Kapitel 7 : micro:bit med Python

Övning 1 : Displayen

I denna övning ska vi skriva ett Python-program som ska styra displayen på micro:bit olika sätt.
Övningen består av två delar:
Del 1: Möte med utomjordingar,
Del 2: visa bilder

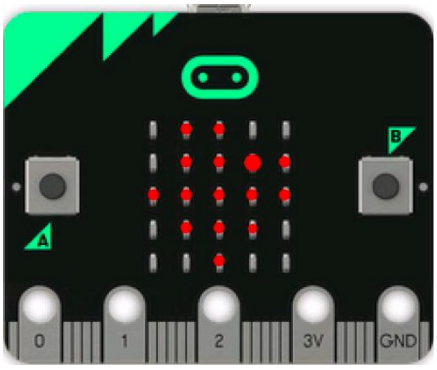
Mål :

- Att styra diplayen på micro:bit med metoden **.show**(), **.scroll**() och **.show**(image).
- Att styra displayens hastighet med kommandot **sleep**(ms)

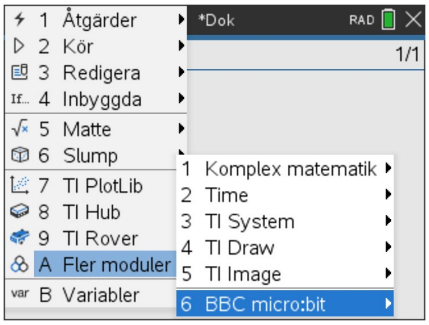
1. Innan du börjar, se till att:
   - Se till att din TI-Nspire CX II-T har OS **5.3** eller högre.
   - Du är enkelt att programmera i Python, när du har slutfört kapitlen 1 till 5 innan du börjar med detta kapitel.
   - din micro:bit är ansluten till din TI-Nspire CX II-T
   - Du har följt installationsanvisningarna och gjort filöverföringarna beskrivna i denna guide: https://education.ti.com/sv/larare/microbit

   Denna installationsprocess behöver bara göras en gång, men håll dig regelbundet informerad om uppgraderingar.

2. Om installationen har gjorts korrekt och allt fungerar så ska din micro:bit se ut som i bilden till höger. Displayen, som består av 25 lysdioder ska visa Texas Instruments logotyp och en av lysdioderna ska lysa lite kraftigare. Den motsvarar positionen för Dallas, där Texas Instruments huvudkontor ligger.



3. Micro:bit -modulen ska nu vara installerad i ditt Python-bibliotek. Öppna ett nytt Python program: **[menu], Nytt dokument, Lägg till Phyton, Nytt…** och ge programmet ett lämpligt namn. Tryck sedan på **[meny]> Fler moduler** för att se att **BBC micro:bit** är listat under TI-modulerna.



Observera att i OS version 5.3 och senare, lagras Python-modulerna i Pylib-mappen under menyn Fler moduler. Här läggs de i bokstavsordning efter filnamn.

Lärartips: Komma igång med micro:bit.

Denna enhet förutsätter att micro:bit och TI-Nspire CX-handenhet (eller datorprogramvaran) är "ready to go". Se till att du (och dina elever) har slutfört installationsstegen på sina individuella handenhet och mikro:bit-enheter. Var noga med att läsa och utföra installationsanvisningarna i guiden "Getting Started" som finns i nedladdningsmappen från TI Education -webbplatsen. *https://education.ti.com/sv/larare/microbit*. Som med all ny programvara och hårdvara, se till att hålla dig informerad om uppgraderingar och uppdateringar.

Det finns också ett inledande TI-Nspire-dokument, **my first program.tns**, som ser till att din micro:bit är ansluten och fungerar korrekt.

BBC micro:bit-modulen (microbit.tns) är en speciell modul från Texas Instruments som är installerad i Pylib-mappen på varje handenhet, och i TI-Nspire CX datorprogramvara visas den under **[meny]> Fler moduler > som BBC micro:bit**.

