Full Circle

LE MAGAZINE INDÉPENDANT DE LA COMMUNAUTÉ UBUNTU LINUX

ÉDITION SPÉCIALE SÉRIE PROGRAMMATION

ÉDITION SPÉCIALE
SÉRIE PROGRAMMATION

PYTHON

Volume Onze

Parties 60 à 66

Au sujet du Full Circle

Le Full Circle est un magazine gratuit, libre et indépendant, consacré à toutes les versions d'Ubuntu, qui fait partie des systèmes d'exploitation Linux. Chaque mois, nous publions des tutoriels, que nous espérons utiles, et des articles proposés par des lecteurs. Le Podcast, un complément du Full Circle, parle du magazine même, mais aussi de tout ce qui peut vous intéresser dans ce domaine.

Clause de non-responsabilité :

Cette édition spéciale vous est fournie sans aucune garantie ; les auteurs et le magazine Full Circle déclinent toute responsabilité pour des pertes ou dommages éventuels si des lecteurs choisissent d'en appliquer le contenu à leurs ordinateur et matériel ou à ceux des autres.

Spécial Full Circle Magazine

Full Circle

LE MAGAZINE INDÉPENDANT DE LA COMMUNAUTÉ UBUNTU LINUX

Bienvenue dans une nouvelle édition spéciale consacrée à un seul sujet !

En réponse aux requêtes des lecteurs, nous avons réuni le contenu de certains articles consacrés à la programmation en python.

Pour l'instant, il s'agit d'une réédition directe de la série **Programmer en Python**, **parties 60 à 66**, des numéros 102 à 105 et 107, 108, 109, par l'incomparable professeur en Python Greg Walters.

Veuillez considérer l'origine de la publication ; les versions actuelles du matériel et des logiciels peuvent différer de ceux que nous présentons. Ainsi, vérifiez bien votre matériel et la version de vos logiciels avant d'émuler les tutoriels de cette édition spéciale. Vous pouvez installer des versions de logiciels plus récentes ou disponibles dans les dépôts de votre distribution.

Amusez-vous !

Sommaire

Partie 60 :	page 3
Partie 61 :	page 4
Partie 62 :	page 10
Partie 63 :	page 14
Partie 64 :	page 17
Partie 65 :	page 20
Partie 66 :	page 22
Écrire pour le FCM :	page 26
Mécènes :	page 27
Comment contribuer :	page 28

Python dans le monde RÉEL

Écrit par Greg D. Walters

Bienvenue amis pythoniens. Comme disent les enfants ici dans la par-tie centrale des États-Unis : « What's Shakin' Bacon ? ». Je ne sais pas exac-tement ce que cela est censé signi-fier, mais je suppose que c'est une bonne chose.

Vous avez peut-être remarqué le nouveau titre. J'ai décidé que je vous avais enseigné tout ce que je pouvais concernant les bases de programmation « générale » en Python, alors maintenant nous allons plonger dans l'utilisation de Python pour parler à d'autres types d'ordinateurs et des contrôleurs, comme le Raspberry Pi et le micro-contrôleur Arduino. Nous allons nous intéresser à des choses comme des capteurs de température, des moteurs d'asservissement, des LED clignotantes et bien plus.

Cette fois-ci, nous allons nous concentrer sur ce dont nous aurons besoin pour faire cela et sur quelques-uns des projets que nous étudierons les prochaines fois. Dans le prochain numéro, nous commencerons le premier projet.

Une des choses dont nous parlerons la prochaine fois sera le Raspberry Pi. Le Pi est un ordinateur de la taille d'une carte de crédit qui fait tourner nativement Linux sur une carte SD. Sa sortie va sur votre téléviseur via HDMI. Il dispose également d'une connexion Ethernet pour l'accès Internet.

Vous pouvez en apprendre plus sur le site officiel https://www.raspberrypi.org. Si vous voulez suivre les projets, vous aurez besoin d'un Pi, d'une carte SD, d'un clavier, d'une souris, d'une alimentation 5 volts comme celle des téléphones mobiles modernes, et de l'accès à un moniteur ou un téléviseur HDMI. Enfin, vous devriez également envisager de trouver une plaque d'essai et quelques fils de connexion pour quand nous commencerons à nous interfacer avec le monde extérieur. Vous pouvez trouver de nombreux sites sur Internet qui vendent le Pi. Ici, aux États-Unis, ils coûtent environ 35 $.

Une autre chose à propos du Pi est qu'il donne accès à une série de broches qui supportent GPIO (General Purpose Input/Output ou entrées/sorties à usage général). Cela signifie essentiellement que vous pouvez écrire des programmes qui envoient des signaux aux broches de sortie et lisent les signaux des broches d'entrée. Ceci peut être utilisé pour s'interfacer à des choses comme des LED, capteurs, boutons poussoirs, etc. Beaucoup de gens ont fait des systèmes domotiques, des systèmes multi-processeurs (en reliant ensemble 40 Pi ou davantage pour émuler un super-calculateur), des stations météorologiques, même des drones. Vous pouvez donc imaginer que les possibilités sont infinies. Voilà pourquoi j'ai décidé de commencer avec le Pi pour cette série d'articles.

Au bout d'un moment, nous allons commencer à travailler avec l'Arduino. Selon le site officiel (https://www.arduino.cc) : *« Arduino est une plate-forme électronique Open Source basée sur du matériel et des logiciels faciles à utiliser. Il est destiné à toute personne créant des projets interactifs ».*

Encore une fois, c'est un dispositif passionnant avec lequel travailler. Dans cette partie de la série, nous essaierons de parler à l'Arduino, d'abord dans son langage de script natif, puis en Python et finalement en interfaçant le Pi avec l'Arduino.

Je sais que l'article de ce mois-ci est assez court, mais j'ai eu quelques problèmes de santé, donc je garde mes forces pour le prochain article. Jusque-là, trouvez un peu d'électronique et préparez-vous à vous amuser !

Greg Walters est propriétaire de RainyDay Solutions LLC, une société de consultants à Aurora au Colorado, et programme depuis 1972. Il aime faire la cuisine, marcher, la musique et passer du temps avec sa famille.

Python dans le monde RÉEL

Bienvenue à nouveau dans ma nouvelle série d'articles sur Python. Au cas où vous l'auriez raté le mois dernier, je donne une nouvelle direction à cette série dédiée à la programmation sur Python, qui date de cinq ans maintenant, pour l'orienter vers ce que l'on appelle le Physical Computing utilisant Python. Quand vous voyez le terme « Physical Computing » pensez à des boutons, à des LED, à des moteurs, à des capteurs (de température, d'humidité, de déplacement, de pression, etc.) et bien d'autres choses. Je me suis décidé à faire cela car il me semble qu'après cinq ans j'ai couvert à peu près tout ce qui est nécessaire pour la programmation « normale ». Concentrons-nous donc sur ce que je considère être le futur de la programmation en ce qui concerne les petits ordinateurs et les micro-contrôleurs.

Ce mois-ci je vais m'occuper du choix d'un Raspberrry Pi (oui, il y en a plus d'un) qui va satisfaire nos objectifs, installer un OS sur la carte SD et démarrer pour la première fois le RPi avec le nouvel OS.

Le mois prochain nous apprendrons à répondre aux interrupteurs et aux contrôles à LED. Dans de futurs articles, nous bâtirons une interface avec les capteurs et le microprocesseur de l'Arduino.

Une brève histoire du RPi

J'ai glané l'essentiel de cette information sur le site officiel de Raspberry Pi (http://www.raspberrypi.org) et des souvenirs de mon premier achat de RPi. Au départ, quand le Raspberry Pi est sorti, il y avait deux modèles : le A+ et le B+. L'arbre de décision était plutôt simple puisque les deux modèles correspondaient à une version « simple ou complète » comme le montre le tableau élémentaire ci-dessous. (Ils sont appelés maintenant les modèles RPi 1…)

Raspberry Pi Model A+	Raspberry Pi Model B+
256 MB Ram	512 Meg Ram
One USB Port	4 USB Ports (original Model B only had 2 ports)
40 GPIO Pins (original Model A only had 26 pins)	40 GPIO Pins (original Model B only had 26 pins)
No Ethernet Port	Ethernet Port

En février 2015, ces deux modèles étaient remplacés par le RPi 2 modèle B. Il reprend la plupart des équipements du RPi 1 B+, mais a un processeur ARM Cortex-A7 à quatre cœurs cadencé à 900 MHz et 1GB de Ram.

On trouve tout un tas de versions du RPi sur Internet. Mon humble suggestion serait de retenir le RPi 2 modèle B si vous acceptez la différence de prix avec le RPi 1 modèle B (qui ne doit pas être bien importante). Tous les codes que nous allons créer dans les prochains articles devraient fonctionner sans problème avec toutes les versions du RPi.

Quand vous rechercherez votre RPi sur Internet, vous trouverez tout un tas de kits et d'accessoires comme des caméras, des servo-contrôleurs, des contrôleurs de moteurs, etc. Pour l'instant nous n'avons besoin d'aucun accessoire, mais nous pourrions en utiliser dans le futur ; alors, si quelque chose vous intéresse, faites selon vos envies. En ce qui concerne les kits, avant que vous n'achetiez le « kit extraordinaire » je voudrais vous prévenir qu'il y a un certain nombre d'éléments dont nous allons avoir besoin dans les articles à venir :

- Un ordinateur Raspberry Pi.
- Une alimentation. Pour le P1 une alimentation 5 VCC 1-1,2 A avec une connexion micro USB (standard pour un grand nombre de smartphones aujourd'hui) fonctionnera parfaitement. Pour le P2 je ne saurais trop vous suggérer d'acquérir une alimentation 5VCC 2,5 A avec un connecteur micro USB.
- Un clavier USB et une souris. Alors qu'on trouve souvent des combinaisons de petits claviers et souris, pour programmer et utiliser l'ordinateur de façon « normale », il sera préférable de choisir une version de taille standard pour les deux. Vous pourrez passer à la petite version sans fil dans le futur si vous décidez d'utiliser le RPi à

des fins telles que le multimédia ou l'automatisation étendue de la maison. Habituellement, lorsque je travaille sur le Pi, j'utilise un serveur VNC sur le Pi et un client VNC sur ma machine Linux pour ne pas avoir plusieurs claviers et souris sur mon bureau.

• Une carte SD 4-8 Go de classe 10. Les versions P1 A et B utilisent une carte SD. Les version P1 B+ et au-dessus ne supportent plus que les micro SD. Ayez bien cela en tête lorsque vous achèterez votre machine. Bien sûr, vous pouvez utiliser des cartes de plus grosses capacités. Officiellement les tests ont été réalisés avec des cartes de 32 Go et ils ne prévoient pas que des cartes de valeurs plus importantes posent problème. Soyez vigilants en achetant des cartes SD, car elles ne sont pas toutes fabriquées de la même manière. Ce n'est pas parce qu'une carte bon marché est étiquetée « classe 10 » qu'elle fonctionnera aussi bien qu'une carte plus chère.

• Une connexion Internet, soit un adaptateur WiFi USB, soit un câble Ethernet.

• Un moniteur/une télévision HDMI et un câble, pour la sortie. Si vous ne possédez pas d'HDMI le P1 A et B ont une prise RCA de sortie vidéo composite et un connecteur 3,5 mm pour le son. Les versions P1 B+ et au-delà ont laissé tomber le connecteur vidéo RCA et l'ont remplacé par un jack 3,5 mm qui combine l'audio et la vidéo. Vous aurez besoin d'un câble jack 3,5 mm vers 3 RCA pour connecter un ancien téléviseur.

• Des haut-parleurs ou des écouteurs (à moins que l'appareil que vous utilisez supporte le son en HDMI).

Alors que nous venons de voir la liste minimum des requis pour cet article, vous trouverez ci-dessous ce que vous DEVREZ avoir pour faire notre premier projet…

• Une carte d'expérimentation (breadboard). La carte d'expérimentation sera nécessaire pour pouvoir commencer à travailler avec des composants discrets comme des LED, des résistances, des interrupteurs, etc., sans devoir les souder.

• Une carte d'extension GPIO et un câble plat qui permettra de connecter les sorties GPIO du RPi à la carte d'expérimentation. Voyez http://sparkfun.com ou http://www.Adafruit.com pour cet élément. Le composant qu'il faut regarder chez Adafruit est le « Pi T-Cobbler Plus ». Notez bien que cet élément NE fonctionnera PAS avec le RPi v1 A ou B. Il fonctionnera seulement avec les versions plus récentes. Il coûte, à l'heure actuelle, aux environs de 8 $ US. Si vous utilisez un modèle A ou B, vous devrez acheter le « Pi T-Cobbler » qui coûte aux environs de 7 $. Si vous regardez chez SparkFun, leur composant s'appelle le « Pi Wedge ». À moins que vous ne vouliez fabriquer le vôtre (c'est-à-dire en soudant de tout petits éléments), vous devrez acheter une version pré-assemblée qui coûte environ 10 $. Je pense qu'ils ne font plus (ils l'ont retirée du stock) la version pour le Pi 1A et le 1B. Vous pouvez choisir de ne PAS acheter la carte d'extension et le câble plat et d'utiliser des câbles individuels avec une extrémité femelle (côté Pi) et une extrémité mâle (côté carte d'expérimentation). Cela fonctionnera ; toutefois, dans certaines des expériences que nous ferons plus tard, si vous vous trompez de connexion côté Pi, vous risquez de l'endommager.

• Toute une variété de résistances, de LED et de petits boutons poussoirs. Je vous en donnerai une liste avant que nous n'en ayons besoin pour vous laisser largement le temps de les obtenir. Il y a énormément de fournisseurs.

• Une dernière chose que vous pouvez envisager est un boîtier, mais seulement si vous avez la carte d'extension. Cela protégera votre Pi durant vos manipulations.

Configuration de votre RPi

Nous arrivons à la partie la plus fastidieuse du projet… la configuration. Voici les différentes étapes :

• Télécharger l'image de l'OS.
• Décompresser le fichier image et le mettre dans un endroit où on le retrouvera facilement.
• Installer l'OS sur la carte SD.
• Connecter le RPi.
• Mettre en route le RPi avec le nouvel OS.

Bon, allons chercher l'image de l'OS. Allez sur le site officiel de Raspberry Pi, dans la section téléchargements (downloads) (https://www.raspberry-pi.org/downloads). Vous y trouverez un grand nombre de versions d'images que vous pouvez télécharger, y compris 2 versions d'Ubuntu (la version GUI est Ubuntu Mate), Windows 10 IOT et d'autres. Si vous avez un modèle plus ancien (les premiers modèles A ou B), aucune des images Ubuntu ou Windows ne fonctionnera. Le processeur ARM V7 et la mémoire additionnelle sont nécessaires pour faire tourner ces images.

