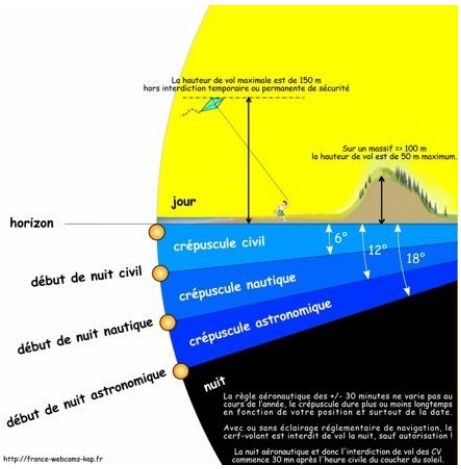
Lektion 6 : Verwendung der Module ti_hub & ti_rover — Übung 1 : Die eingebauten Sensoren des Hub

In dieser ersten Übung von Lektion 6 wird gezeigt, wie das Modul **ti_hub** zur Steuerung der in den **TI-Innovator™- Hub** integrierten Geräte verwenden wird.

Lernziele :

- Entdeckung des Moduls **ti-hub.**
- Erstellen eines Programmes, das die eingebauten Sensoren des **TI-Innovators™** nutzt.

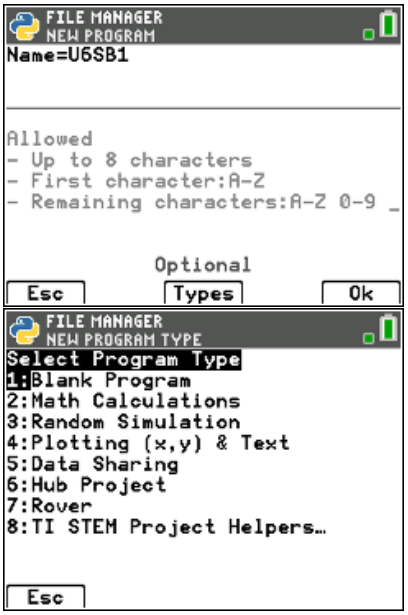
In dieser Übung wird die **ti_Innovator**-Bibliothek verwendet, um Helligkeitsänderungen zu messen, so dass man einen Dämmerungsschalter simulieren kann oder um eine Messreihe bei Sonnenaufgang oder Dämmerung aufzuzeichnen.
Anschließend wird diese Bibliothek mit den bereits bekannten Modulen **(ti_plotlib** & **ti_system**) kombiniert, um die Messungen auch grafisch darzustellen.



Das Programm entspricht dem folgenden einfachen Algorithmus:

```
Messung der momentanen Helligkeit:
Lum0 ← Messung ± (Abweichung ?)
Veränderung der Helligkeit (hellere Lampe,
Abschattung des Lichtsensors) :
Lum1 ← Messung
   Falls Lum1 > Lum 0 :
      Eine RGB LED für 2s rot einschalten
   Oder falls Lum0 < Lum1 :
      Eine RGB LED für 2s grün einschalten
    Oder : nichts tun
```

Anlegen eines neuen Programmes mit dem Namen **U6SB1**.

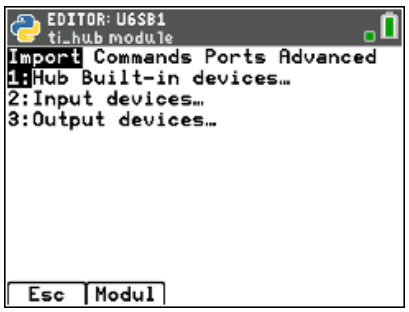


Dieses Programm muss das Modul **ti_hub** enthalten. Dafür gibt es mehrere Möglichkeiten :

a) Drücken von **f3** [Types] und anschließende Wahl von **6 : Hub Project**. Mit enter und Ok bestätigt man die Auswahl.
b) Oder man wählt – wie bisher – ein Programm ohne Vorauswahl. Das ist die Voreinstellung, wenn man ein neues Python-Programm anlegt.

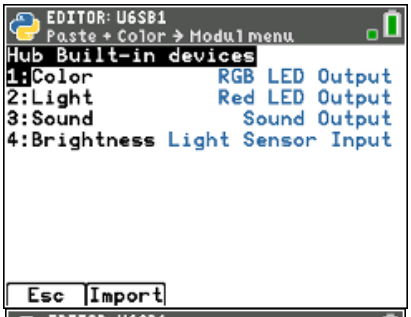
Hat man den Typ **6 : Hub Project** gewählt, ergibt sich das nebenstehende Bild.

Für das Programm wird der im **TI-Innovator** eingebaute Helligkeitssensor verwendet ebenso wie die eingebaute RGB-LED. Damit das Programm sie verwenden kann, müssen sie vorab integriert werden. Dazu wählt man im Menü zunächst das **Modul** und dann **6 : ti_hub…** und schließlich im Untermenü **1 : Hub Built-in devices.**

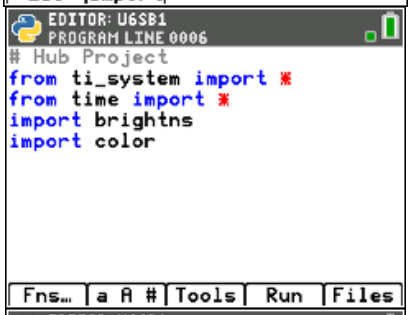


Lehrertipp : Die anderen Möglichkeiten beziehen sich auf Sensoren und Aktoren, die an die Anschlüsse IN1…IN3 bzw. OUT1…OUT3 oder BBx angeschlossen werden können.

