde linje sidder endnu en gang uden for løkken.

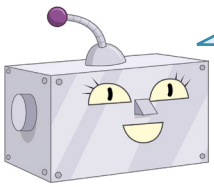
```
print("Løkke starter!")
for i in range(10):
    print("Løkkenummer", i)
print("Løkke færdig!")
```

Klik på ikonet Run (Kør), gem programmet som **Indrykning**, og se output i shell-området **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 færdig!
```



▲ **Figur 5-3:** Udfører en løkke



TÆL NED FRA NUL

Python er et nulindekseret sprog – hvilket betyder, at det begynder at tælle fra 0, ikke fra 1 – hvorfor dit program udskriver tallene 0 til 9 i stedet for 1 til 10. Hvis du ville, kunne du ændre denne adfærd ved at ændre instruktionen `range(10)` til `range(1, 11)` – eller en anden rækkefølge, hvis du vil.

Indrykning er en stærk del af Python og en af de mest almindelige årsager til, at et program ikke fungerer som forventet. Når du leder efter problemer i et program, en proces kaldet *debugging*, skal du altid kontrollere indrykningen – især når du begynder at indlejre løkker i løkker.

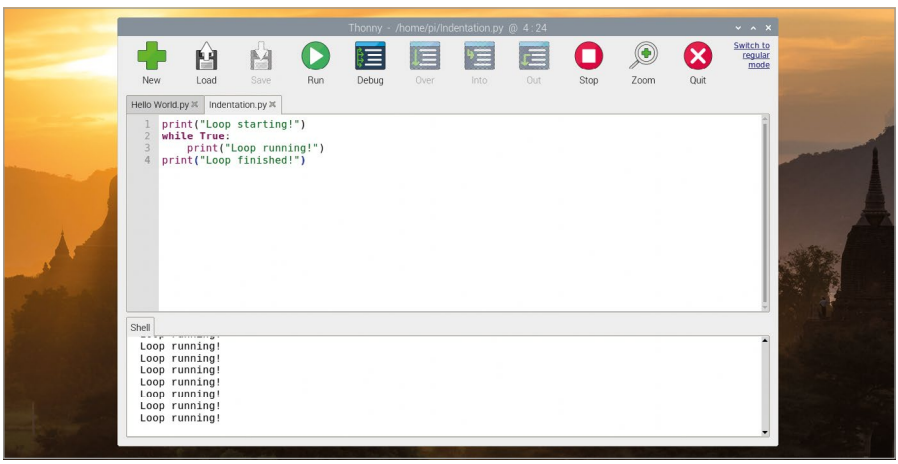
Python understøtter også *uendelige* løkker, der kører uden ende. Hvis du vil ændre dit program fra en definitiv løkke til en uendelig løkke, skal du redigere linje 2, så den ser sådan ud:

```
while True:
```

Hvis du klikker på ikonet Run (Kør), får du en fejl: **`name 'i' is not defined`**. Dette skyldes, at du har slettet linjen, der oprettede og tildelte en værdi til variablen **`i`**. For at løse dette skal du blot redigere linje 3, så den ikke længere bruger variablen:

```
    print("Løkke kører!")
```

Klik på ikonet Run (Kør), og – hvis du er hurtig – får du vist meddelelsen "Løkke starter!" efterfulgt af en uendelig streng af "Løkke starter!"-meddelelser (**Figur 5-4**). Meddelelsen "Løkke færdig!" skrives aldrig, fordi løkken ikke har nogen ende: Hver gang Python er færdig med at skrive meddelelsen "Løkke kører!", går den tilbage til begyndelsen af løkken og skriver den igen.



▲ **Figur 5-4:** En uendelig løkke fortsætter, indtil du stopper programmet

Klik på Stop-ikonet på Thonny-værktøjslinjen for at bede programmet om at stoppe, hvad det laver – kendt som at afbryde programmet. Du vil se, at en meddelelse vises i Pythons shell-område, og programmet stopper – og uden nogensinde at nå linje 4.

UDFORDRING: LØKKE I LØKKEN

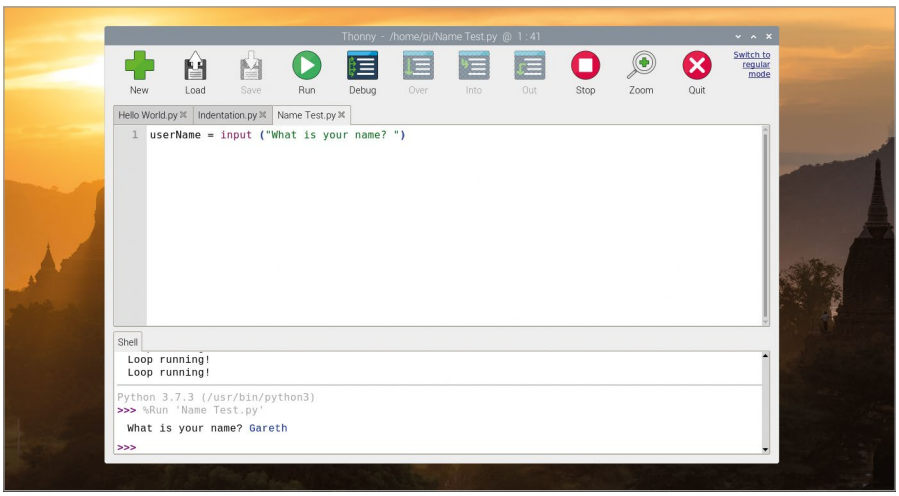
Kan du ændre løkken tilbage til en definitiv løkke igen? Kan du tilføje en anden definitiv løkke programmet? Hvordan vil du tilføje en løkke i en løkke, og hvordan forventer du, at den fungerer?

Betingelser og variabler

Variabler, ligesom i alle programmeringssprog, findes til mere end bare at kontrollere løkker. Start et nyt program ved at klikke på ikonet New (Nyt) i Thonny-menuen, og skriv derefter følgende i scriptområdet:

```
userName = input("Hvad er dit navn? ")
```

Klik på ikonet Run (Kør), gem dit program som **Navnetest**, og se hvad der sker i shell-området: Du bliver bedt om dit navn. Indtast dit navn i shell-området, efterfulgt af **ENTER**. Fordi det er den eneste instruktion i dit program, sker der ikke andet (**Figur 5-5**). Hvis du rent faktisk vil gøre noget med de data, du har placeret i variablen, skal du bruge flere linjer i dit program.



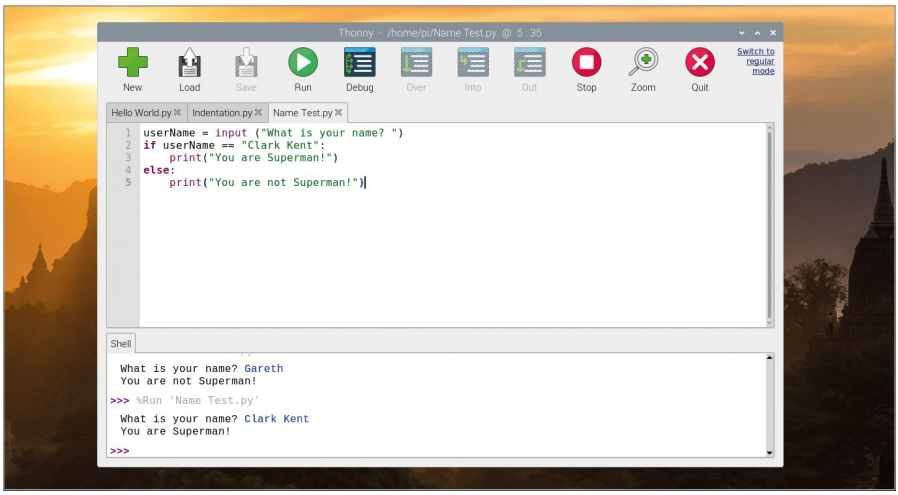
Figur 5-5: Med `input`-funktionen kan du bede en bruger om noget tekstinput

For at få dit program til at gøre noget nyttigt med navnet skal du tilføje en *betinget erklæring* ved at skrive følgende:

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

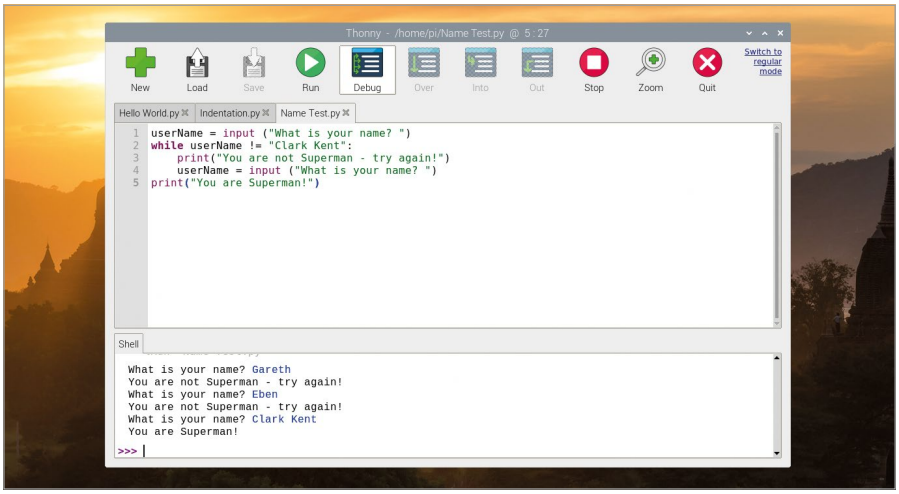
Husk, at når Thonny ser, at din kode skal indrykkes, vil den gøre det automatisk – men det ved ikke, hvornår din kode skal stoppe med at blive indrykket, så du bliver nødt til selv at slette mellemrummene.

Klik på ikonet Run (Kør), og indtast dit navn i shell-området. Medmindre dit navn tilfældigvis er Clark Kent, får du vist meddelelsen "Du er ikke Superman!". Klik på Run (Kør) igen, og skriv denne gang navnet "Clark Kent" – sørg for at skrive det nøjagtigt som i programmet med store bogstaver C og K. Denne gang genkender programmet, at du faktisk er Superman (**Figur 5-6**).

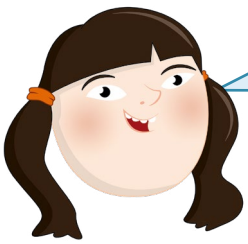


▲ **Figur 5-6:** Bør du ikke være ude for at redde verden?

Symbolerne **==** beder Python om at foretage en direkte sammenligning og se om variablen **userName** matcher teksten – kendt som en *streng* – i dit program. Hvis du arbejder med tal, er der andre sammenligninger, du kan lave: **>** for at se om et tal er større end et andet tal, **<** for at se om det er mindre end, **=>** for at se om det er lig med eller større end, **=<** for at se om det er lig med eller mindre end. Der er også **!=**, hvilket betyder ikke lig med – det er det stik modsatte af **==**. Disse symboler er teknisk kendt som *sammenligningsoperatorer*.



Figur 5-7: Du vil blive ved med at blive bedt om dit navn, indtil du siger, det er "Clark Kent"



BRUG AF = AND ==

Nøglen til at bruge variabler er at lære forskellen mellem = and ==. Husk: = betyder "gør denne variabel lig med denne værdi", mens == betyder "tjek for at se om variablen er lig med denne værdi". At blande dem sammen er en sikker måde at ende med et program, der ikke fungerer!

Sammenligningsoperatører kan også bruges i løkker. Slet linje 2 til 5, og skriv derefter følgende på deres sted:

```
while userName != "Clark Kent":
    print("Du er ikke Superman - prøv igen!")
    userName = input("Hvad er dit navn? ")
print("Du er Superman!")
```

Klik på ikonet Run (Kør) igen. Denne gang, i stedet for at afslutte, fortsætter programmet med at bede om dit navn, indtil det bekræfter, at du er Superman (**Figur 5-7**) – som en meget simpel adgangskode. For at komme ud af løkken skal du enten skrive "Clark Kent" eller klikke på Stop-ikonet på Thonny-værktøjslinjen. Tillykke: Du ved nu, hvordan du bruger betingelses- og sammenligningsoperatorer!

UDFORDRING: TILFØJ FLERE SPØRGSMÅL

Kan du ændre programmet til at stille mere end t spørgsmål og gemme svarene i flere variabler? Kan du lave et program, der bruger betingelses- og sammenligningsoperatorer til at udskrive, om et tal, der er indtastet af brugeren, er højere eller lavere end 5, ligesom det program, du oprettede i **Kapitel 4, Programmering med Scratch**?

Projekt 1: Skildpadde snefnug

Nu hvor du forstår hvordan Python fungerer, det er tid til at lege med grafik og oprette et snefnug ved hjælp af et værktøj kendt som en *turtle* (skilpadde).

ONLINEPROJEKT

Dette projekt er også tilgængeligt online på **rpf.io/turtle-snowflakes**

Oprindeligt blev fysiske robotter formet som deres dyrs navnebrødre, og skildpadder er designet til at bevæge sig i en lige linje, dreje og til at løfte og sænke en pen – i den digitale version, hvilket simpelthen betyder at starte eller stoppe med at tegne en linje, når den bevæger sig. I modsætning til visse andre sprog, f.eks. Logo og dets mange varianter, har Python ikke et indbygget skildpaddeværktøj – men det leveres med et *bibliotek* med tilføjelseskode for at give det skildpaddekraft. Biblioteker er samlinger af kode, der tilføjer nye instruktioner for at udvide Pythons muligheder og bringes ind i dine egne programmer ved hjælp af en import-kommando.

Opret et nyt program ved at klikke på ikonet New (Nyt) , og skriv følgende:

```
import turtle
```

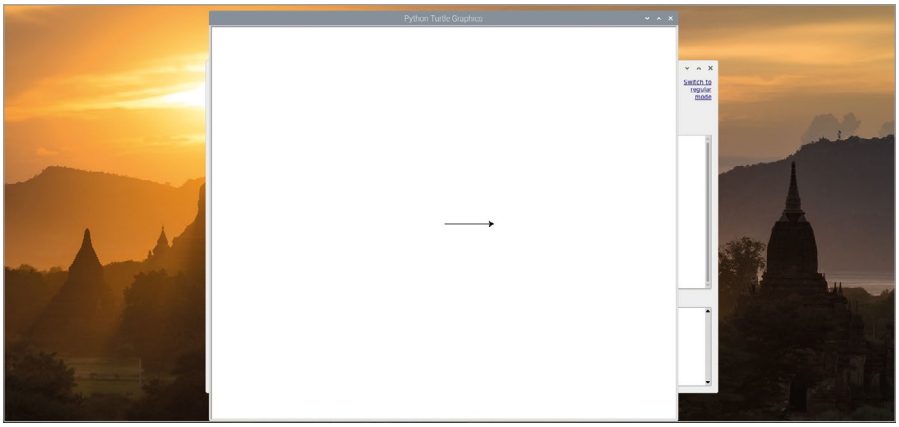
Når du bruger instruktioner, der er inkluderet i et bibliotek, skal du bruge biblioteksnavnet efterfulgt af et punktum og derefter instruktionsnavnet. Det kan være irriterende at skulle skrive hver gang, så du kan tildele et kortere variabelnavn i stedet – det kan være et enkelt bogstav, men vi synes, at det kunne være rart, hvis det også var et kæledyrsnavn til skildpadden. Skriv følgende:

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

For at teste dit program skal du give din skildpadde noget at gøre. Skriv:

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

Klik på ikonet Run (Kør) , og gem dit program som **Skildpadde snefnug**. Når programmet er gemt, vises et nyt vindue kaldet "Turtle Graphics" (Skilpaddegrafik), og du vil se resultatet af dit program: Din skildpadde, Pat, bevæger sig 100 enheder fremad og tegner en lige linje (**Figur 5-8**).



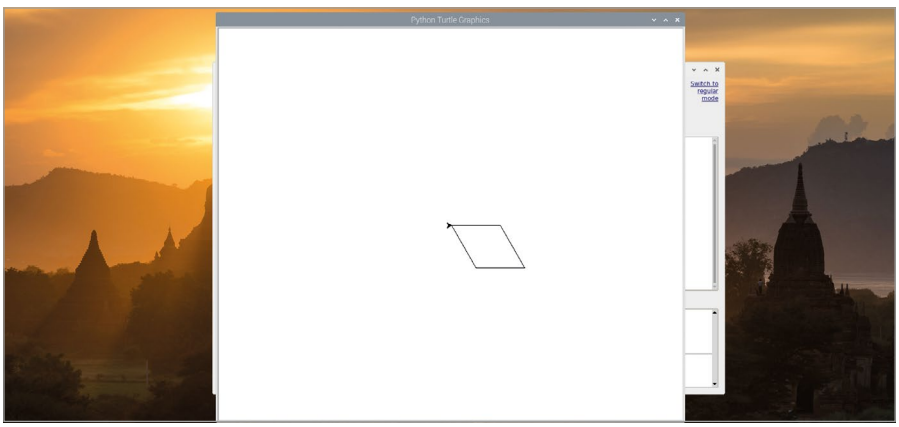
Figur 5-8: Skildpadden bevæger sig fremad og tegner derved en lige linje

Skift tilbage til det primære Thonny-vindue – hvis det er skjult bag vinduet Turtle Graphics, skal du enten klikke på minimeringsknappen i Turtle Graphics-vinduet eller klikke på Thonny-posten på proceslinjen øverst på skærmen – og klikke på Stop-knappen for at lukke vinduet Turtle Graphics.

At indtaste hver enkelt bevægelsesinstruktion i hånden ville være kedeligt, så slet linje 3, og opret en løkke for at gøre det hårde arbejde med at skabe former:

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

Kør dit program, og Pat tegner et enkelt parallelogram (**Figur 5-9**).



Figur 5-9: Ved at kombinere drejninger og bevægelser kan du tegne former

For at gøre det til en snefnuglignende form skal du klikke på Stop-ikonet i det primære Thonny-vindue og oprette en løkke om din løkke ved at tilføje følgende linje som linje 3:

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

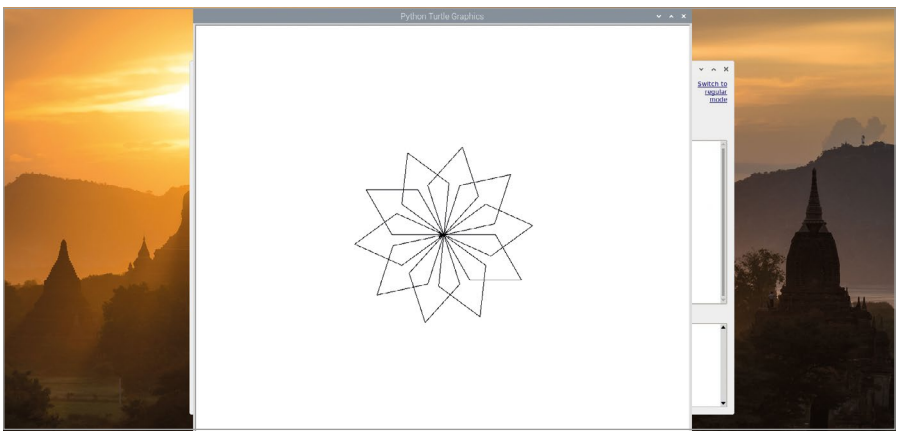
... og følgende nederst i dit program:

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

Dit program vil ikke køre som det er nu, fordi den eksisterende løkke ikke er indrykket korrekt. For at rette op på det skal du klikke på starten af hver linje i den eksisterende løkke – linje 4 til 8 – og trykke på **SPACE**-tasten (mellemrum) fire gange for at rette indrykningen. Dit program skal nu se sådan ud:

```
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)
```

Klik på ikonet Run (Kør), og se skildpadden: Den tegner et parallelogram, som før, men når det er gjort, drejer den 36 grader og tegner et nyt, derefter et andet og så videre, indtil der er ti overlappende parallelogrammer på skærmen – der ligner et snefnug (**Figur 5-10**).

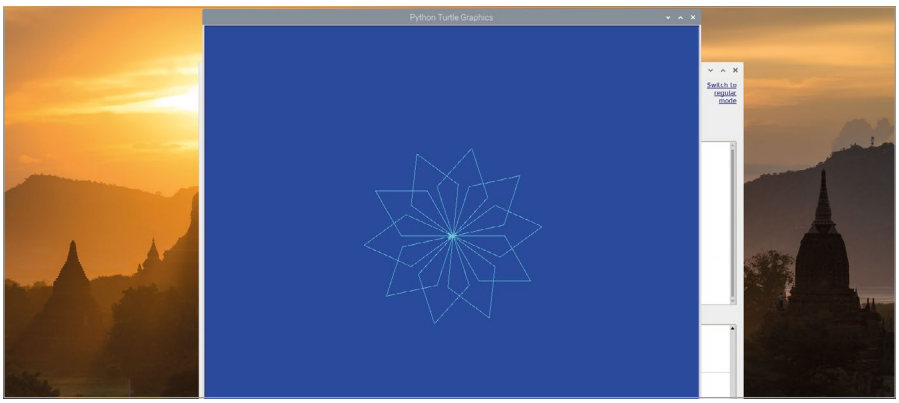


▲ **Figur 5-10:** Gentag formen for at gøre den mere kompleks

Mens en robotskildpadde tegner med en enkelt farve på et stort stykke papir, kan Pythons simulerede skildpadde bruge en række forskellige farver. Tilføj en ny linje 3 og 4, idet de eksisterende linjer skubbes ned:

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

Kør dit program igen, og du vil se effekten af din nye kode: Baggrundsfarven i vinduet Turtle Graphics er ændret til blå, og snefnugget har nu farven cyan (**Figur 5-11**).



▲ **Figur 5-11:** *Ændring af baggrunds- og snefnugfarver*

Du kan også få farverne valgt tilfældigt fra en liste via biblioteket `random`. Gå tilbage til toppen af dit program, og indsæt følgende som linje 2:

```
import random
```

Skift baggrundsfarve i hvad der nu er linje 4 fra "blue" til "grey", og opret derefter en ny variabel kaldet "farver" ved at indsætte en ny linje 5:

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

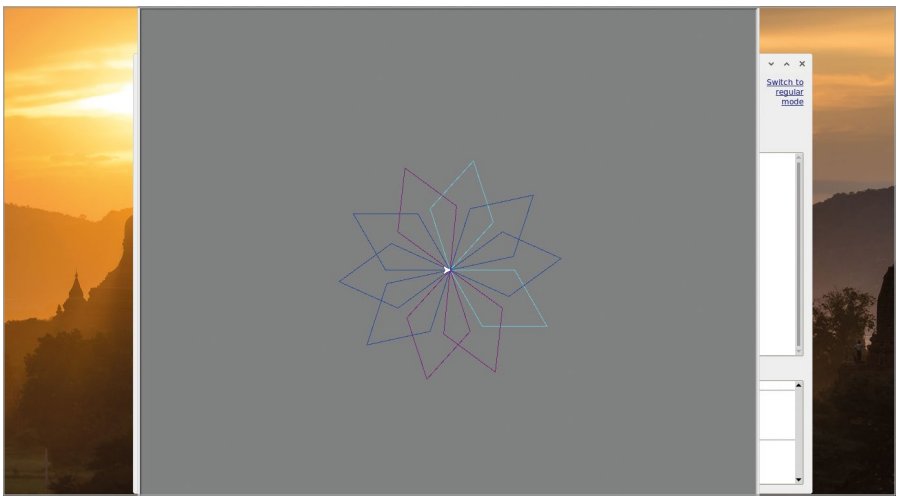
Amerikansk stavemåde

Mange programmeringssprog bruger amerikansk engelsk stavemåde, og Python er ingen undtagelse: Kommandoen til at ændre farven på skildpaddens pen staves *color*, og hvis du staver det på den engelske måde som *colour*, virker det simpelthen ikke. Variabler kan dog staves, som du har lyst - det er derfor, du kan kalde din nye variabel *farver* og få Python til at forstå det.

Denne type variabel kaldes en liste og er omsluttet af firkantede parenteser. I dette tilfælde er listen fyldt med mulige farver til snefnugsegmenterne – men du skal stadig bede Python om at vælge en, hver gang løkken gentages. I slutningen af programmet skal du indtaste følgende – sørg for, at det er indrykket med fire mellemrum, så det udgør en del af den ydre løkke, ligesom linjen over det:

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

Klik på ikonet Run (Kør), hvorefter snefnug-ninja-stjernen tegnes igen. Denne gang vil Python dog vælge en tilfældig farve fra din liste, når den tegner hvert kronblad – hvilket giver snefnugget en flot, flerfarvet finish (**Figur 5-12**).



▲ **Figur 5-12:** Brug af tilfældige farver til "kronbladene"

For at få snefnugget til at se mindre ud som en ninjastjerne og mere som et rigtigt snefnug, skal du tilføje en ny linje 6 direkte under din liste **`farver`** og skrive følgende:

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

Instruktionerne **penup** og **pendown** flytter en fysisk pen ud og videre til papiret, hvis du bruger en skildpaddrobot, men i den virtuelle verden skal du blot bede din skildpadde om at stoppe og begynde at tegne linjer. Denne gang skaber du i stedet for at bruge en løkke en *funktion* – et kodesegment, som du kan kalde når som helst, som at oprette din helt egen Python-instruktion.

Start med at slette koden til tegning af dine parallelogram-baserede snefnug: Det er alt mellem og inklusive instruktionen **`pat.color("cyan")`** i linje 10 til **`pat.right(36)`** i linje 17. Lad instruktionen **`pat.color(random.choice(farver))`** være, men tilføj et hashtag (**#**) i starten af linjen. Dette kaldes at *kommentere ud* en instruktion og betyder, at Python vil ignorere den. Du kan bruge kommentarer til at tilføje forklaringer til din kode, hvilket gør den meget lettere at forstå, når du kommer tilbage til den et par måneder senere eller sender den videre til en anden!

Opret din funktion, som kaldes "branch", ved at skrive følgende instruktion på linje 10 under **`pat.pendown():`**

```
def branch():
```

Dette *definerer* din funktion, **`branch`**. Når du trykker på **ENTER**-tasten, tilføjer Thonny automatisk indrykning til funktionens instruktioner. Skriv følgende, og sørg for at være opmærksom på indrykning – for på et tidspunkt skal du indlejre kode tre indrykningsniveauer dybt!

```
    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)
```

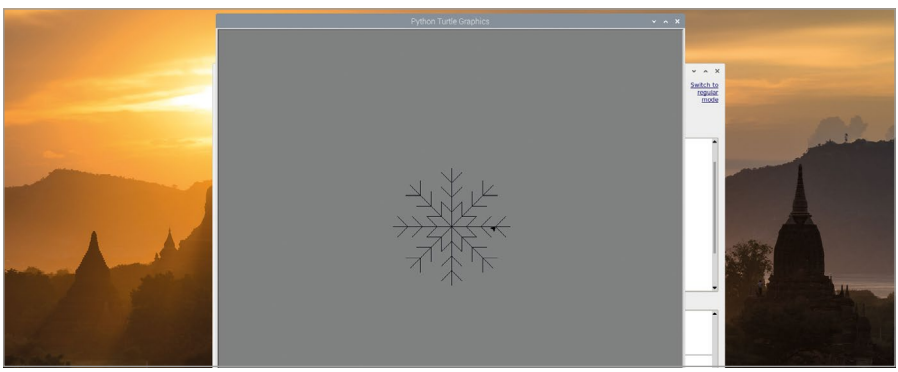
Til sidst skal du oprette en ny løkke i slutningen af dit program – men over den kommenterede farvelinje – for at køre eller *kalde* din nye funktion:

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

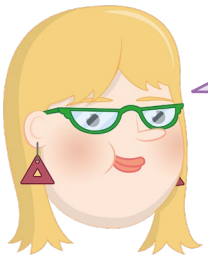
Dit færdige program skal se sådan ud:

```
import turtle
import random
pat = turtle.Turtle()
turtle.Screen().bgcolor("grey")
farver = ["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(farver))
```

Klik på Run (Kør) og se grafikvinduet, når Pat tegner ved at følge dine instruktioner. Tillykke: Dit snefnug ligner nu meget mere et snefnug (**Figur 5-13**)!



▲ **Figur 5-13:** Ekstra grene får det til at ligne et snefnug



UDFORDRING: HVAD ER DET NÆSTE?

Kan du bruge dine udkommenterede instruktioner til at få snefnuggets grene tegnet i forskellige farver? Kan du oprette en "snefnug"-funktion og bruge den til at tegne mange snefnug på skærmen? Kan du få dit program til at ændre størrelsen og farven på snefnuggene tilfældigt?

Projekt 2: Uhyggelig Find forskellen

Python kan også håndtere billeder og lyde såvel som skildpaddebaseret grafik, der kan bruges med stor effekt som et prank på dine venner – et spil om at finde forskellen med en skræmmende hemmelighed inderst inde, perfekt til Halloween!

ONLINEPROJEKT

Dette projekt er også tilgængeligt online på **rpf.io/scary-spot**

Dette projekt har brug for to billeder – dit find forskellen-billede plus et "uhyggeligt" overraskelsesbillede – og en lydfil. Klik på hindbærikonet for at indlæse Raspberry Pi OS-menuen, vælg internetkategorien og klik på Chromium Web Browser. Når det er indlæst, skal du skrive **rpf.io/spot-pic** på adresselinjen og derefter trykke på **ENTER**-tasten. Højreklik på billedet, og klik på "Save image as..." (Gem billede som ...), vælg mappen Home (Hjem) på listen til venstre, og klik derefter på Save (Gem). Klik tilbage på Chromiums adresselinje, og skriv derefter **rpf.io/scary-pic** og tryk på **ENTER**-tasten. Som før skal du højreklikke på billedet, klikke på "Save image as..." (Gem billede som ...), vælge mappen Home (Hjem) på listen til venstre, og derefter klikke på Save (Gem).

For den lydfil, du har brug for, skal du klikke tilbage på adresselinjen og skrive **rpf.io/scream** og trykke på **ENTER**-tasten. Denne fil, lyden af et skrig for at give vedkommende, som afspiller programmet, en virkelig overraskelse, afspilles automatisk – men den skal gemmes, før du kan bruge den. Højreklik på den lille lydafspiller, klik på "Save as" (Gem som ...), vælg mappen Home (Hjem), og klik på Save (Gem). Du kan nu lukke Chromium-vinduet.

Klik på ikonet New (Nyt) på værktøjslinjen Thonny for at starte et nyt projekt. Du skal som før bruge et bibliotek til at udvide Pythons muligheder: Pygame-biblioteket, der som navnet antyder blev oprettet med spil i tankerne. Skriv følgende:

```
import pygame
```

Du skal bruge nogle dele fra andre biblioteker og også fra en underafdeling af Pygame-biblioteket. Importer disse ved at skrive følgende:

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

Instruktionen **`from`** fungerer anderledes end instruktionen **`import`**, idet den gør det muligt kun at importere dele af et bibliotek i stedet for hele biblioteket. Dernæst skal du konfigurere Pygame; dette kaldes *initialisering*. Pygame har brug for at kende bredden og højden på afspillerens skærm eller tv, kendt som dens *opløsning*. Skriv følgende:

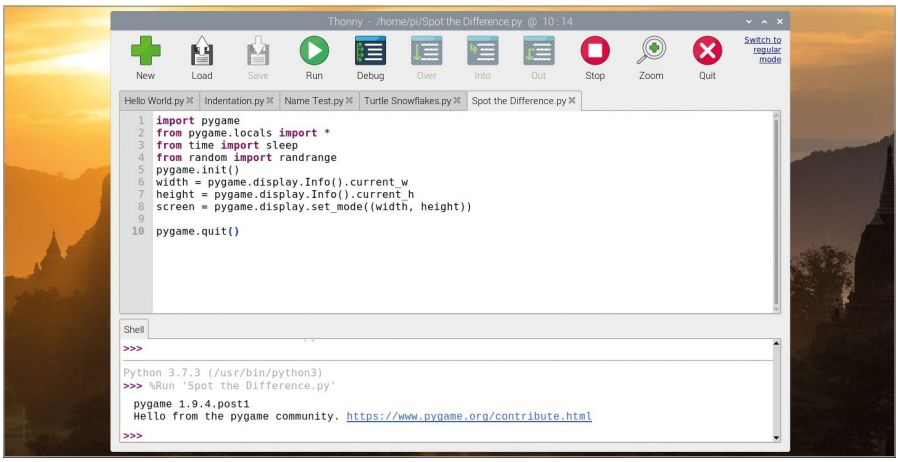
```
pygame.init()
width = pygame.display.Info().current_w
height = pygame.display.Info().current_h
```

Det sidste trin i opsætning af Pygame er at oprette dets vindue, som Pygame kalder en skærm. Skriv følgende:

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

```
pygame.quit()
```

Bemærk den tomme linje i midten; her skal dit program skrives. Men start med at klikke på ikonet Run (Kør), gem dit program som **Find forskellen** og se: Pygame opretter et vindue og udfylder det med en sort baggrund, som derefter næsten øjeblikkeligt forsvinder, når det når til instruktionen om at afslutte. Bortset fra en kort besked i shell-området (**Figur 5-14**) har programmet hidtil ikke udført meget.



▲ **Figur 5-14:** Dit program er funktionelt, men gør ikke ret meget endnu

For at vise dit Find forskellen-billede skal du skrive følgende linje på pladsen over **`pygame.quit()`**:

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

For at sikre, at billedet fylder hele skærmen, skal du skalere det, så det passer til din skærm eller tv'ets opløsning. Skriv følgende:

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

Nu er billedet i hukommelsen, og du skal bede Pygame om at vise det på skærmen – en proces kaldet *blitting* eller en *bitblokoverførsel*. Skriv følgende:

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

Den første af disse linjer kopierer billedet på skærmen startende i øverste venstre hjørne; det andet beder Pygame om at tegne skærmen igen. Uden denne anden linje vil billedet være på det rigtige sted i hukommelsen, men du vil aldrig se det!

Klik på ikonet Run (Kør), og billedet vises kortvarigt på skærmen (**Figur 5-15**).

▲ **Figur 5-15:** Dit Find forskellen-billede

For at se billedet i længere tid skal du tilføje følgende linje lige oven over **`pygame.quit()`**:

```
sleep(3)
```

Klik på Run (Kør) igen, og billedet forbliver nu længere på skærmen. Tilføj dit overraskelsesbillede ved at skrive følgende lige under linjen **`pygame.display.update()`**:

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

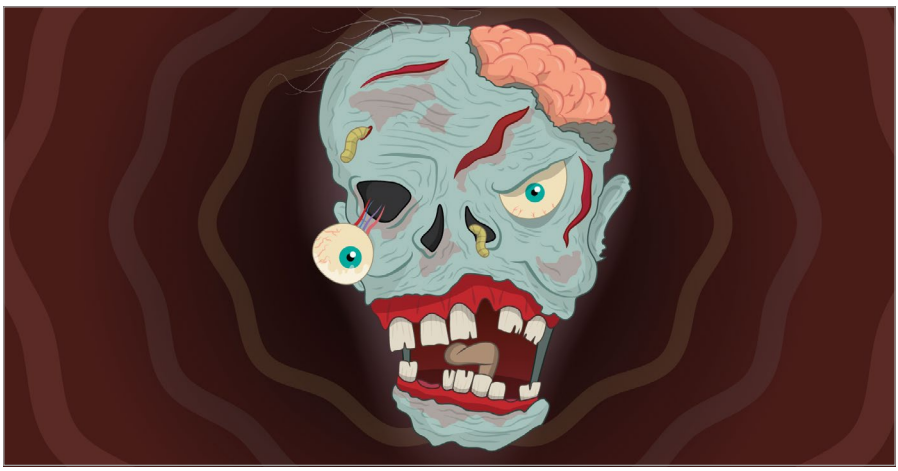
Tilføj en forsinkelse, så zombiebilledet ikke vises med det samme:

```
sleep(3)
```

Blit derefter billedet til skærmen, og opdater det, så det vises for afspilleren:

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

Klik på ikonet Run (Kør) og se hvad der sker: Pygame indlæser dit Find forskellen-billede, men efter tre sekunder vil det blive erstattet med den skræmmende zombie (**Figur 5-16**)!



▲ **Figur 5-16:** Det vil give nogen en skræmmende overraskelse

At have forsinkelsen indstillet til tre sekunder gør dog tingene lidt forudsigelige. Rediger linjen **`sleep(3)`** over **`screen.blit(zombie, (0,0))`** til:

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

Dette vælger et tilfældigt tal mellem 5 og 15 og forsinker programmet så længe. Tilføj dernæst følgende linje lige over instruktionen **`sleep`** for at indlæse skrigelydfilen:

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

Gå under instruktionen **`sleep`**, og skriv følgende på en ny linje for at starte lydafspilningen,

så den starter lige før det skræmmende billede faktisk for afspilleren:

```
scream.play()
```

Til sidst skal du bede Pygame om at stoppe med at afspille lyden ved at skrive følgende linje lige over **pygame.quit()**:

```
scream.stop()
```

Klik på Run (Kør)-ikonet og se dit mesterværk: Efter et par sekunder med uskyldig find-forskellen-sjov vises din skræmmende zombie sammen med et skrækindjagende skrig – som nok skal gøre alle bange! Hvis du finder ud af, at zombiebilledet vises, før lyden begynder at spille, kan du kompensere ved at tilføje en lille forsinkelse lige efter instruktionen **scream.play()** og før instruktionen **screen.blit**:

```
sleep(0.4)
```

Dit færdige program skal se sådan ud:

```
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()
```

Nu er alt, hvad der er tilbage at gøre, at invitere dine venner til at spille Find forskellen – og selvfølgelig sørge for, at der er skruet op for højttalerne!

UDFORDRING: SKIFT UDSEENDE

Kan du ændre billederne for at gøre pranken mere passende til andre situationer, som f.eks. jul? Kan du tegne dine egne Find forskellen-billeder og uhyggelige billeder (ved hjælp af en grafikeditor som GIMP)? Kan du spore brugeren, der klikker på en forskel, for at gøre det mere overbevisende?

Projekt 3: RPG-labyrint

Nu hvor du er ved at være god til Python, det er tid til at bruge Pygame til at gøre noget lidt mere kompliceret: Et fuldt funktionelt tekstbaseret labyrintspil, der er baseret på klassiske rollespil. Disse spil, der også kaldes teksteventyr eller interaktiv fiktion, går tilbage til dengang, hvor computere ikke kunne håndtere grafik. Spillene har dog stadig har deres fans, som hævder, at ingen grafik nogensinde vil være så levende som den, du har i din fantasi!

ONLINEPROJEKT

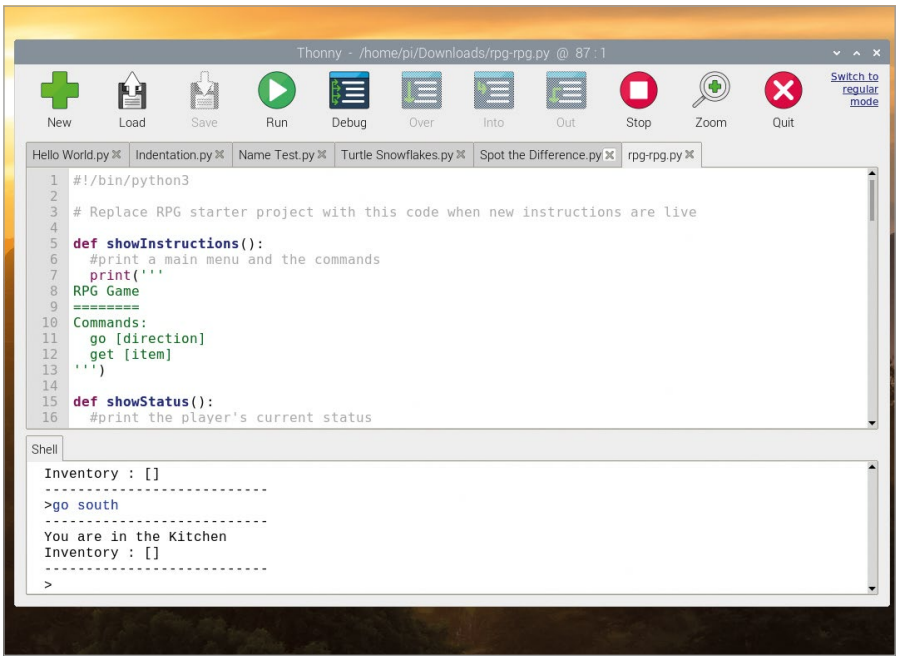
Dette projekt er også tilgængeligt online på **rpf.io/python-rpg**

Dette program er noget mere komplekst end de andre i dette kapitel, så for at gøre tingene lettere starter du med en version, der allerede er delvist skrevet. Åbn Chromium-webbrowseren, og gå til følgende adresse: **rpf.io/rpg-code**.

Chromium-webbrowseren downloader automatisk koden til programmet til mappen Overførsler, men advarer dig om, at filtypen – et Python-program – kan skade din computer. Du har downloadet filen fra Raspberry Pi Foundation, en pålidelig kilde, så klik på knappen Keep (Behold) i advarselsmeddelelsen, som vises nederst på skærmen. Gå tilbage til Thonny, og klik derefter på ikonet Load (Indlæs) . Find filen **rpg-rpg.py** i mappen Overførsler, og klik på knappen Load (Indlæs).

Start med at klikke på ikonet Run (Kør) for at gøre dig bekendt med, hvordan et teksteventyr fungerer. Spillets output vises i shell-området i bunden af Thonny-vinduet; gør Thonny-vinduet større ved at klikke på maksimeringsknappen for at gøre det lettere at læse.

Spillet, som det ser ud nu, er meget simpelt: der er to rum og ingen objekter. Spilleren starter i hallen, det første af de to rum. For at gå til køkkenet skal du blot skrive "go south" (gå mod syd) og trykke på tasten **ENTER** (**Figur 5-17**). Når du er i køkkenet, kan du skrive "go north" (gå mod nord) for at vende tilbage til hallen. Du kan også prøve at skrive "go west" (gå mod vest) og "go east" (gå mod øst), men da der ikke er nogen rum i disse retninger, viser spillet dig en fejlmeddelelse.



Figur 5-17: Der er kun to værelser indtil videre

Rul ned til linje 29 i programmet i scriptområdet for at finde en variabel med navn **`rooms`**. Denne type variabel er kendt som en *ordbog* og fortæller spillet om værelserne, deres udgange, og hvilket rum en given udgang fører til.

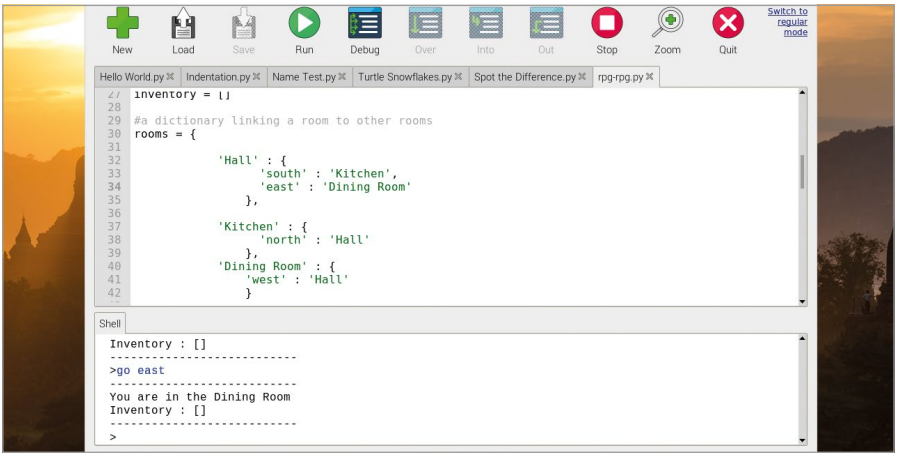
For at gøre spillet mere interessant skal du tilføje et andet rum: En spisestue øst for hallen. Find variablen **`rooms`** i scriptsområdet, og udvid den ved at tilføje et komma-symbol (**`,`**) efter **`}`** på linje 38, og skriv derefter følgende (nøjagtig indrykning er ikke nødvendig i en ordbog):

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

Du skal også bruge en ny udgang i hallen, da der ikke automatisk oprettes en til dig. Gå til slutningen af linje 33, tilføj et komma, og tilføj derefter følgende linje:

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

Klik på ikonet Run (Kør), og prøv dit nye rum: Skriv "go east" (gå mod øst), mens du er i hallen for at komme ind i spisestuen (**Figur 5-18**, på næste side), og skriv "go west" (gå mod vest), mens du er i spisestuen for at komme ind i hallen. Tillykke: Du har selv oprettet et rum!

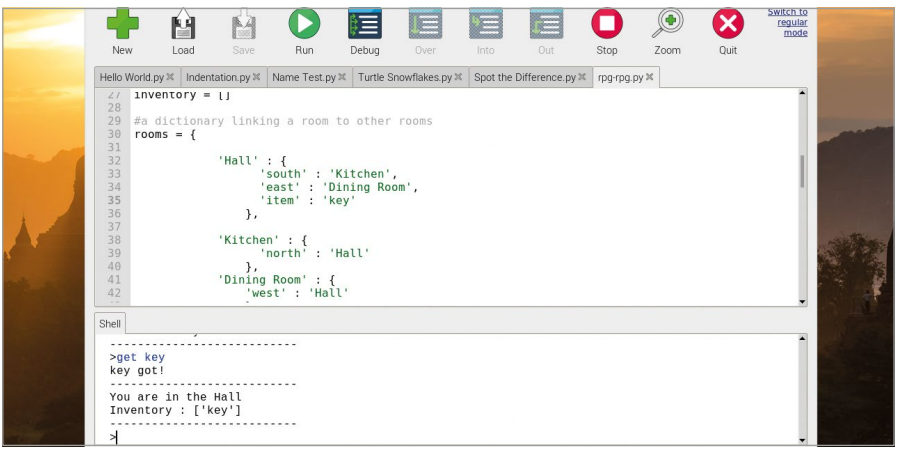


▲ **Figur 5-18:** Du har tilføjet et ekstra rum

Tomme rum er dog ikke særligt sjove. Hvis du vil føje et element til et rum, skal du ændre det pågældende rums ordbog. Stop programmet ved at klikke på Stop-ikonet. Find **`Hall`**-ordbogen i scriptområdet, og tilføj derefter et komma i slutningen af linjen **`'east' : 'Dining Room'`** inden du trykker på **ENTER** og skriver følgende linje:

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

Klik på Run (Kør) igen. Denne gang vil spillet fortælle dig, at du kan se dit nye emne: en nøgle. Skriv "get key" (**Figur 5-19**), så kan du derefter hente den og føje den til listen over elementer, som du har adgang til – kendt som din *beholdning*. Din beholdning forbliver hos dig, når du rejser fra rum til rum.



▲ **Figur 5-19:** Den indsamlede nøgle føjes til din beholdning

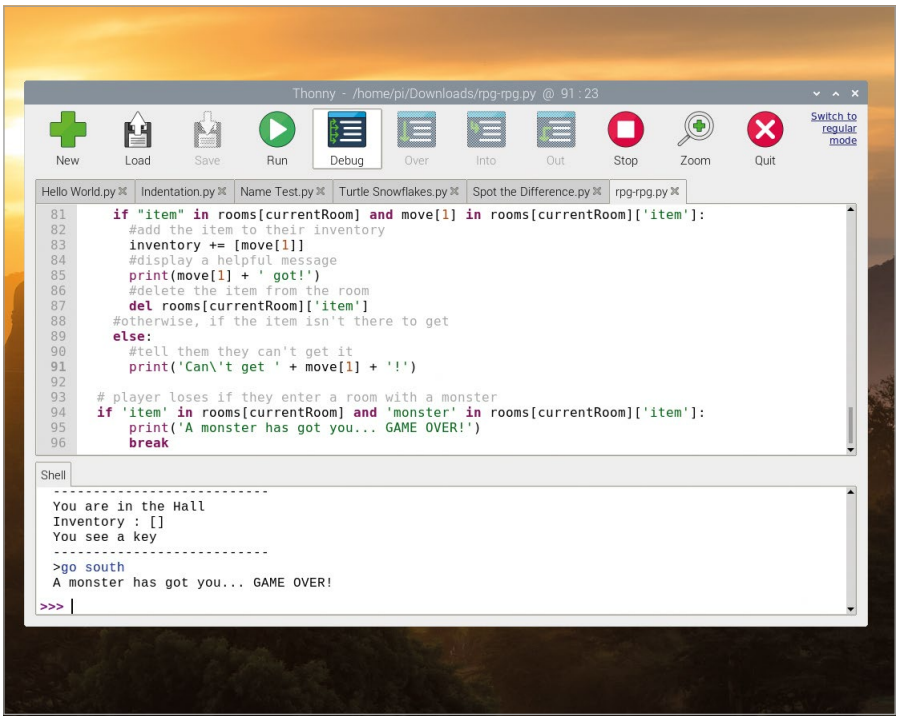
Klik på Stop-ikonet, og gør spillet mere interessant ved at tilføje et monster, der skal undgås. Find **`Kitchen`**-ordbogen, og tilføj et "monster"-element på samme måde som du tilføjede elementet "key" – husk at tilføje et komma til slutningen af linjen ovenfor:

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

For at monsteret skal kunne angribe spilleren, skal du tilføje noget logik til spillet. Rul helt til bunden af programmet i scriptområdet, og tilføj følgende linjer – inklusive kommentaren markeret med et hash-symbol, som hjælper dig med at forstå programmet, hvis du kommer tilbage til det en anden dag – og sørg for at indrykke linjerne:

```
    # spiller taber, hvis han/hun går ind i et rum med et monster
    if 'item' in rooms[currentRoom] and 'monster' in
rooms[currentRoom]['item']:
        print('Monsteret har spist dig ... SPILLET ER FORBI!')
        break
```

Klik på Run (Kør), og prøv at gå ind i køkkenrummet (**Figur 5-20**) – monsteret bliver ikke for imponeret, når du gør det!



▲ **Figur 5-20:** Glem alt om rotter, der er et monster i køkkenet

For at gøre dette eventyr til et rigtigt spil skal du bruge flere ting, et andet rum og muligheden for at "vinde" ved at forlade huset med alle genstandene sikkert i din beholdning. Start med at tilføje et ekstra rum, ligesom du gjorde for Dining Room (spisestuen) – men denne gang skal det være en have. Tilføj en udgang fra Dining Room-ordbogen, og husk at tilføje et komma i slutningen af linjen ovenfor:

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

Føj derefter dit nye rum til **rooms**-ordbogen, og husk igen at tilføje et komma efter **}** på linjen ovenfor som før:

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

Føj et trylledriksobjekt (potion) til Dining Room-ordbogen, og husk igen at tilføje det nødvendige komma til linjen ovenfor:

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

Til sidst skal du rulle til bunden af programmet og tilføje den nødvendige logik for at kontrollere, om spilleren har alle emnerne, og i så fald fortælle, at vedkommende har vundet spillet:

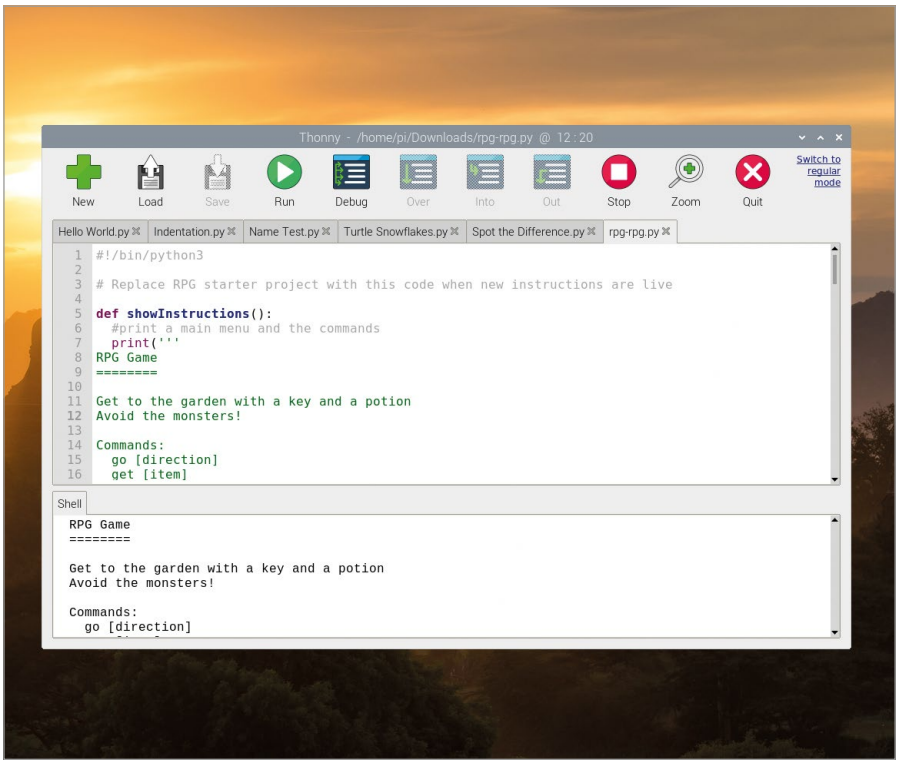
```
   # spilleren vinder, hvis vedkommende når til haven med en nøgle
og en trylledrik
      if currentRoom == 'Garden' and 'key' in inventory and
'potion' in inventory:
          print('Du slap ud af huset ... DU VINDER!')
          break
```

Klik på Run (Kør), og prøv at afslutte spillet ved at hente nøglen og trylledrikken, inden du går ud i haven. Husk ikke at gå ind i køkkenrummet, for det er her monsteret er!

Som en sidste finjustering af spillet skal du tilføje nogle instruktioner, der fortæller spilleren, hvordan man fuldfører spillet. Rul til toppen af programmet, hvor funktionen **showInstructions()** er defineret, og tilføj følgende:

```
Nå ud i haven med en nøgle og en trylledrik
Undgå monstrene!
```

Kør spillet en sidste gang, så ser du dine nye instruktioner vises helt i starten (**Figur 5-21**). Tillykke: Du har lavet et interaktivt tekstbaseret labyrint-spil!



▲ **Figur 5-21:** Nu ved spilleren, hvad der skal gøres

UDFORDRING: UDVID SPILLET

Kan du tilføje flere rum for at få spillet til at vare længere? Kan du tilføje et element for at beskytte dig mod monsteret? Hvordan ville du tilføje et våben for at dræbe monsteret? Kan du tilføje rum, der er over og under de eksisterende rum, som der er adgang til via trapper?

Kapitel 6

Physical computing med Scratch og Python

Programmering drejer sikke ikke kun om at styre ting på skærmen – du kan også styre elektroniske komponenter, der er tilsluttet til din Raspberry Pi's GPIO-stik

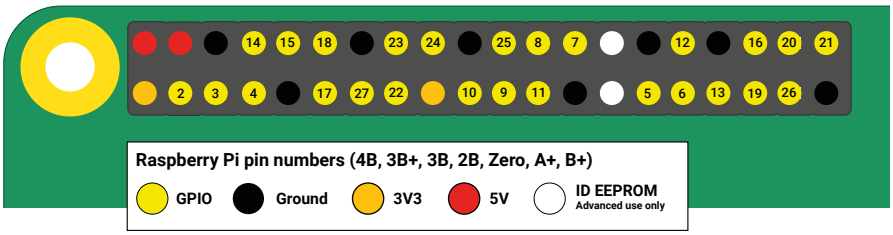
Når man nævner "programmering" eller "kodning", tænker man normalt på software. Kodning kan dog handle om mere end bare software, fordi man også kan påvirke den virkelige verden via hardware. Dette område inden for informatik kaldes oftest med den engelske betegnelse *physical computing*, dvs. anvendelse af computerteknologi på fysiske genstande.

Som navnet antyder, drejer physical computing sig om at kontrollere ting i den virkelige verden ved hjælp af software, dvs. man snarere styrer hardware end software. Når du sætter din vaskemaskine til et program, ændrer temperaturen på din programmerbare termostat eller trykker på en knap ved en trafiklysstyret fodgængerovergang, kommer du i berøring med physical computing.

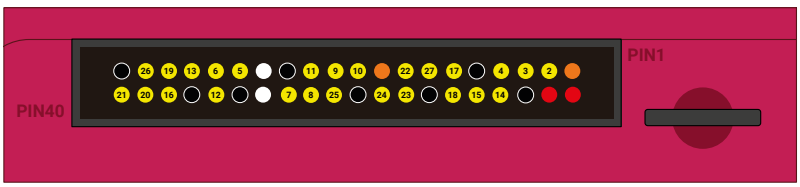
Raspberry Pi er en supereffektiv enhed til at lære om physical computing, takket være dens meget vigtige komponent: *GPIO-stikket til generel input/output*.

Introduktion til GPIO-stikket

GPIO-stikket findes i den øverste del af Raspberry Pi-printkortet eller på bagsiden af Raspberry Pi 400. Det ligner to lange rækker nåle eller ben og bruges til at tilslutte hardware som LED-lys og switches til Raspberry Pi, så de kan kontrolleres via de programmer, du koder. Benene kan bruges til både input og output.



GPIO-stikket på Raspberry Pi er et hanstik med 40 ben. Nogle af disse ben kan bruges til at sende og modtage signaler fra fysiske objekter, nogle er strømførende, mens andre bruges til kommunikation med hardwaretilbehør som Sense HAT (se **Kapitel 7**).



Raspberry Pi 400 har det samme GPIO-stik med samme ben, men stikket vender på hovedet i forhold til andre Raspberry Pi-modeller. Diagrammet viser GPIO-stikket som set fra bagsiden af Raspberry Pi 400. Dobbelttjek altid, at du bruger den rigtige orientering, når du tilslutter noget til GPIO-stikket på din Raspberry Pi 400 – det er nemt at tilslutte noget omvendt, selvom der er "Pin 40"- og "Pin 1"-mærker på kabinettet.

GPIO-UDVIDELSER

Man kan fint bruge Raspberry Pi 400s GPIO-stikket som det er, men en udvidelse kan gøre tingene lidt nemmere. Med en udvidelse har du benene på siden af Raspberry Pi 400, så du kan forbinde og kontrollere dine ledninger uden hele tiden at skulle kigge om på bagsiden.

Der er flere kompatible udvidelser, blandt andet Black HAT Hack3r-serien fra **pimoroni.com** og Pi T-Cobbler Plus fra **adafruit.com**.

Hvis du beslutter dig at købe en udvidelse, skal du være opmærksom på, at de kan have forskellige layouts – f.eks. har Pi T-Cobbler Plus et andet layout end GPIO-stikkene i din Raspberry Pie. Brug altid producentens instruktioner, hvis du er i tvivl.

Benene i stikket er opdelt i forskellige kategorier med bestemte funktioner:

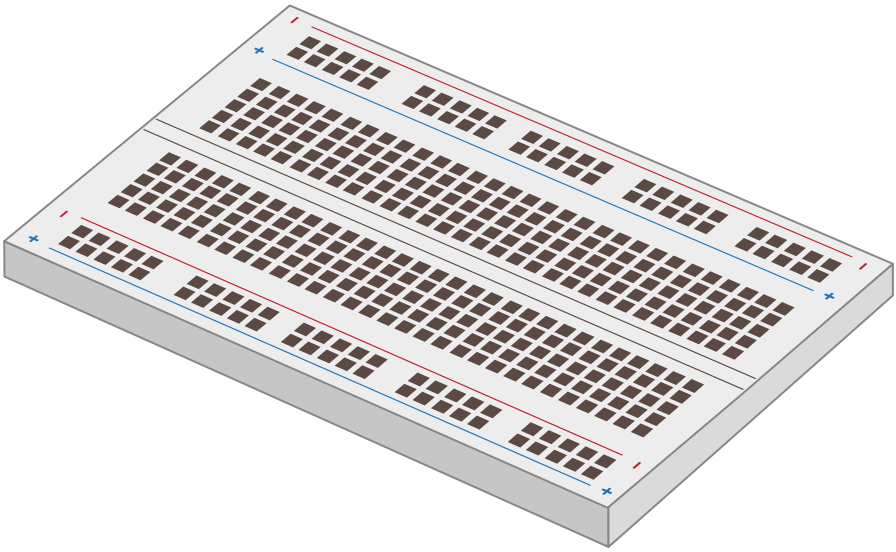
3V3	**Strøm med 3,3 V-spænding**	En permanent strømkilde med 3,3 V – samme spænding, som bruges internt i Raspberry Pi
5V	**Strøm med 5 V-spænding**	En permanent strømkilde med 5 V – samme spænding, som bruges i mikro-USB-strømstikket på Raspberry Pi
Ground (GND)	**Jord med 0 V-spænding**	En jordforbindelse, der bruges til at afslutte et kredsløb, der tilsluttes en strømkilde
GPIO XX	**GPIO-stik nummer "XX"**	De GPIO-ben, som du kan bruge til dine programmer. De er nummereret fra 2 til 27
ID EEPROM	**Ben reserveret til særlige formål**	Ben reserveret til brug med HAT-hardware (Hardware Attached on Top) og andet tilbehør

ADVARSEL!

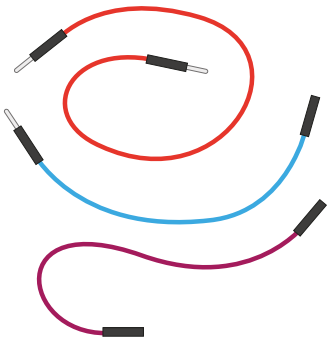
GPIO-stikket på din Raspberry Pi er en sjov og sikker måde at eksperimentere med physical computing på, men det skal håndteres forsigtigt. Pas på ikke at bøje benene, når du tilslutter og afbryder hardware. Forbind aldrig to ben direkte med hinanden, hverken tilfældigt eller bevidst, medmindre instruktionerne til projektet udtrykkeligt beder dig om det. Dette kaldes en kortslutning og kan, afhængigt af hvilke ben der kortsluttes, permanent beskadige din Raspberry Pi.

Elektroniske komponenter

GPIO-stikket er kun en del af det, du har brug for at begynde at arbejde med physical computing. Den anden halvdel består af elektriske komponenter, dvs. de enheder, du styrer via GPIO-stikket. Der findes tusindvis af forskellige komponenter, du kan bruge, men de fleste GPIO-projekter indebærer følgende komponenter.

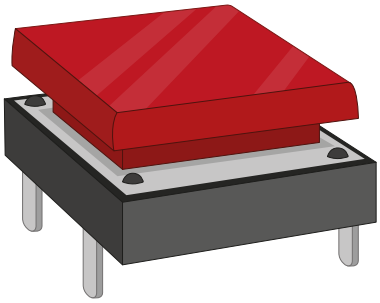


Et *breadboard*, eller som det også nogle gange kaldes en *eksperimentplade*, gør det meget nemmere at arbejde med physical computing-projekter. I stedet for at have en masse separate komponenter, der skal loddes og forbindes med ledninger, kan du med et breadboard forbinde komponenterne via forbindelser skjult under breadboardets overflade. Mange breadboards har også særlige afsnit til strøm, hvilket gør det endnu lettere at opbygge kredsløb. Et breadboard er ikke et must som sådan for at komme i gang med physical computing, men det gør tingene noget nemmere.

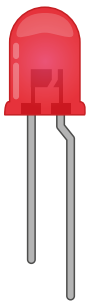


Jumperledninger, der bruges til at forbinde komponenterne med din Raspberry Pi og, hvis du ikke bruger et breadboard, med hinanden. De fås i tre versioner: M2F (han-til-hun), som forbinder et breadboard til GPIO-stikket, F2F (hun-til-hun), som kan forbinde de enkelte komponenter sammen, hvis du ikke bruger et breadboard, og M2M (han-til-han), som bruges til at lave forbindelser inden for breadboardet. Afhængigt af dit projekt har du muligvis brug for alle tre typer jumperledninger. Hvis du bruger et breadboard, kan du normalt nøjes med bare M2F- og M2M-jumperledninger.

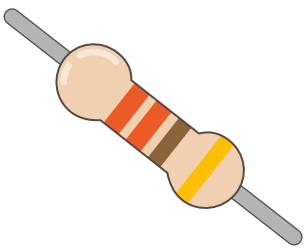
En *trykknapkontakt* er en simpel og effektiv kontakt, der blandt andet bruges i controllere til spillekonsoller. De findes normalt med to eller fire ben, og begge typer fungerer sammen med Raspberry Pi. En trykknapkontakt er en inputenhed, som dit program kan overvåge og udføre en opgave, når der trykkes på kontakten. En anden almindelig type kontakt er en *omskifter*. Til forskel fra en trykknapkontakt, der kun er aktiv, når den holdes nede, forbliver en omskifter aktiveret, indtil du trykker på den igen.



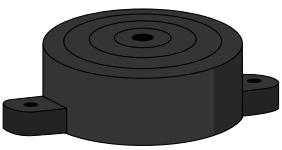
En *LED-lysdiode* er en *outputenhed*, der kan styres direkte fra dit program. En lysdiode lyser, når den er tændt. Lysdioder findes overalt i vores hjem og kan være små – f.eks. det lille lys, der viser, at vaskemaskinen er tændt – og store, når de bruges til belysning af rum. Lysdioder fås i en bred vifte af former, farver og størrelser, men ikke alle er egnede til brug med Raspberry Pi. Du kan f.eks. ikke bruge lysdioder, der kører på 5 V eller 12 V.



Modstande er komponenter, der kan begrænse gennemstrømningen af den *elektriske strøm*. De findes i forskellige udgaver, og deres evne til at yde modstand måles ved hjælp af en enhed kaldet *ohm* (Ω). Jo flere ohm, jo større modstand. I physical computing-projekter, der håndteres vha. Raspberry Pi, bruges en modstand typisk til at forhindre lysdioder i at trække for meget strøm og derved beskadige sig selv eller din Raspberry Pi. Modstande, der bruges til dette formål, er typisk på ca. 330 Ω, men mange elektronikbutikker sælger praktiske pakker, der indeholder en række forskellige modstande, så du kan finde noget til ethvert formål.



En *piezoelektrisk buzzer*, normalt bare kaldet en buzzer eller en summer, er en anden outputenhed. Mens en lysdiode producerer lys, producerer en buzzer en summelyd. Inde i buzzerens plastikhus er der et par metalplader, og når disse plader aktiveres, vibrerer de mod hinanden og udsender en summelyd. Der findes to typer buzzere: *aktive buzzere* og *passive buzzere*. Det anbefales at bruge en aktiv buzzer i dine projekter, da de er nemmest at bruge.



Andre almindelige elektriske komponenter er motorer, der skal bruges sammen med et særligt kontrolkort, før de kan tilsluttes Raspberry Pi, infrarøde bevægelsessensorer, temperatur- og fugtighedssensorer, der kan bruges til vejrprognoser, og såkaldte LDR-komponenter, som er lysafhængige inputenheder, der fungerer som en slags omvendte lysdioder, idet de kan registrere lys.

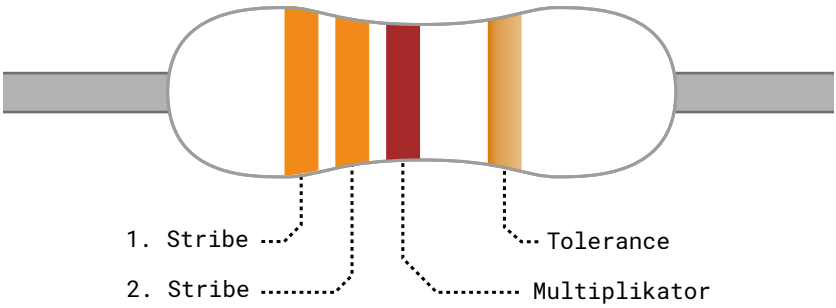
Der er mange producenter over hele verden, der leverer physical computing-komponenter til Raspberry Pi, enten som separate dele eller som sæt, så du altid kan finde, hvad du har brug for. Der er en oversigt over de forskellige producenter på **rpf.io/products**. Når du klikker på Raspberry Pi 4, får du en liste over Raspberry Pi-partneres onlinebutikker (de er allesammen godkendte af os) i dit land eller område.

Du skal bruge følgende for at gennemføre projekterne i dette kapitel:

- 3 × lysdioder: rød, grøn og gul eller orange
- 2 × trykknapkontakter
- 1 × aktiv buzzer
- M2F- (han- til hunstik) og F2F-jumperledninger (hun- til hunstik)
- Det er ikke strengt nødvendigt men ville være praktisk at have et breadboard og M2M-jumperledninger (han- til hanstik)

Farvekoder på modstande

Der findes mange modstande med forskellige værdier, fra varianter med nul modstand, der faktisk kun består af et stykke ledning, til kraftige modstande, der er lige så tykke som dit lår. Kun få modstande har deres modstandsværdi angivet med tal. I stedet bruger man specielle farvekoder i form af en kombination af farvestriber påtrykt på selve modstanden.



	1./2. Stribe	Multiplikator	Tolerance
Sort	0	$\times 10^{0}$	-
Brun	1	$\times 10^{1}$	±1%
Rød	2	$\times 10^{2}$	±2%
Orange	3	$\times 10^{3}$	-
Gul	4	$\times 10^{4}$	-
Grøn	5	$\times 10^{5}$	±0.5%
Blå	6	$\times 10^{6}$	±0.25%
Violet	7	$\times 10^{7}$	±0.1%
Grå	8	$\times 10^{8}$	±0.05%
Hvid	9	$\times 10^{9}$	-
Guld	-	$\times 10^{-1}$	±5%
Sølv	-	$\times 10^{-2}$	±10%
Ingen	-	-	±20%

Værdien af en modstand aflæses ved at placere modstanden, så gruppen af farvestriberne er i venstre side, og den enkelte stribe er i højre side. Start med den første stribe, og brug kolonnen "1./2. Stribe" i tabellen til at aflæse farvekoden på det første og det andet ciffer. I dette eksempel har vi to orange striber, der hver har værdien "3", hvilket til sammen giver "33". Hvis modstanden har fire striber i gruppen i stedet for tre, skal du også aflæse værdien af den tredje stribe (du kan læse om, hvordan du afkoder farvekoden på modstande med fem eller seks striber, i **rpf.io/5-6band**).

Aflæs farvekoden på den sidste stribe i gruppen – dvs. den tredje eller fjerde stribe – i kolonnen "Multiplikator". Dette fortæller dig, hvilket tal du skal gange resultatet af striberne til

venstre med for at få modstandens faktiske værdi. I dette eksempel har vi en brun stribe, så vi skal gange med 10^1. Det ser måske forvirrende ud, men det et simpelt *matematisk udtryk*: "$\times 10^1$" betyder, at vi skal tilføje et enkelt nul efter tallet. Hvis striben havde været blå, havde den betydet $\times 10^6$, dvs. vi skulle sætte seks nuller efter tallet.

Når vi sætter et nul efter 33 fra de orange striber, får vi tallet 330 – og dette er modstandens værdi målt i ohm. Det sidste stribe, der er placeret alene i højre side, angiver modstandens *tolerance*. Denne værdi udtrykker, hvor tæt på den angivne værdi vil modstanden være. Billige modstande har typisk en sølvfarvet stribe, hvilket betyder, at den faktiske modstand kan være 10 procent højere eller 10 procent lavere end den angivne værdi, og de helt billige har slet ikke nogen stribe, hvilket giver en afvigelse på ±20 procent. Dyre kvalitetsmodstande vil så have en grå stribe, hvilket kun giver en ±0,05 procent af den angivne værdi. I små hobbyprojekter er denne afvigelse ikke så vigtig, så du kan bruge modstande med alle tolerancevæ rdier.

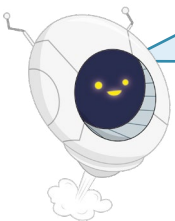
Modstandsværdier på over 1000 ohm (1000 Ω) udtrykkes normalt i kiloohm (kΩ), og værdier på over en million ohm udtrykkes i megohm (MΩ). En modstand på 2200 Ω ville så være angivet som 2,2 kΩ, og en modstand på 2200000 Ω som 2,2 MΩ.

KAN DU FINDE UD AF DET?

Hvilke farvestriber ville en modstand på 100 Ω have? Hvilke farvestriber ville en modstand på 2,2 MΩ have? Hvilken farve på tolerancestriben ville du gå efter, hvis du var på udkig efter den billigste modstand?

Dit første physical computing-program: Hello, LED!

Ligesom visning af et "Hello, World"-hilsen på skærmen er et vigtigt første skridt, når man lærer et programmeringssprog, er det at få en lysdiode til at lyse den traditionelle introduktion i undervisningen i physical computing. I dette projekt skal du bruge en lysdiode og en modstand på 330 ohm (330 Ω) eller så tæt på 330 Ω som du kan finde, plus nogle F2F-jumperledninger (hun- til hunstik).

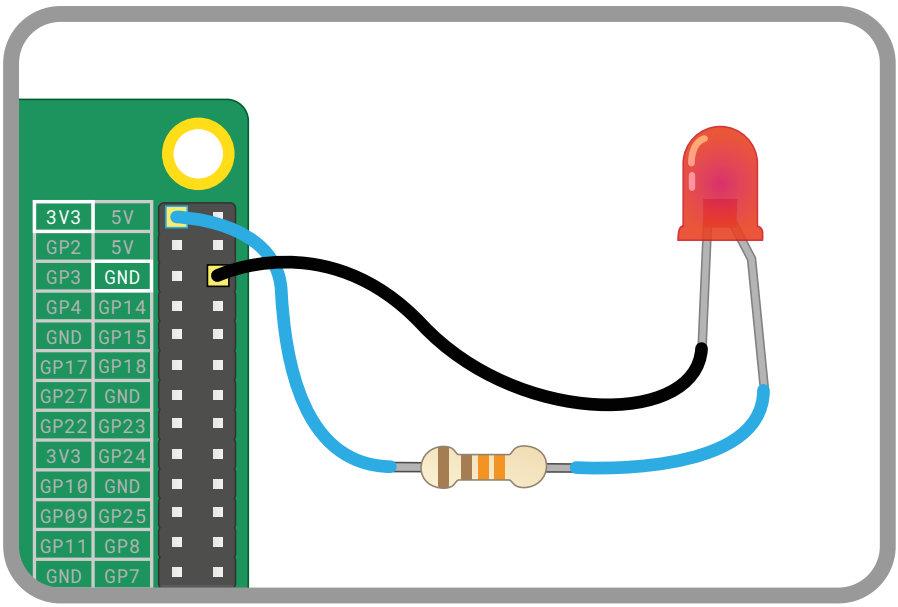


MODSTAND ER VIGTIG

En modstand er en vigtig komponent i dette kredsløb: den beskytter Raspberry Pi og lysdioden ved at reducere den mængde strøm, som lysdioden kan trække. Uden det ville lysdioden trække for meget strøm og brænde ud – og muligvis også beskadige din Raspberry Pi. Når en modstand bruges på denne måde, er dens funktion at *begrænse mængden af strøm*. Den nøjagtige mængde modstand, du har brug for, afhænger af den lysdiode, du bruger, men en modstand på 330 Ω ville fungere med de fleste almindelige lysdioder. Jo stærkere modstand, jo svagere lyser lysdioden, og jo svagere modstand, jo stærkere lyser lysdioden.

Forbind aldrig en lysdiode til en Raspberry Pi uden en modstand, der begrænser mængden af strøm til lysdioden, medmindre du ved, at lysdioden har en indbygget modstand med tilstrækkelig høj værdi.

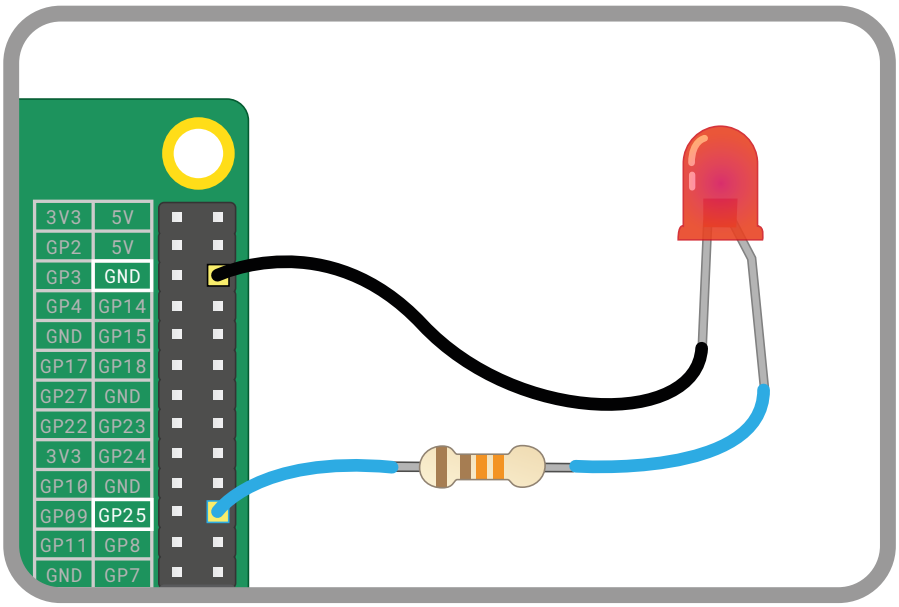
Start med at kontrollere, at lysdioden fungerer. Placer din Raspberry Pi, så GPIO-stikket vender lodret og er i højre side af printkortet. Tilslut den ene ende af 330 Ω-modstanden til det første 3,3 V-ben (mærket 3V3 på **Figur 6-1**) med en F2F-jumperledning, og tilslut derefter den anden ende til det lange ben – positivt ben eller anode – på lysdioden med en anden F2F-jumperledning. Tilslut det korte ben – negativt ben eller katode – på lysdioden til det første jordede ben (mærket GND på **Figur 6-1**), ligeledes med en F2F-jumperledning.



▲ **Figur 6-1:** Tilslut lysdioden til disse ben – og husk modstanden!

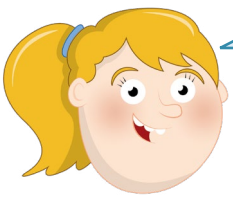
Hvis din Raspberry Pi er tændt, skulle lysdioden gerne lyse. Hvis den ikke gør det, skal du kontrollere kredsløbet: Sørg for, at du ikke har brugt en for stærk modstand, kontrollér, at alle ledninger er korrekt tilsluttet, og dobbelttjek, at du har brugt de rigtige ben i GPIO-stikket, som vist i diagrammet. Tjek også benene på lysdioden, fordi dioder kun fungerer, hvis strømmen ledes igennem dem i den rigtige retning: det lange ben skal være tilsluttet den positive pol i kredsløbet og det korte ben den negative pol.

Når du kar bekræftet, at lysdioden fungerer, er du klar til at programmere den. Frakobl jumperledningen fra 3,3 V-benet (mærket 3V3 på **Figur 6-2**), og tilslut den til ben 25 på GPIO-stikket (mærket GP25 på **Figur 6-2**)). Lysdioden slukker, men bare rolig – det er normalt.



Figur 6-2: Frakobl ledningen fra 3V3, og tilslut den ben 25 på GPIO-stikket

Du er nu klar til at skrive et Scratch- eller Python-program, der tænder og slukker for lysdioden.

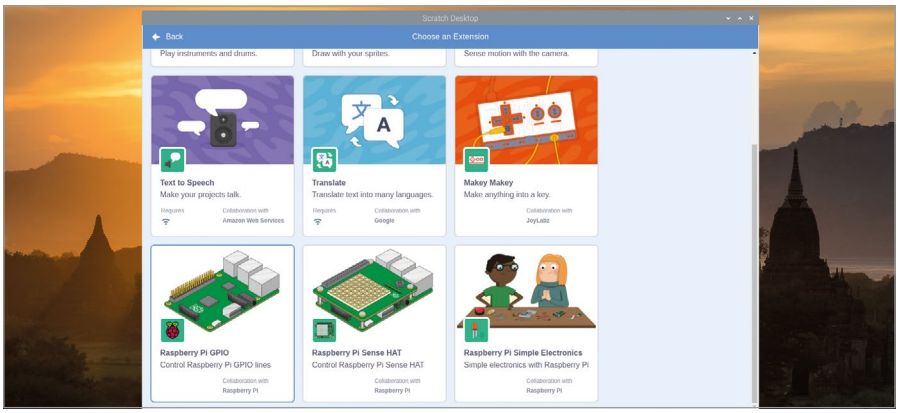


VIDEN OM PROGRAMMERING

Projekterne i dette kapitel kræver, at du er fortrolig brug af Scratch 3 og udviklingsmiljøet Thonny Python. Hvis du ikke allerede har gjort det, skal du gennemføre projekterne i **Kapitel 4, Programmering med Scratch 3** og **Kapitel 5, Programmering med Python**.

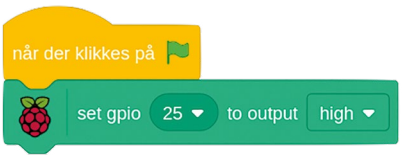
Sådan styrer du en lysdiode med Scratch

Start Scratch 3, og klik på ikonet Add Extension (Tilføj udvidelse) . Scrol ned, og find udvidelsen "Raspberry Pi GPIO" (se **Figur 6-3** på næste side), og klik på den. Dette indlæser de brikker, du har brug for for at styre GPIO-stikket i din Raspberry Pi ved hjælp af Scratch 3. De nye brikker, der skal bruges, findes i kategorien Raspberry Pi GPIO.

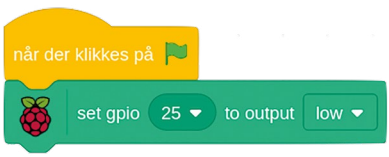


▲ **Figur 6-3:** Tilføj Raspberry Pi GPIO-udvidelsen til Scratch 3

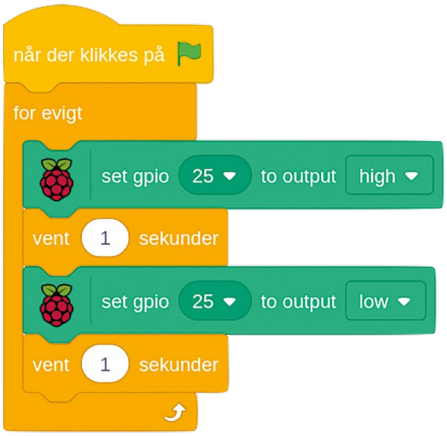
Start med at trække en **når der klikkes på** -hændelsesbrik til kodeområdet, og placer derefter en **set gpio to output high** -brik direkte under den. Du skal angive nummeret på det ben, du bruger: Klik på den lille pil for at åbne rullemenuen, og klik på "25" for at fortælle Scratch, at du styrer ben 25 på GPIO-stikket.



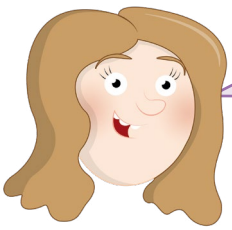
Klik på det grønne flag for at køre programmet. Når lysdioden lyser, har du fuldført dit første physical computing-projekt! Klik på den røde ottekant for at stoppe programmet. Bemærk i øvrigt, at lysdioden stadig er tændt. Det er, fordi programmet kun havde bedt Raspberry Pi om at tænde lysdioden – dette styres med parameteren "output high" på **set gpio 25 to output high** -brikken. For at slukke lysdioden skal du klikke på pileikonet for enden af brikken og vælge "low" på listen.



Klik på det grønne flag igen. Denne gang slukker programmet for lysdioden. For at gøre tingene mere interessante skal du tilføje en **for evigt** -kontrolbrik og et par **vent 1 sekunder** -brikker for at lave et program, der får lysdioden til at blinke med et sekunds mellemrum.



Klik på det grønne flag, og se, hvordan lysdioden opfører sig: Den er tændt i et sekund, slukket i et sekund, tændt i et sekund, og fortsætter så med det, indtil du klikker på den røde ottekant for at stoppe programmet. Se hvad der sker, når du klikker på ottekanten, mens lysdioden er tændt eller slukket.



UDFORDRING: KAN DU ÆNDRE DET?

Hvordan ville du ændre programmet, hvis du ville have lysdioden til at lyse i længere tid ad gangen? Eller hvis den skulle lysere i kortere tid? Hvad er den mindste forsinkelse, du kan bruge, for stadig at være i stand til at se, at lysdioden tændes og slukkes?

Sådan styrer du en lysdiode med Python

Indlæs Thonny fra afsnittet Programmering i hindbærmenuen, klik på knappen Ny for at starte et nyt projekt, og gem projektet under navnet **Hello LED** ved at klikke på Gem. For at styre GPIO-benene vha. Python skal du bruge et bibliotek kaldet GPIO Zero. Til dette projekt skal du kun bruge den del af biblioteket, der bruges til at styre lysdioder. Importér denne del af biblioteket ved at skrive følgende i Python-shellområdet:

```
from gpiozero import LED
```

Nu skal du fortælle GPIO Zero, hvilket GPIO-ben lysdioden er sluttet til.
Skriv følgende:

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

Den første kommando gør det muligt for Python at styre lysdioder, der er tilsluttet GPIO-stikket på din Raspberry Pi, og den anden angiver, hvilket – eller hvilke, hvis du har flere lysdioder i kredsløbet – ben, lysdioderne er sluttet til. Lysdioden tændes vha. følgende kommando:

```
led.on()
```

Lysdioden slukkes igen vha. følgende kommando:

```
led.off()
```

Tillykke, du kan nu bruge Python til at styre GPIO-stikket på din Raspberry Pi! Prøv at skrive de to kommandoer igen. Hvis lysdioden er slukket, gør **`led.off()`** ingenting, og det samme gør sig gældende, når du skriver **`led.on()`**, mens lysdioden er tændt.

For at lave et rigtigt program skal du skrive følgende i scriptområdet:

```
from gpiozero import LED
from time import sleep

led = LED(25)

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

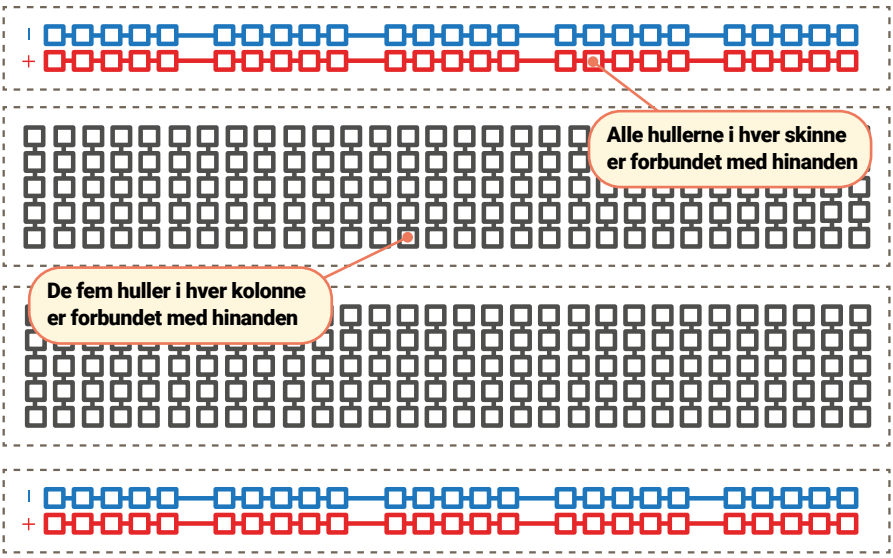
Dette program importerer funktionen LED fra biblioteket **`gpiozero`** (GPIO Zero), funktionen **`sleep`** fra biblioteket **`time`** og derefter laver en uendelig løkke, der tænder lysdioden i et sekund, slukker den i et sekund, og gentager dette i det uendelige. Klik på knappen Kør for at se resultatet: Lysdioden begynder at blinke. Som med Scratch-programmet skal lægge mærke til, hvad der sker, når du klikker på stopknappen, mens lysdioden er tændt, og når den er slukket.

UDFORDRING: LYS I LÆNGERE TID

Hvordan ville du ændre programmet, hvis du ville have lysdioden til at lyse i længere tid ad gangen? Eller hvis den skulle lysere i kortere tid? Hvad er den mindste forsinkelse, du kan bruge, for stadig at være i stand til at se, at lysdioden tændes og slukkes?

Brug af et breadboard

De næste projekter i dette kapitel er lettest at gennemføre, hvis du forbinder komponenterne ved hjælp af et breadboard.



Et breadboard har huller, der er placeret med et 2,54 mm imellem. Under disse huller er der en række metalbånd, der forbinder hullerne med hinanden og fungerer som de jumperledninger, du har brugt indtil nu. Disse metalbånd løber i rækker på tværs af boardet, og de fleste boards har en fordybning i midten for at opdele boardet i to halvdele. Mange breadboards har også bogstaver øverst og tal på siderne. Disse angivelser gør det lettere at finde et bestemt hul: A1 er hullet i øverste venstre hjørne, B1 er hullet til højre for det, og B2 er et hul ned fra det forrige. A1 er forbundet med B1 med et af de skjulte metalbånd, men intet hul i række 1 har nogen forbindelse til noget hul i række 2, medmindre du selv forbinder dem med en jumperledning.

Større breadboards har også hullrækker lang siderne, og disse er typisk markeret med røde og sorte eller røde og blå striber. Disse er *strømskinner*, der er designet til at gøre ledningsføringen lettere. Du kan slutte en enkelt ledning fra Raspberry Pi's jordbenet til en af strømskinnerne – typisk markeret med en blå eller sort stribe og et minussymbol – for at skabe en *fælles jordforbindelse* til mange komponenter på breadboardet, og du kan gøre det samme, hvis du skal bruge 3,3 V- eller 5 V-spænding i kredsløbet.

Det er nemt at montere elektroniske komponenter på et breadboard: Du skal blot stikke deres ben (de metaltråde, der stikker ud) ind i hullerne. Du skal gøre det forsigtigt, indtil komponenten er på plads. Hvis du har brug for at forbinde komponenterne uden om breadboardet, kan du bruge M2M-jumperledninger (han- til hanstik), og for at forbinde breadboardet med din Raspberry Pi skal du bruge M2F-jumperledninger (han- til hunstik).

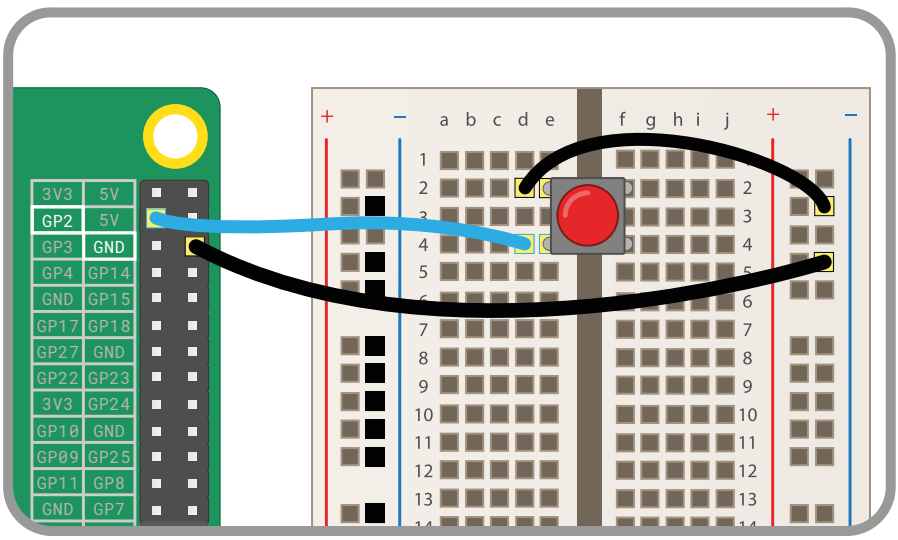
Forsøg aldrig at klemme mere end én komponents ben eller én jumperledning ind i det samme

hul på breadboardet. Husk: hullerne er forbundet i kolonner bortset fra fordybningen i midten, så et komponentben i hul A1 er elektrisk forbundet til alt, hvad du sætter i hul B1, C1, D1 og E1.

Næste trin: Registrering af input fra en knap

Styring af outputkomponenter som lysdioder er kun halvdelen af det sjove, fordi som navnet antyder, kan GPIO-stikket bruges til styring af både output- og inputkomponenter. Til dette projekt skal vi bruge et breadboard, M2M- (han- til hanstik) og M2F-jumperledninger (han- til hunstik) og en trykknapkontakt. Hvis du ikke har et breadboard, kan du bruge F2F-jumperledninger (hun- til hunstik), dog skal du være forsigtig, når du trykker på knappen, så du ikke afbryder kredsløbet.

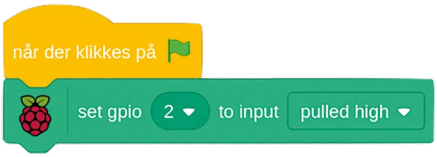
Start med at montere trykknappen på breadboardet. Hvis trykknappen kun har to ben, skal du sørge for at sætte dem i huller med forskellige numre på breadboardet. Hvis den har fire ben, skal du placere den, så de sider, der har ben, flugter med breadboardets rækker, og siderne uden ben vender mod toppen og bunden af breadboardet. Forbind jordskinnen på breadboardet til et jordben på din Raspberry Pi (markeret GND på **Figur 6-4**) med en M2F-jumperledning (han- til hunstik), og forbind derefter det ene ben på trykknappen til jordskinnen med en M2M-jumperledning (han- til hanstik). Forbind til sidst det andet ben – benet på samme side som det ben, du lige har forbundet, hvis du bruger en kontakt med fire ben – til GPIO 2-benet (markeret GP2 på **Figur 6-4**) på din Raspberry Pi med en M2F-jumperledning (han- til hunstik).



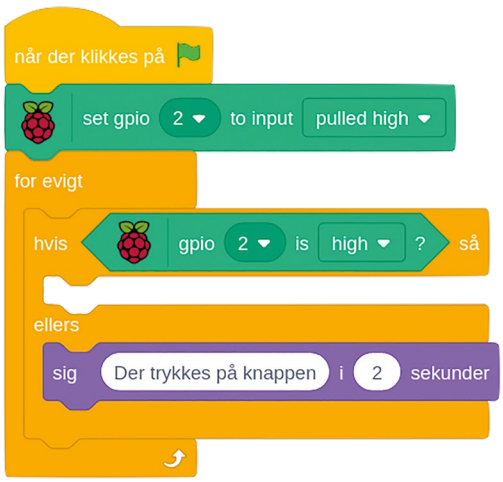
▲ **Figur 6-4:** Tilslutning af en trykknap til GPIO-stikket

Registrering af input fra en knap vha. Scratch

Start et nyt Scratch-program, og træk en [når der klikkes på]-brik til kodeområdet. Tilføj en [set gpio to input pulled high]-brik, og vælg tallet 2 i rullemenuen for at matche det GPIO-ben, du forbandt trykknappen til.



Hvis du klikker på det grønne flag nu, sker der intet. Det er, fordi du har bedt Scratch om at bruge benet som input, men ikke har angivet, hvad denne input skal bruges til. Tilføj en **for evigt** -brik nederst i sekvensen, og læg en **hvis så** -brik inden i denne brik. Træk **gpio is high?** -brikken til den diamantformede åbning i **hvis så** -delen af brikken, og vælg tallet 2 for at angive, hvilket ben på GPIO-stikket der skal kontrolleres. Læg en **sig Hej! i 2 sekunder** -brik i **ellers** -delen af brikken, og skriv "Der er trykket på knappen!" i den. Lad "hvis så"-delen af brikken være tom for nu.