De första fem Python kapitlen i **10 Minutes of Code** bör göras innan detta kapitel.

Programmeringen som vi kommer att göra här är inte komplicerade, men det bör finnas en kunskap om programmering i Python på TI-Nspire CX II-T som är i nivå med de tidigare kapitlen.

Denna enhet är utformad för att fungera med micro:bit version 1 och 2 men tar inte upp funktioner som är unika för version 2. BBC micro:bit modulen stöder dock dessa funktioner.

Alla micro:bit-projekt som finns online (inklusive micro:bit version 2-projekt) kan utvecklas på TI-Nspire CX II-T med hjälp av Python och en micro:bit. Men det finns två viktiga skillnader:

- Du kan inte koppla bort micro:bit-enheten från handenheten medan programmet körs även om ett batteri är anslutet till enheten. Micro:bit runtime (.hex -fil) är konfigurerad för att ta emot instruktioner från en TI-Nspire-handenhet, eller programvara. Kalkylatorn kontrollerar och styr mikro:bit-enheten.

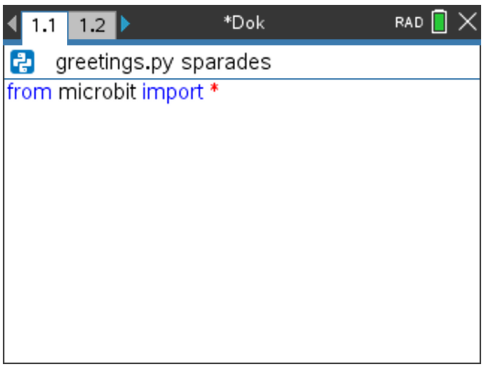
- I många av demoprogrammen som finns online används **While True:** för att skapa en oändlig loop. En sådan loop körs då direkt på micro:bit -enheten till dess programmet ersätts med en annan nedladdad .hex -fil. När du använder TI-Nspire styr handenheten micro:bit så att slingan som används mest är

```
While get_key() != 'esc':
```

vilket gör att användaren kan trycka på **[esc]** -knappen för att avsluta programmet.

Om meddelandet " micro:bit card is not connected" visas, kopplar du bara ur kabeln ur micro:bit-kortet eller ur handenheten och ansluter den igen. Detta nollställer micro:bit-enheten

4. **Del 1: Möte med utomjordingar**



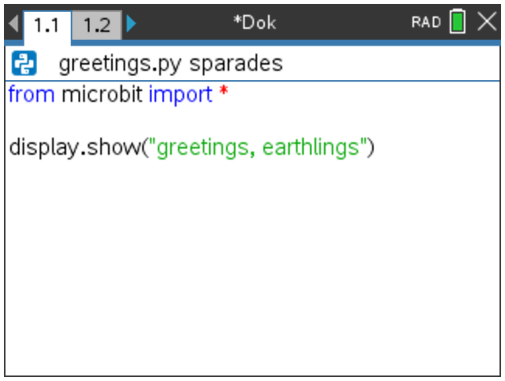
Naturligtvis, som med alla andra första program i ett nytt språk eller med en ny programmeringsmiljö så börjar vi med att visa ett meddelande. Vi ska skriva en text med hjälp av micro:bit-skärmen.

Starta ett nytt TI-Nspire-dokument och välj **Lägg till Python> Nytt...** för att starta ett nytt program (tomt program), med namnet 'greetings'. I Python Editor använder du **[menu]> Fler moduler> BBC micro:bit** för att lägga till ett importkommando i din kod:

from microbit import *

Tips: *Om meddelandet " micro:bit card is not connected" någonsin visas, koppla bara ur kabeln ur micro: bit-enheten eller ur grafräknaren och anslut den igen. Detta nollställer micro:bit-enheten*

Lärartips: Om en mikro:bit-enhet inte är ansluten, kommer ett fel rapportersas vid import-raden då programmet körs. Eftersom python är ett "modulärt" språk, läggs till ytterligare funktioner efter behov. Det här importmeddelandet laddar in specifika funktioner/metoder som krävs för att driva micro:bit-enheten med TI-Nspire CX II-T. Det laddar också några användbara metoder från andra standardmoduler och ti_ -moduler.



