Micro:bit

Python: Bilete

Skrevet av: Omsett frå microbit-micropython.readthedocs.io (https://microbit-micropython.readthedocs.io/en/latest/tutorials/images.html)

Oversatt av: Stein Olav Romslo

Kurs: Microbit
Tema: Elektronikk, Tekstbasert
Fag: Programmering
Klassetrinn: 5.-7. klasse, 8.-10. klasse, Videregående skole

Introduksjon

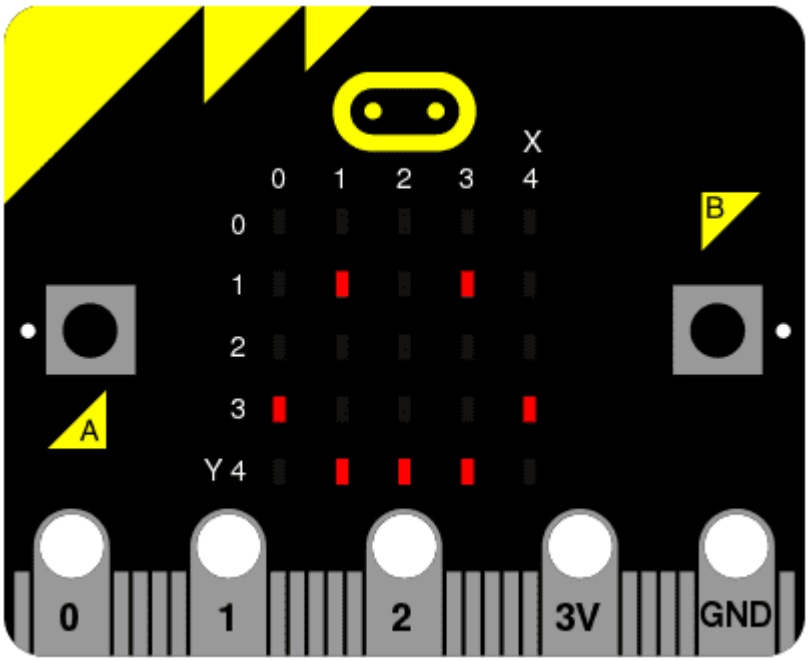
MicroPython er omtrent like god som deg til å lage kunst viss det einaste du har er eit 5x5 rutenett med raude LED-lys ("light emitting diodes" -- dei tinga som lyser opp på framsida av micro:bit-en). I denne oppgåva skal me sjå at MicroPython gir deg stor kontroll over skjermen, slik at du kan lage ei rekkje ulike interessante effektar.

MicroPython har ei rekkje bilete bygd inn, desse er klare til å visast på skjermen. Til dømes kan du få micro:bit-en til å sjå lykkeleg ut viss du skriv

```
from microbit import *

display.show(Image.HAPPY)
```

Me går ut frå at du hugsar kva den fyrste linja med kode gjer. Den andre linja brukar `display` -objektet til å vise (`show`) eit bilete som er bygd inn. Smilefjeset me vil vise er ein del av `Image` -objektet og heiter naturleg nok `HAPPY` . Me kan fortelje `show` at den skal bruke biletet ved å leggje det mellom parentesane `(` og `)` .



Liste over tilgjengelege bilete

- `Image.HEART`
- `Image.HEART_SMALL`
- `Image.HAPPY`
- `Image.SMILE`
- `Image.SAD`
- `Image.CONFUSED`
- `Image.ANGRY`
- `Image.ASLEEP`
- `Image.SURPRISED`
- `Image.SILLY`
- `Image.FABULOUS`
- `Image.MEH`
- `Image.YES`
- `Image.NO`