Der foregår flere ting her, men lad os begynde med at teste, hvordan det fungerer: Klik på det grønne flag, og tryk derefter på knappen, du har monteret på dit breadboard. Hvis spriten fortæller dig, at der er blevet trykket på knappen, har du fået programmet til at registrere input fra GPIO-stikket.

Som du måske allerede har bemærket, er **hvis gpio 2 is high? så** -delen af brikken tom. Men den kode, der kører, når der trykkes på knappen, findes i **ellers** -delen af brikken. Det kan virke forvirrende, for man skulle tro, at når man trykker på knappen, skifter tilstanden til "høj"? Men faktisk forholder det sig omvendt: Når benene på GPIO-stikket på din Raspberry Pi er konfigureret til at registrere input, er deres standardtilstand "høj", dvs. de er tændt. Når du trykker på knappen, skifter deres tilstand til "lav", dvs. slukket.

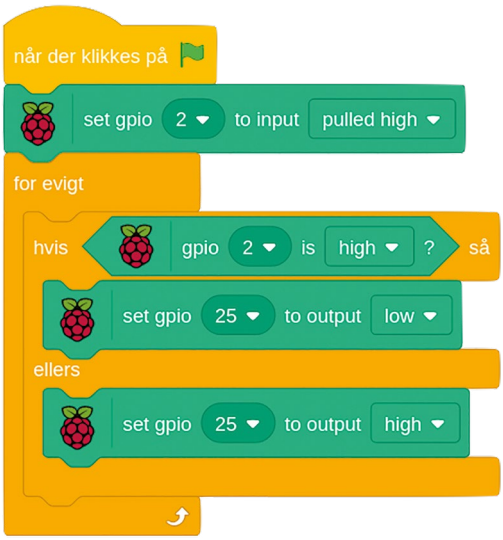
Se kredsløbet igen, og bemærk, hvordan knappen er forbundet til GPIO 2-benet, der forsyner kredsløbet med plus, og jordbenet, der forsyner kredsløbet med minus. Når du trykker på knappen,

ledes spændingen til jordbenet på GPIO-stikket, og Scratch-programmet stopper med at køre koden i hvis gpio 2 is high? så-brikken og kører i stedet koden i ellers -delen af brikken.

Hvis det stadig lyder forvirrende, så husk blot dette: En knap på et Raspberry Pi GPIO-ben trykkes, når benet skifter til lav, ikke når den er i høj.

For at gøre programmet lidt mere avanceret skal du nu tilføje en lysdiode og en modstand til kredsløbet: Husk at forbinde modstanden til GPIO 25-benet og det lange ben på lysdioden, og forbinde den korte ben på lysdioden til jordskinnen på breadboardet.

Fjern sig Der er trykket på knappen! i 2 sekunder-brikken fra kodeområdet ved at flytte den tilbage til brikpaletten, og udskift den med en set gpio 25 to output high-brik. Husk også at ændre nummeret på GPIO-benet i rullemenuen. Tilføj en set gpio 25 to output low-brik (husk ligeledes at ændre værdien) til den del af hvis gpio 2 is high? så -brikken, der lige nu er tom.



Klik på det grønne flag, og tryk på knappen. Lysdioden lyser, når du holder knappen nede, og slukker, når du slipper knappen. Tillykke: Du har lært, hvordan du styrer et GPIO-ben baseret på input fra et andet ben.

UDFORDRING: LAD DET VÆRE TÆNDT

Hvordan ville du ændre programmet, hvis du ville have lysdioden til at blive ved med at lyse i nogle sekunder, efter at du har sluppet knappen? Hvad skal du ændre, så lysdioden er tændt, mens du ikke rører ved knappen, og slukker, når der trykkes på knappen?

Registrering af input fra en knap vha. Python

Start et nyt projekt i Thonny ved at klikke på knappen New (Ny), og gem projektet under navnet **Knapinput**. Et GPIO-ben bruges som input til en knap stort set på samme måde, som et ben bruges som output til en lysdiode, bortset fra, at du skal importere en anden del af GPIO Zero-biblioteket. Skriv følgende i scriptområdet:

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

GPIO Zero-biblioteket indeholder funktionen **wait_for_press**, der bruges til at starte en programfunktion, når der trykkes på en knap. Skriv følgende:

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

Klik på knappen Run (Kør), og tryk derefter på trykknapkontakten. Meddelelsen vises i Pythons shell-område i bunden af Thonny-vinduet. Nu ved du, hvordan du får programmet til at registrere input fra et GPIO-ben. For at køre programmet igen skal du igen klikke på knappen Run (Kør). Der er ikke nogen løkke i programmet, så programmet afsluttes, når det har vist beskeden på skærmen.

For at gøre programmet lidt mere avanceret skal du nu tilføje en lysdiode og en modstand til kredsløbet, hvis du ikke har gjort det allerede: Husk at forbinde modstanden til GPIO 25-benet og det lange ben på lysdioden, og forbinde den korte ben på lysdioden til jordskinnen på breadboardet.

For både at kunne kontrollere en lysdiode og registrere input fra en knap skal du både importere **Button**- og **LED**-funktionerne fra GPIO Zero-biblioteket. Du skal også bruge funktionen **sleep** fra **time**-biblioteket. Vend tilbage til toppen af dit program, og indsæt følgende som de to første linjer:

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

Skriv følgende under **button = Button(2)**:

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

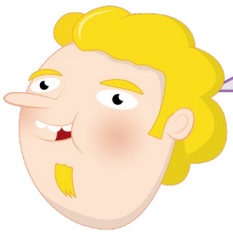
Slet linjen **print("Du har trykket på mig!")**, og skriv i stedet:

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

Det færdige program skal se sådan ud:

```
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()
```

Klik på knappen Run (Kør), og tryk derefter på trykknapkontakten. Lysdioden tændes i tre sekunder, derefter slukkes, og programmet afsluttes. Tillykke: Nu ved du, hvordan du bruger Python til at tænde for en lysdiode, når der trykkes på en knap.



UDFORDRING: TILFØJ EN LØKKE

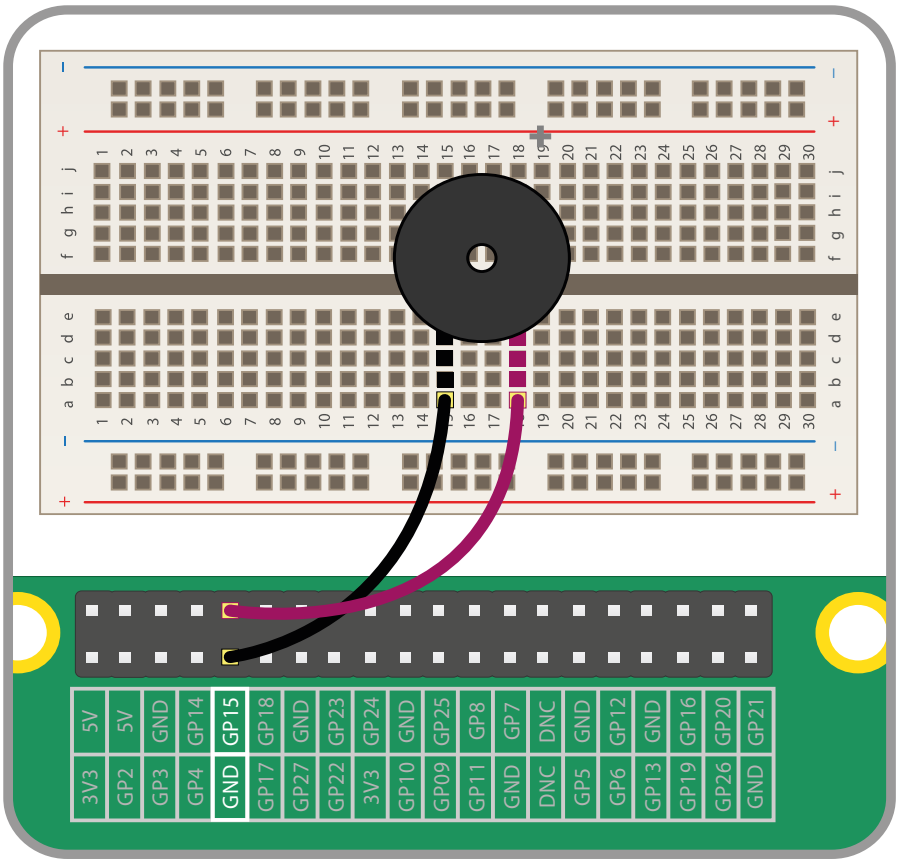
Hvordan ville du lave en løkke, hvis du ville have programmet til at køre igen i stedet for at afslutte, når du har trykket på knappen? Hvad skal du ændre, så lysdioden er tændt, mens du ikke rører ved knappen, og slukker, når der trykkes på knappen?

Lad os lave lidt støj: Sådan bruger du en buzzer

Lysdioder er gode som outputenheder, men de kræver, at man holder øje med dem. En løsning kan være en buzzer, der udsender et lydsignal, der kan høres i hele rummet. Til dette projekt skal vi bruge et breadboard, M2F-jumperledninger (han- til hunstik) og en aktiv buzzer. Hvis du ikke har et breadboard, kan du forbinde buzzeren ved hjælp af F2F-jumperledninger (hun- til hunstik) i stedet.

En aktiv buzzer håndteres nøjagtigt på samme måde som en lysdiode med hensyn til placering i kredsløb og programmering. Så du skal lave det samme kredsløb, som du lavede til lysdioden, bortset fra, at du skal bruge en buzzer i stedet for en lysdiode, og du skal ikke bruge nogen modstand, fordi en buzzer bruger mere strøm i forhold til en lysdiode. Forbind det ene ben på buzzeren til GPIO 15-benet (mærket GP15 på **Figur 6-5**) og det andet til jordbenet (mærket GND i diagrammet) ved hjælp af dit breadboard og M2F-jumperledninger (han- til hunstik).

Hvis buzzeren har tre ben, skal du forbinde benet, der er markeret med et minussymbol (-), til jordbenet, og benet, der er markeret med et "S" eller "SIGNAL", til GPIO 15. Det sidste ben, der normalt sidder i midten, skal forbindes til 3,3 V (GPIO-benet mærket 3V3.)



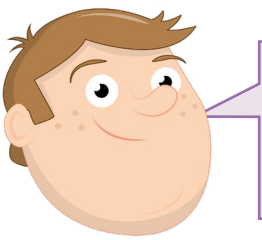
▲ **Figur 6-5:** Tilslutning af en buzzer til GPIO-stikket

Sådan styrer du en buzzer med Scratch

Du kan bruge det samme projekt, du lavede til at få en lysdiode til at blinke, så indlæs det, hvis du havde gemt det, inden du lavede projektet med knappen. Ellers så lav det forfra på samme måde, som før. Vælg tallet 15 i rullemenuen i **set gpio to output high**-brikkerne, så Scratch styrer det rigtige ben i GPIO-stikket.

Når du klikker på det grønne flag, begynder buzzeren at summe. Den summer i et sekund og holder så et sekunds pause. Hvis buzzeren kun laver en kliklyd en gang i sekundet, bruger du en passiv buzzer og ikke en aktiv buzzer. I modsætning til en aktiv buzzer, der selv genererer et signal med hurtigt skiftende frekvens – en såkaldt *oscillation* – der får metalpladerne til at vibrere og udsende summelyden, skal en passiv buzzer have oscillationssignalet tilført udefra. Når du tænder for en passiv buzzer ved hjælp af Scratch, bevæger pladerne sig kun én gang og derefter stopper – hvilket giver kliklyden – indtil næste gang, programmet tænder eller slukker for benet.

Klik på den røde ottekant for at stoppe buzzeren, men gør det, mens den er stille, for ellers fortsætter buzzeren med at summe, indtil du starter programmet igen.



UDFORDRING: EN ANDEN SUMMEN

Hvordan ville du ændre programmet, hvis du ville have buzzeren til at summe i kortere tid ad gangen? Kan du lave et kredsløb til at styre buzzeren med en knap?

Sådan styrer du en buzzer med Python

En aktiv buzzer styres vha. GPIO Zero-biblioteket næsten på samme måde som en lysdiode, idet en buzzer også har en on- og off-tilstand (tændt og slukket). Du skal dog bruge en anden funktion, kaldet **`buzzer`**. Start et nyt projekt i Thonny, og gem det som **Buzzer**. Skriv derefter følgende:

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

Som det også er tilfældet med lysdioder, skal GPIO Zero vide, hvilket ben buzzeren er forbundet til. Skriv følgende:

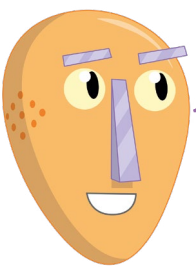
```
buzzer = Buzzer(15)
```

Herfra er programmet stort set identisk med det, du lavede til styring af lysdioden. Den eneste forskel (bortset fra et andet GPIO-bennummer) er, at du bruger en **`buzzer`** i stedet for en **`led`**. Skriv følgende:

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

Når du klikker på knappen Run (Kør), begynder buzzeren at summe. Den summer i et sekund og holder så et sekunds pause. Hvis du bruger en passiv buzzer i stedet for en aktiv buzzer, vil du kun høre et kort klik hvert sekund i stedet for en kontinuerlig summelyd. Dette skyldes, at en passiv buzzer ikke har en indbygget *oscillator* til at generere signalet med hurtigt skiftende frekvens, som får pladerne inde i buzzeren til at vibrere.

Klik på knappen Stop for at afslutte programmet, men gør det, mens buzzeren er stille, for ellers fortsætter den med at summe, indtil du kører programmet igen.



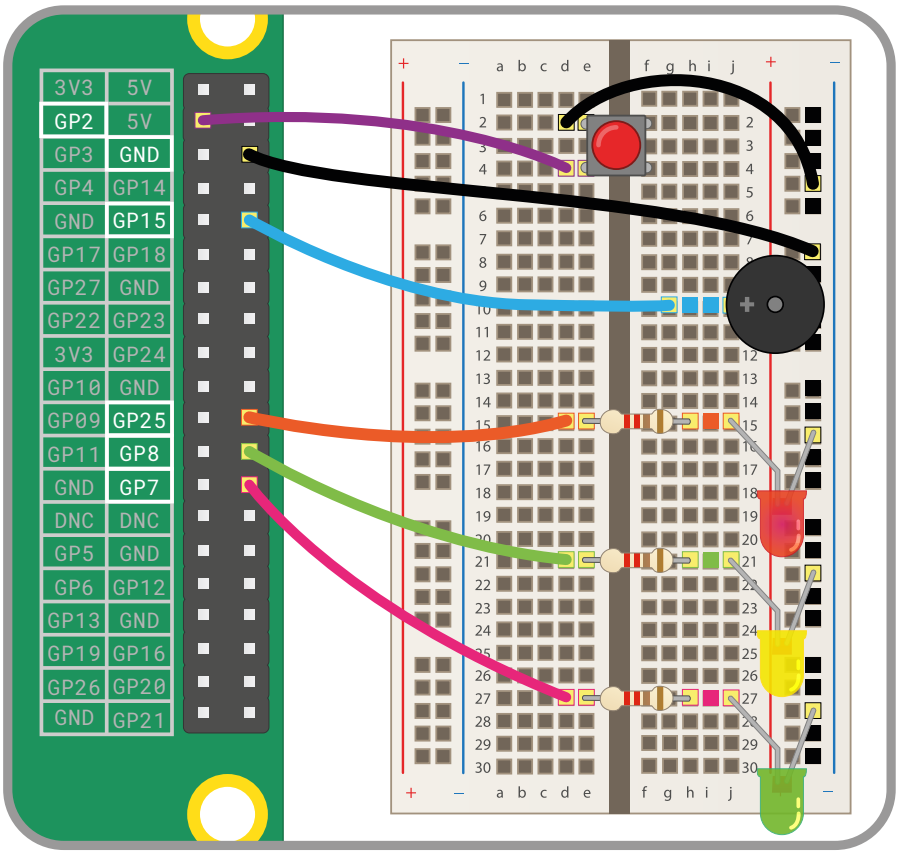
UDFORDRING: EN BEDRE SUMMEN

Hvordan ville du ændre programmet, hvis du ville have buzzeren til at summe i kortere tid ad gangen? Kan du lave et kredsløb til at styre buzzeren med en knap?

Scratch-projekt: Trafiklys

Nu, når du ved, hvordan du bruger knapper, buzzere og lysdioder som in- og output, er du klar til at kaste dig ud i et projekt, der sagtens kunne bruges i den virkelige verden: et trafiklys med en knap, du trykker på, når du skal krydse vejen. Til dette projekt skal vi bruge et breadboard, en rød, en gul og en grøn lysdiode, tre 330 Ω-modstande, en buzzer, en trykknapkontakt samt en håndfuld M2M- (han- til hanstik) og M2F-jumperledninger (han- til hunstik).

Start med at opbygge kredsløbet som vist på (**Figur 6-6**): Forbind buzzeren til GPIO 15-benet (mærket GP15 på **Figur 6-6**), den røde lysdiode til GPIO 25-benet (mærket GP25), den gule lysdiode til GPIO 8 (GP8), den grønne lysdiode til GPIO 7 (GP7) og kontakten til GPIO 2 (GP2). Husk at forbinde 330 Ω-modstandene mellem GPIO-benene og de lange ben på lysdioderne, og tilslut så de andre ben på komponenterne til jordskinnen på breadboardet. Tilslut til sidst jordskinnen til et jordben (mærket GND) på Raspberry Pi for at færdiggøre kredsløbet.

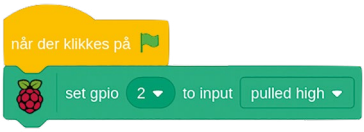


▲ **Figur 6-6:** Elektrisk diagram til trafiklysprojektet

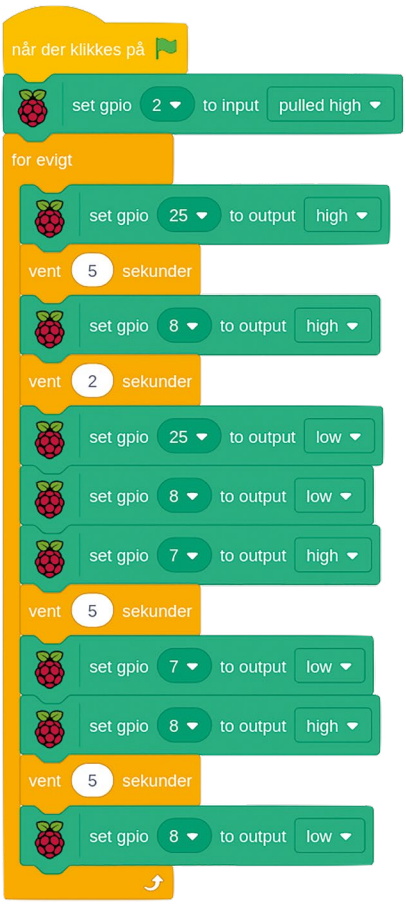
Start et nyt Scratch 3-projekt, og træk en 

-brik til kodeområdet. Nu skal du fortælle Scratch, at GPIO 2-benet, som er tilsluttet trykknapkontakten i kredsløbet, er en

inputkilde og ikke en outputkilde. Dette gøres ved at trække en **set gpio to input pulled high** -brik fra Raspberry Pi GPIO-kategorien i blokpaletten og placere den under **når der klikkes på** -blokken. Klik på pilen ned ud for "0", og vælg 2 i rullemenuen.



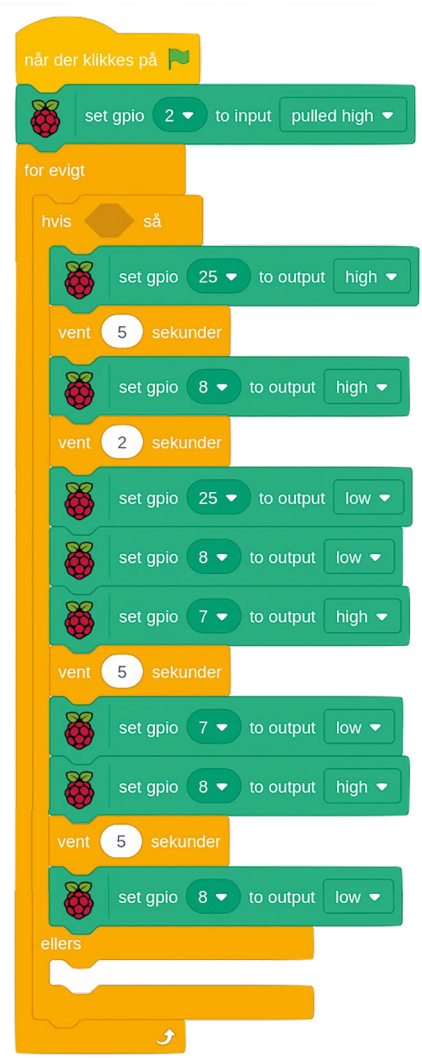
Derefter skal du lave trafiklyssekvensen. Tilføj en **for evigt** -brik til programmet, og fyld den derefter med brikker for at tænde og slukke for dioderne i trafiklyset. Husk, hvilke komponenter der sidder på de forskellige GPIO-ben. Den røde lysdiode er forbundet til ben 25, den gule til ben 8 og den grønne til ben 7.



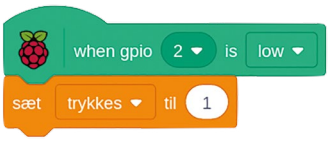
Klik på det grønne flag, og se, hvordan lysdioderne reagerer: Det røde trafiklys tændes, så både det røde og det gule, så det grønne, så det gule, og så gentages sekvensen, der starter

med det røde trafiklys. Dette er det mønster, der bruges i lyssignaler i mange lande, heriblandt i Danmark.

For at simulere trafiksignalerne ved en fodgængerovergang skal du få program til at registrere, at der trykkes på knappen. Klik på den røde ottekant for at stoppe programmet, hvis det kører. Træk en **hvis så** -brik til kodeområdet, og læg den, så den er direkte under **for evigt** -brikken i "hvis så"-afsnittet. Lad den diamantformede åbning være tom for nu.

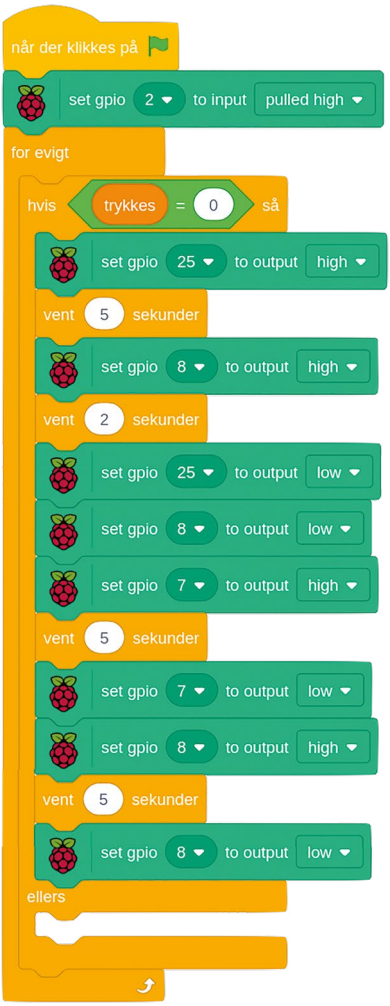


Lyssignalet ved en fodgængerovergang skifter ikke til rødt, så snart der trykkes på knappen. I stedet venter signalet på det næste røde lys i sekvensen. For at bygge dette ind i dit program skal du trække en **when gpio is low** -brik til kodeområdet og vælge "2" i rullemenuen. Læg derefter en **sæt trykket til 1** -brik under den.



Denne briksekvens holder øje med, om der trykkes på knappen, og når dette sker, sættes variablen "trykket" til 1. Ved at sætte en variabel på denne måde kan du gemme den oplysning, at der blev trykket på knappen, til senere.

Gå tilbage til din oprindelige briksekvens, og find hvis så -brikken. Læg en = -operatorbrik i den diamantformede åbning i hvis så -brikken, og læg en trykket -rapportbrik i det første tomme felt. Erstat "50" med "0" i højre side af brikken.



Klik på det grønne flag, og se, hvordan lyssignalet opfører sig nu. Tryk på trykkontakten. Først sker der ikke noget nyt, men når sekvensen når afslutningen – dvs. når den gule lysdiode tændes – slukkes trafiklysene, og de tændes ikke igen, fordi programmet har reageret på variablen "trykket".

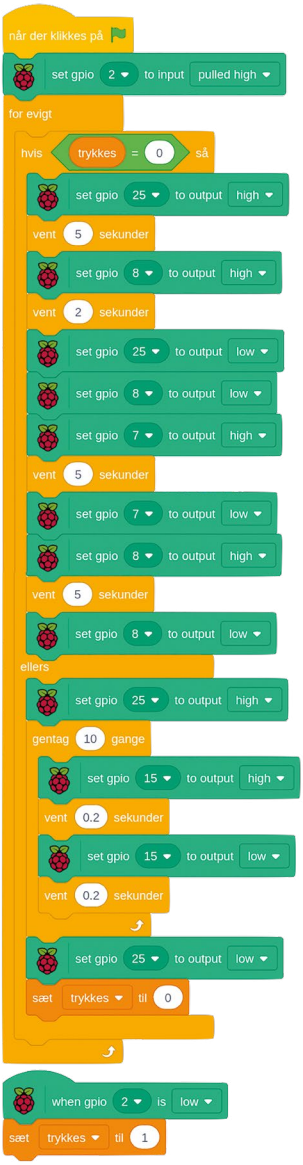
Nu skal du blot programmere fodgængerknappen til at gøre noget mere end blot at slukke for lyssignalet. Find **ellers** -brikken i hovedstakken, og træk en **set gpio 25 to output high** -brik ind i den. Husk at ændre standardværdien for GPIO-benet, så den matcher det ben, den røde lysdiode er forbundet til.

Opret et mønster til buzzeren i **ellers** -brikken ved at tilføje en **gentag 10** -brik og fylde den med en **set gpio 15 to output high** -, en **vent 0.2 sekunder** -, en **set gpio 15 to output low** - og en **vent 0.2 sekunder** -brik. Husk at ændre værdierne for GPIO-ben til at matche det ben, som buzzeren er forbundet til.

Til sidst – direkte under **gentag 10** -brikken og stadig i **ellers** -brikken – skal der lægges en **set gpio 25 to output low** -brik og en **sæt trykket til 0** -brik. Den sidstnævnte brik nulstiller værdien af variablen, der husker, at det blev trykket på knappen, for ellers ville buzzersekvensen blive gentaget igen og igen.

Klik på det grønne flag, og tryk derefter på knappen på breadboardet. Når sekvensen er færdig, tændes det røde lys for bilerne, og buzzeren summer for at lade fodgængere vide, at det er sikkert at krydse vejen. Efter et par sekunder stopper buzzeren, og trafiklyssekvensen starter forfra og fortsætter, indtil du igen trykker på knappen.

Tillykke – du har programmeret dit eget fuldt funktionelle trafiklys, inklusive fodgængerovergang.



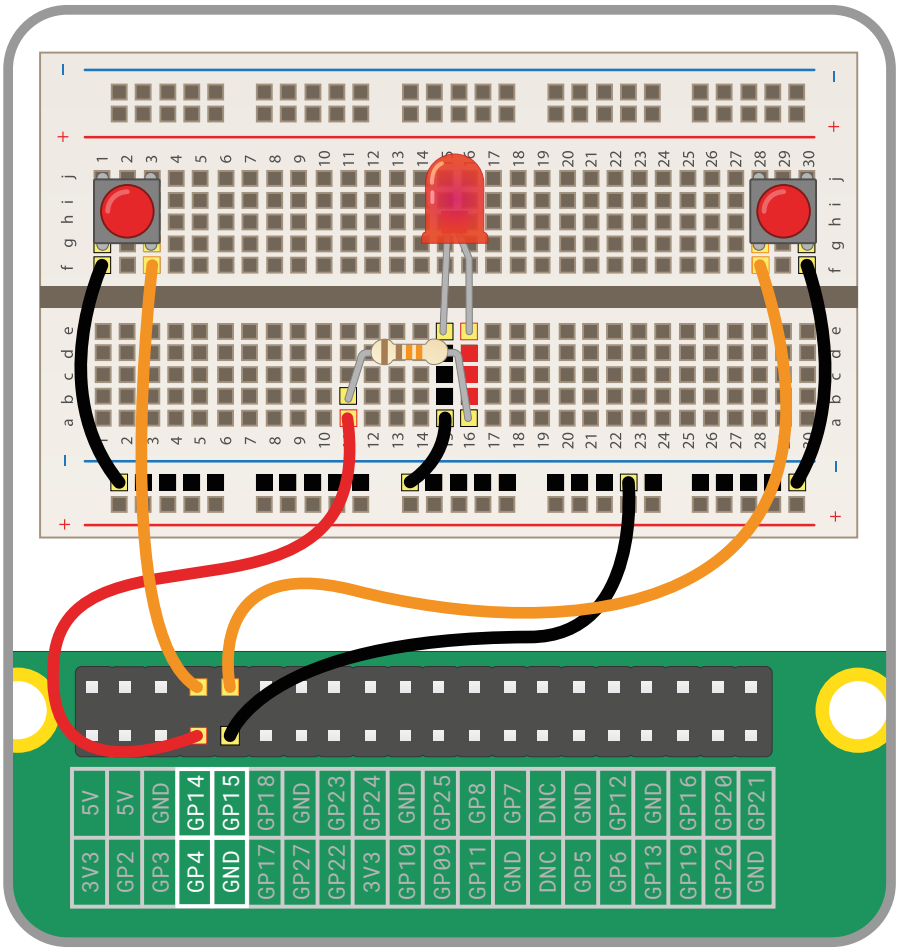
UDFORDRING: KAN DU GØR DET ENDNU BEDRE?

Kan du ændre programmet, så fodgængeren har længere tid at krydse vejen? Kan du finde oplysninger om, hvordan lyssignaler skifter i andre lande, og omprogrammere dit trafiklys? Hvordan ville du reducere lysstyrken i lysdioderne?

Python-projekt: Reflexsspil

Nu, når du ved, hvordan du bruger knapper og lysdioder som in- og output, er du klar til at kaste dig ud i et større projekt: et spil for to deltagere, der tester, hvem der har de hurtigste reflekser. Til dette projekt skal vi bruge et breadboard, en lysdiode, en 330 Ω-modstand, to trykknapkontakter samt nogle M2F- (han- til hunstik) og M2M-jumperledninger (han- til hanstik).

Start med at lave kredsløbet (**figur 6-7**): Monter den første kontakt på venstre side af breadboardet, og forbind den til GPIO 14-benet (mærket GP14 på **figur 6-7**), monter den anden kontakt på højre side af BREADBOARDET, og forbind den til GPIO 15-benet (mærket GP15), forbind det lange ben på lysdioden til 330 Ω-modstanden, og forbind så modstanden til GPIO 4-benet (mærket GP4) på Raspberry Pi. De resterende ben på alle komponenter skal så tilsluttes jordskinne på breadboardet. Fuldfør kredsløbet ved at forbinde jordskinnen til jordbenet (mærket GND) på din Raspberry Pi.



▲ **Figur 6-7:** Elektrisk diagram til reflekssspillet

Opret et nyt projekt i Thonny, og gem det som **Reflekssspil**. Vi skal bruge funktionen **LED** og **button** fra GPIO Zero-biblioteket samt funktionen **sleep** fra time-biblioteket. I stedet for at importere de to GPIO Zero-funktioner ved hjælp af to separate linjer, kan du spare lidt tid ved at sætte dem begge på samme linje med et komma (**,**) imellem. Skriv følgende i scriptområdet:

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

Som før skal du fortælle GPIO Zero, hvilke ben de to knapper og lysdioden er forbundet til. Skriv følgende:

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

Tilføj nu instruktioner, der tænder og slukker for lysdioden, så du kan kontrollere, at den fungerer, som den skal:

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

Klik på knappen Run (Kør). Lysdioden tændes i fem sekunder, derefter slukkes, og programmet afsluttes. Men i et reaktionsspil er det dog lidt forudsigeligt, hvis lysdioden slukker efter nøjagtigt 5 sekunder hver gang. Tilføj følgende under linjen **from time import sleep**:

```
from random import uniform
```

Biblioteket random er opkaldt efter det engelske ord for "tilfældig" og indeholder en funktion til at generere tilfældige tal (her med en ensartet fordeling – se **rpf.io/uniform**). Find linjen **sleep(5)**, og ret den til:

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

Klik på knappen Run (Kør) igen. Denne gang forbliver lysdioden tændt i et tilfældigt antal sekunder mellem 5 og 10. Tæl, i hvor mange sekunder lysdioden er tændt, og klik derefter på knappen Run (Kør) et par gange mere. Som du kan se, er tiden forskellig, hver gang du kører programmet, hvilket gør det hele mindre forudsigeligt.

For at tildele knapperne til hver af de to spillere skal du tilføje en funktion. Skriv følgende helt i bunden af programkoden:

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

Husk, at Python bruger indrykning til at afgøre, hvilke linjer der er en del af funktionen. Thonny indrykker automatisk linje nummer to. Tilføj til sidst følgende to linjer for at registrere, hvornår spillerne trykker på knapperne. Husk, at disse linjer ikke må være indrykket, for ellers vil Python betragte dem som en del af din funktion.

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

Kør programmet, og prøv denne gang at trykke på en af de to kontakter, så snart lysdioden slukker. Når du trykker på en af knapperne, vil du se en meddelelse i Pythons shell-område nederst i Thonny-vinduet. Desværre vises vindermeddelelsen, uanset hvilken knap der trykkes på, og desuden refereres der til numre på GPIO-ben i stedet for navne, som man bedre kunne forholde sig til.

Det sidste kan nemt løses ved at bede spillerne om at indtaste deres navne. Skriv følgende under linjen **`from random import uniform`**:

```
left_name = input("Venstre spiller hedder ")
right_name = input("Højre spiller hedder ")
```

Gå tilbage til din funktion, og erstat linjen **`print(str(button.pin.number) + " har vundet spillet")`** med:

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

Klik på knappen Run (Kør), og skriv derefter navnene på begge spillere i Pythons shell-område. Læg mærke til, at når du trykker på knappen denne gang og husker at gøre det så hurtigt som muligt, efter at lysdioden slukkes, vises navnet på spilleren i stedet for nummeret på GPIO-benet.

Problemet med, at alle knaptryk rapporteres som vindertryk, kan løses ved at tilføje funktionen **`exit`** fra biblioteket sys – hvilket er en forkortelse for *system*. Skriv følgende under den sidste **`import`**-linje:

```
from os import _exit
```

Skriv derefter følgende under linjen **`print(right_name + " har vundet spillet")`**:

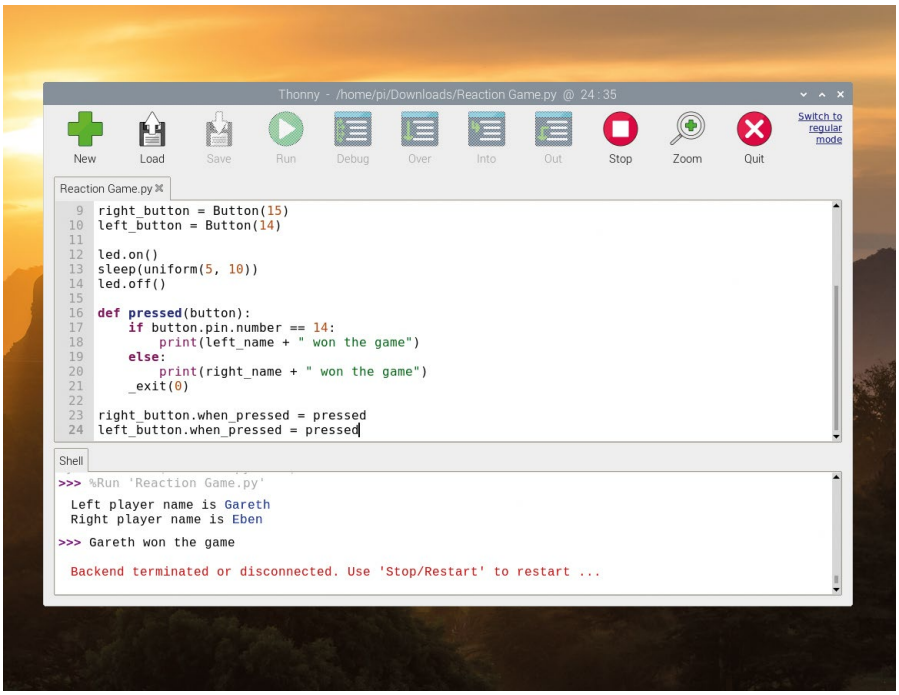
```
    _exit(0)
```

Her er det vigtigt at huske indrykning: **`_exit(0)`** skal være indrykket vha. fire mellemrum, så det flugter med **`else:`**, der befinder sig to linjer over det og **`if`** yderligere to linjer op. Denne instruktion fortæller Python at stoppe programmet, så snart der trykkes på den første knap, så den spiller, der trykker som nummer to, ikke også bliver udnævnt til vinder.

Det færdige program skal se sådan ud:

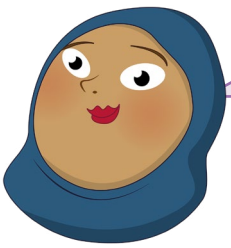
```
from gpiozero import LED, Button
from time import sleep
from random import uniform
from os import _exit

left_name = input("Venstre spiller hedder ")
right_name = input ("Højre spiller hedder ")
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 + " har vundet spillet")
    else:
        print(right_name + " har vundet spillet")
    _exit(0)

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

Klik på knappen Kør, indtast spillernes navne, og programmet vil vise navnet på vinderen, der trykker på knappen som den første, når lysdioden slukkes. Python viser også følgende meddelelse: **`Backend terminated or disconnected . Use 'Stop/Restart' to restart ...`**Denne meddelelse betyder blot, at Python modtog din **`_exit(0)`**-kommando og standsede programmet, men at du stadigvæk skal klikke på Stop-ikonet for at afslutte programmet helt, så det er klart til en ny omgang (**Figur 6-8**).



▲ **Figur 6-8:** Når vinderen er fundet, skal du stoppe programmet

Tillykke: Du har designet dit eget spil med fysisk input!



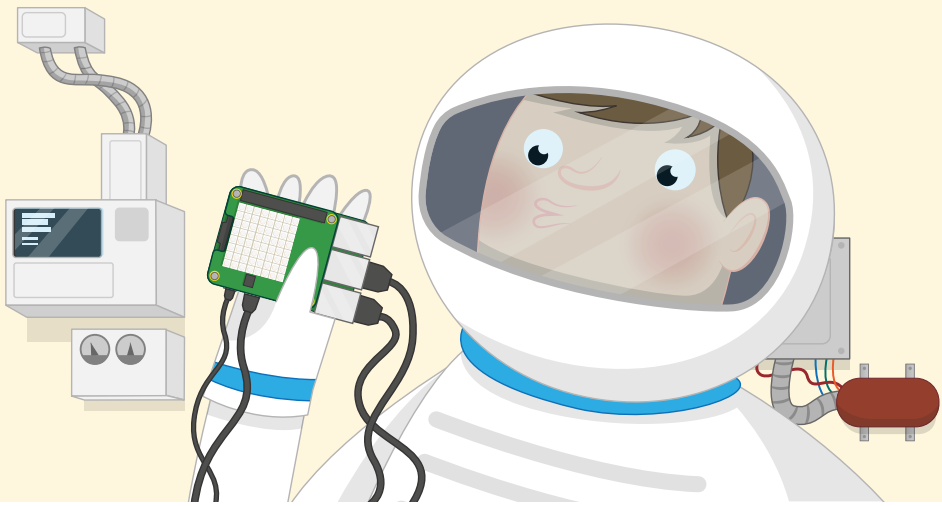
UDFORDRING: GØR SPILLET BEDRE

Kan du tilføje en løkke, så spillet kører uendeligt? Husk at fjerne instruktionen `_exit(0)` først. Kan du tilføje en scoreliste, så du kan se, hvem der vinder over flere omgange? Kan du lave en timer, så du kan se, hvor hurtigt du reagerede, efter lyset blev slukket?

Kapitel 7

Fysisk programmering med Sense HAT

Sense HAT bliver brugt på den internationale rumstation og er et multifunktionelt tilføjelseskort til Raspberry Pi, udstyret med sensorer og LED-matrixvisning



Raspberry Pi leveres med understøttelse af en speciel type tilføjelseskort kaldet *Hardware Attached on Top (HAT)*. HAT'er kan tilføje alt fra mikrofoner og lys til elektroniske relæer og skærme til Raspberry Pi, men især én HAT er meget speciel: Sense HAT.

Sense HAT blev designet specielt til Astro Pi-rummissionen. I et fælles projekt mellem Raspberry Pi Foundation, UK Space Agency og European Space Agency, Astro Pi blev Raspberry Pi-kort og Sense HAT'er fragtet op til den internationale rumstation ombord på en Orbital Science Cygnus-lastraket. Efter at disse Sense HAT'er – kaldet Ed og Izzy af astronauterne – nåede højt op over jorden, er de blevet brugt til at køre kode og udføre videnskabelige eksperimenter med bidrag fra skolebørn over hele Europa.

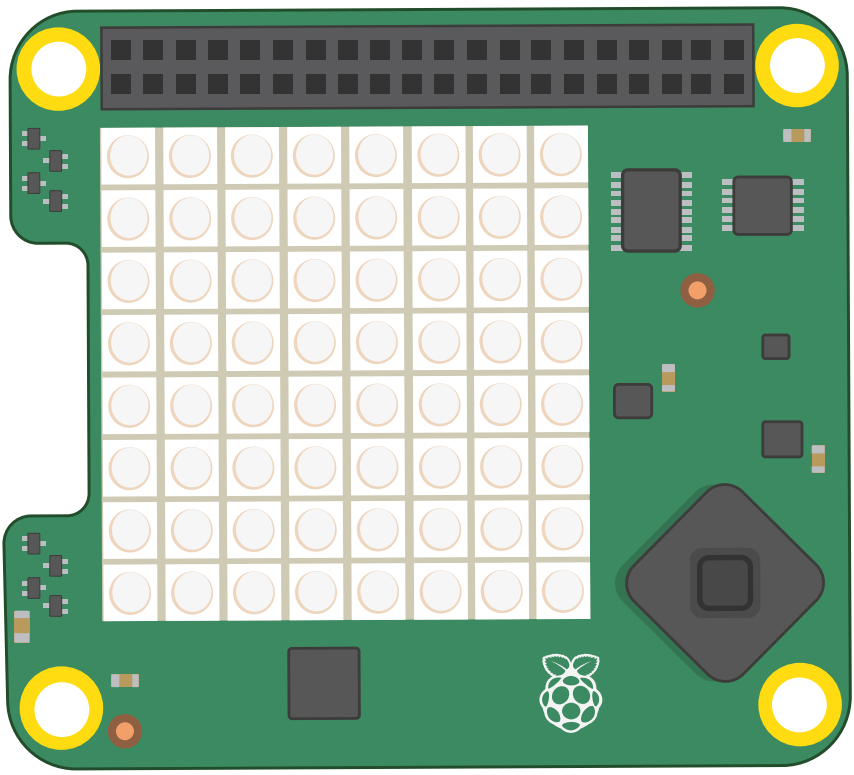
Selvom Ed og Izzy er lidt langt væk til, at du selv kan bruge dem, kan du finde den samme Sense HAT-hardware her på Jorden hos alle Raspberry Pi-forhandlere – og hvis du ikke vil købe en Sense HAT lige nu , kan du simulere en i software!

VIRKELIG ELLER SIMULERET

Du får mest ud af dette kapitel ved at have en ægte Sense HAT knyttet til en Raspberry Pi's GPIO-stik, men hvis du ikke har en, kan du springe over afsnittet med titlen "Installation af Sense HAT" og blot prøve projekterne i Sense HAT Emulator; de fungerer lige så godt!

Introduktion til Sense HAT

Sense HAT er en avanceret, multifunktionel tilføjelse til Raspberry Pi. Ud over en 8 × 8 matrix med 64 røde, grønne og blå (RGB) programmerbare lysdioder, der kan styres til at producere enhver farve blandt millioner, inkluderer Sense HAT en femvejs joystick-controller og seks indbyggede sensorer.



Gyroskopsensor: Bruges til at mærke ændringer i vinklen over tid, teknisk kendt som *vinkelhastighed*, ved at holde styr på retningen af jordens tyngdefelt – den kraft, der trækker ting ned mod centrum af planeten. Enkelt sagt så ved den gyroskopiske sensor, hvornår du roterer Sense HAT i forhold til jordens overflade, og hvor hurtigt den roterer.

Accelerometer: Svarer til gyroskopsensoren, men i stedet for at registrere en vinkel i forhold til jordens tyngdekraft måler den accelerationskraft i flere retninger. Tilsammen kan aflæsninger (data) fra de to sensorer hjælpe dig med at registrere, hvorhen en Sense HAT peger, og hvordan den flyttes.

Magnetometer: Måler styrken af et magnetfelt og er en anden sensor, der kan hjælpe med at registrere Sense HATs bevægelser: Ved at måle Jordens naturlige magnetfelt kan magnetometeret udregne retningen af magnetisk nord. Den samme sensor kan også bruges til at detektere metalgenstande og endda elektriske felter. Alle disse tre sensorer er indbygget i en enkelt chip mærket "ACCEL/GYRO/MAG" på Sense HATs printkort.

Fugtighedssensor: Måler mængden af vanddamp i luften, kendt som den *relative fugtighed*. Relativ fugtighed kan variere fra 0 %, hvis der slet ikke er vand, til 100 %, hvis luften er fuldstændig mættet. Fugtighedsdata kan bruges til at registrere, hvornår det sandsynligvis begynder at regne!

Barometertryksensor: også kendt som *barometeret*, måler lufttryk. Selvom de fleste mennesker nok kender barometertryk fra vejrudsigten, har barometeret en hemmelig anden anvendelse: Det kan registrere, når du klatrer op eller ned ad en bakke eller et bjerg, da luften bliver tyndere og får et lavere tryk, jo længere væk fra Jordens havoverflade du kommer.

Temperaturmåler: Måler, hvor varmt eller koldt det omgivende miljø er, selvom det også påvirkes af, hvor varmt eller koldt Sense HAT er: Hvis du bruger et kabinet, vil aflæsningerne muligvis være højere end forventet. Sense HAT har ikke en separat temperatursensor; i stedet bruger den temperatursensorer indbygget i fugtigheds- og barometertryksensorerne. Et program kan bruge en eller begge af disse sensorer; det er op til dig.

SENSE HAT PÅ RASPBERRY PI 400

Sense HAT er fuldt kompatibel med Raspberry Pi 400 og kan sættes direkte i GPIO-stikket på bagsiden. Dette betyder dog, at lysdioderne vender væk fra dig, og brættet vil være orienteret på hovedet.

For at løse dette skal du bruge et GPIO-forlængerkabel eller -kort. Kompatible udvidelser inkluderer Black HAT Hack3r-serien fra Pimoroni; du kan bruge Sense HAT med selve Black HAT Hack3r-kortet eller blot bruge det medfølgende 40-benede båndkabel som udvidelse. Kontroller dog altid producentens instruktioner for at være sikker på, at du tilslutter kablet og Sense HAT den rigtige vej rundt!

Installation af Sense HAT

Hvis du har en fysisk Sense HAT, skal du starte med at pakke den ud og sørge for, at du har alle delene: Du skal have selve Sense HAT, fire søjler af metal eller plast kendt som *afstandsstykker* og otte skruer. Du kan også have nogle metalstifter i en sort plaststrimmel, som GPIO-stifterne på Raspberry Pi; i så fald skal du skubbe denne strimmel med stifterne opad gennem bunden af Sense HAT, indtil du hører et klik.

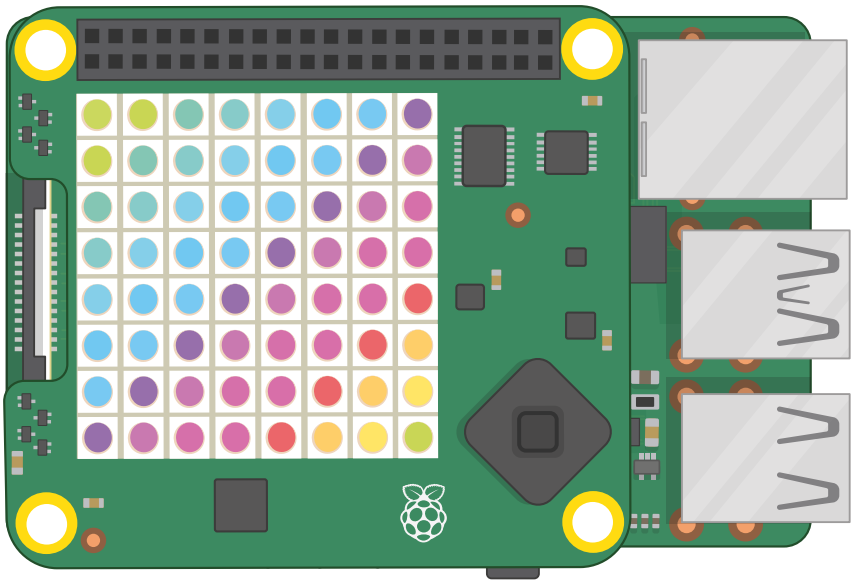
Afstandsstykkerne er designet til at forhindre, at Sense HAT bøjes og spændes, når du bruger joysticket. Sense HAT kan godt fungere, uden at de er installeret, men de hjælper med at beskytte dit Sense HAT, Raspberry Pi og GPIO- stik mod at blive beskadiget.

ADVARSEL!

Hardware i form af HAT-moduler (Hardware Attached on Top), må kun tilsluttes og fjernes fra GPIO-stikket, mens din Raspberry Pi er slukket og afbrudt fra strømforsyningen. Vær altid forsigtig med at holde din HAT flad, når du installerer den, og dobbelttjek, at den er justeret i forhold til GPIO-stikkets ben, inden du skubber den ned.

Installer afstandsstykkerne ved at skubbe fire af skruerne op nedefra bunden af Raspberry Pi gennem de fire monteringshuller i hvert hjørne, og drej derefter afstandsstykkerne ind på skruerne. Skub Sense HAT ned på Raspberry Pi's GPIO-stik, og sørg for at justere den korrekt i forhold til stifterne nedenunder og holde den så flad som muligt. Til sidst skrues de sidste fire skruer gennem monteringshullerne på Sense HAT og ind i afstandsstykkerne, som du tidligere har installeret. Hvis den er installeret korrekt, skal Sense HAT være flad og plan og bør ikke bøje eller vippe, når du trykker på joysticket.

Slut strømmen til din Raspberry Pi, hvorefter du vil se lysdioderne på Sense HAT lyse op i et regnbuemønster (**Figur 7-1**), og sluk derefter igen. Din Sense HAT er nu installeret!

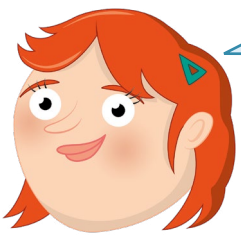


▲ **Figur 7-1:** Et regnbuemønster vises, når strømmen tilsluttes første gang

Hvis du vil fjerne Sense HAT igen, skal du bare løsne de øverste skruer, løfte HAT'en af – pas på ikke at bøje stifterne på GPIO-stikket, da HAT'en sidder godt fast (det kan være nødvendigt at lirke den af med en lille skruetrækker) – og fjern derefter afstandsstykkerne fra Raspberry Pi.

Hej, Sense HAT!

Som med alle programmeringsprojekter er der et naturligt sted at starte med Sense HAT: At rulle en velkomstbesked over dens LED-skærm. Hvis du bruger Sense HAT-emulatoren, skal du indlæse den nu ved at klikke på Raspberry Pi OS-menuikonet, vælge programmeringskategorien og klikke på Sense HAT-emulator.

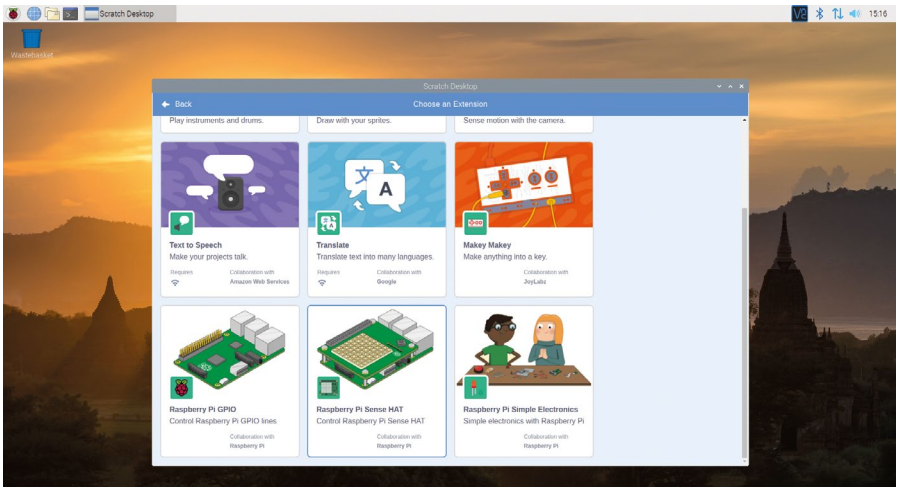


PROGRAMMERINGSOPLEVELSEN

Dette kapitel forudsætter erfaring med Scratch 3 eller Python og det Thonny-integrerede udviklingsmiljø (IDE), afhængigt af om du arbejder med Scratch- eller Python-kodeeksemplerne – eller dem begge! Hvis du ikke allerede har gjort det, skal du se **Kapitel 4, Programmering med Scratch** eller **Kapitel 5, Programmering med Python** og først gennemgå projekterne i det pågældende kapitel.

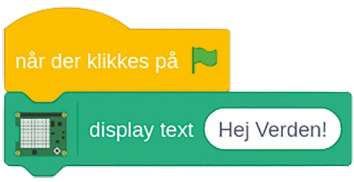
Hilsener fra Scratch

Indlæs Scratch 3 fra startmenuen i Raspberry Pi OS. Klik på knappen Add Extension (Tilføj udvidelse) nederst til venstre i Scratch-vinduet. Klik på Raspberry Pi Sense HAT-udvidelsen (**Figur 7-2**). Dette indlæser de brikker, du har brug for for at kontrollere de forskellige funktioner i Sense HAT, inklusive dens LED-skærm. Når du har brug for dem, finder du dem i Raspberry Pi Sense HAT-kategorien.

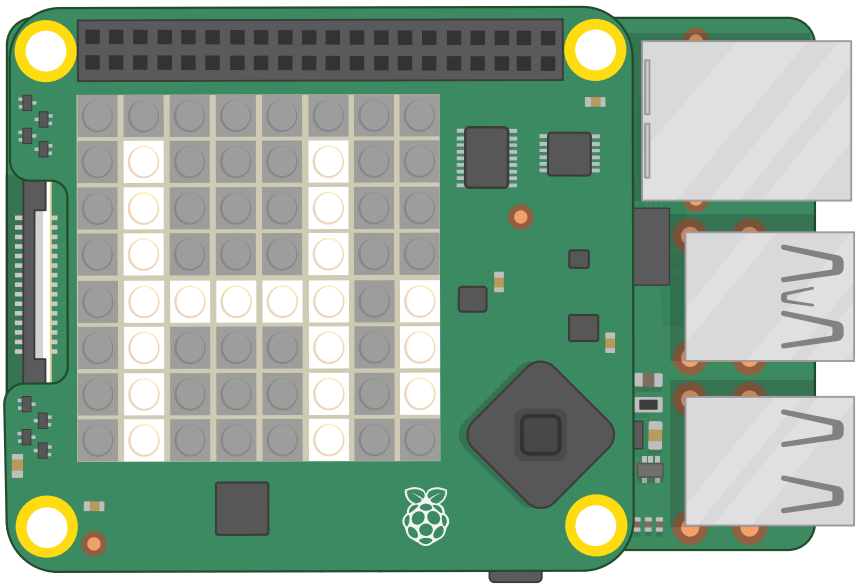


▲ **Figur 7-2:** Tilføjelse af Raspberry Pi Sense HAT-udvidelse til Scratch 3

Start med at trække en når der klikkes på hændelsesbrik til scriptområdet, og træk derefter en display text Hello! brik direkte under den. Rediger teksten, så der på brikken står display text Hej verden!.

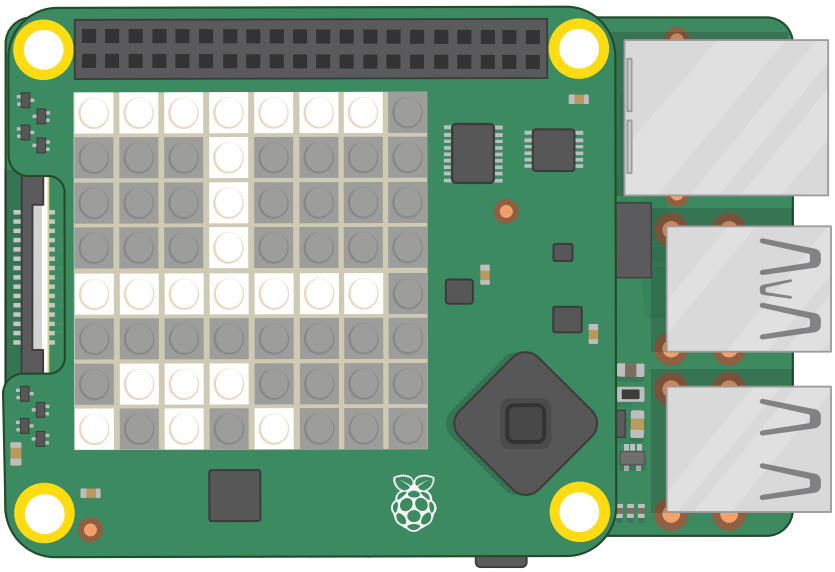


Klik på det grønne flag på scenen, og se din Sense HAT eller Sense HAT-emulator: Meddelelsen vil rulle langsomt hen over Sense HATs LED-matrix og oplyse LED-pixlerne for at danne hvert bogstav efter hinanden (**Figur 7-3**, på næste side). Tillykke: Programmet er udført!



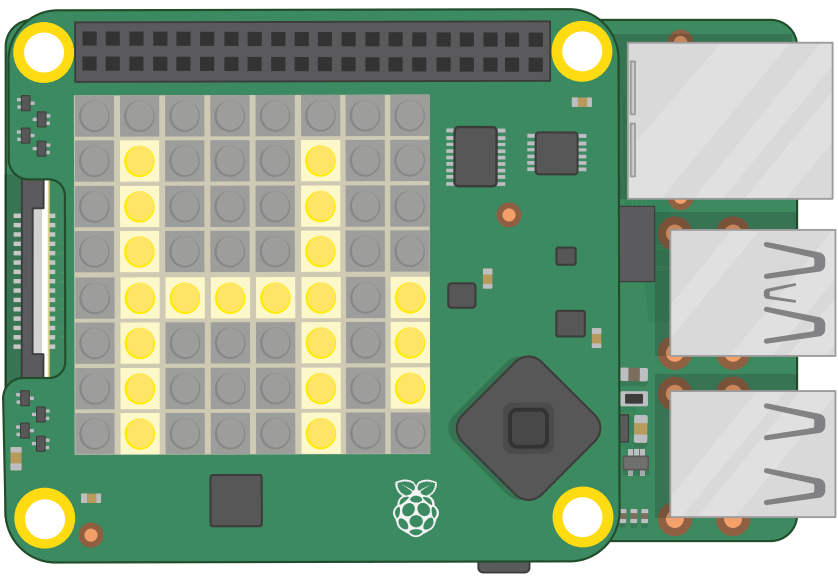
Figur 7-3: Din besked ruller over LED-matrixen

Nu kan du rulle en simpel besked, og så er det tid til at se på, hvordan du styrer, hvordan meddelelsen vises. Udover at kunne ændre den besked, der skal vises, kan du også ændre rotation – dvs. hvilken vej meddelelsen vises på Sense HAT. Træk en `set rotation to 0 degrees` brik fra brikpaletten og indsæt den under `når der klikkes på` og over `display text Hej verden!`, og klik derefter på pil ned ud for 0, og skift den til 90. Klik på det grønne flag, så får du vist den samme besked som før, men i stedet for at rulle fra venstre mod højre ruller den fra bund til top (**Figur 7-4**) – du skal dreje dit hoved eller Sense HAT, for at kunne læse det!



▲ **Figur 7-4:** Denne gang ruller beskeden lodret

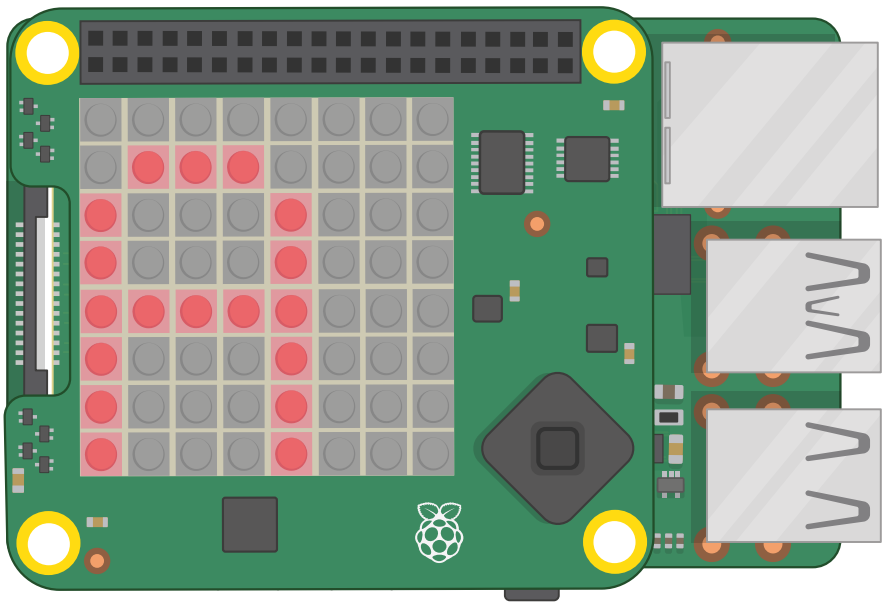
Skift nu rotationen tilbage til 0, og træk derefter en `set colour` brik mellem `set rotation to 0 degrees` og `display text Hej verden!`. Klik på farven i slutningen af brikken for at få Scratchs farvevælger op og finde en dejlig lysegul farve, og klik derefter på det grønne flag for at se, hvordan dit programs output har ændret sig (**Figur 7-5**).



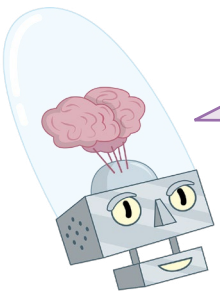
▲ **Figur 7-5:** Ændring af tekstens farve

Træk til sidst en set background brik mellem set colour to gul og display text Hej verden! , og klik derefter på farven for at få vist farvevælgeren igen. Denne gang påvirker valg af farve ikke de lysdioder, der udgør meddelelsen, men de lysdioder, der ikke gør det – kendt som baggrunden. Find en dejlig blå farve, og klik derefter på det grønne flag igen: Denne gang vil din besked være i lysegul på en blå baggrund. Prøv at ændre disse farver for at finde din yndlingskombination – ikke alle farver fungerer lige godt sammen!

Ud over at kunne rulle hele meddelelser kan du vise individuelle bogstaver. Træk din display text brik fra scriptområdet for at slette den, og træk derefter en display character A brik til scriptområdet i stedet for. Klik på det grønne flag, så ser du forskellen: Denne brik viser kun ét bogstav ad gangen, og bogstavet forbliver på Sense HAT uden at rulle eller forsvinde, indtil du vælger at det skal gøre noget andet. De samme farvekontrolbrikker gælder for denne brik som display text brikken: Prøv at ændre bogstavets farve til rød (**Figur 7-6**).



▲ **Figur 7-6:** Viser et enkelt bogstav



UDFORDRING: GENTAG MEDDELELSEN

Kan du bruge din viden om løkker til at få en rullebesked til at gentage sig selv? Kan du lave et program, der staver et ord bogstav for bogstav i forskellige farver?

Hilsener fra Python

Indlæs Thonny ved at klikke på hindbærmenuikonet, vælge Programming (Programmering) og klikke på Thonny. Hvis du bruger Sense HAT-emulatoren, og den bliver dækket af Thonny-vinduet, skal du klikke og holde museknappen nede på et af vinduets titellinjer - øverst i blåt - og trække det for at flytte det rundt på skrivebordet, indtil du kan se begge vinduer.

ÆNDRING AF LINJE I PYTHON

Python-kode skrevet til en fysisk Sense HAT kan køre på Sense HAT-emulatoren og vice versa med kun en enkelt ændring. Hvis du bruger Sense HAT-emulatoren med Python, skal du ændre linjen fra `sense_hat import SenseHat` i alle programmerne fra dette kapitel til `from sense_emu import SenseHat` i stedet. Hvis du vil køre dem på en fysisk Sense HAT igen, skal du bare ændre linjen tilbage!

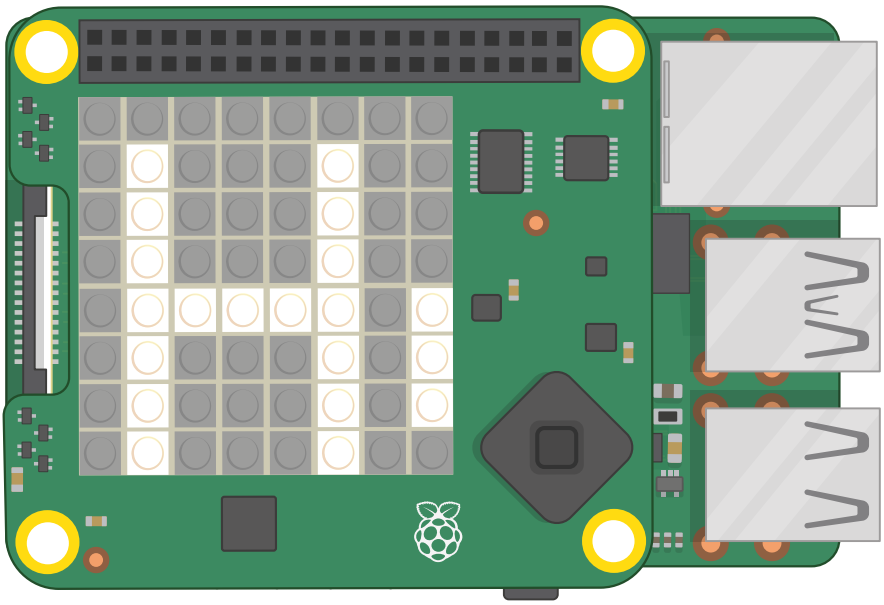
For at bruge Sense HAT eller Sense HAT-emulatoren i et Python-program skal du importere Sense HAT-biblioteket. Skriv følgende i scriptområdet, og husk at bruge **`sense_emu`** (i stedet for **`sense_hat`**), hvis du bruger Sense HAT-emulatoren:

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

Sense HAT-biblioteket har en enkel funktion til at tage en besked, formatere den, så den kan vises på LED-skærmen og rulle den jævnt. Skriv følgende:

```
sense.show_message("Hej verden!")
```

Gem dit program som **Hej Sense HAT**, og klik derefter på knappen Run (Kør). Du vil se din besked rulle langsomt hen over Sense HATs LED-matrix og oplyse LED-pixels for at danne hvert bogstav efter hinanden (**Figur 7-7**, på næste side). Tillykke: Programmet er udført!



Figur 7-7: Rulning af en besked over LED-matrixen

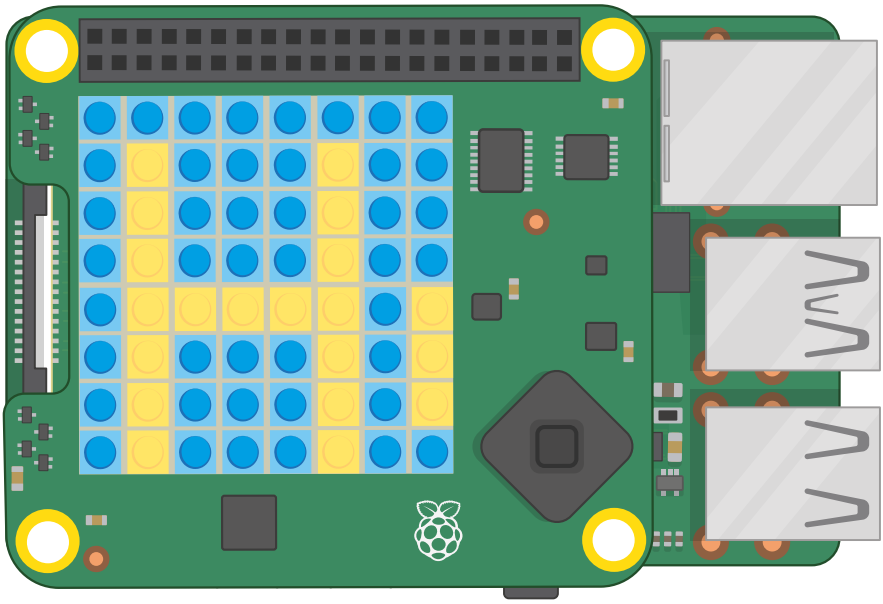
`show_message()`-funktionen har dog flere tricks i ærmet end det. Gå tilbage til dit program, og rediger den sidste linje, så der står:

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

Disse ekstra instruktioner, adskilt af kommaer, er kendt som *parametre*, og de styrer forskellige aspekter af `show_message()`-funktionen. Den enkleste er `scroll_speed=()`, som ændrer, hvor hurtigt beskeden ruller over skærmen. En værdi på 0.05 her ruller med omtrent det dobbelte af den sædvanlige hastighed. Jo større tal, jo lavere hastighed.

Parameterne `text_colour=()` og `back_colour=()` – stavet på britisk engelsk måde, i modsætning til de fleste Python-instruktioner – indstiller henholdsvis skriftfarven og baggrundsfarven. De accepterer dog ikke farvenavne; du skal angive den farve, du ønsker, som en trio af tal. Det første tal repræsenterer mængden af rød i farven, fra 0 for slet ingen rød til 255 for så meget rød som muligt; det andet tal er mængden af grønt i farven; og det tredje tal mængden af blå. Tilsammen er disse kendt som *RGB* – for rød, grøn og blå.

Klik på ikonet Run (Kør) og se Sense HAT: Denne gang vil meddelelsen rulle betydeligt hurtigere og være i lysegul på en blå baggrund (**Figur 7-8**). Prøv at ændre parametrene for at finde en hastighed og farvekombination, der fungerer for dig.



Figur 7-8: Ændring af farven på meddelelsen og baggrunden

Hvis du vil anvende brugrvenlige navne i stedet for RGB-værdier til at indstille dine farver, skal du oprette variabler. Over linjen `sense.show_message()` skal du tilføje følgende:

```
gul = (255, 255, 0)
blå = (0, 0, 255)
```

Gå tilbage til linjen `sense.show_message()` og rediger den, så den lyder:

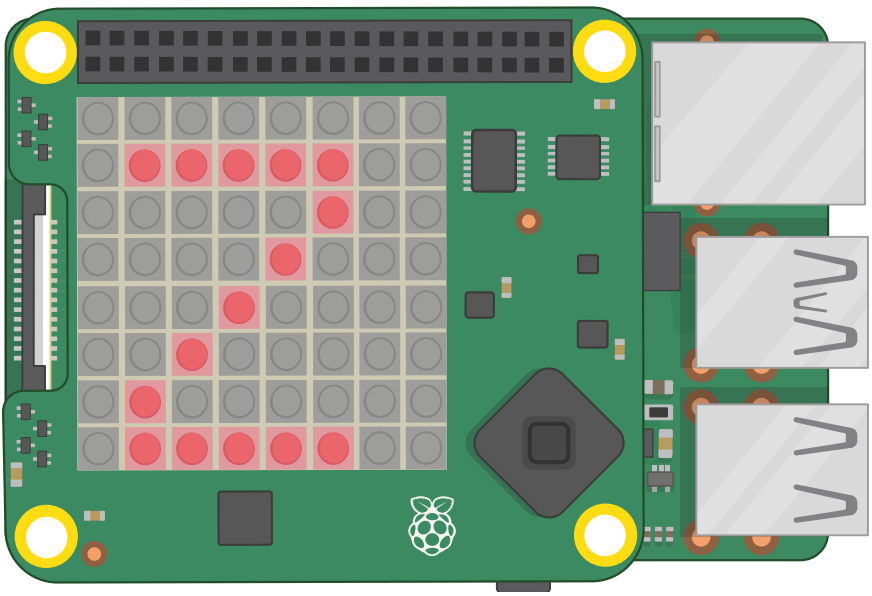
```
sense.show_message("Hej verden!", text_colour=(gul), back_
colour=(blå), scroll_speed=(0.05))
```

Klik på ikonet Run (Kør) igen, og du vil se, at intet er ændret: Din besked er stadig i gul på en blå baggrund. Denne gang har du dog brugt variabelnavne til at gøre din kode mere læselig: I stedet for en streng med tal fortæller koden, hvilken farve den indstiller. Du kan definere lige så mange farver, som du vil: Prøv at tilføje en variabel kaldet "rød" med værdierne 255, 0 og 0; en variabel kaldet "hvid" med værdierne 255, 255, 255; og en variabel kaldet "sort" med værdierne 0, 0 og 0.

Ud over at kunne rulle hele meddelelser kan du vise individuelle bogstaver. Slet linjen `sense.show_message()` helt, og skriv følgende i stedet for:

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

Klik på Run (Kør), så vises bogstavet 'Z' på Sense HATs display. Denne gang forbliver det der: Individuelle bogstaver, i modsætning til meddelelser, ruller ikke automatisk. Du kan også styre `sense.show_letter()` med de samme farveparametre som `sense.show_message()`: Prøv at ændre bogstavets farve til rød (**Figur 7-9**).



▲ **Figur 7-9:** Viser et enkelt bogstav

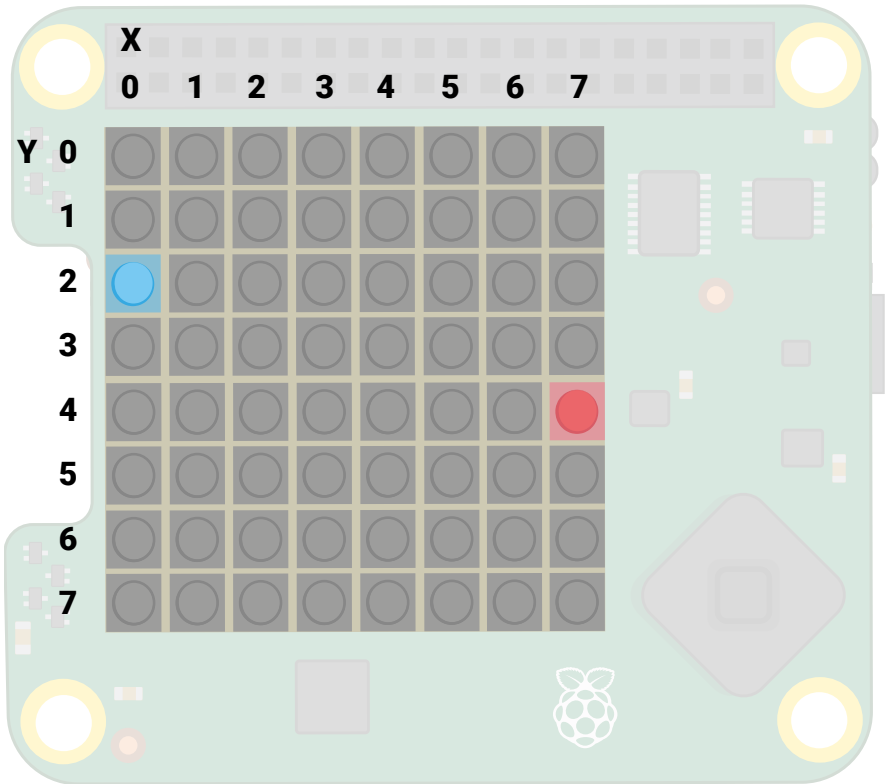
UDFORDRING: GENTAG MEDDELELSEN

Kan du bruge din viden om løkker til at få en rullebesked til at gentage sig selv? Kan du lave et program, der staver et ord bogstav for bogstav i forskellige farver? Hvor hurtigt kan du få en besked til at rulle?

Næste skridt: Tegning med lys

Sense HATs LED-skærm er ikke kun til meddelelser: Du kan også vise billeder. Hver LED kan behandles som en enkelt pixel – kort for *billedelement* – i et billede, som du selv vælger, så du kan pifte dine programmer op med billeder og endda animation.

For at oprette tegninger skal du dog kunne ændre individuelle lysdioder. For at kunne gøre det skal du forstå, hvordan Sense HATs LED-matrix er organiseret for at kunne skrive et program, der tænder eller slukker for de rigtige LED'er.



▲ **Figur 7-10:** Koordinatsystem over LED-matrix

Der er otte lysdioder i hver række på skærmen og otte i hver kolonne (**Figur 7-10**). Når du tæller lysdioderne, skal du – som i de fleste andre programmeringssprog – starte ved 0 og slutte ved 7. Den første LED er i øverste venstre hjørne, den sidste er nederst til højre. Ved hjælp af numrene fra rækkerne og kolonnerne kan du finde *koordinaterne* til enhver LED på matrixen. Den blå LED i den afbildede matrix er ved koordinaterne 0, 2; den røde LED er ved koordinaterne 7, 4. X-aksen på tværs af matrixen kommer først, efterfulgt af Y-aksen ned ad matrixen.

Når du planlægger billeder, der skal tegnes på Sense HAT, kan det hjælpe at tegne dem manuelt først på gitterpapir, eller du kan planlægge tingene i et regneark som LibreOffice Calc.

Billeder i Scratch

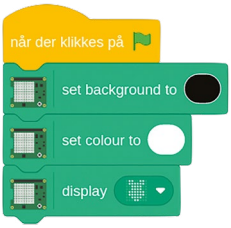
Start et nyt projekt i Scratch, og gem dit eksisterende projekt, hvis du vil beholde det. Hvis du har arbejdet dig gennem projekterne i dette kapitel, vil Scratch 3 stadig have Raspberry Pi Sense HAT-udvidelsen indlæst; hvis du har lukket og genåbnet Scratch 3 siden dit sidste projekt, skal du indlæse udvidelsen igen ved hjælp af knappen Add Extension (Tilføj udvidelse). Træk en 

hændelsesbrik til kodeområdet, og træk derefter brikkerne 

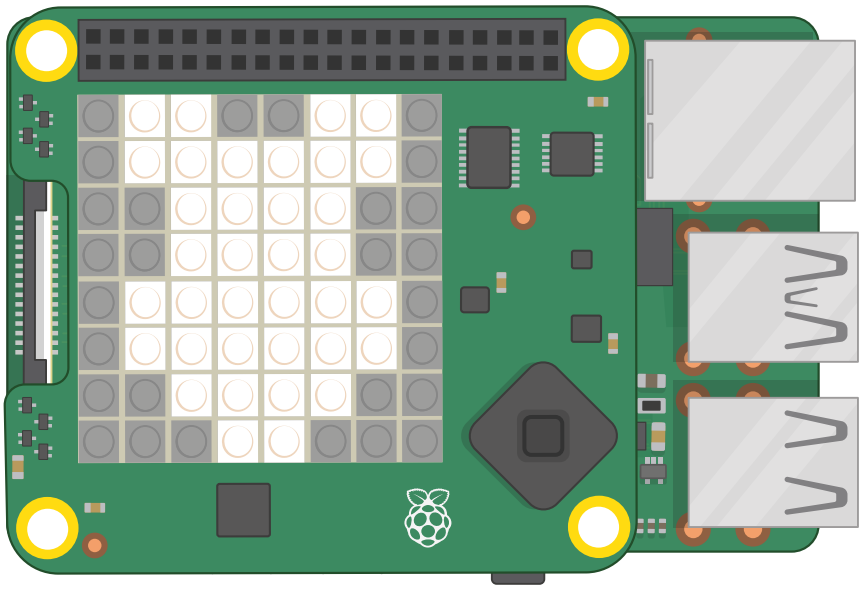
og 

under den. Rediger begge for at indstille baggrundsfarven til

sort og farven til hvid: Gør farven sort ved at skubbe skyderne Brightness (Lysstyrke) og Saturation (Mætning) til 0; gør farven hvid ved at skubbe Brightness (Lysstyrke) til 100 og Saturation (Mætning) til 0. Du skal gøre dette i starten af hvert Sense HAT-program, ellers bruger Scratch simpelthen de sidste farver, du valgte – også selvom du valgte dem i et andet program. Til sidst skal du trække en **display hindbær** brik til bunden af dit program.



Klik på det grønne flag: Du vil se Sense HATs lysdioder oplys et hindbær (**Figur 7-11**).

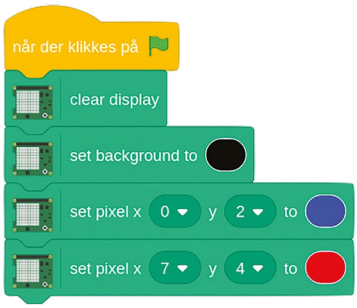


▲ **Figur 7-11:** Se ikke direkte på lysdioderne, når de lyser kraftigt hvidt

Du er heller ikke begrænset til den forudindstillede hindbærform. Klik på pil ned ved siden af hindbæret for at aktivere tegnetilstand: Du kan klikke på en hvilken som helst LED i mønsteret for at tænde eller slukke for det, mens de to knapper i bunden indstiller alle lysdioder til slukket eller tændt. Prøv at tegne dit eget mønster nu, og klik derefter på den grønne pil for at se det på Sense HAT. Prøv også at ændre farven og baggrundsfarven ved hjælp af brikkerne ovenfor.

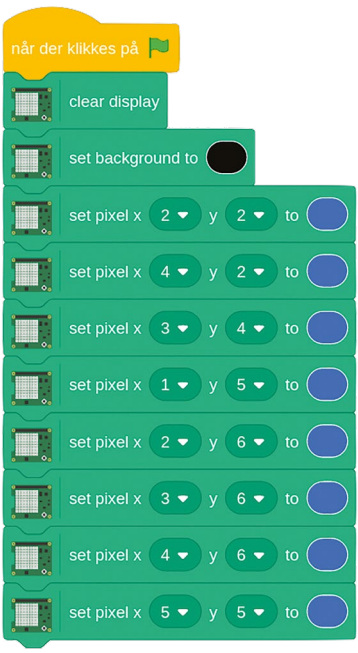
Når du er færdig, skal du trække de tre brikker ind i brikpaletten for at slette dem og placere en **clear display** brik under **når der klikkes på** ; klik på det grønne flag, og alle lysdioder slukkes.

For at lave et billede skal du kunne kontrollere individuelle pixels og give dem forskellige farver. Du kan gøre dette ved at kæde redigerede **display hindbær** brikker med **set colour** brikker, eller du kan adressere hver pixel individuelt. For at oprette din egen version af LED-matrixeksemplet, der er afbildet i starten af dette afsnit, med to specifikt udvalgte lysdioder, der lyser rødt og blåt, skal du beholde **clear display** brikken øverst i dit program og trække en **set background** brik under den. Skift **set background** brikken til sort, og træk derefter to **set pixel x 0 y 0** brikker under den. Til sidst skal du redigere disse brikker som følger:

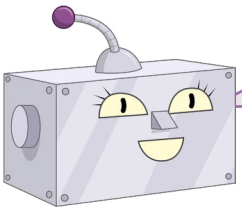


Klik på det grønne flag, og du vil se dine lysdioder lyse op, så de matcher matrixbilledet (**Figur 7-10**) på side 165. Tillykke: Du kan styre individuelle lysdioder!

Rediger dine eksisterende sæt pixelbrikker som følger, og træk flere til bunden, indtil du har følgende program:



Inden du klikker på det grønne flag, skal du se, om du kan gætte, hvilket billede der vises, baseret på de LED-matrixkoordinater, du har brugt. Kør derefter dit program og se om du har ret!



UDFORDRING: NYE DESIGN

Kan du designe flere billeder? Prøv at få fat på noget graf- eller gitterpapir, og brug det til først at planlægge dit billede manuelt. Kan du tegne et billede og få farverne til at ændre sig?

Billeder i Python

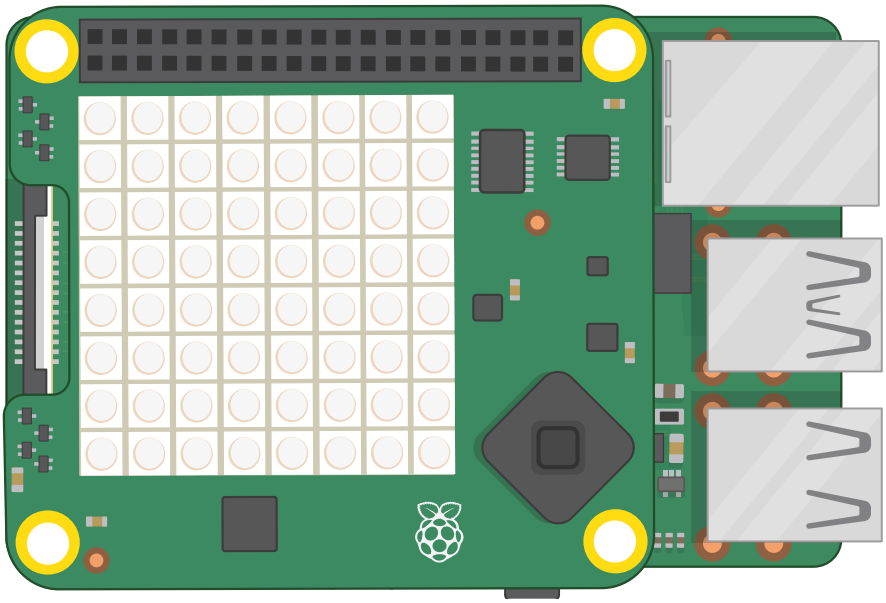
Start et nyt program i Thonny, og gem det som Sense HAT-tegning, og skriv derefter følgende – husk at bruge **`sense_emu`** (i stedet for **`sense_hat`**), hvis du bruger emulatoren:

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

Husk at du skal have begge disse linjer i dit program for at kunne bruge Sense HAT. Skriv derefter:

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

Lad være med at se direkte på Sense HATs LED'er, og kli på ikonet Run (Kør): Du vil se, at de alle lyser hvidt (**Figur 7-12**) – derfor skal du ikke se direkte på dem, når du kører dit program!

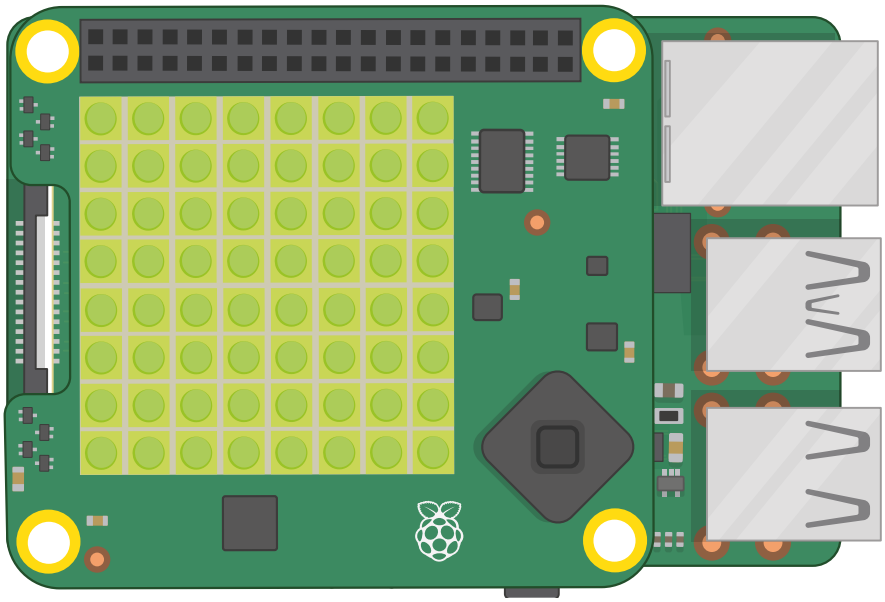


▲ **Figur 7-12:** Se ikke direkte på matrixen, når den lyser kraftigt hvidt

`sense.clear()` er designet til at rydde lysdioderne fra enhver tidligere programmering, men accepterer RGB-farveparametre – hvilket betyder, at du kan ændre skærmen til en hvilken som helst farve, som du kan lide. Prøv at redigere linjen til:

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

Klik på Run (Kør), og Sense HAT bliver lysegrøn (**Figur 7-13**, på næste side). Eksperimenter med forskellige farver, eller tilføj de farvenavnvariabler, som du oprettede til dit Hej verden-program for at gøre tingene nemmere at læse.

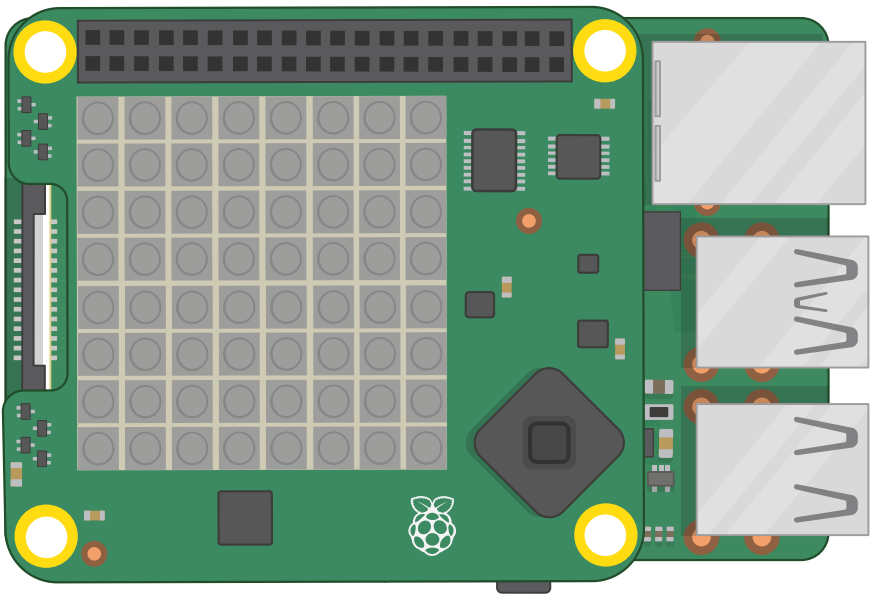


▲ **Figur 7-13:** LED-matrixen lyser op i lysegrøn

For at rydde lysdioderne skal du bruge RGB-værdierne til sort: 0 rød, 0 blå og 0 grøn. Der er dog en lettere måde. Rediger linjen i dit program, så der står:

```
sense.clear()
```

Sense HAT bliver mørkt; for **`sense.clear()`**-funktionen, uden noget mellem parenteserne, svarer til at bede om at gøre alle lysdioder sorte – dvs. slukke dem (**Figur 7-14**). Når du har brug for at slukke alle lysdioderne i dine programmer, er det den funktion, du skal bruge.



Figur 7-14: Brug `sense.clear`-funktionen til at slukke for alle lysdioder

For at oprette din egen version af LED-matrixen, der er afbildet tidligere i dette kapitel, med to specifikt udvalgte LED'er, der lyser rødt og blåt, skal du tilføje følgende linjer til dit program efter **`sense.clear()`** :

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

Det første par tal er pixelens placering på matrixen, X-aksen (tværs) akse efterfulgt af Y-aksen (ned). Den anden, i deres egne parenteser, er RGB-værdierne for pixelfarve. Klik på knappen Run (Kør), og du vil se effekten: To lysdioder på din Sense HAT lyser, ligesom i **Figur 7-10** på side 165.

Slet disse to linjer, og skriv 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))
```

Inden du klikker på Run (Kør), skal du se på koordinaterne og sammenligne dem med matrixen: Kan du gætte hvilket billede disse instruktioner vil tegne? Klik på Run (Kør) for at se, om du har ret!

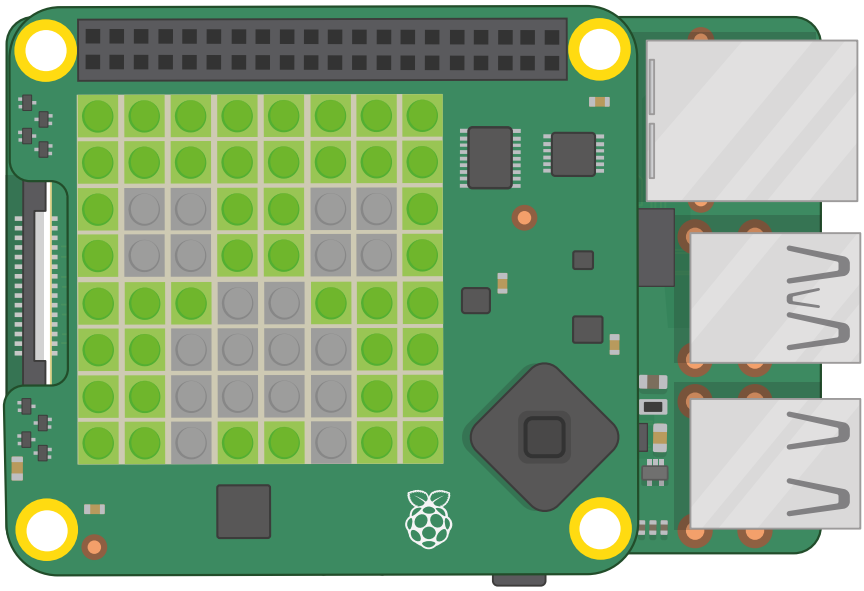
Det er dog langsomt at tegne et detaljeret billede ved hjælp af individuelle **`set_pixel()`**-funktioner. For at fremskynde tingene kan du ændre flere pixels på samme tid. Slet alle dine **`set_pixel()`**-linjer, og skriv følgende:

```
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)
```

Der er meget der, men start med at klikke på Run (Kør) for at se, om du genkender et bestemt lille element. De første to linjer opretter to variabler til at holde farver: Grøn og sort. For at gøre tegningskoden lettere at skrive og læse, er variablerne enkelt bogstaver: "g" for grøn og "b" for sort.

Den næste kodeblok opretter en variabel, der indeholder farveværdier for alle 64 pixels på LED-matrixen, adskilt af kommaer og mellem firkantede parenteser. I stedet for tal bruger den dog de farvevariabler, du oprettede tidligere: se nøje, husk at "g" er for grønt og "b" er for sort, og du kan allerede se det billede, der vises (**Figur 7-15**).

Endelig tager **`sense.set_pixels(creeper_pixels)`** den variabel og bruger **`sense.set_pixels()`**-funktionen til at tegne på hele matrixen på én gang. Meget lettere end at prøve at tegne pixel-for-pixel!