5. För att visa ett textmeddelande på micro:bit-skärmen ska vi använda metoden: **display.show (bild eller text)**.

*En **metod** är en funktion som ligger inbakad i en modul, eller snarare något som kallas för ett objekt. Dessa nås med modulnamnet.funktionsnamnet().*

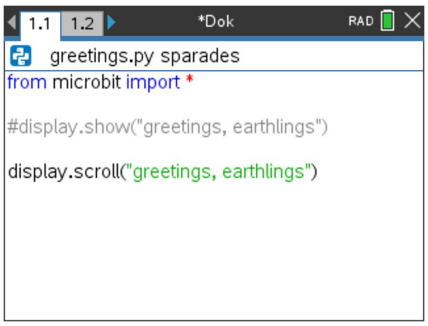
Du finner denna metod under:
[menu]> Fler moduler> BBC micro:bit> Display> Methods

Metoden infogas som **display.show (bild eller text)**, men (bild eller text) är bara en platshållare som måste ersättas med något. Inom parentes, ersätt bild eller text genom att skriva din meddelandesträng med citattecken: ”greetings, earthlings”.

När du kör det här programmet genom att trycka på **[ctrl] [R]** kommer du att se bokstäverna i ditt meddelande visas, en bokstav i taget, på displayen. De små bokstäverna 'e' visas två gånger men du kan inte skilja dem åt.

Om du gör ett misstag .... gå tillbaka till programeditorn för att redigera ditt program och kör sedan programmet igen. Glömde du att kanske att lägga till citattecken runt texten du ville visa?

6. En annan, i detta fall bättre, metod för att visa meddelandet är:

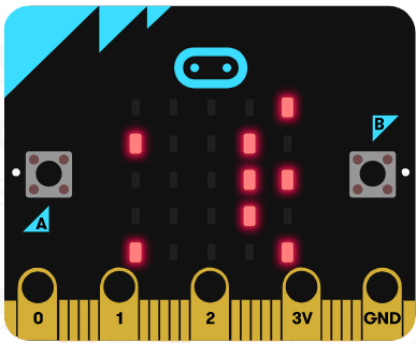


display.scroll (" greetings, earthlings")

Detta kommando finns också under:
[menu]> Fler moduler> BBC micro: bit> Display> Metods

För att slutföra .scroll () -uttrycket kan du kopiera/klistra in strängen från .show-metoden.

Gör föregående .show()-rad till ett #kommentar. Placera markören på raden och tryck på [ctrl] [T] för att göra den till en kommentar – som alltså inte kommer att köras - och kör sedan programmet igen.



Du kan också helt enkelt ändra .show till .scroll genom att skriva direkt in i koden.

7. Metoden **.scroll()** får meddelandet att rulla (flytta) från höger till vänster som en banner över displayen vilket gör det lättare att läsa. Du kan styra rullningens hastighet genom att lägga till parametern **delay =** tid i millisekunder. Exempelvis:

display.scroll ("greetings, earthlings", delay = 200)

Detta ger en fördröjning på 200 millisekunder (0,2 sekunder) mellan varje steg i skrollningen. Prova gärna med andra fördröjnings-värden.

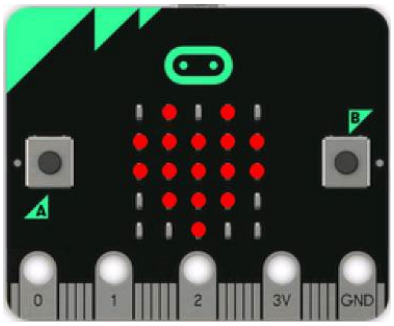
8. **Del 2**: **Hjärtslag**

   Tryck på [ctrl] [doc] för att infoga en ny sida och välj **Lägg till Python> Nytt....** för att lägga till ett nytt Python -program i ditt dokument (vi valde att döpa den till 'hjartslag' eftersom man i denna implementation av Python inte kan använda å, ä eller ö i dokumentnamn).
   I Python Editor, använd **[menu]> Fler moduler> BBC micro:bit** och välj importutsagan högst upp i listan:
   from microbit import *

