À LA DÉCOUVERTE DU
RASPBERRY PI

Matt Richardson

Shawn Wallace



EYROLLES

SERIAL MAKERS

Petit mais costaud !

Avec plus de deux millions d'exemplaires écoulés en moins de deux ans, le Raspberry Pi est incontestablement le roi des nano-ordinateurs. De la taille d'une carte de crédit, sans écran ni clavier, et d'un prix modique, il s'avère pourtant suffisamment puissant et ouvert pour offrir une grande variété d'applications : robot, station météo, serveur web, média center, PC de bureau...

Cet ouvrage vous permettra d'appréhender le formidable potentiel de cet ordinateur, avec à la clé de petits programmes et des projets simples à réaliser. Vous serez ainsi amené à coder en Python et en Scratch, à manipuler des sons et des images avec Pygame, à communiquer via les broches d'entrées-sorties, à intégrer des périphériques, et beaucoup d'autres choses encore.

À qui s'adresse ce livre ?

- Aux makers, développeurs, électroniciens, hackers, designers, bricoleurs, bidouilleurs, artistes...
- Aux amateurs comme aux professionnels

Au sommaire

Présentation du Raspberry Pi et premier démarrage • Linux pour le Raspberry Pi • Programmer en Python • Animation et multimédia avec Python • Programmer en Scratch • Raspberry et Arduino • Les broches d'entrées-sorties • Programmation des entrées-sorties avec Python • Utiliser des webcams • Accéder à Internet avec Python.

Sur www.serialmakers.com

- Téléchargez le code source des exemples de l'ouvrage
- Consultez les compléments (liens utiles, news, etc.)

Fou de technologie, **Matt Richardson** est producteur vidéo et contributeur du magazine *Make*.

Shawn Wallace est écrivain et designer d'open hardware au Modern Device de Providence, aux États-Unis.

Make:
makezine.com

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

SERIAL MAKERS
www.serialmakers.com

www.editions-eyrolles.com

Code éditeur : G13747
ISBN : 978-2-212-13747-7
Couverture : Studio Eyrolles / Gijsbert Peijs FlickR
http://www.flickr.com/photos/gijsbertpeijs

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

À LA DÉCOUVERTE DU
RASPBERRY PI

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Dans la collection « Serial Makers »

C. Platt. – **L'électronique en pratique.**
N°13507, 2013, 344 pages.

E. Bartmann. – **Le grand livre d'Arduino.**
N°13701, 2014, 490 pages.

M. Berchon. – **L'impression 3D.**
N°13522, 2013, 200 pages.

B. Pettis, A. Kaziunas France et J. Shergill. – **Imprimer en 3D avec la MakerBot.**
N°13748, 2013, 266 pages.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Matt Richardson

Shawn Wallace

À LA DÉCOUVERTE DU RASPBERRY PI

EYROLLES

SERIAL MAKERS

ÉDITIONS EYROLLES
61, bd Saint-Germain
75240 Paris Cedex 05
www.editions-eyrolles.com

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Authorized French translation of the English edition of *Getting started with Raspberry Pi.*
ISBN 978-1-449-34421-4 © 2013 Matt Richardson and Shawn Wallace, published by Maker Media Inc.
This translation is published and sold by permission of O'Reilly Media, Inc., which owns or controls all rights to sell the same.

Traduction autorisée de l'ouvrage en langue anglaise intitulé *Getting started with Raspberry Pi* de Matt Richardson et Shawn Wallace (ISBN : 978-1-449-34421-4), publié par Maker Media, Inc.

Adapté de l'anglais par Janet Hoffmann et Yann Guidon

En application de la loi du 11 mars 1957, il est interdit de reproduire intégralement ou partiellement le présent ouvrage, sur quelque support que ce soit, sans l'autorisation de l'Éditeur ou du Centre Français d'exploitation du droit de copie, 20, rue des Grands Augustins, 75006 Paris.
© Groupe Eyrolles, 2014, ISBN : 978-2-212-13747-7

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

REMERCIEMENTS

Merci à tous ceux qui nous ont fourni leurs connaissances, leur aide, leurs conseils et leurs retours sur cet ouvrage : Brian Jepson, Marc de Vinck, Eben Upton, Tom Igoe, Clay Shirky, John Schimmel, Phillip Torrone, Limor Fried, Kevin Townsend, Ali Sajjadi et Andrew Rossi.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

AVANT-PROPOS

On peut comprendre pourquoi l'annonce du Raspberry Pi a laissé pas mal de monde sceptique. Un ordinateur de la taille d'une carte de crédit, vendu à 35 €, c'était trop beau pour être vrai. C'est pour cela qu'avant même de sortir d'usine, le Raspberry Pi croulait déjà sous les commandes.

Pendant des mois, la demande a largement dépassé l'offre, et la liste d'attente pour ces micro-ordinateurs s'allongeait de jour en jour. Hormis pour le prix, pourquoi une foule de passionnés était-elle prête à attendre si longtemps ? Mais avant de découvrir ce qui rend le Raspberry Pi si fantastique, parlons un peu du public auquel il est destiné.

En 2006, Eben Upton et ses collègues de l'université de Cambridge ont remarqué que les aspirants à un diplôme en informatique avaient moins d'expérience que ceux des années 1990. Ils ont attribué ce recul, entre autres facteurs, à « l'émergence chez les particuliers du PC et des consoles de jeu qui ont remplacé les Amiga, BBC Micro, Spectrum ZX et Commodore 64, des machines avec lesquelles la génération précédente avait appris à programmer ». Le fait que l'ordinateur soit devenu si important pour tous les membres du foyer peut aussi décourager les jeunes de s'aventurer dans ce domaine, à cause du risque d'endommager un outil important pour toute la famille. Mais récemment, les technologies qui ont permis aux téléphones mobiles et tablettes de devenir de moins en

moins chers et de plus en plus puissants, ont projeté le Raspberry Pi dans le monde des cartes « ultra-économiques-mais-super-faciles-à-utiliser ». Dans une interview pour BBC News, Linus Torvalds, fondateur du noyau Linux, a même dit que le Raspberry Pi « rend l'échec abordable ».

Que peut-on faire avec un Raspberry Pi ?

L'une des grandes forces du Raspberry Pi, c'est qu'il peut être utilisé dans une grande variété de projets. C'est une plate-forme souple, de divertissement, de travail ou d'expérimentation, vous permettant aussi bien de regarder des vidéos et de surfer sur le Web, que de bidouiller, d'apprendre ou de réaliser des montages électroniques. Voici quelques-unes de ses applications.

UN VÉRITABLE ORDINATEUR

Il ne faut pas oublier que le Raspberry Pi est avant tout un ordinateur et qu'il peut donc être utilisé comme tel. Après son installation, décrite au chapitre 1, vous pouvez choisir de le démarrer dans un environnement de bureau graphique incluant un navigateur web, ce qui constitue la manière la plus courante d'utiliser un ordinateur de nos jours. Au-delà du Web, vous pouvez installer une grande variété de logiciels libres et gratuits comme la suite bureautique LibreOffice (http://www.libreoffice.org) qui permet de travailler sur des documents texte et des feuilles de calculs quand vous n'avez pas de connexion à Internet.

UN SUPPORT D'APPRENTISSAGE DE LA PROGRAMMATION

Le Raspberry Pi étant à l'origine un outil pédagogique pour encourager l'apprentissage de l'informatique, il est fourni avec des interpréteurs et des compilateurs pour différents langages de programmation. En particulier, pour le débutant, il y a Scratch, un langage de programmation graphique créé par le MIT, qui sera présenté au chapitre 5. Mais si vous voulez commencer à coder tout de suite, le langage de programmation Python est un bon moyen de vous lancer : vous en trouverez les bases au

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

chapitre 3. Vous pouvez également programmer sur votre Raspberry Pi dans d'autres langages tels que C, Ruby, Java ou Perl.

UNE PLATE-FORME POUR INTERAGIR AVEC LE MONDE PHYSIQUE

Le Raspberry Pi se différencie d'un ordinateur standard non seulement par son prix et sa taille mais aussi par sa capacité à s'intégrer dans des projets électroniques. À partir du chapitre 7, nous vous montrerons comment utiliser le Raspberry Pi pour contrôler des LED et des appareils connectés au secteur, ainsi que pour lire l'état de boutons et d'interrupteurs.

UN OUTIL INDISPENSABLE POUR LES MAKERS

Les makers et bidouilleurs disposent déjà d'un très grand choix de plates-formes et de briques électroniques pour construire des projets utilisant les nouvelles technologies. Les cartes à microcontrôleur comme Arduino sont récemment devenues très populaires grâce à leur simplicité d'utilisation. Mais les nouvelles plates-formes à base de *System on a Chip* (SoC) comme le Raspberry Pi diffèrent des microcontrôleurs traditionnels sur de nombreux points. En réalité, un Raspberry Pi partage plus de caractéristiques avec votre ordinateur qu'avec Arduino.

Cela ne veut pas dire que le Raspberry Pi est mieux qu'un microcontrôleur traditionnel, ils sont tout simplement différents. Par exemple, si vous voulez réaliser un simple thermostat, il sera probablement plus approprié d'utiliser un Arduino Uno ou un microcontrôleur similaire, pour des raisons de simplicité, de coût et de consommation. Mais si vous souhaitez pouvoir modifier les paramètres de ce thermostat par le Web et télécharger son historique dans un fichier, le Raspberry Pi constituera un meilleur choix.

Le choix de l'un ou l'autre dépend des spécifications et des contraintes de votre projet, mais vous n'êtes pas nécessairement obligé de choisir entre les deux. Au chapitre 6, vous verrez comment utiliser le Raspberry Pi pour programmer Arduino et comment les faire communiquer ensemble.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

UN LECTEUR MULTIMÉDIA

Grâce à ses sorties vidéo (HDMI et composite), le Raspberry Pi peut être facilement branché sur un téléviseur. Il est aussi doté d'une puissance processeur suffisante pour décoder une vidéo haute définition en plein écran. Tirant parti de ces avantages, les contributeurs de XBMC (http://www.xbmc.org), un lecteur multimédia libre et open source, ont porté leur projet sur le Raspberry Pi. Ce lecteur, qui supporte de nombreux formats multimédias, offre une interface avec de grands boutons et de gros caractères, ce qui le rend facile à utiliser depuis votre canapé. Il transforme votre Raspberry Pi en un home cinéma entièrement personnalisable.

UN OUTIL DE PROGRAMMATION BAS NIVEAU

La plupart des développeurs écrivent des programmes qui tournent sur un système d'exploitation comme Windows, Mac OS, ou, dans le cas de Raspberry Pi, Linux. Mais si vous écriviez du code qui s'exécute directement sur le processeur sans passer par un système d'exploitation ? Avec votre Raspberry Pi, vous pouvez même écrire votre propre système d'exploitation à partir de rien si cela vous branche. Le laboratoire d'informatique de l'université de Cambridge a ainsi publié un cours gratuit en ligne (http://www.cl.cam.ac.uk/projects/raspberrypi/tutorials/os) qui vous guide dans l'écriture de votre propre système d'exploitation en langage machine.

Linux et Raspberry Pi

Un ordinateur standard fonctionne avec un système d'exploitation comme Windows, OS X ou Linux. C'est ce système qui démarre quand vous allumez votre machine et qui fournit à vos logiciels un accès aux fonctionnalités matérielles de votre ordinateur. Par exemple, si vous codez une application qui utilise Internet, vous pouvez, pour ce faire, utiliser les fonctions fournies par le système d'exploitation. Vous n'avez pas besoin de comprendre et d'écrire du code pour chaque modèle d'interface Ethernet ou Wi-Fi disponible.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Comme tous les autres ordinateurs, le Raspberry Pi utilise un système d'exploitation. Celui fourni par défaut est une version de GNU/Linux appelée Raspbian[1]. GNU/Linux est idéal pour le Raspberry Pi car il est libre et open source. Non seulement il rend la plate-forme bon marché, mais aussi très souple et personnalisable. Et vous n'êtes pas limité à Raspbian puisqu'il existe d'autres versions (encore appelées distributions) différentes de Linux que vous pouvez charger dans le Raspberry Pi. Quelques systèmes non Linux sont également disponibles. Dans ce livre, nous utiliserons la distribution standard Raspbian, téléchargeable sur la page du site de Raspberry Pi (http://www.raspberry.org/downloads).

Si vous ne connaissez pas Linux, ne vous inquiétez pas : le chapitre 2 vous fournira toutes les bases pour vous en servir.

Ce que les autres ont déjà fait avec le Raspberry Pi

Quand vous avez accès à une technologie nouvelle et prometteuse, il est parfois difficile de décider ce que l'on va faire avec. Si vous hésitez encore, sachez que ce ne sont pas les idées intéressantes qui manquent et que vous pourrez vous inspirer d'une foule d'exemples de projets créatifs utilisant un Raspberry Pi. En tant que rédacteurs du magazine américain *MAKE*, nous avons vu passer ainsi beaucoup d'applications fantastiques du Raspberry Pi. En voici quelques-unes.

UNE TABLE BASSE À JEUX D'ARCADE INTÉGRÉS

Membre du site Instructables, grahamhelding y a publié un tutoriel décrivant pas à pas comment construire une table basse qui sert aussi d'émulateur de jeux d'arcade classiques (http://www.instructables.com/id/Coffee-Table-Pi). Pour faire fonctionner les jeux sur Raspberry Pi, il a utilisé MAME *(Multiple Arcade Machine Emulator)*, un projet libre et open

1. NdT : Linux est le cœur du système, appelé *kernel*, GNU est un projet qui fournit de nombreux logiciels indispensables au fonctionnement du système, et Raspbian est dérivé de Debian, une distribution qui combine harmonieusement le kernel, les outils et les applications.

source qui vous permet de jouer aux jeux d'arcade sur des ordinateurs modernes. Sur la table elle-même, il a monté un écran LCD de 24 pouces connecté au Raspberry Pi par HDMI. Pour les entrées, il a utilisé un joystick et des boutons d'arcade classiques connectés aux broches GPIO.

LE RASPOD

Lors du concours Summer Coding 2012, sponsorisé par la Fondation Raspberry Pi, un adolescent indien, Aneesh Dogra, a remporté la deuxième place en concevant Raspod, un lecteur MP3 contrôlable par Internet (https://github.com/lionaneesh/RasPod). Programmé en Python avec l'aide du framework web Tornado, Raspod permet aux utilisateurs de se connecter à leur Raspberry Pi pour écouter de la musique, la stopper, changer le volume, sélectionner des chansons et gérer les listes de lecture. La musique, qui sort par le connecteur audio du Raspberry Pi, peut être écoutée avec deux enceintes d'ordinateur ou sur une chaîne hi-fi.

UN SUPERORDINATEUR

Beaucoup de superordinateurs sont créés à partir de grappes *(clusters)* d'ordinateurs standards, qui se répartissent la charge de travail entre les différents processeurs. En suivant cette idée, un groupe d'ingénieurs en informatique de l'université de Southampton, au Royaume-Uni, a connecté 64 Raspberry Pi pour créer un superordinateur bon marché (http://www.southampton.ac.uk/mediacentre/features/raspberry_pi_supercomputer.shtml). Même s'il est beaucoup moins puissant que les superordinateurs actuels, ce projet illustre les grands principes de fonctionnement de tels systèmes. Le plus incroyable, c'est que la structure supportant tous les Raspberry Pi a été construite en Lego par le fils de 6 ans du chef de ce projet !

Télécharger et utiliser les exemples de code source

Les exemples de code source figurant dans les différents chapitres sont disponibles sur le minisite associé au livre :
http://www.serialmakers.com/livres/raspberry-pi.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce livre a été conçu pour vous aider dans vos projets. En règle générale, vous pouvez donc utiliser le code source de ses exemples dans vos programmes et votre documentation. Vous n'avez pas besoin de nous contacter pour demander notre autorisation, à moins que vous ne reproduisiez une quantité significative de code.

Ainsi, vous pouvez écrire un programme qui utilise plusieurs morceaux de code de ce livre sans nous demander d'autorisation. En revanche, il vous en faudra une pour distribuer ou vendre un CD-Rom contenant les exemples des livres.

Vous pouvez également citer ce livre et les codes des exemples, mais l'incorporation d'une quantité significative de code dans la documentation d'un produit nécessitera une autorisation.

Si vous souhaitez utiliser les codes d'exemples de ce livre en dehors de ces cadres, merci de contacter les éditions Eyrolles.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

TABLE DES MATIÈRES

Chapitre 1. **PRÉSENTATION DU RASPBERRY PI ET PREMIER DÉMARRAGE** 1

La carte 2
Les périphériques 7
Le boîtier 11
La distribution 12
Préparer la carte SD 13
Démarrer le Raspberry Pi 15
Configurer le Pi 17
Éteindre le Raspberry Pi 20
En cas de problème 20
Pour aller plus loin 22

Chapitre 2. **À LA DÉCOUVERTE DE LINUX POUR LE RASPBERRY PI** 23

Utilisation de la ligne de commande 27
Les fichiers et le système de fichiers 28

D'autres commandes Linux 32
Les processus 35
sudo et les permissions 36
Le réseau 38
Le répertoire /etc 40
Réglage de la date et de l'heure 40
Installation de nouveaux logiciels 41
Pour aller plus loin 42

Chapitre 3. **PROGRAMMER EN PYTHON** 43

Bonjour Python ! 45
Un peu plus de Python 47
Objets et modules 50
Encore plus de modules 54
Déceler et corriger les erreurs 55
Pour aller plus loin 56

Chapitre 4. **ANIMATION ET MULTIMÉDIA AVEC PYTHON** 59

Bonjour Pygame ! 60
Les surfaces de Pygame 61
Dessiner sur les surfaces 64
Gestion des événements et des entrées 65
Les sprites 68
Jouer des sons 70
Lire une vidéo 73
Pour aller plus loin 74

Chapitre 5. **PROGRAMMER EN SCRATCH** 75

Bonjour Scratch ! 76
La scène 81
Deux autres choses à savoir sur les sprites 82

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Un exemple plus élaboré : le jeu Astral Trespassers 84
Scratch et le monde réel 91
Partagez vos programmes ! 92
Pour aller plus loin 93

Chapitre 6. **RASPBERRY PI ET ARDUINO** 95

Installer Arduino sur Raspbian 97
Identification du port série 99
Communiquer via le port série 100
Pour aller plus loin 104

Chapitre 7. **LES BROCHES D'ENTRÉES-SORTIES** 107

Utilisation des broches d'entrées-sorties 110
Sortie numérique : allumage d'une LED 112
Entrée numérique : lecture d'un bouton 117
Projet : séquenceur de lampe avec cron 121
Programmer un script shell 121
Connexion d'une lampe 123
Planifier des commandes avec le démon cron 125
Formater une tâche pour cron 126
Pour aller plus loin 127

Chapitre 8. **PROGRAMMATION DES ENTRÉES-SORTIES AVEC PYTHON** 129

Installation et test du module Python GPIO 130
Faire clignoter une LED 133
Lire l'état d'un bouton 135
Projet : boîte à bruitages 138
Pour aller plus loin 143

Chapitre 9. **UTILISER DES WEBCAMS** 145

Tester sa webcam 147
Installation et test de SimpleCV 148

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Afficher une image 149
Modifier une image 151
Accéder à la webcam 154
Reconnaissance de visage 156
Projet : cabine photographique 158
Pour aller plus loin 161

Chapitre 10. **ACCÉDER À INTERNET AVEC PYTHON** 163

Télécharger des données d'un serveur web 164
Charger les prévisions météo 166
Le Raspberry Pi comme serveur web 173
Les bases de Flask 174
Connecter le Web au monde réel 178
Projet : une lampe contrôlée par Internet 180
Pour aller plus loin 185

Annexe A. **ENREGISTRER UNE IMAGE DU SYSTÈME D'EXPLOITATION SUR UNE CARTE SD** 187

Écrire une carte SD à partir de Mac OS X 187
Écrire une carte SD à partir de Windows 189
Écrire une carte SD à partir de Linux 189

Annexe B. **LE JEU ASTRAL TRESPASSERS AU COMPLET** 193

Annexe C. **ENTRÉES ANALOGIQUES** 199

Conversion analogique/numérique 199

Annexe D. **À PROPOS DES CONTRIBUTEURS** 207

Les auteurs 207
Les traducteurs 207

Index 209

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

CHAPITRE 1

PRÉSENTATION DU RASPBERRY PI ET PREMIER DÉMARRAGE

Lorsqu'on parle du Raspberry Pi, certains qualificatifs reviennent sans cesse : il est petit, bon marché, "bidouillable" et adapté à des fins pédagogiques. Cependant, on ne peut pas dire qu'il soit *plug and play*, même s'il est facile de le brancher à un téléviseur et de lui faire afficher des choses à l'écran. Contrairement à la plupart des ordinateurs actuels, le Raspberry Pi n'est pas un appareil autonome et livré prêt à l'emploi. En fonction de l'usage que vous souhaitez en faire, vous devrez installer certains périphériques et logiciels appropriés.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Si vous ne possédez pas encore votre Pi, vous pourrez vous en procurer un pour 30 ou 40 €[1] auprès de certains fabricants avec lesquels la fondation Raspberry Pi a mis en place des partenariats. Ce sont :

- Premier Farnell/element14 (http://www.element14.com/community/groups/raspberry-pi), distributeur britannique de composants électroniques possédant de nombreuses filiales dans le monde entier (dont Newark et MCM aux États-Unis).

- RS Components (http://www.rs-components.com/raspberrypi), autre distributeur de matériel électronique basé au Royaume-Uni (Allied Electronics est leur filiale aux États-Unis).

Le but de la fondation Raspberry Pi est de rendre le mini-ordinateur le plus accessible possible. Le Raspberry Pi peut ainsi être acheté à l'unité par tout un chacun directement auprès d'un distributeur, au même prix que les grossistes, ce qui constitue un mode de distribution peu commun. Beaucoup de revendeurs potentiels ont été consternés à l'annonce de cette organisation car ils voyaient difficilement comment faire leur marge. C'est pour cette raison que vous trouverez des Pi neufs vendus 10 à 20 % plus chers que le prix officiel. Les particuliers peuvent acquérir des Pi au prix d'origine directement auprès des fabricants, mais les détaillants et revendeurs présentent l'avantage d'expédier les commandes plus rapidement. Parmi eux, citons par exemple Maker Shed (http://www.makershed.com/category_s/227.htm, filiale de Make) et Adafruit (http://www.adafruit.com/category/105).

Après cette parenthèse microéconomique, passons à la technique et étudions de plus près la carte du Raspberry Pi.

La carte

Lorsque vous sortirez votre Pi de sa boîte, vous trouverez les différents éléments que nous allons examiner ci-après.

1. NdT : le prix officiel d'un Raspberry Pi est de 35 dollars US hors taxe. Les variations de taux de change ainsi que les différentes taxes, frais de douane et de port augmentent le coût final.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Vous serez peut-être tenté de considérer le Raspberry Pi comme une carte à microcontrôleur comme Arduino, ou comme un substitut à votre ordinateur portable. En réalité, la structure du Pi ressemble davantage à celle d'un appareil mobile, doté de plusieurs connecteurs permettant d'accéder facilement à ses nombreux ports et fonctions. La figure 1-1 représente les différents composants de la carte.

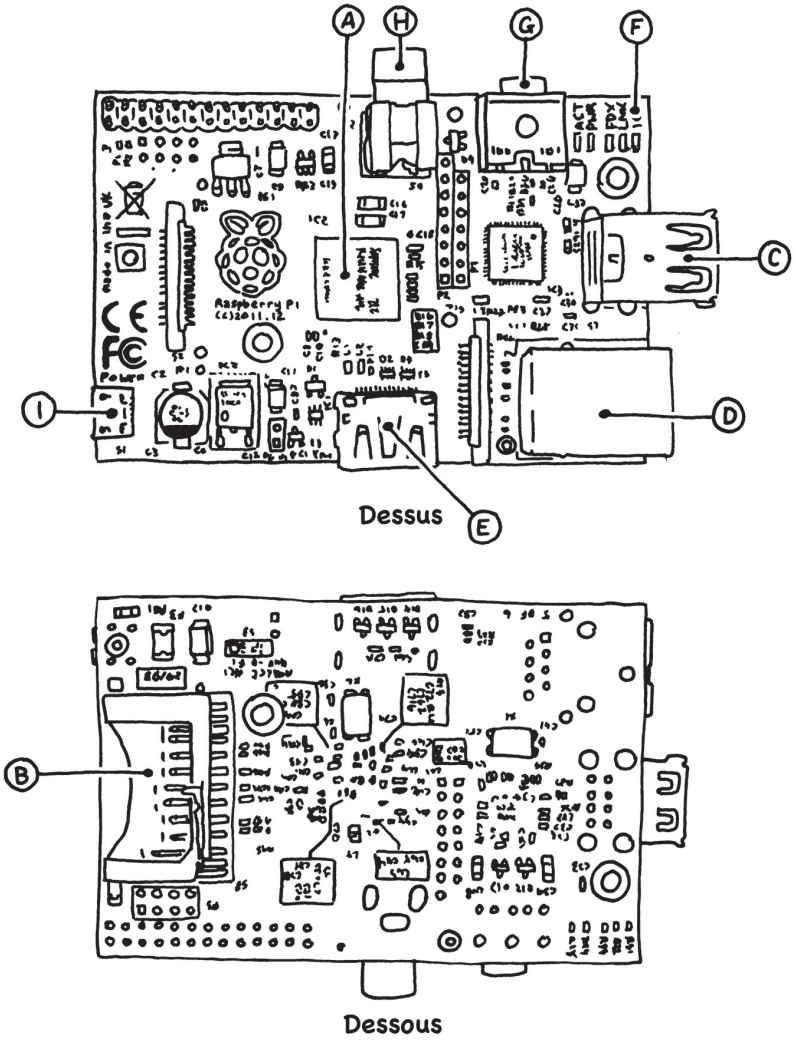


Figure 1-1. Emplacement des ports et interfaces matérielles du Raspberry Pi

A. **Le processeur** – Le cœur du Raspberry Pi est équipé d'un processeur de même type que celui de l'iPhone 3G et du Kindle 2, sa puissance est donc comparable. La puce centrale est un *System on Chip*[2] (SoC) 32 bits cadencé à 700 MHz, utilisant l'architecture ARM11 conçue par la société anglaise ARM Holdings. Les processeurs ARM, fabriqués sous licence par des centaines de sociétés, sont déclinés en plusieurs familles aux caractéristiques et prix variés. Le Pi modèle B est doté de 512 Mo de RAM alors que le modèle A ne dispose que de 256 Mo (mais il consomme moins d'électricité). À noter que la RAM des premières versions du modèle B était de 256 Mo.

B. **Le support de carte SD *(Secure Digital)*** – Comme vous pourrez rapidement le constater, le Pi ne possède pas de disque dur, tout est stocké sur une carte SD. Son support est relativement fragile et il est conseillé de protéger le Raspberry Pi dans un boîtier afin d'éviter que le support SD soit accidentellement arraché.

C. **La prise USB** – Le modèle B possède deux ports USB 2.0, contrairement au modèle A qui n'en possède qu'un. Les premières cartes Raspberry Pi ne pouvaient fournir qu'un courant limité aux périphériques USB. Alors que certains appareils USB peuvent consommer jusqu'à 500 mA, la carte Pi originale ne pouvait fournir que 100 mA environ. Ceci a évolué et les dernières versions respectent désormais la norme USB 2.0. Si vous souhaitez déterminer le type de votre carte, assurez-vous que deux composants (des *polyfuses*, ou « fusibles réarmables ») limitent le courant (figure 1-2). Ce n'est donc pas une très bonne idée de charger votre téléphone portable avec le Pi, car certains smartphones nécessitent un courant de 1 A… Si vos périphériques USB ont besoin de beaucoup d'énergie, utilisez un hub USB avec une alimentation dédiée.

2. NdT : un *System on Chip* est une puce qui intègre tous les éléments d'un ordinateur, à savoir le(s) microprocesseur(s), la gestion de mémoire, les coprocesseurs, les contrôleurs de nombreux périphériques et interfaces, etc. Dans le cas du Pi, la mémoire est même directement soudée sur le SoC. Observez-le de près et admirez le délicat sandwich électronique !

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

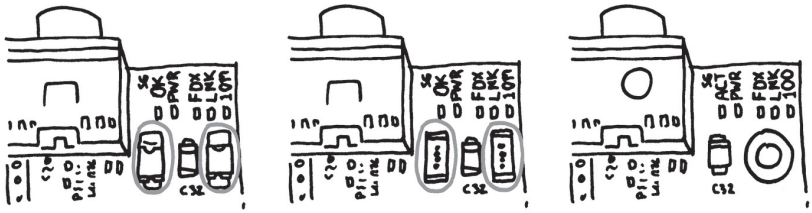


Figure 1-2. Les premières versions des cartes disposent de polyfuses pour protéger le port USB (à gauche). Sur d'autres cartes, les polyfuses ont été court-circuités par des jumpers (au centre). La dernière version du modèle B les a supprimés pour fournir un perçage de fixation (à droite).

D. **La prise Ethernet** – Le modèle B du Raspberry Pi possède une prise Ethernet standard au format RJ45. Le modèle A n'en a pas mais il est possible de le connecter à un réseau avec un adaptateur USB vers Ethernet (le port Ethernet du modèle B est en fait un adaptateur USB vers Ethernet soudé sur la carte). Pour se connecter en Wi-Fi, il faut utiliser un *dongle* USB.

E. **La prise HDMI** – Le port HDMI fournit les sorties vidéo et audio numériques et il supporte 14 résolutions vidéo différentes. En utilisant des adaptateurs externes, le signal HDMI peut être converti dans d'autres formats comme le DVI (utilisé par la plupart des moniteurs informatiques), composite (un signal analogique vidéo habituellement transmis par un connecteur RCA jaune) ou SCART/Peritel (le standard européen pour connecter les appareils audiovisuels analogiques).

F. **Les LED d'état** – Le Pi dispose de cinq LED qui renseignent sur l'activité de la carte (tableau 1-1).

Tableau 1-1. Les cinq LED d'état

LED	Couleur	Commentaires
ACT	Vert	S'allume lors des accès à la carte SD (correspond à la LED « OK » des premières versions du Pi).
PWR	Rouge	Témoin de l'alimentation 3,3 V
FDX	Vert	S'allume lorsque le port Ethernet est en mode *full duplex*.
LNK	Vert	Témoin d'activité du port Ethernet
100	Jaune	Indique que la connexion au réseau est en mode 100 Mbps (certaines anciennes cartes ont un marquage incorrect « 10 M »).

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

G. **La sortie audio analogique** – La carte comporte une prise jack standard de 3,5 mm qui fournit un signal sonore stéréo à haute impédance, destiné à des appareils comme des haut-parleurs amplifiés. La qualité du son sera mauvaise si l'on y branche un casque ou des haut-parleurs normaux. Pour l'instant, la qualité de la sortie analogique est très inférieure à celle d'un téléviseur connecté au port HDMI, car le SoC a été conçu pour les téléphones portables et non pour la Hi-Fi. Le *driver* audio est aussi en train d'évoluer.

H. **La sortie vidéo composite** – C'est une prise standard de type RCA qui fournit les signaux vidéo composites NTSC ou PAL. Le format composite a une résolution très basse par rapport à celle du HDMI. Si votre écran possède une entrée HDMI, utilisez-la plutôt que l'entrée composite.

I. **Le connecteur d'alimentation** – L'alimentation du Pi n'est pas contrôlée par un interrupteur. Le connecteur micro USB sert uniquement à alimenter la carte et il fonctionne comme un interrupteur. Il ne s'agit pas d'un port USB supplémentaire. Le format micro USB a été choisi parce que ce connecteur est bon marché et qu'on trouve très facilement des alimentations de ce type pour les smartphones.

La figure 1-3 montre tous les connecteurs d'alimentation et d'entrées-sorties du Raspberry Pi :

A. **Les broches d'entrées-sorties généralistes (GPIO)** – Vous verrez aux chapitres 7 et 8 comment utiliser ces broches pour connaître l'état des boutons ou des interrupteurs, et contrôler des LED, des relais ou des moteurs.

B. **Le connecteur *Display Serial Interface* (DSI)** – Ce connecteur accepte un câble plat à 15 conducteurs qui peut communiquer avec un afficheur LCD ou OLED.

C. **Le connecteur *Camera Serial Interface* (CSI)** – Ce port permet de connecter directement à la carte un module caméra spécifique.

D. **Les points d'accès P2 et P3** – Ces deux rangées de contacts sont destinées aux pointes de test JTAG, utilisées en usine pour valider le fonctionnement de la puce centrale Broadcom (P2) et la puce d'inter-

face USB réseau LAN9512 (P3). Les spécifications du SoC Broadcom étant confidentielles, ces signaux sont pratiquement inutiles.

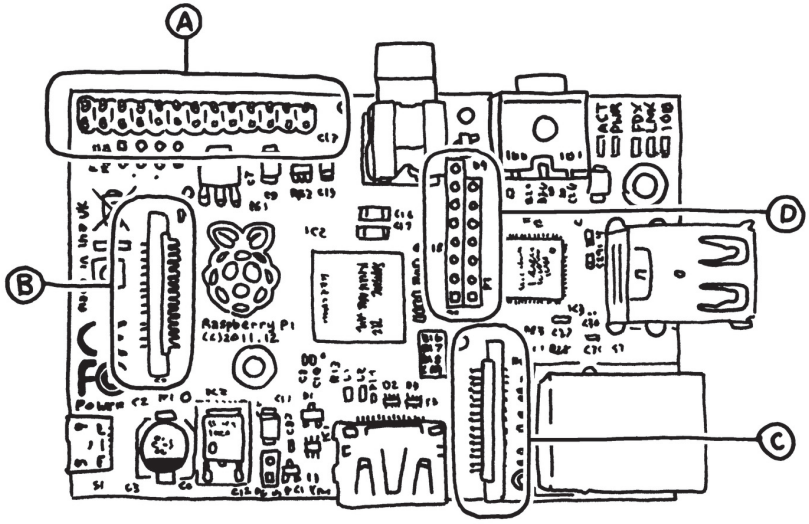


Figure 1-3 . Les connecteurs du Raspberry Pi

À l'automne 2012, la fondation Raspberry Pi a lancé la version 2 de la carte, ajoutant deux trous de montage de 2,5 mm ainsi que des signaux sur lesquels on peut brancher un bouton de réinitialisation *(reset)*. Elle est également dotée d'un emplacement spécifique à côté des GPIO pour un connecteur à 2 × 4 broches (non soudé), destiné à connecter des cartes d'extension pour le son ou pour une horloge temps réel (montées en dessous de la carte principale).

Les périphériques

Maintenant que vous connaissez tous les composants de la carte, vous devez savoir certaines choses sur les périphériques dont le Pi a besoin

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

(figure 1-4). Il existe de nombreux kits d'initiation ou de démarrage complets, qui contiennent des éléments dont la compatibilité a été vérifiée. Si vous êtes amené à ajouter des équipements à votre Raspberry Pi, ayez en tête les quelques points et pièges suivants à éviter.

A. **L'alimentation** – C'est *le* périphérique indispensable sur lequel il ne faut surtout pas se tromper. Vous devez impérativement utiliser un adaptateur 5 V avec une fiche micro USB, capable de fournir au moins 700 mA de courant (ou 500 mA pour le modèle A). Une alimentation de 1 000 mA (1 A) est recommandée. Un chargeur classique de téléphone portable ne suffira probablement pas, même s'il possède la bonne fiche, car il ne pourra pas fournir plus de 400 mA. Pour connaître sa puissance, consultez ses caractéristiques techniques figurant au dos. Un Pi sous-alimenté pourra avoir l'air de fonctionner mais il sera instable et déclenchera des erreurs inattendues.



Depuis la nouvelle version, vous pouvez alimenter la carte par l'un des ports USB au moyen d'un hub USB alimenté. Cependant, cette solution n'est pas idéale car les circuits de protection seront inefficaces en cas de court-circuit. C'est particulièrement le cas quand un accident ou une faute de câblage se produit pendant que vous testez vos prototypes électroniques.

B. **Une carte SD** – Vous aurez besoin d'une carte d'une capacité d'au moins 4 Go et de classe 4 (ou plus) de préférence. Les cartes de classe 4 sont capables de transférer au moins 4 Mo par seconde. Certaines cartes Raspberry Pi précédentes avaient des difficultés à utiliser les cartes de classe 6 et supérieures, qui sont capables d'aller plus vite mais qui s'avèrent moins stables. Une carte microSD dans un adaptateur convient également.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

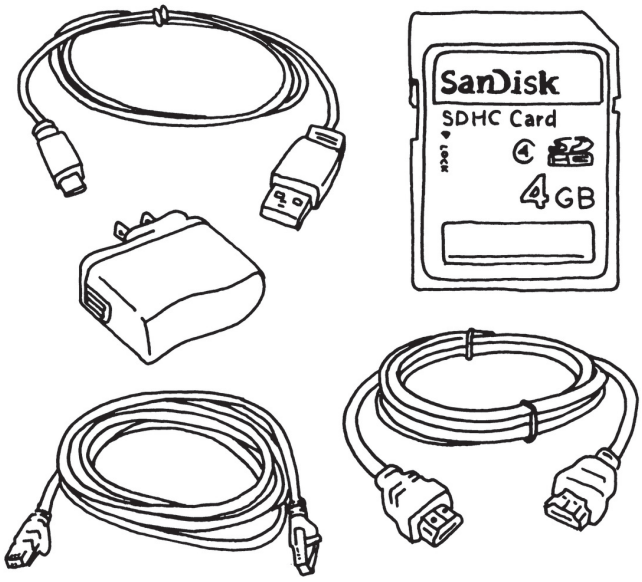


Figure 1-4. Les périphériques indispensables : une alimentation micro USB, des câbles et une carte SD. Vous aurez besoin d'une carte SD de 4 Go au minimum (les cartes microSD avec des adaptateurs fonctionnent aussi). Les cartes SD génériques sont de qualité très inégale donc il vaut mieux utiliser une carte de marque connue (voir http://elinux.org/RPi_SD_cards).