Les images qui nous intéressent pour notre projet sont la NOOBS et la RASPBIAN. Je vais utiliser la RASPBIAN Wheezy datée du 05/05/2015 pour nos premiers projets. Mais si vous préférez avoir la possibilité de démarrer sur une autre image sur la même carte, n'hésitez pas et télé-

chargez la NOOBS. Rappelez-vous toutefois que si vous avez plus d'un OS sur la même carte, vous aurez moins d'espace disponible pour l'image RASPBIAN et vous aurez le problème que je rencontrais souvent : pas assez de place pour toutes les choses que vous voulez essayer. En supposant que vous travailliez sur une machine Linux, vous trouverez toutes les instructions officielles d'installation à https://www.raspberrypi.org/documentation/installation/installing-images/linux.md. Si vous utilisez une machine Windows ou un Mac, suivez les liens fournis. Je vais partir du principe que vous utilisez une machine Linux et vous donner les instructions ici.

Avant que nous ne commencions, vous pourriez vous demander pourquoi, alors qu'il existe une version plus récente/meilleure, j'utilise celle-ci. J'ai eu quelques problèmes avec la version « Jessie » et, à l'heure actuelle, je me sens plus en confiance avec la « Wheezy ». Je doute que ce soit un problème de version, probablement un mauvais téléchargement, mais je voulais que vous le sachiez. Pour les quelques prochains articles, utilisez la « Wheezy » et sentez-vous libre de vous amuser avec d'autres versions.

Décompressez l'archive et mettez-la dans un dossier que vous retrouverez facilement.

Installation de l'image de l'OS sur la carte SD

Si vous utilisez une des premières versions du Pi, vous utiliserez une carte SD de taille standard. À l'inverse, pour une version plus récente, vous utiliserez une micro SD. Pour éviter de devoir faire la distinction à chaque fois, j'utiliserai le terme « SD » dans la documentation. Une dernière chose avant de commencer. Je recommande FORTEMENT de ne pas utiliser un appareil connecté à un multiplicateur de ports USB externe pour installer l'image sur la carte SD. Je sais qu'en théorie ça fonctionne, mais pour moi ça n'a jamais bien marché.

Bon, allons-y. Avant de mettre la carte dans votre ordi Linux, ouvrez un terminal et tapez :

```
sudo -i
```

La plupart des commandes n'ont pas besoin du niveau super-utilisateur, mais ça ne peut pas faire de mal et, en l'utilisant, ni vous ni moi n'avons besoin de nous souvenir si c'est nécessaire. Maintenant, lancez la commande « df -h » pour voir les appareils qui sont montés sur le système. La réponse de mon système se trouve ci-dessous. Oui, j'ai appelé ma machine Slartibartfast.

```
Slartibartfast ~ # df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda1       451G  336G   93G  79% /
none            4.0K     0  4.0K   0% /sys/fs/cgroup
udev            3.9G  4.0K  3.9G   1% /dev
tmpfs           796M  1.5M  794M   1% /run
none            5.0M     0  5.0M   0% /run/lock
none            3.9G  124M  3.8G   4% /run/shm
none            100M   32K  100M   1% /run/user
/dev/sdd1       2.8T  2.5T  314G  89% /media/greg/TOSHIBA
EXT
/dev/sdb1       1.8T  1.5T  294G  84% /media/greg/extramedia
/dev/sdc1       917G  681G  190G  79% /media/greg/MoreMedia2
Slartibartfast ~ #
```

Remarquez que j'ai 4 disques (sda1, sdb1, sdc1 et sdd1). Je souhaite que lorsque je branche la carte SD, elle monte comme /dev/sde1. Ce sera important à savoir parce que si nous obtenons le mauvais /dev/, nous corromprons tout ! Maintenant, branchez votre carte SD dans l'ordinateur et exécutez « df -h » à nouveau. Le système répond :

```
Slartibartfast - # df-h
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda1       451G  336G   93G  79% /
none            4.0K     0  4.0K   0% /sys/fs/cgroup
udev            3.9G  4.0K  3.9G   1% /dev
tmpfs           796M  1.5M  794M   1% /run
none            5.0M     0  5.0M   0% /run/lock
none            3.9G  124M  3.8G   4% /run/shm
none            100M   36K  100M   1% /run/user
/dev/sdd1       2.8T  2.5T  314G  89% /media/greg/TOSHIBA
EXT
/dev/sdb1       1.8T  1.5T  294G  84% /media/greg/extramedia
/dev/sdc1       917G  681G  190G  79% /media/greg/MoreMedia2
/dev/sde1        56M   20M   37M  36% /media/greg/boot
/dev/sde2        30G  3.0G   25G  11% /media/greg/13d368bf-
6dbf-4751-8ba1-88bed06bef77
Slartibartfast - #
```

Dieu merci ! Toutefois /dev/sde1 a deux partitions. Ça sera important à l'étape suivante. Si vous êtes comme moi, écrivez les informations concernant le disque, pour ne pas faire d'erreur. Maintenant nous démontons la carte SD.

```
Slartibartfast ~ # umount
/dev/sde2

Slartibartfast ~ # umount
/dev/sde1

Slartibartfast ~ # df -h
```

Notez bien que j'ai lancé un autre « df -h » afin de vérifier que l'élément est bien démonté.

Si votre carte SD a déjà été utilisée, il faudra en supprimer les partitions avant d'aller plus loin. Certaines personnes diront que ce n'est pas nécessaire, mais pourquoi pas ? Cela ne prend que quelques secondes et évitera des problèmes par la suite. Utilisez Gparted pour supprimer toutes les partitions.

```
Slartibartfast Raspbian # dcfldd bs=4M if=2015-05-05-raspbian-wheezy.img of=/dev/sde
768 blocks (3072Mb) written.
781+1 records in
781+1 records out
Slartibartfast Raspbian #
```

Nous sommes maintenant prêts à écrire l'image Raspbian sur la carte SD. Il y a deux façons de le faire. D'abord en utilisant la commande « dd » EN TANT QU'ADMINISTRATEUR (sudo) qui sera, j'en suis sûr, la première chose qui viendra à l'esprit de chacun. Toutefois, souvenez-vous que lorsque l'on utilise « dd » nous n'avons aucune idée de ce qui se passe et, si ça prend 5 mn ou plus pour écrire l'image, nous n'aurons pendant ce temps aucune information concernant la progression. Puisqu'il existe d'autres méthodes, je propose que nous utilisions la commande « dcfldd » (voir tout en haut). Une fois lancée (ce qui peut prendre en gros une minute), elle donnera des informations sur ce qui a été écrit. Choisissez votre « arme » préférée. Je vais, en ce qui me concerne, expliquer « dcfldd ». Maintenant, en tant qu'ADMINISTRATEUR (sudo), déplacez-vous dans le dossier où vous avez rangé l'image que vous allez utiliser.

```
Slartibartfast Raspbian # dd bs=4M if=/dev/sde of=wheezy-
2015-11-07.imgsafe
7609+1 records in
7609+1 records out
31914983424 bytes (32 GB) copied, 1675.51 s, 19.0 MB/s
Slartibartfast Raspbian # truncate --reference 2015-05-05-
raspbian-wheezy.img wheezy-2015-11-07.imgsafe
Slartibartfast Raspbian # diff -s wheezy-2015-11-07.imgsafe
2015-05-05-raspbian-wheezy.img
Files wheezy-2015-11-07.imgsafe and 2015-05-05-raspbian-
wheezy.img are identical
```

Je montre (ci-dessous) une commande « ls ». Je fais cela pour me rappeler le nom du fichier avec lequel je vais travailler et en avoir l'orthographe exacte.

Sur ma machine, le processus a pris 10 minutes au total.

La prochaine étape (ci-dessus) est tout à fait optionnelle, mais, si vous êtes comme moi, vous vous sentez le besoin de vérifier l'écriture et de vous assurer que cela correspond bien à l'image. Nous allons générer une image de la carte SD que nous venons d'écrire et l'envoyer dans un fichier image temporaire au disque dur. Puisque notre carte SD donnera un fichier plus gros que celui de l'image de la distribution, nous allons devoir la tronquer pour correspondre à la taille originale et enfin utiliser « diff » pour s'assurer que les deux images sont identiques. N'oubliez pas que cela peut prendre un certain temps, surtout si vous utilisez une carte plus grosse que 8 Go. Pour ma part, j'en utilise une de 32 Go et copier l'image sur le drive nécessitera probablement plus de 30 minutes.

```
Slartibartfast Raspbian # ls -al
total 7424016
drwxr-xr-x 2 greg greg       4096 Oct 31 12:02 .
drwxr-xr-x 3 greg greg       4096 Oct 23 20:11 ..
-rw-r--r-- 1 greg greg 3276800000 May  7  2015 2015-05-05-raspbian-wheezy.img
-rw-r--r-- 1 greg greg 4325376000 Sep 24 16:14 2015-09-24-raspbian-jessie.img
Slartibartfast Raspbian #
```

Comme on peut le voir, les images sont identiques et nous savons que, si

quelque chose se passe mal à partir de maintenant jusqu'au démarrage de la machine, ce n'est pas de notre faute. On peut retenir cette méthode à l'avenir pour faire une copie du « disque » de notre Pi, au cas où quelque chose se passerait mal.

Enfin, lançons la commande sync afin de vider le cache et être sûr que tout a été écrit avant de démonter la carte SD.

On peut faire maintenant quelque chose d'un peu plus « excitant » : allumer le Pi.

Se préparer à allumer votre RPi

Notez bien les mots utilisés pour donner un titre à ce chapitre. Il y a un certain nombre de choses que vous devez faire avant d'alimenter votre RPi. Il y a un risque de l'endommager si vous ne faites pas les choses dans le bon ordre.

Branchez le clavier et la souris dans le(s) port(s) USB.

Branchez le câble Ethernet dans le port Ethernet ou l'adaptateur Wi-Fi dans le port USB.

Allumez votre moniteur ou votre téléviseur et réglez-le sur la bonne entrée (HDMI ou Composite).

Branchez le câble vidéo (HDMI ou Composite).

Mettez la carte SD (ou micro SD) en place. Que vous utilisiez une carte SD ou une micro SD, vous devrez l'insérer avec l'étiquette tournée vers le bas, pas vers le haut, vers le fond du Pi. Et, quoi que vous fassiez, NE RETIREZ PAS la carte SD pendant que le RPi démarre.

Nous sommes maintenant prêts à allumer le RPi donc prenez une grande inspiration, croisez les doigts et branchez-le.

Si ça a marché, on peut continuer, sinon réessayez les instructions ci-dessus.

Quand le RPi démarre pour la première fois dans une distribution, il va basculer automatiquement sur l'application raspi-config (configuration du raspberry). Nous allons régler quelques paramètres. Cela ne se fait qu'une seule fois.

Un écran apparaît avec 9 options. Nous allons paramétrer les numéros 1, 3 et 4.

Option n° 1 – Demande si l'on veut étendre le fichier système. Faites-le, car cela vous donnera tout l'espace possible. Cette option ne sera effective qu'au prochain démarrage du système.

Option n° 3 – Permet de démarrer comme un ordinateur de bureau ou sans interface. Vous devez choisir ordinateur de bureau et vous identifier comme l'utilisateur « Pi » sur l'interface graphique.

Option n° 4 – Elle règle un certain nombre de choses qui se font automatiquement dans les programmes de configuration auxquels nous sommes habitués. Cela comprend la localisation, le fuseau horaire et la disposition du clavier.

• Choisir d'abord la localisation. Puisque cet ordinateur vient du Royaume-Uni, les valeurs par défaut correspondent à celles de quelqu'un vivant là-bas. En ce qui me concerne, j'ai besoin de changer un certain nombre de choses. Je suis descendu dans la liste jusqu'à EN_US.UTF-8 UTF-8 et l'ai sélectionné. Choisissez ce que l'appareil propose, ça conviendra.

• Ensuite, je dois indiquer mon fuseau horaire. Puisque j'habite aux USA, dans le Colorado, j'ai sélectionné l'Amérique dans la zone géographique et Denver comme fuseau horaire.

• Enfin j'ai dû choisir ma disposition clavier. Le système pose un tas de questions. J'ai donc répondu « Generic », « US », « US », « Default », « No Compose Key » et « Non » pour une clé permettant d'arrêter le serveur X.

Voilà, j'ai fini. J'ai donc sélectionné « Finish » et « yes ». Votre Pi doit redémarrer et vous devez voir l'espace de bureau normal. Nous souhaitons maintenant mettre à jour le système et installer quelques applications dont nous allons nous servir tout de suite, puis le laisser redémarrer une fois encore.

Ouvrir un terminal depuis la barre de menu supérieure et faites :

```
sudo apt-get update
sudo apt-get dist-upgrade
```

Nous allons installer maintenant TightVNCServer. Bien que ce ne soit pas obligatoire, je trouve beaucoup plus pratique d'utiliser une fenêtre dédiée sur mon ordinateur Linux plutôt que d'avoir deux écrans, deux claviers et deux souris. Cela me perturbe et je me demande sans arrêt sur quelle machine je suis.

```
sudo apt-get install
tightvncserver
```

Une fois installé, le programme vous demandera de créer un mot de passe pour protéger l'accès à votre écran. Choisissez-en un facile à mémoriser.

La chose suivante à faire sera de lancer TightVNCServer au démarrage. Ainsi nous n'aurons pas besoin de clavier ni de souris.

• Allez dans le répertoire home, si vous n'y êtes pas déjà :

```
$ cd /home/pi
```

• Ensuite allez dans le répertoire .config :

```
$ cd .config
```

• Nous créerons à cet endroit un nouveau répertoire appelé « autostart » :

```
$ mkdir autostart
```

• Déplaçons-nous dans le répertoire que nous venons de créer :

```
$ cd autostart
```

• Créons maintenant un nouveau fichier de configuration $ nano tightvnc.desktop où vous entrez les lignes suivantes :

```
[Desktop Entry]
Type=Application
Name=TightVNC
Exec=vncserver :1
StartupNotify=false
```

• Enregistrez le fichier (^O) et quittez (^X).

On a quasiment terminé maintenant. Il nous reste à installer l'IDE, qui s'appelle Geany, que nous allons utiliser pour notre développement.

```
sudo apt-get install geany
```

Allez sur votre ordinateur habituel et installez-y VNCViewer. Quand tout est terminé, prenez un petit moment pour redémarrer l'ordinateur et vous assurer que le VNC a bien démarré et s'est connecté. Si tout fonctionne correctement, vous avez fini.