9. Vi ska visa en hjärt-symbol i displayen För att visa den på micro:bit -displayen, så ska vi använda använd metoden **display.show()** igen. Vi finner denna under [**[menu]> Fler moduler> BBC micro:bit> Display> Methods**

   Inuti parentesen ersätter du platshållaren, **värde**, genom att välja: **Image.HEART** från **[menu]> Fler moduler> BBC micro: bit> Display> Images> Set 1> HEART**

10. Kör programmet (tryck **[ctrl] [R]**) för att se hjärtat visas på den lilla 5 ggr. 5 -rutnätet på micro:bit-enheten. Hjärtat kommer att visas tills något annat program körs eller till du kopplar ut enheten.

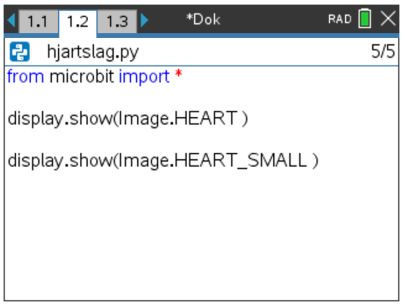


11. Vi ska nu försöka få hjärtat att bulta. Gå tillbaka till programredigeraren på föregående sida och lägg till ytterligare en rad för att visa ett litet hjärta:

    display.show (Image.HEART_SMALL)

    Du kan hitta den här bilden under samma bildmapp dom den tidigare bilden: **[menu]> Fler moduler> BBC micro: bit> Display> Images> Set 1> HEART_SMALL**

    Tips: du kan också kopiera och klistra in den tidigare raden och sedan lägga till _SMALL till HEART. Du lägger till ett understrykningstecken och så ordet SMALL med stora bokstäver.



12. Om du kör programmet så kommer det stora hjärtat visas ett kort ögonblick varefter ett litet hjärta visas. Om du kör programmet ytterligare en gång hinner du nog se det stora hjärtat.

13. Ett hjärtslag verkar lite för lite. För att få hjärtat att blinka upprepade gånger så behöver vi lägga in de två show-radera i en loop. På raden Innan de två show-raderna ska vi lägga till ett while-kommando:

    while get_key() != “esc”:

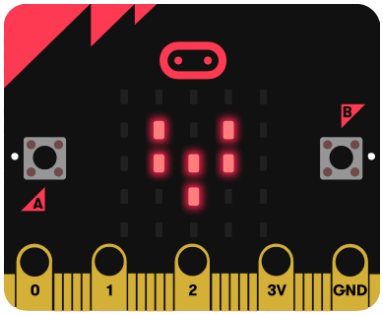
    som vi hittar under **[menu]> Fler moduler> BBC micro:bit> Commands**

    OBS: Se till att göra korrekt indrag på de två show-raderna så att de bildar det block av kod som ska loopas.

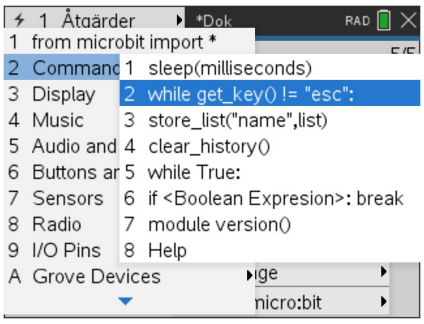
    ***Viktigt**: Indragning, indentering, av text är avgörande i Python-program. Indragen gör så att Python tolkar koden på rätt sätt i tillexempel loopar och if-satser. Kod som följer varandra och är lika mycket indragna är på samma nivå. Om de två show-raderna inte är indragna med samma antal mellanslag kommer du att få ett syntaxfel, alternativt kommer programmet inte fungera som tänkt. Använd två mellanslag eller **[tab]** -knappen för att dra in de två raderna lika mycket. Indragningen kommer att indikeras med två ljusgrå romber: (◊ ◊) för att ge extra visuell hjälp med att ha korrekta textindrag.*

14. Kör ditt program igen för att se det bultande hjärtat. Du kan trycka på på **[esc]** -knappen för att avsluta programmet. Det var det som var poängen med **get_key() != “esc”:**-raden.