Figur 7-15: Visning af et billede på matrixen

Du kan også rotere og vende billeder, enten som en måde at vise billeder den rigtige vej op, når din Sense HAT drejes rundt, eller som en måde at skabe enkle animationer ud fra et enkelt asymmetrisk billede.

Start med at redigere din **`creeper_pixels`**-variabel for at lukke hans venstre øje ved at udskifte de fire "s"-pixels, startende med de første to på tredje linje og derefter de første to på den fjerde linje, med "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
]
```

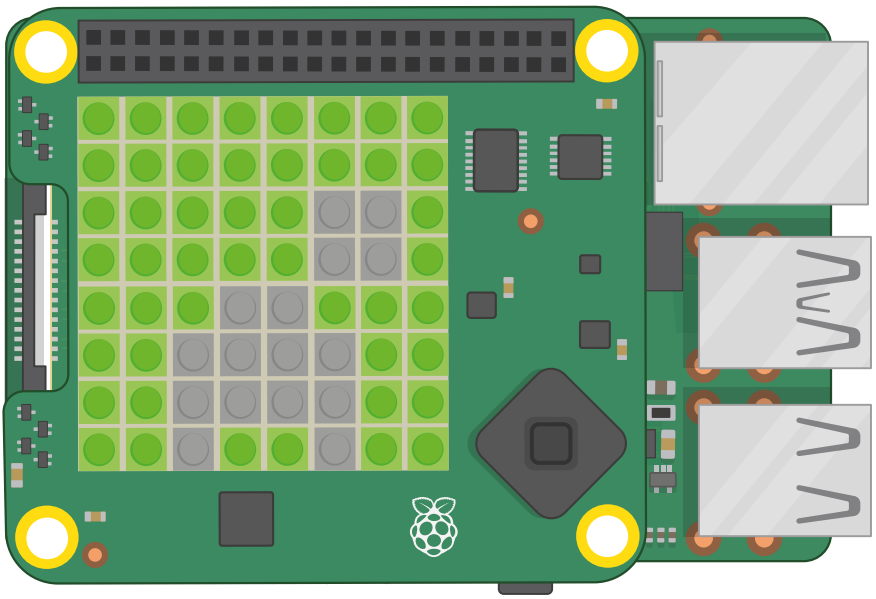
Klik på Run (Kør), og du vil se elementets venstre øje lukke (**Figur 7-16**, på næste side). For at lave en animation skal du gå til toppen af dit program og tilføje linjen:

```
from time import sleep
```

Gå derefter til bunden og skriv:

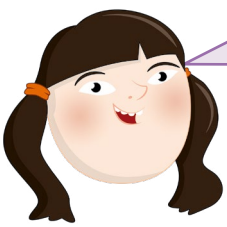
```
while True:
    sleep(1)
    sense.flip_h()
```

Klik på Run (Kør), og se elementet, når det lukker og åbner øjnene, et ad gangen!



Figur 7-16: Viser en enkel animation med to billeder

Funktionen **`flip_h()`** vender et billede på den vandrette akse på tværs; Hvis du vil vende et billede på dets lodrette akse, skal du erstatte **`sense.flip_h()`** med **`sense.flip_v()`** i stedet. Du kan også rotere et billede med 0, 90, 180 eller 270 grader ved hjælp af **`sense.set_rotation(90)`** og ændre antallet efter hvor mange grader, du vil rotere billedet. Prøv at bruge dette for at få elementet til at dreje rundt i stedet for at blinke!

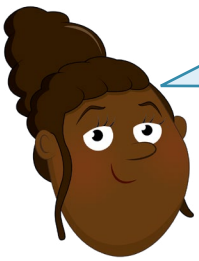


UDFORDRING: NYE DESIGN

Kan du designe flere billeder og animationer? Prøv at få fat på noget graf- eller gitterpapir, og brug det til først at planlægge dit billede i hånden, så det bliver nemmere bagefter at skrive variablen. Kan du tegne et billede og få farverne til at ændre sig? Tip: Du kan ændre variablerne, når du allerede har brugt dem en gang.

Måling af verden omkring dig

Sense HATs virkelige kraft ligger i de forskellige sensorer, den har. De giver dig mulighed for at aflæse alt lige fra temperatur til acceleration og bruge disse aflæsninger i dine programmer, som du finder det passende.



EMULERING AF SENSORERNE

Hvis du bruger Sense HAT-emulatoren, skal du aktivere simulering af inerti og miljøsensorer: Klik på Edit (Rediger) i emulatoren, derefter på Preferences (Indstillinger), og marker dem derefter. I den samme menu skal du vælge "180°..360°|0°..180°" under Orientation Scale (Retningsskala) for at sikre dig, at tallene i emulatoren matcher de numre, der er rapporteret af Scratch og Python, og klik derefter på knappen Close (Luk).

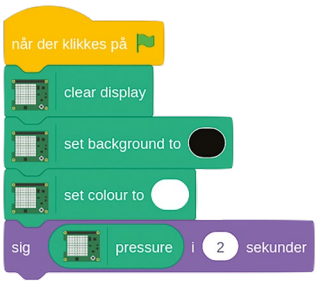
Måling af omgivelserne

Sensoren for barometrisk tryk, fugtighedssensoren og temperatursensoren er alle miljøsensorer; de foretager målinger fra miljøet omkring din Sense HAT.

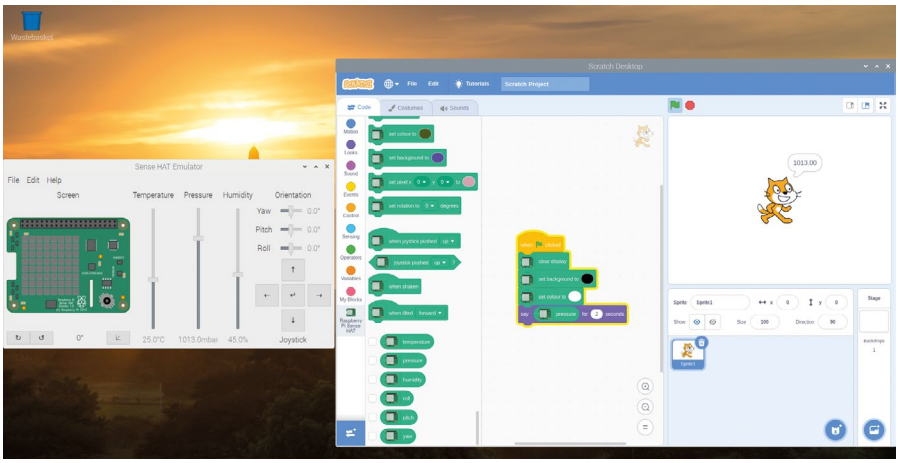
Måling af omgivelserne i Scratch

Start et nyt program i Scratch, og gem dit gamle, hvis du ønsker det, og tilføj Raspberry Pi Sense HAT-udvidelsen, hvis den ikke allerede er indlæst. Træk en når der klikkes på hændelsesbrik til dit kodeområde, derefter en clear display brik nedenunder og en set background to sort brik derunder. Dernæst tilføj en set colour to hvid brik – brug skyderne Brightness (Lysstyrke) og Saturation (Mætning) til at vælge den rigtige farve. Det er altid en god ide at gøre dette i starten af dine programmer, da det vil sikre, at Sense HAT ikke viser noget fra et gammelt program, samtidig med at du sikrer, hvilke farver du bruger.

Træk en sig Hej! i 2 sekunder Udseende-brik direkte under dine eksisterende brikker. For at fortage en aflæsning fra tryksensoren skal du finde pressure brikken i Raspberry Pi Sense HAT-kategorien og trække den over ordet "Hej!" I din sig Hej! i 2 sekunder brik.



Klik på det grønne flag, og Scratch-katten fortæller dig den aktuelle aflæsning fra tryksensoren i *millibar*. Efter to sekunder forsvinder beskeden; prøv at blæse på Sense HAT (eller flytte skyderen Pressure (Tryk) op i emulatoren) og klikke på det grønne flag for at køre programmet igen; du skulle gerne kunne se en højere aflæsning denne gang (**Figur 7-17**, på næste side).

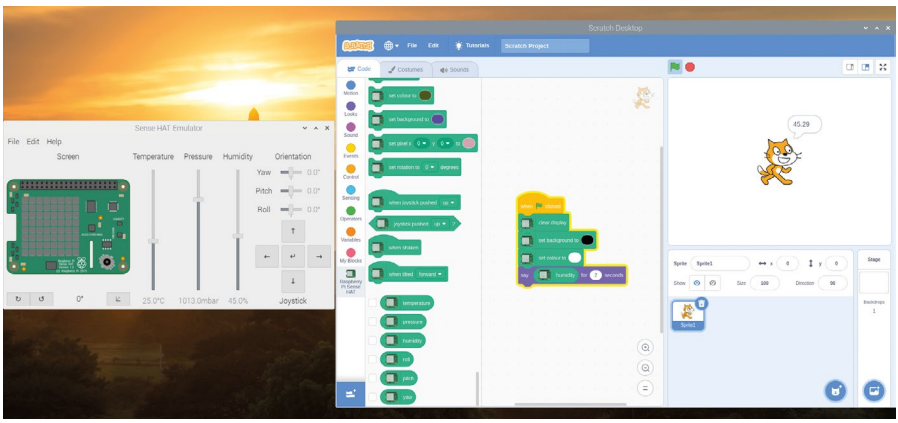


▲ **Figur 7-17:** Viser trykfølerens aflæsning

ÆNDRING AF VÆRDIER

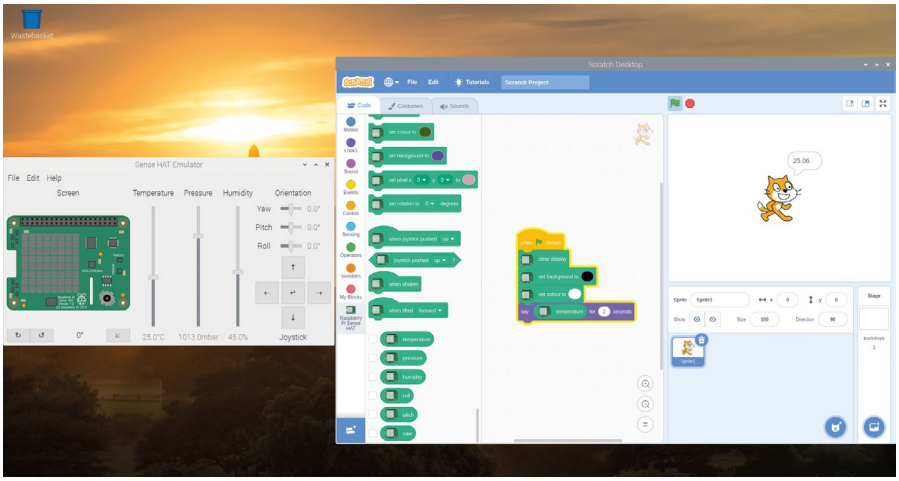
Hvis du bruger Sense HAT-emulatoren, kan du ændre de værdier, der rapporteres af hver af de emulerede sensorer, ved hjælp af skyderne og knapperne. Prøv at skubbe tryksensorindstillingen ned mod bunden, og klik derefter på det grønne flag igen.

For at skifte til fugtighedssensoren skal du slette `pressure` brikken og erstatte den med `humidity`. Kør dit program igen, så ser du den aktuelle relative luftfugtighed i dit værelse. Du kan prøve at køre det igen, mens du blæser på Sense HAT (eller flytter skyderen Humidity (Fugtighed) i emulatoren opad) for at ændre aflæsningen (**Figur 7-18**) – din ånde er overraskende fugtig!

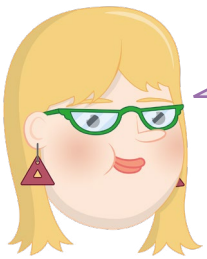


▲ **Figur 7-18:** Viser aflæsningen fra fugtighedssensoren

For temperatursensoren er det blot at slette humidity brikken og erstatte den med temperature og derefter køre dit program igen. Du ser en temperatur i grader Celsius (**Figur 7-19**). Dette er muligvis ikke den helt nøjagtige temperatur i dit værelse, men: Raspberry Pi genererer varme, når den kører, og dette opvarmer også Sense HAT og dens sensorer.



▲ **Figur 7-19:** Viser temperatursensorens aflæsning



UDFORDRING: RULLE OG LØKKE

Kan du ændre dit program til at foretage en aflæsning fra hver af sensorerne igen og derefter rulle dem over LED-matrixen i stedet for at skrive dem til scenen? Kan du få dit program til at køre i løkke, så det hele tiden skriver de aktuelle miljøforhold?

Måling af omgivelserne i Python

For at begynde at foretage målinger fra sensorer skal du oprette et nyt program i Thonny og gemme det som **Sense HAT-sensorer**. Skriv følgende i scriptområdet, som du altid skal, når du bruger Sense HAT – og husk at bruge `sense_emu`, hvis du bruger emulatoren:

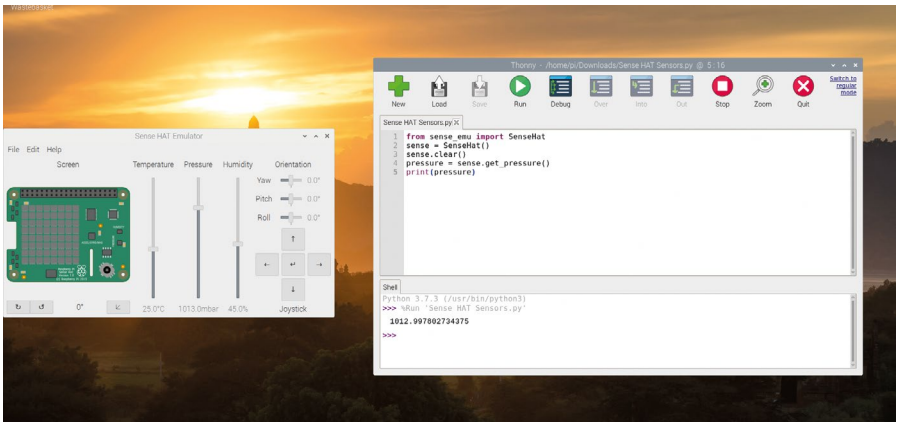
```
from sense_hat import SenseHat
sense = SenseHat()
sense.clear()
```

Det er altid en god ide at medtage `sense.clear()` i starten af dine programmer for det tilfælde, at Sense HATs skærm stadig skulle vise noget fra det sidste program, som blev kørt.

For at fortage en aflæsning fra tryksensoren skal du skrive:

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

Klik på Run (Kør), og du vil se et nummer skrevet til Pythons shell-område i bunden af Thonny-vinduet. Dette er lufttryksaflæsningen registreret af barometertryksensoren i *millibar* (**Figur 7-20**). Prøv at blæse på Sense HAT (eller flytte skyderen Pressure (Tryk) opad i emulatoren), mens du klikker på ikonet Run (Kør) igen; tallet bør nu være højere.



▲ **Figur 7-20:** Udskrivning af en trykmåling fra Sense HAT

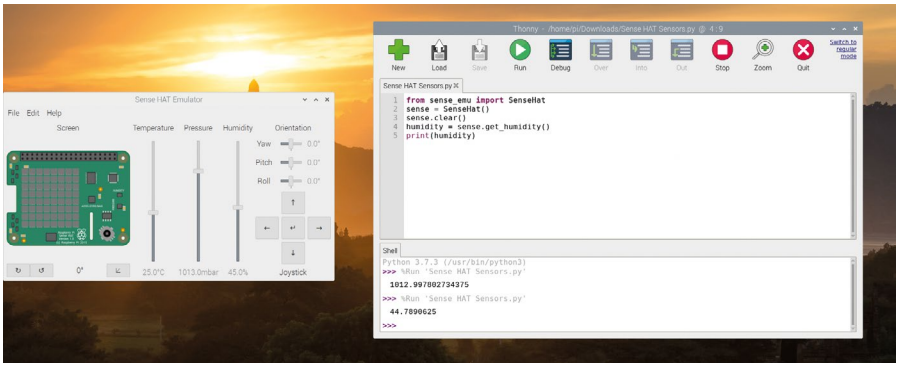
ÆNDRING AF VÆRDIER

Hvis du bruger Sense HAT-emulatoren, kan du ændre de værdier, der rapporteres af hver af de emulerede sensorer, ved hjælp af skyderne og knapperne. Prøv at skubbe tryksensorindstillingen ned mod bunden, og klik derefter på Run (Løb) igen.

For at skifte til fugtighedssensoren skal du fjerne de sidste to linjer med kode og erstatte dem med:

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

Klik på Run (Kør), og du vil se et andet nummer skrevet til Pythons shell-område: Denne gang er det den aktuelle relative fugtighed i dit værelse angivet i procent. Igen kan du prøve at blæse på Sense HAT (eller flytte skyderen Humidity (Fugtighed) opad i emulatoren), og du vil se den gå op, når du kører dit program igen (**Figur 7-21**) – din ånde er overraskende fugtig!

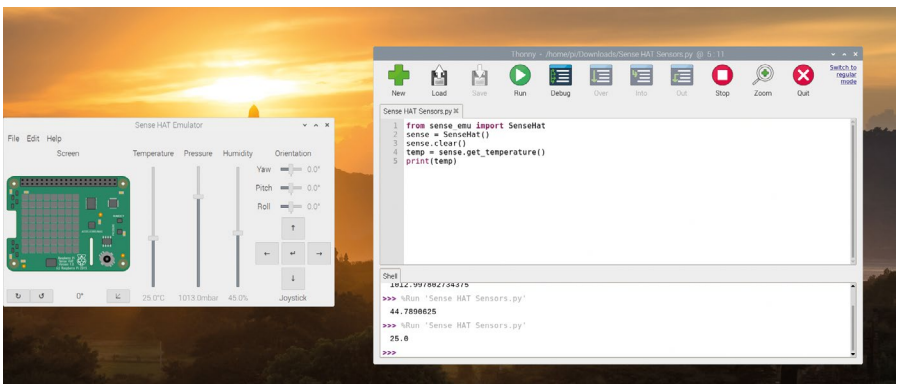


▲ **Figur 7-21:** Visning af fugtighedssensorens aflæsning

Med hensyn til temperaturføleren skal du fjerne de sidste to linjer i dit program og erstatte dem med:

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

Klik på Run (Kør) igen, så vil du se en temperatur i grader Celsius (**Figur 7-22**). Dette er muligvis ikke den helt nøjagtige temperatur i dit værelse, men: Raspberry Pi genererer varme, når den kører, og dette opvarmer også Sense HAT og dens sensorer.

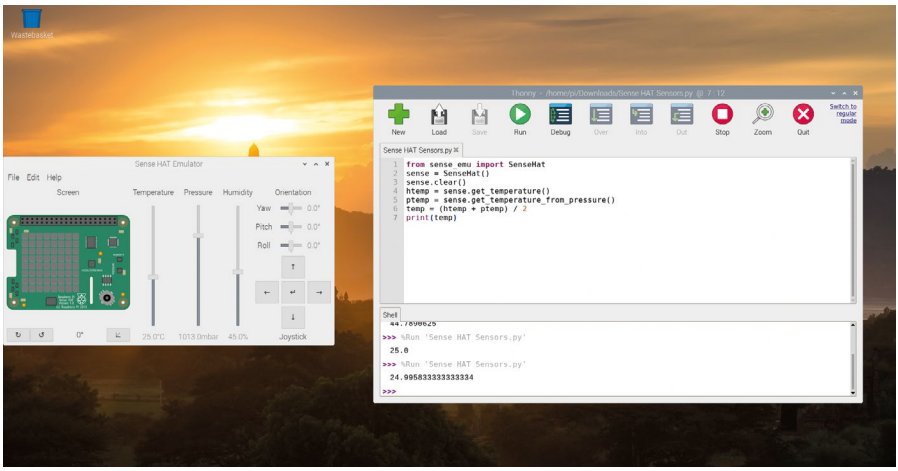


▲ **Figur 7-22:** Viser den aktuelle temperaturaflæsning

Normalt rapporterer Sense HAT temperaturen baseret på en aflæsning fra temperatursensoren indbygget i fugtighedssensoren; Hvis du i stedet vil bruge aflæsningen fra tryksensoren, skal du bruge **`sense.get_temperature_from_pressure()`**. Det er også muligt at kombinere de to aflæsninger for at få et gennemsnit, som kan være mere nøjagtigt end at bruge en sensor alene. Slet de to sidste linjer i dit program, og skriv:

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

Klik på ikonet Run (Kør), og du vil se et nummer, der er udskrevet på Python-konsollen (**Figur 7-23**). Denne gang er det baseret på aflæsninger fra begge sensorer, som du har lagt sammen og divideret med to – antallet af aflæsninger – for at få et gennemsnit af begge. Hvis du bruger emulatoren, viser alle tre metoder – fugtighed, tryk og gennemsnit – det samme tal.



▲ **Figur 7-23:** En temperatur baseret på aflæsningerne fra begge sensorer

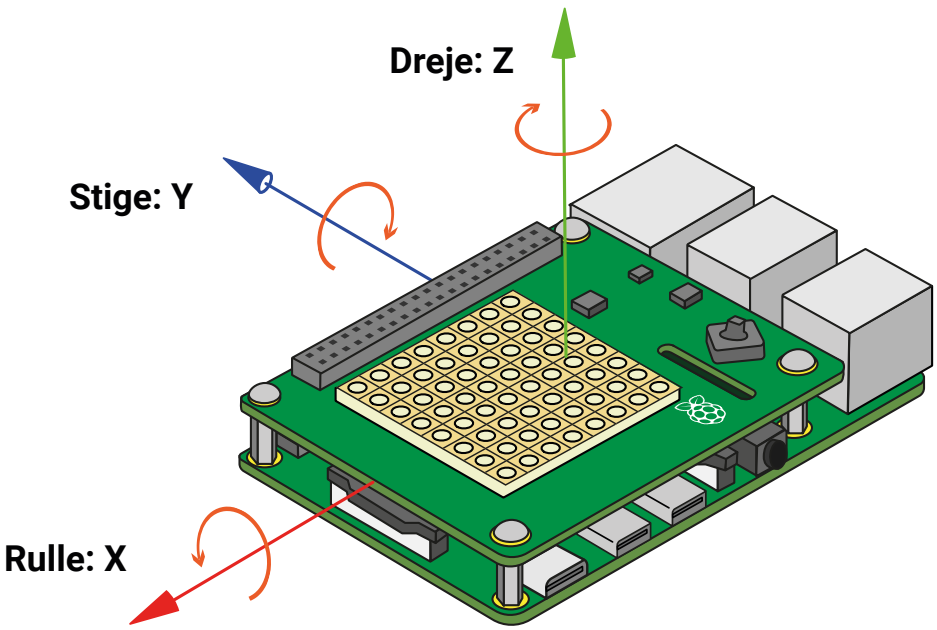
UDFORDRING: RULLE OG LØKKE

Kan du ændre dit program til at foretage en aflæsning fra hver af sensorerne igen og derefter rulle dem over LED-matrixen i stedet for at skrive dem til shell-området? Kan du få dit program til at køre i løkke, så det hele tiden skriver de aktuelle miljøforhold?

Inertimåling

Den gyroskopiske sensor, accelerometer og magnetometer danner det, der er kendt som en *inertimåleenhed*. Mens disse sensorer teknisk set tager målinger fra det omgivende miljø ligesom miljøsensorerne – magnetometeret måler for eksempel magnetfeltstyrken – bruges de normalt til data om bevægelsen af selve Sense HAT. Inertimåleenheden er summen af flere sensorer; nogle programmeringssprog giver dig mulighed for at tage målinger fra hver sensor uafhængigt, mens andre kun giver dig en kombineret aflæsning.

Før du kan forstå inertimåleenheden, skal du dog først forstå, hvordan tingene bevæger sig. Sense HAT og den Raspberry Pi, som den er fastgjort til, kan bevæge sig langs tre rumlige akser: Side-til-side på X-aksen; fremad og bagud på Y-aksen; og op og ned på Z-aksen (**Figur 7-24**). Det kan også rotere langs disse tre samme akser, men deres navne ændres: Rotering på X-aksen kaldes *rulle*, rotering på Y-aksen kaldes *stige*, og rotation på Z-aksen kaldes *dreje*. Når du roterer Sense HAT langs dens korte akse, justerer du dens stigning; roter langs dens lange akse, og det er at rulle; drej den rundt, mens du holder den fladt på bordet, og du justerer dens drejning. Tænk på dem som et fly: Når det starter, øger det sin stigning for at komme op; når det laver en sejrsrulning, drejer det sig bogstaveligt talt langs sin egen rulleakse; når det bruger roret til at dreje som en bil, uden at rulle, er det en drejning.

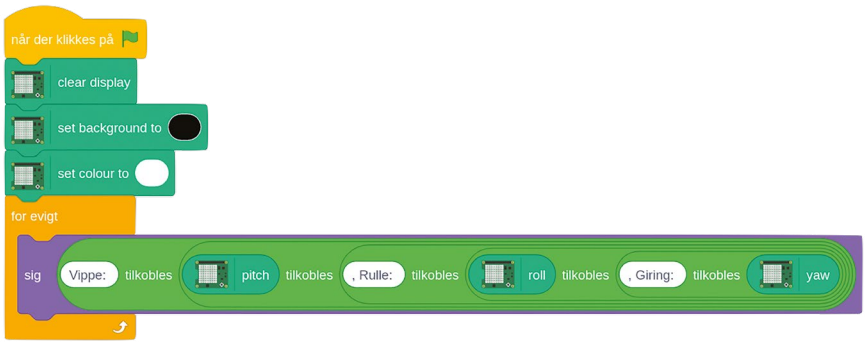


▲ **Figur 7-24:** De rumlige akser i Sense HAT's inertimåleenheder

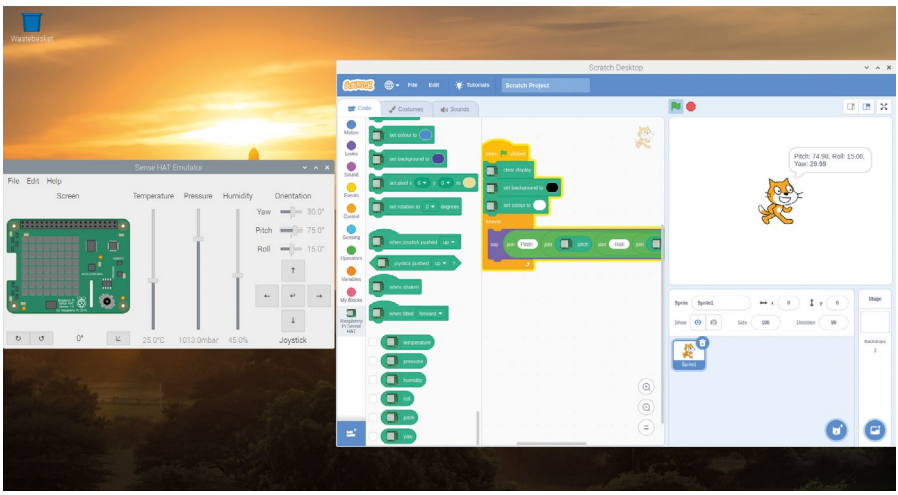
Inertimåling i Scratch

Start et nyt program i Scratch, og indlæs Raspberry Pi Sense HAT-udvidelsen, hvis den ikke allerede er indlæst. Start dit program på samme måde som før: Træk en når der klikkes på hændelsesbrik til dit kodeområde, og træk derefter en clear display brik under den og træk og rediger derefter en set background to sort og en set colour to hvid brik.

Træk derefter en for evigt brik til bunden af dine eksisterende brikker og udfyld den med en sig Hej! brik. For at vise en aflæsning for hver af de tre akser i inertimåleenhederne – stige, rulle og dreje – skal du tilføje tilkobles operatørbrikker plus de tilsvarende Sense HAT-brikker. Husk at medtage mellemrum og kommaer, så output er let at læse.



Klik på det grønne flag for at køre dit program, og prøv at flytte Sense HAT og Raspberry Pi rundt – pas på ikke at fjerne nogen kabler! Når du vipper Sense HAT gennem dens tre akser, vil du se, at værdierne for stigning, rulning og drejning ændres i overensstemmelse hermed (**Figur 7-25**).



▲ **Figur 7-25:** Visning af værdier for stigning, rulning og drejning

Inertimåling i Python

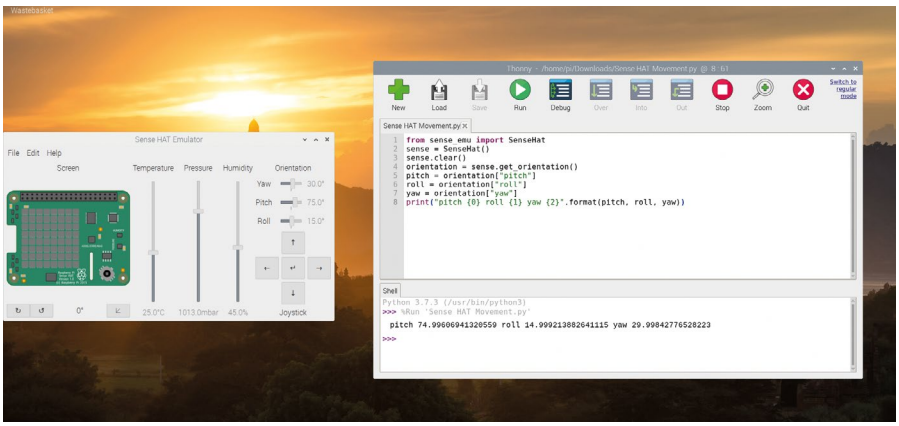
Start et nyt program i Thonny, og gem det som **Sense HAT-bevægelse**. Udfyld de sædvanlige startlinjer, og husk at bruge `sense_emu`, hvis du bruger Sense HAT-emulatoren:

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

Hvis du vil bruge oplysninger fra inertimåleenheden til at fastlægge den aktuelle retning af Sense HAT på dens tre akser, skal du skrive følgende:

```
orientation = sense.get_orientation()
pitch = orientation["pitch"]
roll = orientation["roll"]
yaw = orientation["yaw"]
print("stige {0} rulle {1} dreje {2}".format(pitch, roll, yaw))
```

Klik på Run (Kør), så får du vist målingerne for Sense HATs retning delt over de tre akser (**Figur 7-26**). Prøv at rotere Sense HAT og klikke på Run (Kør) igen; tallene skulle nu gerne ændre sig for at afspejle den nye retning.



▲ **Figur 7-26:** Viser Sense HATs værdier for stigning, rulning og drejning

Inertimåleenhederne kan dog mere end måle orienteringen: De kan også registrere bevægelse. For at få nøjagtige aflæsninger af bevægelse skal inertimåleenheden læses hyppigt i en løkke: I modsætning til retning, giver en enkelt læsning ikke dig nyttige oplysninger med hensyn til at registrere bevægelse. Slet alt efter `sense.clear()`, skriv derefter følgende kode:

```
while True:
    acceleration = sense.get_accelerometer_raw()
    x = acceleration["x"]
    y = acceleration["y"]
    z = acceleration["z"]
```

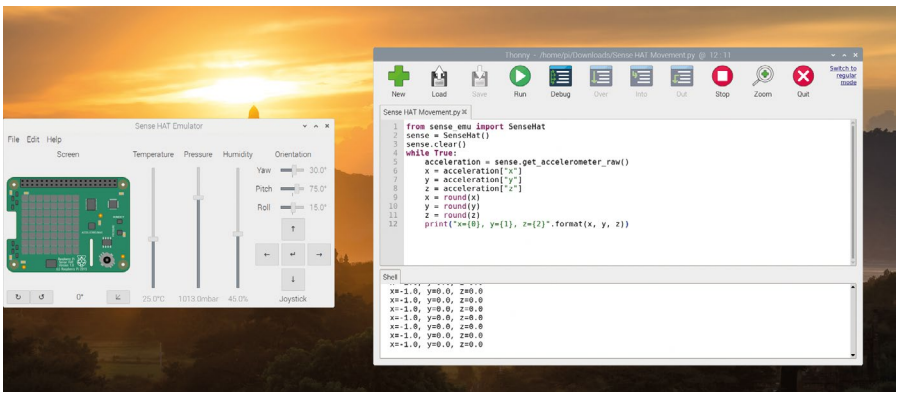
Du har nu variabler, der indeholder de aktuelle accelerometeraflæsninger for de tre rumlige akser: X eller venstre og højre; Y, eller fremad og bagud; og Z, eller op eller ned. Tallene fra accelerometersensoren kan være vanskelige at læse, så skriv følgende for at gøre dem lettere at forstå ved at afrunde dem til nærmeste heltal:

```
    x = round(x)
    y = round(y)
    z = round(z)
```

Til sidst skal du skrive de tre værdier ved at skrive følgende linje:

```
    print("x={0}, y={1}, z={2}".format(x, y, z))
```

Klik på Run (Kør), så får du vist værdier fra accelerometeret, der er skrevet til Python-shellområdet (**Figur 7-27**). I modsætning til dit tidligere program skrives disse kontinuerligt; for at stoppe skrivningen skal du klikke på den røde stopknap for at stoppe programmet.



▲ **Figur 7-27:** Accelerometeraflæsninger afrundet til nærmeste heltal

Du har måske bemærket, at accelerometeret fortæller dig, at en af akserne – Z-aksen, hvis din Raspberry Pi ligger fladt på bordet – har en accelerationsværdi på 1,0 (1G), selvom Sense HAT ikke bevæger sig. Det skyldes, at det registrerer Jordens tyngdekraft – den kraft, der trækker Sense HAT ned mod midten af jorden, og grunden til, at hvis noget ryger ud over kanten på dit skrivebord, så falder det på gulvet.

Når dit program kører, skal du forsigtigt tage Sense HAT og Raspberry Pi op og dreje dem rundt – men sørg for ikke at komme til at løsne nogen af kablerne! Når Raspberry Pi's netværk og USB-porte peger mod gulvet, vil du se, at værdierne ændres, så Z-aksen viser 0G, og X-aksen nu viser1G; drej dem igen, så HDMI- og strømstikkene peger mod gulvet, og nu er det Y-aksen, der viser 1G. Hvis du gør det modsat, så HDMI-stikket peger mod loftet, vil du se -1G på Y-aksen i stedet.

Ved hjælp af viden om, at jordens tyngdekraft er omkring 1G, og din viden om de rumlige akser, kan du bruge aflæsninger fra accelerometeret til at finde ud af, hvilken vej der er ned – og på samme måde, hvilken vej der er op. Du kan også bruge den til at registrere bevægelse: Prøv forsigtigt at ryste Sense HAT og Raspberry Pi, og se på tallene, mens du gør det: Jo mere du ryster, jo større acceleration.

Når du bruger **`sense.get_accelerometer_raw()`**, beder du Sense HAT om at slukke for de to andre sensorer i inertimåleenheden – den gyroskopiske sensor og magnetometeret – og kun returnere data fra accelerometeret. Du kan naturligvis også gøre det samme med de andre sensorer.

Find linjen **`acceleration = sense.get_accelerometer_raw()`** og rediger den til:

```
orientation = sense.get_gyroscope_raw()
```

Rediger ordet **`acceleration`** i alle tre linjer under det til **`orientation`**. Klik på Run (Kør), så ser du retningen på Sense HAT for alle tre akser afrundet til nærmeste heltal. I modsætning til sidste gang du kontrollerede orienteringen, kommer denne dataene denne gang udelukkende fra gyroskopet uden brug af accelerometer eller magnetometer. Dette kan være nyttigt, hvis du f.eks. vil kende retningen af en Sense HAT, som bevæger sig, fordi den er placeret bag på en robot, uden at bevægelsen kommer til at påvirke de øvrige data, eller hvis du bruger Sense HAT i nærheden af et stærkt magnetfelt

Stop dit program ved at klikke på den røde stopknap. For at bruge magnetometeret skal du slette alt fra dit program undtagen de første fire linjer, og derefter skrive følgende under linjen **`while True`**:

```
north = sense.get_compass()
print(north)
```

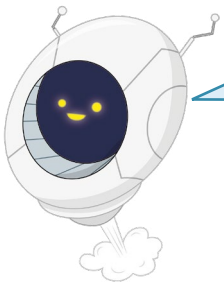
Kør dit program, og du vil se retningen af magnetisk nord skrevet gentagne gange til shell-området i Python. Drej forsigtigt Sense HAT, og du vil se overskriften ændre sig, når Sense HATs retning ændrer sig i forhold til nord: Du har bygget et kompas! Hvis du har en magnet – en køleskabsmagnet er fint – kan du prøve at flytte den rundt på Sense HAT for at se, hvad det gør ved magnetometerets aflæsninger.

UDFORDRING: AUTOMATISK ROTATION

Kan du bruge det, du har lært om LED-matrixen og sensorerne til inertimåleenheden, til at skrive et program, der roterer et billede afhængigt af placeringen af Sense HAT?

Joystick-kontrol

Sense HATs joystick, der findes i nederste højre hjørne, er muligvis lille, men det er overraskende kraftigt: Ud over at kunne genkende input fra fire retninger – op, ned, venstre og højre – har den også et femte input, som fås ved at skubbe den ned ovenfra som en trykknapkontakt.



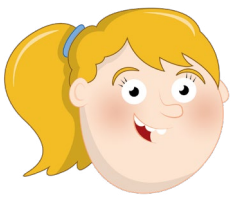
ADVARSEL!

Sense HAT-joysticket bør kun bruges, hvis du har monteret afstandsstykkerne som beskrevet i starten af dette kapitel. Uden afstandsstykkerne kan tryk på joysticket bøje Sense HAT-kortet og beskadige både Sense HAT og Raspberry Pi's GPIO-stik.

Joystick-kontrol i Scratch

Start et nyt program i Scratch med Raspberry Pi Sense HAT-udvidelsen indlæst. Træk på samme måde som før en [når der klikkes på] hændelsesbrik til dit scriptområde, og træk derefter en [clear display] brik under den og træk og rediger derefter en [set background to sort] og en [set colour to hvid] brik.

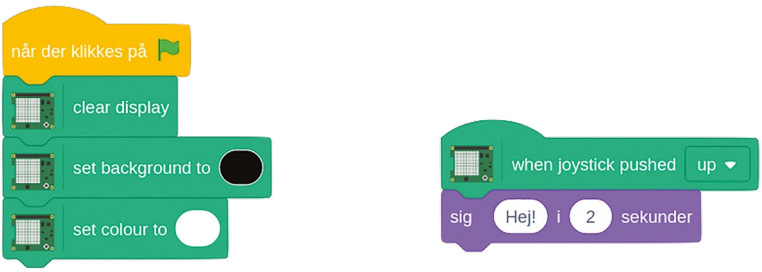
I Scratch tildeles Sense HATs joystick til markørtasterne på tastaturet: At skubbe joysticket op svarer til at trykke på pil op-tasten, at skubbe det ned er det samme som at trykke på pil ned-tasten, og at skubbe det til venstre svarer til venstre piletast, og at skubbe den til højre svarer til højre piletast; at skubbe joysticket indad som en trykknapkontakt svarer til at trykke på **ENTER**-tasten.



ADVARSEL!

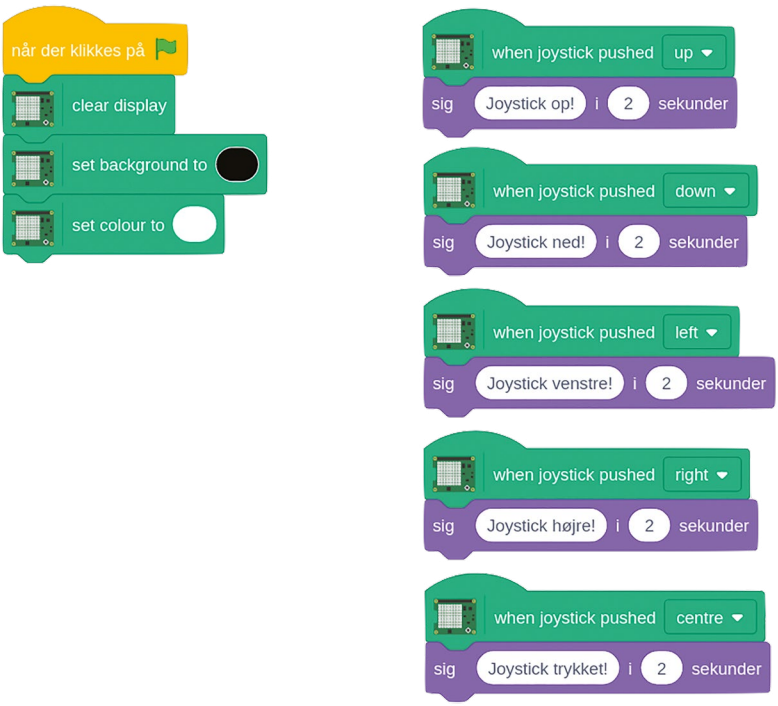
Joystick-kontrol er kun tilgængelig på den fysiske Sense HAT. Når du bruger Sense HAT-emulatoren, skal du bruge de tilsvarende taster på dit tastatur til at simulere joystick-tryk i stedet.

Træk en **when joystick pushed up** brik til dit kodeområde. Træk derefter en **sig Hej! i 2 sekunder** brik under den for at give den noget at lave.

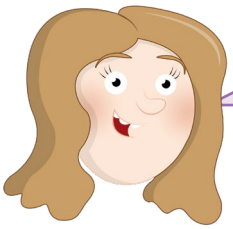


Skub joysticket opad, så ser du Scratch-katten sige et muntert "Hej!"

Derefter skal du redigere **sig Hej! i 2 sekunder** brikken til en **sig Joystick op! i 2 sekunder** brik og fortsætte med at tilføje hændelses- og Udseende-brikker, indtil du har noget til hver af de fem måder, joysticket kan trykkes på.



Prøv at skubbe joysticket i forskellige retninger for at se, at dine beskeder så bliver vist!



SIDSTE UDFORDRING

Kan du bruge Sense HATs joystick til at styre en Scratch sprite på scenen? Kan du gøre det således, at hvis sprite samler en anden sprite, der repræsenterer en genstand, så viser Sense HATs lysdioder en munter besked?

Joystick-kontrol i Python

Start et nyt program i Thonny, og gem det som Sense HAT-Joystick. Begynd med de sædvanlige tre linjer, der konfigurerer Sense HAT, og ryd LED-matrixen:

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

Konfigurer derefter en uendelig løkke:

```
while True:
```

Bed derefter Python om at lytte efter input fra Sense HAT-joysticket med følgende linje, som Thonny automatisk indrykker for dig:

```
    for event in sense.stick.get_events():
```

Tilføj til sidst følgende linje – som Thonny igen vil indrykke for dig – for faktisk at gøre noget, når der registreres et joystick-tryk:

```
        print(event.direction, event.action)
```

Klik på Run (Kør), og prøv at skubbe joysticket i forskellige retninger. Du vil se den retning, du har valgt, skrevet til Python-shellområdet: Op, ned, venstre, højre og midten, når du har skubbet joysticket ned som en trykknapkontakt.

Du vil også se, at du får to hændelser, hver gang du skubber joysticket én gang: en hændelse, **`pressed`**, til når du først skubber i en retning; den anden hændelse, **`released`**, når joysticket vender tilbage til centrum. Du kan bruge dette i dine programmer: Tænk på en karakter i et spil, som man kunne få det til at begynde at bevæge sig, når joysticket trykkes i en retning, og stoppe, så snart det frigives.

Du kan også bruge joysticket til at udløse funktioner i stedet for at være begrænset til at bruge en løkke. Slet alt under **`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 funktioner ændrer hele Sense HAT LED-matrixen til en enkelt farve: Rød, blå, grøn eller gul – hvilket vil gøre det ekstremt nemt at se, at dit program fungerer! For rent faktisk at udløse dem skal du fortælle Python, hvilken funktion der passer til hvilket joystickinput. 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
```

Endelig har programmet brug for en uendelig løkke – kendt som *hovedløkken* – for at fortsætte med at køre og derfor holde øje med joystickinputs snarere end bare at køre gennem den kode, som du har skrevet, én gang og så stoppe. Skriv følgende to linjer:

```
while True:
    pass
```

Klik på Run (Kør), og prøv at flytte joysticket: Du vil se lysdioderne lyse i strålende farver! For at slukke for LED'erne skal du trykke på joysticket som en trykknap: Den midterste retning (**`middle`**) er indstillet til at bruge **`sense.clear()`**-funktionen til at slukke for dem alle. Tillykke: Du kan fange input fra joysticket!

SIDSTE UDFORDRING

Kan du bruge det, du har lært, til at tegne et billede til skærmen, og derefter få det til at rotere i uanset hvilken retning joysticket skubbes? Kan du få midterinputtet til at skifte mellem mere end ét billede?

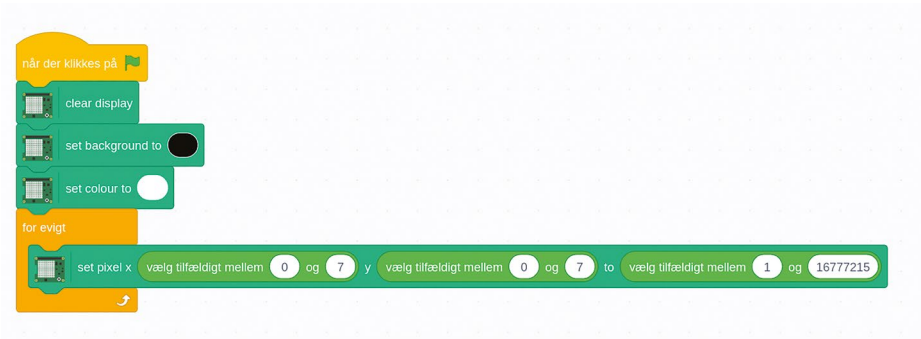
Scratch-projekt: Sense HAT-stjernekaster

Nu hvor du efterhånden føler dig godt hjemme i Sense HAT, er det på tide at samle alt, hvad du har lært, for at opbygge en varmefølsom stjernekaster – en enhed, der har det bedst, når den er kold, og som gradvist sætter tempoet ned, jo varmere den bliver.

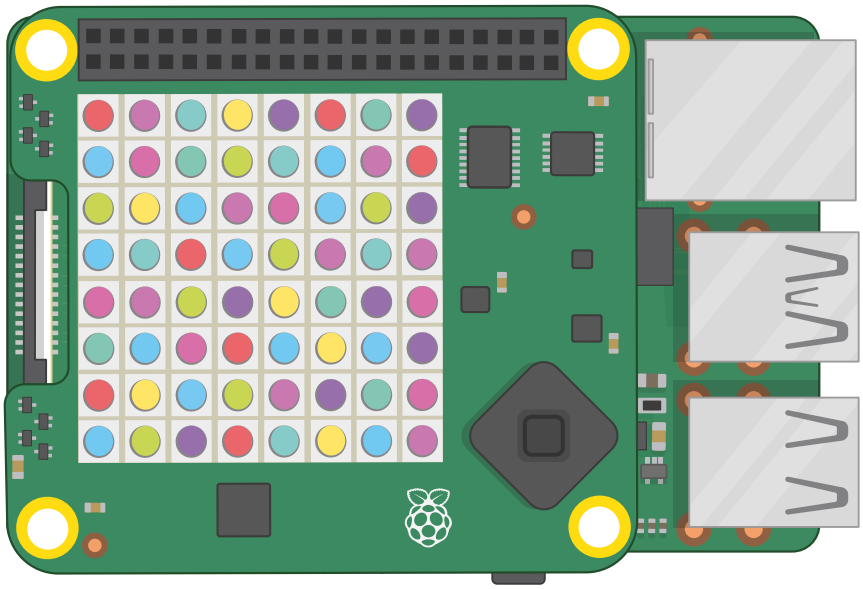
Start et nyt Scratch-projekt, og tilføj Raspberry Pi Sense HAT-udvidelsen, hvis den ikke allerede er indlæst. Begynd som altid med fire brikker: når der klikkes på , clear display , set background to sort og set colour to hvid , idet du husker, at du skal ændre farverne fra deres standardindstillinger.