Comme je l'ai dit au début, vous aurez besoin de quelques petites choses pour le mois prochain :
• quelques câbles mâle-mâle, femelle-femelle,
• la carte d'expérimentation (breadboard),
• la carte d'extension,
• le câble plat,
• une poignée de composants que vous achèterez en boutique électronique…,
• quelques petites LED. Essayez d'en acheter une dizaine de chaque : rouges, vertes, jaunes et transparentes.
• quelques résistances d'1/4 de watt de 220 Ω, 4,7 kΩ et 10 kΩ et d'autres résistances « habituelles » de bricoleurs. Là encore, une dizaine de chaque. Le vendeur de la boutique locale vous aidera à faire votre choix,
• quelques petits interrupteurs (spst) qui s'adaptent à la carte d'expérimentation et sont habituellement munis de 4 connecteurs.

C'est vraiment tout ce dont vous aurez besoin pour le prochain article. En attendant, amusez-vous avec Linux sur votre Pi. Je pense que vous serez surpris par la puissance de cette petite machine.

Jusqu'au mois prochain, je vais vous laisser avec un slogan que l'on entend souvent ici aux USA :

« Attendez… ce n'est pas fini !!!!! »

Greg Walters est propriétaire de Rainy-Day Solutions LLC, une société de consultants à Aurora au Colorado, et programme depuis 1972. Il aime faire la cuisine, marcher, la musique et passer du temps avec sa famille.

ubuntu uk podcast

Le Podcast Ubuntu couvre toutes les dernières nouvelles et les problèmes auxquels sont confrontés les utilisateurs de Linux Ubuntu et les fans du logiciel libre en général. La séance s'adresse aussi bien au nouvel utilisateur qu'au plus ancien codeur. Nos discussions portent sur le développement d'Ubuntu, mais ne sont pas trop techniques. Nous avons la chance d'avoir quelques supers invités, qui viennent nous parler directement des derniers développements passionnants sur lesquels ils travaillent, de telle façon que nous pouvons tous comprendre ! Nous parlons aussi de la communauté Ubuntu et de son actualité.

Le podcast est présenté par des membres de la communauté Ubuntu Linux du Royaume-Uni. Il est couvert par le Code de Conduite Ubuntu et est donc adapté à tous.

L'émission est diffusée en direct un mardi soir sur deux (heure anglaise) et est disponible au téléchargement le jour suivant.

http://ubuntupodcast.org

Programmer en Python - Partie 62

Au moment où vous lirez ces lignes, ce ne sera probablement plus d'actualité qu'un nouveau Rasperry Pi est sorti le 26 novembre 2015. Il s'agit du Raspberry Pi Zéro, au prix incroyable de 5$ US ou 4 £, ou encore, moins de 5€. Je n'ai pas réussi à trouver ses dimensions réelles, mais il paraît qu'il est de la taille d'un chewing-gum. Donc, si vous étiez freiné par le prix pour acheter votre nouveau Pi, vous n'avez maintenant plus d'excuse. Nous discuterons du Pi Zero dans de prochains articles.

Maintenant, revenons à ma série de programmation dans le monde réel. Cette fois-ci, nous allons commencer à vraiment contrôler les choses. J'espère que vous avez été en mesure de vous procurer des LED, des résistances, des interrupteurs, des cavaliers et une plaque à trous.

Durant cette série d'articles, je vais utiliser un outil de conception gratuit appelé Fritzing pour fournir une représentation visuelle de ce à quoi le câblage du projet devrait ressembler.

Vous pouvez vous en procurer un exemplaire sur leur site Web (http://fritzing.org/home/). Cela vous permettra non seulement de conserver localement des copies de nos projets, mais vous pourrez aussi vous amuser à concevoir vos propres circuits.

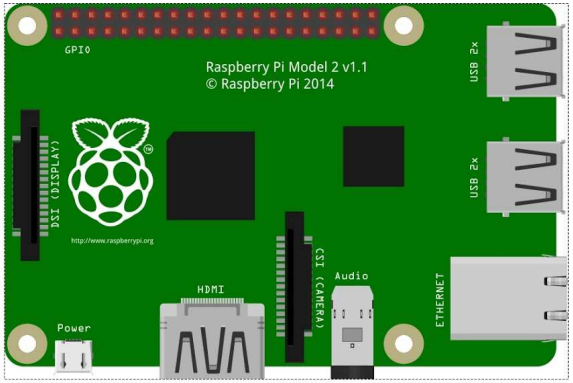
Petit tour de nos composants

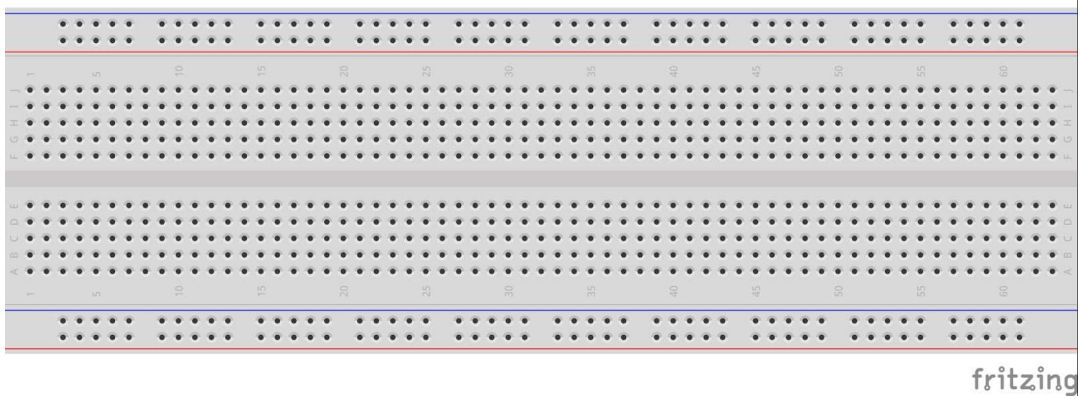
Une dernière chose avant de commencer : une discussion rapide sur certains des composants électroniques que nous utiliserons cette fois, des résistances, des LED et des interrupteurs.

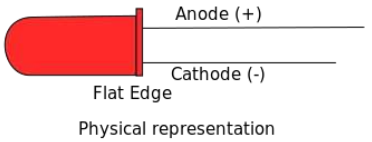
Resistances

Une résistance est un dispositif qui « résiste » plus ou moins à la circulation de l'électricité. Cela nous permettra de limiter la quantité d'électricité qui circule dans un circuit ou une partie de circuit. Dans le cas des projets avec des LED, nous utiliserons des résistances pour réduire la quantité d'électricité qui passe à travers la LED (et la broche GPIO [Ndt : pour les entrées/sorties]), pour l'empêcher de brûler.

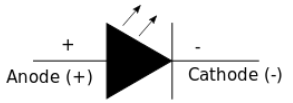
Pour une explication plus détaillée sur les résistances, vous pouvez regarder ici : https://learn.sparkfun.com/tutorials/resistors.







Physical representation



Schematic Representation

Les LED

Les LED sont des diodes électroluminescentes et sont les remplaçantes « standard » pour les ampoules de toutes sortes. En étant attentifs dans la conception, elles dureront presque éternellement. Une LED a deux

bornes/fils appelés anode et cathode. L'anode est le côté positif et la cathode est le côté négatif.

Si vous sortez une nouvelle LED de sa boîte, vous remarquerez que l'un des fils est plus long que l'autre. C'est l'anode ou le côté positif. Si les deux fils sur une LED neuve ont la même longueur (ou si vous recyclez des composants d'un ancien circuit), cherchez le côté plat. Ce sera toujours le côté de la cathode ou côté négatif.

Interrupteurs

L'interrupteur que j'ai choisi pour ce projet est un modèle qui se monte facilement dans la plaque à trous ou sur un circuit imprimé. Il est tout simplement carré avec un petit bouton rond sur le dessus. Il a également 4 broches. L'astuce est de trouver les deux broches dont nous avons besoin. Vous pourriez prendre un ohmmètre et essayer toutes les combinaisons de broches jusqu'à trouver un couple qui fonctionne, ou vous pouvez simplement regarder la disposition des broches qui le relient à la plaque à trous. Les deux broches à utiliser doivent s'insérer dans la plaque en se faisant face. Vous avez seulement besoin d'un couple de broches, choisissez celui que vous souhaitez.

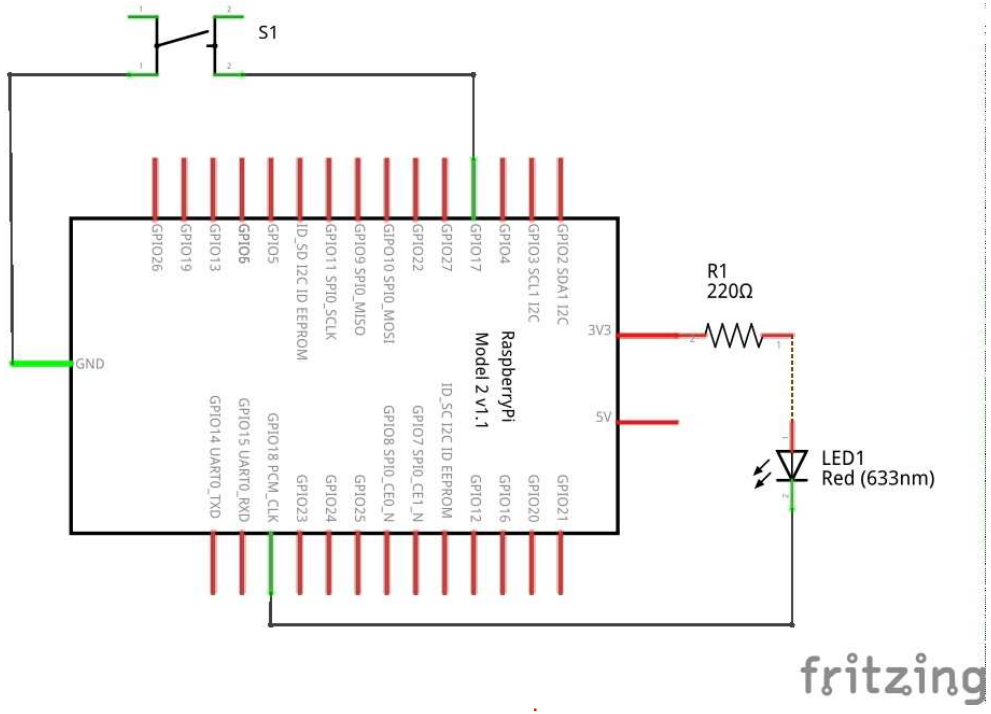
Notre premier projet...

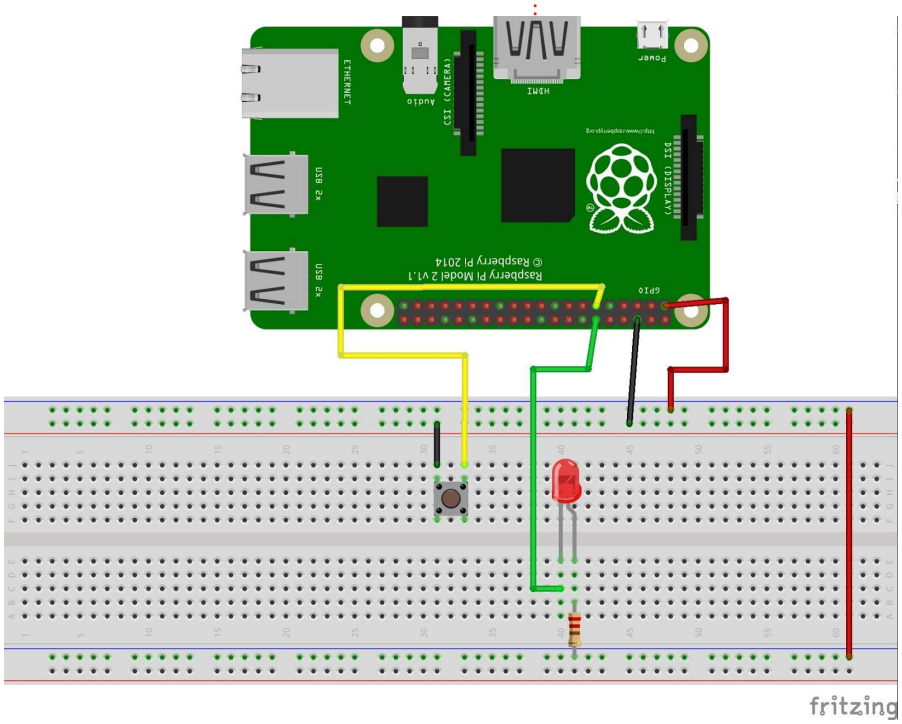
Maintenant, commençons notre premier projet de construction. C'est une version électronique très simple du « hello world » (« bonjour à tous »). Nous allons connecter un interrupteur à l'une des broches GPIO et la surveiller pour visualiser l'appui sur le bouton.

À droite se trouve le schéma exact avec lequel nous allons travailler.

Nous avons donc un interrupteur connecté entre la terre et la broche 17 du GPIO, qui est la broche physique 11. Nous avons aussi une LED connectée par sa cathode à la broche 18 du GPIO (broche physique 12) et dont l'anode est reliée à une résistance qui se connecte à la broche 3,3 V sur le Pi. C'est ici que vous devez prendre une décision. Allez-vous référencer les broches par leur position sur la carte ou par les numéros GPIO ? Nous y reviendrons dans une minute. En attendant, voici le schéma de câblage…

Vous pouvez voir sur la plaque les trois composants : l'interrupteur, la LED et la résistance. La première broche sur le RPi est en haut à droite. Cette broche fournit les 3,3 V dont nous avons besoin pour alimenter notre projet. La broche en-dessous est la





broche n° 2. La broche n° 6 est une broche de masse. Notez que ces deux broches sont connectées aux longs bus horizontaux sur la plaque à trous. Certaines plaques ont un « + » et un « - » sur le bus d'alimentation pour vous aider à les distinguer. J'ai aussi un long cavalier qui va du bus 3,3 V en haut de la plaque au bus du bas. Entre les deux, le bus que vous utilisez pour l'alimentation n'a vraiment pas d'importance, tant que vous êtes cohérent.

Il y a un cavalier court allant du bus de masse du haut à un côté de l'interrupteur, et l'autre côté de l'interrupteur est connecté à la broche physique 11 sur le RPi (ou broche 17 GPIO). Quant à la LED, la cathode est connectée à la broche physique 12 sur le RPi (18 du GPIO) et l'anode est connectée à la résistance, qui est à son tour reliée au bus 3,3 V inférieur. Notez également que le câblage est codé en couleur. Le rouge sera TOUJOURS (dans mes schémas) une tension positive, le noir étant pour la masse. Toutes les autres couleurs signifieront des interconnexions pour les données.