C. **Un câble HDMI** – Vous en aurez besoin si vous connectez la carte à un moniteur HDMI. Un moniteur DVI nécessitera en plus un adaptateur approprié. Vous pouvez aussi faire fonctionner le Pi sans écran (voir page 39). Les prix des câbles HDMI sont très variables. Pour un câble d'1 ou 2 mètres, il est inutile de dépenser plus de 3 €. Si vous avez besoin de longueurs inhabituelles, privilégiez les câbles de bonne qualité et évitez les modèles premier prix.

D. **Un câble Ethernet** – Chez vous, vous disposez probablement de moins de câbles Ethernet qu'il y a 5 ans. De nos jours, de plus en plus d'appareils fonctionnent sans fil et vous trouverez donc peut-être que le port Ethernet du Pi n'est pas très pratique. Reportez-vous à l'encadré « Travailler sans écran » page 39 si vous voulez vous passer de la connexion Ethernet filaire.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Si vous avez des projets plus ambitieux pour votre Raspberry Pi, vous pourrez recourir à d'autres accessoires dont nous parlerons plus loin dans l'ouvrage. Citons dès à présent les plus courants.

- **Un hub USB alimenté :** un hub USB 2.0 disposant de sa propre alimentation est recommandé. Il déchargera l'alimentation du Pi, lequel sera alors plus stable.

- **Un dissipateur :** un dissipateur, ou radiateur, est un petit morceau de métal, usiné avec des ailettes pour augmenter la surface en contact avec l'air afin de mieux dissiper la chaleur. Des dissipateurs sont souvent fixés sur les composants qui chauffent pour les faire refroidir. Le SoC du Pi a été conçu pour les applications mobiles, il est donc inutile normalement d'ajouter un dissipateur. Comme nous le verrons plus loin, il peut s'avérer nécessaire dans certains cas de faire tourner le processeur plus vite, ou bien de lui faire exécuter des calculs complexes, ce qui peut le faire chauffer et le rendre instable. Des utilisateurs ont aussi rapporté que l'interface réseau peut chauffer.

- **Une horloge temps réel :** pour réduire le prix de vente, le Pi n'est pas équipé de composants pour maintenir l'heure ou la date quand l'alimentation est éteinte. L'horloge du microprocesseur est mise à jour avec NTP *(Network Time Protocol)* lorsque Linux se connecte à Internet. Vous pouvez ajouter un circuit d'horloge temps réel (comme le DS1307) pour que l'horloge reste à l'heure lorsque le réseau, ou le courant, est coupé.

- **Un module caméra :** depuis début 2013, il existe un module officiel pour ajouter une caméra 5 mégapixels au Raspberry Pi. Vous pouvez aussi utiliser une webcam USB (voir chapitre 9 pour un exemple complet).

- **Un moniteur LCD :** certains afficheurs LCD peuvent être ajoutés en passant par les broches de GPIO. En 2013, des écrans LCD compatibles avec l'interface DSI, plus performants, seront aussi disponibles.

- **Un dongle Wi-Fi USB :** un grand nombre de dongles Wi-Fi USB sont compatibles avec le Pi. Choisissez-en un qui ne consomme pas trop d'énergie.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

- **Une station de base ou dock d'extension :** certains utilisateurs ont modifié leur dock de téléphone mobile (comme le lapdock Atrix) pour servir de station de base au Raspberry Pi.

Le boîtier

Vous aurez rapidement besoin d'un boîtier pour votre Raspberry Pi. Les câbles rigides qui l'entourent de chaque côté rendent le Pi difficile à stabiliser. Certains des composants comme le support de la carte SD peuvent être endommagés, même en utilisation normale sans protection.

Contrairement à beaucoup de cartes à microcontrôleurs répandues, dont le circuit imprimé a deux couches (une sur chaque face de la platine), le Pi nécessite six couches pour relier ses composants à haute densité. Entre les deux faces, on trouve aussi des pistes sur quatre autres niveaux. Si la carte est tordue, ces pistes peuvent rompre, ce qui est indétectable et irréparable.

Pour pallier ce problème, nous vous conseillons de faire l'acquisition d'un boîtier.

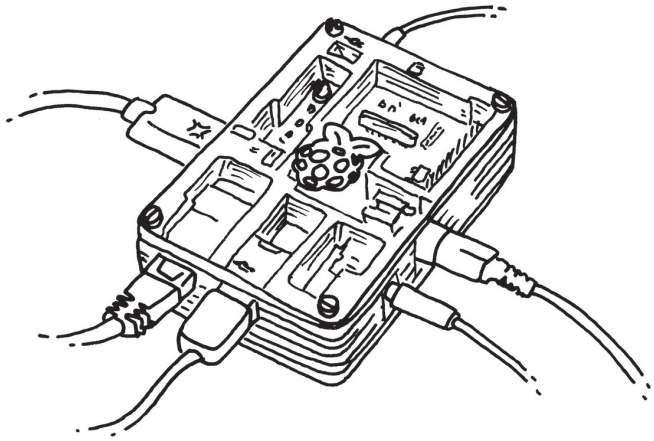


Figure 1-5. Le boîtier Pibow est multicolore.

Il existe un large choix de boîtiers préfabriqués, ainsi que de nombreux plans à télécharger pour fabriquer son propre boîtier par fraisage numérique, découpe au laser ou impression 3D. Il vaut mieux éviter les boîtiers en acrylique dont les languettes à angles droits sont fragiles. Dans le cas du boîtier Pibow (figure 1-5), l'empilement de différentes couches en acrylique forme un arc-en-ciel.

Ce serait évidemment une erreur de poser votre Raspberry Pi sur une surface métallique ou conductrice. Retournez votre Pi et regardez de près, vous verrez beaucoup de composants et de soudures sensibles aux courts-circuits. Autant de raisons importantes pour protéger votre Pi !

La distribution

Le Raspberry Pi fonctionne généralement avec un système d'exploitation basé sur Linux. Techniquement parlant, Linux n'est qu'un noyau alors qu'un système d'exploitation est un ensemble de différents éléments (drivers, services et applications). Il existe de nombreuses distributions Linux, très diversifiées et qui évoluent au fil des années. Les distributions les plus courantes sur les ordinateurs de bureau sont Ubuntu, Debian, Fedora et Arch. Chacune d'entre elles est adaptée à des applications spécifiques et possède sa propre communauté d'utilisateurs.

Comme le Pi est basé sur un SoC pour appareil mobile, il n'utilise pas les mêmes logiciels qu'un ordinateur de bureau. Il n'existe pas de documentation sur les caractéristiques de ce processeur construit par Broadcom, gardées confidentielles. Il faut donc ajouter des drivers spécifiques, contenant des données binaires propriétaires, et adapter les distributions Linux en conséquence.

Tandis que la plupart des ordinateurs offrent plusieurs gigaoctets de RAM et des centaines de gigaoctets de stockage, le Pi est quant à lui plus limité dans ces deux domaines. On trouve de plus en plus de distributions Linux développées ou adaptées spécifiquement pour le Pi, les plus connues sont les suivantes.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

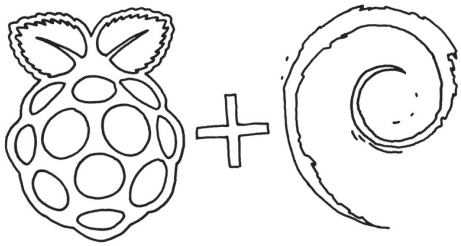


Figure 1-6. Raspberry Pi + Raspbian

- **Raspbian :** basée sur Debian, Raspbian est la distribution recommandée par la fondation Raspberry Pi (figure 1-6). Le site http://www.raspbian.org/ n'est pas géré par la fondation mais par la communauté des utilisateurs. Pour télécharger la distribution officielle, rendez-vous sur la page http://www.raspberrypi.org/downloads.

- **Raspberry Pi Linux version éducative d'Adafruit (Occidentalis) :** la distribution d'Adafruit basée sur Raspbian contient un certain nombre d'outils et de drivers utiles pour apprendre l'électronique.

- **Arch Linux :** destinée aux ordinateurs dotés de processeurs ARM, la distribution Arch Linux supporte le Pi depuis le début de son développement.

- **XBian :** il s'agit d'une distribution pour les utilisateurs de Raspbian qui souhaitent utiliser le Pi comme un centre multimédia (voir aussi OpenELEC et Raspbmc).

- **QtonPi :** cette distribution est basée sur le framework Qt 5.

Dans cet ouvrage, nous nous concentrerons sur la distribution officielle Raspbian.

Préparer la carte SD

Plusieurs distributeurs de Raspberry Pi proposent également des cartes SD sur lesquelles un système d'exploitation est préinstallé. Pour

de nombreux utilisateurs, cette solution est idéale pour débuter. Si vous n'avez pas reçu la dernière version de Raspbian, vous pouvez facilement la mettre à jour au démarrage de votre Pi, une fois celui-ci connecté à Internet.

Raspbian dispose aussi d'un installateur par réseau. Pour utiliser cet outil, vous devez charger les fichiers de l'installateur sur une carte SD (formatée en FAT32, le type de partition typique pour ces cartes), puis démarrer le Pi avec la carte insérée dans son support. Il faut évidemment que la carte ait accès à Internet.

La première étape consiste à télécharger Raspbian à partir de la page de téléchargements http://www.raspberrypi.org/downloads. Le système d'exploitation est distribué sous la forme d'une « image » du disque. Il s'agit d'un fichier qui décrit le contenu exact de la carte SD et qui comporte les systèmes de fichiers et même la table de partitions.

Téléchargez l'image compressée (`fichier.zip`) et décompressez-la. Vous obtenez alors un fichier de 2 Go environ. Ensuite, selon le système d'exploitation installé sur votre ordinateur, suivez les instructions figurant dans l'annexe A de cet ouvrage.

Des téléchargements plus rapides grâce à BitTorrent

Sur la page de téléchargements du site http://www.raspberrypi.org, il est indiqué que l'image peut être téléchargée plus rapidement au moyen d'un fichier *torrent*. Ce dernier contient les informations nécessaires pour récupérer toutes les données grâce à BitTorrent, un système décentralisé de distribution de fichiers. Cette méthode est très rapide dans la mesure où les différentes parties du fichier sont chargées simultanément à partir de plusieurs ordinateurs et non d'un unique serveur centralisé. Pour télécharger l'image de cette manière, vous aurez besoin d'un client BitTorrent, dont les plus populaires sont :

- Vuze (http://www.vuze.com) – Intègre le téléchargement et la recherche dans les torrents.
- Miro (http://www.getmiro.com) – Lecteur de musique et de vidéo open source qui peut lire les torrents.
- MLDonkey (http://mldonkey.sourceforge.net/Main_Page) – Outil de partage de fichiers pour Windows et Linux.
- Transmission (http://www.transmissionbt.com) – Petit logiciel pour Mac et Linux uniquement, aussi utilisé dans les systèmes embarqués.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

À noter qu'il ne suffit pas de recopier ce fichier dans la carte SD avec votre ordinateur, car la table des partitions ne serait alors pas mise à jour. Il faut copier l'image bit par bit grâce à un accès de bas niveau à la carte SD. En plus d'un lecteur de cartes (souvent une interface USB), les utilisateurs de Windows ou de Mac auront besoin d'un utilitaire spécial pour écrire l'image du disque sur la carte.

Démarrer le Raspberry Pi

Suivez ces étapes pour démarrer votre Raspberry Pi pour la première fois.

1. Insérez la carte SD dans son support.
2. Branchez un clavier USB et une souris au Raspberry Pi. Si vous possédez le modèle A, branchez-les sur un hub alimenté, puis connectez ce dernier à votre Pi.
3. Connectez la sortie HDMI à votre téléviseur ou moniteur. Assurez-vous que celui-ci est allumé.
4. Branchez l'alimentation du Pi. De manière générale, vérifiez que tous les autres éléments sont bien branchés avant de connecter l'alimentation.

Se connecter à Internet

Vous pouvez vous connecter à Internet de plusieurs manières. Si vous avez facilement accès à un routeur ou un switch Ethernet, il suffit d'utiliser un câble Ethernet standard. Vous pouvez aussi vous connecter sans fil si vous possédez un dongle Wi-Fi USB. Une icône située sur le Bureau (dans l'interface graphique) vous permettra d'accéder aux paramètres de configuration de votre connexion. Attention, tous les dongles Wi-Fi ne sont pas compatibles : consultez la liste des périphériques fonctionnant avec le Raspberry Pi sur http://elinux.org/RPi_VerifiedPeripherals.

Si vous avez un ordinateur portable à proximité, ou si votre Pi n'est pas équipé d'un écran, vous pouvez partager le Wi-Fi de ce portable avec votre Pi (figure 1-7). Ceci est très facile à réaliser avec un Mac : il suffit de permettre le partage Internet *(Internet Sharing)* dans vos paramètres de partage réseau, puis de connecter le Mac et le Pi grâce à un câble Ethernet.

Sous Windows, il faut activer l'option Permettre à d'autres utilisateurs du réseau de se connecter à Internet au travers de cet ordinateur *(Allow other network users to connect through this computer's Internet connection)*, située dans les propriétés de partage de connexion Internet *(Internet Connection Sharing)*. Une fois branché, le Pi devrait obtenir une adresse IP automatiquement.

Un câble croisé sera probablement nécessaire pour un PC sous Windows, mais vous pouvez utiliser n'importe quel câble Ethernet sur du matériel Apple car ce type de câble est détecté automatiquement.

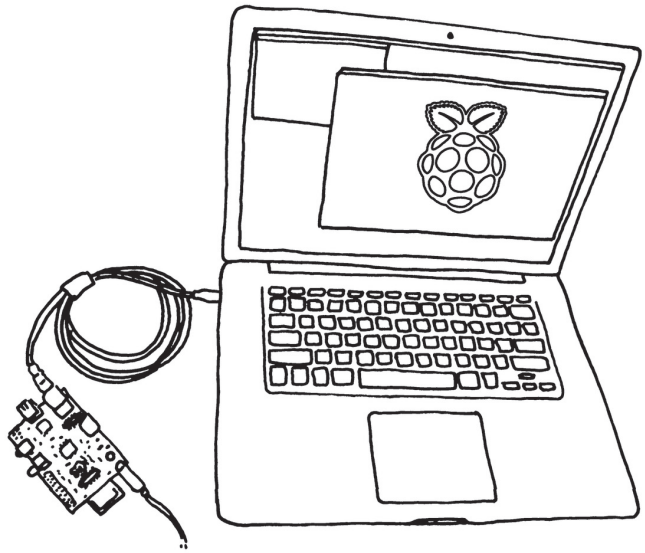


Figure 1-7. Une astuce très pratique consiste à partager la connexion Wi-Fi de votre ordinateur portable avec le Pi. Il peut même fonctionner sans écran, ce qui est très utile si vous utilisez votre Pi en voyage.

Si tout se passe correctement, vous verrez une liste de messages défiler à l'écran. En cas de problème, reportez-vous à la section « Dépannage » à la fin de ce chapitre. Ces messages de log montrent tous les processus du système d'exploitation qui se lancent au démarrage de votre Pi. Vous verrez l'interface réseau s'initialiser et les différents périphériques seront détectés. Ces messages peuvent être examinés par la suite, en saisissant la commande `dmesg` sur la ligne de commande de `bash`.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Au tout premier démarrage de votre Pi, l'outil `raspi-config` se lancera automatiquement pour personnaliser votre carte (figure 1-8). La configuration par défaut devra être adaptée pour que votre Raspberry Pi fonctionne conformément à vos attentes. Si besoin, vous pourrez relancer ultérieurement cet outil de configuration en saisissant la commande suivante :

```
sudo raspi-config
```

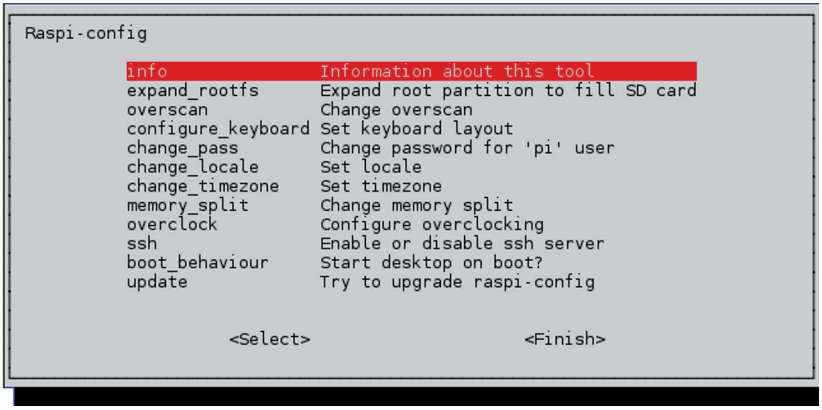


Figure 1-8. Le menu principal de l'outil raspi-config

Configurer le Pi

Nous allons à présent passer en revue les différents paramètres de base et distinguer ceux qui sont essentiels de ceux qui pourront être réglés par la suite.

Dans l'outil `raspi-config`, utilisez les flèches de direction du clavier haut et bas pour naviguer dans la liste. La barre d'espacement permet de sélectionner une option et la touche Tab de passer d'un champ à un autre ou de déplacer le curseur en bas de l'écran pour quitter.

Les options disponibles sont les suivantes[3].

- **`expand_rootfs` :** vous devez toujours choisir cette option, qui agrandit le système de fichiers Linux, afin d'utiliser l'intégralité d'une carte SD. Par exemple, si vous disposez d'une carte SD de 4 Go, Linux pourra accéder à 2 Go supplémentaires.

- **`overscan` :** pour commencer, laissez cette option désactivée. Si vous possédez un moniteur haute définition, il est possible que du texte dépasse de votre écran. Pour résoudre ce problème, activez l'option `overscan` et modifiez les valeurs pour que l'image s'adapte à la taille de votre écran. Entrez des valeurs positives si l'image sort de l'écran, des valeurs négatives si des marges noires apparaissent sur les côtés.

- **`keyboard` :** par défaut, les paramètres du clavier sont réglés pour un clavier britannique (QWERTY). Pour que les bonnes lettres soient saisies lorsque vous tapez sur les touches correspondantes, vous devez choisir le type de clavier approprié. Heureusement, Linux supporte énormément de langues et de dispositions de touches. Souvenez-vous que vos paramètres de localisation (option `change_locale` ci-après) peuvent affecter les paramètres du clavier.

- **`password` :** nous vous conseillons de modifier le mot de passe par défaut (`raspberry`) pour le rendre plus fiable.

- **`change_locale` :** si vous ne vous trouvez pas en Angleterre, vous devrez modifier la valeur de cette option pour activer votre langue et vos préférences d'encodage des caractères. Le paramètre par défaut est l'anglais (`en`) britannique (`UK`) avec un encodage de caractères en UTF-8 (ce qui donne `en_GB.UTF-8`). Il est conseillé de sélectionner `fr_FR.UTF-8` si vous êtes en France.

- **`change_timezone` :** modifiez cette option pour sélectionner le fuseau horaire correspondant au lieu où vous vous trouvez.

- **`memory_split` :** cette option vous permet de changer la quantité de mémoire allouée à l'affichage. Le partage par défaut convient pour le moment.

3. NdT : l'outil `raspi-config` est démarré en anglais et les termes originaux sont conservés ici pour référence.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

- **overclock :** vous avez la possibilité de faire tourner le processeur à des vitesses supérieures à 700 MHz. Pour votre premier démarrage, conservez les paramètres par défaut. Si par la suite vous devez modifier ce paramètre, essayez les valeurs Medium ou Modest. Le mode Turbo permet de monter à 1 GHz avec un dissipateur.

- **ssh *(Serveur Secure Shell)* :** cette option active l'accès au shell sécurisé, ce qui vous permet de vous connecter à votre Raspberry Pi en passant par le réseau. Activez cette option, elle est vraiment très pratique.

- **boot_behaviour :** cette option est sélectionnée par défaut et active le démarrage automatique de l'environnement de bureau graphique. Si vous sélectionnez `No`, le Pi restera en mode Texte. Vous pourrez toujours lancer l'interface graphique manuellement, comme suit :

```
raspberrypi login: pi
Password: raspberry (ou le mot de passe que vous venez
de paramétrer)
pi@raspberrypi ~ $ startx
```

  Quand vous êtes dans l'interface graphique, il n'y a plus de ligne de commande. Vous pouvez en obtenir une ou plusieurs en lançant un programme de terminal. Pour cela, cliquez sur le menu Bureau en bas à gauche et choisissez Accessoires>LXTerminal.

- **update :** si vous êtes connecté à Internet, cette option vous permettra de mettre à jour l'utilitaire de configuration. Ne lancez pas de mise à jour du système d'exploitation lors du premier démarrage. Au chapitre 2, nous étudierons d'autres méthodes permettant les mises à jour.

Une fois la configuration terminée, sélectionnez `Finish` (accédé avec la touche Tab). Selon les options que vous aurez modifiées, il se peut que l'outil `raspi-config` redémarre le système. Sinon, vous serez redirigé vers la ligne de commande sur laquelle vous saisirez :

```
pi@raspberrypi ~ $ sudo reboot
```

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

afin que votre Pi redémarre et prenne en compte vos nouveaux paramètres. Si tout se passe correctement (et si vous avez choisi de démarrer directement dans l'environnement de bureau graphique), vous verrez le gestionnaire de fenêtres Openbox démarrer sur LXDE *(Lightweight X11 Desktop Environment)*. Votre Raspberry Pi est maintenant configuré et prêt !

Éteindre le Raspberry Pi

Il n'existe pas de bouton d'alimentation sur le Raspberry Pi, quoique les nouvelles cartes soient équipées d'une broche permettant d'accueillir un bouton de réinitialisation. Pour éteindre correctement le système d'exploitation, allez dans le menu Logout de l'interface graphique et sélectionnez `shutdown`.

Vous pouvez aussi éteindre le Pi depuis la ligne de commande en tapant :

```
pi@raspberrypi ~ $ sudo shutdown -h now
```

ou bien une version plus courte :

```
pi@raspberrypi ~ $ sudo Ralt
```

Veillez bien à arrêter le système d'exploitation comme indiqué ici et à ne pas débrancher sauvagement votre Pi. Si vous coupez le courant sans arrêter le système correctement, vous risquez d'endommager la carte SD.

En cas de problème

Si les choses ne fonctionnent pas comme prévu, voici une liste des erreurs les plus courantes et des étapes souvent sautées. En cas de dysfonctionnement, vérifiez par conséquent tous les points suivants.

- Est-ce que la carte SD se trouve dans son support et est-ce qu'elle est bien enfoncée dedans ? Est-ce que vous possédez une carte SD compatible ?

- Est-ce que l'image disque a été écrite correctement sur la carte ? Essayez de la recopier et de la lire avec un autre lecteur de carte, elle doit contenir deux partitions.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

- Est-ce que la carte SD est protégée en écriture ? Un petit interrupteur à glissière se trouve sur le côté de la carte et il se pourrait qu'il soit dans la mauvaise position.

- Vérifiez l'intégrité de votre image disque d'origine. Pour cela, vous pouvez utiliser un logiciel de signature numérique : faites-lui calculer la signature SHA *(Secure Hash Algorithm)* de l'image et comparez le résultat à l'empreinte de 40 caractères publiée sur la page de téléchargements.

- Est-ce que le Pi redémarre ? A-t-il des problèmes intermittents ? Vérifiez votre alimentation car une carte sous-alimentée aura parfois l'air de fonctionner en souffrant d'erreurs étranges.

- Vous avez un *kernel panic* (panique du noyau) au démarrage ? Il s'agit du BSOD (*Blue Screen of Death* ou écran bleu de la mort, l'ordinateur a planté) sous Windows. La plupart du temps, cela vient d'un problème avec un appareil sur le hub USB. Essayez de débrancher les appareils USB, puis redémarrez.

Si ces astuces ne vous sont pas d'un grand secours, rendez-vous sur la page de dépannage du wiki consacrée à Raspberry Hub (http://elinux.org/R-Pi_Troubleshooting). Vous y trouverez des solutions pour toutes sortes de problèmes rencontrés par d'autres utilisateurs.

Quelle carte possédez-vous ?

Quand vous sollicitez de l'aide par e-mail ou sur un forum, il est souvent utile d'indiquer la version de votre système d'exploitation et le modèle de votre carte. Pour connaître la version de votre système d'exploitation, saisissez la commande suivante dans un terminal :

```
cat /proc/version
```

Pour connaître la version de votre carte, saisissez :

```
cat /proc/cpuinfo
```

Pour aller plus loin

- À propos du Raspberry Hub : hébergé par elinux.org, le site http://elinux.org/RPi_Hub est une mine d'informations sur le matériel compatible avec le Pi et sa configuration.

- À propos des périphériques compatibles avec le Raspberry Pi : consultez-en la liste sur http://elinux.org/RPi_VerifiedPeripherals.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

CHAPITRE 2

À LA DÉCOUVERTE DE LINUX POUR LE RASPBERRY PI

Pour profiter au mieux de toutes les possibilités offertes par votre Raspberry Pi, il est souhaitable de connaître un peu Linux. Dans ce chapitre, nous vous proposons donc un tour d'horizon de ce système d'exploitation. Nous vous présenterons également un certain nombre de concepts et de commandes, qui vous permettront d'être à l'aise avec le système de fichiers. Vous verrez comment installer des paquets en ligne de commande ou avec l'interface graphique, et vous apprendrez à utiliser des outils dont vous aurez régulièrement besoin.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

La distribution Raspbian est livrée avec LXDE *(Lightweight X11 Desktop Environment)*, un environnement de bureau graphique simplifié qui tourne sur des ordinateurs Unix et Linux depuis les années 1980. Une partie des outils visibles sur le Bureau et dans le menu (l'éditeur de texte Leafpad et le shell LX Terminal, par exemple) sont fournis avec LXDE.

Openbox est un gestionnaire de fenêtres *(window manager)* qui contrôle l'aspect des fenêtres et des menus sous LXDE. Pour modifier l'apparence de votre Bureau, lancez simplement l'outil de configuration d'Openbox. Pour cela, cliquez sur le menu Desktop en bas à gauche et choisissez Other>Openbox Configuration Manager. Contrairement à Mac OS X et Windows, il est facile de personnaliser votre environnement de travail ou d'installer d'autres gestionnaires de fenêtres. D'autres distributions Linux pour le Raspberry Pi offrent des environnements différents, destinés à des applications telles que les lecteurs multimédias, les systèmes téléphoniques et les pare-feu pour les réseaux. Pour obtenir la liste complète et actualisée des distributions, consultez la page suivante : http://elinux.org/RPi_Distributions.

Les différents outils que vous serez amené à utiliser fréquemment sont les suivants.

- **Le gestionnaire des fichiers *(File Manager)***

  Si vous préférez déplacer vos fichiers sans passer par la ligne de commande (voir plus loin), sélectionnez File Manager dans le menu Accessories. Vous pourrez alors naviguer dans le système de fichiers grâce à des icônes et des dossiers, comme vous en avez probablement l'habitude.

- **Le navigateur web**

  Le navigateur web par défaut, Midori (http://twotoasts.de/index.php/midori/), a été élaboré de manière à pouvoir fonctionner avec peu de ressources. On oublie facilement la multitude de tâches effectuées par les navigateurs de nos jours. Raspbian a été conçu pour être une distribution très légère et certaines fonctionnalités que vous attendez d'un navigateur peuvent faire défaut. Par exemple, les plug-ins Flash et Java ne sont pas installés sur Midori, vous ne pourrez donc pas accéder à YouTube, ni lire de vidéos HTML5. Nous verrons plus loin comment

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

installer d'autres logiciels tels que Java. Vous trouverez les outils et les options de Midori dans le menu déroulant situé en haut à droite de la fenêtre (figure 2-2). Vous pouvez également installer d'autres navigateurs comme NetSurf (http://www.netsurf-browser.org/) et Dillo (http://www.dillo.org/).

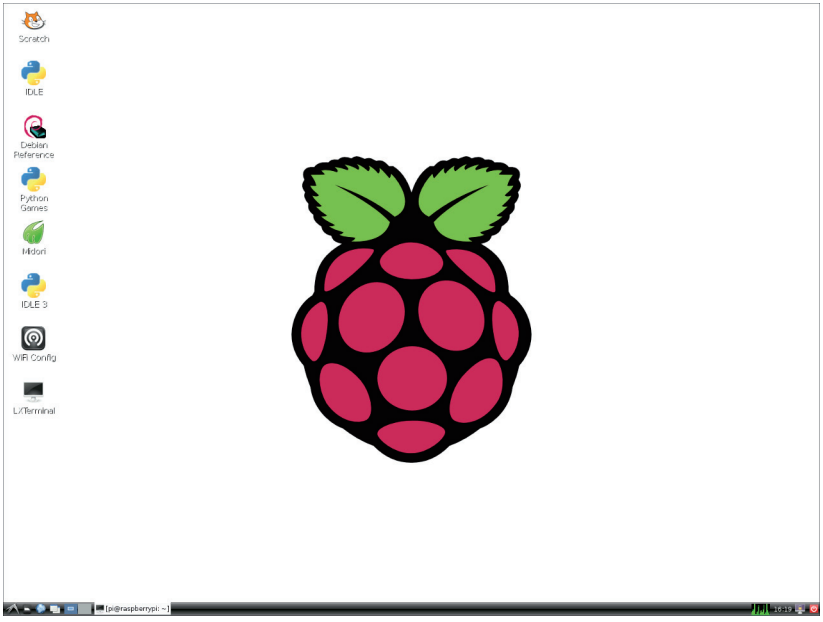


Figure 2-1. L'interface graphique

La vidéo et l'audio

La lecture multimédia est assurée par omxplayer, un logiciel encore en version bêta, et qui existe seulement en ligne de commande. omxplayer a été conçu pour fonctionner avec le GPU *(Graphics Processing Unit)* du SoC Broadcom, ce qui libère le CPU de lourds calculs. D'autres logiciels gratuits comme VLC et mPlayer fonctionnent moins bien car ils ne sont pas adaptés pour travailler avec le GPU.

Pour réduire son prix de vente, le Raspberry Pi n'est pas fourni avec les licences permettant de décoder certains types de vidéos. Si vous souhaitez regarder des programmes télévisés ou des DVD encodés au format MPEG-2 (ou le format VC-1 de Microsoft), vous serez obligé

d'acheter une clé de licence sur la boutique en ligne de la fondation Raspberry Pi (http://www.raspberrypi.com/). Une licence de décodage et d'encodage H.264 (MPEG-4) est toutefois livrée avec le Raspberry Pi.

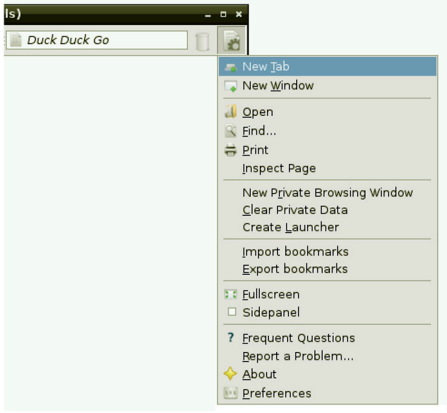


Figure 2–2. Le menu déroulant du navigateur web

L'éditeur de texte

Leafpad (http://wiki.lxde.org/en/Leafpad) est l'éditeur de texte par défaut disponible dans le menu principal. En ligne de commande, nano (http://www.nano-editor.org) est un autre éditeur très simple et facile à utiliser. Les autres éditeurs de texte traditionnels Unix (comme vim et emacs) ne sont pas installés par défaut mais sont faciles à obtenir (voir la section « Installation de nouveaux logiciels », page 41).

La fonction copier-coller

Les fonctions Copier et Coller fonctionnent la plupart du temps sans problème entre les applications, mais certains programmes ne gèrent pas ces fonctions correctement. Si votre souris est équipée d'une molette centrale, sélectionnez le texte à copier comme vous le feriez habituellement (cliquez-glissez avec le bouton gauche de la souris). Placez ensuite le curseur de la souris sur la fenêtre de destination et collez le texte en appuyant sur la molette centrale.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Le shell

Beaucoup de tâches nécessitent l'utilisation de la ligne de commande. Le programme LXTerminal vous permet d'accéder à cette ligne de commande et au shell. Le shell par défaut sur Raspbian, ainsi que sur la majorité des systèmes Linux, s'appelle Bourne-Again Shell (ou bash, http://www.gnu.org/software/bash/manual/bashref.html). Il existe une alternative appelée dash (http://en.wikipedia.org/wiki/Debian_Almquist_shell). Vous pouvez changer de shell par le menu Programme ou par la commande `chsh`.

Utilisation de la ligne de commande

Si cela peut vous aider, vous pouvez imaginer la ligne de commande comme un jeu d'aventure textuel à la Zork, dans lequel les fichiers et le système de fichiers remplacent les monstres et les labyrinthes. Si cette métaphore ne vous aide pas, ne vous inquiétez pas. Toutes les commandes et tous les concepts que nous allons aborder dans cette section sont des standards Linux qu'il est intéressant de connaître.

Avant toute chose, ouvrez le programme LXTerminal (figure 2-3). Deux fonctionnalités vous faciliteront la vie dans le shell : l'autocomplétion et la mémorisation des dernières commandes. La plupart du temps, vous n'aurez qu'à saisir que les premiers caractères d'une commande ou du nom d'un fichier, puis à appuyer sur la touche Tab. Le shell vous proposera de compléter la commande en fonction des noms de fichier qui se trouvent dans le répertoire courant, ou des programmes situés dans les répertoires souvent utilisés (le shell va chercher les programmes exécutables à des emplacements tels que `/bin` ou `/usr/bin/`).

Si vous appuyez sur la flèche du haut dans la ligne de commande, vous reviendrez en arrière dans l'historique des commandes. Cette fonction s'avère fort utile si vous avez mal saisi un caractère dans une longue chaîne de commandes ou si vous devez répéter et modifier des opérations.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

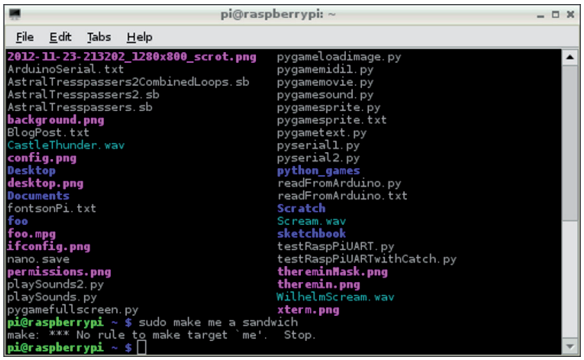


Figure 2-3. LXTerminal vous permet d'accéder à la ligne de commande (ou shell).

LES FICHIERS ET LE SYSTÈME DE FICHIERS

Le tableau 2-1 liste certains des répertoires les plus importants dans le système de fichiers de Raspbian. La plupart d'entre eux suivent le standard Linux pour leur structure. Quelques-uns sont spécifiques au Raspberry Pi. Le répertoire `/sys` permet d'accéder à tout le matériel de votre Pi.