Start med at skabe en enkel, men kunstnerisk, stjernekaster. Træk en for evigt brik til kodeområdet, og udfyld den derefter med en set pixel x 0 y 0 to farve brik. I stedet for at bruge faste tal skal du udfylde hver af x-, y- og farvesektionerne for den pågældende brik med en vælg tilfældigt mellem 1 og 10 operatørbrik.

Værdierne 1 til 10 er ikke særlig nyttige her, så du er nødt til at redigere lidt. De første to tal i set pixel brikken er X- og Y-koordinaterne for pixlen på LED-matrixen, hvilket betyder at de skal være mellem 0 og 7 – så skift de to første brikker, så de hedder vælg tilfældigt mellem 0 og 7 . Det næste afsnit er den farve, pixelen skal indstilles til. Når du bruger farvevælgeren, vises den farve, som du vælger, direkte i scriptområdet; internt er farverne dog repræsenteret af et tal, og du kan bruge tallet direkte. Rediger den sidste vilkårlige brik, så der står vælg tilfældigt mellem 0 og 16777215 .

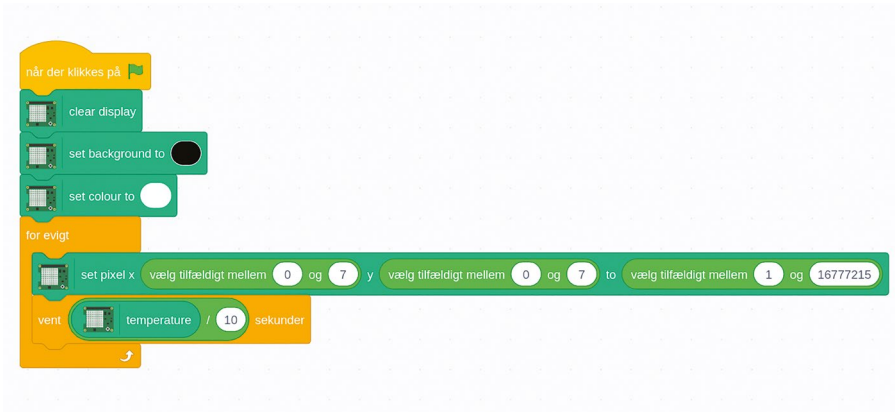


Klik på det grønne flag, så ser du LED'erne på Sense HAT begynder at lyse i tilfældige farver (**Figur 7-28**). Tillykke: Du har lavet en elektronisk stjernekaster!



▲ **Figur 7-28:** Oplysning af pixels i tilfældige farver

Stjernekasteren er ikke særligt interaktiv. For at ændre dette skal du starte med at trække en **vent 1 sekunder** brik, så den er under **set pixel** brikken, men inden for **for evigt** brikken. Træk en **● / ●** operatørbrik over 1, og skriv derefter 10 på den anden plads. Træk til sidst en **temperature** brik over den første plads i delingsoperatørbrikken.



Klik på det grønne flag, og du vil bemærke – medmindre du bor et sted, der er meget koldt – at stjernekasteren er betydeligt langsommere end før. Det skyldes, at du har oprettet en temperaturafhængig forsinkelse: Programmet venter nu et antal sekunder svarende til *den aktuelle temperatur divideret med 10*, før hver løkke. Hvis temperaturen i dit værelse er 20 °C, venter programmet 2 sekunder, før det går i løkke; hvis temperaturen er 10 °C, venter det 1 sekund; hvis temperaturen er under 10 °C, venter programmet i mindre end ét sekund.

Hvis din Sense HAT aflæser en negativ temperatur – under 0 °C, dvs. vandets frysepunkt – prøver den at vente mindre end 0 sekunder; fordi det er umuligt – i hvert fald uden at opfinde tidsrejser – vil du se den samme effekt, som om den ventede 0 sekunder. Tillykke: Du kan nu begynde at integrere Sense HATs forskellige funktioner i dine egne programmer!

Python-projekt: Sense HAT Tricorder

Nu hvor du føler dig hjemme i Sense HAT, er tid til at samle alt, hvad du har lært, for at opbygge en tricorder – en enhed, der umiddelbart er kendt af fans af en bestemt science fiction-franchise som værende i stand til at rapportere om forskellige sensorer indbygget i den.

Start et nyt projekt i Thonny, og gem det som **Tricorder**, og start derefter med de traditionelle linjer, der skal bruges for at lave et Sense HAT-program:

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

Dernæst skal du begynde at definere funktioner til hver af Sense HATs forskellige sensorer. Start med inertimåleenheden ved at skrive:

```
def orientation():
    orientation = sense.get_orientation()
    pitch = orientation["pitch"]
    roll = orientation["roll"]
    yaw = orientation["yaw"]
```

Fordi du skal rulle resultaterne fra sensoren hen over lysdioderne, er det fornuftigt at afrunde dem, så du ikke skal vente på snesevis af decimaler. I stedet for hele tal kan du afrunde dem til én decimal ved at skrive følgende:

```
    pitch = round(pitch, 1)
    roll = round(roll, 1)
    yaw = round(yaw, 1)
```

Endelig skal du bede Python om at rulle resultaterne til lysdioderne, så tricorderen fungerer som en håndholdt enhed uden at skulle være forbundet til en skærm eller et tv:

```
    sense.show_message("Stige {0}, Rulle {1}, Dreje {2}".
format(pitch, roll, yaw))
```

Nu hvor du har en fuld funktion til læsning og visning af orienteringen fra inertimåleenhederne, skal du oprette lignende funktioner til hver af de andre sensorer. Start med temperatursensoren:

```
def temperature():
    temp = sense.get_temperature()
    temp = round(temp, 1)
    sense.show_message("Temperatur: %s grader Celsius" % temp)
```

Se nøje på den linje, der udskriver resultatet til lysdioderne: **%s** er kendt som en pladsholder og erstattes med indholdet af variablen **temp**. Ved hjælp af dette kan du formatere outputtet pænt med en etiketten "Temperatur:" og en måleenhed, "grader Celsius", hvilket gør dit program meget mere brugervenligt.

Dernæst definerer du en funktion til fugtighedssensoren:

```
def humidity():
    humidity = sense.get_humidity()
    humidity = round(humidity, 1)
    sense.show_message("Fugtighed: %s procent" % humidity)
```

Derefter tryksensoren:

```
def pressure():
    pressure = sense.get_pressure()
    pressure = round(pressure, 1)
    sense.show_message("Tryk: %s millibar" % pressure)
```

Og endelig kompasaflæsningen 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**-løkke i denne funktion udfører ti aflæsninger fra magnetometeret for at sikre, at det har nok data til at give dig et nøjagtigt resultat. Hvis du opdager, at den rapporterede værdi bliver ved med at skifte, kan du prøve at udvide løkken til 20, 30 eller endda 100 aflæsninger for at forbedre nøjagtigheden yderligere.

Dit program har nu fem funktioner, som hver især foretager en aflæsning fra en af Sense HATs sensorer og ruller dem hen over lysdioderne. Det har dog brug for en måde at vælge, hvilken sensor du vil bruge, og joysticket er perfekt til det.

Skriv følgende:

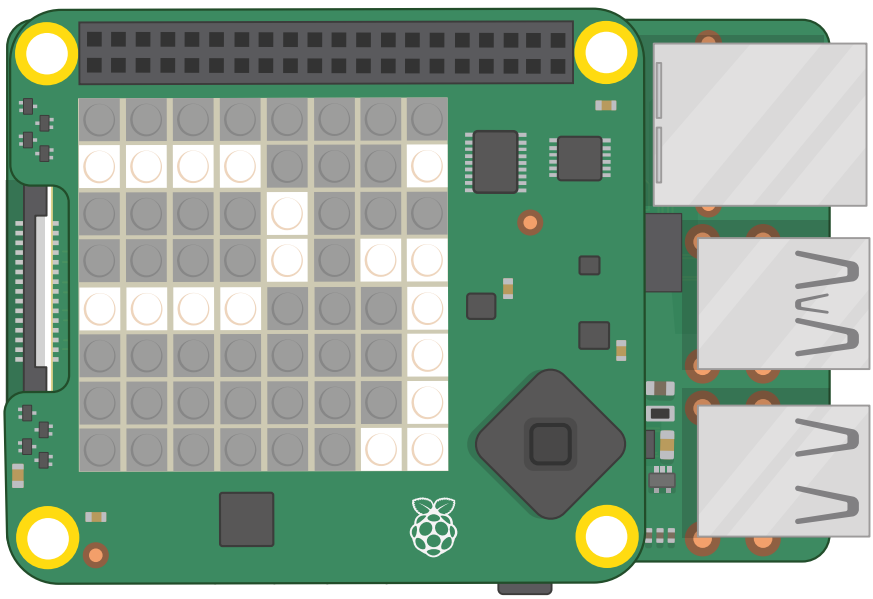
```
sense.stick.direction_up = orientation
sense.stick.direction_right = temperature
sense.stick.direction_down = compass
sense.stick.direction_left = humidity
sense.stick.direction_middle = pressure
```

Disse linjer tildeler en sensor til hver af de fem mulige retninger på joysticket: Op læser fra orienteringssensoren; ned læser fra magnetometeret; venstre læser fra fugtighedssensoren; højre fra temperatursensoren; og når du trykker på midten læses fra trykføleren.

Endelig har du brug for en hovedløkke, så programmet fortsætter med at lytte efter joystick-tryk og ikke bare straks afslutter. Nederst i dit program skal du skrive følgende:

```
while True:
    pass
```

Klik på Run (Kør), og prøv at flytte joysticket for at få en læsning fra en af sensorerne (**Figur 7-29**). Når den er færdig med at rulle resultatet, skal du trykke på en anden retning. Tillykke: Du har bygget et håndholdt tricorder, der vil gøre Den Forenede Føderation af Planeter stolt!



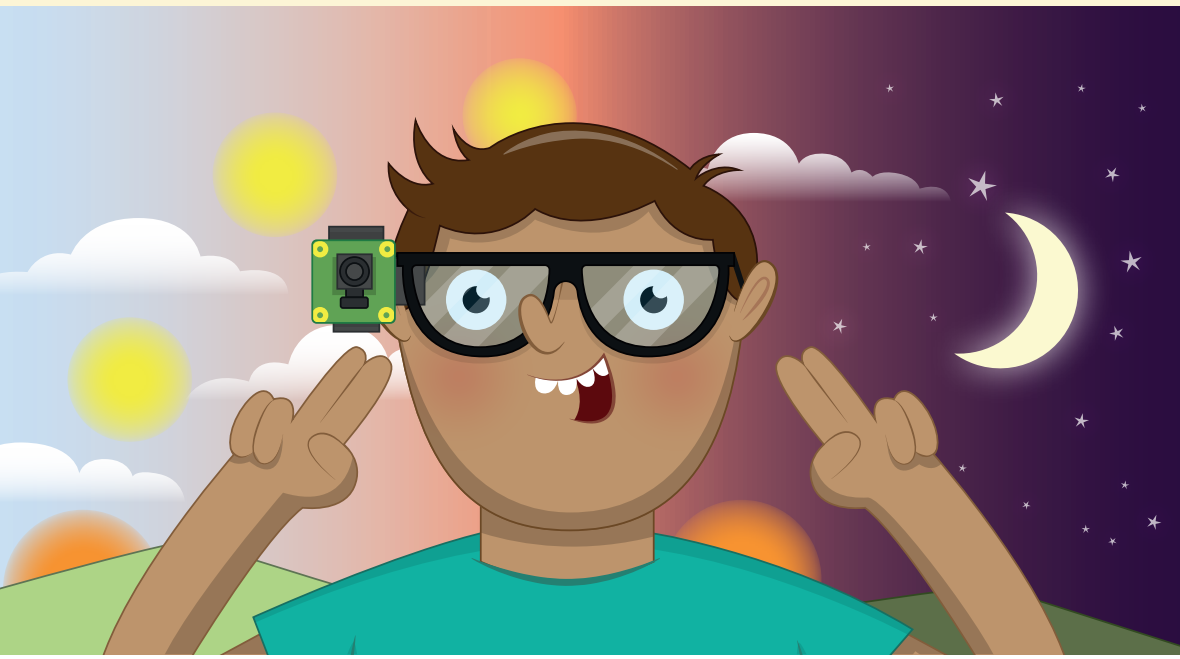
▲ **Figur 7-29:** Hver aflæsning ruller over skærmen

For flere Sense HAT-projekter, følg linkene i **Bilag D, Yderligere læsning**.

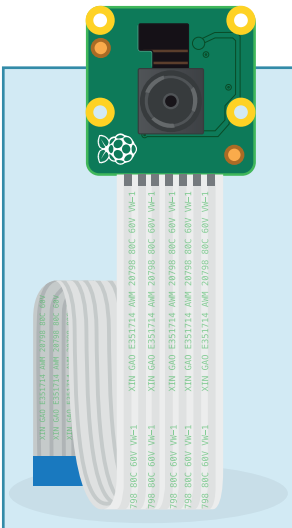
Kapitel 8

Raspberry Pi's Camera Module

Tilslutning af et Camera Module eller HQ Camera til Raspberry Pi giver dig mulighed for at tage fotos i høj opløsning og optage videoer og oprette fantastiske computer vision-projekter



Hvis du nogensinde har ønsket at bygge noget, der kan se – kendt inden for robotteknologi som *computer vision* – så er Raspberry Pi's valgfrie Camera Module eller det nye kamera i høj kvalitet et godt udgangspunkt. Via et lille firkantet kredsløbskort med et tyndt båndkabel tilsluttes Camera Module/HQ Camera til CSI-stikket (Camera Serial Interface) på din Raspberry Pi (ikke tilgængelig på Raspberry Pi 400) og giver stillbilleder i høj opløsning og videosignaler, der kan bruges som de er eller integreres i dine egne programmer.



KAMERATYPER!

Tre typer Raspberry Pi-kameraer er tilgængelige: Standardkameramodulet, "NoIR"-versionen og HQ Camera (High Quality). Hvis du vil tage normale billeder og video i veloplyste miljøer, skal du bruge standardkameramodulet. Hvis du vil bruge specielle objektiver og ønsker den bedste billedkvalitet, skal du bruge HQ Camera-modulet. NoIR-kameramodulet – såkaldt fordi det ikke har noget infrarødt eller IR-filter – er designet til brug med infrarøde lyskilder til at tage billeder og video i totalt mørke. Hvis du bygger en redekasse, et sikkerhedskamera eller et andet projekt, der involverer nattesyn, skal du bruge NoIR-versionen – men husk samtidig at købe en infrarød lyskilde!

Standard- og NoIR Raspberry Pi Camera Module er baseret på en Sony IMX219-billedsensor. Dette er en *8 megapixelsensor*, hvilket betyder, at den kan tage billeder med op til 8 millioner pixels i dem. Det gør den ved at tage billeder der er op til 3280 pixels brede og 2464 høje. Ud over stillbilleder kan Camera Module optage videooptagelser i Full HD-opløsning med en hastighed på 30 billeder pr. sekund (30 fps). For en jævnere bevægelse eller for at skabe en slowmotion-effekt kan kameraet indstilles til at optage med en højere billedhastighed ved at sænke opløsningen: 60 fps til 720p-videooptagelser og op til 90 fps til 480p-optagelser (VGA).

Kameraet i høj kvalitet bruger en Sony IMX477-sensor på 12,3 megapixel, som også er større end den i standard- og NoIR-kameramodulerne – hvilket betyder, at den kan samle mere lys, hvilket igen fører til billeder af højere kvalitet. I modsætning til Camera Module inkluderer HQ Camera dog ikke et objektiv, uden hvilket det ikke kan tage billeder eller videoer. Du kan bruge ethvert objektiv med et C- eller CS-beslag; andre objektivbeslag kan bruges med en passende C- eller CS-beslagadapter. For oplysninger om, hvordan man monterer et objektiv, se **Bilag F: Kamera i høj kvalitet**.

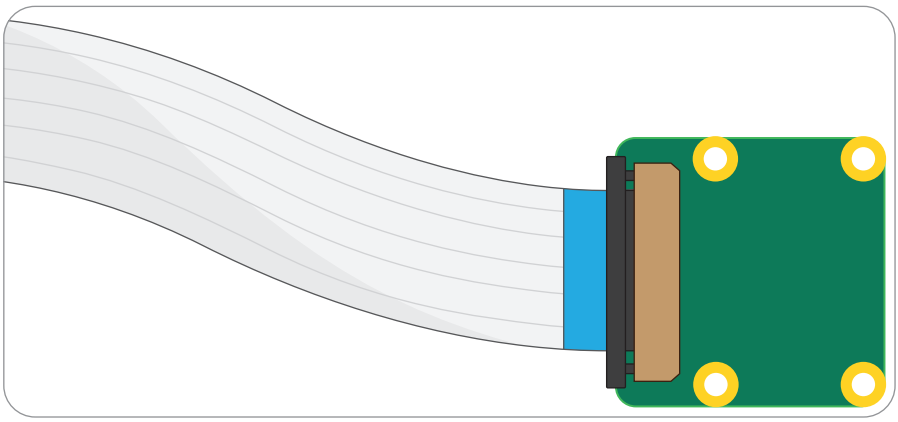
RASPBERRY PI 400

Desværre er Raspberry Pi Camera Module ikke kompatibel med Raspberry Pi 400. Du kan bruge USB-webkameraer som et alternativ, men vil så ikke kunne bruge de specifikke softwareværktøjer til Raspberry Pi Camera Module, som er inkluderet i Raspberry Pi OS.

Installation af kameraet

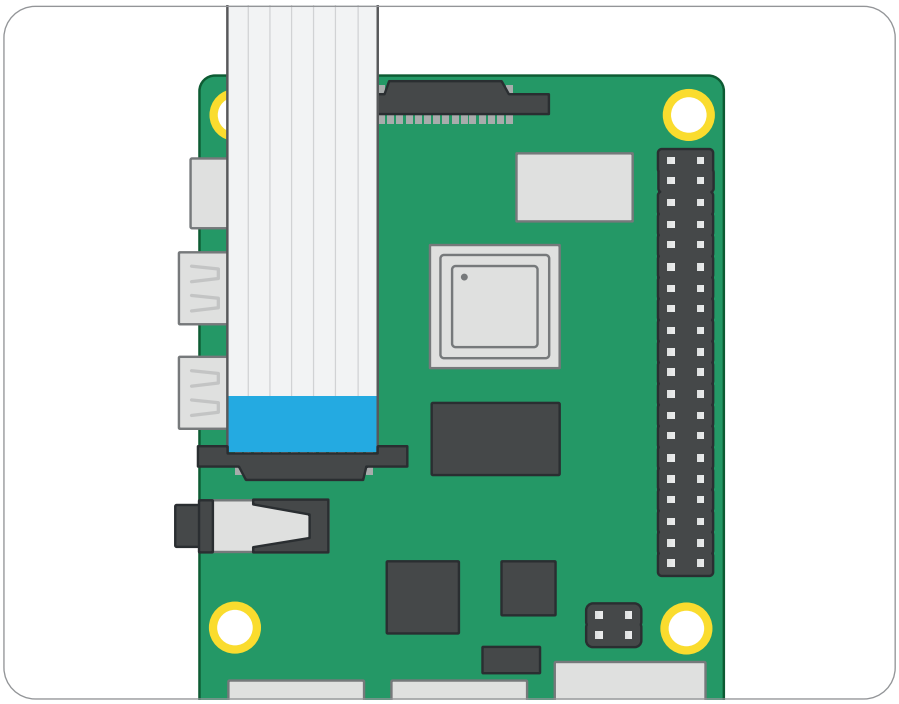
Som enhver hardwaretilføjelse skal Camera Module eller HQ Camera kun tilsluttes eller frakobles Raspberry Pi, når strømmen er slukket, og strømkablet er taget ud. Hvis din Raspberry Pi er tændt, skal du vælge Shutdown (Luk ned) i hindbærmenuen, vente på at den slukkes og tage stikket ud.

I de fleste tilfælde vil det medfølgende båndkabel allerede være forbundet til Camera Module eller HQ Camera; hvis det ikke er det, skal du vende kameraet på hovedet, så sensoren er i bunden og se efter et fladt plaststik. Sæt forsigtigt fingerneglene under de udstikkende kanter, og træk udad, indtil stikket er trukket delvist ud. Skub båndkablet med sølvkanterne nedad og med den blå plastik opad, under den flap, du lige har trukket ud, og skub derefter flappen forsigtigt tilbage på plads med et klik (**Figur 8-1**); det betyder ikke noget, hvilken ende af kablet du bruger. Hvis kablet er installeret korrekt, vil det være lige og ikke komme ud, hvis du trækker forsigtigt i det; i modsat fald skal du trække flappen ud og prøve igen.



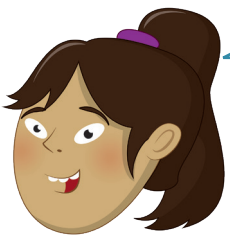
▲ **Figur 8-1:** Tilslutning af båndkablet til Camera Module

Installer den anden ende af kablet på samme måde. Find kamerastikket (eller CSI-stikket) på Raspberry Pi, og træk flappen forsigtigt opad. Hvis din Raspberry Pi er installeret i et kabinet, er det muligvis nemmere, hvis du fjerner det først. Med Raspberry Pi placeret, så HDMI-stikket vender mod dig, skal du skubbe båndkablet ind, så sølvkanterne er til venstre og den blå plastik til højre (**Figur 8-2**), og derefter forsigtigt skubbe flappen på plads igen. Hvis kablet er installeret korrekt, vil det være lige og ikke komme ud, hvis du trækker forsigtigt i det; i modsat fald skal du trække flappen ud og prøve igen.



Figur 8-2: Tilslutning af båndkablet til kamera-/CSI-stikket på Raspberry Pi

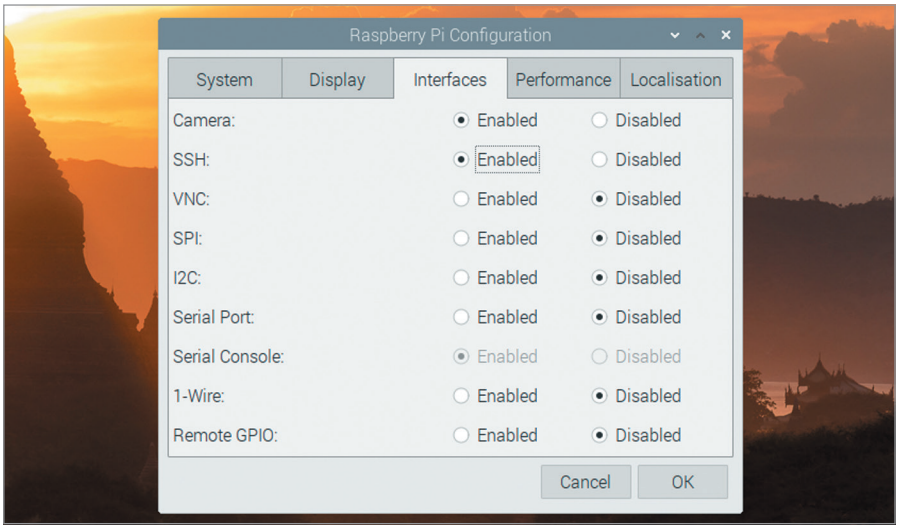
Med Camera Module følger et lille stykke blåt plastik, der dækker objektivet for at beskytte det mod ridser under fremstilling, forsendelse og installation. Find den lille plastikflap, og træk den forsigtigt af objektivet for at gøre kameraet klar til brug.



JUSTERING AF FOKUS

Camera Module leveres normalt med et lille plasthjul til justering af objektivets fokus. Selvom det fabriksindstillede fokus normalt er perfekt, kan du skubbe hjulet over linsen og forsigtigt dreje det for at justere fokus manuelt, hvis du bruger dit kamera til at optage meget tæt på. For HQ Camera-fokusering, se **Bilag F**.

Tilslut igen strømmen til Raspberry Pi, og lad den indlæse Raspberry Pi OS. Inden du kan bruge kameraet, skal du fortælle Raspberry Pi, at det er tilsluttet: Åbn hindbærikonemenuen, vælg kategorien Preferences (Indstillinger) og klik på Raspberry Pi Configuration (Raspberry Pi-konfiguration). Når værktøjet er indlæst, skal du klikke på fanen Interfaces (Grænseflader), finde kameraet på listen og klikke på den runde alternativknap til venstre for Enabled (Aktiveret) for at slå det til (**Figur 8-3**, på næste side). Klik på OK, hvorefter værktøjet vil bede dig om at genstarte din Raspberry Pi. Gør det, så er dit kamera klar til brug!

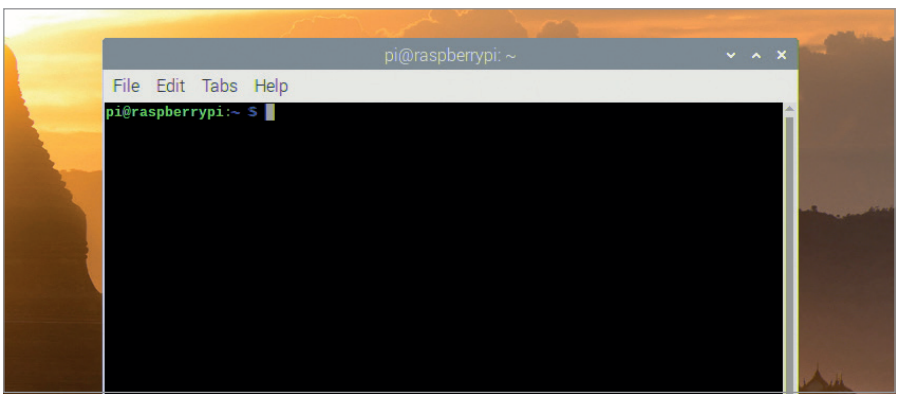


▲ **Figur 8-3:** Du skal aktivere kameraet i Raspberry Pi Configuration (Raspberry Pi-konfiguration)

Test af kameraet

For at bekræfte, at dit Camera Module eller HQ Camera er korrekt installeret, og at du har aktiveret grænsefladen i Raspberry Pi Configuration Tool (værktøjet til Raspberry Pi-konfiguration), kan du bruge værktøjet **raspistill**. Sammen med **raspivid** til videoer er det designet til at tage billeder fra kameraet ved hjælp af Raspberry Pi's *kommandolinjegrænseflade (CLI)*.

I modsætning til de programmer, du hidtil har brugt, finder du ikke raspistill i menuen. Du skal i stedet klikke på hindbærikonet for at indlæse menuen, vælge kategorien Accessories (Tilbehør) og derefter klikke på Terminal. Et sort vindue med grøn og blå skrift i vises (**Figur 8-4**): Dette er *terminalen*, som giver dig adgang til kommandolinjegrænsefladen.



▲ **Figur 8-4:** Åbn et terminalvindue for at indtaste kommandoer

For at teste kameraet skal du skrive følgende i terminalen:

```
raspistill -o test.jpg
```

Så snart du trykker på **ENTER**, vises et stort billede af det, som kameraet ser, på skærmen (**Figur 8-5**). Dette kaldes *live forhåndsvisning*, og medmindre du fortæller raspistill andet, varer det i 5 sekunder. Når disse 5 sekunder er gået, optager kameraet et enkelt stillbillede og gemmer det i mappen Home (Hjem) under navnet **test.jpg**. Hvis du vil tage endnu et billede, skal du skrive den samme kommando igen – men sørg for at ændre outputfilnavnet efter **`-o`**, ellers overskriver du dit første billede!

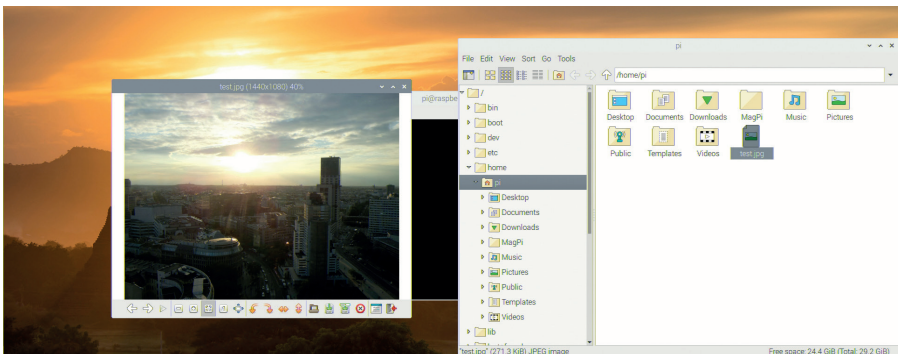
▲ **Figur 8-5:** Live forhåndsvisning fra kameraet

Hvis live forhåndsvisning var på hovedet, skal du fortælle raspistill, at kameraet er roteret. Camera Module er designet til at båndkablet skal komme ud af den nederste kant; hvis det kommer ud af siderne eller toppen, som med noget tilbehør fra kameraer fra tredjeparter, kan du rotere billedet 90, 180 eller 270 grader ved hjælp af parameteren **`-rot`**. Hvis kameraet er monteret, så kablet kommer ud af toppen, skal du blot bruge følgende kommando:

```
raspistill -rot 180 -o test.jpg
```

Hvis båndkablet kommer ud af højre kant, skal du bruge en rotationsværdi på 90 grader; hvis det kommer ud af venstre kant, skal du bruge 270 grader. Hvis din oprindelige optagelse var i den forkerte vinkel, skal du prøve en anden ved at rette den ved hjælp af parameteren **`-rot`**.

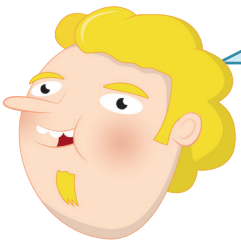
For at se dit billede skal du åbne File Manager (Filhåndtering) fra kategorien Accessories (Tilbehør) i hindbærmenuen: det billede, du har taget, kaldet **test.jpg**, vil være i din **home/pi**-mappe. Find det på listen over filer, og dobbeltklik derefter på det for at indlæse det i en billedfremviser (**Figur 8-6**). Du kan også vedhæfte billedet til e-mails, uploade det til websteder via browseren eller trække det til en ekstern lagerenhed.



▲ **Figur 8-6:** Åbning af det optagne billede

Introduktion til picamera

Den mest fleksible måde at kontrollere Camera Module eller HQ Camera på er at bruge Python via det praktiske picamera-bibliotek. Dette giver dig fuld kontrol over kameraets forhåndsvisning, billede og videooptagelsesmuligheder og giver dig mulighed for at integrere dem i dine egne programmer – endda kombinere dem med programmer, der bruger GPIO-modulet via GPIO Zero-biblioteket!



PYTHON PROGRAMMERING

Projekterne i dette kapitel forudsætter erfaring med Python-programmeringssproget, Thonny IDE og Raspberry Pi's GPIO-stik. Hvis du ikke allerede har gjort det, skal du først gennemgå projekterne i **Kapitel 5, Programmering med Python** og **Kapitel 6, Physical computing med Scratch og Python**!

Luk terminalen, hvis den stadig er åben, ved at klikke på X øverst til højre i vinduet og indlæse Thonny fra kategorien Programming (Programmering) i hindbærmenuen. Gem dit nye projekt som **Kamera**, og start derefter med at importere de biblioteker, dit program har brug for, ved at skrive følgende i scriptområdet:

```
from picamera import PiCamera
from time import sleep
camera = PiCamera()
```

Den sidste linje giver dig mulighed for at styre Camera Module eller HQ Camera ved hjælp af **camera**-funktionen. For at starte skal du skrive følgende:

```
camera.start_preview()
sleep(10)
camera.stop_preview()
```

Klik på Run (Kør), så forsvinder dit skrivebord. I stedet for vil du se en forhåndsvisning i fuld skærm af, hvad kameraet kan se (**Figur 8-7**). Prøv at flytte den rundt, eller vink din hånd foran objektivet, så vil du se billedet på skærmen skifte, så det matcher. Efter 10 sekunder lukkes forhåndsvisningen, og dit program afsluttes – men i modsætning til forhåndsvisningen fra raspistill vil der ikke blive gemt noget billede bagefter.

▲ **Figur 8-7:** En live forhåndsvisning i fuld skærm af kameravisningen

Hvis din forhåndsvisning var den forkerte vej op, kan du rotere billedet for at få det til den rigtige vej op igen. Lige under linjen **camera = PiCamera()** skal du skrive:

```
camera.rotation = 180
```

Hvis forhåndsvisningen var på hovedet, vil denne linje få tingene til at se rigtigt ud igen. Som med raspistill giver **camera.rotation** dig mulighed for at rotere billedet 90, 180 eller 270 grader, afhængigt af om kablet kommer ud fra højre, øverste eller venstre side af Camera Module. Husk at bruge **camera.rotation** i starten af ethvert program, som du skriver, for at undgå at tage billeder eller video, der er den forkerte vej op!

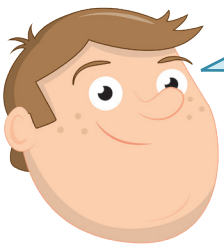
Optagelse af stillbilleder

For at tage et billede i stedet for bare at vise en live forhåndsvisning skal dit program ændres. Start med at reducere forsinkelsen for forhåndsvisning: Find linjen **`sleep(10)`**, og rediger den, så der står:

```
sleep(5)
```

Direkte under denne linje skal du tilføje følgende:

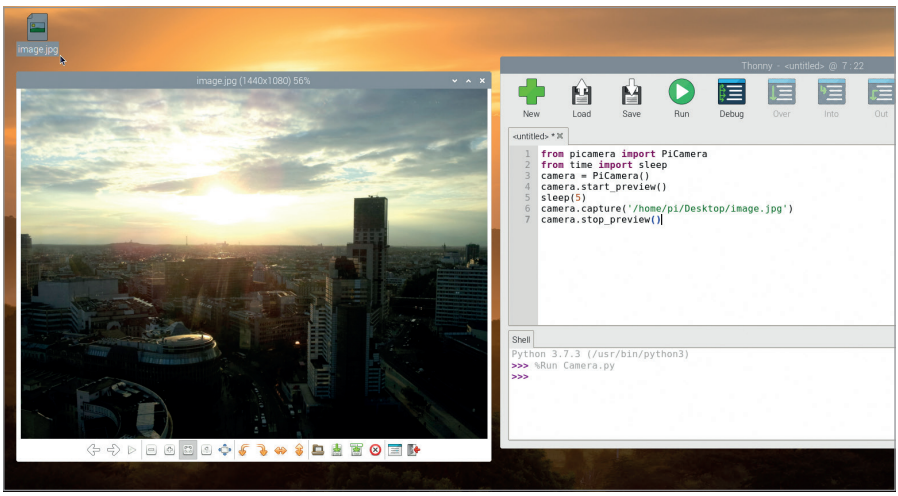
```
camera.capture('/home/pi/Desktop/image.jpg')
```



TID TIL JUSTERING

Når kameraet er i forhåndsvisningstilstand, analyserer det videoen for at se, om det har brug for at justere dets indstillinger for at få den bedste kvalitet. Du kan se dette, hvis du er i et meget mørkt eller meget lyst miljø, hvor forhåndsvisning først er umulig at se og derefter bliver klarere. For at give kameraet tid til at justere skal du altid tilføje mindst 2 sekunders forhåndsvisning til dit program, før du tager et billede.

`camera.capture`-funktionen beder Python om at gemme et stillbillede, og det skal ikke kun vide, hvad billedet skal kaldes, men også i hvilken mappe det skal gemmes. I dette eksempel gemmer du det på skrivebordet – find det ved at se lige under papirkurven. Hvis Thonny-vinduet er i vejen, skal du bare klikke og trække i titellinjen for at flytte det. Dobbeltklik på filen for at se det billede, du har taget (**Figur 8-8**). Tillykke: Du har programmeret et kamera!



▲ **Figur 8-8:** Åbning af det optagne billede

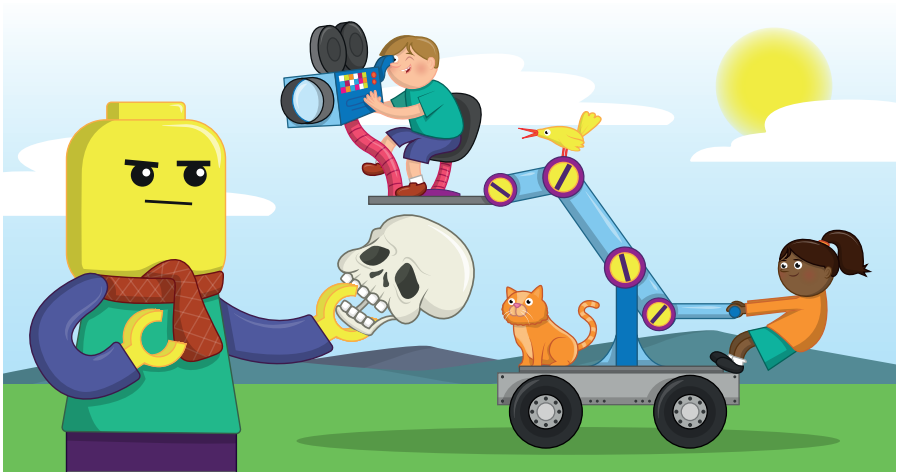
Optagelse af video

Ud over at tage stillbilleder kan du optage video. Slet alt mellem linjerne **`camera.start_preview()`** og **`camera.stop_preview()`**, skriv derefter følgende under **`camera.start_preview():`**

```
camera.start_recording('/home/pi/Desktop/video.h264')
sleep(10)
camera.stop_recording()
```

Kameraeksemplet vises som før, men denne gang optages det også og gemmes som en fil på skrivebordet. Vent de 10 sekunder, som du har bedt Python om at gå i dvale – lav måske en lille dans foran kameraet for at gøre videoen interessant – så når forhåndsvisningen er færdig, finder du din videofil på skrivebordet.

For at afspille videoen skal du blot dobbeltklikke på filen **video.h264** på skrivebordet. Videoen begynder at afspilles – og hvis du dansede, afspilles det for dig! Når videoen er færdig, afsluttes afspillersoftwaren med en besked i terminalen. Tillykke: Du kan nu optage video ved hjælp af dit Raspberry Pi Camera Module eller HQ Camera!



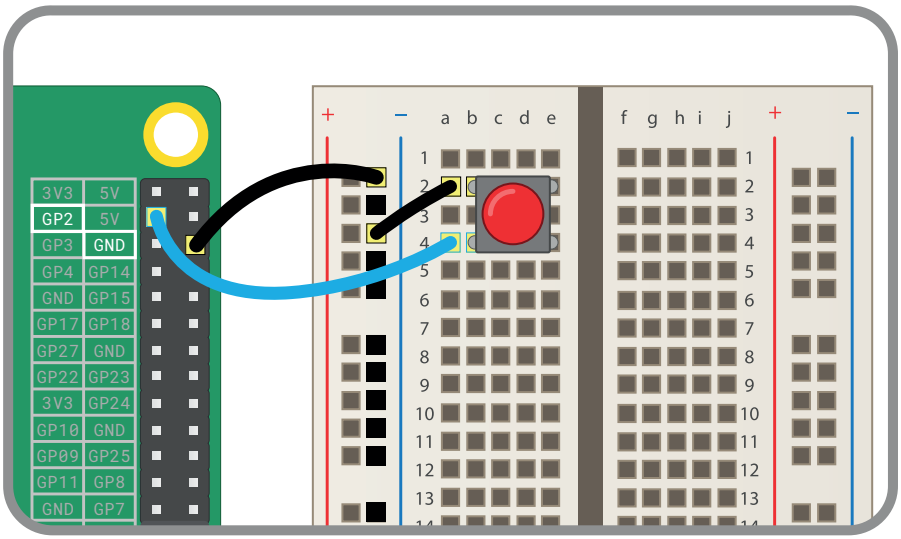
Stop-motion-animation med trykknap

Ved hjælp af det, du har lært i dette kapitel, og din viden om, hvordan du forbinder hardware til Raspberry Pi's GPIO-header fra **Kapitel 6**, **Physical Computing**, er det tid til at bygge noget specielt: Dit helt eget stop-motion-animationsstudie.

Stop-motion animation er processen med at tage masser af stillbilleder, som f.eks. modelbiler eller actionfigurer, og flytte objekterne en smule mellem hvert billede. Selvom objekterne aldrig bevæger sig på nogen af billederne, vil det, hvis du viser dem efter hinanden hurtigt nok, se ud som om de bevæger sig så hurtigt eller så langsomt, som du vil!

Til dette projekt skal vi bruge en trykknapkontakt, et breadboard, M2M- (han- til hanstik) og M2F-jumperledninger (han- til hunstik). Hvis du ikke har et breadboard, kan du tilslutte kontakten ved hjælp af hun-til-hun-kabler (F2F) i stedet, men det bliver sværere at trykke på. Hvis du har brug for at genopfriske din viden om disse komponenter, skal du gå til **Kapitel 6, Physical computing med Scratch og Python**. Du skal også bruge objekter for at kunne animere: Disse kan være alt fra en klat ler til en legetøjsbil eller en actionfigur.

Start med at oprette kredsløbet: Tilføj trykknappen til dit breadboard, og forbind derefter jordskinnen på dit breadboard til et jordben på Raspberry Pi (markeret GND på **Figur 8-9**) med en jumberledning (han- til hunstik). Brug en jumperledning (han- til hanstik) til at forbinde det ene ben af kontakten til jordskinnen på dit breadboard, derefter en jumperledning (han- til hunstik) til at forbinde det andet ben på kontakten til GPIO-ben 2 (markeret GP2 på **Figur 8-9**).



▲ **Figur 8-9:** Ledningsdiagram til tilslutning af en trykknap til GPIO-benene

Opret et nyt projekt i Thonny, og gem det som **Stop Motion**. Start med at importere og indstille de biblioteker, du har brug for til brug af kameraet og GPIO-porten:

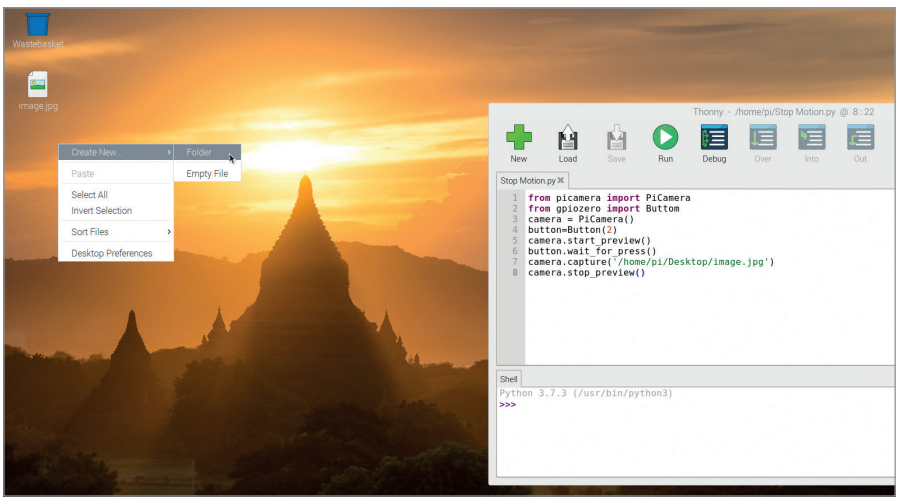
```
from picamera import PiCamera
from gpiozero import Button
camera = PiCamera()
button = Button(2)
```

Skriv derefter følgende:

```
camera.start_preview()
button.wait_for_press()
camera.capture('/home/pi/Desktop/image.jpg')
camera.stop_preview()
```

Klik på Run (Kør), så får du vist en forhåndsvisning af, hvad dit kamera peger på. Forhåndsvisningen forbliver på skærmen, indtil du trykker på trykknappen: Tryk på den nu, og forhåndsvisningen lukkes, når dit program gemmer et billede på skrivebordet. Find billedet, kaldet **image.jpg**, og dobbeltklik for at åbne det og bekræfte, at programmet fungerer.

Stop-motion-animation indebærer at skabe masser af stillbilleder for at give indtryk af bevægelse, når de alle er samlet. At have alle disse individuelle billeder på dit skrivebord vil gøre det rodet, så du har brug for en mappe til at gemme dem i. Højreklik et vilkårligt sted på skrivebordet, der ikke allerede har en fil eller et ikon, og vælg derefter Create New (Opret ny) og Folder (Mappe) (**Figur 8-10**). Kald mappen **animation** med små bogstaver, og klik derefter på OK-knappen.



▲ **Figur 8-10:** Opret en ny mappe til dine optagne billeder

At skulle genstarte dit program hver gang du tager et billede til din animation er ikke særlig godt, så du skal ændre det til at køre i en løkke. I modsætning til de tidligere løkker, du har oprettet, har denne dog brug for en måde at lukke pænt på – ellers kan du ikke se skrivebordet mere, hvis du stopper programmet, mens kameraeksemplet vises! For at gøre dette skal du bruge to specielle instruktioner: `try` og `except`.

Start med at slette alt efter **camera.start_preview(),**, og skriv derefter:

```
frame = 1
```

Dette skaber en ny variabel, **frame**, som dit program vil bruge til at gemme det aktuelle billednummer. Du vil snart komme til at bruge dette til at sikre, at du gemmer en ny fil hver gang. Uden det vil du bare gemme oven på dit første billede hver gang du trykker på knappen!

Indstil derefter din løkke ved at skrive:

```
while True:
    try:
```

Den nye instruktion **try** beder Python køre den kode, der er indeni – hvilket bliver koden til at tage billeder. Skriv:

```
button.wait_for_press()
camera.capture('/home/pi/Desktop/animation/frame%03d.jpg' % frame)
frame += 1
```

Der er et par smarte tricks i disse tre kodelinjer. Den første er i billednavnet: Ved hjælp af **%03d** bedes Python om at tage et tal og tilføje så mange nuller foran, som det har brug for for at gøre det tre cifre langt. Således bliver "1" til "001", "2" bliver til "002", og "10" bliver til "010". Du har brug for dette i dit program for at holde dine filer i den rigtige rækkefølge og for at sikre, at du ikke overskriver en fil, som du allerede har gemt.

% frame i slutningen af denne linje fortæller Python at bruge nummeret på billedvariablen i filnavnet. For at sikre, at hver fil er unik, forøger den sidste linje – **frame += 1** – frame-variablen med 1. Første gang du trykker på knappen, øges **frame** fra 1 til 2; næste gang fra 2 til 3; og så videre.