- `Image.CLOCK12`, `Image.CLOCK11`, `Image.CLOCK10`, `Image.CLOCK9`, `Image.CLOCK8`, `Image.CLOCK7`, `Image.CLOCK6`, `Image.CLOCK5`, `Image.CLOCK4`, `Image.CLOCK3`, `Image.CLOCK2`, `Image.CLOCK1`
- `Image.ARROW_N`, `Image.ARROW_NE`, `Image.ARROW_E`, `Image.ARROW_SE`, `Image.ARROW_S`, `Image.ARROW_SW`, `Image.ARROW_W`, `Image.ARROW_NW`
- `Image.TRIANGLE`
- `Image.TRIANGLE_LEFT`
- `Image.CHESSBOARD`
- `Image.DIAMOND`
- `Image.DIAMOND_SMALL`
- `Image.SQUARE`
- `Image.SQUARE_SMALL`
- `Image.RABBIT`
- `Image.COW`
- `Image.MUSIC_CROTCHET`
- `Image.MUSIC_QUAVER`
- `Image.MUSIC_QUAVERS`
- `Image.PITCHFORK`
- `Image.XMAS`
- `Image.PACMAN`
- `Image.TARGET`
- `Image.TSHIRT`
- `Image.ROLLERSKATE`
- `Image.DUCK`
- `Image.HOUSE`
- `Image.TORTOISE`
- `Image.BUTTERFLY`
- `Image.STICKFIGURE`
- `Image.GHOST`

- `Image.SWORD`
- `Image.GIRAFFE`
- `Image.SKULL`
- `Image.UMBRELLA`
- `Image.SNAKE`

Ting å prøve ut

Det er ganske mange bilete! Modifiser koden som får micro:bit-en til å sjå lykkeleg ut til å vise ulike bilete. Klarar du å løyse nokre av oppgåvene under?

- [ ] Får du eininga til å vise ein strekmann?
- [ ] Kor mange ulike dyr klarar du å vise?
- [ ] Få eninga til å vise kva time det er akkurat no!

Lag dine eigne bilete

Sjølvsagt har du òg lyst til å vise dine eigne bilete på micro:bit-en, ikkje sant?

Det er enkelt.

Kvar LED-piksel på den fysiske skjermen kan vere ein av ti verdiar. Viss ein piksel er sett til `0` (null) så er den av. Det er bokstavleg tala ein lysstyrke på null. Dersom verdien er sett til `9`, så er den sett til maks.

Bevæpna med denne informasjonen kan me lage nye bilete slik som dette:

```
from microbit import *

panda = Image("05050:"
              "05050:"
              "05050:"
              "99999:"
              "09990")

display.show(panda)
```

Kva viser denne koden når du sender den til einings di? Hint: Det er ikkje ein panda!

Har du funne ut av korleis du kan teikne eit bilete? Har du lagt merke til at kvar linje på skjermen er representert ved ei linje av tal som sluttar med `:` og med eit par hermeteikn `"` kring? Kvart tal representerer ein spesifikk lysstyrke. Det er fem linjer med fem tal, så det er mogleg å spesifisere lysstyrken for kvar av dei fem pikslane på kvar av dei fem linjene på skjermen. Det er slik me lagar eit bilete.

Enkelt!

Faktisk treng me ikkje skrive det over fleire linjer ein gong. Viss du klarar å halde styr på kvar enkelt linje, så kan du skrive det slik:

```
boat = Image("05050:05050:05050:99999:09990")
```

Ting å prøve ut

- [ ] Lag eit smilefjes der det eine auget lyser svakare enn det andre.

- [ ] Lag eit hjarte som lyser så svakt som mogleg. Hint: Du kan bruke `IMAGES.HEART` for å sjå korleis du kan teikne eit hjarte.

Animasjon

Statiske bilete (bilete som står i ro) er artige, men det er endå meir morosamt å få dei til å bevege seg. Dette er veldig enkelt å gjere med MicroPython -- det er berre å bruke ei rekkje bilete etter kvarandre!