    ***Tips:** om du någonsin tror att ditt program har fastnat i en oändlig loop, tryck in och håll ner **[home/on]** -knappen på din TI-Nspire för att avsluta programmet. Detta kan hända om du tillexempel använder **while True**: från kommandomenyn felaktigt. I dessa lektioner undviker den typen av strukturer.*

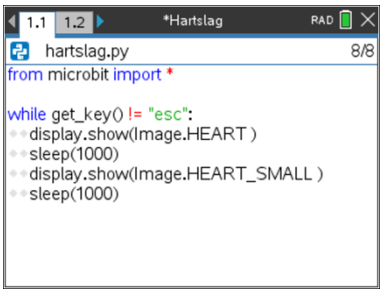


15. Under micro:bit Commands finns några användbara Python-kommandon som också finns på andra menyer. Micro:bit -modulen importerar dessa kommandon åt dig.
    Du kan också finna dessa och andra Python-kommandon under andra menyer. Du är inte begränsad till att bara använda BBC micro:bit -menyn när du programmerar micro:bit, men du kan behöva lägga till rätt importkommandon.

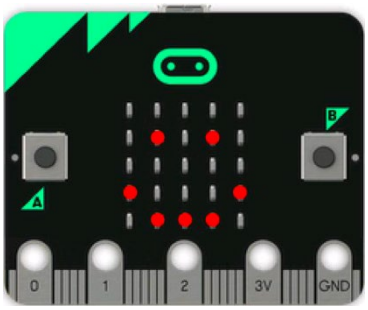


16. För att kontrollera pulsslagens takt så kan du lägga till ett sleep-kommando efter varje displayvisningsrad. **delay(1000)** betyder 1000 millisekunders fördröjning, alltså en fördröjning på en sekund.
    Du finner detta kommando **under [menu]> Fler moduler> BBC micro: bit> Commands**.

    ***Tips**: Tänk på indragen!*



17. **Extra**: Prova att ”göra ansikten”. Använd en liknande programstruktur som i "hjärtslag " men använd de ansiktsbilder som finns i bildmapp 1 istället.



Lärartips: Online lektioner för micro:bit använder vanligtvis en **While True**: loop i Python och en forever: loop i MakeCode. Dessa "oändliga loopar" körs direkt på micro:bit-enheten tills den stängs av eller ersätts av ett annat program (.hex -fil). När du använder TI-Nspire för att styra micro:bit är den oändliga slingan inte nödvändig eftersom räknaren har full kontroll. Visserligen finns **While True**: i kommandomenyn men ska användas tillsammans med **if (booleskt uttryck): break** för att komma ur den oändliga slingan.

Om en elev fastnar i en oändlig loop på handenheten, så kan man avbryta programmet genom att trycka och hålla nere **[on]** -knappen. Se guiden "Komma igång" (PDF) som följde med micro:bit -programvaran från TI, om hur man bryter Python-program på datorn.

Menyn **Commands** innehåller en samling vanliga, ofta använda, Python-kommandon som är praktiska när du kodar micro:bit. Micro:bit -modulen importerar dessa kommandon från andra moduler vid behov. Men alla Python-kommandon är tillgängliga från andra menyer och kan användas i micro:bit-projekt. Det är dock viktigt att använda rätt import-kommando vid behov.

För att få displayen tillbaka i sitt "TI" -läge (Texas TI -logotyp) i slutet av ett program, lägg till raden **display.show (ti)** eller skriv helt enkelt bokstäverna ti (små bokstäver)
efter While-loopen. Tänk på att ta bort textindraget.