Si vous avez suivi jusqu'à présent, vous remarquerez que je donne à la fois le numéro de broche physique et le numéro de broche BCM GPIO. Le « BCM » signifie Broadcom, et, dans notre code, nous devrons indiquer à la bibliothèque RPi.GPIO quelle numérotation nous utilisons, physique ou BCM. D'où la décision dont je parlais plus tôt. Dans notre code, nous devrons être cohérents avec l'un ou l'autre système de numérotation. Dans le code que nous allons regarder, je fournis les deux, et vous pouvez commenter celui que vous ne voulez pas utiliser. Ma préférence personnelle est d'utiliser les numéros BCM GPIO, mais, pour ce projet, je vais rester sur les numéros physiques de broches. Maintenant, entrons dans le code.

Comme toujours, je vais découper le code en morceaux et expliquer chacun. D'abord (en haut à droite) nous devons importer la bibliothèque RPi.GPIO, et nous allons créer un alias « GPIO » pour rendre les choses plus faciles à saisir. Ensuite, nous définissons deux variables, LedPin et BtnPin, correspondant au schéma de numérotation des broches que nous souhaitons utiliser. Ici, je me suis décidé à utiliser la numérotation de broche physique, car vous n'avez probablement pas encore de nappe de connexion. J'ai trouvé celle de SparkFun très agréable, mais elle indique seulement le numéro BCM des broches. Notre prochain bout de code (ci-contre) sera une fonction appelée « setup », dans laquelle nous réglons les informations

```
import RPi.GPIO as GPIO

# If you are using the BCM GPIO pin numbers...
#LedPin = 18
#BtnPin = 17
# Otherwise the physical board numbers...
LedPin = 12
BtnPin = 11
```

sur la bibliothèque à utiliser.

Notez que la première ligne est commentée puisque je vais utiliser la numérotation de la carte dans cet exemple, mais elle est là pour vous montrer comment indiquer votre choix.

Les lignes 3 et 4 montrent comment définir le rôle des broches, entrée ou sortie, et si nous utilisons les résistances de rappel internes, intégrées sur le RPi, ou pas. Cette partie du code dit donc simplement d'utiliser les numéros physiques des broches comme références, et définit quelle broche de sortie pilotera la LED et par quelle broche le signal du bouton arrivera. Notez également que nous réglons la broche du bouton pour qu'elle ait une résistance de rappel. Cela signifie que le signal nominal sera à 3,3 V et que, lorsque le bouton est pressé, il sera ramené vers la masse.

Notre prochaine fonction (page suivante, en haut à droite) est appelée « loop » [Ndt : boucle] et, comme son nom l'indique, nous faisons simplement une boucle, pour vérifier si le signal sur la broche d'entrée du bouton a diminué. Si oui, alors nous allumons la LED, sinon nous relevons le LedPin. Cela peut sembler contre-intuitif, mais rappelez-vous que nous avons l'anode connectée au bus de 3,3 V à travers la résistance. Cela signifie que pour allumer la LED, nous devons ramener la cathode à la masse (0 V) pour permettre à la LED de s'allumer.

```
def setup():
        #GPIO.setmode(GPIO.BCM)
        GPIO.setmode(GPIO.BOARD)
        GPIO.setup(LedPin, GPIO.OUT)
        GPIO.setup(BtnPin, GPIO.IN,pull_up_down=GPIO.PUD_UP)
```

La fonction « destroy » (ci-dessous) nettoie simplement les états des broches, afin de ne pas recevoir d'erreurs la prochaine fois que nous allons les utiliser.

Enfin, nous utilisons la « boucle principale » (en bas) pour appeler les routines dans le bon ordre et pour permettre de sortir facilement de la boucle en appuyant sur Ctrl-C.

Voilà, chargez le programme dans votre RPi et exécutez-le. Vous remarquerez que le texte "… LED Off" se répète à l'écran jusqu'à ce que vous appuyiez sur le bouton. Cela est dû au fait que notre routine « loop » lit le niveau ou le statut de la broche du bouton, et une fois que la tension baisse, elle dit « oh… l'entrée du bouton est basse, donc je dois allumer la LED ».

Une autre chose à noter est que nos première et deuxième routines sont nommées « setup » et « loop ». C'est une bonne chose de garder ce format, car, quand nous arriverons à la programmation de l'Arduino, ces deux routines seront obligatoires.

Nous allons nous arrêter ici pour ce mois-ci, car je veux laisser de la place pour les articles d'autres auteurs. Gardez tout à portée de main, car nous allons utiliser la même configuration matérielle la prochaine fois.

Amusez-vous bien et je vous revois le mois prochain.

```
def loop():
        while True:
                if GPIO.input(BtnPin) == GPIO.LOW:
                        print('...LED On')
                        GPIO.output(LedPin,GPIO.LOW)
                else:
                        print('...LED Off')
                        GPIO.output(LedPin,GPIO.HIGH)
```

```
def destroy():
        GPIO.output(LedPin,GPIO.HIGH)
        GPIO.cleanup()
```

```
if __name__ == '__main__':
        setup()
        try:
                loop()
        except KeyboardInterrupt:
                destroy()
```

Greg Walters est propriétaire de Rainy-Day Solutions LLC, une société de consultants à Aurora au Colorado, et programme depuis 1972. Il aime faire la cuisine, marcher, la musique et passer du temps avec sa famille.

Le Podcast Ubuntu couvre toutes les dernières nouvelles et les problèmes auxquels sont confrontés les utilisateurs de Linux Ubuntu et les fans du logiciel libre en général. La séance s'adresse aussi bien au nouvel utilisateur qu'au plus ancien codeur. Nos discussions portent sur le développement d'Ubuntu, mais ne sont pas trop techniques. Nous avons la chance d'avoir quelques supers invités, qui viennent nous parler directement des derniers développements passionnants sur lesquels ils travaillent, de telle façon que nous pouvons tous comprendre ! Nous parlons aussi de la communauté Ubuntu et de son actualité.

Le podcast est présenté par des membres de la communauté Ubuntu Linux du Royaume-Uni. Il est couvert par le Code de Conduite Ubuntu et est donc adapté à tous.

L'émission est diffusée en direct un mardi soir sur deux (heure anglaise) et est disponible au téléchargement le jour suivant.

http://ubuntupodcast.org

Python dans le monde réel - P. 63

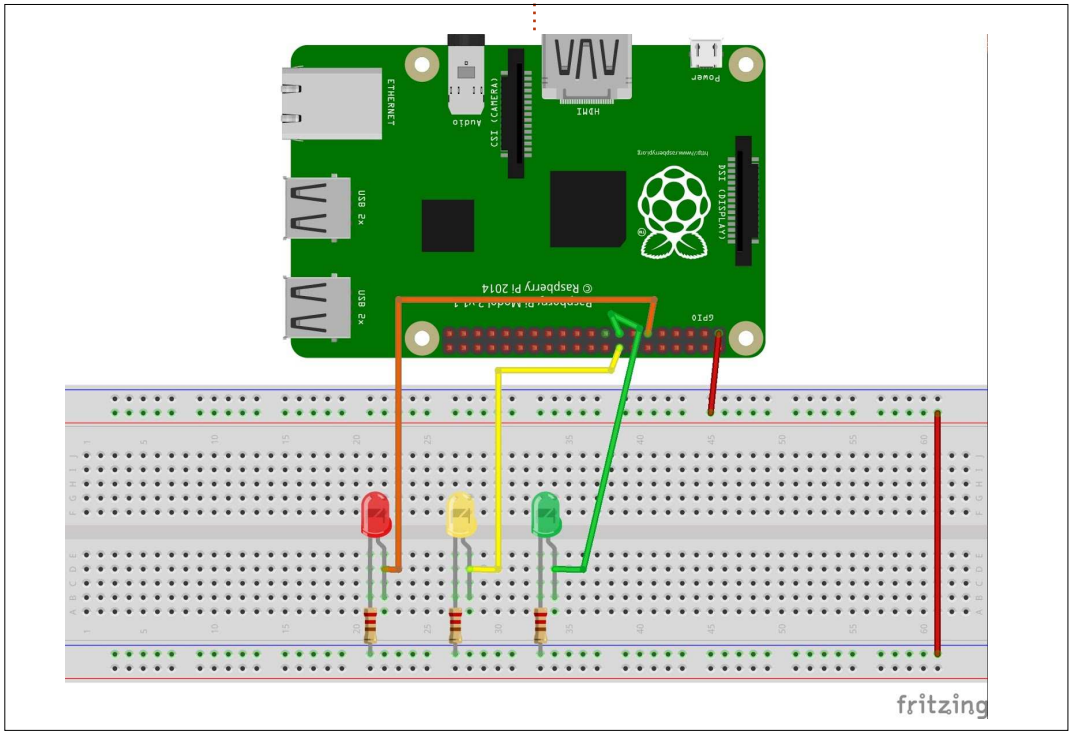
Heureux de vous revoir dans notre série sur la programmation dans le monde réel. La dernière fois, nous avons programmé le RPi pour allumer et éteindre une LED quand on a appuyé sur un bouton. Très simple, mais c'est un bon démarrage. Ce mois-ci, nous allons réaliser un autre projet simple, un simulateur de feux tricolores routiers utilisant 3 LED, une rouge, une jaune et une verte. Majoritairement, le code va être très proche de celui utilisé le mois dernier ; il ne devrait donc pas y avoir de problème. Si vous avez des questions, je vous suggère de regarder l'article du mois dernier qui devrait répondre à toutes ces interrogations.

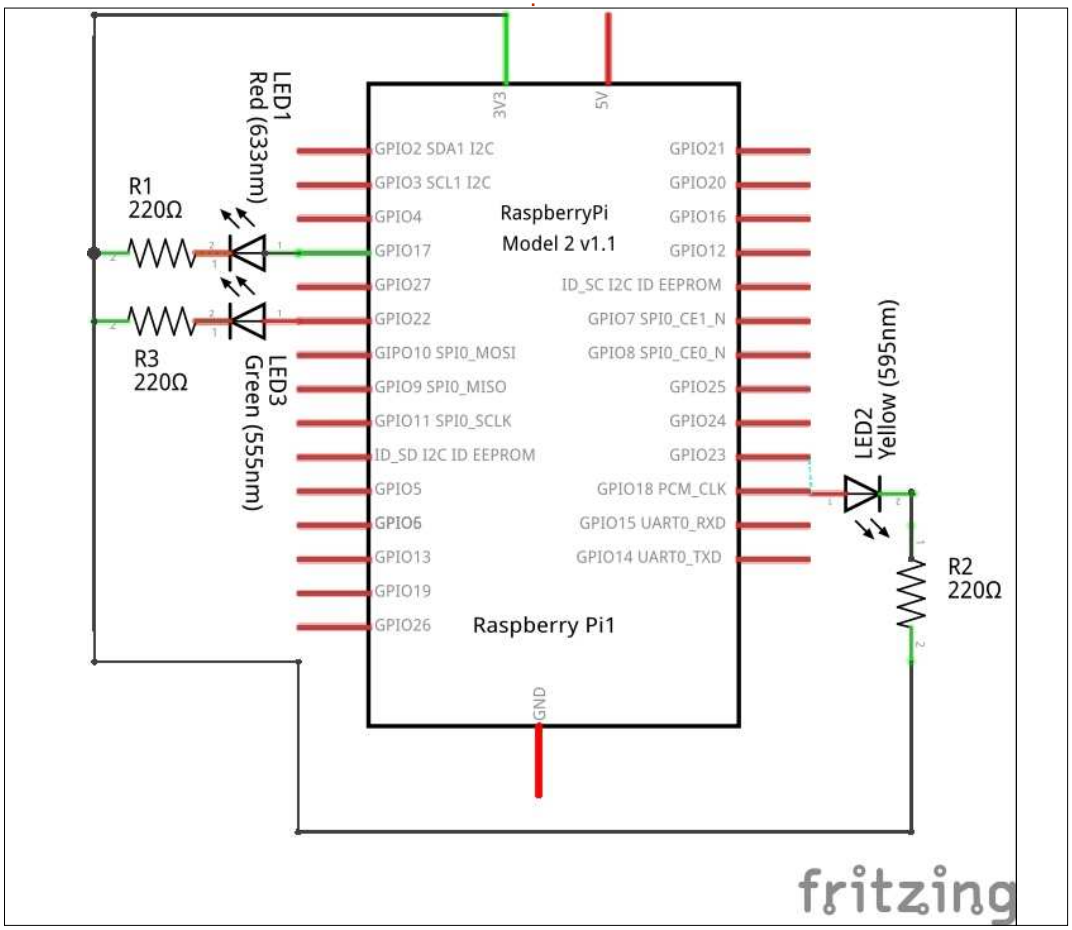
D'abord, regardons le schéma et la plaque d'essai (ci-dessous).

Notez que les couleurs des câbles correspondent à leur fonction, à l'exception du fil orange. Les fils rouges fournissent 3,3 volts. Le câble vert pilote la LED verte, le jaune, la LED jaune et le fil orange contrôle la LED rouge, puisque le fil rouge est déjà pris.

La cathode de la LED rouge est connecté à GPIO 17 (picot physique 11), la cathode de la LED jaune est connectée à GPIO 23 (picot physique 16) et la cathode de la LED verte est reliée à GPIO 22 (picot physique 15). Les anodes des trois LED sont connectées à une extrémité des résistances de 220 Ω dont les autres terminaisons sont reliées au point haut de 3,3 V. Nous n'avons pas besoin de la masse pour ce projet-ci.

Comme je n'ai conduit qu'aux États-Unis, j'ai basé la simulation sur la séquence US. Un long feu rouge (10 se-





condes), un vert habituellement plus court que le rouge (8 secondes) et un orange assez court (2 secondes). Ces valeurs sont actuellement codées en dur dans les appels de fonction time.sleep(). Vous pouvez les changer comme bon vous semble.

Maintenant, commençons à travailler sur le code.

```
#!/usr/bin/env python

# Traffic Light Simulator
# Written by G. D. Walters

#-------------------------

import RPi.GPIO as GPIO
import os
import time
import datetime

#-------------------------

RedLedPin = 17
YellowLedPin = 23
GreenLedPin = 22
```

Les 9 premières lignes sont des déclarations classiques d'import, plus quelques lignes de commentaires. Les 3 lignes suivantes définissent les numéros des contacts BCM de nos picots de LED. Si vous voulez utiliser les numéros physiques des picots, assurez-vous de changer la ligne GPIO.setmode() dans la routine suivante (en haut à droite).