Men før me kan lære om animasjonar må me fyrst lære litt om lister. Her er ei handleliste:

```
Egg
Bacon
Tomatar
```

Den same lista kan du skrive i Python på denne måten:

```
shopping = ["Egg", "Bacon", "Tomatar"]
```

Her har me laga ei liste som heiter `shopping`, og ho inneheldt tre element. Python veit at dette er ei liste fordi det er klammeparentesar (`[` og `]`) kring. Elementa i lista er skilt

frå kvarandre med komma (`,`), og her er det tre ord (altså tekst) som er elementa: `Egg` , `Bacon` og `Tomatar` . Me veit at dei er tekst fordi dei har hermeteikn (`"`) kring seg.

Ei liste kan innehalde nesten alt i Python. Her er ei liste med tal:

```
primes = [2, 3, 5, 7, 11, 13, 17, 19]
```

Merk

Tall treng ikkje å vere i hermeteikn fordi dei representerer ein verdi. Det er ein forskjell på `2` (talverdien 2) og `"2"` (ein tekst med talet 2). Viss det er vanskeleg å forstå no kan du slappe av, du kjem snart til å bli vant til det.

Det er mogleg å ha ulike typar element i den same lista:

```
mixed_up_list = ["hello!", 1.234, Image.HAPPY]
```

Me kan fortelje MicroPython at den skal animere ei liste med bilete. Til dømes har me allereie eit par lister med bilete som er innebygd. Desse heiter høvesvis `Image.ALL_CLOCKS` og `Image.ALL_ARROWS` .

```
from microbit import *

display.show(Image.ALL_CLOCKS, loop=True, delay=100)
```

Som for eitt enkelt bilete brukar me `display.show` til å vise bileta på displayet. Merk at me ba MicroPython om å bruke `Image.ALL_CLOCKS` og den forsto at me ville sjå kvart bilete etter kvarandre. Me kan òg be MicroPython om å vise dei om att og om att, slik at animasjonen fortset for alltid, ved å skrive `loop = true` . Vidare kan me seie at me vil at pausa mellom kvart bilete berre skal vere 100 millisekund (eit tidels sekund) med argumentet `delay = 100` .

Ting å prøve ut

- [ ] Animer alle bileta av piler frå `Image.ALL_ARROWS` -lista.
- [ ] Gjer slik at animasjonen berre køyrer ein gong (hint: det motsette av `true` er `false`).
- [ ] Endre hastigheita på animasjonen slik at den går dobbelt så raskt.

Skipshavari

Endeleg kjem me til delen der du kan lage din eigen animasjon. I dømet vårt skal me få ein båt til å synke til botnen av skjermen.

```
from microbit import *

boat1 = Image("05050:"
              "05050:"
              "05050:"
              "99999:"
              "09990")

boat2 = Image("00000:"
              "05050:"
              "05050:"
              "05050:"
              "99999")

boat3 = Image("00000:"
              "00000:"
              "05050:"
              "05050:"
              "05050")

boat4 = Image("00000:"
              "00000:"
              "00000:"
              "05050:"
              "05050")

boat5 = Image("00000:"
              "00000:"
              "00000:"
              "00000:"
              "05050")

boat6 = Image("00000:"
              "00000:"
              "00000:"
              "00000:"
              "00000")

all_boats = [boat1, boat2, boat3, boat4, boat5, boat6]
display.show(all_boats, delay=200)
```

```
display.show(all_boats, delay=200)
```

Her er ei forklaring av korleis koden virkar:

- Me lagar seks `boat` -bilete, akkurat som forklart tidlegare.
- Så lagar me ei liste `all_boats` av alle seks båtbileta.
- Så seier me at `display.show` skal animere lista med ei pause på 200 millisekund mellom kvart bilete.
- Sidan me ikkje har lagt til `loop = True` vil båten berre synke ein gong (sidan båtar stort sett berre gjer det ein gong).

Ting å prøve ut

- [ ] Bruk smilefjeset du laga tidlegare og animer det til å blunke med eitt auge.
- [ ] Animer hjartet du laga tidlegare til å banke (hint: du kan la lysstyrken gå frå `0` til `9` og tilbake att).
- [ ] Vel ditt eiga bilete og animer det. Treng du inspirasjon kan du sjå på lista over innebygde bilete over.