Tableau 2-1. Certains des répertoires les plus importants dans le système de fichiers de Raspbian

Répertoire	Description
`/`	La « racine », c'est l'entrée du labyrinthe du système de fichiers.
`/bin`	Programmes et commandes pour tous les utilisateurs
`/boot`	Tous les fichiers nécessaires au démarrage du noyau
`/dev`	Fichiers spéciaux qui représentent les périphériques installés. Des fichiers virtuels qui permettent de communiquer avec des périphériques, comme les ports série, l'écran...
`/etc`	Fichiers de configuration du système
`/etc/init.d`	Scripts pour démarrer les services du système
`/etc/X11`	Fichiers de configuration pour le mode graphique X11
`/home`	Répertoires personnels des utilisateurs
`/home/pi`	Répertoire personnel pour l'utilisateur Pi
`/lib`	Bibliothèques partagées et modules du noyau
`/media`	Points de montage pour les supports amovibles (c'est ici qu'apparaîtra votre clé USB Flash, par exemple)
`/proc`	Répertoire virtuel contenant des informations sur les processus en cours et le système d'exploitation.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

`/sbin`	Programmes pour l'administration du système
`/sys`	Répertoire spécial contenant des fichiers qui contrôlent le matériel.
`/tmp`	Emplacement dans lequel les programmes peuvent créer des fichiers temporaires (ils sont effacés à chaque redémarrage).
`/usr`	Programmes et données accessibles par tous les utilisateurs
`/usr/bin`	La plupart des programmes du système d'exploitation se trouvent ici.
`/usr/games`	Oui, des jeux
`/usr/lib`	Autres bibliothèques partagées pour fournir des fonctions standards à la plupart des applications
`/usr/local`	Logiciels spécifiques à une machine
`/usr/sbin`	Autres programmes d'administration du système
`/usr/share`	Fichiers communs aux applications, comme les icônes et les polices de caractères
`/usr/src`	Linux est open source, voici donc les codes sources.
`/var`	Journal des messages du système et données en attente
`/var/backups`	Sauvegardes des fichiers système les plus critiques
`/var/cache`	Contient les fichiers de cache des programmes (comme `apt-get` ou un navigateur web).
`/var/log`	Archive des messages du système *(logs)*
`/var/mail`	Les e-mails sont mis en attente ici, si votre Pi est paramétré pour les gérer.
`/var/spool`	Les données en attente (e-mails entrants, documents à imprimer, etc.)

Votre répertoire actuel s'affichera devant l'invite de commande, aussi appelée le *prompt*. Sous Linux, votre répertoire personnel *(home)* est aussi représenté avec le caractère ~ pour faire plus court. À l'ouverture de LXTerminal, vous vous trouverez dans votre répertoire personnel et votre prompt ressemblera à ceci :

```
pi@raspberry ~ $
```

Voici une explication de ce prompt :

```
pi❶@raspberry❷ ~❸ $❹
```

❶ Votre nom d'utilisateur, `pi`, suivi du caractère arobase (`@`).

❷ Le nom de votre ordinateur (`raspberry` est le nom par défaut).

❸ Le répertoire courant *(current working directory)* du shell. Lorsque vous démarrez une session, vous vous trouverez toujours dans votre répertoire personnel (`~`).

❹ Le prompt du shell (ou invite de commande). Tout texte que vous saisirez apparaîtra à sa droite. Appuyez sur la touche Entrée après chacune de vos commandes pour les exécuter.

La commande `cd` *(change directory)* vous permet de naviguer dans le système de fichiers. Les deux commandes suivantes ont le même résultat (aller dans le répertoire personnel) pour l'utilisateur du Pi :

```
cd /home/pi/
cd ~
```

Si le nom du répertoire que vous indiquez commence par le caractère `/`, il sera interprété comme un chemin « absolu » (c'est-à-dire par rapport à la racine). Sinon, le répertoire sera considéré comme « relatif », c'est-à-dire par rapport au répertoire courant. Vous pouvez aussi employer `.` pour indiquer le répertoire courant et `..` pour le répertoire parent. Par exemple, voici comment remonter vers la racine du système de fichiers :

```
pi@raspberrypi ~ $ cd ..
pi@raspberrypi /home $ cd ..
```

Vous pouvez aussi y arriver directement avec le chemin absolu (`/`) :

```
pi@raspberrypi ~ $ cd /
```

Dès que vous arrivez dans un répertoire, utilisez la commande `ls` pour afficher tous les fichiers et sous-répertoires :

```
pi@raspberrypi / $ ls
bin   etc   lost+found   opt   run   srv   var
boot  home    media       proc   sbin      sys   usr
dev   lib   mnt      root   selinux   tmp
```

La plupart des commandes proposent des paramètres supplémentaires ou « arguments » pour activer des options et comportements divers. Par exemple, l'argument `-l` de `ls` fournit une liste plus détaillée montrant la taille des fichiers, leurs dates et leurs permissions :

```
pi@raspberrypi / $ ls –l
total 8
drwxr-xr-x 2 pi pi 4096 Oct 12 14:26 Desktop
drwxrwxr-x 2 pi pi 4096 Jul 20 14:07 python_games
```

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

L'argument `-a` liste tous les fichiers, même ceux qui sont normalement invisibles :

```
pi@raspberrypi ~ $ ls -la
total 80
drwxr-xr-x 11 pi   pi   4096 Oct 12 14:26 .
drwxr-xr-x 3  root root 4096 Sep 18 07:48 ..
-rw------- 1  pi   pi     25 Sep 18 09:22 .bash_history
-rw-r--r-- 1  pi   pi    220 Sep 18 07:48 .bash_logout
-rw-r--r-- 1  pi   pi   3243 Sep 18 07:48 .bashrc
drwxr-xr-x 6  pi   pi   4096 Sep 19 01:19 .cache
drwxr-xr-x 9  pi   pi   4096 Oct 12 12:57 .config
drwx------ 3  pi   pi   4096 Sep 18 09:24 .dbus
drwxr-xr-x 2  pi   pi   4096 Oct 12 14:26 Desktop
-rw-r--r-- 1  pi   pi     36 Sep 18 09:35 .dmrc
drwx------ 2  pi   pi   4096 Sep 18 09:24 .gvfs
drwxr-xr-x 2  pi   pi   4096 Oct 12 12:53 .idlerc
-rw------- 1  pi   pi     35 Sep 18 12:11 .lesshst
drwx------ 3  pi   pi   4096 Sep 19 01:19 .local
-rw-r--r-- 1  pi   pi    675 Sep 18 07:48 .profile
drwxrwxr-x 2  pi   pi   4096 Jul 20 14:07 python_games
drwx------ 4  pi   pi   4096 Oct 12 12:57 .thumbnails
-rw------- 1  pi   pi     56 Sep 18 09:35 .Xauthority
-rw------- 1  pi   pi    300 Oct 12 12:57 .xsession-
                                          errors
-rw------- 1  pi   pi   1391 Sep 18 09:35 .xsession-
                                          errors.old
```

Utilisez la commande `mv` pour renommer ou déplacer un fichier. La commande `touch` peut être employée pour créer un fichier vide :

```
pi@raspberrypi ~ $ touch foo
pi@raspberrypi ~ $ ls
foo Desktop python_games
pi@raspberrypi ~ $ mv foo baz
pi@raspberrypi ~ $ ls
baz Desktop python_games
```

Pour effacer un fichier, utilisez la commande `rm`. Si vous souhaitez supprimer un répertoire, vous pouvez utiliser `rmdir` si le répertoire est vide, et `rm -r` s'il ne l'est pas. L'argument `-r` est ajouté à la commande `rm`, il indique que tout le contenu du répertoire doit être supprimé récursivement (y compris les sous-répertoires).

Si vous voulez connaître tous les arguments proposés par une commande particulière, utilisez la commande `man` ou l'argument `--help` :

```
pi@raspberrypi ~ $ man curl
pi@raspberrypi ~ $ rm --help
```

Pour créer un nouveau répertoire, utilisez `mkdir`.

Pour regrouper tous les fichiers d'un répertoire dans un seul fichier d'archive, utilisez la commande `tar` (créée à l'origine pour gérer les archives sur bandes magnétiques). Vous noterez que beaucoup d'archives de fichiers, d'applications ou de codes sources sont distribués sous forme de fichiers de type `tar`. Ces fichiers sont souvent compressés avec la commande `gzip` et sont semblables à des archives au format `.zip` sur d'autres plates-formes. Essayez ceci :

```
pi@raspberrypi ~ $ mkdir myDir
pi@raspberrypi ~ $ cd myDir
pi@raspberrypi myDir $ touch foo bar baz
pi@raspberrypi myDir $ cd ..
pi@raspberrypi ~ $ tar -cf myDir.tar myDir
pi@raspberrypi ~ $ gzip myDir.tar
```

Les deux dernières commandes peuvent être combinées avec l'argument `-z` de `tar` :

```
pi@raspberrypi ~ $ tar -czf myDir.tar.gz myDir
```

Vous obtenez une archive au format `.tar.gz` contenant tous les fichiers de ce répertoire. Elle pourra être sauvegardée ou encore partagée par e-mail ou via Internet.

D'autres commandes Linux

Le succès de Linux (et d'Unix) tient, entre autres, à sa philosophie modulaire, qui encourage la création de systèmes complexes à partir des programmes simples et qui peuvent communiquer entre eux facilement. Nous allons comprendre comment y arriver en examinant deux techniques : les tuyaux et les redirections.

Les tuyaux sont un moyen simple de chaîner deux programmes ensemble pour que la sortie de l'un aille dans l'entrée de l'autre. La plupart des programmes Linux peuvent lire des données sur leur « entrée standard » (souvent appelée `stdin`), écrire des données sur leur « sortie standard »

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

(stdout) et envoyer les messages d'erreur sur leur erreur standard (stderr). Un tuyau vous permet de connecter le stdout d'un programme au stdin d'un autre (figure 2-4) au moyen de l'opérateur |, comme dans cet exemple :

```
pi@raspberrypi ~ $ ls -la | less
```

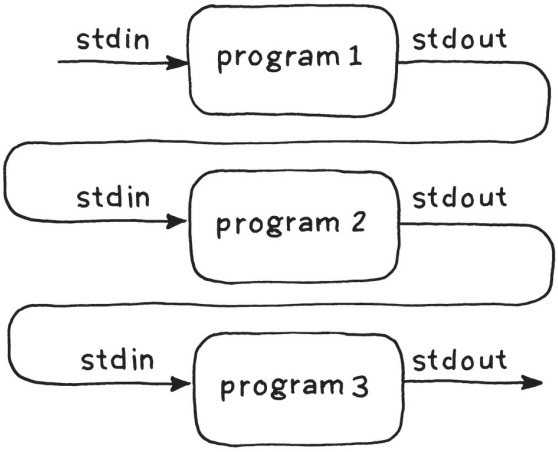


Figure 2-4. Les tuyaux permettent de chaîner des programmes simples pour réaliser des traitements complexes.

Appuyez sur la touche Q pour sortir du programme less.

Et maintenant, quelque chose de plus amusant à essayer :

```
pi@raspberrypi ~ $ sudo cat /boot/kernel.img | aplay
```

Il vaut mieux vérifier le volume des haut-parleurs avant de tenter cela. En effet, cette commande lit l'image du kernel et « crache » tous ses bits dans le lecteur audio. Vous pouvez entendre votre kernel !

Comme vous le verrez plus loin dans d'autres exemples de ce livre, nous allons aussi utiliser les redirections afin d'exécuter une commande et envoyer sa sortie stdout dans un fichier. L'autre philosophie de Linux, héritée de son inspiration Unix, est que « tout est fichier ». Linux traite donc beaucoup de choses de la même manière qu'il traite des fichiers ordinaires (comme les broches d'entrées et sorties généralistes de la carte Pi). Les redirections sont souvent très utiles. Pour rediriger la sortie stdout d'un programme dans un fichier, utilisez l'opérateur > :

```
pi@raspberrypi ~ $ ls > directoryListing.txt
```


Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

La liste des fichiers du répertoire courant est alors écrite dans un fichier.

Des touches de contrôle spéciales

En plus de l'autocomplétion (touche Tab) et de l'historique des commandes (flèche de direction vers le haut), il existe aussi des combinaisons de touches dont vous aurez besoin pour utiliser le shell. En voici quelques-unes :

• Ctrl + C
Termine le programme en cours d'exécution. Cette combinaison ne fonctionne pas pour tous les programmes, comme les éditeurs de texte qui l'interprètent différemment.

• Ctrl + D
Termine la session dans le shell, lorsque cette combinaison de touches est tapée dans une ligne de commande vide (ne saisissez rien après le **$** avant de taper Ctrl + D).

• Ctrl + A
Déplace le curseur au début de la ligne (équivalent de la touche Début).

• Ctrl + E
Déplace le curseur à la fin de la ligne (équivalent de la touche Fin).

Il existe d'autres touches et combinaisons spéciales, nous ne vous présentons que celles que vous serez amené à utiliser tous les jours.

Vous voudrez parfois afficher le contenu d'un fichier sur votre écran. S'il s'agit d'un fichier texte et que vous avez envie de le lire page à page, utilisez `less` :

```
pi@raspberrypi ~ $ ls > flob.txt
pi@raspberrypi ~ $ less flob.txt
```

Si vous voulez simplement envoyer tout le contenu d'un fichier sur la sortie standard, utilisez `cat` (pour *concatenate*). Ceci peut s'avérer pratique si vous souhaitez transmettre votre fichier à un autre programme ou le rediriger quelque part. Et puisque sous Linux, « tout est fichier », on peut « brancher » le générateur de nombres aléatoires sur la mémoire de l'écran :

```
pi@raspberrypi ~ $ cat /dev/urandom > /dev/fb0
```

La commande `cat` permet aussi de concaténer des fichiers, dont on indique les noms en argument. Par exemple, l'exemple suivant crée un fichier, le recopie, puis concatène ces deux fichiers dans un troisième :

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

```
pi@raspberrypi ~ $ ls > wibble.txt
pi@raspberrypi ~ $ cat wibble.txt > wobble.txt
pi@raspberrypi ~ $ cat wibble.txt wobble.txt > wubble.txt
```

Pour examiner les dernières lignes d'un fichier, comme les derniers ajouts à un fichier de log, utilisez `tail`. Pour voir le début, utilisez `head`.

Si vous cherchez un mot ou une chaîne de caractères à l'intérieur d'un ou plusieurs fichiers, utilisez le vénérable programme `grep`. Ici, on lui demande de chercher le mot-clé « Puzzle » dans tous les fichiers des sous-répertoires :

```
pi@raspberrypi ~ $ grep Puzzle */*
```

`grep` est un outil puissant grâce à son système d'*expressions régulières* sophistiquées spécialement conçu pour lui. Ces dernières permettent de décrire la suite de caractères à rechercher mais sa syntaxe devient rapidement absconse et participe injustement à la réputation de Linux comme étant un système compliqué pour les novices.

LES PROCESSUS

Chaque programme du Pi s'exécute séparément et simultanément, et une application peut se décomposer en un ou plusieurs processus. À tout moment, il peut y avoir des dizaines de processus en cours d'exécution. Quand vous démarrez le Pi pour la première fois, 75 processus environ sont lancés, chacun réalisant une tâche ou fournissant un service différent. Si vous voulez voir les processus, lancez le programme `top`, qui affichera également des informations sur l'utilisation du processeur et l'occupation de la mémoire. Ce programme vous aidera à reconnaître les processus qui consomment le plus de ressources. Utilisez la commande `ps` pour afficher tous les processus et leurs identifiants (PID pour *Process IDentifier*, première colonne à droite). Essayez :

```
pi@raspberrypi ~ $ ps -aux | less
```

Vous pouvez filtrer la sortie avec un mot-clé grâce à `grep` :

```
pi@raspberrypi ~ $ ps -aux | grep bash
```

Il peut arriver que vous vouliez arrêter un processus qui ne fonctionne pas correctement ou qui ne répond plus. Pour ce faire, utilisez `ps` pour trouver son identifiant, puis `kill` pour l'arrêter.

```
pi@raspberrypi ~ $ kill 95689
```

Vous ne pouvez pas tuer les processus que vous n'avez pas lancés vous-même, en particulier ceux du système d'exploitation. Nous allons justement parler de `sudo`, commande qui vous permettra parfois d'intervenir dans ce genre de cas.

SUDO ET LES PERMISSIONS

Linux est un système d'exploitation multi-utilisateur. En règle générale, chacun est propriétaire de ses propres fichiers et a le droit de les créer, de les modifier et de les effacer dans son propre espace du système de fichiers.



En tant qu'utilisateur **`pi`**, vous ne risquez pas de causer beaucoup de dégâts au système. Cependant, quand vous vous connectez avec **`sudo`** au compte administrateur (appelé **`root`**), vous pouvez faire des ravages, par accident ou intentionnellement. Il faut faire attention quand vous utilisez **`sudo`**, surtout quand vous déplacez ou modifiez des fichiers. En cas d'accident, il n'est pas possible d'annuler les commandes et vous devrez charger une nouvelle image dans la carte SD (voir annexe A).

Il existe quelques outils comme `sudo` qui permettent aux utilisateurs d'obtenir les droits d'administration sans les risques (et les responsabilités) qui peuvent exister en étant constamment connecté au compte `root`. Vous utiliserez souvent `sudo` pour manipuler le matériel directement, ou lorsque vous souhaiterez modifier les paramètres du système.

Chaque fichier appartient à un utilisateur spécifique et à un groupe en particulier. Utilisez les commandes `chown` et `chgrp` pour changer le propriétaire ou le groupe d'un fichier. À noter que vous devez être l'utilisateur `root` pour employer l'une de ces deux commandes :

```
pi@raspberrypi ~ $ sudo chown pi garply.txt
```

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

```
pi@raspberrypi ~ $ sudo chgrp staff plugh.txt
```

Chaque fichier possède aussi un ensemble de permissions (ou droits d'accès) qui spécifient si un fichier peut être lu, écrit ou exécuté. Ces permissions peuvent être accordées au propriétaire du fichier, au groupe ou à toute autre personne (figure 2-5).

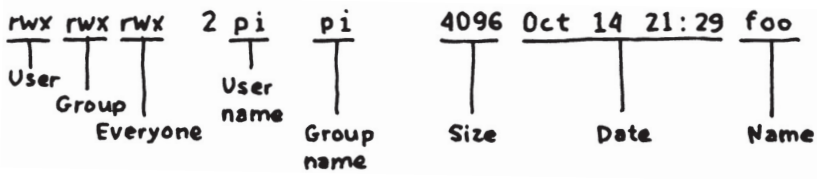


Figure 2-5. Les permissions pour le propriétaire du fichier, le groupe ou tout autre utilisateur. La commande affiche les permissions pour chaque fichier, à commencer par les autorisations de lecture/écriture/exécution pour le propriétaire, le groupe et le reste du système. Avec sudo ou dans le compte root, toutes les permissions sont accordées.

Pour définir des autorisations individuelles, utilisez la commande `chmod` (pour *change mode*). Les arguments de `chmod` sont décrits dans le tableau 2-2.

Tableau 2-2. Arguments pouvant être utilisés avec chmod

Argument	Description
`u`	Utilisateur
`g`	Groupe
`o`	Autres utilisateurs (ne faisant pas partie du groupe)
`a`	Tout le monde (affecte u, g et o)
`r`	Autorisation de lire
`w`	Autorisation d'écrire
`x`	Autorisation d'exécuter
`+`	Ajouter une autorisation
`-`	Supprimer une autorisation

Voici quelques exemples qui vous montrent comment combiner ces arguments :

```
chmod u+rwx,g-rwx,o-rwx wibble.txt ❶
chmod g+wx wobble.txt ❷
```


Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

```
chmod -xw,+r wubble.txt ❸
```

❶ Autorise uniquement l'utilisateur à lire, écrire et exécuter.

❷ Ajoute une permission d'écriture et d'exécution pour les membres du groupe.

❸ Permission de lecture uniquement pour tout le monde.

Mais comme cela peut vite devenir illisible, les arguments sont souvent condensés en notation octale :

```
chmod 744 wabble.txt
```

`744` signifie « tout le monde peut lire le fichier mais seul le propriétaire peut le modifier ou l'exécuter ». Comme il y a peu de combinaisons d'arguments utiles ou sûres, on n'a besoin de mémoriser que quelques nombres.

À noter que la seule chose qui protège votre système de fichiers et vos fichiers est votre mot de passe. Par conséquent, nous vous conseillons d'en choisir un qui soit très fiable et difficile à trouver. Pour le modifier, utilisez la commande `passwd`, notamment lorsque vous connectez votre Raspberry Pi à un réseau.

LE RÉSEAU

Lorsque vous serez connecté à un réseau, vous utiliserez régulièrement un certain nombre d'outils Linux. Par exemple, si votre connexion est défectueuse, utilisez `ifconfig` qui affiche toutes les interfaces de votre Pi ainsi que les éventuelles adresses IP associées (figure 2-6).

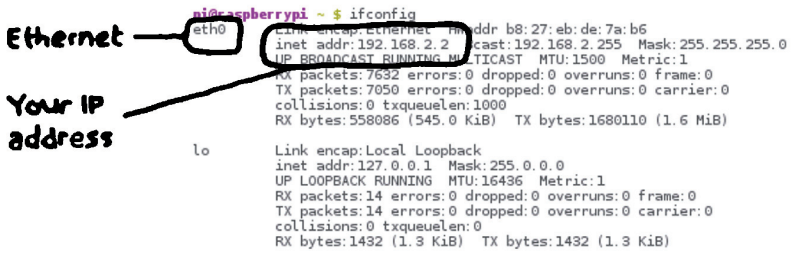


Figure 2-6. La commande ifconfig affiche les informations relatives à toutes vos interfaces réseau.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

La commande `ping` est l'outil le plus basique pour vérifier si une connexion réseau fonctionne. `ping` teste s'il y a une connexion bidirectionnelle entre deux adresses IP sur un réseau local ou sur Internet. Notez que beaucoup de sites web bloquent les paquets de `ping`, donc vous aurez probablement besoin de « pinger » plusieurs sites avant de valider votre connexion :

```
ping yahoo.com
ping altavista.com
ping lycos.com
ping netscape.com ❶
```

❶ Ne répond pas, bien que son site web fonctionne.

Pour vous connecter sur un autre ordinateur à distance, en toute sécurité et avec des mots de passe cryptés, vous utiliserez `ssh`, le terminal sécurisé *(Secure Shell)*. L'ordinateur distant doit faire tourner un serveur de protocole SSH pour que ça marche. Le programme client `ssh` est fourni par défaut avec Raspbian. C'est le meilleur moyen de travailler sur votre Raspberry Pi sans l'équiper d'un moniteur ou d'un clavier dédiés (voir l'encadré suivant « Travailler sans écran »).

Dans la même catégorie que `ssh`, le programme `sftp` vous permet de transférer des fichiers d'un ordinateur à un autre en sécurité. Le programme `scp`, quant à lui, vous permet de copier les fichiers d'un compte à un autre par un réseau ou par Internet (il est plus puissant que `sftp`). Le point commun de ces outils est qu'ils utilisent tous le protocole SSL *(Secure Sockets Layer)* pour transférer les informations dans une connexion cryptée. Ces outils standards font partie des piliers de Linux.

Travailler sans écran

Si vous avez envie de travailler sur votre Raspberry Pi sans moniteur ni clavier ni souris, voici quelques astuces pour le configurer dans ce mode, appelé *headless*. Si vous n'avez besoin que d'un accès à la ligne de commande, vous pouvez simplement brancher votre Pi à un réseau et utiliser un client **`ssh`** pour vous y connecter (nom d'utilisateur : **`pi`**, mot de passe par défaut : **`raspberry`**). L'application Terminal du Mac fonctionnera, ainsi que PuTTY (http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html) sous Windows ou Linux, ou encore **`ssh`** sous Linux. Le serveur **`ssh`** du Raspberry Pi est activé par défaut (exécutez **`raspi-config`** s'il ne se lance pas au démarrage).

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Un autre moyen de vous connecter à votre Pi via un réseau consiste à installer un serveur VNC *(Virtual Network Computing)* sur le Pi et à vous y connecter grâce à un client VNC. Vous pouvez alors faire tourner une interface graphique complète dans une fenêtre de votre ordinateur portable ou de bureau. Cette solution est idéale pour un environnement de développement portable. Rendez-vous sur la page du hub Raspberry Pi (http://elinux.org/RPi_VNC_Server) relative à l'installation détaillée de TightVNC (http://www.tightvnc.com/), un serveur VNC léger.

Le répertoire /etc

Le répertoire `/etc` contient tous les fichiers de configuration du système et les scripts de démarrage. Lorsque vous avez lancé le programme de configuration `raspi-config` au premier démarrage, vous avez modifié des valeurs dans différents fichiers du répertoire `/etc`. Pour éditer ces fichiers, utilisez la commande `sudo` pour obtenir les droits du super utilisateur `root`. Ensuite, utilisez un éditeur de texte comme `nano` et lancez-le avec `sudo` :

```
pi@raspberrypi ~ $ sudo nano /etc/hosts
```

Réglage de la date et de l'heure

Les ordinateurs portables et les ordinateurs de bureau standards sont équipés de circuits et d'une pile (une petite batterie rechargeable ou une pile bouton au lithium) pour sauvegarder et rafraîchir l'heure et la date. Le Raspberry Pi n'en a pas, mais Raspbian est configuré pour synchroniser l'heure et la date automatiquement quand il est branché à un réseau, en communiquant avec un serveur NTP (*Network Time Protocol*, le protocole de synchronisation de l'heure).

Il est important que votre Pi soit à l'heure, surtout pour certaines applications (voir l'exemple du chapitre 7 utilisant `cron` pour contrôler une

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

lampe). Pour régler l'heure et la date manuellement, utilisez le programme `date` :

```
$ sudo date --set="Sun Nov 18 1:55:16 EDT 2012"
```

La commande `man date` liste tous les arguments, toutes les options et tous les formats pour l'affichage et la modification de l'heure et de la date.

Installation de nouveaux logiciels

En ce qui concerne la gestion de l'installation et de la mise à jour des logiciels et des applications, Linux domine incontestablement les autres systèmes d'exploitation tels Windows et Mac OS. Les gestionnaires de paquets *(Package Managers)* gèrent automatiquement le téléchargement et l'installation des logiciels. Chaque distribution dispose de son propre gestionnaire, couplé avec une énorme bibliothèque de logiciels précompilés et prêts à télécharger : plus de 29 000 paquets pour Debian, dont Raspbian est dérivé. Vous trouverez certainement ce qu'il vous faut et même beaucoup plus !

Les gestionnaires de paquets détectent aussi les autres paquets nécessaires au fonctionnement de chaque application. Ils les téléchargent et les installent dans le bon ordre. En effet, fidèles à l'approche modulaire de Linux, de nombreuses applications dépendent d'autres logiciels et partagent même des modules entre eux. Les gestionnaires de paquets sous Linux sont remarquablement robustes et assurent la cohérence et la stabilité du système.

Raspbian est une distribution plutôt minimaliste et vous aurez rapidement besoin de télécharger et d'installer de nouveaux programmes. Les exemples de ce livre utilisent tous la ligne de commande, moyen le plus flexible et le plus rapide pour installer un logiciel.

Le programme `apt-get` avec l'argument `install` va télécharger et installer des logiciels dont vous connaissez déjà le nom. `apt-get` va aussi télécharger les modules nécessaires à votre paquet pour que vous n'ayez pas besoin de chercher partout les dépendances. Les logiciels doivent

être installés par l'administrateur, donc il faut toujours employer `sudo`. Par exemple, la commande suivante installe l'éditeur de texte `emacs` :

```
pi@raspberrypi ~ $ sudo apt-get install emacs
```

Les captures d'écran

Une des premières fonctionnalités dont nous avons eu besoin pour rédiger ce livre a été de pouvoir réaliser des captures d'écran du Raspberry Pi. Pour ce faire, nous avons trouvé un programme nommé **`scrot`** *(SCReenshOT)*. Une autre solution consiste à installer Gimp *(GNU Image Manipulation Program)* ou ImageMagick. Pour installer **`scrot`**, qui a bien fonctionné pour nous, tapez :

```
sudo apt-get install scrot
```

Pour aller plus loin

Linux est un environnement très riche qui recèle tellement de trésors qu'il est impossible de tous les mentionner dans ce livre. Voici quelques ressources qui vous aideront à approfondir vos connaissances :

- *Linux Pocket Guide* de Daniel J. Barrett, paru aux éditions O'Reilly Media en mars 2012. Un pense-bête de référence, très pratique.

- *Linux in a Nutshell* de E. Siever, S. Figgins, R. Love et A. Robbins, paru aux éditions O'Reilly Media en septembre 2009. Une autre référence, plus détaillée.

- Le wiki Debian
  Dans la mesure où Raspbian est basé sur Debian, un grand nombre des informations publiées sur le wiki du projet Debian (http://wiki.debian.org/FrontPage) concernent également Raspbian.

- *The Jargon File*
  Aussi connu comme le *New Hacker's Dictionary*, cette collection de définitions, d'expressions *geeks* et d'anecdotes (http://catb.org/jargon) est une référence historique de la communauté Unix/Linux.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

CHAPITRE 3

PROGRAMMER EN PYTHON

Python est un excellent langage pour les néophytes en programmation. Ses règles sont claires et les développeurs peuvent démarrer rapidement. En outre, la communauté des utilisateurs de Python est très active, favorisant le partage du code et les questions entre développeurs.

Python a rapidement été reconnu comme un langage adapté à la pédagogie par son créateur Guido van Rossum. En 1999, celui-ci a rédigé une proposition largement disséminée, *Computer Programming for Everybody* (« Tout le monde peut programmer un ordinateur », http://www.python.org/doc/essays/cp4e.html). Cet article résumait un projet ambitieux pour enseigner la programmation aux élèves dans les écoles primaires et secondaires avec Python. Plus d'une décennie après, il semble que ce rêve se réalise enfin avec l'arrivée du Raspberry Pi.

Python est un *langage interprété*, ce qui veut dire que vous pouvez écrire un programme ou un script et l'exécuter directement sans le compiler en code machine. Les langages interprétés sont souvent un peu plus rapides à écrire que d'autres, et ils fournissent quelques avantages supplémentaires. Avec Python, par exemple, il n'est pas nécessaire d'indiquer à votre ordinateur si une variable est un nombre, une liste ou une chaîne de caractères. L'interpréteur devine les types de données lors de l'exécution d'un script.

L'interpréteur de Python est capable de fonctionner de deux façons différentes : comme un shell interactif pour exécuter les commandes individuelles, ou comme un programme en ligne de commande pour exécuter les scripts indépendamment. L'environnement intégré de développement (IDE) pour Python fourni avec le Raspberry Pi s'appelle IDLE (figure 3-1).

Quelle version de Python ?

Vous remarquerez qu'il existe deux versions d'IDLE parce que deux versions de Python sont installées sur le Pi. Cela pourrait sembler un peu déroutant mais il y a une logique derrière tout ça. Lors de l'écriture de ce livre, Python 3 était la version la plus récente, mais les changements entre les versions 2 et 3 ont sacrifié une partie de la compatibilité entre les programmes. Même si Python 3 existe depuis des années, il lui a fallu beaucoup de temps pour être adopté, donc beaucoup de bibliothèques de fonctions créées par les utilisateurs ne sont toujours pas à jour. La situation se complique encore un peu quand vous cherchez de la documentation sur Python. Vérifiez que vous avez trouvé des informations valides pour votre version !

Les exemples dans ce livre fonctionnent avec Python 2.7 ou 3.X, sauf mention contraire.

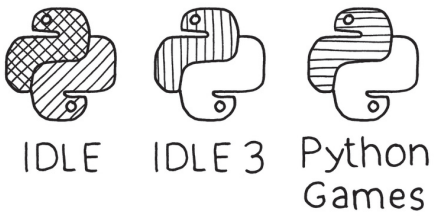


Figure 3-1. Les options de Python sur le Bureau Raspbian : l'environnement de développement intégré IDLE pour Python 2.0 (à gauche), IDLE pour Python 3.0 (au centre) et une collection de jeux créés avec Pygame de inventwithpython.com (à droite)

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Bonjour Python !

Le meilleur moyen pour apprendre Python, c'est de se jeter à l'eau. Vous pouvez choisir n'importe quel éditeur de texte pour commencer à coder, ici nous utilisons IDLE 3. Pour le lancer, double-cliquez sur l'icône IDLE 3 du Bureau ou rendez-vous dans le menu du Bureau, en bas à gauche de votre écran, et choisissez Programming>IDLE 3.

Après quelques secondes, la fenêtre d'IDLE apparaît avec son shell interactif. Le triple chevron (>>>) correspond à l'invite de commande, ou prompt interactif. Quand le prompt est visible, cela signifie que l'interpréteur attend vos commandes. Saisissez le code suivant :

```
>>> print("Saluton Mondo!")
```

Appuyez ensuite sur la touche Entrée. Python exécutera alors cette commande et vous verrez le résultat dans la fenêtre du shell. Notez que la commande `print()` a été modifiée dans Python 3.0. Si vous obtenez un message d'erreur de syntaxe, vérifiez que vous utilisez bien la version 3.0 d'IDLE.

Dans la mesure où Python est un langage interprété, vous pouvez utiliser le shell comme une calculatrice pour tester les déclarations ou les calculs :

```
>>> 3+4+5
12
```

Le code que vous saisissez dans le shell interactif est comme un programme que vous exécutez ligne par ligne. Vous pouvez définir vos variables ou importer des modules :

```
>>> import math
>>> (1 + math.sqrt(5)) / 2
1.618033988749895
```

La commande `import math` rend toutes les fonctions mathématiques de Python disponibles pour votre programme (voir la section « Objets et modules » de ce chapitre, page 50, pour plus d'informations sur les modules). Pour créer vos variables, utilisez l'opérateur d'affectation (=) :

```
>>> import math
>>> radius = 20
```

```
>>> radius * 2 * math.pi
125.66370614359172
```

Si vous souhaitez effacer toutes les variables et réinitialiser l'environnement, sélectionnez Shell>Restart Shell. Vous pouvez aussi utiliser le shell interactif pour chercher des informations sur l'utilisation d'une déclaration particulière, d'un module ou de tout autre sujet relatif à Python avec la commande `help()` :

```
help("print")
```

Pour afficher une liste de tous les sujets disponibles, essayez :

```
help("topics")
help("keywords")
help("modules")
```

L'interpréteur de Python est pratique pour tester des déclarations ou des opérations simples, mais vous aurez souvent besoin d'exécuter vos scripts Python comme des applications autonomes. Pour cela, il vous faut un éditeur plus classique et non un shell. Pour créer un nouveau programme Python, sélectionnez File>New Window afin d'ouvrir une nouvelle fenêtre d'édition de code (figure 3-2).

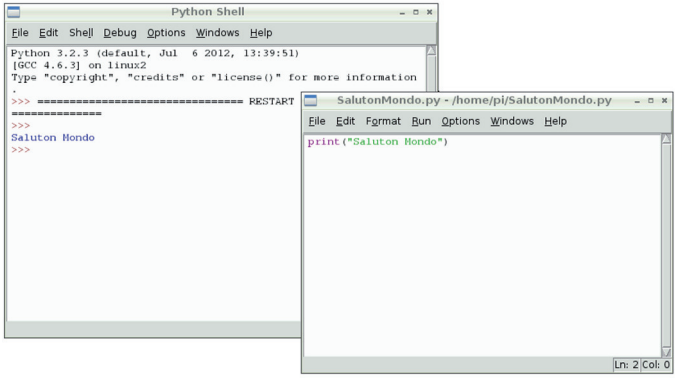


Figure 3-2. Le shell interactif d'IDLE (à gauche) et une fenêtre de l'éditeur (à droite)

Si vous tapez une ligne de code et sélectionnez le menu Run>Run Module, un avertissement s'affichera : *« Source Must Be Saved OK To Save? »* (soit « Le code source doit être sauvegardé, voulez-vous le sauvegarder ? »).

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Vous devez enregistrer votre script (par exemple, dans votre répertoire personnel `/home/pi` sous le nom `SalutonMondo.py`) pour qu'il puisse s'exécuter dans le shell.

Parfois, vous aurez peut-être envie de travailler avec un autre environnement qu'IDLE. Pour exécuter un script en ligne de commande, ouvrez LXTerminal et saisissez le code suivant :

```
python SalutonMondo.py
```

Vous venez d'apprendre à éditer et lancer un programme en Python. Comme vous pouvez le constater, les mécanismes de base nécessaires sont donc très simples. Intéressons-nous maintenant aux détails du langage.

Ligne de commande ou IDLE ?

Vous remarquerez rapidement qu'IDLE affiche les résultats très lentement si vous exécutez du code qui écrit dans le shell (avec `print()`, par exemple). Pour vous donner une idée de cette lenteur, faisons une petite expérience.

Lancez un nouveau LXTerminal tout en ayant IDLE ouvert en parallèle. Éditez le script décrit dans les pages suivantes avec IDLE, enregistrez-le sous le nom `CountEvens.py` et tapez la commande shell suivante dans le terminal :

```
python CountEvens.py
```

Mais avant de faire cela, exécutez le même script dans IDLE en passant par le menu Run Module, il aura besoin d'une bonne longueur d'avance… Vous constaterez rapidement qu'IDLE ralentit l'exécution des programmes, ce qui est d'autant plus visible avec les ressources relativement limitées du Pi. C'est pour cela que tous les exemples de ce livre sont prévus pour pouvoir tourner en ligne de commande, mais vous pouvez utiliser IDLE pour éditer le code si vous le désirez.

Un peu plus de Python

Si vous avez une expérience du monde Arduino, vous êtes sans doute habitué à écrire les programmes (appelés *sketches* en Arduino et *scripts* en Python) en utilisant une organisation de type *setup/loop.* `setup()` est une fonction qui s'exécute une fois seulement alors que `loop()` s'exécute dans une boucle infinie. Les exemples suivants montrent les codes

équivalents en Python. Sélectionnez le menu New Window du shell dans IDLE 3 et saisissez le code suivant :

```
# Setup : initialisation
n = 0

# Loop: début de la boucle infinie
while True:
    n = n + 1
    # Le signe % correspond à l'opération modulo
    if ((n % 2) == 0):
        # Affiche uniquement les nombres pairs
        print(n)
```

Sélectionnez le menu Run Module et nommez votre script (par exemple, `EvenIntergers.py`). Normalement, vous verrez tous les nombres entiers pairs s'imprimer à l'écran. Appuyez sur le raccourci clavier Ctrl + C pour interrompre le programme sinon il ne s'arrêtera jamais.