Comme indiqué dans l'article précédent, GPIO.setmode doit être changé de « GPIO.BCM » en « GPIO.BOARD » si vous voulez utiliser les numéros des picots physiques à la place des numéros BCM dans nos définitions. Les trois lignes suivantes déclarent les picots de LED comme des sorties, puis les éteignent toutes les trois pour démarrer le programme, en plaçant la valeur de sortie à HIGH (haut).

```
def setup():
    GPIO.setmode(GPIO.BCM)                  # Numbers GPIOs by physical location

    GPIO.setup(RedLedPin, GPIO.OUT)         # Set the 3 LedPins mode as output
    GPIO.setup(YellowLedPin, GPIO.OUT)
    GPIO.setup(GreenLedPin, GPIO.OUT)

    GPIO.output(RedLedPin, GPIO.HIGH)       # Turn off LEDs
    GPIO.output(YellowLedPin, GPIO.HIGH)
    GPIO.output(GreenLedPin, GPIO.HIGH)
```

```
def LEDLoop():
    print "Green On..."

GPIO.output(GreenLedPin,0)
    time.sleep(8)
   GPIO.output(GreenLedPin,1)
    print "Green Off..."
    print "Yellow On..."
GPIO.output(YellowLedPin,0)
    time.sleep(2)

GPIO.output(YellowLedPin,1)
    print "Yellow Off..."
    print"Red On..."
    GPIO.output(RedLedPin, 0)
    time.sleep(10)
    GPIO.output(RedLedPin,1)
    print "Red Off..."
```

La routine LEDLoop est très simple :

- imprimer « <color> On… » (<couleur > allumée) sur la console,
- allumer la LED en mettant la valeur de sortie à 0 (niveau bas),
- ensuite une période d'attente de la valeur convenue,
- remettre la sortie à la valeur 1 (niveau haut),
- puis imprimer que la LED est éteinte.

Puis, ceci est copié pour les LED jaune et rouge. La routine loop() force simplement l'appel répété à l'infini de la routine LEDLoop() jusqu'à ce que l'uilisateur tape <Ctrl> C sur le clavier du RPi.

```
def loop():
    while True:
      LEDLoop()
```

La routine destroy() et la boucle principale sont les mêmes que le mois

```
def destroy():
    GPIO.output(RedLedPin, GPIO.HIGH)          # led off
    GPIO.output(YellowLedPin, GPIO.HIGH)       # led off
    GPIO.output(GreenLedPin, GPIO.HIGH)        # led off
    GPIO.cleanup()                             # Release resource

if __name__ == '__main__':          # Program start from here
    setup()
    try:
            loop()
    except KeyboardInterrupt:  # When 'Ctrl+C' is pressed, the child program destroy()
will be executed.
            destroy()
```

dernier : nous mettons toutes les sorties de LED au point haut, pour les éteindre, puis nous appelons GPIO.cleanup().

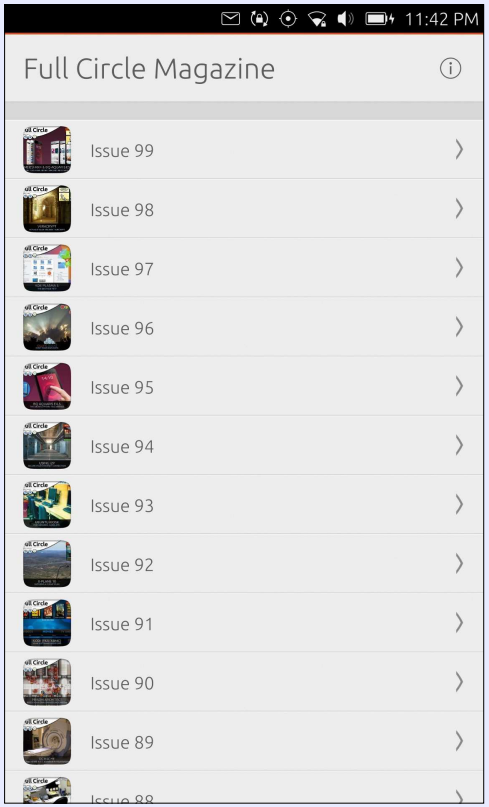
Je ne suis pas sûr que nous puissions construire un programme plus simple pour faire ce que nous devons faire.

Si vous voulez, vous pouvez dupliquer les 3 LED et programmer une simulation de carrefour avant la prochaine fois.

La prochaine fois, nous aurons quelque chose d'un peu plus corsé. Jusque-là, bonne programmation.

Greg Walters est propriétaire de RainyDay Solutions LLC, une société de consultants à Aurora au Colorado, et programme depuis 1972. Il aime faire la cuisine, marcher, la musique et passer du temps avec sa famille.

L'application officielle Full Circle pour Ubuntu Touch

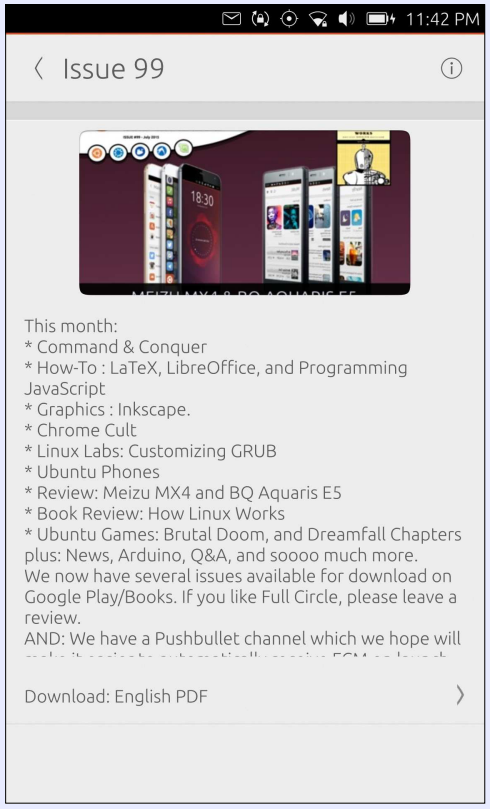


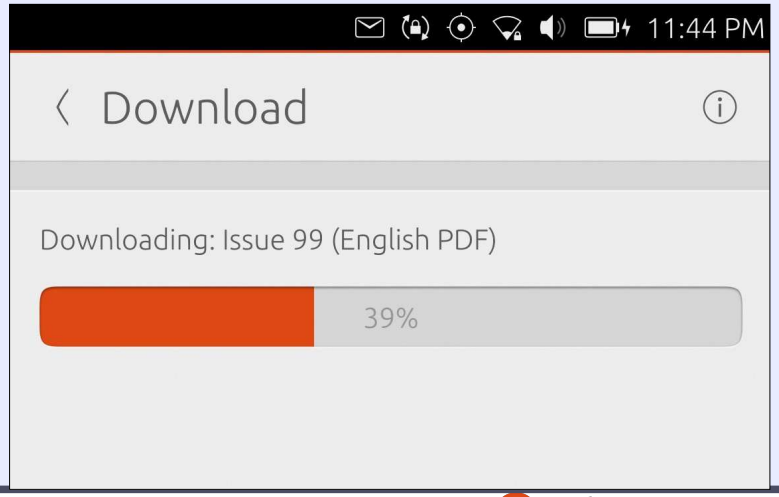
Brian Douglass a créé une appli fantastique pour les appareils Ubuntu Touch, qui vous permettra de voir les numéros actuels et les numéros plus anciens, et de les télécharger et de les lire sur votre téléphone/tablette Ubuntu Touch.

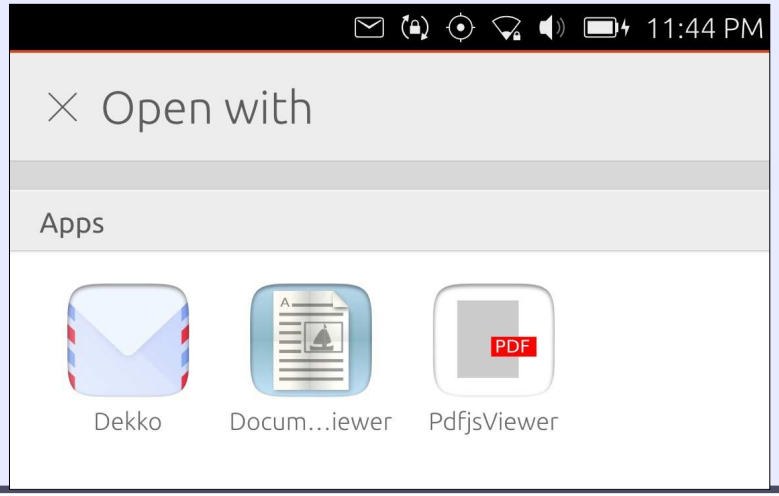
Installation

Soit vous cherchez « full circle » dans le magasin Ubuntu Touch et vous cliquez sur Installer, soit vous affichez l'URL ci-dessous sur votre appareil et vous cliquez sur Installer pour être transféré sur la page des téléchargements.

https://uappexplorer.com/app/fullcircle.bhdouglass







Bon retour dans le monde assez fou de Python Programming dans le monde réel. Avant de commencer, je dois vous avouer quelque chose. La dernière fois, j'ai fait une grosse faute : les images dans la partie 63 sont erronées. Les LED sont à l'inverse de ce qu'elles devraient être. Brian Kelly l'a remarqué et a été assez courageux pour signaler les erreurs du vieillard que je suis. Merci, Brian. Si vous suivez le texte, cela devrait aller.

Ensuite, je dois vous présenter des excuses, car je n'ai pas pu faire mon article le mois dernier (dans le FCM n° 106). J'ai encore des problèmes de santé qui m'empêchent de rester assis pendant longtemps. En principe, cela devrait être corrigé bientôt.

Bon, ça suffit. Maintenant voyons ce que je vous propose ce mois-ci.

La LED mystère

Dans les deux derniers articles, vous avez appris à allumer et à éteindre des LED avec un programme. C'était assez simple. Il s'agit de sortie numérique et non pas analogique. Le Rpi, contrairement à l'Arduino, ne supporte pas d'entrée/sortie analogique, et nous sommes limités à n'utiliser une sortie du GPIO (et une LED dans ce cas) que dans l'état allumé ou éteint. Cette fois-ci, nous allons utiliser nos connaissances pour faire quelque chose de très intéressant.

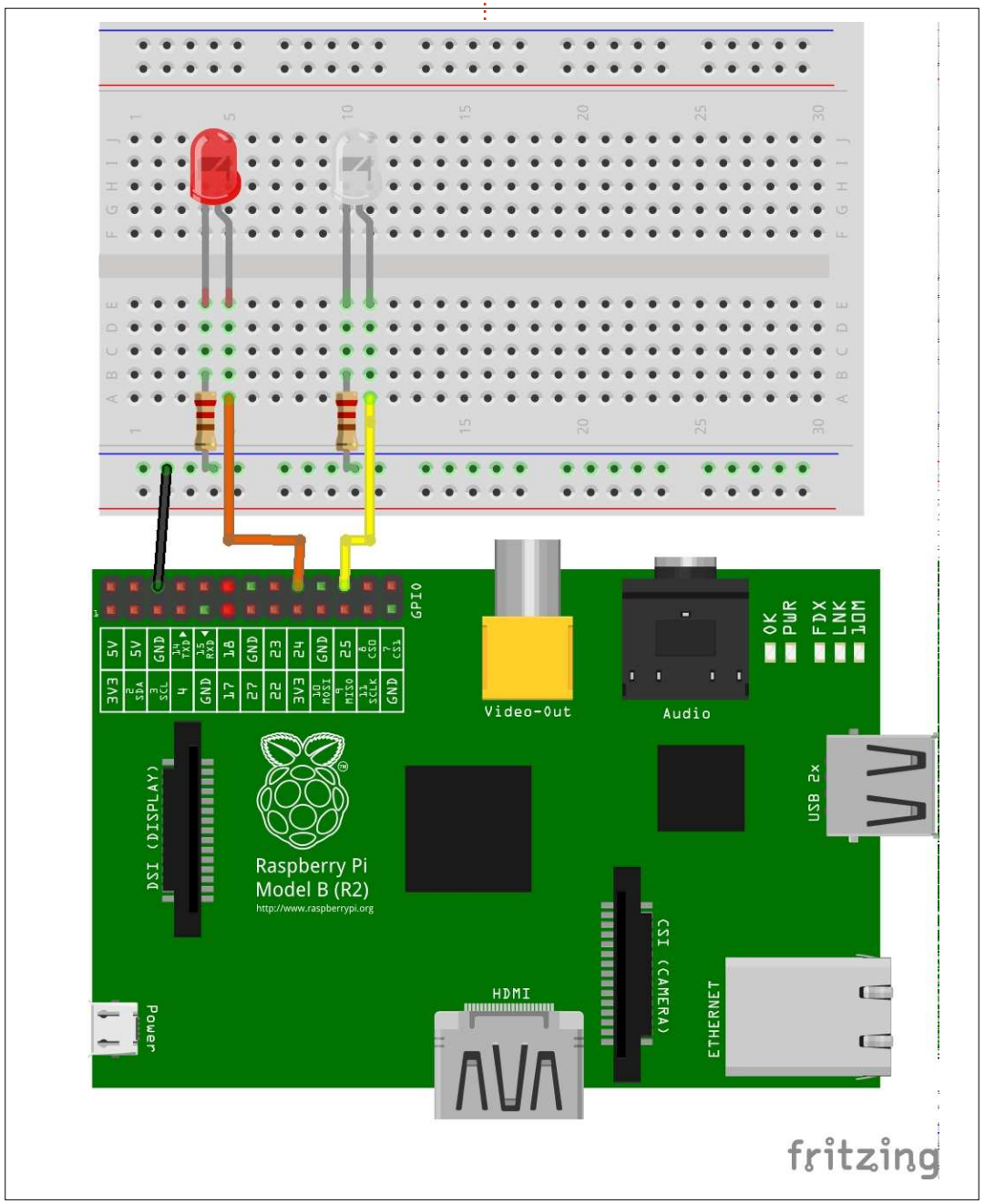
Allez chercher votre Pi, votre carte d'expérimentation et au travail !

Le câblage

Il vous faut un Raspberry Pi, une carte d'expérimentation, deux LED, une rouge et une blanche, deux résistances de 220 Ohms et 3 fils de raccordement.