I øjeblikket lukker din kode dog ikke korrekt, når du er færdig med at tage billeder. For at få det til at ske, har du brug for en **except** for din **try**. Skriv følgende, og husk at fjerne et niveau af indrykning på første linje, så Python ved, at det ikke er en del af afsnittet **try**:

```
    except KeyboardInterrupt:
        camera.stop_preview()
        break
```

Dit færdige program vil se sådan ud:

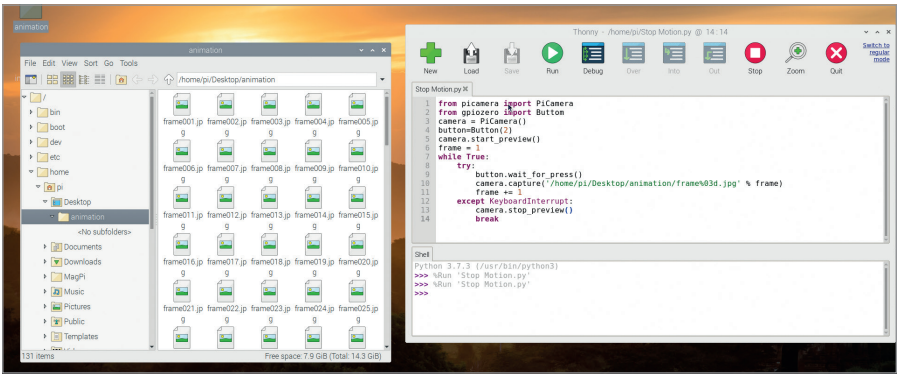
```
from picamera import PiCamera
from time import sleep
from gpiozero import Button
camera = PiCamera()
button = Button(2)
camera.start_preview()
frame = 1
while True:
    try:
        button.wait_for_press()
        camera.capture('/home/pi/Desktop/animation/frame%03d.jpg'
% frame)
        frame += 1
    except KeyboardInterrupt:
        camera.stop_preview()
        break
```

Prøv at klikke på Run (Kør), men i stedet for at trykke på knappen så tryk på tasterne **CTRL** og **C** på tastaturet. Du behøver ikke at trykke på begge taster på samme tid: Bare hold **CTRL** nede, tryk på og slip **C**, og slip derefter **CTRL**. Disse to nøgler fungerer som en afbrydelse, der beder Python om at stoppe, hvad den laver. Uden linjen `except KeyboardInterrupt:` ville Python straks afslutte og lade kameraets forhåndsvisning blokere din skærm; med den linje på plads vil Python dog køre den kode der er indeni – i dette tilfælde kode, der fortæller at kameraets forhåndsvisning skal stoppes, og der skal afsluttes.

Nu er du klar til at begynde at optage din stop-motion-animation! Placer Camera Module eller HQ Camera, hvor det kan se de objekter, du vil animere, og sørg for, at det ikke bevæger sig – hvis kameraet bevæger sig, ødelægger det effekten. Placer dine objekter i deres startpositioner, og klik derefter på Run (Kør) for at starte dit program. Kontroller, at alt ser godt ud i forhåndsvisningen, og tryk på trykknappen for at optage dit første billede.

Flyt objekterne lidt – jo mindre du flytter dem mellem billederne, jo jævnere bliver den færdige animation – og tryk på trykknappen igen for at optage endnu et billede. Fortsæt med at gøre dette, indtil din animation er færdig: Jo flere billeder du optager, jo længere bliver din animation.

Når du er færdig, skal du trykke på **CTRL+C** for at lukke dit program og derefter dobbeltklikke på mappen **animation** på skrivebordet for at se de billeder, du har taget (**Figur 8-11**, på næste side). Dobbeltklik på et hvilket som helst billede for at åbne det og se det mere detaljeret!



▲ **Figur 8-11:** De optagne billeder i mappen

I øjeblikket er det eneste du har en mappe fuld af stillbilleder. For at oprette en animation skal du gøre dem til en video. For at gøre det skal du klikke på hindbærikonet for at indlæse menuen, vælge Accessories (Tilbehør) og derefter klikke på Terminal. Dette åbner en *kommandolinjegrænseflade*, der er beskrevet mere detaljeret i **Bilag C**, som giver dig mulighed for at skrive kommandoer til Raspberry Pi. Når terminalen indlæses, skal du starte med at skifte til den mappe, som du oprettede, ved at skrive:

```
cd Desktop/animation
```

Det er vigtigt, at "D" i "Desktop" er med store bogstaver; i Raspberry Pi OS er der *forskel på store og små bogstaver*, hvilket betyder, at hvis du ikke skriver en kommando eller et mappenavn nøjagtigt som det oprindeligt blev skrevet, fungerer det ikke! Når du har skiftet mapper, skal du skrive følgende:

```
ffmpeg -i frame%03d.jpg -r 10 animation.h264
```

Dette bruger et program kaldet **ffmpeg** til at tage stillbillederne i mappen og konvertere dem til en video kaldet **animation.h264**. (Bemærk: Hvis ffmpeg ikke er tilgængelig, kan du installere det med `sudo apt-get install ffmpeg`.) Afhængigt af hvor mange stillbilleder du har taget, kan denne proces tage et par minutter; du ved, at den er færdig, når du ser terminalprompten igen.

For at afspille videoen skal du finde filen **animation.h264** i mappen **animation** og dobbeltklikke på den for at åbne den. Alternativt kan du afspille den fra terminalen ved at skrive følgende:

```
omxplayer animation.h264
```

Når videoen er indlæst, kan du se din stop-motion-animation komme til live. Tillykke: Du har forvandlet din Raspberry Pi til et avanceret animationsstudie!

Hvis din animation bevæger sig for hurtigt eller for langsomt, skal du ændre **`-r 10`**-delen af kommandoen **`ffmpeg`** til et lavere eller højere tal: dette er billedhastigheden, eller hvor mange stillbilleder der er i et sekund af videoen. Et lavere tal får din animation til at køre langsommere, men ser mindre jævn ud; et højere tal ser jævnere ud, men får animationen til at køre hurtigere.

Hvis du vil gemme din video, skal du sørge for at trække og slippe den fra skrivebordet til mappen Videoer; ellers overskriver du filen, næste gang du kører dit program!

Avancerede kameraindstillinger

Hvis du har brug for mere kontrol over Raspberry Pi Camera Module eller HQ Camera, kan du bruge Python picamera-biblioteket til at få adgang til forskellige indstillinger. Disse indstillinger sammen med deres standardværdier er beskrevet nedenfor til inkludering i dine egne programmer.

`camera.awb_mode = 'auto'`

Dette indstiller kameraets automatiske hvidbalancetilstand og kan indstilles til en af følgende tilstande: **`off`**, **`auto`**, **`sunlight`**, **`cloudy`**, **`shade`**, **`tungsten`**, **`fluorescent`**, **`incandescent`**, **`flash`** eller **`horizon`**. Hvis du synes, at dine billeder og videoer ser lidt blå eller gule ud, kan du prøve en anden tilstand.

`camera.brightness = 50`

Dette indstiller kamerabilledets lysstyrke, fra mørkest ved 0 til lysest ved 100.

`camera.color_effects = None`

Dette ændrer den farveeffekt, som kameraet i øjeblikket bruger. Normalt skal denne indstilling ikke røres, men hvis du angiver et par tal, kan du ændre den måde, kameraet optager farve på: Prøv **`(128, 128)`**) for at oprette et sort-hvidt billede.

`camera.contrast = 0`

Dette indstiller billedets kontrast. Et højere tal får det hele til at se mere dramatisk og skarpt ud; et lavere tal får tingene til at se mere udvaskede ud. Du kan bruge et hvilket som helst tal mellem -100 for minimal kontrast og 100 for maksimal kontrast.

`camera.crop = (0.0, 0.0, 1.0, 1.0)`

Dette giver dig mulighed for at beskære billedet, skære dele fra siden og toppen for kun at medtage den del af billedet, som du har brug for. Tallene repræsenterer X-koordinat, Y-koordinat, bredde og højde og optager som standard hele billedet. Prøv at reducere de to sidste tal – 0,5 og 0,5 er et godt udgangspunkt – for at se hvilken effekt denne indstilling har.

`camera.exposure_compensation = 0`

Dette indstiller kameraets *eksponeringskompensation*, så du manuelt kan kontrollere, hvor meget lys der fanges for hvert billede. I modsætning til at ændre lysstyrkeindstillingen styrer dette faktisk selve kameraet. Gyldige værdier spænder fra -25 for et meget mørkt billede til 25 for et meget lyst billede.

`camera.exposure_mode = 'auto'`

Dette indstiller *eksponeringstilstand* eller den logik, som Camera Module/HQ Camera bruger til at bestemme, hvordan et billede skal eksponeres. Mulige tilstande er: **`off`**, **`auto`**, **`night`**, **`backlight`**, **`spotlight`**, **`sports`**, **`snow`**, **`beach`**, **`verylong`**, **`fixedfps`**, **`antishake`** og **`fireworks`**.

`camera.framerate = 30`

Dette indstiller antallet af billeder, der er taget for at oprette en video pr. sekund eller *billedhastigheden*. En højere billedhastighed giver en jævnere video, men optager mere lagerplads. Højere billedhastigheder kræver en lavere opløsning, som du kan indstille via **`camera.resolution`**.

`camera.hflip = False`

Dette vender kamerabilledet hen over den vandrette akse eller X-aksen, når den er indstillet til **`True`**.

`camera.image_effect = 'none'`

Dette gælder en af en række billedeffekter til videoen, som vil være synlig i forhåndsvisningen såvel som de gemte billeder og videoer. Mulige effekter er: **`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 ændrer kameraets ISO-indstilling, hvilket påvirker, hvor følsom det er for lys. Som standard justerer kameraet dette automatisk afhængigt af det tilgængelige lys. Du kan selv indstille ISO ved hjælp af en af følgende værdier: 100, 200, 320, 400, 500, 640, 800. Jo højere ISO, jo bedre fungerer kameraet i omgivelser med svagt lys, men billedet eller videoen bliver mere kornet.

`camera.meter_mode = 'average'`

Dette styrer, hvordan kameraet bestemmer mængden af tilgængeligt lys, når det indstiller eksponeringen. Standard er gennemsnittet af den tilgængelige mængde lys i hele billedet; andre mulige tilstande er **`backlit`**, **`matrix`** og **`spot`**.

`camera.resolution = (1920, 1080)`

Dette indstiller opløsningen på det optagede billede eller video, repræsenteret af to tal for bredde og højde. Lavere opløsninger optager mindre lagerplads og giver dig mulighed for at bruge en højere billedhastighed; højere opløsninger giver bedre kvalitet, men optager mere lagerplads.

`camera.rotation = 0`
Dette styrer billedets rotation fra 0 grader gennem 90, 180 og 270 grader. Brug dette, hvis du ikke kan placere kameraet, så båndkablet kommer ud af bunden.

`camera.saturation = 0`
Dette styrer mætning af billedet, eller hvor levende farverne er. Mulige værdier varierer fra -100 til 100.

`camera.sharpness = 0`
Dette styrer billedets skarphed. Mulige værdier varierer fra -100 til 100.

`camera.shutter_speed = 0`
Dette styrer, hvor hurtigt lukkeren åbnes og lukkes, når du tager billeder og videoer. Du kan indstille lukkerhastigheden manuelt i mikrosekunder, hvor længere lukkerhastigheder fungerer bedre i svagere lys, og hurtigere lukkerhastigheder fungerer i lysere lys. Du bør normalt bare bruge den automatiske standardindstilling.

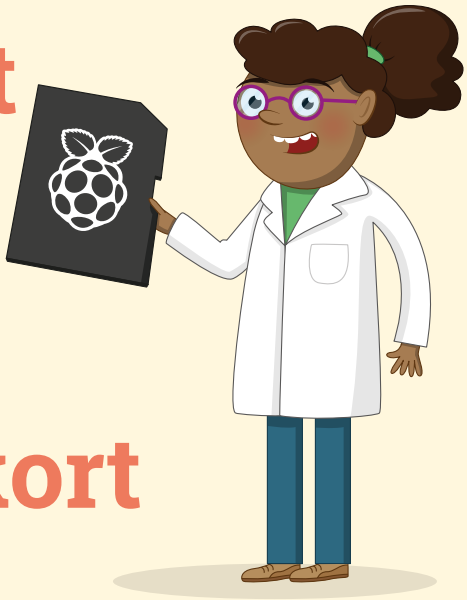
`camera.vflip = False`
Dette vender kamerabilledet hen over den vandrette akse eller Y-aksen, når den er indstillet til **`True`**.

`camera.video_stabilization = False`
Når det er indstillet til **`True`**, aktiverer det videostabilisering. Du har kun brug for dette, hvis Camera Module eller HQ Camera bevæger sig, mens du optager, f.eks. hvis det er fastgjort til en robot eller bæres, for at reducere rystelserne i den optagede video.

Flere oplysninger om disse indstillinger samt yderligere indstillinger, der ikke er dokumenteret her, findes i **picamera.readthedocs.io**.

Bilag A

Installer et operativ -system på et microSD-kort



Du kan købe microSD-kort med NOOBS (New Out of the Box Software), der er forudinstalleret på dem, fra alle gode Raspberry Pi-forhandlere, så du nemt kan installere Raspberry Pi OS (tidligere kendt som Raspbian) til din Raspberry Pi. Alternativt kan du med følgende instruktioner bruge Raspberry Pi Imager til også at installere et operativsystem manuelt på dit eget tomme (eller genbrugte) microSD-kort.

> **ADVARSEL!**
>
> Hvis du har købt et microSD-kort med NOOBS allerede forudinstalleret, behøver du ikke gøre andet end at slutte det til din Raspberry Pi. Disse instruktioner er til tomme microSD-kort eller til tidligere anvendte kort, som du vil installere et nyt operativsystem på. Hvis du udfører disse instruktioner på et microSD-kort med filer på det, mister du disse filer, så sørg for at du først har sikkerhedskopieret det, som ligger på kortet!

Download Raspberry Pi Imager

Raspberry Pi OS, der er baseret på Debian, er det officielle operativsystem til Raspberry Pi. Den nemmeste måde at installere Raspberry Pi OS på et microSD-kort til din Raspberry Pi er at bruge Raspberry Pi Imager-værktøjet, der kan downloades fra **rpf.io/downloads**. Bemærk, at denne metode erstatter installation af dit operativsystem via NOOBS, selvom sidstnævnte stadig er tilgængelig fra den samme downloadside.

Raspberry Pi Imager-programmet er tilgængeligt til computere med Windows-, macOS- og Ubuntu Linux, så vælg den relevante version til dit system.

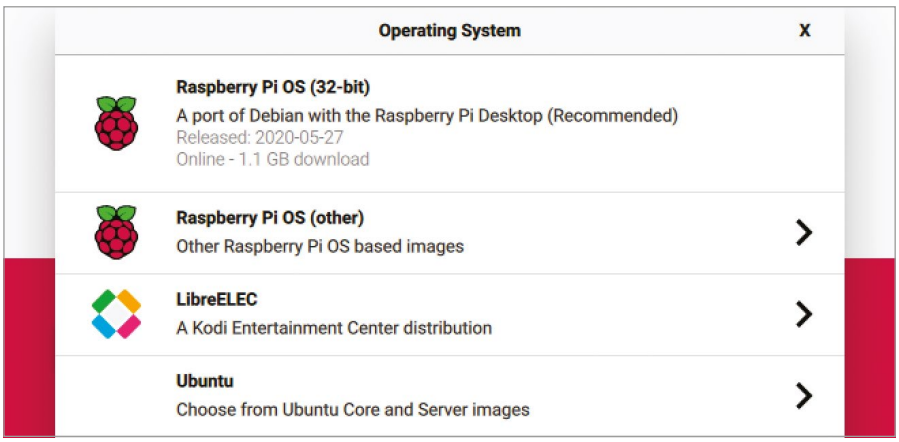
På macOS skal du dobbeltklikke på den downloadede DMG-fil. Du skal muligvis ændre din indstilling for sikkerhed og anonymitet for at tillade apps, der er downloadet fra "App Store og identificerede udviklere", for at gøre det muligt at køre dem. Du kan derefter bare trække Raspberry Pi Imager-ikonet til mappen Programmer.

På en Windows-pc skal du dobbeltklikke på den downloadede EXE-fil. Når du bliver bedt om det, skal du vælge knappen "Ja" for at køre filen. Klik derefter på knappen "Installer" for at starte installationen.

Skriv operativsystemet til microSD-kortet

Sæt microSD-kortet i din pc eller Mac-computer: Du skal bruge en microSD-kort USB-adapter, medmindre der er en indbygget kortlæser. Bemærk, at kortet ikke behøver at være forudformateret.

Start Raspberry Pi Imager-programmet. Klik på knappen "Choose OS" (Vælg OS) for at vælge, hvilket operativsystem du vil installere. Den øverste mulighed er standard Raspberry Pi OS – hvis du foretrækker den mindre Lite-version eller den fulde version (med al anbefalet software forudinstalleret), skal du vælge "Raspberry Pi OS (other)". Der er også mulighed for at installere LibreELEC (vælg version til din Raspberry Pi-model) og Ubuntu Core eller Server.



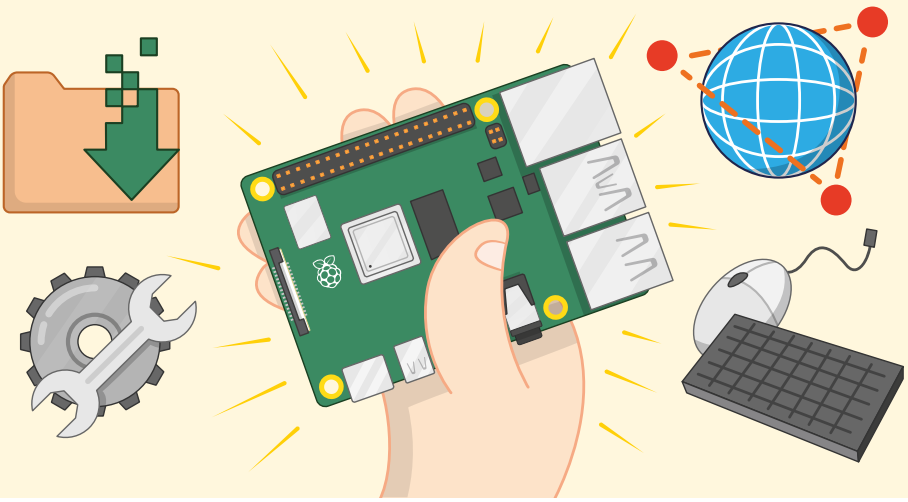
Bemærk: Hvis du vil installere et andet operativsystem, f.eks. Lakka, skal du blot downloade dets image-fil fra det relevante websted og derefter vælge "Use Custom" (Brug brugerdefineret) i Raspberry Pi Imager.

Når et operativsystem er valgt, skal du klikke på knappen "Choose SD card" (Vælg SD-kort) og vælge dit microSD-kort (der vil typisk kun være den ene mulighed).

Endelig skal du klikke på knappen "Write" (Skriv) og vente, mens værktøjet skriver det valgte operativsystem til dit kort og derefter bekræfter det. Når det er færdigt, kan du fjerne microSD-kortet. Du kan derefter indsætte det i din Raspberry Pi og starte den op i det operativsystem, du lige har installeret.

Bilag B

Installation og afinstallation af software



Raspberry Pi OS leveres med et udvalg af populære softwarepakker, der er håndplukket af Raspberry Pi Foundation, men disse er ikke de eneste pakker, der fungerer på en Raspberry Pi. Ved hjælp af følgende instruktioner kan du gennemse yderligere software, installere det og afinstallere det igen – og dermed udvide funktionerne i din Raspberry Pi.

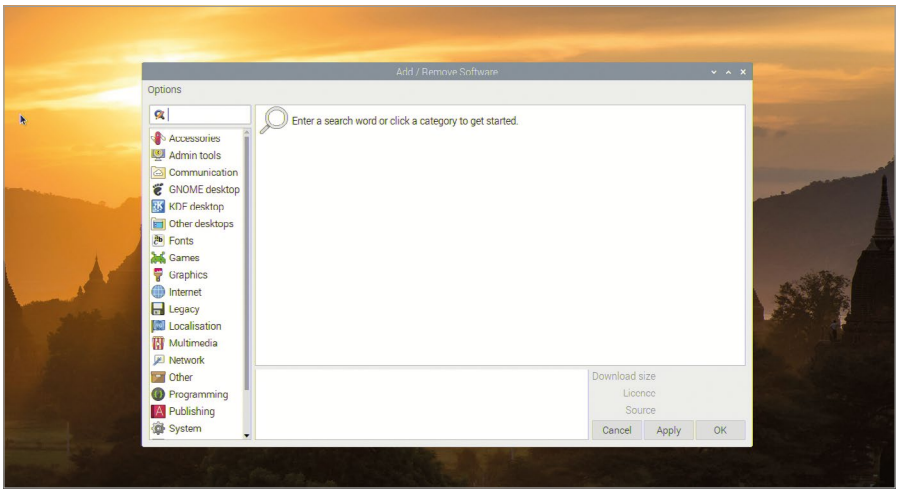
Instruktionerne i dette bilag er et supplement til dem i **Kapitel 3, Sådan bruger du din Raspberry Pi** der forklarer, hvordan du bruger det anbefalede softwareværktøj; hvis du ikke allerede har læst det, skal du gøre det, før du bruger metoderne beskrevet i dette bilag.

KORTKAPACITET

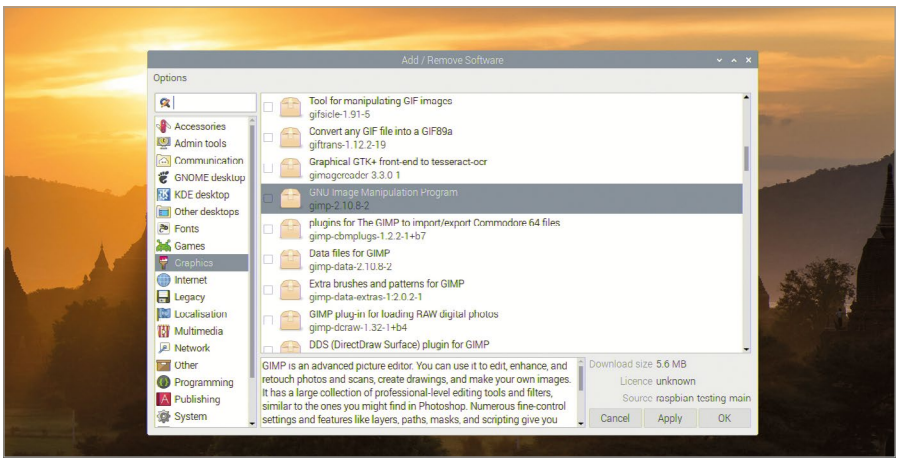
Tilføjelse af mere software til din Raspberry Pi optager plads på dit microSD-kort. Et kort på 16 GB eller mere giver dig mulighed for at installere mere software; for at kontrollere, om det kort, du har til hensigt at bruge, er kompatibelt med Raspberry Pi, skal du gå til **rpf.io/sdcardlist**.

Gennemse tilgængelig software

For at se og søge på listen over tilgængelige softwarepakker til Raspberry Pi OS ved hjælp af det, der kaldes dets *softwarelager*, skal du klikke på hindbærikonet for at indlæse menuen, vælge kategorien Preferences (Indstillinger) og derefter klikke på Add/Remove Software (Tilføj/fjern software). Efter et par sekunder vises værktøjets vindue.



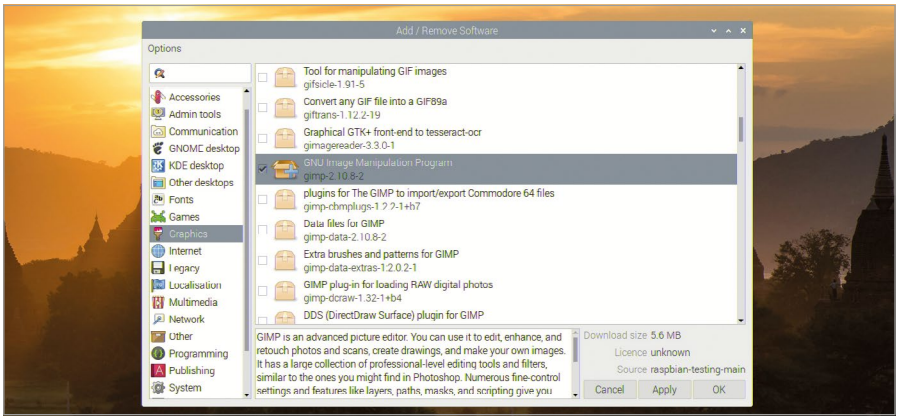
Venstre side af vinduet Add/Remove Software (Tilføj/fjern software) indeholder en liste over kategorier – de samme kategorier, som du finder i hovedmenuen, når du klikker på hindbærikonet. Ved at klikke på en kategori vises en liste over den tilgængelige software i den pågældende kategori. Du kan også indtaste et søgeord i feltet øverst til venstre i vinduet, f.eks. "text editor" eller "game", og få vist en liste med matchende softwarepakker fra enhver kategori. Ved at klikke på en hvilken som helst pakke vises yderligere oplysninger om den nederst i vinduet.



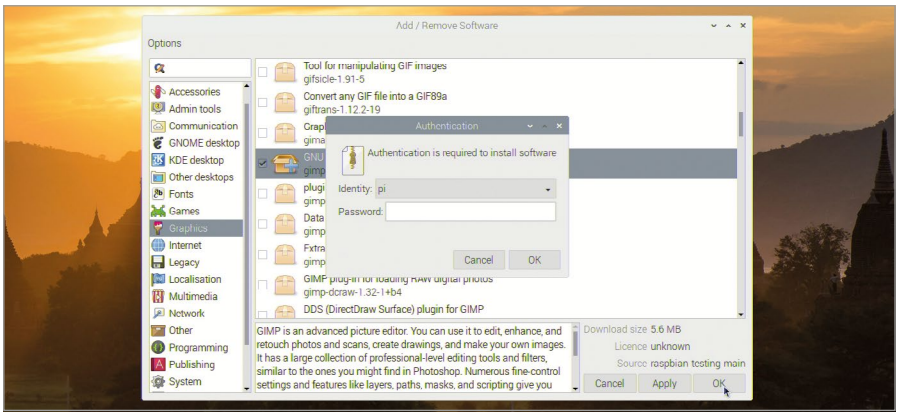
Hvis den kategori, du har valgt, indeholder mange softwarepakker, kan det tage noget tid, før værktøjet Add/Remove Software (Tilføj/fjern software) er færdig med at arbejde sig igennem listen.

Installation af software

For at vælge en pakke til installation skal du markere afkrydsningsfeltet ved siden af den ved at klikke på det. Du kan installere mere end én pakke ad gangen: Bare fortsæt med at klikke for at tilføje flere pakker. Ikonet ved siden af pakken skifter til et åbent felt med et "+" symbol for at bekræfte, at det skal installeres.

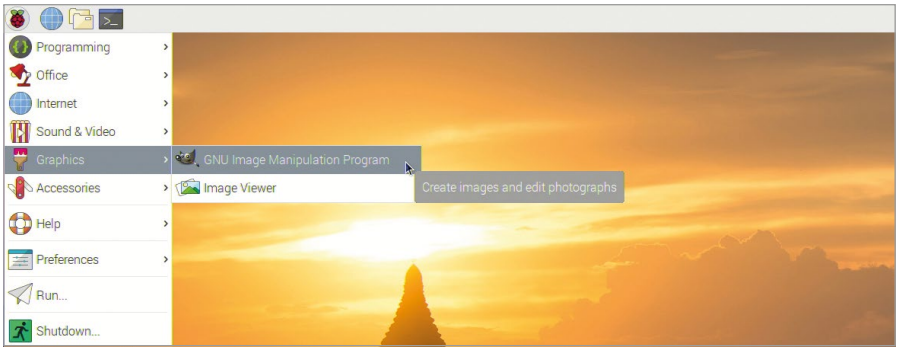


Når du er tilfreds med dine valg, skal du enten klikke på knappen OK eller Apply (Anvend); den eneste forskel er, at OK lukker værktøjet Add/Remove Software (Tilføj/fjern software), når din software er installeret, mens knappen Apply (Anvend) lader det være åbent. Du bliver bedt om at indtaste din adgangskode for at bekræfte din identitet – alle og enhver skal jo ikke kunne tilføje eller fjerne software fra din Raspberry Pi!



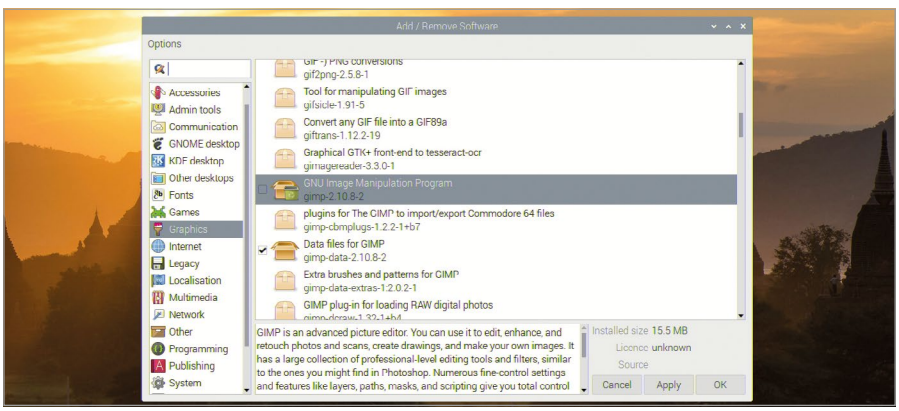
Du kan opleve, at når du installerer en enkelt pakke, installeres andre pakker sammen med den; det kaldes *afhængigheder*, dvs. pakker, som den software, du valgte at installere, har brug for for at kunne fungere – for eksempel lydeffektpakker til et spil eller en database, der kan bruges med en webserver.

Når softwaren er installeret, vil du kunne finde den ved at klikke på hindbærikonet for at indlæse menuen og finde softwarepakkens kategori. Vær opmærksom på, at menukategorien ikke altid er den samme som kategorien fra værktøjet Add/Remove Software (Tilføj/fjern software), mens noget software overhovedet ikke har en punkt i menuen; denne software er kendt som *kommandolinjesoftware* og skal køres i terminalen. For mere information om kommandolinjen og terminalen, henvises til **Bilag C, Kommandolinjegrænsefladen**.



Afinstallation af software

For at vælge en pakke til fjernelse eller *afinstallation* skal du finde den på listen over pakker – søgefunktionen er praktisk her – og fjerne markeringen i afkrydsningsfeltet ved siden af den ved at klikke på det. Du kan afinstallere mere end én pakke ad gangen: Bare fortsæt med at klikke for at fjerne flere pakker. Ikonet ved siden af pakken skifter til et åbent felt ved siden af en lille papirkurv for at bekræfte, at den bliver afinstalleret.



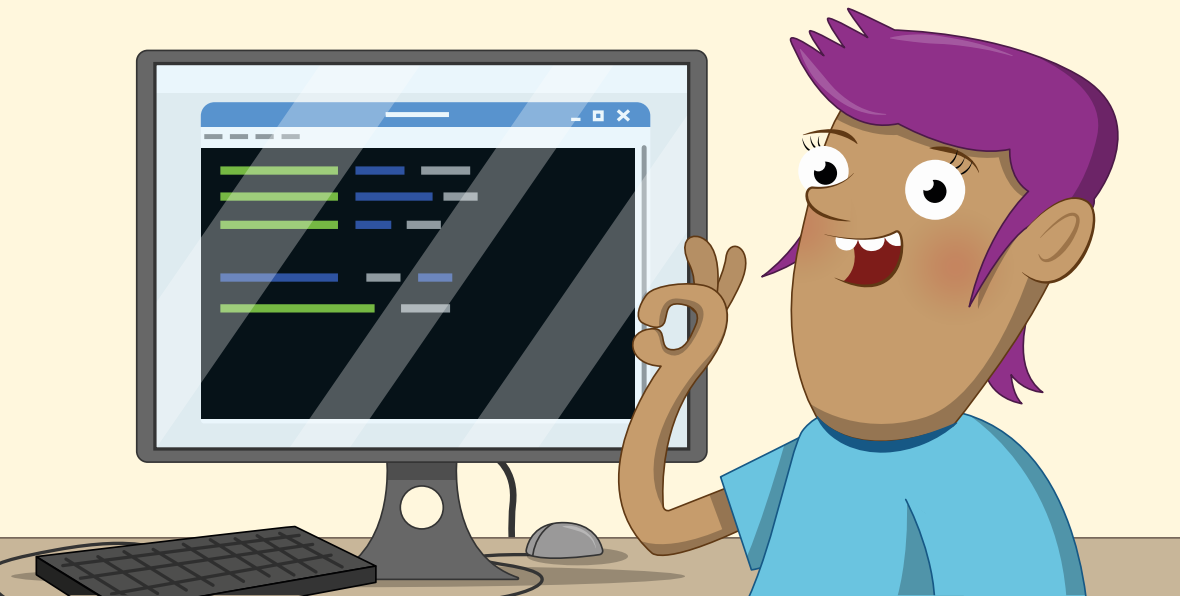
Som før kan du klikke på OK eller Apply (Anvend) for at begynde at afinstallere de valgte softwarepakker. Du bliver bedt om at bekræfte din adgangskode, medmindre du har gjort det inden for de sidste par minutter, og du bliver muligvis også bedt om at bekræfte, at du også vil fjerne eventuelle afhængigheder i forbindelse med din softwarepakke. Når afinstallationen er færdig, forsvinder softwaren fra hindbærikonemenuen, men filer, som du oprettede ved hjælp af softwaren – f.eks. billeder til en grafikpakke eller gemte spil – fjernes ikke.

ADVARSEL!

Al software installeret i Raspberry Pi OS vises i Add/Remove Software (Tilføj/fjern software), inklusive software, der kræves for at din Raspberry Pi kan køre. Det er dog muligt at fjerne så mange pakker, at startsiden ikke længere kan indlæses. For at undgå dette skal du ikke afinstallere ting, medmindre du er helt sikker på at du ikke længere har brug for dem. Hvis det allerede er sket, skal du geninstallere Raspberry Pi OS ved hjælp af instruktionerne i **Kapitel 2, Kom godt i gang med din Raspberry Pi**, eller geninstallere operativsystemet ved at følge vejledningen i **Bilag A**.

Bilag C

Kommandolin -jegrænsefladen



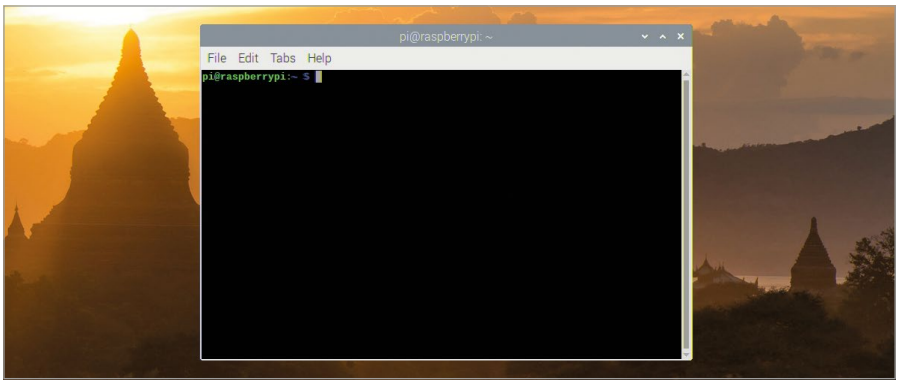
Mens du kan administrere det meste af softwaren på en Raspberry Pi via computeren, er der kun adgang til visse typer software ved hjælp af en tekstbaseret tilstand kendt som *kommandolinjegrænsefladen (CLI)* i et program kaldet Terminal. De fleste brugere vil aldrig for brug for CLI, men for dem, der ønsker at lære mere, giver dette bilag en grundlæggende introduktion.

MERE INFO

Dette bilag er ikke designet til at være en udtømmende guide til Linux-kommandolinjegrænsefladen. For et mere detaljeret kig på brugen af CLI, skal du gå til **rpf.io/terminal** i en webbrowser.

Indlæsning af terminalen

Der er adgang til CLI via Terminal, en softwarepakke, der indlæser det, der teknisk set er kendt som en *virtuel teletype (VTY) terminal*, et navn, der går tilbage til computerens barndom, da brugerne sendte kommandoer via en stor elektromekanisk skrivemaskine snarere end via et tastatur og en skærm. For at indlæse Terminal-pakken skal du klikke på hindbærikonet for at indlæse menuen, vælge kategorien Accessories (Tilbehør) og derefter klikke på LXTerminal.



Terminalvinduet kan trækkes rundt på skrivebordet, ændres, maksimeres og minimeres ligesom ethvert andet vindue. Du kan også gøre teksten større, hvis den er svær at se, eller mindre, hvis du vil have plads til mere i vinduet: Klik på menuen Edit (Rediger), og vælg henholdsvis Zoom In (Zoom ind) eller Zoom Out (Zoom ud), eller hold **CTRL**-tasten på tastaturet nede efterfulgt af **+** eller **-**.

Prompten

Det første du ser i en terminal, er *prompten*, der venter på dine instruktioner. Prompten på en Raspberry Pi, der kører Raspberry Pi OS, ser sådan ud:

```
pi@raspberrypi:~ $
```

Den første del af prompten, **pi**, er dit brugernavn; den anden del efter **@** er værtsnavnet på den computer, du bruger, som er **raspberrypi** som standard. Efter **:** er en tilde, **~**, der er en kort beskrivelse af din hjemmemappe og som repræsenterer din *nuværende arbejdsmappe (CWD)*. Endelig indikerer **$**-symbolet, at din bruger er en *uprivilegeret* bruger, hvilket betyder at der kræves en adgangskode for at kunne udføre opgaver som f.eks. at tilføje eller fjerne software.

Komme rundt

Prøv at skrive følgende, og tryk derefter på **ENTER**-tasten:

```
cd Desktop
```

Du vil se, at prompten ændres til:

```
pi@raspberrypi:~/Desktop $
```

Det viser dig, at din nuværende arbejdsmappe er ændret: Du var i din hjemmemappe før, angivet med symbolet **~**, og nu er du i underbiblioteket på **Desktop** under din hjemmemappe. For at gøre det brugte du **cd**-kommandoen – *change directory* (skift mappe).

FORSKEL PÅ STORE OG SMÅ BOGSTAVER

Raspberry Pi OSs kommandolinjegrænseflade skelner mellem store og små bogstaver, hvilket betyder, at det gør en forskel, om kommandoer eller navne har store eller små bogstaver. Hvis du modtager en meddelelse at typen "no such file or directory" (en sådan fil eller mappe findes ikke), da du forsøgte at ændre mapper, skal du kontrollere, om der var et stort D i starten af "Desktop".

Der er fire måder at gå tilbage til din hjemmemappe: Prøv hver metode efter hinanden, idet du hver gang skifter tilbage til **Desktop**-undermappen. Den første er:

```
cd ..
```

..-symbolerne er en anden genvej, denne gang for "mappen over denne", også kendt som *overordnet mappe*. Da mappen over **Desktop** er din hjemmemappe, bliver du sendt derhen. Skift tilbage til undermappen **Desktop**, og prøv den anden vej:

```
cd ~
```

Dette bruger symbolet **~** og betyder bogstaveligt "skift til min hjemmemappe". I modsætning til **cd ..**, som bare sender dig til den overordnede mappe i forhold til den mappe, du aktuelt er i, fungerer denne kommando fra hvor som helst – men der er en nemmere måde:

```
cd
```

Uden at få navnet på en mappe, går **cd** simpelthen som standard tilbage til din hjemmemappe. Den sidste måde at komme tilbage til din hjemmemappe er at skrive:

```
cd /home/pi
```

Dette bruger det, der kaldes en *absolut sti*, som fungerer uanset hvad der er den aktuelle arbejdsmappe. Så som **cd** alene eller **cd ~**, vil dette returnere dig til din hjemmemappe, uanset hvor du er; men i modsætning til de andre metoder skal det dog kende dit brugernavn.

Håndtering af filer

For at øve dig i at arbejde med filer skal du skifte til mappen **Desktop** og skrive følgende:

```
touch Test
```

Der vises en fil kaldet **Test** på skrivebordet. `touch`-kommandoen bruges normalt til at opdatere dato og klokkeslæt for en fil, men hvis filen, som i dette tilfælde, ikke findes, oprettes den.

Prøv følgende:

```
cp Test Test2
```

En anden fil, **Test2**, vises på skrivebordet. Dette er en *kopi* af den oprindelige fil, identisk på alle måder. Slet den ved at skrive:

```
rm Test2
```

Dette *fjerner* filen, og du ser den forsvinde.

> **ADVARSEL!**
>
> I modsætning til at slette filer ved hjælp af den grafiske File Manager (Filhåndtering), der gemmer dem i papirkurven til senere hentning, er filer, der slettes ved hjælp af `rm`, forsvundet for evigt. Sørg for at være omhyggelig, når du taster!

Prøv derefter:

```
mv Test Test2
```

Denne kommando *flytter* filen, og du vil se din originale **Test**-fil forsvinde og blive erstattet af **Test2**. Flytkommandoen, `mv`, kan bruges på denne måde til at omdøbe filer.

Når du ikke er på skrivebordet, skal du stadig kunne se, hvilke filer der er i en mappe. Skriv:

```
ls
```

Denne kommando viser en *liste* med indholdet af den aktuelle mappe eller enhver anden mappe, som du angiver. For flere detaljer, herunder en liste over skjulte filer og rapportering af filstørrelser, kan du prøve at tilføje nogle parametre:

```
ls -larth
```

Disse parametre styrer kommandoen **ls**: **l** skifter output til en lang lodret liste; **a** fortæller, at den skal vise alle filer og mapper, inklusive dem, der normalt ville være skjult; **r** vender den normale sorteringsrækkefølge; **t** sorterer efter ændringstid, som kombineret med **r** viser de ældste filer øverst på listen og de nyeste filer nederst; og **h** bruger filstørrelser, der kan læses af mennesker, hvilket gør listen lettere at forstå.

Kørsel af programmer

Nogle programmer kan kun køres på kommandolinjen, mens andre har både grafiske og kommandolinjegrænseflader. Et eksempel på sidstnævnte er Raspberry Pi's Software Configuration Tool (softwarekonfigurationsværktøj), som du normalt indlæser fra hindbærikonemenuen.

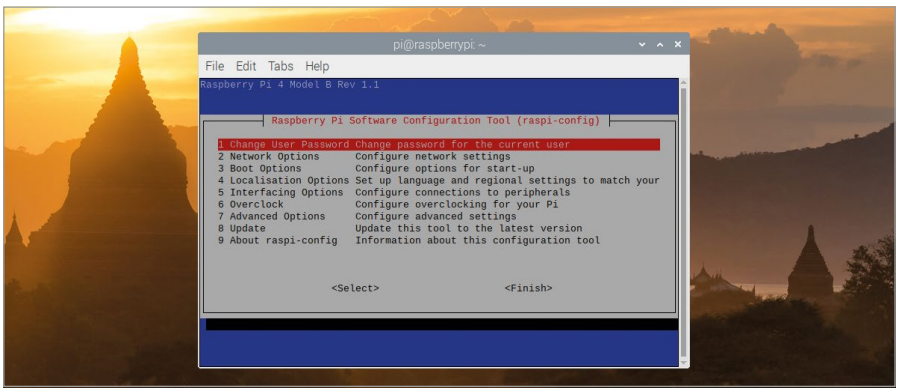
Skriv:

```
raspi-config
```

Du får en fejl, der fortæller dig, at softwaren kun kan køres som *root* (rod), superbrugerkontoen på din Raspberry Pi. Den fortæller dig også, hvordan du gør det ved at skrive:

```
sudo raspi-config
```

sudo-delen af kommandoen betyder *switch-user do*, og fortæller Raspberry Pi OS at køre kommandoen som rodbruger.



Du skal kun bruge **sudo**, når et program har brug for forhøjede *rettigheder*, f.eks. når det installerer eller afinstallerer software eller justerer systemindstillinger. Et spil skal for eksempel aldrig køres ved hjælp af **sudo**.

Tryk to gange på **TAB**-tasten for at vælge Finish (Udfør), og tryk på **ENTER** for at afslutte Raspberry Pi Software Configuration Tool (softwarekonfigurationsværktøj) og vende tilbage til kommandolinjegrænsefladen. Skriv til sidst:

```
exit
```

Dette afslutter din kommandolinjegrænsefladesession og lukker Terminal-appen.

Brug af TTY'erne

Terminal-appen er ikke den eneste måde at bruge kommandolinjegrænsefladen på: Du kan også skifte til en af et antal allerede kørende terminaler kendt som *teletyper* eller *TTY'er*. Hold tasterne **CTRL** og **ALT** nede på tastaturet, og tryk på **F2**-tasten for at skifte til "tty2".

Du skal logge på igen med dit brugernavn og din adgangskode, hvorefter du kan bruge kommandolinjegrænsefladen ligesom i Terminalen. Brug af disse TTY'er er praktisk, hvis skrivebordsgrænseflade af en eller anden grund ikke fungerer.

For at skifte væk fra TTY skal du holde **CTRL+ALT** nede og derefter trykke på **F7**: Skrivebordet vises igen. Tryk på **CTRL+ALT+F2** igen, så skifter du tilbage til "tty2" – og alt hvad du kørte i det vil stadig være der.

Inden du skifter igen, skal du skrive:

```
exit
```

Tryk derefter på **CTRL+ALT+F7** for at komme tilbage til skrivebordet. Grunden til at afslutte, før du skifter væk fra TTY, er, at alle med adgang til tastaturet kan skifte til en TTY, og hvis du stadig er logget på, kan de få adgang til din konto uden at skulle kende din adgangskode!

Tillykke: Du har taget de første skridt til at mestre Raspberry Pi OS-kommandolinjegrænsefladen!

Bilag D

Yderligere læsning

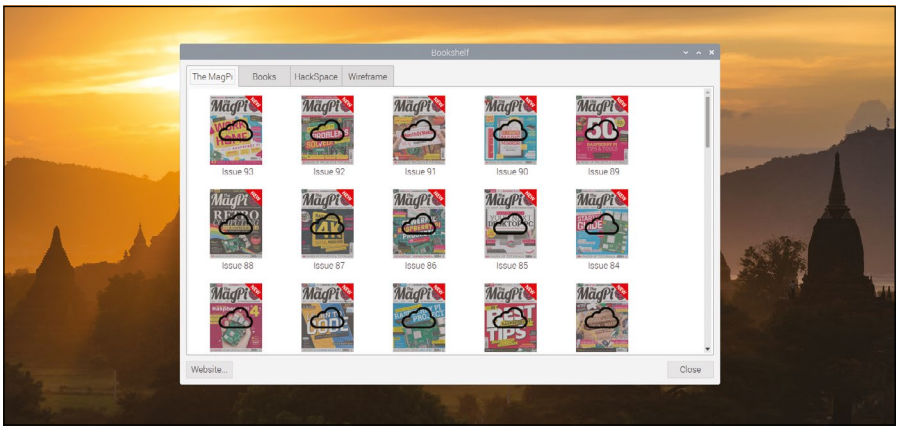
Den officielle begyndervejledning til Raspberry Pi er designet til at hjælpe dig i gang med at bruge din Raspberry Pi, men det er på ingen måde en komplet oversigt over, hvad Raspberry Pi ellers kan bruges til. Raspberry Pi-brugerskaren er et globalt samfund, og folk bruger de små computere til alt fra spil og registrering af input til avanceret automatisering og kunstig intelligens, så mulighederne er nærmest uendelige, og der er masser af inspiration at hente.

Dette bilag indeholder nogle ideer til nye projekter, planlægning af videre træning og andet materiale, der fungerer som et naturligt næste trin, efter at du har tygget dig igennem *begyndervejledningen*.