Dans l'exemple précédent, vous ne voyez probablement pas que chaque niveau d'indentation (le décalage vers la droite du début de la ligne) est constitué de quatre espaces et non d'une tabulation. D'ailleurs, IDLE convertit la tabulation de la touche Tab en une série d'espaces. En effet, contrairement à la plupart des langages classiques, l'indentation des lignes a un sens structurel en Python. Cela veut dire que le nombre d'espaces en début de ligne indique la structure du programme.

Ceci est un des points parfois difficiles à comprendre pour les débutants ou quand vous copiez et collez du code. Nous pensons que l'utilisation des espaces rend le langage Python plus facile à lire. Consultez le *Python Style Guidelines* afin de découvrir des astuces pour écrire du code clair et lisible. D'autres ressources en français sont mentionnées sur la page http://wiki.python.org/moin/FrenchLanguage.

Faites bien attention aux espaces. Python est un langage très structuré et les espaces déterminent la structure de vos scripts. Dans l'exemple suivant, chaque ligne indentée d'un niveau (décalée à droite) en dessous de la fonction `loop()` est considérée comme faisant partie de cette fonction. La fin d'une fonction ou d'une boucle est déterminée par l'endroit où l'indentation remonte d'un niveau (revient à gauche) ou à la fin du

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

fichier. C'est une approche différente des langages comme le C qui délimitent les blocs de code par des accolades, des parenthèses ou d'autres caractères.

Les fonctions vous permettent de regrouper des morceaux de code dans un bloc qui pourra être appelé à d'autres endroits de votre script. Pour réécrire l'exemple précédent avec des fonctions, faites comme ci-après (avant de l'exécuter, sauvegardez-le sous le nom `CountEvens.py`) :

```
# Déclare une variable globale
n = 0 ❶

# Fonction d'initialisation
def setup(): ❷
    global n
    n = 100

def loop(): ❸
    global n
    n = n + 1
    if ((n % 2) == 0):
        print(n)

# Main : le corps du programme ❹
setup()
while True:
    loop()
```

Cet exemple affichera chaque nombre pair à partir de 102. Regardons de plus près comment ce programme fonctionne :

❶ Cette ligne définit `n` comme une variable globale, qui pourra ensuite être utilisée dans n'importe quel bloc du script.

❷ Ici, la fonction `setup()` est définie (mais pas encore exécutée).

❸ De façon similaire, voici la définition de la fonction `loop()`.

❹ Dans le bloc principal du code, `setup()` est appelée une fois, puis la boucle infinie appelle `loop()` en permanence.

Le mot-clé `global` dans la première ligne de chaque fonction est important car il indique à l'interpréteur qu'il faut utiliser la variable globale `n` au lieu de créer une deuxième variable `n` (locale ou privée) utilisable uniquement dans cette fonction.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce tutoriel est trop court pour être une référence complète sur Python. Pour vraiment apprendre ce langage, nous vous suggérons les livres *Think Python* et *Python Pocket Reference*. La suite de ce chapitre vous donnera assez de notions pour que vous puissiez comprendre les exemples qui suivent, et vous présentera les caractéristiques et les modules de base disponibles. Le chapitre 4 présente le framework PyGame qui permet de créer des programmes multimédias avec le Raspberry Pi.

Objets et modules

Vous devrez maîtriser la syntaxe de base pour gérer les objets et modules qui figurent dans les exemples de ce livre. Python est un langage propre avec seulement 34 mots-clés réservés (tableau 3-1), qui constituent le cœur du langage et permettent de contrôler le flux d'activité de votre script.

La plupart des éléments qui ne sont pas des mots-clés peuvent être considérés comme des *objets*. Un objet est une combinaison de données et de comportements à laquelle on a donné un nom. Vous pouvez changer les données d'un objet, en extraire des informations et même manipuler d'autres objets avec.

Tableau 3-1. Les 34 mots-clés réservés de Python

Conditions	Boucles	Fonctions prédéfinies	Classes, modules, fonctions	Gestion des erreurs
`if`	`for`	`print`	`class`	`try`
`else`	`in`	`pass`	`def`	`except`
`elif`	`while`	`del`	`global`	`finally`
`not`	`break`		`lambda`	`raise`
`or`	`as`		`nonlocal`	`assert`
`and`	`continue`		`yield`	`with`
`is`			`import`	
`True`			`return`	
`False`			`from`	
`None`				

En Python, les chaînes de caractères, les listes, les fonctions, les modules et même les nombres sont des objets. Un objet Python peut être vu

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

comme une collection ou un ensemble d'attributs et de méthodes. Vous pouvez avoir accès à ces attributs et méthodes à partir d'une syntaxe simple, à savoir un point (`.`). Par exemple, tapez la commande suivante au prompt du shell interactif pour créer un objet de type chaîne de caractères et appeler la méthode qui lui demande de s'afficher en majuscules :

```
>>> myString = "quux"
>>> myString.capitalize()
'QUUX'
```

De même, vous pouvez employer `reverse()` pour inverser l'ordre d'une liste :

```
>>> myList = ['a', 'man', 'a', 'plan', 'a', 'canal']
>>> myList.reverse()
>>> print(myList)
['canal', 'a', 'plan', 'a', 'man', 'a']
```



Les chaînes de caractères et les listes sont supportées par des modules de base de la bibliothèque standard, elles sont donc toujours disponibles pour tous les programmes Python. Les modules `String` et `List` définissent de nombreuses fonctions qui traitent les chaînes de caractères et les listes, comme `capitalize()` et `reverse()`.

Certains modules de la bibliothèque standard ne sont pas disponibles directement. Il est nécessaire d'indiquer votre intention de les utiliser avec la commande `import`. Par exemple, le module `time` de la bibliothèque standard vous donne accès aux fonctions de gestion du temps et à l'horloge :

```
import time
```

Vous pouvez aussi rencontrer l'utilisation de `import as` qui importe un module mais lui donne un autre nom dans votre programme :

```
import time as myTime
```

Vous rencontrerez aussi `from import` qui charge des fonctions spécifiques d'un module :

```
from time import clock
```

Voici un petit exemple de script Python qui imprime l'heure une fois par seconde en utilisant les modules `time` et `datetime` de la bibliothèque standard :

```
from datetime import datetime
from time import sleep

while True:
    now = str(datetime.now())
    print(now)
    sleep(1)
```

La fonction `sleep(1)` suspend l'exécution du programme pendant une seconde. Pendant l'exécution de ce script, vous remarquerez que l'heure se décale légèrement après chaque affichage. Cela s'explique ainsi :

1. Le code ne prend pas en compte le temps nécessaire pour calculer l'heure (0,9 seconde serait un meilleur choix).
2. D'autres processus utilisent le processeur du Pi et peuvent s'exécuter brièvement à la place de votre programme. Il est important de se rappeler, quand on programme sur le Raspberry Pi, que le code ne s'exécute pas dans un environnement temps réel.

Si vous utilisez la fonction `sleep()`, vous verrez que sa précision est de l'ordre de 5 millisecondes sur le Pi.

Maintenant, modifions l'exemple pour ouvrir un fichier texte et y enregistrer des données. Lorsqu'on traite des fichiers texte, tout est considéré comme une chaîne de caractères. Utilisez la fonction `str()` pour convertir les nombres en chaînes de caractères. À l'inverse, la fonction `int()` transforme vos chaînes en nombres entiers.

```
from datetime import datetime
from time import sleep
import random

log = open("log.txt", "w")
for i in range(5):
    now = str(datetime.now())
    # Génère un nombre aléatoire entre 0 et 1 024
    data = random.randint(0, 1024)
    log.write(now + " " + str(data) + "\n")
    print(".")
```

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

```
    sleep(.9)
log.flush()
log.close()
```



Dans une véritable application d'enregistrement de données, il faudra vérifier que votre Pi est configuré avec la date et l'heure précises, comme nous l'avons vu au chapitre 1.

Voici un autre exemple (`ReadFile.py`) qui lit un nom de fichier dans l'argument de la ligne de commande. Il faut exécuter ce programme dans le shell avec la commande `python3 ReadFile.py nomDuFichierLog`. Le programme ouvre alors le fichier que vous avez indiqué, puis il lit chaque ligne comme une chaîne de caractères et l'imprime. Notez que `print()` fonctionne comme `println()` dans d'autres langages : il ajoute un retour à la ligne à chaque chaîne imprimée.

```
# Ouvre et lit le fichier dont le nom est en argument de
la ligne de commande

import sys

if (len(sys.argv) != 2):
    print("Usage: python ReadFile.py filename")
    sys.exit()

scriptname = sys.argv[0]
filename = sys.argv[1]

file = open(filename, "r")

lines = file.readlines()
file.close()

for line in lines:
    print(line, end = '')
```


Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Encore plus de modules

L'une des raisons de la popularité de Python est le grand nombre de modules créés par ses utilisateurs pour étoffer la bibliothèque standard. Le *Python Package Index* (PyPI, https://pypi.python.org/pypi) est la liste officielle des modules (souvent appelés packages). Certains des modules les plus populaires, qui sont particulièrement utiles sur le Raspberry Pi, figurent dans le tableau 3-2. Les exemples de ce livre vont utiliser ces modules, en particulier le module `GPIO` qui donne accès aux broches d'entrées et sorties (GPIO) du Pi.

Tableau 3-2. Quelques modules particulièrement intéressants pour les utilisateurs du Pi

Modules	Description	URL	Nom du module
`RPi.GPIO`	Accès aux broches GPIO	http://code.google.com/p/raspberry-gpio-python/	`python-rpi.gpio`
`Pygame`	Framework pour écrire des jeux	http://pygame.org	`pythonpygame`
`SimpleCV`	Interface simple pour la reconnaissance d'images	http://simplecv.org/	Pas de script
`Scipy`	Calcul scientifique	http://www.scipy.org/	`python-scipy`
`Numpy`	Les routines numériques à la base de Spicy	http://numpy.scipy.org/	`python-numpy`
`Flask`	Micro framework de développement web	http://flask.pocoo.org/	`python-flask`
`Requests`	« HTTP pour les Humains »	http://ocs.pythonrequests.org	`pythonrequests`
`PIL`	Traitement d'images	http://www.pythonware.com/products/pil/	`pythonimaging`
`wxPython`	Ajoute une interface graphique à Python	http://wxpython.org	`pythonwxgtk2.8`
`PySerial`	Accès au port série	http://pyserial.sourceorge.net/	`python-serial`
`pyUSB`	Accès aux puces d'interface FTDI-USB	http://bleyer.org/pyusb	Pas de script

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Pour utiliser l'un de ces modules, vous devrez télécharger le code correspondant, configurer le module et l'installer. Le module `serial` peut être installé ainsi :

```
sudo apt-get install python-serial
```

Si un module est fourni par son auteur en suivant les standards (avec l'outil `distutils` de Python), vous n'avez qu'à télécharger le module, le décompresser et saisir le code suivant :

```
python setup.py install
```

Vous pouvez aussi essayer l'installateur de modules `Pip`, outil qui facilite grandement l'installation des modules du PyPI.

Déceler et corriger les erreurs

Vous rencontrerez sans doute de petits soucis avec votre code, ce qui vous amènera à isoler des bogues et à les résoudre. Le mode interactif d'IDLE vous sera alors très utile : le menu Debug offre plusieurs outils pour vous aider à comprendre en détail comment votre code s'exécute. Il comporte également une option qui vous permet de visualiser toutes vos variables et de contrôler l'exécution de votre code ligne par ligne.

Les erreurs de syntaxe sont les plus faciles à corriger, généralement il s'agit juste d'une faute de frappe ou d'un aspect mal compris du langage. En revanche, les erreurs sémantiques, c'est-à-dire quand le programme est syntaxiquement valide mais ne fonctionne pas comme prévu, sont parfois plus difficiles à corriger. C'est pour cette raison que le débogueur peut s'avérer vraiment utile pour démêler une situation délicate. L'apprentissage en profondeur du débogage peut prendre des années. Voici quelques petites astuces pour vous donner un coup de pouce pour programmer en Python sur le Pi.

- Utilisez `print()` pour indiquer quand le programme arrive à un certain point.
- Utilisez `_print()+` pour montrer les valeurs des variables pendant l'exécution du programme.

- Vérifiez les espaces (indentation) pour vous assurer que les blocs de code sont définis comme vous le souhaitez.

- En corrigeant les erreurs de syntaxe, souvenez-vous que l'erreur que vous cherchez a pu être introduite bien avant qu'elle ne soit signalée par l'interpréteur.

- Revérifiez toutes vos variables (globales et locales), leur nom ainsi que leur type.

- Vérifiez que toutes les parenthèses vont par paires : il doit y avoir une parenthèse fermante pour chaque parenthèse ouvrante.

- Assurez-vous que l'ordre des opérations est correct lors des calculs. Insérez autant de parenthèses que nécessaire en cas de doute. Par exemple, `3 + 4 * 2` et `(3 + 4) * 2` produisent des résultats différents.

Lorsque vous aurez acquis plus d'expérience avec Python, intéressez-vous aux modules `code` et `logging`, ils contiennent d'autres outils de débogage du code.

Pour aller plus loin

Pour approfondir le sujet et être plus à l'aise avec Python, ces livres de référence pourraient vous intéresser :

- *Think Python* de Allen Downey, paru aux éditions O'Reilly Media en août 2012
  Une approche de la programmation claire et concise (qui utilise justement Python).

- *The Python Pocket Reference* de Mark Lutz, paru aux éditions O'Reilly Media en septembre 2009
  Parfois, il est plus pratique de feuilleter un livre que de se perdre dans une douzaine de messages postés sur le site Stack Overflow.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

- Stack Overflow
  Le site web Stack Overflow (http://stackoverflow.com) est une excellente source de savoir collectif. Il est particulièrement pratique si vous cherchez une solution spécifique ou des détails sur un message d'erreur. Il est fort probable que quelqu'un d'autre ait rencontré (et parfois résolu) le même problème.

- *Learn Python the Hard Way*
  Un livre génial disponible en ligne (http://learnpythonthehardway.org). Lisez au moins l'introduction « The Hard Way is Easier » (« L'approche difficile est la meilleure »).

- *Python for Kids* de Jason R. Briggs, paru aux éditions No Starch Press en novembre 2012
  Autre livre (écrit pour les jeunes) sur la programmation générale qui utilise Python.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

CHAPITRE 4

ANIMATION ET MULTIMÉDIA AVEC PYTHON

Pygame est un framework léger, conçu pour créer des jeux simples en Python. Vous pouvez aussi le considérer comme un outil de programmation multimédia : il permet de dessiner facilement des images sur l'écran, de jouer des sons et de gérer les événements de la souris et du clavier.

Pygame est une surcouche logicielle *(wrapper software)* englobant une autre bibliothèque appelée *Simple DirectMedia Layer* (SDL). Celle-ci gère l'ensemble des processus de bas niveau venant du clavier, la souris, les pilotes audio et vidéo. Pygame simplifie énormément SDL.

Ce chapitre explique les fonctions multimédias de base de Pygame sans être un vrai tutoriel sur la programmation des jeux. Des ressources supplémentaires pour apprendre la programmation des jeux sont fournies en fin de chapitre.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Bonjour Pygame !

Pygame est préinstallé sur le Raspberry Pi. Il existe une version de Pygame pour Python 3.0, mais au moment de la rédaction de ces lignes, la version livrée avec Raspbian marche uniquement avec la version 2.7. Vous pouvez soit basculer vers une version d'IDLE autre que la version 3.0 (double-cliquez alors sur l'icône d'IDLE au lieu de l'icône d'IDLE 3), soit exécuter votre script à partir de la ligne de commande afin de lancer l'interpréteur 2.7.

Les exemples suivants montrent les quelques étapes nécessaires pour écrire un programme utilisant Pygame. L'extrait de code ci-après permet de dessiner un cercle rouge dans une nouvelle fenêtre :

```
import pygame ❶

width = 640 ❷
height = 480
radius = 100
stroke = 1

pygame.init() ❸

window = pygame.display.set_mode((width, height)) ❹

window.fill(pygame.Color(255, 255, 255)) ❺

while True: ❻
    pygame.draw.circle(window, ❼
                       pygame.Color(255, 0, 0),
                       (width/2, height/2),
                       radius, stroke)
    pygame.display.update() ❽
```

❶ Importe le module `pygame`, ce qui met à notre disposition tous les objets et fonctions de Pygame.

❷ Crée des variables globales qui définissent les caractéristiques d'un cercle.

❸ Il faut toujours appeler en premier la fonction `init()` de Pygame, car cela permet d'effectuer les étapes d'initialisation nécessaires avant de pouvoir l'utiliser, et cela va appeler la fonction `init()` de tous les sous-modules de Pygame.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

❹ Crée une fenêtre graphique, qui est un objet de type Surface de Pygame, c'est-à-dire une zone dans laquelle vous pouvez dessiner.

❺ Remplit la fenêtre en blanc.

❻ Boucle infinie. Vous pouvez vous représenter chaque itération de cette boucle comme une image d'une animation sans fin.

❼ Trace un cercle rempli de rouge au milieu de la fenêtre.

❽ Toutes les commandes graphiques dans la boucle infinie vont créer une image individuelle qui est dessinée dans une zone de mémoire (souvent appelée *buffer*) hors de l'écran. Quand vous avez fini de dessiner une image dans le buffer, appelez `display.update()` pour transférer le contenu du buffer à l'écran.



Pour tous ces exemples, il faut taper Ctrl + C en ligne de commande pour stopper le script lorsque vous voulez l'arrêter. Si vous souhaitez ajouter le bouton de fermeture à la fenêtre graphique de manière à mettre fin à votre programme, ajoutez le code suivant à la fin de la boucle :

```
    # Dessine l'image ici...

    if pygame.QUIT in [e.type for e in pygame.event.get()]:
          break
```

Ces lignes de code gèrent l'événement déclenché lorsque le bouton de fermeture est cliqué. Pour plus d'informations sur les événements, consultez la section « Gestion des événements et des entrées » de ce chapitre, page 65.

Pygame se compose d'une collection de sous-modules et d'objets. La suite de ce chapitre vous explique les bases de leur utilisation.

Les surfaces de Pygame

Dans Pygame, un objet de type Surface correspond à une image rectangulaire. Les surfaces peuvent être combinées et superposées pour créer

chaque image d'un jeu ou d'une animation. Les pixels d'une surface sont représentés par une séquence RVB (Rouge, Vert, Bleu) de trois nombres de 8 bits (entre 0 et 255). Par exemple la liste (0, 255, 0) représente la couleur verte (le deuxième nombre, correspondant au vert, est au maximum alors que les deux autres sont au minimum). Pour indiquer la transparence, il faut ajouter un quatrième nombre (0, 255, 0, 127). Cette séquence représente une transparence de (255 × 100) / 127 = 50 (exprimée en pourcentage).

La fenêtre d'affichage principale est elle-même une surface normale sur laquelle toutes les autres surfaces sont affichées. Le module `pygame.display` contrôle et fournit l'information sur la fenêtre d'affichage. Utilisez la fonction `set_mode()` pour créer une nouvelle fenêtre d'affichage, et la fonction `update()` pour la mettre à jour.

Pour charger une image d'un fichier dans une surface, utilisez la fonction `load()` du module `pygame.image`. Dès que vous avez une fenêtre d'affichage, vous pouvez y ajouter des images en créant un nouvel objet de type Surface. Ensuite, utilisez la fonction `blit()` pour fusionner les images :

```
import pygame

pygame.init()
screen = pygame.display.set_mode((450, 450))
background = pygame.image.load("background.png") ❶
background.convert_alpha() ❷
screen.blit(background, (0, 0)) ❸
while True:
    pygame.display.update()
```

❶ Charge une image d'arrière-plan qui doit se trouver dans le même répertoire que le programme Python.

❷ La fonction `convert_alpha()` change le format d'une surface pour s'adapter à l'affichage en cours. Cette étape n'est pas obligatoire, mais elle vous permettra d'accélérer la fusion d'une image.

❸ Par défaut, la surface de l'écran est noire. La fonction `blit()` combine la surface contenant l'image avec un arrière-plan noir.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

L'exemple suivant mélange deux images (figure 4-1) :

```
import pygame

pygame.init()
screen = pygame.display.set_mode((450, 450))
background = pygame.image.load("background.png").
     convert_alpha()
theremin = pygame.image.load("theremin.png").
     convert_alpha()
screen.blit(background, (0, 0))
screen.blit(theremin, (135, 50))
while True:
    pygame.display.update()
```



Figure 4-1. Résultat des deux images combinées

Le module `pygame.transform` possède des fonctions qui permettent la rotation et le redimensionnement (changement d'échelle) des surfaces. Les pixels d'une surface sont accessibles individuellement par les fonctions du module `pygame.surfarray`.

À leur création, les surfaces sont toujours des images rectangulaires. Même si les couches de transparence dans l'exemple précédent ont été combinées pour leur donner une forme arbitraire, une surface reste un rectangle. Si vous souhaitez créer une surface avec des bords non rectangulaires (pour la détection de collision au pixel près, par exemple), vous pouvez créer un masque à partir d'une autre surface avec le module `pygame.mask`. Les masques sont traités de manière plus détaillée dans les livres mentionnés à la fin de ce chapitre.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Dessiner sur les surfaces

Nous avons déjà vu un exemple de la fonction Pygame qui permet de dessiner des cercles. Le module `pygame.draw` comporte d'autres fonctions pour dessiner des rectangles, des lignes, des arcs et des ellipses. Le module `pygame.gfxdraw` permet quant à lui de dessiner des formes variées. Il fournit quelques options supplémentaires, mais souvenez-vous qu'il en est encore au stade expérimental, et son API (les noms des fonctions et leur utilisation) pourrait changer à l'avenir.

Pour dessiner du texte sur l'écran, il faut tout d'abord créer un nouvel objet `font` grâce au module `pygame.font`. Utilisez-le pour charger une police et pour tracer le texte. La fonction `game.font.get_fonts()` affiche une liste de toutes les polices disponibles sur votre Raspberry Pi :

```
import pygame

pygame.init()
for fontname in pygame.font.get_fonts():
    print fontname
```

Comme vous pouvez le constater, peu de polices sont fournies avec la distribution Raspbian de base. L'exemple suivant utilise l'objet `SysFont` pour charger la police `freeserif` :

```
import pygame

pygame.init()
screen = pygame.display.set_mode((725, 92))
font = pygame.font.SysFont("freeserif", 72, bold = 1)
textSurface = fonte.render("1 Theremin Per Child!", 1,
                           pygame.Color(255, 255, 255))
screen.blit(textSurface, (10, 10))
while True:
    pygame.display.update()
```

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Si vous souhaitez tester d'autres polices, essayez les commandes suivantes dans votre terminal :

```
sudo apt-get install ttf-mscorefonts-installer
sudo apt-get install ttf-liberation
```

Gestion des événements et des entrées

Dans Pygame, les événements sont une représentation des actions de l'utilisateur, par exemple la pression d'une touche du clavier ou le déplacement du curseur de la souris. Ces événements sont captés sous forme d'objets de type Event et sont placés dans la *queue d'événements*, une file d'attente où ils sont stockés dans l'ordre d'arrivée jusqu'à ce qu'ils soient traités par le programme. Le module `pygame.event` fournit des fonctions pour gérer les événements enregistrés depuis votre dernière consultation dans la queue. Vous pouvez même créer vos propres types d'événements pour mettre en œuvre un système de messagerie.

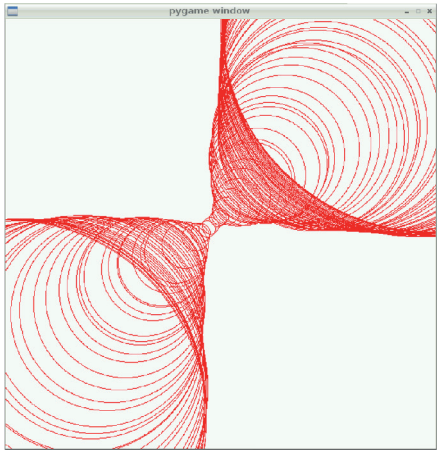


Figure 4-2. Résultat de l'exemple qui utilise les événements de Pygame

La figure 4-2 illustre un exemple d'utilisation de la queue d'événements, dérivé du programme précédent permettant de dessiner un rond rouge. Lors de chaque itération, un cercle est dessiné à l'endroit exact où se trouve la souris à un moment donné, et plus il est proche du bord de la fenêtre, plus le rayon s'élargit.

```
import pygame

from pygame.locals import * ❶

width, height = 640, 640
radius = 0
mouseX, mouseY = 0, 0 ❷

pygame.init()
window = pygame.display.set_mode((width, height))
window.fill(pygame.Color(255, 255, 255))

fps = pygame.time.Clock() ❸

while True: ❹
    for event in pygame.event.get(): ❺
           if event.type == MOUSEMOTION: ❻
               mouseX, mouseY = event.pos
           if event.type == MOUSEBUTTONDOWN: ❼
               window.fill(pygame.Color(255, 255, 255))
    radius = (abs(width/2 - mouseX)+abs(height/2 -
     mouseY))/2 + 1 ❽
    pygame.draw.circle(window, ❾
                       pygame.Color(255, 0, 0),
                       (mouseX, mouseY),
                       radius, 1)
    pygame.display.update()
    fps.tick(30) ❿
```

❶ Le module `pygame.locals` définit des valeurs constantes utiles comme `MOUSEMOTION`. Les importer de la sorte nous permet d'utiliser les constantes du module sans les préfixer en permanence par `pygame`.

❷ Les variables permettant de stocker les coordonnés de la souris.

❸ Cette fonction initialise l'objet `fps` (*Frames Per Second*, soit images par seconde en français), qui permet d'attendre entre chaque affichage pour obtenir un rafraîchissement régulier.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

❹ Boucle sans fin, chacune de ses itérations correspond à une nouvelle image.

❺ Boucle qui balaie la queue d'événements. À chaque itération de la boucle, la variable `event` contient le prochain événement de la queue.

❻ Si l'événement correspond à un déplacement du curseur de la souris, il faut mettre à jour la variable qui contient son nouvel emplacement.

❼ Si l'événement est un clic de souris, il faut effacer l'écran.

❽ Fait varier le rayon en fonction de la distance de la souris par rapport au centre de l'image.

❾ Dessine le cercle.

❿ Temporise pour que l'image soit rafraîchie 30 fois par seconde.

D'autres modules sont utiles pour gérer les événements et les actions de l'utilisateur.

- `pygame.time` : module de gestion du temps.
- `pygame.mouse` : module qui recueille les informations de la souris.
- `pygame.key` : module qui fournit les informations du clavier. Il contient aussi de nombreuses constantes qui représentent les touches.
- `pygame.joystick` : module d'interface avec un joystick.

Passer en mode plein écran

Pour que votre programme s'affiche en plein écran, utilisez l'option **`pygame.FULLSCREEN`** lorsque vous configurez le mode d'affichage. Une fois en mode plein écran, il faut s'assurer qu'il est possible de sortir du script car il sera impossible de l'arrêter avec Ctrl + C :

```
import pygame
import random
from time import sleep

continue = True
pygame.init()
screen=pygame.display.set_mode((0,0), pygame.FULLSCREEN)
while running:
    pygame.draw.circle(
        screen,
        pygame.Color(int(random.random()*255),
```

```
                    int(random.random()*255),
                    int(random.random()*255)),
          (int(random.random()*1500),
           int(random.random()*1500)),
          int(random.random()*500), 0)
      pygame.display.update()
      sleep(.1)
      for event in pygame.event.get():
          if event.type == pygame.KEYDOWN:
              running = False
pygame.quit()
```

Le script s'arrêtera dès qu'une touche du clavier sera pressée.

Les sprites

Dans le monde des jeux vidéo, la plupart des éléments graphiques déplaçables et contrôlables sont appelés des *sprites*. Le module `pygame.sprite` fournit toutes les fonctions de base pour dessiner les sprites à l'écran et gérer leurs collisions. Il est aussi possible de regrouper les sprites pour qu'ils puissent être contrôlés et mis à jour ensemble. La réalisation d'un jeu complet à base de sprites dépasse le cadre de ce livre, mais vous trouverez des ressources intéressantes à la fin de ce chapitre, dans la section « Pour aller plus loin » page 74.

Les sprites permettent aussi de créer à l'écran plusieurs éléments qui partagent du code. Python étant un langage orienté objet, la tâche est aisée. L'exemple suivant montre comment gérer et mettre à jour plusieurs sprites. Le résultat obtenu consiste en deux balles qui rebondissent sur les bords de l'écran. Vous pouvez ajouter des balles supplémentaires en créant de nouveaux sprites avec leurs coordonnées de départ, leurs directions et leurs vitesses respectives.

```
import pygame

class Ball(pygame.sprite.Sprite): ❶

    def __init__(self, x, y, xdir, ydir, speed): ❷
        pygame.sprite.Sprite.__init__(self)
```


Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

```
        self.image = pygame.Surface([20, 20])
        self.image.fill(pygame.Color(255, 255, 255))
        pygame.draw.circle(self.image,
                           pygame.Color(255,0,0),
                           (10,10), 10, 0)
        self.rect = self.image.get_rect()
        self.x, self.y = x, y ❸
        self.xdir, self.ydir = xdir, ydir
        self.speed = speed

    def update(self): ❹
        self.x = self.x + (self.xdir * self.speed)
        self.y = self.y + (self.ydir * self.speed)
        if (self.x < 10) | (self.x > 490):
            self.xdir = self.xdir * -1
        if (self.y < 10) | (self.y > 490):
            self.ydir = self.ydir * -1
        self.rect.center = (self.x, self.y)

pygame.init()
fps = pygame.time.Clock()
window = pygame.display.set_mode((500, 500))
ball = Balle(100, 250, 1, 1, 5) ❺
ball2 = Balle(400, 10, -1, -1, 8)

while True:
    ball.update() ❻
    ball2.update()
    window.fill(pygame.Color(255,255, 255))
    window.blit(ball.image, ball.rect) ❼
    window.blit(ball2.image, ball2.rect)
    pygame.display.update()
    fps.tick(30)
```

❶ Le mot-clé `class` crée un nouveau type d'objet, appelé ici `Ball`, qui se base sur l'objet sprite. Vous pouvez alors écrire vos propres fonctions pour afficher le sprite, et définir son comportement chaque fois qu'il est mis à jour.

❷ Cette fonction est appelée quand la fonction `Ball()` est appelée pour créer une nouvelle balle.

❸ Ces variables sont stockées dans chaque instance de `Ball`. Chaque balle dispose en effet de son propre objet qui contient ses propres variables, indépendamment des autres balles.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

❹ La fonction `update()` est appelée au moment de calculer une nouvelle image. La position de chaque balle change en fonction de leur direction et de leur vitesse, puis elle est testée pour voir si la balle s'approche du bord. Quand elle touche un bord, la direction sur cet axe particulier est inversée.

❺ Crée deux objets de type `Ball`, en indiquant leurs points de départ, leurs directions et leurs vitesses respectives.

❻ Met à jour la position des balles en fonction de leur position précédente, de leur direction et de leur vitesse.

❼ Dessine les balles à leur nouvel emplacement.

Jouer des sons

Avec Pygame, vous pouvez charger des fichiers son et les jouer avec le module `pygame.mixer`. Pour générer de la musique, il existe aussi le module `pygame.midi` qui envoie des événements MIDI à d'autres programmes en cours d'exécution sur le Pi (comme des synthétiseurs logiciels) ou vers des équipements MIDI branchés sur un port USB. L'exemple suivant lit un échantillon sonore au format WAV (disponible en téléchargement sur le site Internet Archive à l'adresse : http://archive.org/details/WilhelmScreamSample).

```
import pygame.mixer
from time import sleep

pygame.mixer.init(48000, -16, 1, 1024)

sound = pygame.mixer.Sound("WilhelmScream.wav")
channelA = pygame.mixer.Channel(1)
channelA.play(sound)
sleep(2.0)
```

Un fichier audio au format WAV est chargé et associé à un canal. Le mixeur joue chaque canal avec un processus individuel, ce qui permet de jouer plusieurs sons en même temps. À la fin du script, `sleep()` est nécessaire pour que le processus principal ne se termine pas avant la fin de la lecture du fichier.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

La lecture des fichiers son via le mixeur sera détaillée au chapitre 8 à travers le projet « Boîte à sons ». En ce qui concerne le protocole MIDI, Pygame fournit des fonctionnalités très intéressantes, permettant par exemple de créer un contrôleur MIDI personnalisé.

Au moment de l'écriture de ce livre, les logiciels de synthèse sonore fonctionnels sur Raspbian ne sont pas encore aboutis. De plus, la sortie analogique (disponible sur le connecteur audio 3,5 mm) n'est pas de bonne qualité, contrairement à la sortie HDMI (qui nécessite un adaptateur ou une télévision). Néanmoins, vous avez toujours la possibilité d'utiliser un équipement MIDI externe en passant par le port USB.

Pour lister les informations sur les équipements MIDI connectés à votre Raspberry Pi, utilisez le script suivant :

```
import pygame
import pygame.midi ❶

pygame.init()
pygame.midi.init() ❷
for id in range(pygame.midi.get_count()): ❸
    print pygame.midi.get_device_info(id) ❹
```

❶ Le module `midi` n'est pas importé automatiquement avec le module de base `pygame`, il faut donc l'ajouter manuellement.

❷ Le module `midi` doit aussi être initialisé séparément.

❸ La fonction `get_count()` retourne le nombre d'équipements MIDI reconnus par votre Pi. Ces équipements peuvent être des périphériques connectés par USB, des logiciels synthétiseurs ou encore des dispositifs virtuels MIDI.

❹ Affiche les informations relatives à l'équipement.

Si vous branchez un clavier MIDI externe, vous verrez quelque chose comme :

```
('ALSA', 'Midi Through Port-0', 0, 1, 0)
('ALSA', 'Midi Through Port-0', 1, 0, 0)
('ALSA', 'USB Uno MIDI Interface MIDI 1', 0, 1, 0)
('ALSA', 'USB Uno MIDI Interface MIDI 1', 1, 0, 0)
```

La première partie de chaque ligne indique que vous utilisez le pilote ALSA (*Advanced Linux Sound Architecture*, pilote qui gère le son sous Linux) et la deuxième décrit chaque port MIDI. Les trois derniers chiffres indiquent si le port est en entrée, en sortie et s'il est ouvert ou non. L'exemple précédent peut être considéré comme une liste de quatre ports numérotés de 0 à 3, dont les rôles sont définis ci-après :

- `'Midi Through Port-0' (0, 1, 0)`
  Le port 0 est une sortie qui communique avec les synthétiseurs logiciels qui tournent sur le Pi.

- `'Midi Through Port-0' (1, 0, 0)`
  Le port 1 est une entrée qui accepte les messages MIDI des contrôleurs logiciels que vous exécutez sur le Pi.

- `'USB Uno MIDI Interface MIDI 1' (0, 1, 0)`
  Le port 2 est une sortie vers une interface USB MIDI externe (de modèle UNO). Cette interface est connectée à un clavier maître MIDI.

- `'USB Uno MIDI Interface MIDI 1' (1, 0, 0)`
  Le port 3 est l'entrée de l'interface USB MIDI décrite au point précédent.

Grâce à ces interfaces, vous pouvez brancher un clavier MIDI sur un convertisseur USB et utiliser Pygame pour contrôler le Pi, comme dans l'exemple suivant :

```
import pygame
import pygame.midi
from time import sleep

instrument = 0 ❶
note = 74
volume = 127

pygame.init()
pygame.midi.init()

port = 2 ❷
midiOutput = pygame.midi.Output(port, 0)
midiOutput.set_instrument(instrument)
for note in range(0, 127):
    midiOutput.note_on(note, volume) ❸
```


Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

```
    sleep(.25)
    midiOutput.note_off(note, volume)
del midiOutput
pygame.midi.quit()
```

❶ Les valeurs MIDI sont comprises entre 0 et 127.

❷ Ouvre le port 2, la sortie associée au clavier contrôlé par l'interface USB MIDI.

❸ Envoie un signal pour jouer une note, attend, puis arrête la note.

Le Raspberry Pi a sans doute beaucoup de potentiel en tant que plate-forme de création musicale !