Pour cet exemple de câblage, j'utilise le Pi original. Si vous avez un Pi B+ ou 2B (ou encore le tout nouveau 3), les broches concernées sont exactement les mêmes.

Pour éviter de s'emmêler les pinceaux (de mon côté), les Cathodes (côté Négatif) des LED sont connectées aux résistances, qui sont connectées, elles, à la terre, et les Anodes (côté Positif) sont connectées par les fils de raccordement aux broches du Pi. Le



côté positif de la LED est habituellement indiqué par la broche la plus longue et le côté négatif est celui qui a un méplat sur la base de la LED.

Le code

Je n'expliquerai pas le code tout de suite. Il suffit de le copier tel quel dans l'éditeur. J'en parlerai après.

Une fois le code entré correctement, lancez-le et regardez ce qui se passe.

```
import RPi.GPIO as GPIO
from time import sleep
GPIO.setmode(GPIO.BCM)
GPIO.setup(25,GPIO.OUT)
GPIO.setup(24,GPIO.OUT)
white = GPIO.PWM(25,100)
red = GPIO.PWM(24,100)
white.start(0) # start white led on 0 percent duty cycle (off)
red.start(100) # red fully on (100%)
pause_time = 0.05
print("Program Starting...Press CTRL+C to exit")
try:
    while True:
        for i in range(0,101): #101 because it stops when it finishes 100
                               #101 parce que ça s'arrête après l'exécution de 100
            white.ChangeDutyCycle(i)
            red.ChangeDutyCycle(100-i)
            sleep(pause_time)
        for i in range(100,-1,-1):
            white.ChangeDutyCycle(i)
            red.ChangeDutyCycle(100-i)
            sleep(pause_time)
except KeyboardInterrupt:
        white.stop()
        red.stop()
        GPIO.cleanup()
```

La révélation

Si vous avez prêté attention aux articles depuis le début, vous avez sans doute compris ce que fait le code. Sinon, ne vous tracassez pas : une explication suivra.

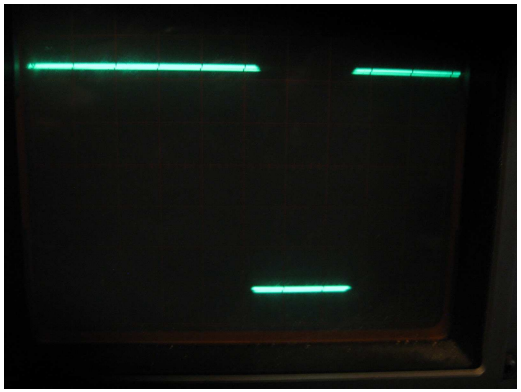
Les LED ne sont ni allumées ni éteintes, mais, à la place, elles basculent rapidement entre allumé et éteint. Puisque j'ai bien dit plus tôt que l'on ne peut émettre (ou lire) qu'un voltage On/Off (ou 1/0 ou High/Low), comment est-ce possible ?

Nous utilisons un truc appelé PWM ou Pulse Width Modulation (modulation de largeur d'impulsions). Nous respectons toujours les règles, mais nous les contournons un peu dans notre intérêt. Les images ci-dessous, de mon oscilloscope connecté au projet, devront vous aider à comprendre un peu mieux. À ce stade, nous nous occuperons d'une seule LED.

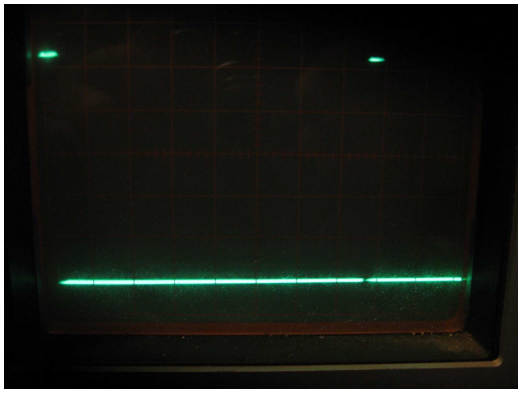
Si nous envoyons un Low sur la sortie GPIO de la LED, cela fait zéro volt. La LED ne reçoit rien sur l'Anode et elle est donc éteinte. Dans les deux derniers articles, nous avons allumé la LED en envoyant un High à l'Anode de la LED. Ainsi, dans la première instance, nous avons un zéro et, dans la seconde, un 1. Comme prévu… soit Off, soit On.

Cette fois-ci, nous varions le laps de temps où le signal GPIO est High et Low. Si nous le faisons lentement, les LED clignoteraient en réponse au voltage. Dans le cas de cette version, nous basculons entre les deux états très rapidement, tout en changeant le laps de temps pendant lequel il est On, comparé à Off, ce qui s'appelle le rapport cyclique.

Comme vous pouvez le voir, le signal est On pendant environ 80 % du temps et Off pendant 20 %, ce qui ferait un rapport cyclique de 80 %. En faisant cela rapidement, la LED réagit en s'assombrissant un peu en dessous



du 100 % permanent. Durant la boucle du programme, le rapport cyclique change et le High est rendu plus long ou plus court, selon telle ou telle



partie de la boucle. Dans l'image ci-dessus, le rapport cyclique est d'environ 5 %. Dans ce cas, la LED est allumée pendant si peu de temps qu'elle est extrêmement sombre et, pour ainsi dire, elle est Off.

Maintenant, examinons le code en détail.

```
import RPi.GPIO as GPIO from
time import sleep
```

Comme toujours, nous commençons par les imports. Nous importons la bibliothèque GPIO et, cette fois-ci, nous importons la fonction sleep (sommeil) de la bibliothèque time (temps). Vous allez rapidement comprendre pourquoi.

```
GPIO.setmode(GPIO.BCM)
GPIO.setup(25,GPIO.OUT)
GPIO.setup(24,GPIO.OUT)
white = GPIO.PWM(25,100)
red = GPIO.PWM(24,100)
```

Dans ces cinq lignes, nous avons réglé le mode GPIO sur BCM et configuré le pins GPIO 24 (pin physique 9) et 25 (pin physique 11) comme pins de sortie. Nous avons déjà fait cela. Maintenant, nous réglons les valeurs du PWM à un rapport cyclique de 100 %.

```
white.start(0) # Démarrer la
LED blanche avec un rapport
cyclique de 0% (off)

red.start(100) # La rouge
allumée en permanence (100%)
```

Ensuite, nous allumons la LED rouge (100%) et réglons la LED blanche à 0 volts.

```
pause_time = 0.05

print("Program Starting…Press
CTRL+C to exit")
try:
    while True:
        for i in range(0,101): # 101 because it stops when it finishes 100
                               # 101 parce que ça s'arrête après l'exécution de 100
            white.ChangeDutyCycle(i)
            red.ChangeDutyCycle(100-i)
            sleep(pause_time)
        for i in range(100,-1,-1):
            white.ChangeDutyCycle(i)
            red.ChangeDutyCycle(100-i)
            sleep(pause_time)
except KeyboardInterrupt:
        white.stop()
        red.stop()
        GPIO.cleanup()
```

Nous réglons la variable pause_time à 0,05 seconde. Cela la rend assez rapide pour (je l'espère) ne pas permettre un scintillement.

Dans le segment de code suivant, nous faisons les boucles. L'objectif de la première boucle est de rendre la LED blanche plus « brillante » et la LED rouge plus « sombre ». Celui de la deuxième est d'inverser le processus. Prenons la première boucle comme exemple : nous utilisons une boucle FOR pour régler la valeur de i, puis mettons le rapport cyclique de la LED blanche à i et celui de la LED rouge à 100-i.

Remarquez que nous l'avons entouré d'un ensemble TRY…EXCEPT, ce qui permet au programme de s'exécuter jusqu'à ce que l'utilisateur fasse CTRL+C. Quand cela arrive, nous quittons le côté TRY afin de pouvoir faire le code de nettoyage.

Ainsi, vous savez maintenant que nous avons la possibilité de contourner les règles dans notre intérêt.

La prochaine fois, nous commencerons notre examen d'une autre bibliothèque GPIO. En attendant, amusez-vous bien.

Greg Walters est propriétaire de RainyDay Solutions LLC, une société de consultants à Aurora au Colorado, et programme depuis 1972. Il aime faire la cuisine, marcher, la musique et passer du temps avec sa famille.

TUTORIEL

Écrit par Greg D. Walters

Python dans le monde réel - P. 65

Bienvenue. Je vais vous donner plein d'informations en vrac ce mois-ci. La raison principale est que des améliorations techniques importantes se font jour et que vous allez avoir besoin d'un peu de temps pour approvisionner des pièces pour les prochains articles.

Dans un futur proche, nous ajouterons l'Arduino dans notre boîte à outils. Je suggère de démarrer avec le modèle UNO ou une de ses copies que l'on trouve à moins de 30 $ (22 £ ou 26,32 €). Nous allons avoir aussi besoin de quelques capteurs pour pouvoir vraiment fonctionner. Bien que ces derniers soient optionnels et que vous puissiez vous contenter de lire l'article, la construction représente la plus grande partie du plaisir. Bon, cela dit, voici une liste de composants :

• Un capteur digital de température One Wire – DS18B20.
• Un capteur simple de température et d'humidité DHT11.
• Un afficheur LCD 16×2.
• Des résistances de 4,7 et 10 kΩ 0,25 W (3 ou 4 de chaque).
• Une grande plaque d'essai (60+ x 10 avec des lignes d'alimentation).
• Des potentiomètres de 10 kΩ (2 ou 3).
• Des connecteurs mâles-femelles (du Pi vers la plaque d'essai) d'environ 8″ (20 cm).
• Des connecteurs mâles-mâles (de l'Arduino vers la plaque d'essai) d'environ 8″ (20 cm).
• Des connecteurs mâles-mâles (de la plaque d'essai vers la plaque d'essai), de taille petite à moyenne.
• Un moteur de jouet ou de modélisme 6V CC.
• Un circuit de contrôle de moteurs L293D ou SN754410.
• Un support pour 4 piles AA et les piles correspondantes.

Cela vous suffira fort bien pour les prochains mois. Bien sûr, vous pouvez en acquérir plus et faire vos propres découvertes. La plupart des éléments de cette liste coûtent moins de 10 $. Si vous faites vos achats sur Internet avec soin, vous pouvez tout obtenir à très bon prix. Laissons cela pour le moment, mais, pour la prochaine fois, vous aurez besoin du capteur de température DS18B20 et d'une résistance de 4,7 kΩ ainsi que d'une plaque d'essai et de connecteurs, si vous n'en avez pas déjà.

Il y a eu beaucoup de bruit sur Internet récemment disant que le logiciel Alexa de l'appareil Echo d'Amazon était porté sur Raspberry Pi. La raison principale de l'émoi était due au fait que Echo/Alexa n'est disponible qu'aux États-Unis pour l'instant alors que de nombreuses personnes au Royaume-Uni et dans d'autres pays l'attendent impatiemment. Cela leur donne la possibilité d'apprécier la technologie.

Il y a au moins deux projets qui travaillent à porter Echo sur le Pi. Le premier utilise Java. Les codes et les instructions se trouvent à : https://github.com/amzn/alexa-avs-raspberry-pi. J'ai essayé ce projet sur un Pi version 1B et le nouveau Pi 3B. Il fonctionne bien sur les deux. De nombreuses personnes rencontrent des problèmes pour le faire fonctionner, mais je l'ai fait en 4 heures environ (avec de petites pauses et des interruptions) et ça a marché du premier coup. Le meilleur conseil que je peux vous donner est de prendre votre temps, d'être prêt à passer tout un week-end dessus et de suivre les instructions à la lettre. Mon seul problème était que je devais installer npm et nvm et, qu'à cette époque, les instructions n'étaient pas incluses. Je pense que ce problème a été corrigé.

Le second projet utilise Python et se trouve à : https://github.com/lenny sh/AlexaPi. Pour être honnête, je l'ai essayé, mais ne suis pas arrivé à le faire fonctionner. Je dois vous avouer que je n'ai pas passé autant de temps sur ce projet que sur la version Java à cause de nombreuses visites médicales ces dernières semaines. J'ai l'intention d'y passer un peu plus de temps pour arriver à le faire fonctionner.

Si vous décidez d'essayer l'un ou l'autre des projets, S'IL VOUS PLAÎT, utilisez une carte SD vierge et non pas une qui possède des données que vous voudriez conserver. Chargez l'OS Raspbian ou NOOBS à partir de zéro. De cette façon, si ça se passe mal, vous pourrez simplement recharger l'OS et recommencer au début.

Il y a un certain nombre de choses que vous devez savoir avant de commencer ce projet. Toutes les informations ci-dessous concernent la version java mais certaines peuvent s'appliquer aux deux projets :

• Vous avez besoin d'un microphone USB. Les micros de casques posent problème. J'utilise le micro d'une webcam Logitech et ça fonctionne bien.

• Vous aurez également besoin d'un ensemble de hauts-parleurs - ou un casque - reliés à la prise jack de sortie audio. De

nombreuses personnes rencontrent des difficultés avec des équipements audio bluetooth.

• Vous aurez besoin d'un bouton-poussoir pour que l'Echo/Alexa écoute votre commande. Il ne réagit pas pour le moment au mot « wake » (réveille-toi - voir plus loin).

• Certaines fonctions de l'Echo/Alexa original ne fonctionnent pas actuellement.

• Des choses comme les lieux, le temps, la circulation ne fonctionnent que pour les USA. Dans tout autre pays, vous récupérerez les informations de Seattle, État de Washington, USA.

• Le seul langage supporté à l'heure actuelle est l'anglais. Suivant les informations que j'ai pu glaner dans mes recherches, une fois que l'appareil sera vendu dans un pays, ils ajouteront la langue « officielle » de ce pays. J'en déduis qu'au Royaume-Uni, la langue officielle est l'anglais et qu'aux USA il n'y a pas de langue « officielle » et que l'espagnol qui y est très largement parlé n'est pas encore supporté sur l'appareil. Il y a beaucoup de fils de conversation enflammés sur la toile, si vous voulez exprimer votre colère parce que votre langue n'est pas supportée ou que l'Echo/Alexa n'est pas disponible chez vous. Je ne peux que vous suggérer d'être patient. L'appareil n'avait pas beaucoup de succès et les ventes ont décollé très récemment. Je suis sûr qu'Amazon travaille déjà au support d'autres pays.