Det är inte nödvändigt att lägga till ovanstående, men det hjälper till att indikera att programmet är klart på micro:bit-displayen såväl som på handenhetens skärm.

I början av ett micro:bit-program (innan while-loopen börjar) kan det hjälpa att skriva ut något på grafräknaren, till exempel print ("Startat").

Om **sleep()**: Den enda Python-funktionen som är väsentligt modifierad för micro:biten är kommandot sleep() som finns i time-modulen. Normalt representerar argumentvärdet "sekunder", men micro:bit -modulen ska tiden istället anges i millisekunder (tusendelar av en sekund). Efter att ha importerat micro:bit -modulen representerar **sleep(1000)** en fördröjning på en sekund. Observera att **sleep()** styr hastigheten på kalkylatorprogrammet, inte själva micro:bit-enheten. Var säker på att import av micro:bit-modulen sker *efter* import time * (eller någon modul som kan importera time-modulen).

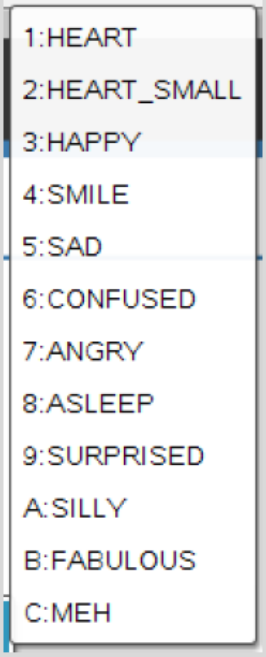
Metoden **.show()** har två valfria parametrar: **delay =** och **wait** =

display.show(value, delay=*xxx* , wait=False/True)

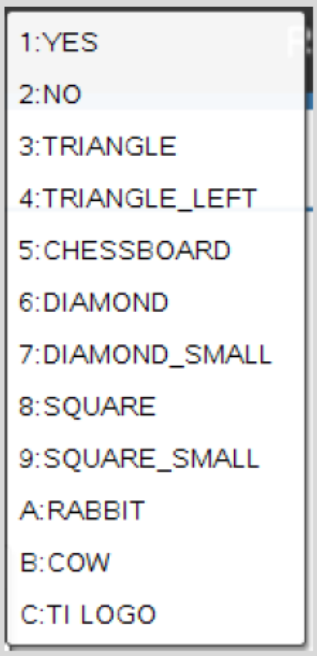
Fördröjningsvärdet är i millisekunder och ger en paus då något visas.
wait = True blockerar programkörningen tills animationen är klar, annars kommer animationen att ske i bakgrunden.

Lista på alla bilder:

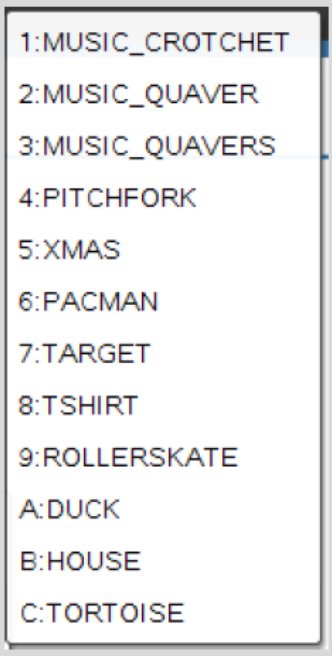
Set 1:



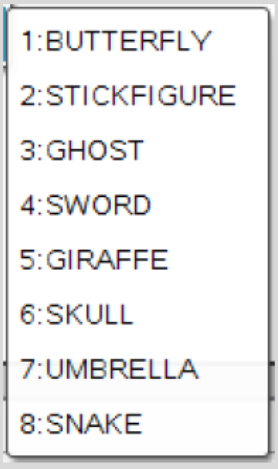
Set 2:



Set 3:



Set 4:



Ett liknande projekt som hjärtslag är att ’göra ansikten’ med samma programstruktur men att använda ansiktsikonerna i Set 1 istället.