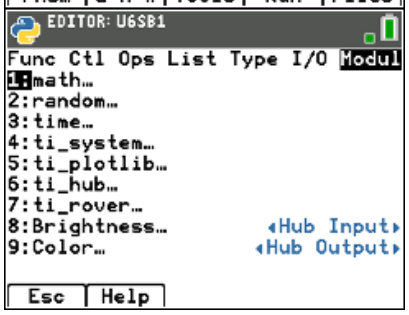
Mit enter erhält man das abgebildete Menü, aus dem 1 : Color und 4 : Brightness ausgewählt werden.



So sieht das Programm nun aus.



Die Liste der Module ist ergänzt worden.



Weiter im Programm.

```
EDITOR: U6SB1
PROGRAM LINE 0014
# Messung
disp_clr()
color.rgb(0,0,255)
lum0=brightns.measurement()
disp_at(5,"Helligkeit aendern","
    left")
sleep(3)
lum1=brightns.measurement()
# Vergleich
if lum0<lum1:
  color.rgb(255,0,0)
  sleep(2)
  color.off()
elif lum0>lum1:
  color.rgb(0,255,0)
  sleep(2)
  color.off()
else:
  color.off()
Fns… | a A # | Tools | Run | Files
```

- Als erstes sollte man den Bildschirm mit **disp_clr()** aus dem Modul **ti_system** löschen.
- Dann wird die LED mit blauem Licht eingeschaltet.
- Anlegen einer Variablen **lum0**, die den Helligkeitswert enthalten wird. Mit dem Befehl **brightns.measurement()** aus dem Modul **9 : Brightness** wird eine Helligkeitsmessung ausgelöst.
- Nun wird der Benutzer durch einen Text (**disp_at()**) aufgefordert, die Helligkeit zu ändern. Der Helligkeitssensor befindet sich an der unten abgebildeten Stelle.

- Für diese Änderung hat er 3s Zeit. Dann läuft das Programm weiter.
- Anlegen einer neuen Variablen **lum1** , in der der neue Helligkeitswert abgespeichert wird.
- Nun werden die Messungen miteinander verglichen. Je nachdem leuchtet die LED jetzt für 2s rot oder grün oder sie geht aus

Eine mögliche Erweiterung:

Mit einem Programm soll während einer frei wählbaren Anzahl von Minuten im Abstand von 1 Minute die Helligkeit gemessen und in einer Liste abgespeichert werden.

Das neue Programm sollte U6SB11 heißen.

Hinweis 1: Der Helligkeitssensor ist nicht in Lux geeicht. Das spielt in diesem Fall aber auch keine Rolle, da nur die Veränderung der Helligkeit erfasst werden soll, nicht der tatsächliche Messwert.
Hinweis 2: Will man die Veränderung der Helligkeit z.B. beim Sonnenuntergang messen, so darf während der Messung die Position des Sensors nicht verändert werden !

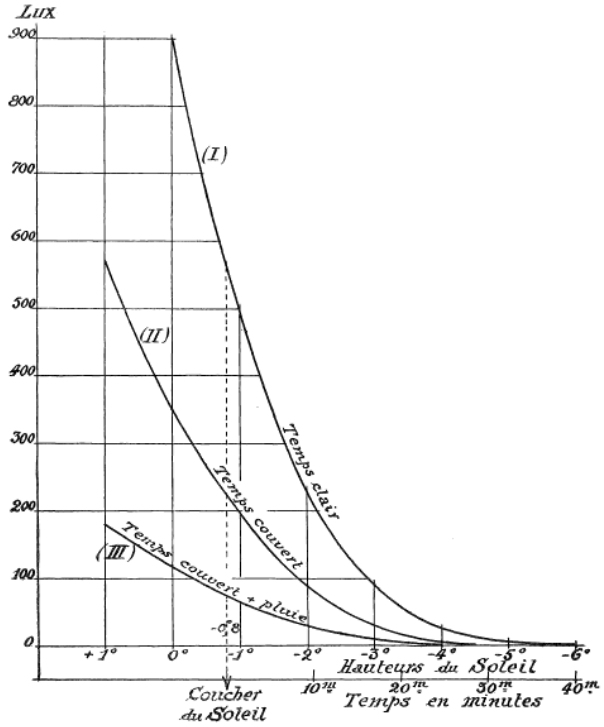


FIG. 2. — Décroissement de la lumière naturelle pendant le crépuscule civil.

Die Funktion **bri(n)** ermittelt während der Messzeit von **n** Minuten die Messwerte für die Helligkeit und speichert sie in der Liste **h[ ]**. Zugleich wird die Liste **t[ ]** der Messzeiten erzeugt.
Anschließend wird die Messung grafisch dargestellt und die Messwerte werden in die Listen L_1 und L_2 des Taschenrechners exportiert.

```
EDITOR: U6SB11
PROGRAM LINE 0001
# Hub Project_
from ti_system import *
from time import *
import ti_plotlib as plt
import brightns
disp_clr()
# Messung
def bri(n):
  t=[]
  h=[]
  for i in range(n):
    h.append(brightns.measuremen
      t())
    t.append(i)
    sleep(60)
# Grafik_
  plt.cls()
  plt.auto_window(t,h)
  plt.labels("t/min","h")
  plt.title("Helligkeit")
  plt.color(0,0,255)
  plt.scatter(t,h,"+")
  store_list("1",t)
  store_list("2",h)
  plt.show_plot()
  return
Fns… | a A # | Tools | Run | Files
```