Lire une vidéo

Pygame peut aussi lire des vidéos grâce au module `pygame.movie`. Les fichiers vidéo doivent être encodés au format MPEG1. Si vous avez une vidéo enregistrée dans un autre format, essayez le programme `ffmpeg` qui convertit les différents formats audio et vidéo (il faudra d'abord exécuter `sudo apt-get install ffmpeg`). Pour lire le fichier vidéo, il suffit de créer un nouvel objet de type `Movie` et d'appeler la fonction `play()` :

```
import pygame
from time import sleep

pygame.init()
screen = pygame.display.set_mode((320,240))
movie = pygame.movie.Movie("foo.mpg")
movie.play()
while True:
    if not(movie.get_busy()):
       print("rewind")
       movie.rewind()
       movie.play()
    if pygame.QUIT in[
       e.type for e in pygame.event.get()
       ]:
       break
```

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Si la vidéo possède une piste audio, il faut fermer le mixeur audio de Pygame avant de lancer la lecture. Pour demander à ce que le mixeur s'arrête, il suffit d'inclure cette ligne avant de commencer la lecture :

```
pygame.mixer.quit()
```

Encore plus d'exemples

Le module **pygame.examples** fournit encore plus d'exemples de programmes. Vous trouverez le code source de ces exemples dans le répertoire **/usr/share/pyshared/pygame/examples**.

Pour aller plus loin

- La documentation officielle de Pygame
  Cette documentation (http://www.pygame.org/docs) n'est pas très complète sur certains sujets, mais pourra vous apporter un certain nombre d'informations supplémentaires.

- *Making Games with Python & Pygame* et *Invent Your Own Computer Games with Python* de Al Sweigart
  Deux livres sous licence Creative Commons. Les jeux développés dans ces livres sont fournis avec le Raspberry Pi.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

CHAPITRE 5

PROGRAMMER EN SCRATCH

Scratch a été développé par le groupe Lifelong Kindergarten au sein du laboratoire Media du MIT, dans le but d'enseigner la programmation aux enfants d'une nouvelle manière. Les programmes *(scripts)* sont construits en assemblant des blocs colorés, chacun réalisant une opération particulière. Ces blocs autonomes permettent de s'affranchir des problèmes de syntaxe de tous les langages classiques à base de texte (comme Python), qui créent souvent des obstacles pour les néophytes. Plus de problèmes de ponctuation ou d'indentation donc !

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Certains pensent que Scratch n'est pas un langage de programmation puissant car ils sont habitués à d'autres systèmes, mais l'objectif n'est pas de les remplacer. C'est un environnement convivial conçu pour créer des programmes rapidement avec un minimum d'efforts et de connaissances. Pour ce faire, un programmeur débutant peut suivre le déroulement du programme en observant les blocs de code qui s'allument au fur et à mesure de leur exécution et lorsque des blocs sont modifiés, l'effet est immédiat.

Comme vous le verrez, tous les programmes écrits en Scratch ont pour vocation de manipuler les sprites d'une scène *(stage)*. La communauté des utilisateurs de Scratch est assez importante et la possibilité de partager des sprites et du code avec ses membres est intégrée par défaut à la plate-forme.

Bonjour Scratch !

Pour montrer à quel point Scratch est facile à utiliser, nous allons commencer avec un programme très simple, intitulé « Le chat se promène sur l'écran ». Au démarrage de Scratch, une fenêtre s'affiche, divisée en plusieurs onglets. Leur fonction respective est décrite à la figure 5-1. Vous pouvez lancer Scratch en double-cliquant sur son icône sur le Bureau ou en cliquant sur le menu du Bureau situé en bas à gauche et en sélectionnant Éducation>Scratch.

Tout programme Scratch comporte des sprites qui interagissent sur une scène. Les sprites sont contrôlés par des scripts construits à partir de blocs disponibles dans l'onglet Palette de blocs. Lorsque vous créez un nouveau document, vous obtenez le sprite par défaut de notre exemple Scratch : le chat. Si vous n'en voulez pas, vous pouvez l'effacer (par un clic droit sur le sprite dans l'onglet Liste des sprites) et le remplacer en créant le vôtre. Vous pouvez aussi importer un sprite au hasard, parmi ceux créés par les autres utilisateurs de la communauté Scratch.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

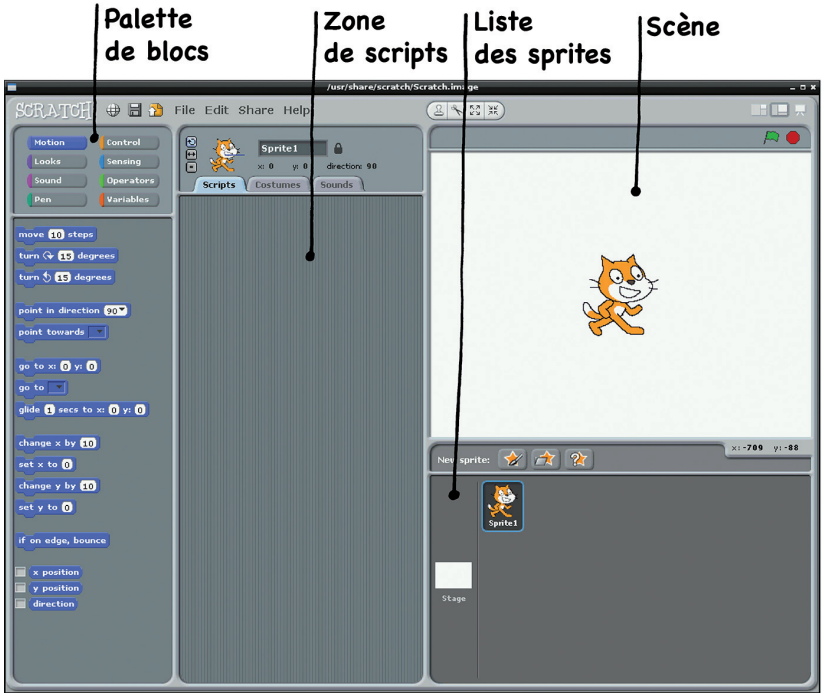


Figure 5-1. La fenêtre principale de l'environnement Scratch est divisée en différents onglets permettant d'accéder à la palette de blocs, à la zone de scripts et des costumes, à la liste des sprites et à la scène.

L'onglet Zone de scripts affiche le script qui contrôle le sprite sélectionné. Cliquez sur le chat dans la liste des sprites afin d'afficher le script de ce sprite, vierge pour le moment. Vous pouvez créer un script à n'importe quel endroit dans cet onglet. Si vous créez un autre sprite, vous pourrez afficher son script en cliquant dessus.

Pour construire un script, cliquez sur un bloc de l'onglet Palette de blocs puis, tout en maintenant le bouton de la souris enfoncé, déplacez-le vers l'onglet Zone de scripts et relâchez le bouton de la souris. Certains blocs peuvent contenir d'autres blocs, ce qui est indiqué par leur forme. Il existe trois types de blocs en Scratch, tels que présentés à la figure 5-2.

- Les blocs de contrôle
  Ce type de bloc est indiqué par un drapeau vert. Les blocs de contrôle sont nécessaires à l'écriture d'un script et figurent toujours au début de celui-ci, attendant qu'un événement se produise.

- Les blocs de commande
  Les blocs présentant une encoche et/ou une bosse à leur base peuvent s'emboîter à d'autres blocs. Ils s'exécutent successivement, en fonction de l'ordre d'empilement, à partir du bloc de contrôle et jusqu'à la fin du script.

- Les blocs de valeurs
  Les blocs de valeurs ont des extrémités pointues ou arrondies et s'emboîtent à l'intérieur des autres blocs, aux endroits où un nombre est attendu. Les blocs de valeurs peuvent être des variables, des valeurs saisies par l'utilisateur ou des informations fournies par l'ordinateur (par exemple, la position du curseur de la souris ou un nombre aléatoire, comme nous allons bientôt le voir).



Figure 5-2. Scratch utilise trois types de blocs : les blocs de contrôle (à gauche), les blocs de commande (au centre) et les blocs de valeurs (à droite).

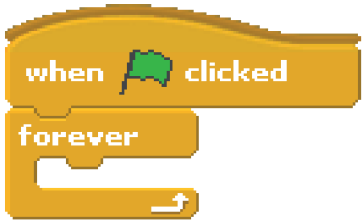
Pour commencer un nouveau script, sélectionnez le bloc de contrôle `when clicked` dans la palette de blocs et déposez-le dans la zone de scripts.



Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Faites de même pour le bloc de commande `forever`, qui exécute toutes les commandes qu'il contient, jusqu'à ce qu'il soit arrêté.



Entraînez-vous à organiser des blocs. Quand vous en déplacez un, vous déplacez également tous les sous-blocs qu'il contient. Pour isoler l'un des sous-blocs, vous devrez souvent dégrouper une pile entière. Vous pouvez aussi cliquer sur un bloc avec le bouton droit de la souris pour le dupliquer (avec tous ses sous-blocs) et obtenir de l'aide et des astuces.

Cliquez ensuite sur le bouton Motion (mouvement) en haut à gauche, sélectionnez un bloc `turn` (tourner) et placez-le dans le bloc `forever`.



Vous pouvez à présent cliquer sur le drapeau vert, en haut à droite de la scène, et démarrer l'exécution du script. Une des particularités de Scratch est que vous pouvez modifier le script pendant qu'il s'exécute et voir vos modifications immédiatement sur la scène. Il est ainsi possible de déboguer le programme à la volée.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Quand vous cliquez sur le drapeau vert, une instruction est envoyée à tous les scripts du projet leur indiquant de s'exécuter. De la même manière, si vous cliquez sur l'icône représentant un cercle rouge, en haut à droite de la scène, vous spécifiez d'arrêter les scripts.

La valeur de l'angle de rotation est indiquée sur le bloc `turn`. Vous pouvez modifier l'angle en saisissant manuellement une autre valeur, ou remplacer la valeur par un autre bloc présentant également des bords arrondis. Nous allons maintenant adapter le script pour que le sprite se déplace aléatoirement dans la scène.

Pour commencer, remplacez la valeur de l'angle par défaut par un bloc `pick random` (nombre aléatoire). Celui-ci se trouve dans la palette des opérateurs, que vous affichez en cliquant sur le bouton Operators. Après avoir placé ce nouveau bloc, modifiez ses valeurs de manière à ce qu'il génère des nombres au hasard dans un intervalle situé entre -10 et 10.



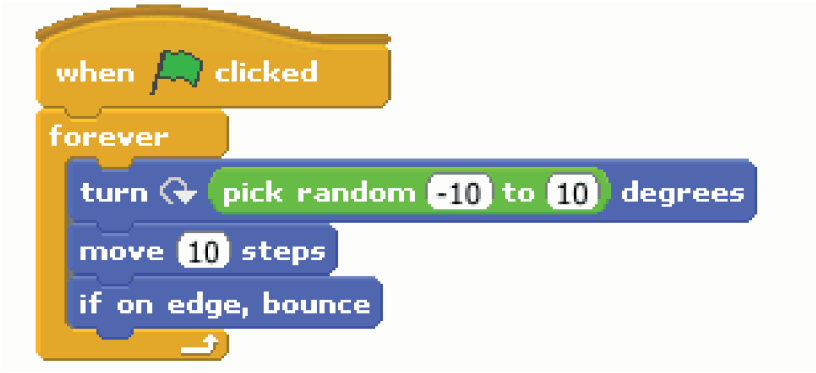
Ajoutez maintenant un bloc `move`, accessible en cliquant sur le bouton Motion. Ce bloc fait avancer le sprite dans sa direction actuelle (que nous venons de modifier avec des angles aléatoires). La direction est indiquée par un nombre et une ligne bleue dans la barre d'informations située au-dessus de la zone de scripts.

La distance de déplacement est indiquée par la valeur (exprimée en pixels) affichée dans le champ à bords ronds. Le chat va donc se déplacer par pas de 10 pixels. Dès que vous placerez le bloc dans le script, votre sprite commencera à se promener aléatoirement.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28



Enfin, pour que le sprite reste sur la scène, ajoutez un bloc `if on edge, bounce` (rebondir si au bord), disponible en cliquant sur le bouton Motion.



Et voilà, vous savez désormais assembler un script Scratch !

La scène

La scène correspond au cadre situé en haut à droite de l'interface de Scratch, dans lequel tous les sprites s'exécutent et interagissent. Comme pour les sprites, la scène peut comporter des scripts permettant de modifier son apparence ou son comportement. Pour remplacer la couleur de fond de la scène, cliquez sur l'onglet Backgrounds (Arrière-plans) de la zone de scripts.

Le système de coordonnées en Scratch est un peu différent des autres environnements multimédias comme Pygame. En effet, l'origine (0,0) se trouve au centre de la scène. La partie visible de la scène s'étend de (-240, 180) à (240, -180) comme illustré à la figure 5-3. Les sprites peuvent continuer en dehors de la scène si on le leur permet. Si on sélectionne un sprite, sa position est indiquée en haut de la zone de scripts, et la position du curseur s'affiche en bas de la scène.

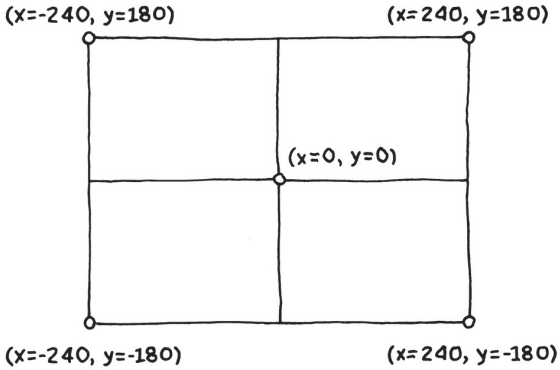


Figure 5-3. Dans le système de coordonnées Scratch, l'origine se trouve au centre de la scène.

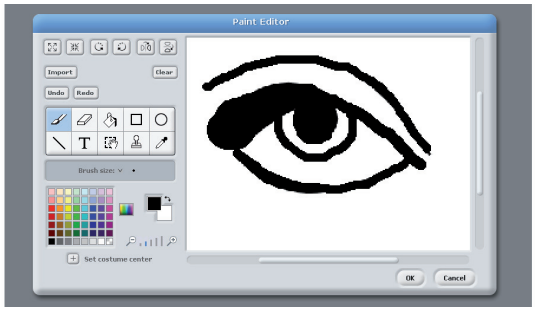
Deux autres choses à savoir sur les sprites

Vous serez certainement amené à utiliser les onglets Costumes et Sounds de la zone de scripts.

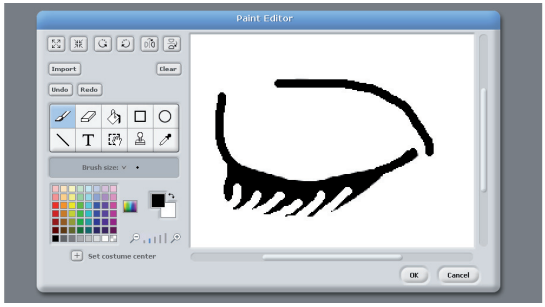
Les costumes correspondent aux images qui peuvent être affichées par les sprites. Par exemple, le costume par défaut du sprite est l'image du chat. Par ailleurs, les costumes peuvent également être utilisés pour créer des animations en boucle ou pour afficher les différents états d'un sprite (par exemple, une navette spatiale qui explose). Pour créer un nouveau sprite représentant un œil, nous allons changer son costume pour le faire cligner. Pour cela, cliquez sur l'icône Paint New Sprite (Dessiner un nouveau sprite), située en haut de l'onglet Liste des scripts. Dessinez un œil et cliquez sur OK.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

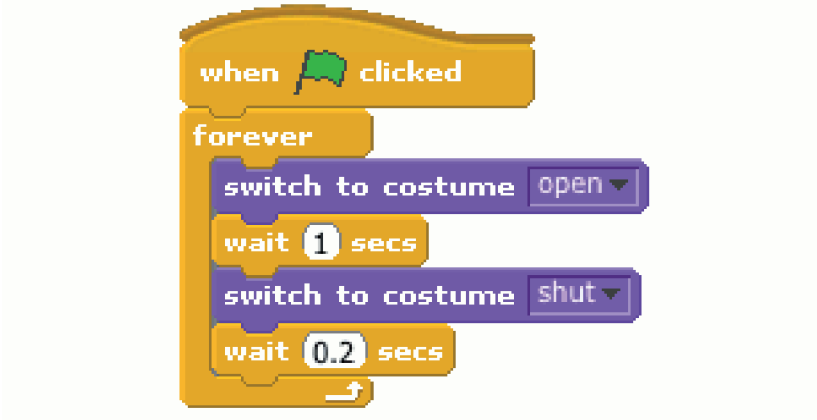
Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28



Cliquez sur l'onglet Costumes pour visualiser l'image que vous venez de dessiner. Renommez votre costume en `ouvert`. Cliquez sur le bouton Copy pour créer un nouveau costume. Cliquez ensuite sur Edit pour effacer la pupille et dessiner une paupière fermée. Validez en cliquant sur OK et renommez votre costume `fermé`.



Cliquez ensuite sur l'onglet Scripts et assemblez ces blocs pour créer un script qui cligne de l'œil toutes les secondes.



Vous pouvez aussi ajouter des effets sonores à un sprite grâce aux options de l'onglet Sounds. Cliquez sur Record pour activer l'enregistreur de sons intégré. Vous aurez besoin d'une carte son USB externe ou d'un microphone. Nommez l'enregistrement sonore et lisez-le grâce aux blocs `play sound` (jouer le son) ou `play sound until done` (jouer le son jusqu'à la fin).

Un exemple plus élaboré : le jeu Astral Trespassers

Dans le jeu Astral Trespassers, le joueur doit tirer sur des navettes spatiales extraterrestres en approche. Le jeu se termine quand un extra-terrestre touche le canon du joueur qui défend sa planète. Ce jeu simple montre comment tous les éléments de Scratch s'imbriquent les uns aux autres.

Pour commencer à créer le jeu, répétez les manipulations effectuées lors du premier exemple. Sélectionnez le menu File>New (Fichier>Nouveau), puis sauvegardez le nouveau jeu à l'aide du menu File>Save As (Fichier>Enregistrer sous). Vous devez ensuite créer des sprites. Pour cela, cliquez droit (ou utilisez Ctrl + clic) sur le chat dans la liste des sprites et sélectionnez Delete (Effacer) dans le menu contextuel. Vous devez maintenant créer des nouvelles images de sprites en les dessinant vous-même ou en les important à partir d'un fichier image. Nous avons dessiné sur papier les cinq images de la figure 5-4, puis nous les avons scannées au format PNG.

Vous verrez ces sprites apparaître sur la scène au fur et à mesure de leur création et il est fort probable qu'ils ne s'afficheront pas à la taille souhaitée. Dans ce cas, sélectionnez les outils Grow Sprite (Agrandir le sprite) ou Shrink Sprite (Rétrécir le sprite), situés au-dessus de la scène, et cliquez sur le sprite pour augmenter ou réduire sa taille. N'oubliez pas de renommer vos sprites avec les noms indiqués à la figure 5-4.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

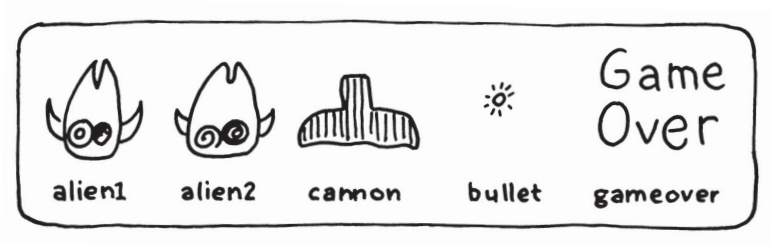


Figure 5-4. Les cinq sprites du jeu Astral Trespassers

Les aliens et le canon présentent tous deux états : normal et explosé. Nous pouvons les gérer par la création d'un costume pour chaque état, comme vous pouvez le voir sur la figure 5-5.

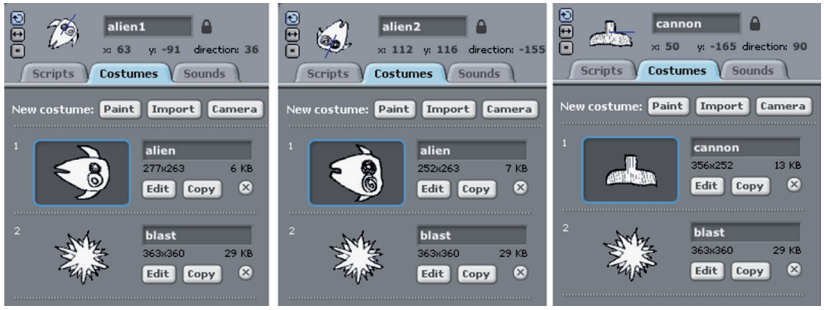


Figure 5-5. Il faut créer deux costumes pour chaque sprite, un pour son état normal et un autre pour son état explosé. Nommez les costumes comme indiqué.

Chaque sprite aura son propre script qui déterminera son comportement dans le jeu. Pour plus de clarté, chaque action sera définie par des petits scripts indépendants qui s'exécuteront simultanément.

L'annexe B présente les scripts complets pour chaque sprite.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Commençons par le premier alien. Sélectionnez le sprite et glissez les blocs suivants vers la zone de scripts.



Ce bloc va placer le sprite à sa position de départ, en s'assurant qu'il est visible et dans son état non explosé.

Ensuite, glissez et assemblez ces blocs quelque part dans la zone de scripts.

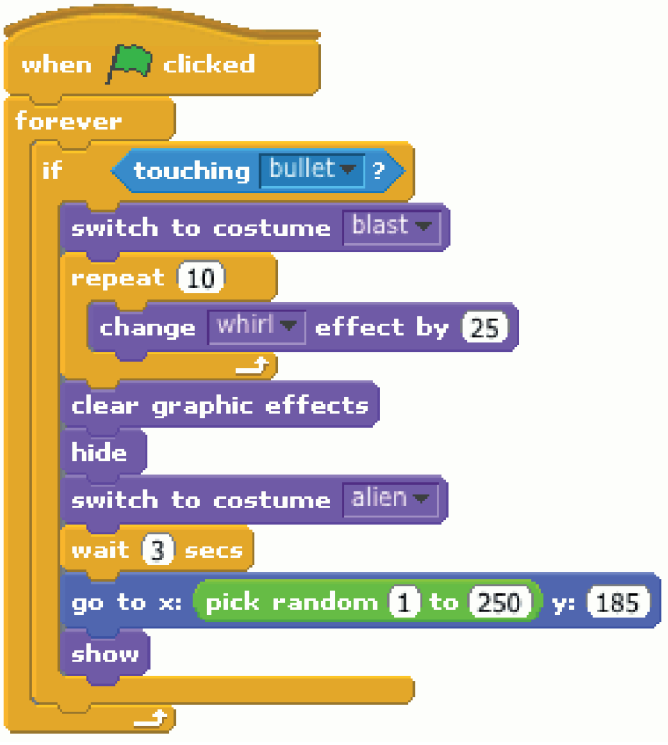


Cliquez sur le drapeau vert, vous constatez alors que ce script ressemble à notre premier exemple du chat qui se promène sur l'écran. Comme il y a ici trois sprites en jeu, vous aurez besoin de gérer ce qui arrive quand les sprites entrent en collision. Si deux aliens se touchent, une façon d'éviter qu'ils se superposent est de les faire tourner de 90 degrés lors d'un impact.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28



Si un boulet touche un alien, nous voulons qu'il explose.



L'explosion est créée en passant au costume `explosion`, puis en faisant tourner le sprite pendant un instant. Le sprite est ensuite caché afin de le sortir de la scène. Après trois secondes, l'alien reviendra à un endroit aléatoire en haut de la scène.

Pas besoin de plus pour gérer l'alien. Si plusieurs sprites effectuent les mêmes actions (ou sont similaires), vous pouvez copier du code entre sprites en cliquant droit sur un sprite et en sélectionnant Duplicate (dupliquer). Glissez les blocs clonés vers le deuxième alien et procédez de la même manière pour chaque bloc. Sélectionnez maintenant le deuxième alien et changez son point de départ ainsi que son test de collision.



Pour le sprite du canon, une fois celui-ci sélectionné, déplacez-le à l'endroit souhaité, dans un état visible et non explosé.



Le canon peut uniquement bouger vers la droite et vers la gauche. Il est contrôlé par les flèches du clavier. Le script permettant de gérer les touches du clavier est le suivant.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

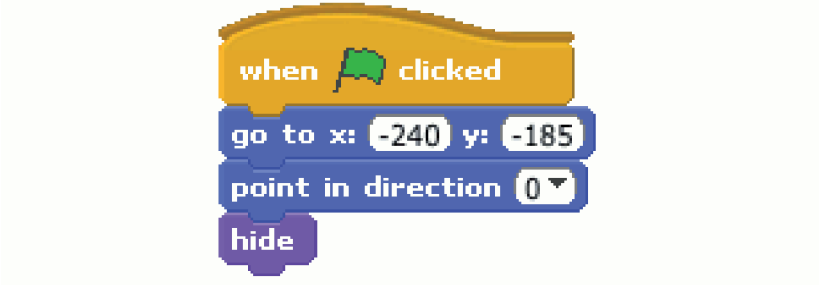
Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28



Il faut à présent tester si le canon est touché par un alien grâce au bloc `touching` (toucher) de la palette des capteurs. S'il est touché, il faut faire exploser le canon d'une manière similaire à l'explosion des aliens. Après avoir explosé, le canon enverra un message à tous les autres sprites pour les informer que le jeu est terminé. Ce message sera intercepté par le sprite `Game Over` qui arrêtera tous les scripts.

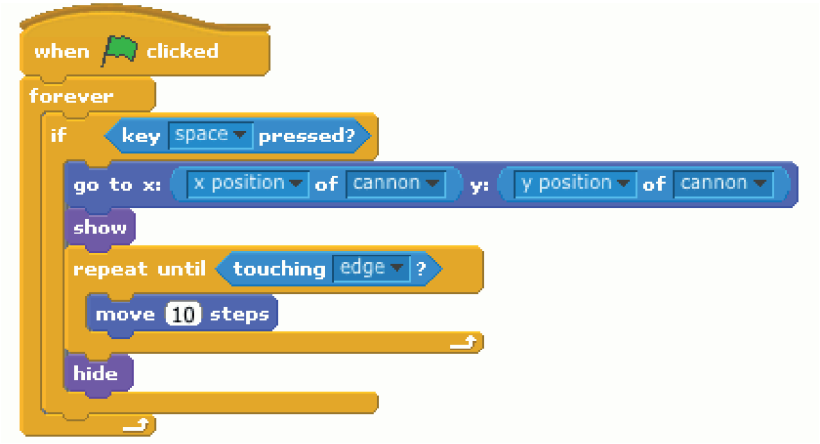


Intéressons-nous maintenant à la gestion du boulet. Les blocs suivants placent le sprite `boulet` à son point de départ. Il reste caché jusqu'à ce que la barre d'espace soit pressée.



Chaque fois que la barre d'espace est pressée, le boulet devra être positionné à l'endroit où se trouve le canon. Il y deviendra visible et se déplacera vers le haut de l'écran jusqu'à toucher son bord.

Les blocs suivants permettent de gérer la barre d'espace.



Le dernier sprite est le message « Game Over ». Il doit rester caché jusqu'à réception du message « GameOver ». À ce moment-là, il apparaît sur l'écran et tous les scripts s'arrêtent.

C'est la fin du jeu. Si vous souhaitez visualiser l'ensemble des blocs, reportez-vous à l'annexe B.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Scratch et le monde réel

Si vous fouillez dans la palette des capteurs, vous remarquerez deux blocs intéressants : `sensor` (capteur) et `sensor value` (valeur de capteur). Ces blocs permettent d'interagir avec une interface externe appelée la PicoBoard (figure 5-6). Il s'agit d'un circuit imprimé équipé d'un micro-contrôleur qui lit des capteurs et envoie les valeurs mesurées à Scratch par une connexion USB.

La PicoBoard est pourvue d'un bouton, d'un potentiomètre rectiligne, d'un capteur de son et de quatre connecteurs jacks pour lire des valeurs analogiques. Des pinces crocodiles spéciales peuvent se connecter sur plusieurs sources pour lire des échantillons.

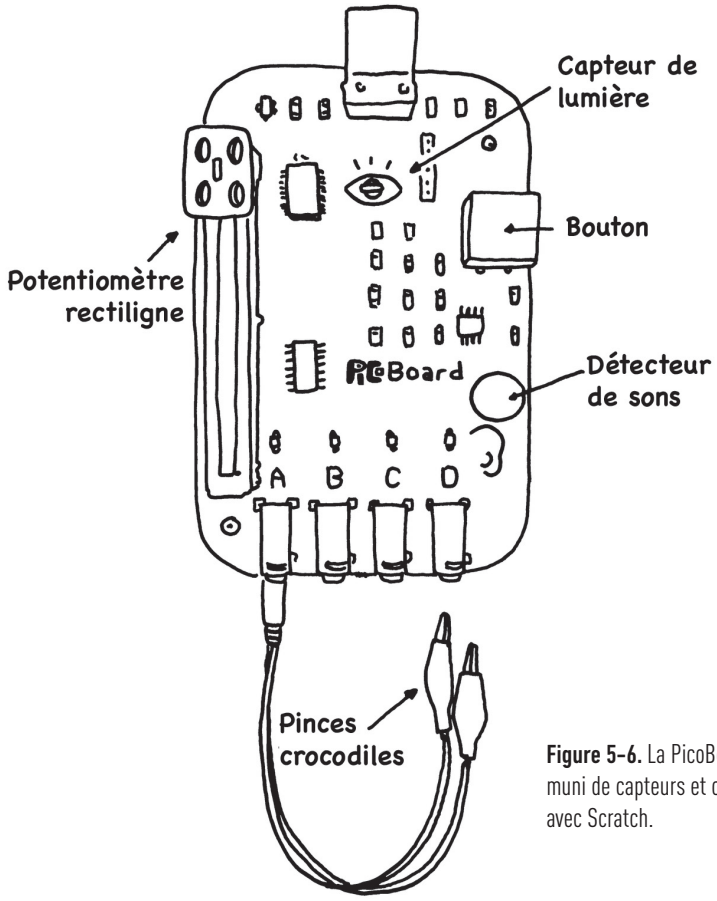


Figure 5-6. La PicoBoard est un accessoire muni de capteurs et conçu pour fonctionner avec Scratch.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

La PicoBoard envoie les valeurs mesurées à Scratch en utilisant son propre protocole PicoBoard. Le projet S4A *(Scratch for Arduino)* permet d'utiliser le même protocole avec une carte Arduino, donc vous pouvez aussi vous connecter au monde réel de cette manière. S4A a besoin d'une version spécifique de Scratch, disponible sur le site web de leur projet (http://www.seaside.citilab.eu/scratch/arduino).

Partagez vos programmes !

L'un des aspects intéressants de Scratch est que vous pouvez entrer en contact avec les membres de sa communauté directement depuis le programme. En plus de l'outil Random Sprite (Obtenir un sprite aléatoire), qui récupère des sprites réalisés par d'autres utilisateurs au hasard, Scratch fournit une fonction de partage qui vous permet d'envoyer vos programmes sur la page du projet Scratch au MIT (http://www.scratch.mit.eu/). À l'écriture de ce livre, 1,2 million d'utilisateurs Scratch ont mis à disposition plus de 2,8 millions de programmes.

Le grand nombre de programmes partagés s'explique par le fait que c'est très facile à faire. Après l'ouverture de votre compte sur la page du projet Scratch au MIT, il suffit de sélectionner Share This Project Online (Partager ce projet en ligne) dans le menu Share (Partager). Vous devrez répondre à quelques questions (figure 5-7), puis votre projet sera téléchargé sur le site. Les projets soumis ne doivent pas dépasser 10 Mo, vous devrez donc compresser au préalable vos images et vos sons (voir les options du menu Edit (Éditer)). Le site du projet Scratch est une ressource incontournable pour qui veut découvrir toutes les possibilités de cet environnement.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

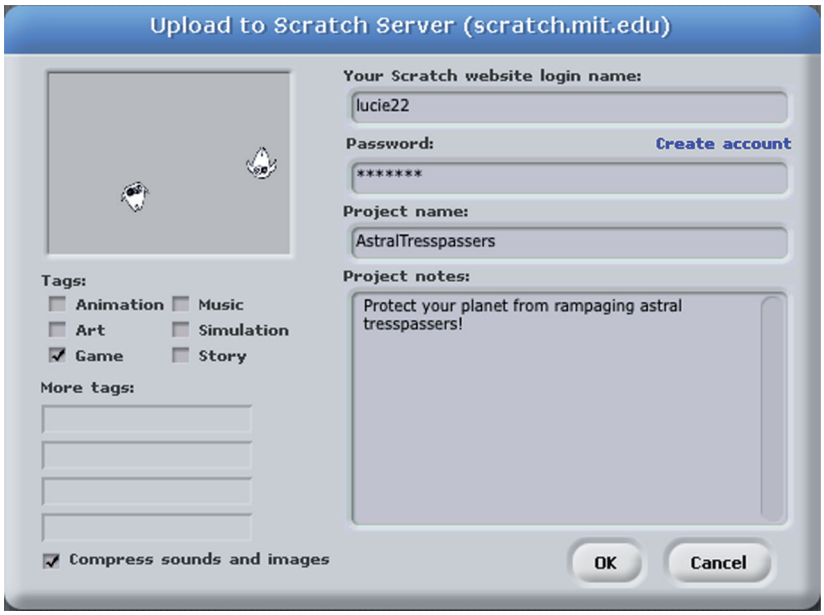


Figure 5-7. Quand vous aurez réalisé un projet intéressant, partagez-le sur le site officiel de Scratch grâce à l'outil de partage intégré. Malheureusement, le site web utilise Flash donc vous ne pourrez pas y accéder directement à partir du Raspberry Pi.

Pour aller plus loin

- La page principale du wiki de Scratch
  Ce site (http://www.wiki.scratch.mit.edu/wiki/Scratch_Wiki) contient les documentations de référence de Scratch.

- La page du projet Scratch du MIT
  La page de la communauté Scratch (http://www.scratch.mit.eu) contient des millions de projets et de membres enregistrés.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

CHAPITRE 6

RASPBERRY PI ET ARDUINO

Comme vous le verrez plus loin, les broches GPIO du Raspberry Pi peuvent être connectées à des capteurs (par exemple, des boutons) et à des actionneurs (comme des LED ou des moteurs). Si vous avez déjà utilisé la plate-forme de développement du microcontrôleur Arduino, vous pourrez appliquer vos connaissances à votre Raspberry Pi.

À l'annonce de la sortie du Raspberry Pi, de nombreuses personnes se sont demandé si les jours de l'Arduino étaient comptés. En effet, pour à peu près le même prix, vous avez un processeur plus puissant et des interfaces plus performantes. De plus, le son, la vidéo, le stockage de masse et le réseau sont déjà gérés. Pourquoi alors continuer à utiliser Arduino si vous possédez un Raspberry Pi ? Il s'avère que les deux plates-formes sont en fait complémentaires et le Pi est même un excellent hôte pour l'Arduino, que vous pourrez utiliser pour plusieurs raisons.

- Beaucoup de codes, d'exemples et de bibliothèques sont disponibles pour l'Arduino.

- Le microcontrôleur de l'Arduino possède des entrées analogiques pour mesurer des tensions. Le Raspberry Pi nécessite un ou des circuits externes pour remplir cette fonction.

- Vous voulez ajouter plus de puissance processeur à l'un de vos projets Arduino. Par exemple, vous avez un contrôleur MIDI connecté à un synthétiseur et vous souhaitez à présent synthétiser le son directement avec le Pi.

- Vous avez affaire à des niveaux logiques 5 V : le Pi travaille sous 3,3 V et ses broches ne tolèrent pas 5 V. De plus, il est parfois impossible d'ajouter un adaptateur de tension.

- Vous réalisez un prototype qui sort de votre domaine de compétences et cela risque d'abîmer les circuits. On a pu voir des élèves tenter de connecter des moteurs directement sur une broche de leur Arduino (ne le faites pas !). Lorsque le microcontrôleur d'un Arduino est grillé, on peut le sortir de son support et le remplacer par un neuf (pour moins de 10 €). Ce n'est pas du tout possible avec le Pi.

- Quand vous rencontrez un problème qui nécessite un contrôle précis en temps réel, à la microseconde près (par exemple, pour un contrôleur d'imprimante 3D). Comme nous l'avons vu au chapitre 3, Raspbian n'est pas un système d'exploitation temps réel et ses programmes ne peuvent être synchronisés au cycle d'horloge près comme avec un microcontrôleur.

Les exemples de cette section supposent que vous savez utiliser la carte Arduino et son IDE (environnement de développement intégré). Si vous n'avez pas encore une bonne compréhension des bases, nous vous conseillons l'ouvrage *Getting Started with Arduino* de Massimo Banzi. Les tutoriels disponibles sur le site officiel d'Arduino (http://arduino.cc/en/Tutorial/HomePage) sont aussi de bonnes références et vous permettront d'utiliser du code fonctionnel (copiez-le, puis adaptez-le à vos projets).

