du curseur de l'interface graphique, comprise entre 0 et 100, en nombre à virgule flottante, compris entre 0.0 et 1.0, puis l'écrit sur la broche D3.

cleanup() - tout ce que fait cette routine est d'éteindre la DEL, et essayer d'arrêter le programme proprement. Cependant elle ne parvient pas toujours à le faire; parfois il vous faudra en plus appuyer sur CTRL-C dans le terminal.

3. Activer l'interface graphique Tkinter. Tk (et sa version Python, Tkinter) est un GUI plutôt ancien, mais il est aussi relativement simple à utiliser.

Ainsi ici, je crée un curseur graphique de 400 pixels de large qui appelle la routine set_brightness() avec la valeur actuelle du curseur lorsqu'il change de valeur.

Poussé à fond à droite, le curseur appelle set_brightness(100), allumant la DEL au maximum.

Puisque la fenêtre est simple - juste un curseur et une étiquette - j'utilise la méthode basique pack() de Tk pour organiser les items dans la fenêtre.

Il dessine d'abord les items, puis les rassemble, façon Tetris, dans la fenêtre.

```
import pyfirmata
from Tkinter import *

# Création d'un nouvel objet carte
# spécifiant le port série ;
# à remplacer par /dev/ttyUSB0
# pour les anciens Arduinos
board = pyfirmata.Arduino('/dev/ttyACM0')

# Lance un nouveau thread itérateur pour
que
# le buffer série ne déborde pas
iter8 = pyfirmata.util.Iterator(board)
iter8.start()

# Configuration des broches
# A0 Entrée      (LM35)
pin0 = board.get_pin('a:0:i')
# D3 Sortie PWM (LED)
pin3 = board.get_pin('d:3:p')

# IMPORTANT ! ignorer les premières
#lectures jusqu'à ce que A0 retourne
# quelque chose de valide
while pin0.read() is None:
    pass

def get_temp():
     # Lecture du LM35 en °C
    label_text = "Temp: %6.1f C" % (
        pin0.read() * 5 * 100)
    label.config(text = label_text)
    # Redémarre après 1/2 seconde
    root.after(500, get_temp)
```

Une fois que c'est fait, il lance la première lecture de température (qui lance la suivante et ainsi de suite), et demeure finalement dans la boucle principale mainloop() de Tk jusqu'à la fin du programme, en répondant à vos entrées.

Autres pistes

Ceci est un exemple très simple du contrôle d'un Arduino en Python. Alors que Firmata peut piloter d'autres sorties plus complexes comme des servos, il prend le pas sur la totalité de la logique de la carte Arduino.

Un autre capteur qui impliquerait une mise en œuvre complexe ou un contrôle temps réel ne fonctionnerait certainement pas aussi bien.

Cela mis à part, vous avez connecté toute la puissance du Raspberry Pi à la robustesse de l'Arduino ; la seule limite, c'est votre imagination !

Auteur Stewart C. Russell

Stewart C. Russell vit à Toronto, où il crée des fermes éoliennes et des centrales électriques solaires. Quand il ne disparait pas dans les zones venteuses ou ensoleillées de la planète, il est chez lui, pratiquant la radio-amateur (indicatif VA3PID), ou jouant du banjo, bricolant des ordinateurs, ou évitant de jardiner. Son site web est http://scruss.com/blog.


```
def set_brightness(x):
    # Configuration de la DEL
    # Le curseur renvoie 0 .. 100
    # pyfirmata s'attend à 0 .. 1.0
    pin3.write(float(x) / 100.0)

def cleanup():
    # Nettoyage à la sortie
    # et extinction de la DEL
     pin3.write(0)
     board.exit()

# Configuration de l'interface graphique
root = Tk()
# cleanup() sera appelée en quittant
root.wm_protocol("WM_DELETE_WINDOW",cleanup)

# dessin curseur luminosité de la DEL
scale = Scale(root,
              command = set_brightness,
              orient = HORIZONTAL,
              length = 400,
              label = 'Brightness')
scale.pack(anchor = CENTER)

# place l'étiquette à côté du curseur
label = Label(root)
label.pack(anchor = 'nw')

# Démarre boucle lecture de température
root.after(500, get_temp)

# Lance boucle d'événements Tk
root.mainloop()
```