• Quand vous démarrez l'application, vous devez lancer deux processus. Le second va créer, dans une fenêtre de dialogue, une longue chaîne de caractères en forme d'URL que vous devez copier et coller dans un navigateur. Lorsque cela arrive correctement chez Amazon, vous devez cliquer sur le bouton [OK] qui s'affiche à l'écran. Apparaîtra alors un bouton [Start Listening] (commencer l'écoute) et quelques boutons multimédia. Pour « réveiller » Alexa, vous devez cliquer sur le bouton « Start Listening » et, après avoir entendu le « ding », énoncer votre question ou votre commande. Une fois terminé, vous pouvez rappuyer sur ce bouton pour forcer la fin de l'écoute ou attendre un délai (environ 5 secondes) avant le début du traitement. De nombreuses personnes travaillent sur un procédé non commandé (sans écran) et un bouton physique connecté sur un port d'entrée/sortie, alors que d'autres travaillent carrément sur l'option orale « wake ». Vous trouverez des informations complémentaires dans la partie réservée aux problèmes.

• Vous devez (ABSOLUMENT) utiliser une carte SD de bonne qualité. Je suggère d'utiliser au moins une carte de classe 10 d'au minimum 16 Go.

• Dès que vous démarrerez sur le nouvel OS pour la première fois, lancez :

```
sudo raspi-config
```

Assurez-vous bien d'occuper la totalité de la carte avec le fichier système. Vérifiez également que vous avez bien mis en route le SSH. Il faudra alors redémarrer. Puis, vous devrez faire :

```
sudo apt-get update
```

et :

```
sudo apt-get dist-upgrade
```

pour obtenir la dernière version du logiciel.

• Quelques étapes vous demandent d'entrer certaines données. Prenez bien note de ce que vous avez saisi, soit par une copie d'écran, soit à l'aide de votre smartphone ou (HORREUR!!!!!) sur une feuille de papier. Cela vous facilitera les choses.

• Si vous rencontrez quelque souci que ce soit, regardez la section des problèmes. Il est fort probable que quelqu'un ait déjà rencontré un problème identique et qu'il existe une solution.

• Imprimez la page Internet comportant les instructions et travaillez à partir de là. De cette façon vous pourrez cocher les tâches effectuées. Particulièrement utile si vous êtes interrompu.

• Vous trouverez plus d'informations à alexa.amazon.com et pourrez changer certains réglages. J'ai cru comprendre que quelques personnes qui ne résident pas aux USA rencontraient des problèmes avec ce site.

Je pense que cela suffit pour ce mois-ci ; le mois prochain, nous transformerons notre RPi en thermomètre. Une des particularités intéressantes du capteur DS18B20 est de pouvoir en connecter plus d'un sur une seule ligne. Ainsi, vous pouvez en avoir un dans le salon, un dehors, etc. Nous utiliserons plus tard ces capteurs avec l'Arduino et pourrons utiliser ce dernier avec une connexion à distance, ce qui nous évitera de faire des essais avec un long câble et en modifier la résistance à tel point que ça ne fonctionnera plus.

Jusqu'au mois prochain, amusez-vous avec le projet Alexa et, si vous en essayez un, ou les deux, je vous souhaite beaucoup de succès.

Greg Walters est propriétaire de RainyDay Solutions LLC, une société de consultants à Aurora au Colorado, et programme depuis 1972. Il aime faire la cuisine, marcher, la musique et passer du temps avec sa famille.

Le mois dernier, je vous suggérais de vous procurer un certain nombre de pièces et si vous avez été capable de les avoir, j'espère qu'elles ne vous ont pas coûté trop cher. Si vous n'en avez pas, suivez donc du mieux possible et si vous voulez essayer un projet particulier, alors procurez-vous les composants nécessaires. J'essaie de faire ceci avec une sortie minime de fonds, pour vous comme pour moi. Fréquemment, vous pouvez recycler de nombreux éléments issus de vieux matériels informatiques ; beaucoup peuvent être trouvés dans un magasin local d'occasion avec une bonne remise, quelques « pences » par livre sterling. (J'espère que je l'ai bien écrit. Par ici, on dit « pennies on a dollar » (des pennies par dollar) ; aussi, donnez-moi au moins un « AB » pour l'effort… d'accord ?)

Alors que j'étais allongé la semaine dernière, dans l'attente d'une intervention chirurgicale, je me demandais quelle pourrait être ma réponse, si je rencontrais quelqu'un qui me demandais pourquoi je fais tout ça. Avant les merveilleux produits chimiques qui m'ont été injectés pour rendre le processus moins terrible, je réalisais que la VRAIE raison est multiple. Premièrement, c'est pour créer un intérêt chez les « non-programmeurs » en faisant des choses qui, apparemment, ne pourraient pas être réalisées sans une tonne de formation. Deuxièmement, il est démontré que la dernière technologie, comme le Raspberry Pi et l'Arduino, n'est pas incompréhensible au « mec ordinaire » du coin, mais que tout un chacun peut faire des choses qui ont des applications dans le monde réel (d'où le titre de cette série). Cela étant dit, faire clignoter une LED n'est que le même genre de projet pour le monde physique que le programme « Hello World » pour l'univers de la programmation. Vous devez faire des petits pas avant de passer au marathon. Croyez-moi, nous ferons des choses étonnantes avec toutes ces petites pièces, bidules et trucs-machins.

Ce mois-ci, nous utiliserons le capteur de température/humidité simple DHT11 avec notre Raspbery Pi. Le mois prochain, nous ferons quelque chose de similaire en utilisant le capteur de température Dallas DS18B20 et, s'il y a du temps et/ou de l'espace, nous parlerons aussi de l'afficheur LCD 16×2. Dans quelques mois, nous passerons du Raspberry Pi à l'Arduino. Ne vous inquiétez pas, aucune des choses que nous utilisons maintenant ne sera utilisée pour un seul projet. Par exemple, une fois que nous aurons compris les bases de l'Arduino (qui inclura l'apprentissage d'un peu de « C » [désolé pour ça]), nous écrirons des programmes en Python sur le RPi (ou votre ordinateur personnel) pour piloter l'Arduino. Les capteurs que vous avez appris à connaître dans nos expériences avec le RPi seront réutilisés quand nous nous formerons à l'Arduino, et beaucoup seront incorporés dans de plus grands projets. Très prochainement, nous utiliserons des moteurs à courant continu, des solénoïdes et des moteurs pas-à-pas dans quelques projets vraiment élémentaires, mais nous les utiliserons dans des projets plus grands, y compris dans la construction d'un graveur laser piloté par ordinateur (RPi) en utilisant une diode laser récupérée dans un vieux graveur de DVD.

C'est assez pour le futur. Commençons notre projet du mois.

Le DHT11 est l'élément le moins cher d'une famille de capteurs de température et d'humidité. Le DHT11 a une plage de température de 0 à 50 °C avec une précision de ± 2 ° (32 à 122 ºF, ± 3.6 ºF) et une plage d'humidité de 20-90 %RH ± 5 %. Vous pouvez voir que ce n'est pas le capteur le plus précis du marché ; le DHT22 est plus précis et a une gamme plus étendue (gamme de température de -40 à 80 ºC) mais coûte deux fois plus cher.

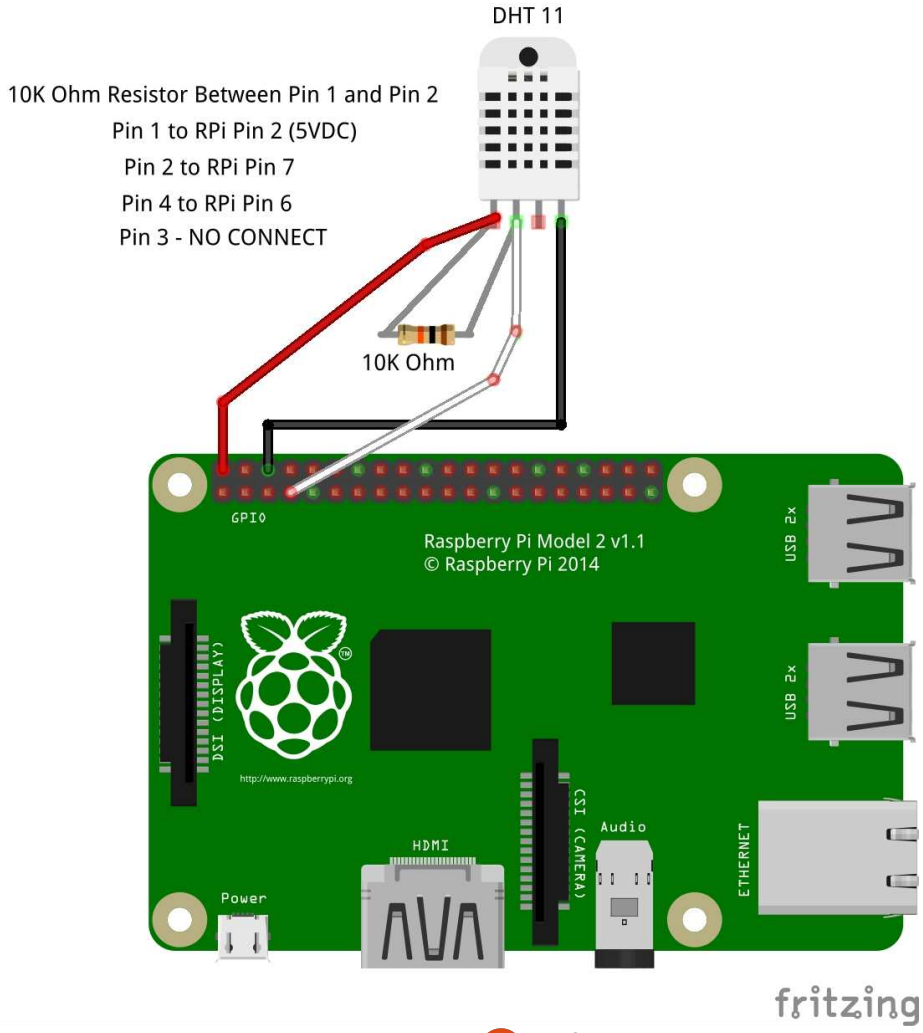
C'est une drôle de pièce. Un boîtier plastique rectangulaire bleu avec des trous et quelque chose de brillant à l'intérieur. Il peut être livré, soit seul avec quatre picots, soit sur un petit circuit imprimé avec 3 ou 4 picots. Quel que soit le format, c'est la même chose à la base. Pour le moment, nous utiliserons le composant discret (seul, sans le circuit imprimé), pour faciliter la présentation et je traiterai les différences au fur et à mesure.

Chaque fois que vous voudrez travailler avec un nouveau capteur, vous devrez prendre la feuille de spécifications (la « data sheet »). Une simple recherche sur le Web devrait retourner un bon nombre de résultats. Essayez de trouver quelque chose venant directement du fabricant si c'est possible. Pour le DHT11, un bon endroit pour

se procurer une des nombreuses data sheets disponibles est http://www.micropik.com/PDF/dht11.pdf. Bien qu'elle ne soit pas du fabricant lui-même, elle est fournie par une société qui le vend et qui a « traduit » les données du fabricant en un fichier PDF de 9 pages.

Vous devrez déjà vous demander : pourquoi en ai-je besoin ? C'est un ensemble d'informations dont vous n'aurez jamais besoin, sauf si vous avez un doctorat en physique ou quelque chose comme ça. Oui, c'est vrai, mais il y a beaucoup d'informations qui sont pertinentes et qui peuvent potentiellement vous éviter de faire exploser, soit le capteur, soit le contrôleur, soit votre banc d'essai. Dans notre cas, nous trouvons que la tension d'alimentation continue doit être entre 3 et 5 volts et qu'il consomme environ 0,5 mA dans des conditions « normales » (voir section 6). Nous trouvons aussi que ce capteur est plutôt lent et qu'il ne faudra pas essayer de recueillir plus d'une valeur par seconde. Essentiellement, nous en prendrons une toutes les cinq secondes environ dans notre programme de test, ce qui est bien plus que ce dont nous aurons réellement besoin. Autre chose : si le câble qui transmet les données du capteur vers le microcontrôleur (notre RPi) est inférieur à 20 mètres, nous devons mettre une résistance de tirage de 5 kΩ entre la ligne de données et l'alimentation du capteur. Une dernière chose (je m'arrête ici, mais il y en a d'autres) : la tension positive va sur le picot 1, les données sont sur le picot 2 et la masse sur le picot 4. Avec ceci, nous avons à peu près tout le nécessaire pour savoir le connecter à notre RPi en toute sécurité. Ci-dessous, voici le schéma de câblage pour un capteur DHT11 « seul » SANS circuit support. Si votre capteur est monté sur circuit, lisez mes explications à gauche du schéma.

Notez que j'ai dit avant qu'il y avait besoin d'une résistance de tirage de 5kΩ. Si vous utilisez une alimentation de 3,3 V DC (le picot 1 du RPi), alors la résistance de 5kΩ marche bien. Cependant, si vous choisissez 5 V DC, comme sur le schéma, utilisez une résistance de 10kΩ.



Vous pouvez voir que c'est plutôt simple, juste trois fils et une résistance. Pour notre projet simpliste, cependant, n'essayez pas avec tous les 20 mètres du fil.

Si votre DHT11 est sur circuit, vous devriez avoir au moins 3 picots. J'ai deux capteurs de marchands différents et (allez savoir pourquoi) ils ont une disposition des picots différente. Sur l'un je trouve [Data] (données) [Positive Voltage] (alim +) [Ground] (alim -) et c'est marqué « S - ». Pour l'autre, [Ground] [Data] [Positive Voltage], marqué ainsi. J'espère que, pour le vôtre, la définition de la disposition des picots est plus ou moins imprimée dessus. Si non, vous pouvez essayer avec un multimètre de suivre la piste entre la masse du capteur et le picot du circuit ; de même pour le +. Vous pouvez en général deviner que, s'il y a trois picots de sortie sur le circuit et que vous connaissez déjà la masse et l'alim, le dernier DEVRAIT ÊTRE celui des données.

Maintenant, notre code de programme.

Pour une mise en route rapide, nous utiliserons du code fourni par des gens sur Adafruit.com - ils fournissent une bibliothèque fonctionnant avec le DHT11. (Ils trouvent que d'essayer de faire tourner la bibliothèque directement en code Python entraîne certains problèmes ; aussi, la bibliothèque est écrite en « C ».) Il y a un certain nombre d'étapes et il faut donc suivre les instructions avec beaucoup d'attention. Je les ai paraphrasées ; ainsi, si quelque chose ne fonctionne pas, vous pouvez trouver aussi les instructions sur le site Web d'Adafruit à : https://learn.adafruit.com/dht-humidity-sensing-on-raspberry-pi-with-gdocs-logging/software-install-updated.

Quand tout est fait, vous pouvez faire tourner mon exemple modifié en Python présenté à la fin des instructions.

