Kapitel 2: Ljusintensitet, If och While

I denna första aktivitet för kapitel 3 ska vi undersöka den inbyggda ljusintensitetsgivaren BRIGHTNESS och använda Disp-satsen för att visa givarens avläsningar.

I de tidigare aktiviteterna har vi bara skickat instruktioner till hubben för att påverka dess inbyggda utrustning (LIGHT, COLOR och SOUND).

I denna enhet ska vi arbeta med den inbyggda ljusgivaren och använda ett värde därifrån i vårt program för att bygga en ”ljus-mätare”. Ljusgivaren producerar värden i intervallet 0 till 100 i decimalform.

För att erhålla värdet på ljusintensiteten från hubben behövs TVÅ satser:

- **Send(“READ BRIGHTNESS”)**
- **Get** <var>

Konfigurera programmet:

1. Starta ett nytt program och ge det namnet **bright1**.
2. Vi använder variabeln b för att lagra värdet på ljusintensiteten så vi deklarerar denna variabel som **Local** till programmet**.** Detta är valfritt.
3. Från hubbmenyn i editorn väljer du nu **2: Send(“READ…** och sedan **1: BRIGHTNESS**.
4. **Från Hubb-**menyn väljer du nu **Get**.
5. Skriv sedan variabelnamnet ***b***. Använd inga parenteser här.

Hur det fungerar:

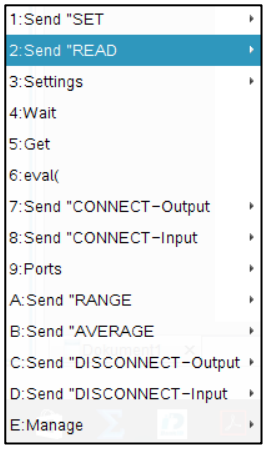
- **READ BRIGHTNESS** ger information till hubben att läsa nivån på ljusintensiteten och lagra det värdet i ett inbyggt buffertminne
- **Get b** är ett kommando för att erhålla ett värde från hubben. Denna sats överför värdet i buffertminnet hos hubben till variabeln ***b*** hos TI-Nspire. Variabeln du använder kan ha vilket giltigt variabelnamn som helst.

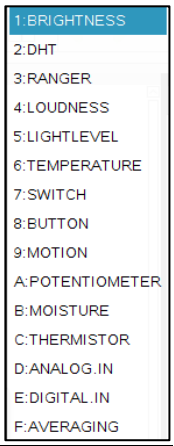
Lärarkommentar: En buffert är en minnesplats hos hubben som temporärt lagrar ett värde. Det uppdateras när ett annat ***READ***- kommando processas så det rekommenderas starkt ”to **READ”** och omedelbart ”**Get”** värdet från hubben till en variabel i TI-Nspire. Det är möjligt att samla in en uppsättning data från hubben

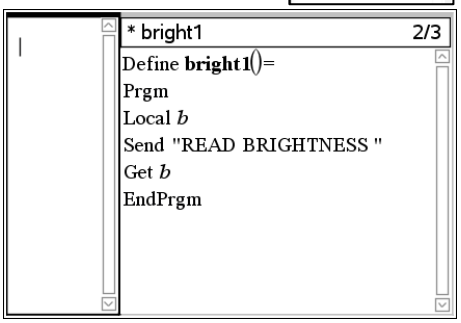
Övning 1: Mäta ljusintensitet

Syfte:

- Läsa av ljusintensitetsgivaren
- Introducera **While** loopen
- Introducera **String**(och samman-länkning







© Texas Instruments 2017

och lagra dessa värden i en lista för senare analys men det är utanför ramen för denna introduktion.

While-loop:

While…EndWhile-loopen (Kontrollmenyn i programeditorn) används för att processa ett block av kod (en följd av kodrader) medan (**while**) ett *villkor* är sant. Ett villkor är ett logiskt påstående som kan bedömas som sant eller falskt.