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

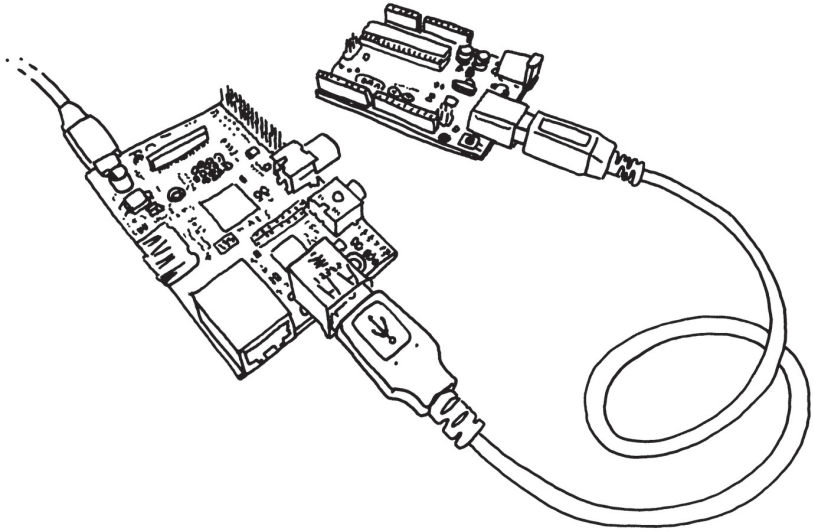


Figure 6-1. L'Arduino et le Raspberry Pi s'entendent très bien.

Installer Arduino sur Raspbian

Pour programmer une carte de développement Arduino, il faut la brancher à un ordinateur avec un câble USB, puis compiler et flasher un programme sur la carte avec l'IDE Arduino. Vous pouvez réaliser cela avec n'importe quel ordinateur, ce qui tombe bien étant donné que le Raspberry Pi en est un !

La compilation sera un peu plus lente sur le Pi que sur des ordinateurs portables ou de bureau récents. Ce n'est pas très grave car une fois la première compilation effectuée, les suivantes seront plus rapides : l'IDE Arduino ne compilera que le code qui a changé depuis la dernière compilation.

Pour installer l'IDE Arduino sur le Raspberry Pi, saisissez les instructions suivantes :

```
sudo apt-get update ❶
sudo apt-get install arduino ❷
```

❶ Vérifie que la liste des paquets de Raspbian est à jour.

❷ Télécharge le paquet Arduino.

Ceci installera automatiquement le système Java et les nombreux modules logiciels nécessaires. L'environnement Arduino apparaît ensuite dans la section Électronique du menu des applications, mais ne le lancez pas pour l'instant.

Vous pouvez simplement brancher l'Arduino dans un port USB libre du Pi. Si aucun port n'est disponible, utilisez un hub USB. Le câble USB devrait vous fournir assez de puissance pour faire fonctionner l'Arduino, mais utilisez si possible une alimentation externe pour une meilleure fiabilité.



Souvenez-vous qu'il faut brancher le câble USB de la platine Arduino après le démarrage du Raspberry Pi. Si l'Arduino est branché lorsque le Pi boote, cela peut bloquer ce dernier pendant qu'il essaie d'identifier tous les périphériques connectés sur le bus USB.

Chaque fois que vous lancez l'IDE Arduino, ce dernier balaie tous les périphériques USB et compile une liste, accessible par le menu Tools>Serial Port (Outils>Port série). Pour autoriser l'accès au port série, il faut que les permissions nécessaires soient attribuées à l'utilisateur `pi`. Pour cela, vous pouvez ajouter l'utilisateur `pi` aux groupes `tty` et `dialout`. Ce paramétrage doit être effectué avant de lancer l'IDE Arduino, en saisissant les lignes de commande suivantes dans un terminal :

```
sudo usermod❶ -a -G❷ tty pi
sudo usermod -a -G dialout pi
```

❶ `usermod` est un programme qui permet de gérer les utilisateurs et leurs permissions.

❷ L'argument `-a -G` ajoute (`-a`) l'utilisateur (`pi`) dans le groupe (`-G`) spécifié. On lance la commande deux fois, une fois pour le groupe `tty`, puis pour le groupe `dialout`.

Vous pouvez maintenant lancer l'IDE. Sélectionnez tout d'abord le port série via le menu Tools>Serial Port (probablement `/dev/ttyACM0`), puis rendez-vous dans le menu Tools>Board (Outils>Carte) pour sélectionner le type de votre carte Arduino (par exemple, `Uno`). Allez ensuite dans le menu File>Examples>01.Basics>Blink (Fichier>Exemples>01.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Basics>Blink) pour charger un *sketch* (un code source dans le jargon Arduino) d'exemple. Cliquez à présent sur le bouton Upload (Charger) situé dans la barre d'outils ou utilisez le menu File>Upload (Fichier>Charger) pour programmer le sketch dans la carte. Dès qu'il est chargé, la LED de la platine Arduino devrait commencer à clignoter.

Identification du port série

Si pour une raison quelconque le port `/dev/ttyACM0` ne fonctionne pas, il devient alors nécessaire de jouer au détective. Pour trouver le port série USB sur lequel est connectée la carte Arduino sans passer par le menu Tools>Serial Port, saisissez la commande suivante dans un terminal, sans que l'Arduino soit branché :

```
ls /dev/tty*
```

Rebranchez l'Arduino, puis exécutez à nouveau la même commande pour voir ce qui a changé.

Sur mon Raspberry Pi, `/dev/ttyAMA0` s'affichait au début (c'est le port série intégré qu'on retrouve sur les broches GPIO). Après le branchement de l'Arduino, `/dev/ttyACM0` est apparu dans la liste des ports.

Améliorer le confort d'utilisation

En utilisant l'IDE, vous avez peut-être remarqué que la qualité de la police par défaut de l'éditeur d'Arduino est loin d'être idéale. Vous pouvez améliorer son apparence en téléchargeant la police open source Inconsolata. Pour l'installer, saisissez les instructions suivantes :

```
sudo apt-get install ttf-inconsolata
```

Éditez ensuite le fichier des préférences Arduino :

```
nano ~/.arduino/preferences.txt
```

Et enfin, modifiez les lignes suivantes comme suit :

```
editor.font=Inconsolata,medium,14
editor.antialias=true
```

Au redémarrage de l'IDE, l'éditeur utilisera la nouvelle police.

Communiquer via le port série

Pour communiquer entre le Raspberry Pi et l'Arduino via une connexion série, vous devez utiliser la bibliothèque Serial du côté de l'Arduino (elle est incluse par défaut). En ce qui concerne le Raspberry Pi, il faut le module Python `pySerial` que vous pouvez installer avec cette commande :

```
sudo apt-get install python-serial python3-serial
```

Nous allons maintenant créer une paire de programmes qui communiquent entre eux : le premier envoie des données sur le port série et l'autre les envoie sur le Pi qui les récupère et les affiche. Ouvrez tout d'abord l'IDE Arduino, saisissez le code suivant et chargez-le dans la platine :

```
void setup() {
  Serial.begin(9600);
}

void loop() {
  for (byte n = 0; n < 255; n++) {
    Serial.write(n);
    delay(50);
  }
}
```

Ce programme compte de 0 à 255 et envoie chaque nombre sur le port série.



Souvenez-vous qu'avec Arduino, **`Serial.write()`** envoie le nombre tel quel, dans un octet. Si vous voulez envoyer le nombre 123 sous la forme d'une chaîne de trois caractères (`123`) au lieu d'un octet, utilisez à la place la commande **`Serial.print()`**.

Ensuite, lorsque que vous avez déterminé sur quel port série USB l'Arduino est connecté, vous pouvez lire les données entrant dans le Pi avec un script Python. Si le port série est autre que `/dev/ttyACM0`, modifiez

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

la valeur de la variable `port`. Sauvegardez alors le script sous le nom `SerialEcho.py` et exécutez-le avec `python SerialEcho.py` :

```
import serial

port = "/dev/ttyACM0"
serialFromArduino = serial.Serial(port,9600) ❶
serialFromArduino.flushInput() ❷
while True:
    if (serialFromArduino.inWaiting() > 0):
        input = serialFromArduino.read(1) ❸
        print(ord(input)) ❹
```

❶ Ouverture du port série connecté à l'Arduino.

❷ Effacement du tampon *(buffer)* d'entrée.

❸ Lecture d'un octet du port série.

❹ Conversion de l'octet entrant en un nombre décimal avec `ord()`.



Vous ne pourrez pas reprogrammer la carte Arduino si Python utilise le port série, donc n'oubliez pas de tuer le programme Python avec Ctrl + C avant de charger un nouveau sketch. Vous pourrez reprogrammer un Arduino Leonardo ou un Arduino Micro sans tuer le programme Python, mais la connexion avec le script Python sera alors coupée et il sera de toute façon nécessaire de redémarrer le script Python.

Dans l'exemple précédent, l'Arduino envoie un octet au script Python, qui l'interprète comme un caractère d'une chaîne de caractères. La variable `input` contiendra cet octet, mais considérera ce nombre comme un index dans la table ASCII (http://en.wikipedia.org/wiki/ASCII). Pour mieux comprendre, remplacez la dernière ligne du script Python par celle-ci :

```
print(str(ord(input)) + " = the ASCII character " + input
      + ".")
```

Ce code affiche le nombre ainsi que sa signification dans le code ASCII.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Indiquer le nom du port série en argument

Si vous souhaitez configurer le nom du port avec un argument de la ligne de commande, utilisez le module **sys** qui permet de récupérer le premier argument :

```
import serial, sys

if (len(sys.argv) != 2):
    print("Usage: python ReadSerial.py port")
    sys.exit()
port = sys.argv[1]
```

Vous pouvez ensuite lancer le programme comme suit :

```
python SerialEcho.py /dev/ttyACM0
```

Ce premier exemple permet d'envoyer uniquement un seul octet à la fois. Cela peut être suffisant si vous souhaitez envoyer une série de codes d'événements à partir de l'Arduino. Par exemple, si le bouton gauche de la souris est pressé, envoyer un 1 ; si c'est le bouton droit, envoyer un 2. Cette technique ne fonctionne que s'il y a moins de 255 événements uniques. Cependant, la plupart du temps, vous souhaiterez envoyer des nombres plus grands ou des chaînes de caractères. Si vous lisez les capteurs analogiques avec l'Arduino, vous enverrez par exemple des nombres entre 0 et 1 023.

Dans de nombreux langages, le décodage de nombres de taille arbitraire qui arrivent octet par octet est parfois une tâche plus compliquée qu'on ne peut l'imaginer. La manière dont Python et `pySerial` gèrent les chaînes rend cette tâche très simple. Pour vous donner un exemple, mettez à jour votre Arduino avec le code suivant qui compte de 0 à 1 024.

```
void setup() {
  Serial.begin(9600);
}
void loop() {
  for (int n = 0; n < 1024; n++)
    Serial.println(n, DEC);
    delay(50);
  }
}
```

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

La principale différence réside dans la commande `println()`. Dans l'exemple précédent, la fonction `Serial.write()` était utilisée pour écrire le nombre directement dans le port série. Avec `println()`, l'Arduino reformate le nombre en une suite de nombres décimaux et envoie les codes ASCII correspondants à cette chaîne. Par conséquent, au lieu d'envoyer le nombre `254`, il envoie la chaîne `"254\r\n"`, où `\r` correspond au code ASCII du retour chariot et `\n` à celui d'une nouvelle ligne. Ce sont des concepts hérités de la machine à écrire et transmis au monde informatique par les télétypes : un retour chariot déplace le curseur au début de ligne, une nouvelle ligne descend le curseur d'une ligne.

Côté Python, vous pouvez utiliser la fonction `readline()` au lieu de `read()`. `readline()` va lire tous les caractères jusqu'aux symboles du retour chariot et de la nouvelle ligne inclus. Ensuite, Python dispose de nombreuses fonctions de conversion entre les différents types de données, et vous pouvez simplement utiliser la fonction `int()` pour convertir la valeur de la chaîne de caractères en un nombre entier.

```
import serial

port = "/dev/ttyACM0"
serialFromArduino = serial.Serial(port,9600)
serialFromArduino.flushInput()
while True:
    input = serialFromArduino.readline() ❶
    inputAsInteger = int(input) ❷
    print(inputAsInteger * 10) ❸
```

❶ Lecture de toute la chaîne de caractères dans la variable `input`.

❷ Conversion de la chaîne de caractères en un nombre entier.

❸ Affichage du nombre multiplié par 10 pour montrer que c'est réellement un nombre entier et non une chaîne de caractères.

Il est facile d'adapter cet exemple pour que l'Arduino lise les entrées analogiques et qu'il envoie les résultats. Il suffit de modifier la boucle comme suit :

```
void setup() {
  Serial.begin(9600);
}
void loop() {
  int n = analogRead(A0);
```

```
  Serial.println(n, DEC);
  delay(100);
}
```

Si vous changez le script Python pour afficher la variable `inputAsInteger` au lieu de `inputAsInteger * 10`, vous obtiendrez des valeurs situées autour de 200 si rien n'est connecté à la broche analogique 0. Avec un morceau de fil, connectez cette broche à GND, la valeur doit alors devenir 0. Connectez-la à la broche 3V3, vous verrez une valeur située autour de 715, et autour de 1 023 si vous la connectez à la broche 5 V.

Pour aller plus loin

Vous vous rendrez vite compte que la mise en œuvre des projets avec Arduino est très simple quand vous avez les bons logiciels. Beaucoup de projets utilisent du code de communication similaire. Les choses se compliquent dès qu'on dépasse le stade du programme basique « Hello world ». Vous devrez créer des protocoles, ou trouver un protocole existant et l'implémenter, afin que le récepteur comprenne l'émetteur. Les détails des protocoles de communication série dépassent largement le cadre de ce livre. La rubrique Interfacing with Software du site web Arduino Playground (http://playground.arduino.cc) propose de très nombreux exemples présentant les solutions utilisées par d'autres personnes pour résoudre ce genre de problème.

Quelques autres informations qui pourront vous être utiles si vous souhaitez aller plus loin :

- Firmata
  Ce protocole série généraliste (http://arduino.cc/en/Reference/Firmata) a été créé il y a quelque temps par Hans-Christoph Steiner. Simple et lisible, Firmata n'est pas adapté à toutes les applications mais constitue un bon point de départ.

- MIDI
  Si votre projet génère de la musique en temps réel, utilisez le standard MIDI comme protocole de communication série. MIDI ayant été conçu

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

à l'origine pour être transmis sur une interface série, il devrait fonctionner sans difficulté.

- Shields Raspberry Pi compatibles Arduino
  Il existe quelques cartes filles (appelées *shields* dans le monde Arduino) sur le marché qui permettent de connecter les broches GPIO du Raspberry Pi avec un microcontrôleur compatible Arduino. Le shield AlaMode de WyoLum (http://baldwisdom.com/projects/alamode) est une bonne solution qui fournit quelques fonctionnalités intéressantes comme une horloge temps réel.

- Communiquer par réseau
  Vous pouvez même laisser de côté la connexion série et communiquer avec l'Arduino par réseau TCP/IP (par Ethernet ou Wi-Fi). Beaucoup de projets très intéressants utilisent le protocole WebSocket (http://www.websocket.org) avec la plate-forme JavaScript Node.js (http://nodejs.org). Le projet Noduino (http://semu.github.io/noduino) est idéal pour explorer cette technologie.

- Utilisation des broches du port série sur le connecteur du Raspberry Pi
  Le Raspberry Pi fournit des broches d'entrées-sorties sur le connecteur GPIO, dont deux qui peuvent être utilisées pour envoyer et recevoir des données au format série sans passer par un convertisseur USB. Pour ce faire, lisez attentivement le chapitre 8 et vérifiez que vous avez un convertisseur de niveau logique pour protéger les broches 3,3 V du Raspberry Pi, car l'Arduino fonctionne sous 5 V.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

CHAPITRE 7

LES BROCHES D'ENTRÉES-SORTIES

Même si le Raspberry Pi est par nature un ordinateur Linux très bon marché, il y a certains points qui le distinguent d'un ordinateur portable ou de bureau, utilisé pour écrire des e-mails, surfer sur le Web ou encore éditer des documents. Ainsi, il offre la particularité de pouvoir être directement intégré dans des projets électroniques car, en plus de fournir des interfaces standards comme USB, HDMI et Ethernet, il met à disposition de l'utilisateur des broches d'entrées-sorties généralistes, appelées *General Purpose Inputs-Outputs*, ou plus simplement GPIO (figure 7-1).

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

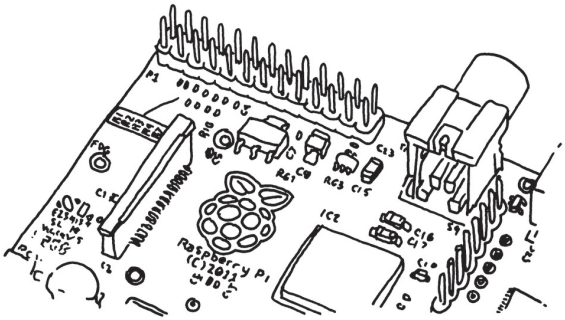


Figure 7-1. Les broches GPIO du Raspberry Pi

Les broches GPIO du Raspberry Pi vous permettent de contrôler d'autres composants électroniques ainsi que des interfaces telles que des LED, des moteurs et des relais. Ces diverses interfaces sont généralement regroupées sous le terme de « sorties ». Pour ce qui est des « entrées », votre Raspberry Pi peut lire et interpréter l'état de boutons, d'interrupteurs, de capteurs de température, de lumière, de mouvement ou de proximité, etc., la liste est très longue. En utilisant les interfaces numériques asynchrones (port série vu au chapitre précédent) et synchrones (SPI et I2C), les applications potentielles sont encore plus nombreuses. Nous allons ici nous intéresser aux applications les plus élémentaires.

Contrairement aux platines Arduino, la puce SoC du Raspberry Pi n'est pas conçue pour être branchée directement à des capteurs analogiques, comme ceux qui mesurent des tensions, des courants, la lumière et la température. Pour ce faire, une puce supplémentaire est nécessaire, que l'on nomme *Convertisseur analogique-numérique* (CAN). Elle convertit une tension en un nombre lisible par le processeur. Consultez l'annexe C pour savoir comment lire des valeurs analogiques avec un CAN.

L'intérêt des broches GPIO est qu'elles permettent d'écrire des programmes qui lisent les entrées et contrôlent les sorties en fonction de règles élaborées, aussi facilement que si vous programmiez un ordinateur de bureau avec une platine Arduino intégrée. Contrairement à une

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

carte microcontrôleur classique, qui ne fournit presque que des broches GPIO, le Raspberry Pi est aussi doté d'interfaces sophistiquées pour un clavier, une souris, une caméra, un moniteur, un port Ethernet, etc., qui sont autant d'entrées et de sorties que vous n'aurez pas à gérer avec des *shields* supplémentaires. Si vous avez déjà créé des projets électroniques avec une carte microcontrôleur comme l'Arduino, vous vous apercevrez que le Raspberry Pi dispose de quelques entrées et sorties supplémentaires. La cerise sur le gâteau est qu'elles sont totalement intégrées, donc il est inutile de fabriquer un circuit additionnel pour les utiliser.

Comme nous l'avons mentionné, le Raspberry Pi dispose d'interfaces pour brancher une souris, un clavier et un moniteur. Ceci ne constitue pas le seul avantage par rapport aux cartes à microcontrôleur standards. Les fonctionnalités supplémentaires suivantes vous seront d'une aide précieuse dans vos projets électroniques.

- **Le système de fichiers**
  La possibilité de lire et d'écrire des données dans le système de fichiers Linux simplifiera énormément la réalisation de vos projets. Par exemple, vous pouvez connecter un capteur de température au Raspberry Pi et programmer sa lecture une fois par seconde. Chaque mesure peut être ajoutée dans un fichier qui pourra être facilement stocké, téléchargé et analysé par un logiciel pour dessiner les courbes de variations. Les températures peuvent aussi être affichées graphiquement par le Raspberry Pi lui-même !

- **Les outils Linux**
  Un large ensemble d'utilitaires en ligne de commande est intégré dans les distributions Linux, ce qui vous permet de travailler avec des fichiers, de contrôler les processus et d'automatiser différentes tâches. Ces outils puissants sont à votre disposition pour tous vos projets. Comme il existe une énorme communauté d'utilisateurs Linux qui dépendent de ces utilitaires de base, vous trouverez facilement des réponses à vos questions en effectuant une simple recherche sur le Web. Pour trouver de l'aide plus générale sur Linux, vous pouvez consulter le site Stack Overflow (http://stackoverflow.com/). Pour des questions spécifiques au Raspberry Pi, rendez-vous sur le forum Raspberry Pi

(http://www.raspberrypi.org/phpBB3/) ou à la section consacrée au Raspberry Pi sur le site Stack Overflow.

- **Les langages**
  Il existe un grand nombre de langages de programmation et les systèmes embarqués utilisant Linux tels que le Raspberry Pi vous permettent de travailler avec le langage de votre choix. Les exemples de ce livre utilisent des scripts en shell et Python, mais il est facile de les adapter à des langages comme C, Java, Perl et bien d'autres.

Utilisation des broches d'entrées-sorties

Pour réaliser les exemples de ce chapitre, vous aurez besoin de quelques fournitures élémentaires en plus de votre Raspberry Pi. La plupart de ces éléments sont disponibles dans n'importe quel magasin de composants électroniques, ou peuvent être commandés en ligne chez Maker Shed, Sparkfun, Adafruit, Farnell, Radiospares, Mouser, Digi-Key... Voici quelques-uns de ces éléments de base :

- une plaque de prototypage (ou platine de prototypage) sans soudure, habituellement appelée *breadboard*, dans laquelle on pique les pattes des composants ;
- des LED diverses ;
- des fils avec des petites fiches mâles ;
- des fils avec des fiches femelles et mâles (ils ne sont pas aussi communs que les fils mâle-mâle, mais nécessaires pour connecter les broches GPIO du Raspberry Pi à la breadboard) ;
- un bouton-poussoir ;
- un assortiment de résistances.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Pour faciliter la connexion entre les composants sur la breadboard et les broches du Pi, nous recommandons aussi le kit Pi Cobbler Breakout d'Adafruit. Celui-ci évite d'avoir à utiliser les fils femelle-mâle. Il est fourni en pièces détachées, donc c'est à vous de souder les composants sur le circuit imprimé. Si vous avez besoin d'aide, n'hésitez pas à consulter le tutoriel d'Adafruit (http://learn.adafruit.com/adafruit-pi-cobbler-kit/overview). Le kit Pi Cobbler Breakout, ainsi que tous les composants de la liste précédente (sauf les câbles femelle-mâle qui ne sont alors plus nécessaires) sont compris dans le Raspberry Pi Starter Kit de MAKE (http://oreil.ly/pikit).

La figure 7-2 présente les broches GPIO que nous pouvons utiliser sans souci, ainsi que les noms correspondants que vous devrez indiquer dans vos programmes. Les autres broches, sans indication, sont attribuées par défaut à d'autres fonctions.

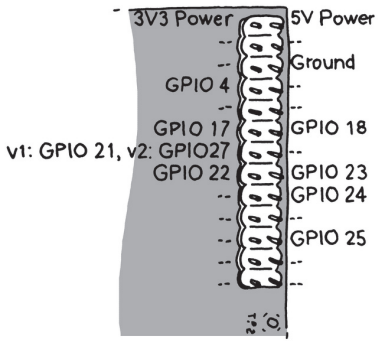


Figure 7-2. Les broches du Raspberry Pi utilisables uniquement comme entrées-sorties. Pour les révisions récentes de la carte, la broche GPIO21 a été remplacée par GPIO27.

Vous avez peut-être remarqué que l'une des broches possède deux numéros de GPIO sur la figure 7.2. Dans les versions récentes de la carte, la broche GPIO21 est devenue la broche GPIO27. Pour déterminer la version de votre carte, tapez `cat /proc/cpuinfo` en ligne de commande. Si votre numéro de révision est `0002` ou `0003`, vous avez une carte de première génération. Si vous voyez un nombre supérieur ou une lettre, votre carte est plus récente.

Sortie numérique : allumage d'une LED

Le moyen le plus facile d'utiliser les broches GPIO en sortie est de les connecter à des LED (diodes électroluminescentes). Vous pouvez alors utiliser la ligne de commande sous Linux pour allumer et éteindre la LED de chaque broche. Une fois que vous aurez compris le fonctionnement de ces commandes, vous serez capable d'allumer une LED quand vous recevrez du courrier, pour indiquer si vous devez prendre un parapluie lorsque vous sortez de la maison ou encore pour signaler qu'il est l'heure d'aller se coucher. Il est aussi très facile d'aller plus loin qu'une simple LED en utilisant, par exemple, un relais pour contrôler l'allumage d'une lampe à certaines heures.

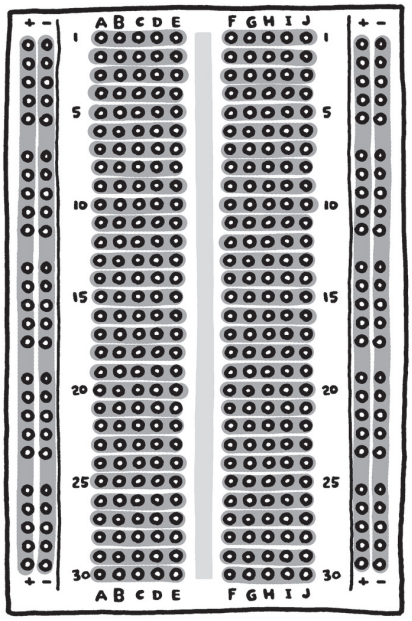


Figure 7-3. La breadboard

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Rappels sur l'utilisation d'une breadboard

Si vous n'avez jamais utilisé une breadboard, il est très important de comprendre comment les points sont connectés entre eux. Dans le diagramme de la figure 7-3, nous avons grisé les connexions internes d'une breadboard typique. Notez que les rails d'alimentation du côté gauche ne sont pas connectés aux rails du côté droit de la breadboard. Il faudra des fils mâle-mâle pour les connecter ensemble si vous avez besoin d'alimenter les deux côtés de votre breadboard. Les rails des breadboards longues sont aussi coupés en leur milieu, n'oubliez pas de faire un pont pour les relier.

Les instructions suivantes vont vous permettre d'allumer et d'éteindre interactivement une LED sans écrire ni compiler de programme.

1. Avec un fil mâle-femelle, connectez la broche GPIO25 du Raspberry Pi à la breadboard. Aidez-vous de la figure 7-2 sur laquelle figure l'emplacement de chaque broche sur le connecteur GPIO du Raspberry Pi.
2. Avec un autre fil, connectez la broche de la masse (appelée GND, de l'anglais *ground*) du Raspberry Pi au rail d'alimentation négative de la breadboard. La masse correspond électriquement au 0 V, c'est la tension de référence.
3. Vous pouvez maintenant connecter une LED (figure 7-4). Mais d'abord, il est important de savoir que les LED sont polarisées, c'est-à-dire qu'elles ne fonctionnent que si leur sens de branchement est respecté. La plupart du temps, les pattes n'ont pas la même longueur afin de vous aider dans le branchement. La patte la plus longue, l'anode, correspond à la tension positive et doit être connectée à une broche GPIO. La patte la plus courte, la cathode, équivaut à la tension négative et doit être branchée à la masse. Un autre moyen de repérer cette différence consiste à regarder la LED par le haut. En effet, on peut voir sur certains modèles un méplat qui indique la cathode (qui doit être connectée à la masse). En cas de doute, vous pouvez tester la polarité de la LED au moyen d'une petite pile bouton 3 V au lithium (par exemple, une CR2032, qu'on trouve couramment sur les cartes mères des PC pour sauvegarder les paramètres du BIOS). Si la LED s'allume, c'est que le sens est correct. Une fois que le bon branchement est trouvé, insérez l'anode de la LED sur la breadboard dans le

même rail conducteur que le fil connecté à la broche GPIO25 (cela connectera donc la broche GPIO25 à la LED). Insérez la cathode de la LED dans le rail connecté à la masse[1].

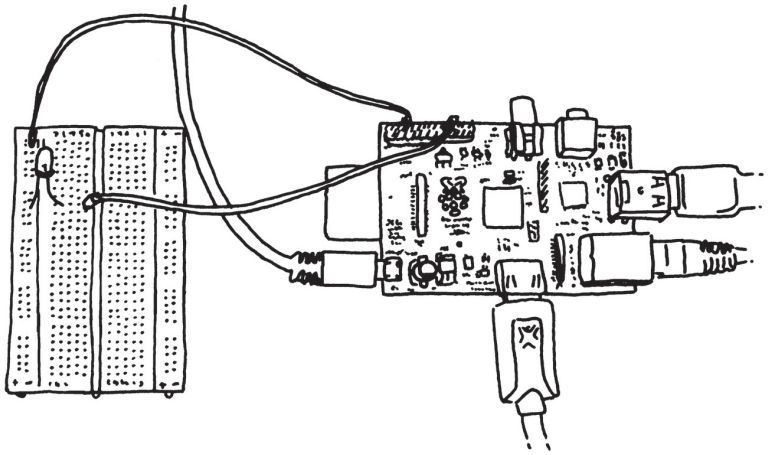


Figure 7-4. Connexion d'une LED au Raspberry Pi

Voici pour les aspects matériels, passons maintenant à la partie logicielle.

4. Le clavier, la souris et l'écran étant branchés, allumez votre Raspberry Pi et tapez votre nom d'utilisateur ainsi que le mot de passe associé. Si vous êtes en ligne de commande, vous êtes prêt. Si vous êtes dans l'environnement graphique, double-cliquez sur l'icône LXTerminal du Bureau pour ouvrir une fenêtre de terminal.

5. Pour accéder aux broches d'entrées-sorties en ligne de commande, vous aurez besoin d'exécuter les commandes en tant qu'utilisateur `root`, qui est le compte administrateur ou super utilisateur du Raspberry Pi. Pour exécuter des commandes avec les droits `root`, tapez `sudo su` en ligne de commande, puis appuyez sur la touche Entrée de votre clavier :

1 NdT : en règle générale, il faut prévoir une résistance de limitation de courant en série avec une LED. Mais dans le cas présent, la puce SoC est alimentée en 3,3 V, ce qui est proche de la tension de fonctionnement d'une LED verte, bleue ou blanche (approximativement 3 V). Brancher une LED directement sur une broche GPIO ne devrait donc pas endommager la LED, ni la broche (qui ne peut fournir qu'un courant limité).

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

```
pi@raspberrypi ~ $ sudo su
root@raspberrypi:/home/pi#
```

Vous noterez que le prompt a subtilement changé, le caractère # au lieu de $ indique que vous exécutez bien les commandes depuis le compte `root`.



Le compte **`root`** accède directement et sans limite à toutes les fonctions et à tous les fichiers du système. Plus rien ne vous protège des erreurs de manipulation ou des fautes de frappe. Soyez donc prudent quand vous exécutez des commandes en tant que **`root`**. Si vous endommagez le système d'exploitation, vous pouvez perdre vos données et votre travail, mais vous pouvez toujours reformater la carte SD avec un Linux fonctionnel. De plus, prenez la bonne habitude de sauvegarder votre travail tous les jours, par exemple en recopiant votre répertoire de travail sur un autre ordinateur. Quand vous aurez terminé d'utiliser le compte **`root`**, tapez **`exit`** pour retourner dans votre compte utilisateur **`pi`**.

6. Avant de pouvoir utiliser en ligne de commande les fonctions permettant d'allumer et d'éteindre une LED sur la broche GPIO25, il faut *exporter* cette broche vers l'espace utilisateur (autrement dit, la rendre accessible aux programmes) en exécutant la commande suivante :

```
root@raspberrypi:/home/pi# echo 25 > /sys/class/gpio/export
```

La commande `echo` envoie le numéro de la broche que vous voulez utiliser (25) vers le fichier `export` qui se trouve dans le dossier `/sys/class/gpio` (attention à ne pas oublier les espaces autour du caractère >). En voyant cela, le kernel va créer un nouveau répertoire dans `/sys/class/gpio` contenant des fichiers de contrôle pour la broche spécifiée. Dans ce cas précis, il va créer le répertoire `/sys/class/gpio/gpio25`.

7. Allez dans ce répertoire avec la commande `cd` et affichez son contenu avec `ls` :

```
root@raspberrypi:/home/pi# cd /sys/class/gpio/gpio25
root@raspberrypi:/sys/class/gpio/gpio25# ls
active_low direction edge power subsystem uevent value
```

Comme nous l'avons vu au chapitre 2, la commande `cd` signifie *change directory*. Elle modifie le répertoire courant et vous évite de taper le chemin complet de chaque fichier. La commande `ls` liste tous les fichiers et dossiers d'un répertoire. Nous utiliserons ici deux fichiers : `direction` et `value`.

8. Le fichier `direction`, comme son nom l'indique, vous permet de configurer la broche comme une entrée (pour lire un bouton-poussoir, par exemple) ou comme une sortie, ce qui est notre cas ici. En effet, une LED est connectée à la broche GPIO25 et nous souhaitons la contrôler. Nous devons donc spécifier que c'est une sortie grâce au mot-clé `out` :

```
root@raspberrypi:/sys/class/gpio/gpio25# echo out >
     direction
```

9. Pour allumer la LED, il faut encore utiliser la commande `echo`, mais cette fois pour écrire la valeur `1` dans le fichier `value` :

```
root@raspberrypi:/sys/class/gpio/gpio25# echo 1 > value
```

10. Appuyez alors sur la touche Entrée du clavier, la LED devrait s'allumer. Pour l'éteindre, il suffit d'utiliser la même commande `echo` mais pour écrire cette fois la valeur `0` dans le fichier `value` :

```
root@raspberrypi:/sys/class/gpio/gpio25# echo 0 > value
```

Les fichiers virtuels

Les fichiers du répertoire **`/sys`** avec lesquels vous travaillez pour le moment ne sont pas de vrais fichiers. Ils ne prennent pas de place sur la carte SD du Raspberry Pi mais ils font partie du système de fichiers virtuel de Linux. Le principe de ce système est que tout peut être vu comme un fichier, afin de réduire le nombre d'outils d'administration et les simplifier. Dans notre cas, lorsque le kernel voit qu'on envoie une donnée dans le fichier de valeur, au lieu de l'écrire sur la carte SD, il va la rediriger vers la broche en question. Ainsi, il n'est pas nécessaire d'écrire un programme spécifique pour contrôler les entrées-sorties.

Puisque l'écriture d'une valeur dans un fichier modifie l'état d'une broche de sortie, comment faire pour lire l'état d'une broche en entrée ? Si vous pensez qu'il faut lire le fichier, vous avez tout à fait raison et c'est justement ce que nous allons faire maintenant.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Entrée numérique : lecture d'un bouton

Les boutons-poussoir, tels que celui de la figure 7-5, permettent de contrôler les entrées numériques simples. De plus, ils se piquent facilement sur une breadboard.

Ces petits boutons-poussoir sont souvent employés dans les projets électroniques. Ici, nous utilisons un modèle appelé KSA, disposant de quatre pattes reliées deux à deux. Sur la figure 7-5, les rangées n° 13 de droite et de gauche sont connectées entre elles, tout comme les deux rangées n° 15. Lorsque vous appuyez sur le bouton, ces deux paires de rangées sont mises en contact. Dans le doute, n'hésitez pas à vérifier les connexions internes avec un multimètre.

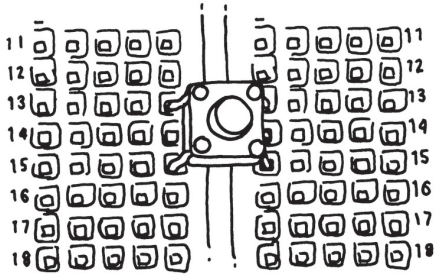


Figure 7-5. Le bouton-poussoir piqué dans la breadboard

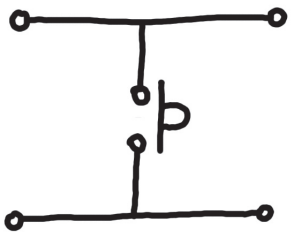


Figure 7-6. Les connexions internes du bouton-poussoir

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Quand vous lisez une entrée numérique avec le Raspberry Pi, la puce SoC détermine si la broche est connectée soit au 3,3 V, soit à la masse. Il faut que ce soit l'un ou l'autre. Si vous essayez de lire une broche qui n'est pas connectée (on dit alors qu'elle est *flottante*), vous aurez des résultats inattendus. Quand vous aurez compris comment les entrées numériques fonctionnent avec un bouton, vous pourrez commencer à utiliser d'autres composants tels que des interrupteurs magnétiques, des joysticks ou même des accepteurs de pièces de distributeurs automatiques.

1. Insérez le bouton dans la breadboard pour que ses pattes chevauchent la rainure du milieu.
2. Avec un fil, connectez la broche GPIO24 du Raspberry Pi à l'une des pattes du bouton-poussoir.
3. Connectez la broche 3,3 V du Raspberry Pi au rail d'alimentation positif de la breadboard.



Assurez-vous que vous avez connecté le bouton à la broche 3,3 V et non à la broche 5 V. Si vous mettez plus de 3,3 V sur une broche d'entrée, votre Raspberry Pi sera définitivement endommagé.