Bookshelf (bogreol)

Hindbærmenu > Help (hjælp) > Bookshelf (bogreol)

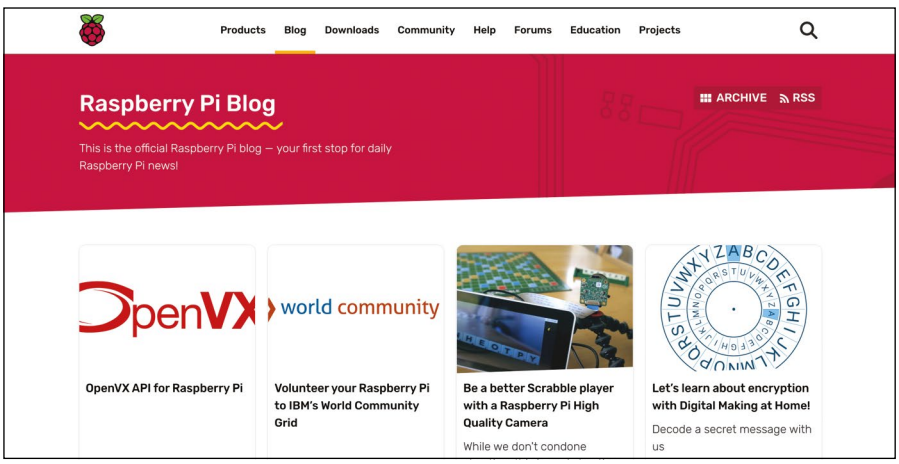
Bookshelf er en applikation, der følger med Raspberry Pi OS, og som du kan bruge til at få et overblik over, downloade og læse digitale versioner af diverse udgivelser fra Raspberry Pi Press – inklusive denne *Begyndervejledning til Raspberry Pi*. Applikationen kan indlæses ved at klikke på hindbærmenuen, vælge Help (hjælp) og klikke på Bookshelf (bogreol). Her har du adgang til de mange gratis bøger og magasiner, der kan downloades og læses, når du har tid og lyst.



Raspberry Pi-bloggen

rpf.io/blog

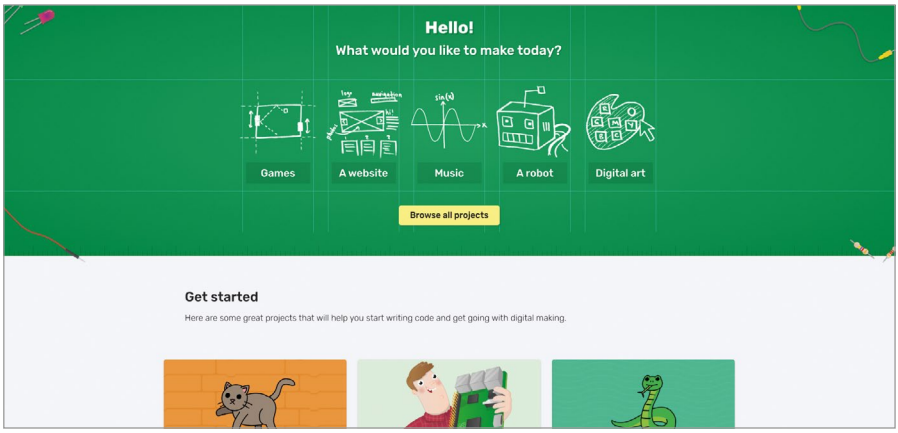
Her får du de seneste nyheder om Raspberry Pi-universet. Bloggen dækker alt fra ny hardware og undervisningsmateriale til en oversigt over de bedste projekter, kampagner og initiativer, der findes derude. Hvis du gerne vil holde dig opdateret om Raspberry Pi-universet, er det her, det sker.



Raspberry Pi-projekter

rpf.io/projects

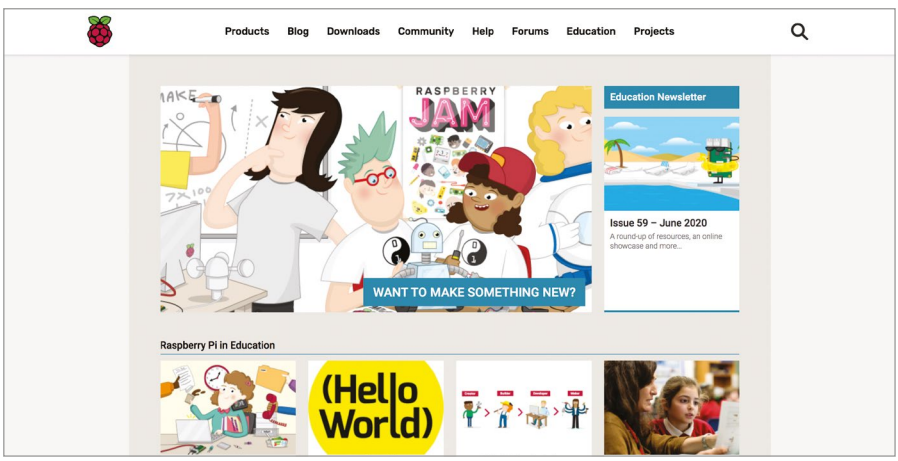
Det officielle websted om Raspberry Pi-projekter indeholder trinvise vejledninger til en række projekter i forskellige kategorier. Her kan du lære alt muligt, f.eks. programmere spil, lave musik, designe websteder eller bygge din egen Raspberry Pi-styret robot. De fleste projekter findes også på en række forskellige sprog og dækker flere sværhedsgrader, så alle kan være med, uanset om de er nybegyndere eller erfarne eksperter.



Raspberry Pi til uddannelse

rpf.io/education

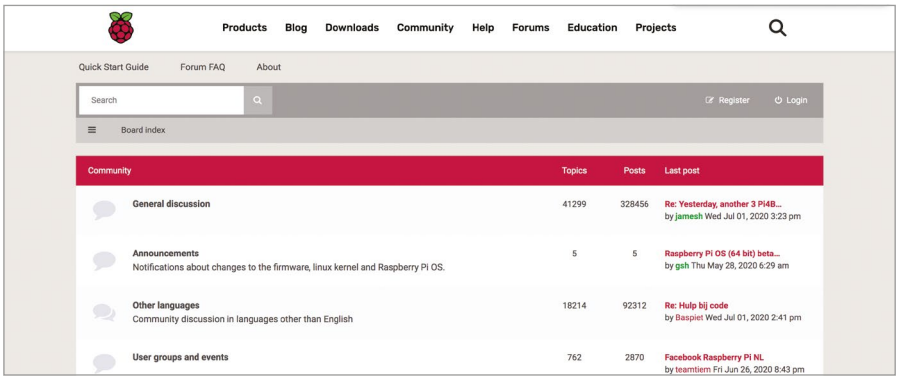
Det officielle websted for Raspberry Pi til uddannelse indeholder nyhedsbreve, onlinetræning og projekter rettet mod lærere og undervisere. Webstedet har også links til andre ressourcer, herunder træningsprogrammet Picademy, diverse kodningsprojekter drevet af frivillige som Code Club og CoderDojo samt Raspberry Jam-events i hele verden.



Raspberry Pi-forummet

rpf.io/forums

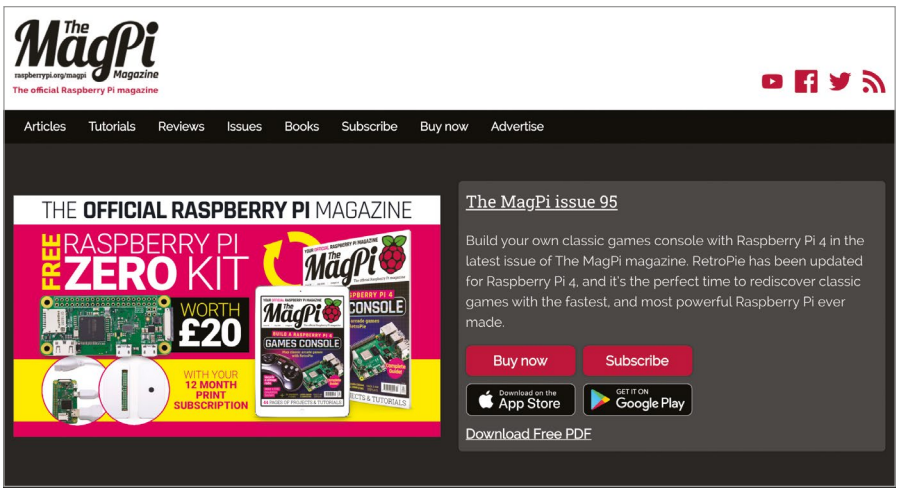
Raspberry Pi-forummet er stedet, hvor Raspberry Pi-fans kan mødes for at chatte om alt fra nybegynderproblemer til dybt tekniske emner – og der er endda et "off-topic"-område til snak om alt muligt.



MagPi-magasinet

magpi.cc

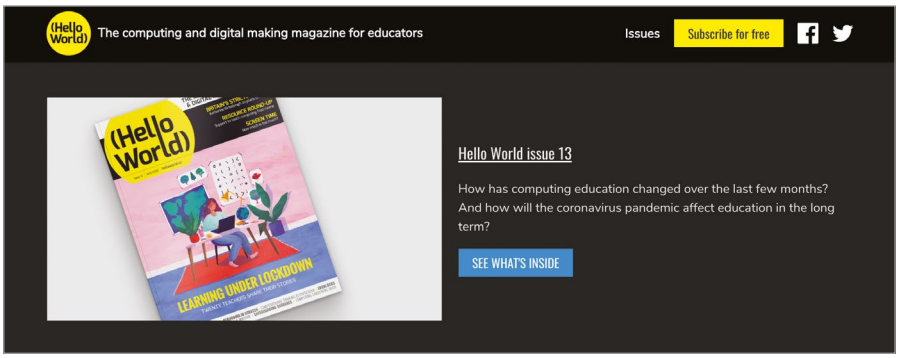
MagPi er navnet på det officielle Raspberry Pi-magasin, der udkommer en gang om måneden. Det dækker alt fra trinvise vejledninger til anmeldelser og nyheder om alt muligt i det globale Raspberry Pi-univers. Magasinet kan købes i mange aviskiosker og supermarkeder, men det kan også downloades gratis digitalt under Creative Commons-licensen. MagPi udgiver også bøger og tidsskrifter om en række emner. Disse kan købes på tryk eller downloades gratis.



Hello World-magasinet

helloworld.cc

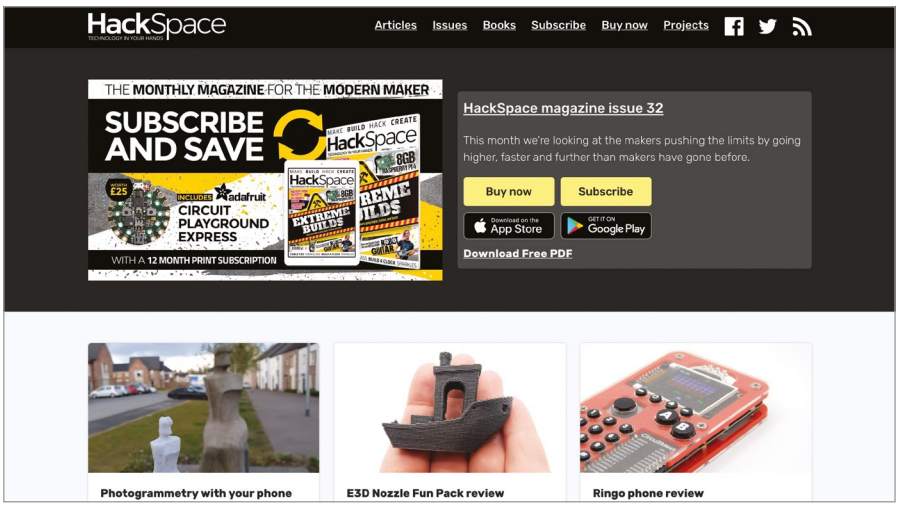
Hello World udkommer tre gange om året og er gratis for lærere i Storbritannien samt frivillige og bibliotekarer. Alle andre kan enten downloade det gratis under Creative Commons-licensen eller købe et abonnement på den trykte version.



HackSpace-magasinet

hsmag.cc

HackSpace henvender sig til et bredere publikum end MagPi. Magasinet omhandler de nyeste trends inden for udviklingen og indeholder blandt andet anmeldelser af hardware og software, vejledninger og interviews. Hvis du gerne vil udvide dine horisonter ud over Raspberry Pi-universet, er HackSpace et godt sted at starte. Magasinet kan købes i supermarkeder og aviskiosker eller downloades gratis i digital udgave.



Bilag E

Raspberry Pi Configuration Tool

Raspberry Pi Configuration Tool er en avanceret pakke til justering af adskillige indstillinger på din Raspberry Pi, lige fra de tilgængelige grænseflader over programmer til styring over et netværk. Det kan dog være en smule skræmmende for nybegyndere, så i dette bilag fører vi dig gennem hver enkelt af indstillingerne og beskriver deres formål.

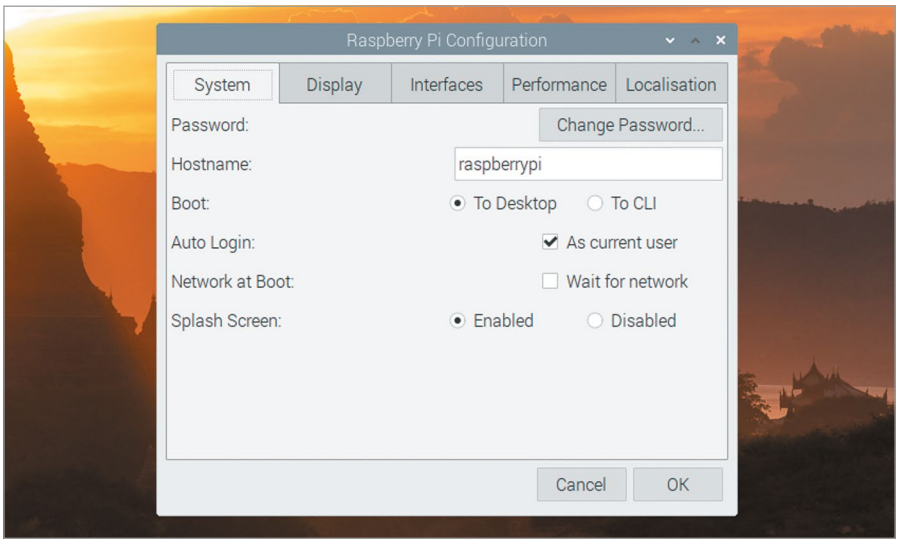
Du kan indlæse Raspberry Pi Configuration Tool fra hindbærikonemenuen under kategorien Preferences (Indstillinger). Det kan også køres fra kommandolinjegrænsefladen eller Terminal ved hjælp af kommandoen **`raspi-config`**. Layoutet af kommandolinieversionen og den grafiske version er forskellige, hvor indstillinger vises i forskellige kategorier, afhængigt af hvilken version du bruger; dette tillæg er baseret på den grafiske version.

ADVARSEL!

Medmindre du ved, at du har brug for at ændre en bestemt indstilling, er det bedst ikke at ændre noget med Raspberry Pi Configuration Tool. Hvis du tilføjer ny hardware til din Raspberry Pi, såsom en lyd-HAT eller et kameramodul, fortæller instruktionerne dig, hvilken indstilling du skal ændre; Ellers skal standardindstillingerne normalt ikke ændres.

Fanen System

Fanen System indeholder valgmuligheder, der styrer forskellige Raspberry Pi OS-systemindstillinger.



- **Password (Adgangskode):** Klik på knappen "Change Password..." (Skift adgangskode ...) for at angive en ny adgangskode til din nuværende brugerkonto. Som standard er dette "pi"-kontoen.

- **Hostname (Værtsnavn):** Navnet, hvormed en Raspberry Pi identificerer sig på netværk. Hvis du har mere end én Raspberry Pi på det samme netværk, skal de hver især have deres eget unikke navn.

- **Boot (Start):** Hvis du indstiller dette til "To Desktop" (Til skrivebord) (standard), indlæses det velkendte Raspberry Pi OS-skrivebord; hvis du vælger indstillingen "To CLI" (Til CLI) indlæses kommandolinjegrænsefladen som beskrevet i **Bilag C, Kommandolinjegrænsefladen**.

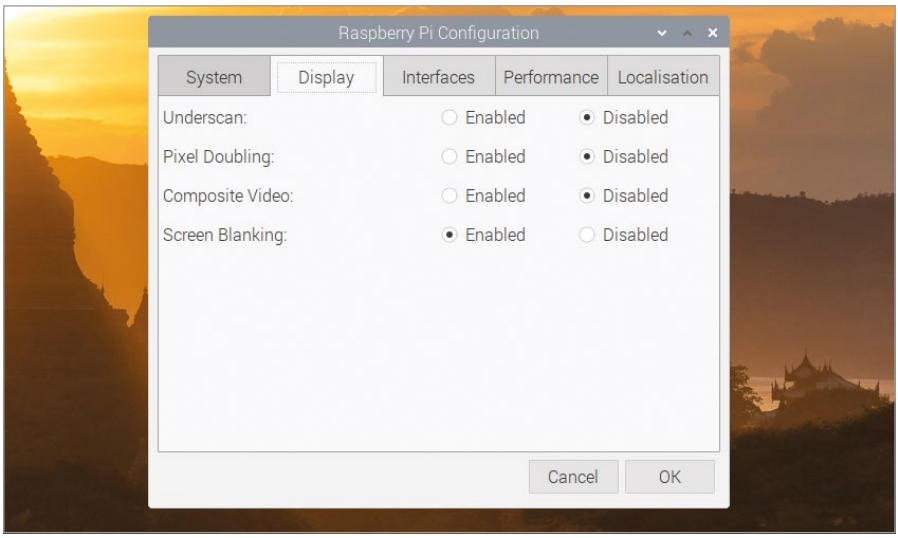
- **Auto Login (Automatisk login):** Når "Login as user 'pi'" (Log på som bruger 'pi') er markeret (standard), indlæser Raspberry Pi OS skrivebordet, uden at du behøver at indtaste dit brugernavn og din adgangskode.

- **Network at Boot (Netværk ved opstart):** Når "Wait for network" (Vent på netværk) er markeret, indlæses Raspberry Pi OS ikke, før den har en fungerende netværksforbindelse.

- **Splash Screen (Velkomstskærm):** Når indstillet til "Enabled" (Aktiveret) (standard), er Raspberry Pi OS's boot-beskeder skjult bag et grafisk velkomstbillede.

Fanen Display (Skærm)

Fanen Display (Skærm) indeholder indstillinger, der styrer, hvordan skærmen vises.



- **Overscan (Overscanning):** Denne indstilling styrer, om videooutputtet på Raspberry Pi inkluderer sorte bjælker rundt om kanterne for at kompensere for rammen på mange tv'er. Hvis du ser sorte bjælker, skal du sætte indstillingen til "Disabled" (Deaktiveret). Hvis ikke, skal du lade det være som "Enabled" (Aktiveret).

- **Pixel Doubling (Pixel-fordobling):** Hvis du bruger en skærm med høj opløsning, men i lille størrelse, kan du aktivere Pixel-fordobling for at gøre alt på skærmen større og lettere at se.

- **Composite Video:** Dette styrer composite video-udgangen, der er tilgængelig på det kombinerede lyd-video-stik (AV), når det bruges med en TRRS-adapter (tip-ring-ring-sleeve). Hvis du vil bruge composite video-output i stedet for HDMI, skal du indstille denne til "Enabled" (Aktiveret). Ellers skal du lade den være deaktiveret.

- **Screen Blanking (Skærmsluk):** Denne indstilling giver dig mulighed for at slå slukning af skærmen til og fra (timeout, der slukker for skærmen efter et par minutter).

Fanen Interfaces (Grænseflader)

Fanen Interfaces (Grænseflader) indeholder indstillinger, der styrer de tilgængelige hardwaregrænseflader på Raspberry Pi.

- **Camera (Kamera):** Aktiverer eller deaktiverer CSI (Camera Serial Interface) til brug med et Raspberry Pi-kameramodul.

- **SSH:** Aktiverer/deaktiverer SSH-grænsefladen (Secure Shell); det giver dig mulighed for at åbne en kommandolinjegrænseflade på Raspberry Pi fra en anden computer på dit netværk ved hjælp af en SSH-klient.

- **VNC:** Aktiverer/deaktiverer VNC-interface (Virtual Network Computing); det giver dig mulighed for at se skrivebordet på Raspberry Pi fra en anden computer på dit netværk ved hjælp af en VNC-klient.

- **SPI:** Aktiverer eller deaktiverer den serielle perifere grænseflade (SPI), som bruges til at styre visse hardwaretilføjelser, der opretter forbindelse til GPIO-benene.

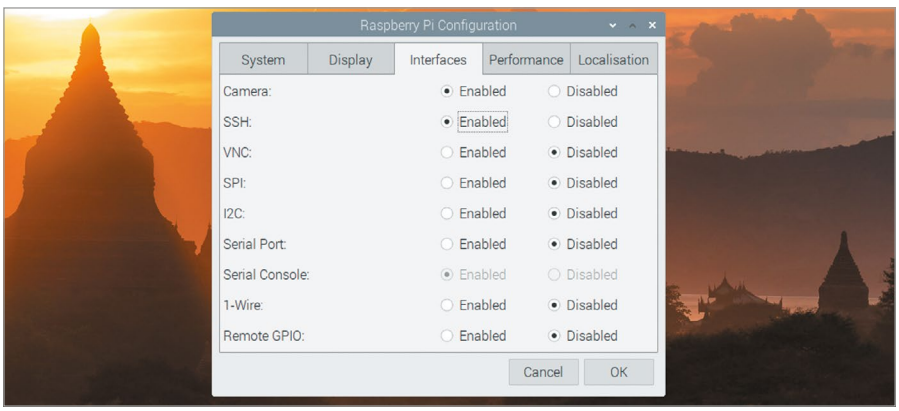
- **I2C:** Aktiverer eller deaktiverer I²C-grænsefladen (Inter-Integrated Circuit), som bruges til at styre visse hardwaretilføjeser, der opretter forbindelse til GPIO-benene.

- **Serial Port (Seriel port):** Aktiverer eller deaktiverer Raspberry Pi's serielle port, tilgængelig på GPIO-benene.

- **Serial Console (Seriel konsol):** Aktiverer eller deaktiverer seriekonsollen, en kommandolinjegrænseflade, der er tilgængelig på den serielle port. Denne indstilling er kun tilgængelig, hvis indstillingen Serial Port (Seriel port) ovenfor er indstillet til Enabled (Aktiveret).

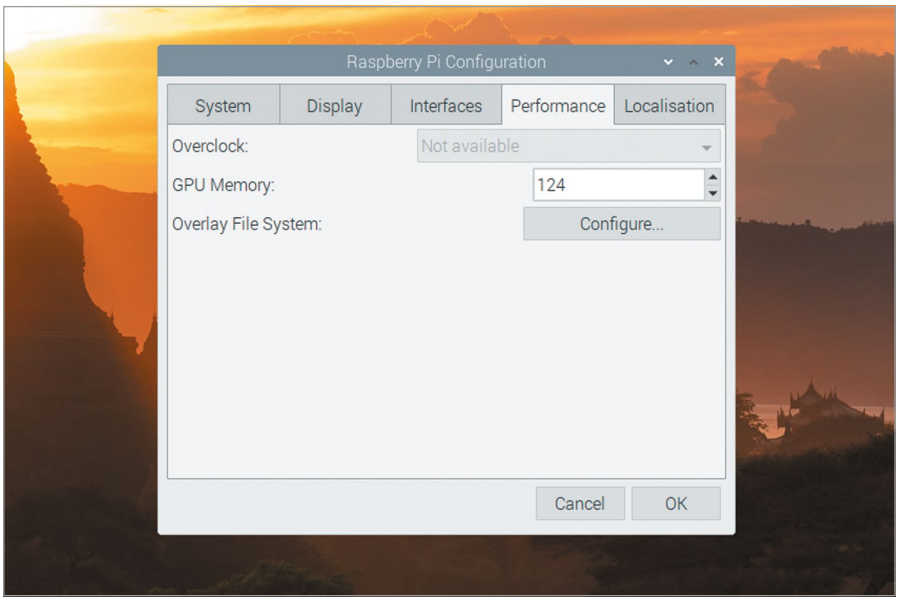
- **1-Wire:** Aktiverer eller deaktiverer 1-Wire-grænsefladen, som bruges til at styre visse hardwaretilføjeser, der opretter forbindelse til GPIO-benene.

- **Remote GPIO (Fjern-GPIO):** Aktiverer eller deaktiverer en netværkstjeneste, som giver dig mulighed for at kontrollere Raspberry Pi's GPIO-ben fra en anden computer på dit netværk ved hjælp af GPIO Zero-biblioteket. Flere oplysninger om fjern-GPIO fås fra **gpiozero.readthedocs.io**.



Fanen Performance (Ydeevne)

Fanen Performance (Ydeevne) indeholder indstillinger, som styrer, hvor meget hukommelse der er tilgængelig, og hvor hurtigt Raspberry Pi's processor kører.



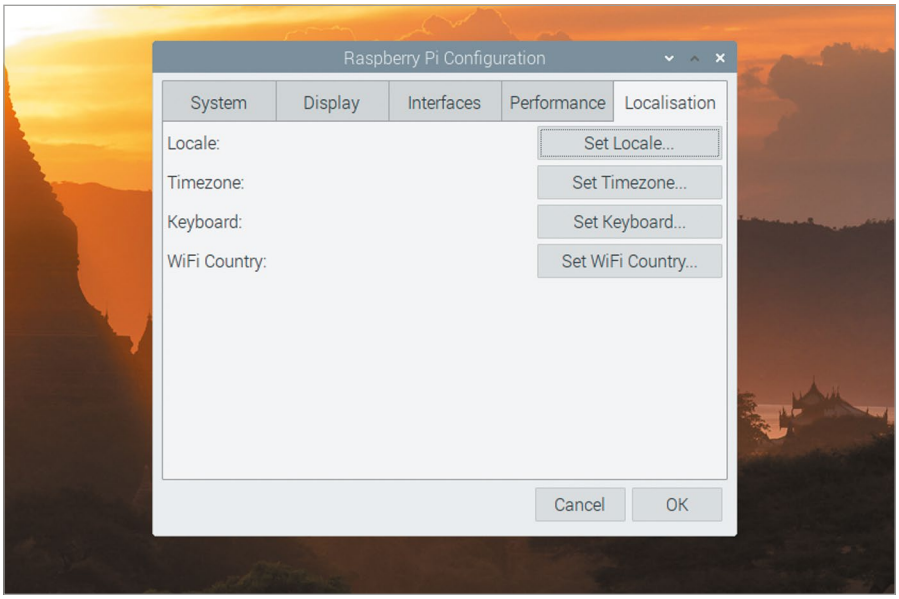
- **Overclock (Overklokke):** Giver dig mulighed for at vælge mellem en række indstillinger, der øger ydeevnen på din Raspberry Pi på bekostning af øget strømforbrug, varmeproduktion og mulig nedsat samlet levetid. Ikke tilgængelig på alle modeller af Raspberry Pi.

- **GPU Memory (GPU-hukommelse):** Giver dig mulighed for at angive mængden af hukommelse, der er reserveret til brug af Raspberry Pi's grafikprocessor. Værdier, der er højere end standard, kan forbedre ydeevnen til komplicerede 3D-gengivelser og generelle GPU-opgaver (GPGPU) på bekostning af at reducere den tilgængelige hukommelse til Raspberry Pi OS; lavere værdier kan forbedre ydeevnen til hukommelseskrævende opgaver på bekostning af, at 3D-gengivelse, kamera og valgte videoafspilningsfunktioner fungerer langsommere eller bliver utilgængelige.

- **Overlay File System (Overlay-filsystem):** Giver dig mulighed for at låse Raspberry Pi's filsystem, så ændringer kun bliver foretaget på en virtuel RAM-disk i stedet for at blive skrevet til mircoSD-kortet, så du går tilbage til en ren tilstand, hver gang du genstarter.

Fanen Localisation (Lokalisering)

Fanen Localisation (Lokalisering) indeholder indstillinger, der styrer, hvilken region din Raspberry Pi er designet til at fungere i, inklusive indstillinger for tastaturlayout.



- **Locale (Lokalitet):** Giver dig mulighed for at vælge dit land, en systemindstilling, der inkluderer sprog, land og tegnsæt. Bemærk, at ændring af sprog her kun ændrer det viste sprog i programmer, som der findes en oversættelse for.

- **Timezone (Tidszone):** Giver dig mulighed for at vælge din regionale tidszone og vælge et område i verden efterfulgt af den nærmeste by. Hvis din Raspberry Pi er tilsluttet netværket, men uret viser det forkerte tidspunkt, skyldes det normalt, at den forkerte tidszone er valgt.

- **Keyboard (Tastatur):** Giver dig mulighed for at vælge tastaturtype, sprog og layout. Hvis dit tastatur skriver de forkerte bogstaver eller symboler, kan du rette det her.

- **WiFi Country (Wi-fi-land):** Giver dig mulighed for at indstille dit land til radioreguleringsformål. Sørg for at vælge det land, hvor din Raspberry Pi bruges: Valg af et andet land kan gøre det umuligt at oprette forbindelse til nærliggende wi-fi-adgangspunkter og kan være en overtrædelse af tv-loven. Et land skal indstilles, før wi-fi-radioen kan bruges.

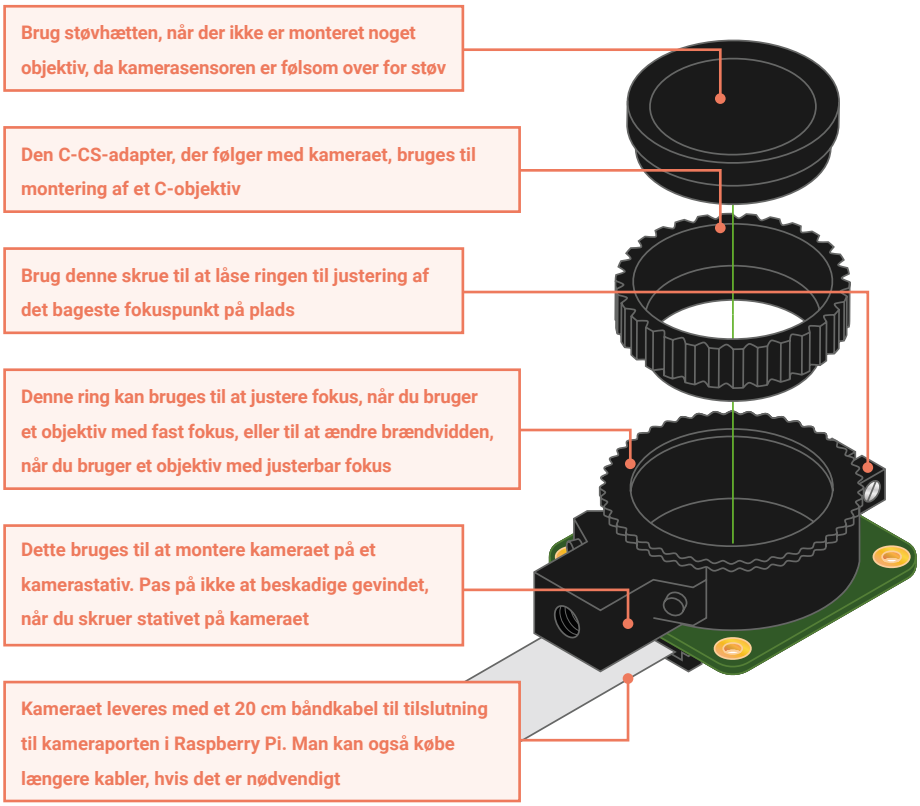
Bilag F

Opsætning af High Quality Camera

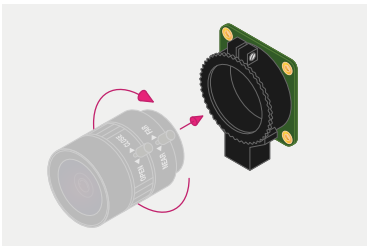
HQ Camera (HQ står for High Quality, dvs. høj kvalitet) kan optage billeder i højere opløsning end standardkameramodulet. I modsætning til standardkameraet har dette kamera ikke noget fastmonteret objektiv. I stedet kan det bruges med alle standard C- eller CS-objektiver. 6 mm- og 16 mm-objektiver kan købes sammen med kameraet.

6 mm CS-objektiv

Der findes et 6 mm-objektiv til HQ Camera til en overkommelig pris. Dette objektiv er velegnet til grundlæggende fotografering. Det kan også bruges til makrofotografering, fordi det kan fokusere på motiver, der befinder sig meget tæt på kameraet.

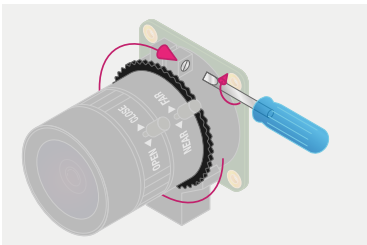


01 Montering af objektivet



6 mm-objektivet er et CS-objektiv, så C-CS-adapteren skal ikke bruges her (se diagrammet ovenfor). Objektivet vil ikke fokusere korrekt, hvis adapteren er monteret – så fjern den, hvis den er monteret. Drej derefter objektivet med uret for at fastgøre det til ringen til justering af det bageste fokuspunkt.

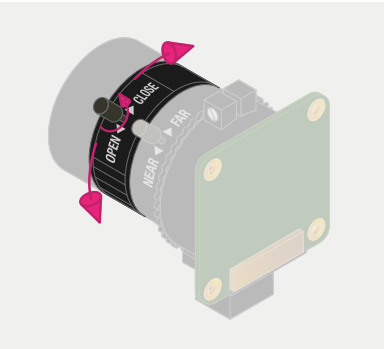
02 Ringen til justering af det bageste fokuspunkt og låseskruen



Ringen til justering af det bageste fokuspunkt skal skrues fast, så det bageste fokuspunkt er så langt tilbage som muligt. Fastgør ringen med låseskruen for at sikre, at den ikke bevæger sig ud af denne position, når blænden eller fokus justeres.

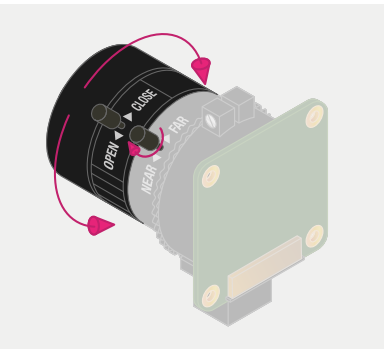
03 Blænde

For at justere blænden skal du holde kameraet, så objektivet vender væk fra dig. Drej på den midterste ring, mens den yderste ring, længst væk fra kameraet, holdes stille. Drej med uret for at lukke blændeåbningen og reducere lysstyrken. Drej mod uret for at åbne blænden. Når lysniveauet er indstillet, skal du stramme skruen på siden af objektivet for at låse blænden.



04 Fokus

Lås først den indre fokusring med skriften "NEAR ◀▶ FAR" (nært/fjernt) på plads ved at stramme skruen. Hold kameraet, så objektivet vender væk fra dig. Hold i de yderste to ringe på objektivet, og drej dem begge med uret, indtil billedet kommer i fokus – de skal nok drejes fire eller fem gange rundt. Fokus justeres ved at dreje de to yderste ringe med uret for at stille skarpt ind på et motiv, der er tæt på kameraet. Drej ringene mod uret for at fokusere på objekter, der er langt væk fra kameraet. Det er muligvis nødvendigt at justere blænden igen, når du er færdig.

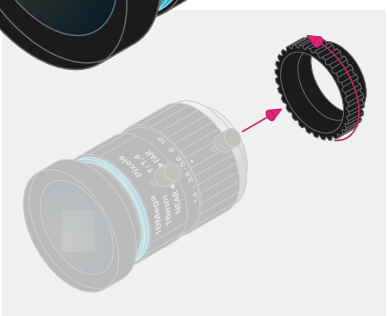


16 mm C-objektiv

16 mm-objektivet tager billeder i endnu højere kvalitet end 6 mm-objektivet. Det har en smal synsvinkel, som er mere velegnet til at tage billeder af motiver, der er længere væk.

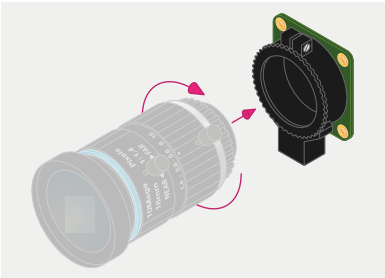
01 Montering af C-CS-adapteren

Sørg for at montere C-CS-adapteren, der følger med HQ Camera, på 16 mm-objektivet. Objektivet er et C-objektiv, så det bageste fokuspunkt ligger længere tilbage end i 6 mm-objektivet, og der skal bruges en adapter.



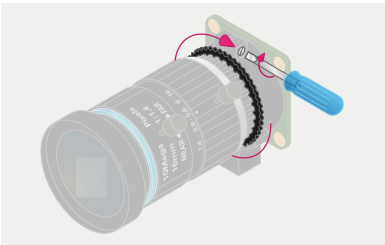
02 Montering af objektivet på kameraet

Drej 16 mm-objektivet og C-CS-adapteren med uret for at fastgøre dem til ringen til justering af det bageste fokuspunkt.



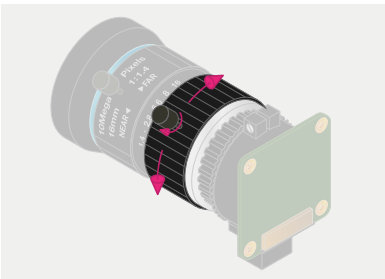
03 Ringen til justering af det bageste fokuspunkt og låseskruen

Ringen til justering af det bageste fokuspunkt skal skrues fast. Fastgør ringen med låseskruen for at sikre, at den ikke bevæger sig ud af denne position, når blænden eller fokus justeres.



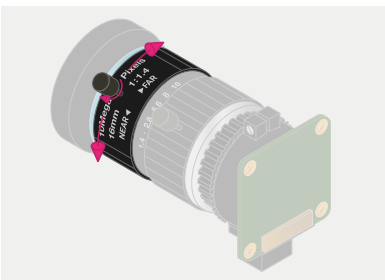
04 Blænde

For at justere blænden skal du holde kameraet, så objektivet vender væk fra dig. Drej den inderste ring (ringen nærmest kameraet), mens kameraet er ubevægeligt. Drej med uret for at lukke blændeåbningen og reducere lysstyrken. Drej mod uret for at åbne blænden. Når lysniveauet er indstillet, skal du stramme skruen på siden af objektivet for at låse blænden.



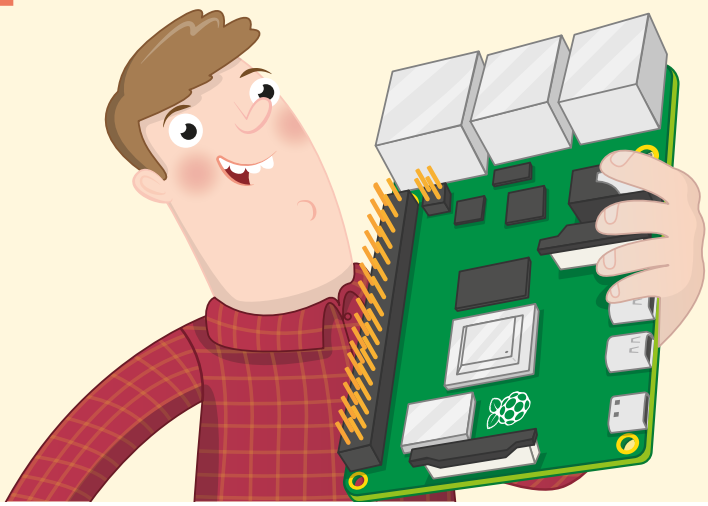
05 Fokus

Hold kameraet, så objektivet vender væk fra dig. Drej på fokusringen med påskriften "NEAR ◀▶ FAR" (nært/fjernt) mod uret for at stille skarpt ind på motiver, der er tæt på kameraet. Drej ringen med uret for at fokusere på objekter, der er langt væk fra kameraet. Det er muligvis nødvendigt at justere blænden igen, når du er færdig.



Bilag G

Raspberry Pi specifikationer

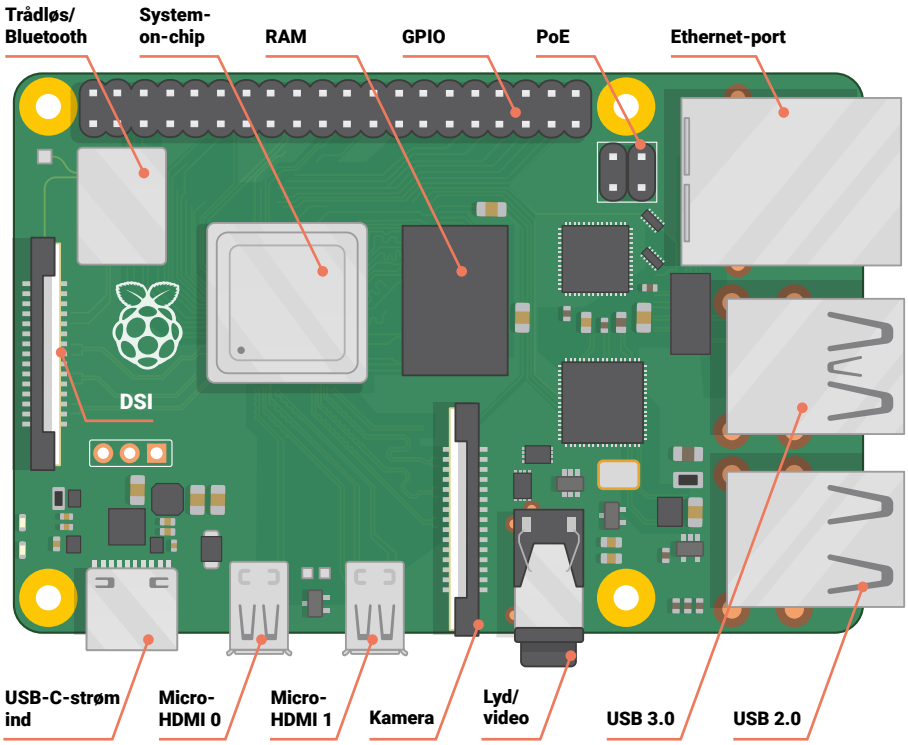


De forskellige komponenter og funktioner i en computer kaldes specifikationer og beregnet til at give et hurtigt overblik, når du f.eks. skal sammenligne to computere. Disse specifikationer kan virke forvirrende, for de er meget tekniske, men du behøver ikke at kende dem udenad for at kunne bruge din Raspberry Pi. Vi tager dem mere med for en god ordens skyld.

Systemchippen i Raspberry Pi 4 Model B og Raspberry Pi 400 er en Broadcom BCM2711B0. Dette fremgår af teksten på metallåget på Raspberry Pi 4, hvis du ellers kommer tæt nok på til at kunne læse det. Chippen har fire 64-bit ARM Cortex-A72 CPU-kerner, der hver kører med 1,5 GHz eller 1,8 GHz (1.500 eller 1.800 millioner cyklusser pr. sekund) og en Broadcom VideoCore VI (seks) grafikkort (GPU), der kører med 500 MHz (500 millioner cyklusser pr. sekund) til almindelige videoopgaver og til 3D-videoopgaver som f.eks. spil.

Systemchippen er tilsluttet 2 GB, 4 GB eller 8 GB (to, fire eller otte tusind millioner byte) – 4 GB på Raspberry Pi 400 – LPDDR4 (Low-Power Double-Data-Rate 4) RAM (Random Access Memory), der kører med 3200 MHz (tre tusind to hundrede millioner cyklusser pr. sekund). Denne hukommelse deles mellem CPU'en og GPU'en. Stikket til miroSD-kort understøtter kort på op til 512 GB (512.000 millioner byte).

Ethernet-porten understøtter netværksforbindelser på op til 1 gigabit (1000 Mbps, 1000-Base-T), mens den trådløse radio understøtter 802.11ac wi-fi-netværk, der kører i 2,4 GHz- og 5 GHz-frekvensbåndet, samt Bluetooth 5.0 og Bluetooth Low Energy (BLE).



Her er en punktopstilling over Raspberry Pi 4-specifikationer:

- **CPU:** 64-bit quad-core ARM Cortex-A72 på 1,5 GHz
- **GPU:** VideoCore VI på 500 MHz
- **RAM:** 1 GB, 2 GB eller 4 GB LPDDR4
- **Netværk:** Gigabit Ethernet, dual-band 802.11ac, Bluetooth 5.0, Bluetooth Low Energy
- **Lyd- og videoudgange:** 3,5 mm analogt AV-jackstik, 2 × micro-HDMI 2.0
- **Tilslutninger til perifært udstyr:** 2 × USB 2.0-porte, 2 × USB 3.0-porte, CSI-stik (Camera Serial Interface), DSI-stik (Display Serial Interface)
- **Lagring:** microSD (op til 512GB)
- **Strøm:** 5 volt ved 3 amp via USB Type-C
- **Ekstra:** 40-pin GPIO-stik, Power over Ethernet-kompatibilitet (med ekstra hardware)

Raspberry Pi 400-specifikationer:

- **CPU:** 64-bit quad-core ARM Cortex-A72 på 1,8 GHz
- **GPU:** VideoCore VI på 500 MHz
- **RAM:** 4GB LPDDR4
- **Netværk:** Gigabit Ethernet, dual-band 802.11ac, Bluetooth 5.0, Bluetooth Low Energy
- **Lyd- og videoudgange:** 2 × micro-HDMI 2.0
- **Tilslutninger til perifært udstyr:** 1 × USB 2.0-port, 2 × USB 3.0-porte
- **Lagring:** microSD (16 GB følger med, understøtter op til 512 GB)
- **Strøm:** 5 volt ved 3 amp via USB Type-C
- **Ekstra:** 40-benet GPIO-stik

Bilag H

Raspberry Pi-sikkerhed og brugervejledning

Raspberry Pi

Designet og distribueret af
Raspberry Pi Trading Ltd
Maurice Wilkes Building
Cowley Road
Cambridge
CB4 0DS
UK
www.raspberrypi.org

Raspberry Pi Oplysninger om overholdelse af lovgivning og sikkerhed

Raspberry Pi 4 Model B
FCC ID: 2ABCB-RPI4B
IC ID: 20953-RPI4B

Raspberry Pi 400
FCC ID: 2ABCB-RPI400
IC ID: 20953-RPI400

VIGTIGT: Se installationsvejledningen, inden du tilslutter strømmen, på **www.raspberrypi.org/safety**

ADVARSEL: Risiko for kræft og nedsat frugtbarhed – www.P65Warnings.ca.gov.

Alle lovgivningsmæssige oplysninger og certifikater findes på **www.raspberrypi.org/compliance**

IFETEL: 2019LAB-ANCE4957
Certifikatnummer på Raspberry Pi 4 Model B.

TRA-registreringsnr.
ER73381/19
Certifikatnummer på Raspberry Pi 4 Model B.

CCAO19LP1120T2
Certifikatnummer på Raspberry Pi 4 Model B.

NTC
Typegodkendelse:
NR.: ESD-GEC-1920098C
Certifikatnummer på Raspberry Pi 4 Model B.

TA-2019/750-GODKENDT
Certifikatnummer på Raspberry Pi 4 Model B.

HDMI-mærket, HDMI High-Definition Multimedia Interface og HDMI-logoer er varemærker eller registrerede varemærker tilhørende HDMI Licensing Administrator, Inc. i USA og andre lande.

DEN OFFICIELLE
Raspberry Pi
Begyndervejledning

Raspberry Pi er en lille, smart, britisk-bygget computer, der er fyldt med potentiale. Raspberry Pi er lavet ved hjælp af den samme teknologi, som du finder i en smartphone, og er designet til at hjælpe dig med at lære kodning, opdage hvordan computere fungerer og bygge dine egne fantastiske ting. Denne bog blev skrevet for at vise dig, hvor let det er at komme i gang.

Lær hvordan du:

- Konfigurerer din Raspberry Pi, installerer operativsystemet og begynder at bruge denne fuldt funktionelle computer.
- Kommer i gang med kodningsprojekter med trinvise vejledninger ved hjælp af programmeringssprogene Scratch 3 og Python.
- Eksperimenterer med tilslutning af elektroniske komponenter og har det sjovt ved at skabe fantastiske projekter.

raspberrypi.org

9 781912 047833

Raspberry Pi PRESS