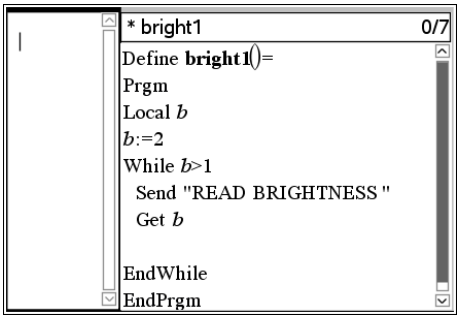
- Relationsoperatorer är **=**, **≠**, **<**, **>**, **≤**, och **≥**.
- Logiska operatorer är **and**, **or**, **not**, och **xor**

Dessa operatorer kan användas tillsammans för att bygga upp sammansatta villkor som **x>0 and y>0.**

Vi ska nu använda en enkel **While-**loop som stoppar när ljus-intensitetsvärdet är mindre än 1. För att avsluta programmet kan du helt enkelt täcka för givaren med din hand.

Lägga till en While-loop

6. Innan **Send(-**satsen i ditt program ska du lägga till satserna:
   B:=2
   While B>1



Dessa satser startar (initierar) loopen. Så länge som villkoret B>1 är *sant* kommer loopen att fortsätta att läsa av ljusgivaren. Så fort villkoret blir falskt, t.ex. om du täcker givaren med din hand så att inget ljus träffar den, kommer loopen och programmet att av-slutas.

*Om du väljer While…**EndWhile** från kontrollmenyn så kommer **EndWhile**-satsen att hamna på fel plats i koden.*

*Den ska placeras efter **Get**-satsen, vilket visas i skärmbilden. Du kan markera satsen och sedan använda Klippa ut (Ctrl-X) och Klistra in (Ctrl-V) för att placera satsen på rätt plats.*

EndWhile i **While**-loopen måste också matas in. **EndWhile**-satsen placera efter **Get**-satsen.

Lärarkommentar: Det är alltid en bra idé att ge dina program ett sätt komma ur en loop. Kom ihåg att varje kontrollstruktur har sitt eget **End.** I större program finns det vanligtvis många förekomster av dessa End-satser. Programprocessorn i räknaren "vet" vilket **End…** som tillhör en viss kontrollstruktur men det är upp till programmeraren att konstruera den korrekta koden.

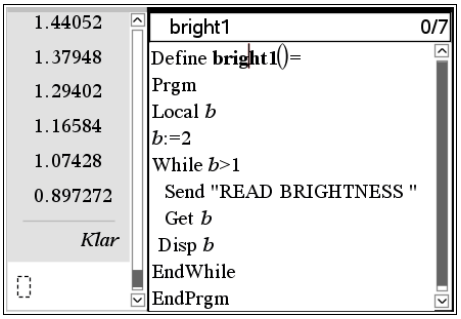
7. Lägg till **Disp**-satsen *efter* Get-satsen och *före* EndWhile i loopen.
   Disp finns i programeditorns **I/O**-meny.
8. Visa värdet hos variabeln *b*.

© Texas Instruments 2017

9. Lagra programmet (Ctrl-B) och kör programmet i Appen Räknare med hubben ansluten.

   Du kommer att se en sekvens av värden skrolla förbi i appen. Dessa värden ändras beroende på ljusintensiteten som avläses av givaren.

10. För att avsluta loopen (och programmet) så kan du täcka över ljusgivaren BRIGHTNESS hos hubben så att ljusintensitetsvärdet blir mindre än 1.



Lärarkommentarer: **Lokala** variabler existerar bara för programkörning. De skapas inte i det aktuella problemet. Om en variabel som används i ett program inte deklareras som **lokal** så kommer den variabeln att skapas i det aktuella problemet vid programkörning. Ibland är detta en fördel men ibland kan det vara problematiskt.

© Texas Instruments 2017