4. Connectez la patte opposée du bouton au bus d'alimentation. À présent, quand vous appuierez sur le bouton, l'alimentation 3,3 V sera connectée à la broche GPIO24.
5. Nous avons mentionné précédemment que les entrées numériques doivent être connectées soit au 3,3 V, soit à la masse. Quand vous relâchez le bouton, la broche GPIO24 n'est connectée ni à l'une, ni à l'autre. Elle est très sensible et reste dans un état flottant influençable par tout parasite électrique, ce qui provoque des résultats imprévisibles. Pour résoudre ce problème, il suffit de connecter une résistance de 10 kΩ (= 10 000 Ω = 10K), identifiée par les bandes colorées marron, noir, orange, puis argent ou or, entre le bouton-poussoir et la masse. Cette résistance est alors appelée *résistance de rappel* et agit comme un élastique : on peut forcer le changement de tension de la broche mais si on ne force plus, la broche retourne à la masse. Quand tout sera branché, le circuit ressemblera à celui représenté à la figure 7-7.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Lorsque le bouton n'est pas appuyé, la résistance « rappelle » la broche GPIO à une tension connue, égale à 0 V, et le CPU lit alors l'état « 0 ». En appuyant sur le bouton, l'alimentation 3,3 V se retrouve connectée à la broche GPIO. Un faible courant parcourt la résistance mais la tension au niveau de la broche GPIO reste égale à 3,3 V, ce que le CPU lit alors comme l'état « 1 ».

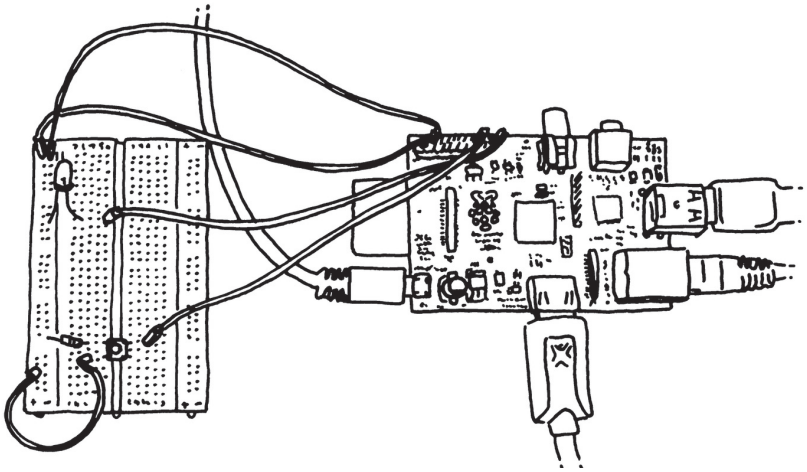


Figure 7-7. La connexion d'un bouton au Raspberry Pi

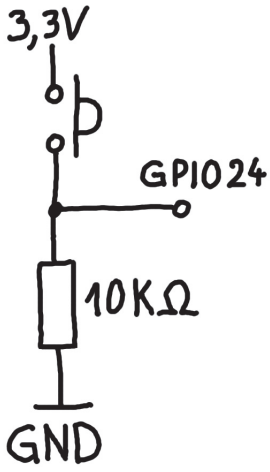


Figure 7-8. Schéma du bouton-poussoir avec la résistance de rappel

6. Maintenant que le circuit est assemblé, vous pouvez lire la valeur de la broche avec la ligne de commande. Si vous n'êtes pas l'utilisateur `root`, tapez `sudo su`.

7. Comme dans l'exemple précédent, vous avez besoin d'exporter la broche d'entrée avant de pouvoir y accéder :

```
root@raspberrypi:/home/pi# echo 24 > /sys/class/gpio/
     export
```

8. Allez dans le répertoire qui a été créé par l'exportation :

```
root@raspberrypi:/home/pi# cd /sys/class/gpio/gpio24
```

9. Spécifiez le sens de la broche pour qu'elle soit en entrée :

```
root@raspberrypi:/sys/class/gpio/gpio24# echo in >
     direction
```

10. Pour lire la valeur d'une broche, utilisez la commande `cat`. Le nom `cat` fait référence au fait que cette commande peut concaténer des fichiers, c'est-à-dire les joindre, les mettre bout à bout. Elle est surtout utilisée pour afficher le contenu d'un fichier dans le terminal.

```
root@raspberrypi:/sys/class/gpio/gpio24# cat value
0
```

11. Le zéro indique que la broche est au niveau de la masse. Appuyez sur le bouton et maintenez-le enfoncé pendant que vous relancez la même commande :

```
root@raspberrypi:/sys/class/gpio/gpio24# cat value
1
```

12. Si vous voyez le chiffre 1, c'est bon !



Pour exécuter une commande que vous avez déjà lancée, appuyez sur la flèche du haut jusqu'à faire apparaître la commande souhaitée, puis appuyez sur la touche Entrée.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Maintenant que vous savez utiliser la ligne de commande pour contrôler une LED et lire l'état d'un bouton, nous allons créer un projet très simple avec des outils intégrés de Linux permettant de gérer des entrées et des sorties numériques.

Projet : séquenceur de lampe avec cron

Imaginons que vous partez demain matin pour un long voyage et que vous souhaitez tenir les cambrioleurs à distance en branchant une lampe sur un séquenceur. Malheureusement, la quincaillerie près de chez vous est fermée pour la journée et votre vol part demain matin. Mais comme vous êtes bricoleur de Raspberry Pi, vous avez déjà le matériel nécessaire chez vous, notamment :

- la carte Raspberry Pi ;
- une breadboard ;
- des fils avec des embouts femelles et mâles ;
- un relais PowerSwitch Tail II ;
- du fil de connexion fin.

Côté logiciel, vous pouvez construire votre propre séquenceur de lampe grâce à deux outils puissants de Linux : les scripts shell et le démon cron.

PROGRAMMER UN SCRIPT SHELL

Un script shell est un fichier qui réunit toutes les commandes que vous auriez à saisir manuellement. Cela permet d'automatiser des séquences d'opérations complexes sans risque de mauvaise manipulation ou d'erreur de copier-coller. Par exemple, vous pouvez mettre dans un script toutes les commandes que vous avez utilisées pour contrôler et lire les broches. C'est d'ailleurs ce que réalise le script shell suivant :

```
#!/bin/bash # ❶
echo Exporte la broche GPIO $1. # ❷
echo $1 > /sys/class/gpio/export # ❸
echo Configuration de la broche en sortie.
echo out > /sys/class/gpio/gpio$1/direction # ❹
echo Met la broche à 1.
echo 1 > /sys/class/gpio/gpio$1/value
```

❶ Cette ligne est nécessaire au début de tous les scripts shell. Après cette ligne, tout ce qui suit un caractère dièse (#) est considéré comme un commentaire et est ignoré.

❷ `$1` est substitué par le shell avec le premier argument de la ligne de commande, qui doit être un numéro valide.

❸ Au lieu d'exporter un numéro de broche spécifique, ce script est flexible car il utilise le premier argument de la ligne de commande.

❹ Notez que le premier argument de la ligne de commande remplace aussi le numéro de la broche ici.

Sauvegardez le script dans un fichier texte nommé `on.sh` et rendez-le exécutable grâce à la commande `chmod` :

```
root@raspberrypi:/home/pi# chmod +x on.sh
```



Ce script contient des commandes qui doivent être exécutées par le super utilisateur `root`. Tapez `sudo su` si des erreurs comme « Permission non accordée » se produisent.

Un argument de ligne de commande est une information qu'on fournit à un programme ou à un script, en la tapant après le nom de la commande. Dans un script shell, vous pouvez récupérer des informations avec des mots-clés qui seront substitués automatiquement : `$1` sera remplacé par le premier argument de la ligne de commande, `$2` sera remplacé par le second, etc. Dans le cas du fichier `on.sh`, tapez le numéro de la broche que vous souhaitez contrôler. Au lieu d'écrire le numéro de la broche directement dans le script, l'argument de ligne de commande rend le

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

script universel et réutilisable pour vos autres projets. Pour exporter la broche GPIO25 et l'allumer, tapez :

```
root@raspberrypi:/home/pi/# ./on.sh 25 ❶
Exporte la broche GPIO 25.
Configuration de la broche en sortie.
Met la broche à 1.
```

❶ Les caractères `./` situés avant le nom de la commande indiquent que vous exécutez le script dans le répertoire où vous vous trouvez.

Si votre LED est encore connectée à la broche GPIO25, elle devrait s'allumer. Essayons maintenant d'écrire un autre script shell, nommé `off.sh`, pour éteindre la LED. Il reprend les commandes que nous avons déjà vues précédemment :

```
#!/bin/bash
echo Met la broche à 0.
echo 0 > /sys/class/gpio/gpio$1/value
echo Désexporte la broche $1
echo $1 > /sys/class/gpio/unexport
```

À présent, rendez le script exécutable, puis exécutez-le :

```
root@raspberrypi:/home/pi# chmod +x off.sh
root@raspberrypi:/home/pi# ./off.sh 25
Met la broche à 0.
Désexporte la broche 25
```

Si tout s'est passé comme prévu, la LED devrait s'éteindre.

CONNEXION D'UNE LAMPE

Bien sûr, une petite LED ne sera pas capable d'émettre assez de lumière pour dissuader d'éventuels voleurs. Branchons maintenant une lampe au Raspberry Pi.

1. Enlevez la LED connectée à la broche GPIO25.
2. Branchez deux brins de fil sur la breadboard : le premier est connecté à la broche GPIO25 du Raspberry Pi, le second est connecté au 0 V de la carte.

3. Branchez l'autre bout du brin de fil connecté à la broche GPIO25 sur le plot « +in » du PowerSwitch Tail.
4. Le brin de fil connecté à la masse doit être connecté sur le plot « -in » du PowerSwitch Tail. Comparez votre branchement à celui représenté à la figure 7-9.
5. Branchez le PowerSwitch Tail à une prise et à une lampe. Vérifiez que l'interrupteur de la lampe reste en position allumée.
6. À présent, quand vous exécutez la commande `./on.sh 25`, la lampe devrait s'allumer. Si vous exécutez `./off.sh 25`, la lampe devrait s'éteindre.

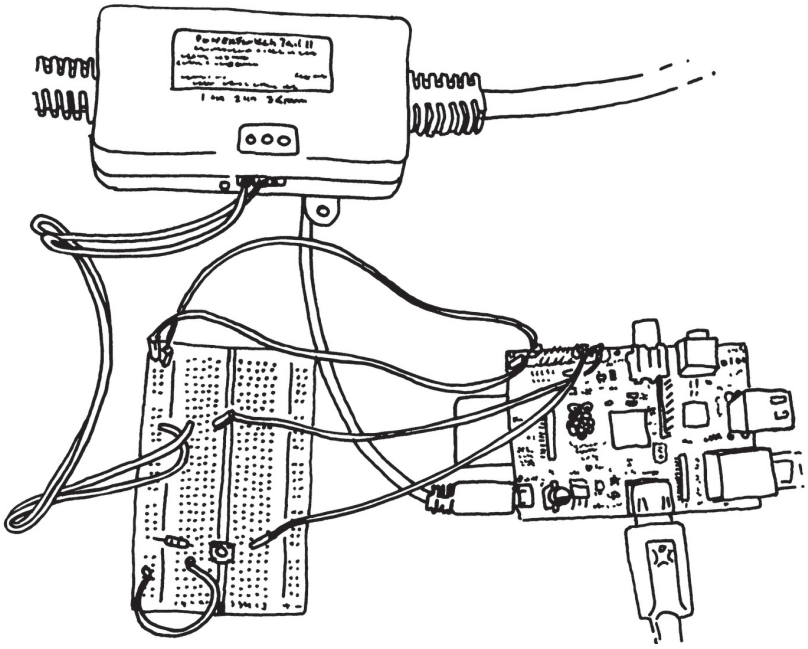


Figure 7-9. Connexion d'un PowerSwitch Tail II au Raspberry Pi

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

À l'intérieur du PowerSwitch Tail se trouvent des composants électroniques qui commutent le courant du secteur en fonction du faible signal du Raspberry Pi. Il contient un relais électromécanique, un interrupteur commandé par un courant électrique, qui fait le « clic » caractéristique lorsqu'il s'allume ou s'éteint. Par ailleurs, assurez-vous de la compatibilité du PowerSwitch Tail avec le type de prise de courant et la tension du secteur de votre pays.

PLANIFIER DES COMMANDES AVEC LE DÉMON CRON

Maintenant que vous contrôlez l'allumage d'une lampe avec un script, en passant par une broche d'entrée-sortie et un adaptateur secteur, il ne reste plus qu'à planifier l'allumage et l'extinction de la lampe à des moments spécifiques de la journée. C'est le travail de cron, le planificateur de Linux. Grâce à lui, vous pouvez exécuter certaines commandes à des heures et des dates données, ou avec une certaine période (par exemple, une fois par heure). Il est bien sûr possible de programmer plusieurs tâches, par exemple une pour allumer la lumière à 20 h, puis une autre pour l'éteindre à 2 h du matin.

Comme pour tout programme qui utilise l'horloge et le calendrier, il est important de vous assurer que le Raspberry Pi est à la bonne date et la bonne heure (voir chapitre 1).

Pour planifier de nouvelles tâches, il sera nécessaire d'éditer la table de cron, c'est-à-dire la liste des commandes à faire exécuter par Linux aux moments indiqués :

```
root@raspberrypi:/home/pi/# crontab -e
```

Ceci va lancer un éditeur de texte (nano) pour modifier la table de cron avec le compte `root`. Au début du fichier, vous verrez des informations et de l'aide à propos des modifications de la table cron. Utilisez les touches

de direction du clavier pour aller tout en bas du fichier, puis ajoutez ces deux lignes à la fin :

```
0 20 * * * /home/pi/on.sh 25
0 2 * * * /home/pi/off.sh 25
```



cron va ignorer toute ligne commençant par le caractère dièse (#). Si vous voulez désactiver une ligne sans l'effacer ou ajouter un commentaire, insérez ce caractère en début de ligne.

Appuyez sur les touches Ctrl + X pour sortir, puis sur la touche O pour sauvegarder le fichier lorsque vous y êtes invité, et enfin sur Entrée pour valider le nom du fichier par défaut. Une fois le fichier sauvegardé et de retour à la ligne de commande, vous verrez le message « Installing new crontab » qui indique que les changements que vous avez effectués seront exécutés par cron.

FORMATER UNE TÂCHE POUR CRON

cron vous permettra de planifier les tâches à des dates, heures ou intervalles spécifiques. Chaque ligne décrit une tâche et contient cinq champs pour indiquer la périodicité (ou six si vous souhaitez planifier sur plusieurs années). Chaque champ, ainsi que la commande à exécuter, est séparé par une espace. Les astérisques indiquent qu'une tâche doit être exécutée pour chaque période.

Tableau 7-1. Entrée cron pour allumer une lampe à 20h00 tous les jours

0	20	*	*	*	/home/pi/on.sh 25
Minute (00)	Heure (20 h)	Chaque jour	Chaque mois	Tous les jours de la semaine	Commande avec le chemin complet

Imaginons maintenant que vous avez envie que votre lampe s'allume seulement chaque jour en semaine, mais pas le week-end. Dans ce cas, l'entrée `crontab` ressemblera ceci :

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Tableau 7-2. Entrée cron pour allumer une lampe tous les jours sauf le week-end

0	20	*	*	1-5	/home/pi/on.sh 25
Minute (00)	Heure (20 h)	Chaque jour	Chaque mois	Du lundi au vendredi	Commande avec le chemin complet

Si vous avez placé un capteur dans votre boîte aux lettres, vous écrirez un script shell qui vérifie si le facteur est passé et vous envoie un e-mail s'il détecte du courrier. Cet exemple montre comment vous pouvez lancer ce script toutes les cinq minutes :

Tableau 7-3. Entrée cron pour vérifier le courrier toutes les cinq minutes

*/5	*	*	*	*	/home/pi/checkMail.sh
Toutes les cinq minutes	Toutes les heures	Tous les jours	Tous les mois	Chaque jour de la semaine	Commande avec le chemin complet

*/5 indique une période de 5 min.

Comme vous pouvez le constater, cron est un outil très utile qui vous permet de planifier des tâches à des dates, heures ou intervalles donnés.

Pour aller plus loin

- Page de référence eLinux pour les GPIO du Raspberry Pi
  Lisez cette documentation de référence (http://elinux.org/RPi_Low-level_peripherals), c'est la page la plus complète sur les broches GPIO du Raspberry Pi. Grâce à ses nombreux détails, elle vous permettra de comprendre et de résoudre de nombreux problèmes, avant ou après leur apparition.

- Adafruit : expansion des ports du Raspberry Pi avec le MCP230xx
  Si vous n'avez pas assez de broches, Adafruit diffuse ce guide d'utilisation du circuit intégré MCP23008 (http://learn.adafruit.com/mcp230xx-gpio-expander-on-the-raspberry-pi), qui fournit 8 broches GPIO en passant par le bus I2C. Son grand frère, le MCP23017, fournit 16 broches GPIO.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

CHAPITRE 8

PROGRAMMATION DES ENTRÉES-SORTIES AVEC PYTHON

Dans le projet proposé à la fin du chapitre précédent, vous avez programmé les broches GPIO du Raspberry Pi avec des scripts shell. Ici, vous allez apprendre à faire la même chose en Python... voire davantage. En effet, Python, comme le shell, permet aussi d'accéder aux broches d'entrées-sorties et d'automatiser la lecture et le contrôle des GPIO.

L'avantage de Python est que le code source est plus facile à lire et à écrire que des scripts shell. Vous aurez également à disposition de nombreux modules Python qui vous permettront de réaliser facilement des tâches complexes grâce à quelques lignes de code seulement. Le tableau 3-2 page 54 liste quelques-uns des modules les plus intéressants dont vous pourriez avoir besoin pour vos projets. Dans ce chapitre,

nous nous concentrerons sur le module `raspberry-gpio-python` (http://code.google.com/p/raspberry-gpio-python/) qui gère les entrées-sorties.

Installation et test du module Python GPIO

Le module `GPIO` est installé par défaut sur les versions les plus récentes de la distribution Raspbian Linux. Si vous utilisez une version plus ancienne, vous devrez probablement l'installer ou effectuer une mise à jour. Pour vérifier si le module est installé, vous pouvez utiliser l'interpréteur en ligne de commande de Python. Celui-ci, comme nous l'avons vu au chapitre 3, vous permet d'exécuter le code Python immédiatement au cours de la saisie, au lieu d'écrire le code dans un fichier pour l'exécuter ensuite.

1. Lancez l'interpréteur Python à partir d'un terminal en tant qu'utilisateur `root`. Puisque le module `raspberry-gpio-python` nécessite les droits du compte `root` pour lire et contrôler les broches, vous devez lancer l'interpréteur Python avec la commande `sudo`.

```
pi@raspberrypi ~ $ sudo python
Python 2.7.3 (default, Jan 1 2012, 11:26:42)
[GCC 4.6.3] on linux2
Type "help", "copyright", "credits" or "license" for more
      information.
>>>
```

2. Au prompt >>>, importez le module en tapant la commande suivante :

```
>>> import RPi.GPIO as GPIO
```

3. Si vous ne recevez pas de message d'erreur, cela signifie que le module est installé.

Si jamais Python ne pouvait pas importer le module `GPIO`, sachez que ce dernier peut être installé grâce à `apt-get`, le gestionnaire de packages utilisé par Raspbian. Pour cela, la procédure est la suivante.

1. Quittez l'interpréteur (utilisez le raccourci clavier Ctrl + D ou tapez la commande `exit()`). Assurez-vous que le Raspberry Pi est correctement connecté à Internet, puis lancez les commandes suivantes pour

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

mettre à jour la liste des packages et démarrer l'installation du module `raspberry-gpio-python` :

```
>>> exit()
pi@raspberrypi ~ $ sudo apt-get update
pi@raspberrypi ~ $ sudo apt-get install python-rpi.gpio
```

2. Relancez l'interpréteur Python et essayez à nouveau d'importer le module (avec les mêmes commandes) :

```
pi@raspberrypi ~ $ sudo python
Python 2.7.3 (default, Jan 1 2012, 11:26:42)
[GCC 4.6.3] on linux2
Type "help", "copyright", "credits" or "license" for more
     information.
>>> import RPi.GPIO as GPIO
>>>
```



Dans ce chapitre, nous utilisons Python 2.7 au lieu de Python 3 car l'un des modules utilisés pour ce projet fonctionne uniquement avec Python 2.x sur le Raspberry Pi. Quand vous tapez la commande `python` dans un terminal du Pi, c'est Python 2.7 qui s'exécute par défaut. Cela pourrait changer à l'avenir et dans ce cas, il sera toujours possible d'exécuter explicitement Python 2.7 en tapant `python2.7` au lieu de `python`.

Une différence importante entre les versions 2.7 et 3 concerne la syntaxe de la commande qui affiche du texte sur la console. Avec Python 2.x, cette commande s'écrit `print "Hello, World!"` mais en Python 3, il faut ajouter des parenthèses : `print("Hello, World!")`.

Si l'erreur d'importation a disparu, alors le système est prêt pour nos premiers essais.

1. Avant d'utiliser les broches, vous devez indiquer au module `GPIO` comment votre code y accédera. Au chapitre 7, les numéros des broches ne correspondaient pas à ceux du connecteur. Vous avez utilisé la numérotation de la puce SoC de Broadcom. Avec le module `GPIO`, vous pouvez choisir entre ces deux types de numérotation. Pour utiliser les numéros du connecteur, il faut taper `GPIO.setmode(GPIO.`

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

`BOARD)`. Mais pour le moment, conservons les numéros des broches du chapitre 7 (avec `GPIO.setmode(GPIO.BCM)`). Ce sont les mêmes que ceux utilisés par le Pi Cobbler d'Adafruit et les autres cartes d'interfaces similaires :

```
>>> GPIO.setmode(GPIO.BCM)
```

2. Définissez la direction de la broche 25 en tant que sortie :

```
>>> GPIO.setup(25, GPIO.OUT)
```

3. Connectez une LED à la broche 25 comme vous l'avez fait au chapitre précédent (voir l'encadré « Rappels sur l'utilisation d'une breadboard », page 113).

4. Allumez la LED :

```
>>> GPIO.output(25, GPIO.HIGH)
```

5. Éteignez la LED :

```
>>> GPIO.output(25, GPIO.LOW)
```

6. Quittez l'interpréteur Python :

```
>>> exit()
pi@raspberrypi ~ $
```



Au chapitre 7, vous avez vu que les broches d'entrées-sorties digitales acceptent ou génèrent des signaux dont la tension est soit 3,3 volts, soit 0 volts (à la masse). Dans le domaine de l'électronique numérique, nous appelons ces niveaux respectivement « haut » et « bas ». N'oubliez pas que d'autres systèmes utilisent d'autres tensions que 3,3 volts pour indiquer le niveau haut, par exemple 1,8 V, 2,5 V ou 5 V. Si vous prévoyez de connecter votre Raspberry Pi à une interface numérique via ses broches GPIO, il est important que toutes les tensions soient compatibles.

Grâce aux étapes précédentes, vous avez vu comment contrôler des broches GPIO avec des commandes Python saisies directement dans l'interpréteur. De la même manière que vous avez créé un script shell pour gérer les broches au chapitre 7, vous allez maintenant pouvoir écrire du code Python pour lire et contrôler les broches dans un vrai programme.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Faire clignoter une LED

Pour faire clignoter une LED, vous allez utiliser certaines commandes Python déjà abordées ainsi que d'autres que nous allons vous présenter ici. Pour cet exemple, nous supposons que vous restez dans l'environnement graphique, mais vous pouvez tout à fait écrire et exécuter ces scripts Python dans un terminal, en ligne de commande.

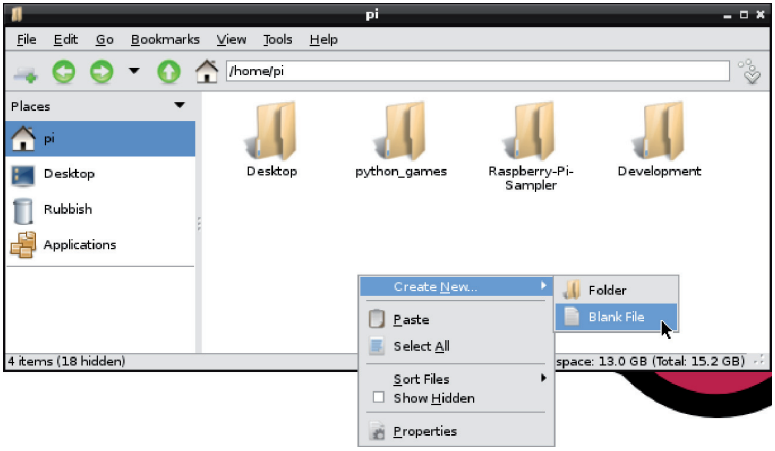


Figure 8-1. Création d'un nouveau fichier dans le répertoire home

1. Ouvrez le gestionnaire de fichiers (File Manager) en cliquant sur son icône dans la barre de menus.
2. Assurez-vous que vous êtes bien dans votre répertoire personnel, qui se trouve par défaut dans `/home/pi`. Si ce n'est pas le cas, cliquez sur l'icône représentant une maison (Home) dans l'arborescence Places.
3. Créez un fichier vide dans votre répertoire `/home/pi`. Pour ce faire, effectuez un clic droit dans la fenêtre du répertoire personnel, sélectionnez Create New..., puis double-cliquez sur Blank File. Nommez ce nouveau fichier `blink.py`.
4. Double-cliquez sur ce fichier pour l'ouvrir avec Leafpad, l'éditeur de texte par défaut en mode graphique.

5. Saisissez le code source suivant dans l'éditeur et enregistrez le fichier :

```
import RPi.GPIO as GPIO ❶
import time ❷

GPIO.setmode(GPIO.BCM) ❸
GPIO.setup(25, GPIO.OUT) ❹

while True: ❺
    GPIO.output(25, GPIO.HIGH) ❻
    time.sleep(1) ❼
    GPIO.output(25, GPIO.LOW) ❽
    time.sleep(1) ❾
```

❶ Importe le code nécessaire pour le contrôle des GPIO.

❷ Importe le code nécessaire pour la fonction `sleep`.

❸ Utilise la numérotation des broches de la puce.

❹ Configure la broche 25 en sortie.

❺ Crée une boucle infinie avec le code indenté.

❻ Allume la LED.

❼ Attend une seconde.

❽ Éteint la LED.

❾ Attend une seconde.



N'oubliez pas que l'indentation est un élément essentiel du code source Python.

6. Ouvrez LXTerminal et lancez les commandes suivantes pour vous assurer que le répertoire de travail est bien `/home/pi`. Exécutez ensuite le script :

```
pi@raspberrypi ~/Development $ cd ~
pi@raspberrypi ~ $ sudo python blink.py
```

7. La LED devrait à présent clignoter.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

8. Appuyez sur les touches Ctrl + C pour arrêter le script et retourner à la ligne de commande.

Essayez de modifier le script pour faire clignoter la LED plus rapidement en utilisant des nombres décimaux dans la commande `time.sleep()`. Vous pouvez aussi ajouter d'autres LED et les faire clignoter comme bon vous semble. Pour cela, les broches GPIO disponibles sont : 4, 17, 18, 22, 23, 24, 25 ou 27 (voir figure 7-2, page 111).



NdT : En informatique, on utilise les conventions anglaises, les nombres décimaux s'écrivent donc avec un point au lieu d'une virgule. Pour une temporisation d'une demi-seconde, il faut donc écrire `time.sleep(0.5)`.

Vérifiez la version de votre carte pour que les numéros des entrées GPIO correspondent bien aux broches du connecteur. Par exemple, l'entrée GPIO21 a été remplacée par GPIO27 sur la broche 13 du connecteur P1 avec la version 2 du Raspberry Pi. Les informations à jour sont disponibles sur la page http://elinux.org/RPi_Low-level_peripherals.

Lire l'état d'un bouton

Pour détecter un événement ou pour déclencher une action lorsqu'un bouton est appuyé, plusieurs techniques peuvent être utilisées. La plus rapide à programmer est celle dite du *polling*, qui consiste à lire le port d'entrée régulièrement. Dans le projet suivant, vous allez vérifier si un bouton (connecté à une broche d'entrée) est à 3,3 volts ou bien à la masse. Afin d'apprendre à « poller » (scruter), vous allez maintenant créer un nouveau script Python qui affichera du texte à l'écran chaque fois que l'utilisateur appuiera sur un bouton.

1. Branchez un bouton comme nous l'avons vu au chapitre 7, dans la section « Entrée numérique : lecture d'un bouton » page 117, où la broche 24 sert d'entrée. N'oubliez pas la résistance de rappel *(pull-down)* connectée entre la masse et l'entrée GPIO24.

2. Créez un fichier dans votre répertoire personnel et nommez-le `button.py`.

3. Ouvrez le fichier dans l'éditeur par défaut et saisissez le code source suivant :

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
GPIO.setup(24, GPIO.IN) ❶

count = 0 ❷

while True:
    inputValue = GPIO.input(24) ❸
    if (inputValue == True): ❹
        count = count + 1 ❺
        print("Bouton appuyé " + str(count) + " fois.") ❻
    time.sleep(.01) ❼
```

❶ Configure la broche 24 en tant qu'entrée.

❷ Crée une variable `count` et l'initialise à la valeur 0.

❸ Transfère la valeur de la broche 24 dans la variable `inputValue`.

❹ Vérifie si la valeur de la variable est `True` (détecte si le bouton est appuyé).

❺ Si la variable vaut `True`, le compteur est incrémenté.

❻ Affiche le texte à l'écran.

❼ Attend un instant et laisse les autres programmes s'exécuter, pour ne pas monopoliser le processeur.

4. Retournez ensuite dans le terminal LXTerminal et exécutez le script suivant :

```
pi@raspberrypi ~ $ sudo python button.py
```

5. Vous pouvez à présent tester le bouton. Si tout se déroule correctement, le message « Bouton appuyé… » apparaîtra lorsque vous cliquerez sur le bouton.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Le code précédent vérifie l'état du bouton 100 fois par seconde. Ainsi, à moins que vos doigts soient extrêmement agiles, ce message s'affichera plusieurs fois à chaque clic sur le bouton. La commande `time.sleep(.01)` contrôle l'intervalle entre les lectures du bouton.

Mais ne serait-il pas mieux de vérifier l'état du bouton de façon continue ? Supposons que vous supprimiez la commande `time.sleep(.01)` du programme : la boucle s'exécuterait alors beaucoup plus rapidement, ce qui vous permettrait de détecter chaque nouveau clic plus rapidement. Cependant, cette approche présente quelques inconvénients. En effet, le processeur du Pi serait sollicité en permanence, laissant peu de ressources disponibles pour le fonctionnement d'autres programmes. Votre Raspberry Pi consommerait aussi plus d'énergie que nécessaire, ce qui le ferait d'ailleurs peut-être chauffer. Dans la mesure où le fichier `button.py` doit partager les ressources du système avec d'autres programmes, il faut rester prudent et s'assurer qu'il ne les accapare pas.

Maintenant, ajoutons quelques lignes de code pour permettre au programme de mieux enregistrer un clic unique sur le bouton :

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
GPIO.setup(24, GPIO.IN)

count = 0

while True:
    inputValue = GPIO.input(24)
    if (inputValue == True):
        count = count + 1
        print("Bouton appuyé " + str(count) + " fois.")
        time.sleep(.3) ❶
    time.sleep(.01)
```

❶ Facilite l'enregistrement d'un clic unique sur le bouton.

Cette ligne de code supplémentaire permet d'enregistrer une seule fois chaque clic sur le bouton, mais cette solution n'est pas parfaite. En effet, si vous maintenez le bouton appuyé un peu plus longtemps que nécessaire, d'autres clics seront enregistrés et s'afficheront trois fois par seconde, même si vous ne pressez le bouton qu'une seule fois. Maintenant, essayez

de cliquer sur le bouton très rapidement et plusieurs fois de suite : tous les clics ne seront pas forcément enregistrés puisque ce programme ne détectera pas plus de trois clics par seconde.

Voici l'un des défis auxquels vous serez confronté en utilisant la technique du polling pour vérifier l'état d'une entrée numérique. De nombreuses méthodes logicielles existent pour les contourner, en fonction du type de périphérique et de ses caractéristiques. Elles reposent sur la mémorisation et la comparaison des états précédents du bouton, et se compliquent un peu plus pour filtrer les parasites électriques et mécaniques (réunis sous le terme de « rebonds »).

Un autre moyen de gérer ce genre de situation est l'interruption. Cette technique, qui utilise des circuits internes du processeur, permet à un bloc spécifique de code de s'exécuter quand une broche change d'état. Pour accéder aux interruptions, utilisez le module `RPi.GPIO`. Vous trouverez de nombreuses informations sur l'utilisation des interruptions à l'adresse suivante : https://pypi.python.org/pypi/RPi.

Projet : boîte à bruitages

Maintenant que vous savez comment lire les entrées du Raspberry Pi en Python, vous pouvez exploiter les fonctions sonores du module Pygame. Dans ce projet, vous allez découvrir comment fabriquer une boîte à bruitages qui émettra des sons préenregistrés lorsque vous cliquerez sur ses différents boutons. Pour réaliser cette boîte à bruitages, vous aurez besoin du matériel suivant (en plus de votre Pi) :

- trois boutons-poussoir ;
- des cavaliers femelle-mâle ;
- des cavaliers standards ou des fils de connexion ajustés à la bonne taille ;
- une plaque de prototypage sans soudure *(breadboard)* ;

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

- trois résistances de 10 kW ;
- des enceintes d'ordinateur ou un moniteur HDMI avec enceintes intégrées.

Vous aurez aussi besoin de fichiers audio non compressés au format WAV. La distribution Raspbian en fournit quelques-uns par défaut pour la réalisation de vos tests.

```
pi@raspberrypi ~ $ cd /
pi@raspberrypi / $ find|grep '[.]wav'
/home/pi/python_games/match3.wav
/home/pi/python_games/match4.wav
/home/pi/python_games/match5.wav
/home/pi/python_games/match2.wav
/home/pi/python_games/match0.wav
(etc.)
```

Dès que votre boîte à bruitages fonctionnera, il sera facile de remplacer ces fichiers avec tout autre son de votre choix (vous devrez peut-être les convertir au format WAV au préalable).

Commençons tout d'abord par la construction du circuit.

1. Avec un cavalier femelle-mâle, connectez la broche de masse du Raspberry Pi au rail négatif de la breadboard.
2. Avec un autre cavalier femelle-mâle, connectez la broche 3,3 V du Raspberry Pi au rail positif de la breadboard.
3. Insérez les trois boutons-poussoir dans la breadboard, en enjambant la rainure centrale.
4. Avec des cavaliers standards ou de petits bouts de fil, connectez le rail positif de la breadboard à la broche supérieure de chaque bouton-poussoir.
5. Vous pouvez à présent ajouter les résistances de rappel. Connectez la patte inférieure de chaque bouton-poussoir à une résistance de 10K reliée à la masse (le rail négatif).
6. Avec des cavaliers femelle-mâle, connectez la patte inférieure de chaque bouton-poussoir (celle avec la résistance de 10K) aux broches GPIO du Raspberry Pi. Pour ce projet, nous utilisons les broches 23, 24 et 25.

La figure 8-2 montre le circuit complet. Ce diagramme a été réalisé avec Fritzing (http://fritzing.org), un logiciel open source qui permet aux débutants de dessiner des schémas ou des circuits imprimés.