Dans votre répertoire « /home/pi », lancez les commandes suivantes :

```
git clone
https://github.com/adafruit/A
dafruit_Python_DHT.git

cd Adafruit_Python_DHT

sudo apt-get update

sudo apt-get install build-
essential python-dev python-
openssl
```

```
#!/usr/bin/python
# simpletest.py
#----------------------------------------------------
# Original code information copyright below.
# Copyright (c) 2014 Adafruit Industries
# Author: Tony DiCola
# Permission is hereby granted, free of charge, to any person obtaining a copy
# of this software and associated documentation files (the "Software"), to deal
# in the Software without restriction, including without limitation the rights
# to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
# copies of the Software, and to permit persons to whom the Software is
# furnished to do so, subject to the following conditions:
# The above copyright notice and this permission notice shall be included in all
# copies or substantial portions of the Software.
# Code modifications by G.D. Walters for Full Circle Magazine
import Adafruit_DHT
from time import sleep
#----------------------------------------------------
# Sensor should be set to Adafruit_DHT.DHT11,
# Adafruit_DHT.DHT22, or Adafruit_DHT.AM2302.
#sensor = Adafruit_DHT.DHT22
#----------------------------------------------------
sensor = Adafruit_DHT.DHT11
#----------------------------------------------------
```

Ignorez toutes les erreurs qui font état d'un paquet déjà installé.

Ensuite, installez la bibliothèque en lançant :

```
sudo python setup.py install
```

Une fois que tout est fait, vous pouvez passer à notre code exemple.

Ci-dessus, voici mon échantillon de code modifié, « emprunté » au code exemple d'Adafruit.

Tout ce qui est au-dessus pourrait être réduit à trois lignes de code. Les deux déclarations d'import et l'assignation de la variable « sensor » (capteur) au code de la classe.

```
pin = 4 sleep(3)
```

Ici, nous définissons que le capteur est connecté au picot 4 du GPIO et que nous attendons 3 secondes pour que tout s'établisse et soit prêt à fonctionner.

Nous utilisons une simple boucle (page suivante, en haut à gauche) pour récupérer les valeurs d'humidité et de température en permanence. Je n'ai jamais trouvé le truc reliant les Celsius à la température « réelle » ; aussi, je la convertis de telle sorte que je la comprends. Si vous voulez des Celsius, commentez simplement la ligne de conversion.

Maintenant (page suivante, en bas à gauche), nous vérifions que les valeurs de température et d'humidité sont réalistes, puis nous les affichons et attendons 5 secondes.

Je dois admettre que, quand je lance le programme avec un certain

```
#---------------------------------------------------
# Here we loop over and over getting and displaying the
data.
# Use <Ctrl><C> to break out.
#---------------------------------------------------
while 1:
    # Try to grab a sensor reading.  Use the read_retry
method which will retry up
    # to 15 times to get a sensor reading (waiting 2
seconds between each retry).

    humidity, temperature =
Adafruit_DHT.read_retry(sensor, pin)

    # Comment out the next line to display Celsius
    temperature = temperature * 9/5.0 + 32
```

capteur, j'obtiens des résultats plutôt délirants pendant les deux ou trois premières minutes, puis des valeurs crédibles s'établissent. L'autre capteur semble se « positionner » plus rapidement ; aussi, je pense que le premier capteur doit avoir un petit problème.

Bon, c'est tout pour ce mois-ci. Souvenez-vous que, la prochaine fois, nous utiliserons le capteur de température de Dallas ; soyez prêt.

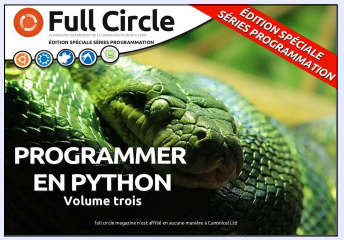
Amusez-vous bien et au mois prochain.

Greg Walters est propriétaire de RainyDay Solutions LLC, une société de consultants à Aurora au Colorado, et programme depuis 1972. Il aime faire la cuisine, marcher, la musique et passer du temps avec sa famille.

Éditions Spéciales Python :

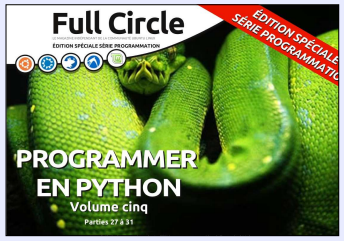
http://www.fullcirclemag.fr/?download/224

http://www.fullcirclemag.fr/?download/230



http://www.fullcirclemag.fr/?download/231

http://www.fullcirclemag.fr/?download/240



http://www.fullcirclemag.fr/?download/268

http://www.fullcirclemag.fr/?download/272

http://www.fullcirclemag.fr/?download/370

http://www.fullcirclemag.fr/?download/371

http://www.fullcirclemag.fr/?download/372

Écrire pour le Full Circle Magazine

Lignes directrices

Notre seule règle : tout article **doit avoir un quelconque rapport avec Ubuntu ou avec l'une de ses dérivées (Kubuntu, Xubuntu, Lubuntu, etc.).**

Autres règles

• Les articles ne sont pas limités en mots, mais il faut savoir que de longs articles peuvent paraître comme série dans plusieurs numéros.

• Pour des conseils, veuillez vous référer au guide officiel *Official Full Circle Style Guide* ici : http://url.fullcirclemagazine.org/75d471

• Utilisez n'importe quel logiciel de traitement de texte pour écrire votre article – je recommande LibreOffice –, mais le plus important est d'en **VÉRIFIER L'ORTHOGRAPHE ET LA GRAMMAIRE !**

• Dans l'article veuillez nous faire savoir l'emplacement souhaité pour une image spécifique en indiquant le nom de l'image dans un nouveau paragraphe ou en l'intégrant dans le document ODT (OpenOffice/LibreOffice).

• Les images doivent être en format JPG, de 800 pixels de large au maximum et d'un niveau de compression réduit.

• Ne pas utiliser des tableaux ou toute sorte de formatage en **gras** ou *italique*.

Lorsque vous êtes prêt à présenter l'article, envoyez-le par courriel à : articles@fullcirclemagazine.org.

Si vous écrivez une critique, veuillez suivre ces lignes directrices :

Traductions

Si vous aimeriez traduire le Full Circle dans votre langue maternelle, veuillez envoyer un courriel à ronnie@fullcirclemagazine.org et soit nous vous mettrons en contact avec une équipe existante, soit nous pourrons vous donner accès au texte brut que vous pourrez traduire. Lorsque vous aurez terminé un PDF, vous pourrez télécharger votre fichier vers le site principal du Full Circle.

Auteurs francophones

Si votre langue maternelle n'est pas l'anglais, mais le français, ne vous inquiétez pas. Bien que les articles soient encore trop longs et difficiles pour nous, l'équipe de traduction du FCM-fr vous propose de traduire vos « Questions » ou « Courriers » de la langue de Molière à celle de Shakespeare et de vous les renvoyer. Libre à vous de la/les faire parvenir à l'adresse mail *ad hoc* du Full Circle en « v.o. ». Si l'idée de participer à cette nouvelle expérience vous tente, envoyez votre question ou votre courriel à :

webmaster@fullcirclemag.fr

Écrire pour le FCM français

Si vous souhaitez contribuer au FCM, mais que vous ne pouvez pas écrire en anglais, faites-nous parvenir vos articles, ils seront publiés en français dans l'édition française du FCM.

CRITIQUES

Jeux/Applications

Si vous faites une critique de jeux ou d'applications, veuillez noter de façon claire :

• le titre du jeu ;
• qui l'a créé ;
• s'il est en téléchargement gratuit ou payant ;
• où l'obtenir (donner l'URL du téléchargement ou du site) ;
• s'il est natif sous Linux ou s'il utilise Wine ;
• une note sur cinq ;
• un résumé avec les bons et les mauvais points.

Matériel

Si vous faites une critique du matériel veuillez noter de façon claire :

• constructeur et modèle ;
• dans quelle catégorie vous le mettriez ;
• les quelques problèmes techniques éventuels que vous auriez rencontrés à l'utilisation ;
• s'il est facile de le faire fonctionner sous Linux ;
• si des pilotes Windows ont été nécessaires ;
• une note sur cinq ;
• un résumé avec les bons et les mauvais points.

Pas besoin d'être un expert pour écrire un article ; écrivez au sujet des jeux, des applications et du matériel que vous utilisez tous les jours.

MÉCÈNES

MÉCÈNES MENSUELS

2016:
Bill Berninghausen
Jack McMahon
Linda P
Remke Schuurmans
Norman Phillips
Tom Rausner
Charles Battersby
Tom Bell
Oscar Rivera
Alex Crabtree
Ray Spain
Richard Underwood
Charles Anderson
Ricardo Coalla
Chris Giltnane
William von Hagen
Mark Shuttleworth
Juan Ortiz
Joe Gulizia
Kevin Raulins
Doug Bruce
Pekka Niemi
Rob Fitzgerald
Brian M Murray
Roy Milner
Brian Bogdan
Scott Mack
Dennis Mack
John Helmers
JT
Elizabeth K. Joseph
Vincent Jobard
Chris Giltnane
Joao Cantinho Lopes
John Andrews

2017:

DONS UNIQUES

2016:
John Niendorf
Daniel Witzel
Douglas Brown
Donald Altman
Patrick Scango
Tony Wood
Paul Miller
Colin McCubbin
Randy Brinson
John Fromm
Graham Driver
Chris Burmajster
Steven McKee
Manuel Rey Garcia
Alejandro Carmona Ligeon
siniša vidović
Glenn Heaton
Louis W Adams Jr
Raul Thomas
Pascal Lemaitre
PONG Wai Hing
Denis Millar
Elio Crivello
Rene Hogan
Kevin Potter
Marcos Alvarez Costales
Raymond Mccarthy
Max Catterwell
Frank Dinger
Paul Weed
Jaideep Tibrewala
Patrick Martindale
Antonino Ruggiero
Andrew Taylor

2017:
Linda Prinsen
Shashank Sharma
Glenn Heaton
Frank Dinger

CHA CHA CHA CHANGEMENT

Notre administrateur est parti, pour de nombreux mois, sans rien dire à personne et je ne savais pas du tout, ni si, ni quand, les frais du site seraient ou ne seraient pas payés. Au départ, nous devions déménager le nom de domaine et le site, qui aurait été hébergé chez moi, et, finalement, j'ai réussi à retrouver l'admin et à me faire transférer le nom de domaine ainsi que l'hébergement du site.

Le nouveau site fonctionne dès à présent. D'ÉNORMES remerciements à Lucas Westermann (Monsieur Command & Conquer) d'avoir bien voulu prendre du temps sur ses loisirs pour recréer complètement le site, ainsi que les scripts, à partir de zéro.

J'ai fait la page Patreon pour pouvoir recevoir de l'aide financière pour ce qui concerne le domaine et les frais d'hébergement. L'objectif annuel a été atteint rapidement grâce à ceux dont les noms figurent sur cette page. Pas d'inquiétude à avoir : le FCM ne va pas disparaître. Plusieurs personnes ont demandé une option PayPal (pour un don ponctuel) et j'ai donc rajouté un bouton sur le côté du site.

Merci infiniment à tous ceux qui ont utilisé Patreon et le bouton PayPal. Cela m'a beaucoup aidé.

https://www.patreon.com/fullcirclemagazine

COMMENT CONTRIBUER

FULL CIRCLE A BESOIN DE VOUS !

Un magazine n'en est pas un sans articles et Full Circle n'échappe pas à cette règle. Nous avons besoin de vos opinions, de vos bureaux et de vos histoires. Nous avons aussi besoin de critiques (jeux, applications et matériels), de tutoriels (sur K/X/L/Ubuntu), de tout ce que vous pourriez vouloir communiquer aux autres utilisateurs de *buntu. Envoyez vos articles à :

articles@fullcirclemagazine.org

Nous sommes constamment à la recherche de nouveaux articles pour le Full Circle. Pour de l'aide et des conseils, veuillez consulter l'Official Full Circle Style Guide :
http://url.fullcirclemagazine.org/75d471

Envoyez vos **remarques** ou vos **expériences** sous Linux à : letters@fullcirclemagazine.org

Les tests de **matériels/logiciels** doivent être envoyés à : reviews@fullcirclemagazine.org

Envoyez vos **questions** pour la rubrique Q&R à : questions@fullcirclemagazine.org

et les **captures d'écran** pour « Mon bureau » à : misc@fullcirclemagazine.org

Si vous avez des questions, visitez notre forum : fullcirclemagazine.org

FCM n° 122

Date de parution du numéro en langue anglaise :
Vendredi 30 juin 2017.

Équipe Full Circle

Rédacteur en chef - Ronnie Tucker
ronnie@fullcirclemagazine.org

Webmaster - Lucas Westermann
admin@fullcirclemagazine.org

Correction et Relecture
Mike Kennedy, Gord Campbell, Robert Orsino, Josh Hertel, Bert Jerred, Jim Dyer et Emily Gonyer

Remerciements à Canonical, aux nombreuses équipes de traduction dans le monde entier et à **Thorsten Wilms** pour le logo du FCM.

Pour la traduction française :
http://www.fullcirclemag.fr

Pour nous envoyer vos articles en français pour l'édition française :
webmaster@fullcirclemag.fr

Pour les Actus hebdomadaires du Full Circle :

Vous pouvez vous tenir au courant des Actus hebdomadaires en utilisant le flux RSS : http://fullcirclemagazine.org/feed/podcast

Ou, si vous êtes souvent en déplacement, vous pouvez obtenir les Actus hebdomadaires sur Stitcher Radio (Android/iOS/web) : http: www.stitcher.com/s?fid=85347&refid=stpr

et sur TuneIn à : http://tunein.com/radio/Full-Circle-Weekly-News-p855064/

Obtenir le Full Circle en français :

http://www.fullcirclemag.fr/?pages/Numéros

Obtenir le Full Circle Magazine :

Format EPUB - Les éditions récentes du Full Circle comportent un lien vers le fichier epub sur la page de téléchargements. Si vous avez des problèmes, vous pouvez envoyer un courriel à : mobile@fullcirclemagazine.org

Issuu - Vous avez la possibilité de lire le Full Circle en ligne via Issuu : http://issuu.com/fullcirclemagazine. N'hésitez surtout pas à partager et à noter le FCM, pour aider à le faire connaître ainsi qu' Ubuntu Linux.

Magzster - Vous pouvez aussi lire le Full Circle online via Magzster : http://www.magzter.com/publishers/Full-Circle. N'hésitez surtout pas à partager et à noter le FCM, pour aider à le faire connaître ainsi qu'Ubuntu Linux.