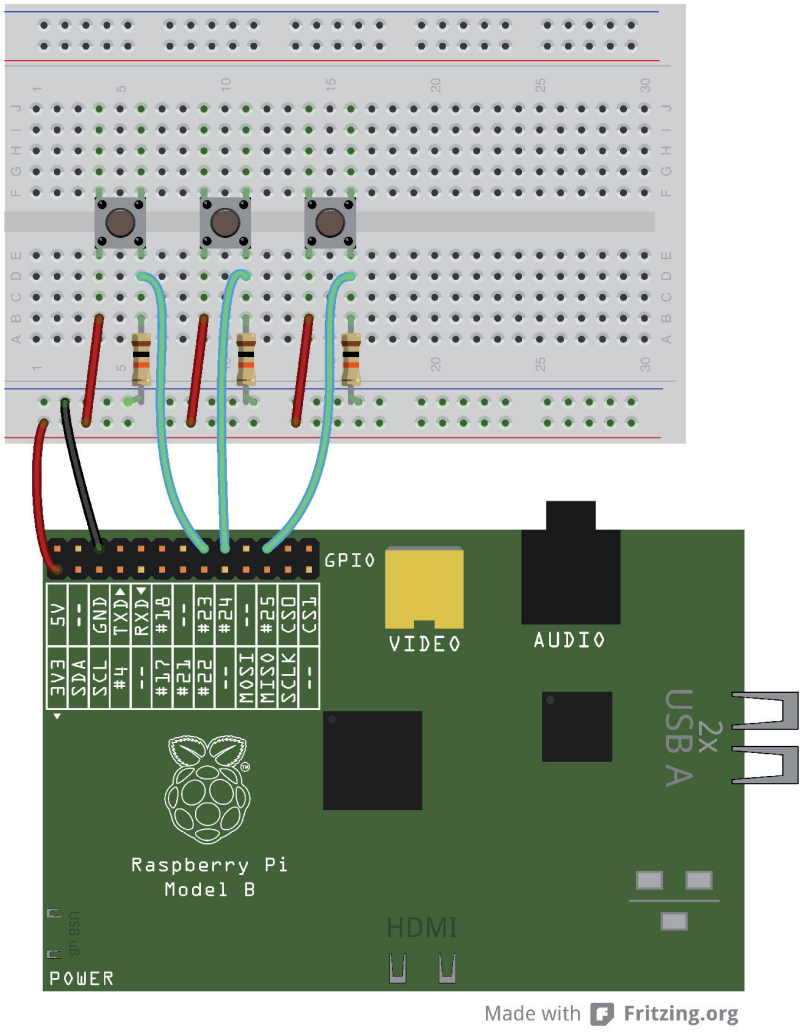


Figure 8-2. Le circuit complet du projet de boîte à bruitages

Maintenant que votre circuit est entièrement câblé, passons au codage.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

1. Créez un nouveau dossier dans votre répertoire personnel et nommez-le `soundboard`. Pour ce faire, vous pouvez saisir la commande suivante dans le shell :

```
pi@raspberrypi xyz $ cd
pi@raspberrypi ~ $ mkdir soundboard
```

2. Ouvrez ce dossier et créez-y un fichier `soundboard.py`.

3. Ouvrez le fichier `soundboard.py` dans l'éditeur par défaut et saisissez le code suivant :

```
import pygame.mixer
from time import sleep
import RPi.GPIO as GPIO
from sys import exit

GPIO.setmode(GPIO.BCM)
GPIO.setup(23, GPIO.IN)
GPIO.setup(24, GPIO.IN)
GPIO.setup(25, GPIO.IN)

pygame.mixer.init(48000, -16, 1, 1024) ❶

soundA = pygame.mixer.Sound("/usr/share/sounds/alsa/
     Front_Center.wav") ❷
soundB = pygame.mixer.Sound("/usr/share/sounds/alsa/
     Front_Left.wav")
soundC = pygame.mixer.Sound("/usr/share/sounds/alsa/
     Front_Right.wav")

soundChannelA = pygame.mixer.Channel(1) ❸
soundChannelB = pygame.mixer.Channel(2)
soundChannelC = pygame.mixer.Channel(3)

print "Soundboard Ready." ❹

while True:
    try:
        if (GPIO.input(23) == True): ❺
            soundChannelA.play(soundA) ❻
        if (GPIO.input(24) == True):
            soundChannelB.play(soundB)
        if (GPIO.input(25) == True):
            soundChannelC.play(soundC)
        sleep(.01) ❼
    except KeyboardInterrupt: ❽
        exit()
```

❶ Initialise le mixeur Pygame.

❷ Charge les fichiers audio à lire.

❸ Configure trois canaux, un pour chaque fichier audio, afin que vous puissiez lire plus d'un fichier à la fois.

❹ Indique à l'utilisateur que la boîte à bruitages est prête (avec la syntaxe Python 2).

❺ Si la broche est à un niveau haut (le bouton correspondant est donc cliqué), la ligne suivante est exécutée.

❻ Lit le fichier audio.

❼ Permet de ne pas monopoliser le processeur avec des vérifications trop fréquentes de l'état des boutons.

❽ Permet de sortir proprement du script en appuyant sur les touches Ctrl + C, sans afficher les informations de débogage.

4. Allez en ligne de commande et naviguez vers le dossier contenant le fichier `soundboard.py`, puis exécutez ce dernier avec Python 2.7 :

```
pi@raspberrypi ~ $ cd ~/soundboard
pi@raspberrypi ~/soundboard $ sudo python soundboard.py
```

5. Dès que le message « Soundboard Ready » apparaît, vous pouvez appuyer sur les boutons pour écouter les fichiers audio.



Bien que le module Pygame soit disponible pour Python 3, l'installation par défaut du Raspberry Pi utilise la version Python 2.

En fonction de la manière dont votre Raspberry Pi est configuré, le son peut être envoyé soit à votre écran par le port HDMI, soit sur la sortie audio analogique par la prise jack 3,5 mm de la carte.

Pour utiliser la sortie audio analogique, sortez du script en appuyant sur les touches Ctrl + C et exécutez la commande suivante :

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

```
pi@raspberrypi ~/soundboard $ sudo amixer cset numid=3 1
```

Pour envoyer le son au moniteur par la sortie HDMI, utilisez la commande suivante :

```
pi@raspberrypi ~/soundboard $ sudo amixer cset numid=3 2
```

Les fichiers audio présents par défaut sur le Raspberry Pi peuvent être supprimés et remplacés par des fichiers de votre choix tels que des applaudissements, des rires, des sonneries ou encore des chants d'oiseaux… Ajoutez-les au répertoire `soundboard` et modifiez votre code pour utiliser ces fichiers. Si vous voulez utiliser plus de sons dans votre boîte à bruitages, vous pouvez ajouter des boutons et adapter le code source.

Pour aller plus loin

- RPi.GPIO
  La bibliothèque RPi.GPIO (http://code.google.com/p/raspberry-gpio-python) est encore en cours de développement au moment où nous écrivons ces lignes. Consultez la page du projet pour vérifier les dernières mises à jour.

- Utilisation du MCP3008
  Adafruit propose un excellent tutoriel (http://learn.adafruit.com/reading-a-analog-in-and-controlling-audio-volume-with-the-raspberry-pi/overview) sur l'utilisation du convertisseur analogique-numérique MCP3008, qui permet au Raspberry Pi de lire les signaux de capteurs analogiques.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

CHAPITRE 9

UTILISER DES WEBCAMS

L'un des grands avantages du Raspberry Pi est qu'il supporte quasiment tous les appareils USB courants grâce aux pilotes fournis par Linux. Ainsi, vous pourrez bien entendu y connecter un clavier et une souris, mais aussi d'autres périphériques comme des imprimantes, des dongles Wi-Fi ou Bluetooth, des cartes mémoire Flash ou encore des disques durs externes (liste non exhaustive). Dans ce chapitre, vous allez découvrir comment intégrer une webcam USB dans vos projets Raspberry Pi.

Bien qu'elle ne soit pas encore aussi fréquemment utilisée que le clavier ou la souris, la webcam est désormais un périphérique standard, installée sur la plupart des ordinateurs modernes. Les grandes marques en proposent pour moins de 20 € et les modèles génériques sont encore moins chers. Par

ailleurs, elles sont souvent reconnues sans difficultés par Linux. Contrairement à Windows, vous n'aurez probablement pas besoin de cliquer, d'installer un pilote ou de modifier des fichiers de configuration avant ou après le branchement de la webcam sur le port USB de votre carte.

Au chapitre 1, nous avons mentionné que le Raspberry Pi dispose d'un connecteur appelé *Camera Serial Interface* (CSI, Interface pour caméra série en français, figure 9-1). Cette interface est assez répandue dans les appareils mobiles (smartphones et tablettes), auxquels le microprocesseur BCM2835 du Raspberry Pi est destiné. Cependant l'interface CSI est confinée au monde industriel et il est actuellement impossible de trouver dans le commerce une webcam au format CSI. Pour pallier ce manque, la fondation Raspberry Pi a développé un module caméra spécialement adapté au connecteur CSI. Au moment où nous rédigeons ces lignes, ce module commence seulement à être disponible. Mais en raison des délais de livraison et du prix, nous vous recommandons d'utiliser pour l'instant une webcam USB classique (figure 9-2).

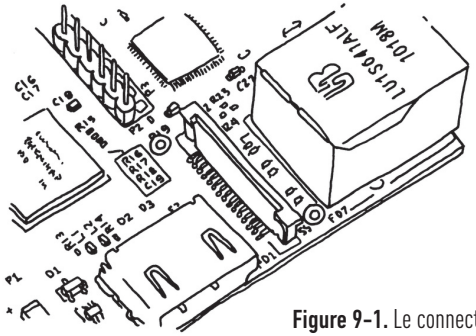


Figure 9-1. Le connecteur de l'interface CSI du Raspberry Pi

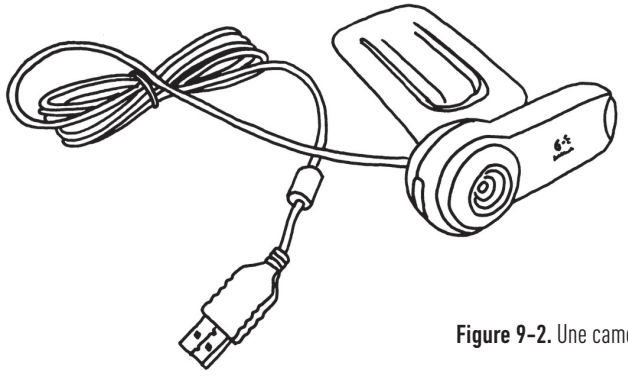


Figure 9-2. Une caméra USB courante

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Tester sa webcam

Il existe de très nombreux modèles de webcams USB, mais rien ne vous garantit qu'ils fonctionneront tous du premier coup avec le Raspberry Pi. Nous vous conseillons de faire une recherche sur Internet avant d'effectuer tout achat afin de choisir un modèle adapté. N'hésitez pas à consulter la rubrique du site Linux.org consacrée aux webcams (http://elinux.org/RPi_VerifiedPeripherals#USB_Webcams) afin de prendre connaissance de la liste des modèles compatibles avec le Raspberry Pi.

Veillez également à brancher votre webcam sur un hub USB avec une alimentation externe. Cette recommandation est importante car les ports USB du Raspberry Pi ne peuvent fournir qu'un courant limité, lequel pourra s'avérer être insuffisant une fois que vous aurez branché le clavier et la souris. Grâce à l'alimentation supplémentaire dédiée au hub, tous ces appareils devraient fonctionner sans souci.

Lorsque votre webcam est prête à être testée, lancez la commande `apt-get` dans un terminal pour installer `luvcview`, une petite application qui permet de lire les données de la webcam :

```
pi@raspberrypi ~ $ sudo apt-get install luvcview
```



D'autres logiciels proposant les mêmes fonctionnalités existent sous Linux, notamment cheese (https://projects.gnome.org/cheese/).

Une fois l'installation terminée, démarrez l'application en tapant `luvcview` dans un terminal de l'environnement graphique. Une fenêtre devrait alors apparaître, affichant l'image capturée par la première source vidéo trouvée dans le répertoire `/dev`, probablement `/dev/video0`. Notez la résolution de l'image indiquée dans le terminal. Si la vidéo est un peu saccadée ou de mauvaise qualité, vous pouvez améliorer l'affichage en réduisant la taille des images. Par exemple, si la dimension par défaut est de 640 × 480 pixels, fermez `luvcview` et relancez l'application en spécifiant une dimension réduite de moitié grâce à la commande suivante :

```
pi@raspberrypi ~ $ luvcview -s 320x240
```

Si aucune vidéo ne s'affiche au bout de dix secondes, il faut diagnostiquer l'origine du blocage avant de pouvoir continuer. Pour vérifier si la webcam (ou tout autre périphérique) a été correctement détectée par Linux lors de son branchement ou du démarrage, saisissez la commande `dmesg` dans un terminal. Cette commande liste les derniers messages du noyau Linux et elle devrait afficher des informations détaillées telles que celles présentées ci-après :

```
usb 1-4: New USB device found, idVendor=1e4e, idProduct=0102
usb 1-4: New USB device strings: Mfr=1, Product=2,
SerialNumber=0
usb 1-4: Product: USB2.0 Camera
usb 1-4: Manufacturer: Etron Technology, Inc.
```

`dmesg` vous fournira toutes les informations nécessaires pour comprendre d'où vient le problème et ce qu'il se passe dans votre système d'exploitation. Vous serez ainsi en mesure de corriger les éventuelles erreurs et rétablir votre système.

Installation et test de SimpleCV

Nous accéderons à la webcam en Python au moyen de SimpleCV, un ensemble très riche de fonctions d'analyse d'images destinées à la « vision artificielle », (branche de l'intelligence artificielle qui traite des images). Avec SimpleCV, il est très facile de lire des images de la webcam, de les afficher à l'écran ou de les enregistrer dans des fichiers. Mais son principal intérêt réside dans sa collection d'algorithmes de vision artificielle, lesquels peuvent donner des résultats fabuleux. En plus des transformations d'images de base, SimpleCV permet ainsi de détecter, reconnaître et suivre des objets sur une image ou une vidéo. Reportez-vous à la section « Reconnaissance de visage », page 156, qui propose un exemple de reconnaissance de visage simple).

Pour installer SimpleCV pour Python, vous devez tout d'abord installer les autres bibliothèques sur lesquelles il s'appuie. Pour cela, saisissez la commande `apt-get` dans un terminal :

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

```
pi@raspberrypi ~ $ sudo apt-get install python-opencv
     python-scipy python-numpy python-pip
```

Cette installation est assez longue, armez-vous de patience. Installez ensuite SimpleCV en utilisant la commande suivante :

```
pi@raspberrypi ~ $ sudo pip install https://github.com/
     ingenuitas/SimpleCV/zipball/master
```

Vérifiez alors que tout s'est correctement déroulé et que l'application fonctionne bien en lançant Python dans un terminal et en important la bibliothèque :

```
pi@raspberrypi ~ $ python
Python 2.7.3rc2 (default, May 6 2012, 20:02:25)
[GCC 4.6.3] on linux2
Type "help", "copyright", "credits" or "license" for more
     information.
>>> import SimpleCV
>>>
```

Si aucune erreur ne se produit, vous pouvez vous lancer dans le monde de la vision artificielle avec le Rapsberry Pi.

Afficher une image

Pour la plupart des exemples de ce chapitre, vous devrez vous trouver dans l'environnement graphique pour pouvoir afficher les images à l'écran. Vous pouvez travailler dans IDLE ou éditer et enregistrer vos fichiers `.py` avec Leafpad, pour les exécuter ensuite dans une fenêtre de terminal.

Pour commencer, nous allons vous présenter les bases de SimpleCV en traitant des images provenant de fichiers, puis vous découvrirez comment les acquérir avec la webcam. Pour terminer, vous verrez comment procéder à la reconnaissance des visages.

1. Créez un nouveau répertoire dans votre répertoire personnel et nommez-le `simplecv-test`.

2. Avec le navigateur Midori, cherchez une image intéressante sur Internet. Nous avons choisi une photographie de framboises trouvée sur Wikipédia et nous l'avons renommée `raspberries.jpg`.
3. Effectuez un clic droit sur l'image et sélectionnez Sauver l'image dans le menu contextuel qui apparaît.
4. Enregistrez l'image dans le répertoire `simplecv-test`.
5. Dans le gestionnaire de fichiers (accessible par le menu Accessoires), ouvrez le répertoire `simplecv-test` et effectuez un clic droit dessus. Dans le menu contextuel, choisissez l'item permettant de créer un fichier vide.
6. Renommez ce fichier vide `affiche-image.py`.
7. Double-cliquez sur ce fichier pour l'ouvrir dans Leafpad.
8. Saisissez le code de l'exemple 9-1 (voir page suivante).
9. Enregistrez le fichier `affiche-image.py` et exécutez-le dans un terminal. Si tout se passe comme prévu, vous verrez l'image enregistrée avec Midori dans une nouvelle fenêtre, comme sur la figure 9-3. Vous pouvez fermer la fenêtre ou utiliser le raccourci clavier Ctrl + C dans le terminal pour arrêter le script.

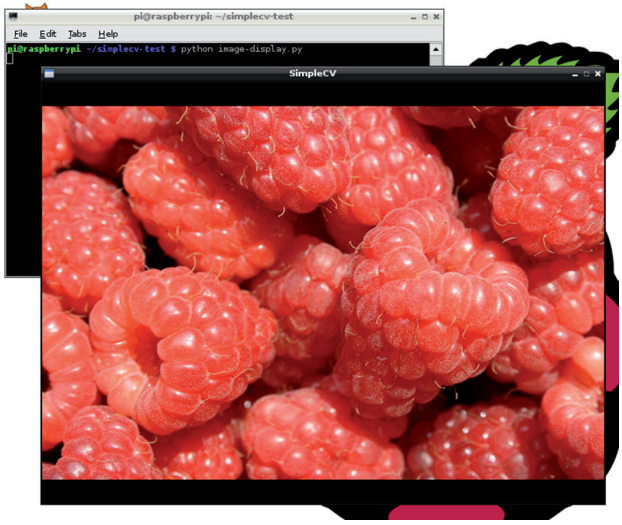


Figure 9-3. L'image des framboises est affichée dans la nouvelle fenêtre.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Exemple 9-1. Code source du fichier affiche-image.py

```
from SimpleCV import Image, Display ❶
from time import sleep

myDisplay = Display() ❷

raspberryImage = Image("raspberries.jpg") ❸

raspberryImage.save(myDisplay) ❹

while not myDisplay.isDone(): ❺
    sleep(0.1)
```

❶ Importe les fonctions d'affichage et d'images de SimpleCV.

❷ Crée un nouvel objet de type fenêtre.

❸ Charge l'image à partir du fichier `raspberries.jpg` dans un objet de type image.

❹ Affiche l'image dans la fenêtre.

❺ Empêche le script de se terminer immédiatement.

Modifier une image

Maintenant que vous savez charger une image et l'afficher, vous allez découvrir comment la modifier avant affichage. Lors de cette opération, vous modifiez la copie de l'image en mémoire et non l'original dans le fichier.

1. Enregistrez le fichier `affiche-image.py` sous un nouveau nom, ici `superpose.py`.
2. Modifiez le fichier avec Leafpad en ajoutant les modifications de l'exemple 9-2.
3. Enregistrez le fichier et exécutez-le dans un terminal.
4. Vous devriez voir un rectangle et du texte se superposer à l'image originale.

Exemple 9-2. Code source du fichier superpose.py

```
from SimpleCV import Image, Display, DrawingLayer, Color ❶
from time import sleep

myDisplay = Display()

raspberryImage = Image("raspberries.jpg")

myDrawingLayer = DrawingLayer((raspberryImage.width,
     raspberryImage.height)) ❷

myDrawingLayer.rectangle((50,20), (250, 60), filled=True) ❸
myDrawingLayer.setFontSize(45)
myDrawingLayer.text("Raspberries!", (50, 20),
     color=Color.WHITE) ❹
raspberryImage.addDrawingLayer(myDrawingLayer) ❺
raspberryImage.applyLayers() ❻
raspberryImage.save(myDisplay)
while not myDisplay.isDone():
   sleep(0.1)
```

❶ En plus des fonctions de traitement d'images et d'affichage de l'exemple précédent, importe les fonctions de couleurs et de calques *(layers)* de SimpleCV.

❷ Crée un calque de la même dimension que l'image que nous venons de lire.

❸ Sur ce calque, dessine un rectangle plein dont les coins opposés sont aux coordonnées (50, 20) et (250, 60).

❹ Sur ce calque, écrit le message « Raspberries ! » en blanc et aux coordonnées (50, 20).

❺ Ajoute le calque `myDrawingLayer` à l'image `raspberryImage`.

❻ Fusionne les différents calques ajoutés à `raspberryImage` (figure 9-4).

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

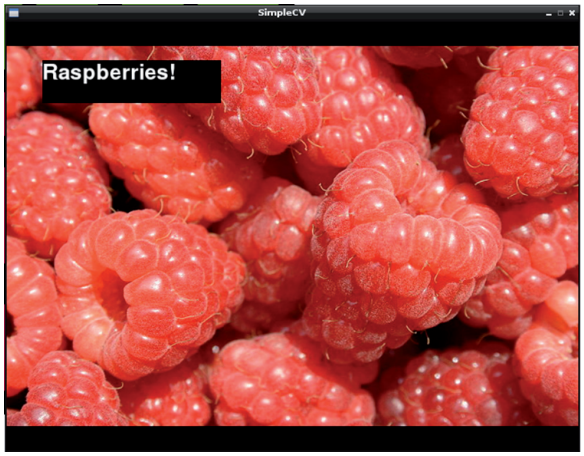


Figure 9-4. L'image des framboises a été modifiée.

L'exemple 9-3 montre comment enregistrer l'image après modification au lieu de l'afficher.

Exemple 9-3. Code source du fichier superpose-sauve.py

```
from SimpleCV import Image, DrawingLayer, Color
from time import sleep

raspberryImage = Image("raspberries.jpg")

myDrawingLayer = DrawingLayer((raspberryImage.width,
     raspberryImage.height))

myDrawingLayer.rectangle((50,20), (250, 60), filled=True)
myDrawingLayer.setFontSize(45)
myDrawingLayer.text("Raspberries!", (50, 20),
     color=Color.WHITE)

raspberryImage.addDrawingLayer(myDrawingLayer)
raspberryImage.applyLayers()

raspberryImage.save("raspberries-titled.jpg") ❶
```

❶ Enregistre l'image modifiée en mémoire dans un nouveau fichier nommé `raspberries-titled.jpg`.

Le code de l'exemple 9-3 n'ouvrant pas de fenêtre, vous pouvez l'exécuter dans un terminal sans mode graphique. Il est même possible d'adapter

le code pour modifier ou ajouter automatiquement votre logo à un lot d'images à l'aide d'une simple commande.

Par ailleurs, vous pouvez insérer d'autres éléments que du simple texte et des rectangles. Voici quelques-unes des fonctions que vous pouvez utiliser dans SimpleCV :

- cercle ;
- ellipse ;
- ligne ;
- polygone ;
- courbe de Bézier.

Pour consulter leur documentation complète, rendez-vous à la page suivante : http://simplecv.org/docs.

Accéder à la webcam

La lecture avec SimpleCV du flux vidéo provenant de la webcam n'est pas très différente de la lecture d'une image provenant d'un fichier. Comme vous allez le découvrir dans cette section, il est facile de réaliser une petite application semblable à `luvcview` à l'aide de quelques lignes de code Python seulement.

1. Créez un nouveau fichier nommé simpleWebcam.py. Éditez-le dans Leafpad en y ajoutant le code de l'exemple 9-4.
2. Après avoir branché et configuré votre webcam, exécutez le script. Vous devriez voir apparaître une nouvelle fenêtre dans l'environnement graphique, elle affiche le flux vidéo de la webcam (figure 9-5).
3. Pour fermer la fenêtre, appuyez sur les touches Ctrl + C dans un terminal.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Exemple 9-4. Ce code source du fichier simpleWebcam.py

```
from SimpleCV import Camera, Display
from time import sleep

myCamera = Camera(prop_set={'width': 320, 'height': 240}) ❶

myDisplay = Display(resolution=(320, 240)) ❷

while not myDisplay.isDone(): ❸
   myCamera.getImage().save(myDisplay) ❹
   sleep(.1) ❺
```

❶ Crée un nouvel objet de type `Camera` ayant pour dimensions 320 × 240 pixels pour une meilleure performance.

❷ Configure aussi la dimension de la fenêtre à 320 × 240 pixels.

❸ Boucle tant que la fenêtre n'est pas fermée.

❹ Lit une image de la webcam pour l'afficher dans la fenêtre.

❺ Attend un dixième de seconde avant de recommencer.

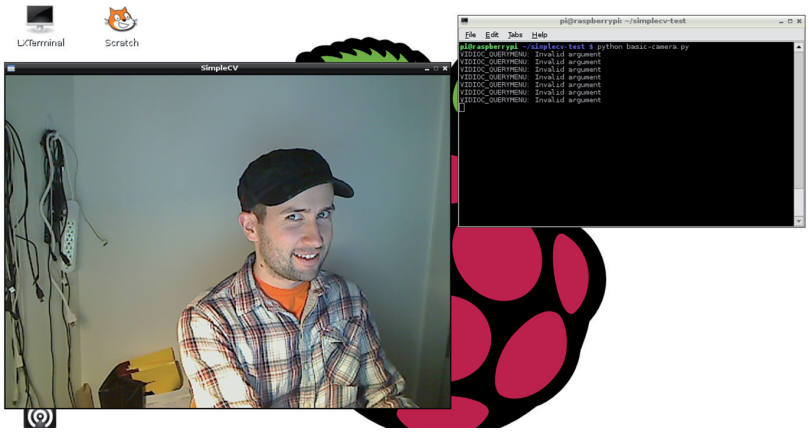


Figure 9-5. Redirection du flux vidéo de la webcam vers l'écran

Vous pouvez combiner le code des deux derniers exemples pour créer un script Python qui va acquérir une image de la webcam, puis l'écrire dans un fichier au format `.jpg` :

```
from SimpleCV import Camera
from time import sleep

myCamera = Camera(prop_set={'width':320, 'height': 240})

frame = myCamera.getImage()
frame.save("camera-output.jpg")
```

Reconnaissance de visage

La fonction `findHaarFeatures` est l'une des puissantes fonctions fournies par SimpleCV. Il s'agit d'un algorithme qui cherche dans une image des motifs correspondant à certains paramètres, appelés « cascades ». SimpleCV contient des cascades prédéfinies pour détecter des visages, des nez, des yeux, des bouches ou encore des corps entiers. Vous pouvez même télécharger ou générer vos propres fichiers de cascades si nécessaire.

La fonction `findHaarFeatures` analyse une image donnée et si elle trouve au moins une correspondance, elle retourne ses coordonnées dans l'image. Cela signifie que vous pouvez détecter et situer des objets tels que des voitures, des animaux ou des personnes dans des images provenant de fichiers ou d'une webcam.

Dans l'exemple suivant, nous allons programmer une simple détection de visage :

1. Créez un nouveau fichier nommé `visage.py` dans le répertoire `simplecv-test`.
2. Ouvrez-le dans un éditeur et saisissez le code de l'exemple 9-5.
3. Branchez la webcam, dirigez-la vers vous et lancez le script dans un terminal.
4. Les coordonnées des visages détectés par la fonction `findHaarFeatures` s'affichent dans la fenêtre. Si vous bougez votre tête, les coordonnées changent. Vous pouvez également tenir une photographie devant la webcam et observer ce qu'il se passe.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Exemple 9-5. Code source du fichier visage.py

```
from SimpleCV import Camera, Display
from time import sleep

myCamera = Camera(prop_set={'width':320, 'height': 240})

myDisplay = Display(resolution=(320, 240))

while not myDisplay.isDone():
    frame = myCamera.getImage()
    faces = frame.findHaarFeatures('face') ❶
    if faces: ❷
        for face in faces: ❸
            print "Visage à " + str(face.coordinates())
    else:
        print "Aucun visage trouvé."
    frame.save(myDisplay)
    sleep(.1)
```

❶ Cherche des visages dans l'image et écrit leurs caractéristiques dans l'objet `faces`.

❷ Si la fonction `findHaarFeatures` a détecté au moins un visage, elle exécute le code qui suit.

❸ Balaie l'objet `faces` et affiche les coordonnées de chaque visage détecté.

Si votre visage est bien dans le cadre de l'image mais que le message « Aucun visage trouvé » apparaît, essayez d'ajuster les paramètres.

- Y a-t-il assez de lumière ? Si vous êtes dans une pièce sombre, l'algorithme aura du mal à détecter les formes. Par conséquent, essayez de mieux éclairer la scène.

- Cette cascade particulière est destinée à détecter des visages de face, orientés normalement. Si vous penchez trop la tête ou si la webcam n'est pas bien horizontale, cela réduit l'efficacité de l'algorithme de Haar.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Projet : cabine photographique

En combinant différentes bibliothèques Python, il est possible de créer facilement des projets assez complexes. Nous allons maintenant réutiliser la bibliothèque GPIO (voir chapitre 8), en plus de tout ce que nous avons appris dans le présent chapitre, pour créer une cabine photographique un peu particulière (figure 9-6) et qui devrait faire fureur lors de vos prochaines soirées entre amis ! Grâce à la fonction `findHaarFeatures` de SimpleCV, vous allez pouvoir ajouter dynamiquement des objets sur les visages, comme des chapeaux, des monocles, des barbes ou des moustaches. Le code de ce projet est basé sur le projet « Mustacheinator », publié dans l'ouvrage *Practical Computer Vision with SimpleCV* de Kurt Demaagd, Anthony Oliver, Nathan Oostendorp et Katherine Scott.



Figure 9-6. La cabine photographique à Raspberry Pi ajoute une moustache à votre visage.

Voici ce dont vous aurez besoin pour ce projet :

- une webcam USB ;
- un écran ou un moniteur HDMI ;
- un bouton-poussoir ;
- du fil de connexion (de longueur adaptée) ;
- une résistance de 10 kΩ.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Avant de commencer, assurez-vous que les bibliothèques RPi.GPIO et SimpleCV sont bien installées et fonctionnent correctement, comme décrit à la section « Installation et test du module Python GPIO » page 130 et à la section « Installation et test de SimpleCV » page 148.

1. Connectez la broche 24 au bouton-poussoir. L'une de ses bornes doit être reliée au 3,3 V, l'autre à la broche 24 du Raspberry Pi. N'oubliez pas la résistance de rappel de 10K entre la broche 24 et la masse.
2. Téléchargez ou dessinez une moustache noire sur fond blanc et enregistrez cette image sous le nom `moustache.png` dans un nouveau répertoire dédié à ce projet. Vous pouvez utiliser notre fichier d'exemple, disponible dans le répertoire `ch09` de l'extension web de cet ouvrage (voir l'avant-propos page VII).
3. Dans ce nouveau répertoire, créez également un fichier nommé `cabinephoto.py`. Éditez-le en saisissant le code de l'exemple 9-6 et enregistrez-le.

Exemple 9-6. Code source du fichier cabinephoto.py

```
from time import sleep, time
from SimpleCV import Camera, Image, Display
import RPi.GPIO as GPIO

myCamera = Camera(prop_set={'width':320, 'height': 240})
myDisplay = Display(resolution=(320, 240))
stache = Image("mustache.png")
stacheMask = \
  stache.createBinaryMask(color1=(0,0,0),
                          color2=(254,254,254)) ❶
stacheMask = stacheMask.invert() ❷

GPIO.setmode(GPIO.BCM)
GPIO.setup(24, GPIO.IN)

def mustachify(frame): ❸
    faces = frame.findHaarFeatures('face')
    if faces:
        for face in faces:
            print "Visage à " + str(face.coordinates())
            myFace = face.crop() ❹
            noses = myFace.findHaarFeatures('nose')
            if noses:
                nose = noses.sortArea()[-1] ❺
                print "Nez à " + str(nose.coordinates())
```

```
                xmust = face.points[0][0] + noze.x -
    (stache.width/2) ❻
                ymust = face.points[0][1] + noze.y +
    (stache.height/3) ❼
            else:
                return frame ❽
        frame = frame.blit(stache, pos=(xmust, ymust),
     mask=stacheMask) ❾
        return frame ❿
    else:
        return frame ⓫

while not myDisplay.isDone():
    bouton = GPIO.input(24)
    frame = myCamera.getImage()
    if bouton == True:
        frame = mustachify(frame) ⓬
       frame.save("mustache-" + str(time()) + ".jpg") ⓭
        frame = frame.flipHorizontal() ⓮
        frame.show()
        sleep(3) ⓯
    else:
        frame = frame.flipHorizontal() ⓰
        frame.save(myDisplay)
    sleep(.05)
```

❶ Crée un masque à partir de la moustache, dont tous les pixels (sauf les noirs) deviendront transparents. Les deux paramètres, `color1` et `color2`, encadrent les valeurs des pixels à rendre transparents dans l'image de la moustache.

❷ Inverse le masque afin que seuls les pixels noirs de l'image soient affichés et que tous les autres pixels soient transparents.

❸ Crée une fonction qui prend une image en argument et renvoie une image avec la moustache superposée si un visage et un nez sont détectés.

❹ Crée une « sous-image » du visage pour que la recherche du nez soit plus rapide.

❺ Si plusieurs nez sont détectés, le plus grand est choisi.

❻ Définit la coordonnée x de la moustache.

❼ Définit la coordonnée y de la moustache.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

❽ Si aucun nez n'a été trouvé, renvoie simplement l'image originale.

❾ Combine l'image originale avec la moustache en utilisant la fonction `blit` *(BLock Image Transfer)* avec le masque.

❿ Renvoie l'image « moustachifiée ».

⓫ Si aucun visage n'a été détecté, renvoie simplement l'image originale.

⓬ Envoie l'image venant de la webcam à la fonction de « moustachification ».

⓭ Enregistre l'image transformée au format `.jpg` en indiquant la date et l'heure dans le nom du fichier.

⓮ Avant d'afficher l'image moustachifiée à l'écran, applique un effet de miroir horizontal.

⓯ Affiche l'image à l'écran pendant trois secondes.

⓰ Si le bouton n'est pas appuyé, affiche juste l'image renversée.

Vous êtes maintenant prêt pour effectuer un essai. Assurez-vous tout d'abord que la webcam est bien connectée. Ensuite, dans le terminal, allez dans le répertoire où se trouve l'image de la moustache et le fichier `photobooth.py`, puis lancez le script :

```
pi@raspberrypi ~ $ sudo photobooth.py
```

La vidéo de la webcam s'affichera à l'écran. Quand le bouton sera appuyé, le script cherchera les visages, y ajoutera une moustache et enregistrera le résultat. Toutes les images seront sauvegardées dans le même répertoire que le script.

Pour aller plus loin

- *Practical Computer Vision with SimpleCV*
  Grâce à ce livre de Kurt Demaagd, Anthony Oliver, Nathan Oostendorp et Katherine Scott, vous apprendrez à utiliser pleinement SimpleCV. Il contient de multiples exemples de code et projets pour exploiter tout le potentiel de la vision artificielle en Python.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

CHAPITRE 10

ACCÉDER À INTERNET AVEC PYTHON

Les développeurs et utilisateurs de Python partagent volontiers leurs connaissances sur les multiples forums de discussion présents sur la Toile. Ils rendent également disponibles de nombreux codes sources, que vous pourrez reprendre et adapter facilement à vos projets. Grâce à ces bibliothèques, vous pourrez notamment configurer la connexion Internet de votre Raspberry Pi pour obtenir les dernières prévisions météorologiques, envoyer un e-mail ou un SMS, suivre le fil d'actualité sur Twitter ou même créer un serveur web.

Dans ce chapitre, vous allez découvrir que ces bibliothèques peuvent transformer votre Raspberry Pi en un objet connecté et communicant. Vous verrez donc comment récupérer des informations d'un site web pour les diffuser via le Raspberry Pi.

Télécharger des données d'un serveur web

Lorsque vous cliquez sur un lien d'une page web ou que vous saisissez une URL dans la barre d'adresse de votre navigateur et validez, votre navigateur a le rôle d'un client. En effet, il établit une connexion avec le serveur demandé, puis il envoie une requête indiquant le document à télécharger. Le serveur répond alors en fournissant ledit document, qui peut être une page web. Bien sûr, ce modèle est très flexible et le client n'est pas obligatoirement un navigateur web. Il peut s'agir d'un système de messagerie, d'une application de prévisions météorologiques sur votre smartphone ou votre ordinateur, ou même d'un jeu qui envoie votre score sur un site de classement des joueurs.

Dans un premier temps, intéressons-nous aux projets qui utilisent le Raspberry Pi comme un client. Nous allons donc écrire un code qui ira chercher des informations sur différents serveurs web. Pour ce faire, vous devez tout d'abord installer une bibliothèque Python appelée Requests (« requêtes » en français), qui permet de se connecter à des serveurs en utilisant le protocole HTTP. Vous pouvez installer cette bibliothèque depuis un terminal en tapant la commande suivante :

```
pi@raspberrypi ~ $ sudo apt-get install python-requests
```

Pour vous assurer que la bibliothèque Requests est bien installée, importez-la dans l'interpréteur Python :

```
pi@raspberrypi ~ $ python
Python 2.7.3rc2 (default, May 6 2012, 20:02:25) [GCC
     4.6.3] on linux2
Type "help", "copyright", "credits" or "license" for more
     information.
>>> import requests
>>>
```


Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Si tout s'est correctement déroulé, vous ne devriez pas obtenir de message d'erreur. Vous pouvez alors poursuivre, par exemple en chargeant une page web. Pour cela, saisissez la commande suivante dans l'interpréteur Python :

```
>>> r = requests.get('http://www.google.com/')
>>>
```

Vous serez peut-être un peu déçu car en apparence, rien ne s'est produit. En réalité, toutes les informations de la page ont été placées dans l'objet `r` et ses sous-éléments. Pour lire le code de statut de la requête, tapez la commande suivante :

```
>>> r.status_code
200
```

Le code 200 indique que le serveur a pu traiter la requête sans rencontrer aucun problème. Pour consulter la liste des codes définis par le protocole HTTP, rendez-vous à la page suivante : http://fr.wikipedia.org/wiki/Liste_des_codes_HTTP. Le tableau 10-1 présente les codes les plus utilisés.

Tableau 10-1. Codes de statut HTTP les plus courants

Code	Description
200 *(OK)*	Requête traitée avec succès
301 *(Moved permanently)*	Document déplacé de façon permanente
302 *(Moved temporarily)*	Document déplacé de façon temporaire
307 *(Temporary redirect)*	La requête doit être redirigée temporairement vers l'URI spécifiée.
401 *(Unauthorized)*	Une authentification est nécessaire pour accéder à la ressource.
403 *(Forbidden)*	L'accès à la ressource est interdit.
404 *(Not found)*	Ressource non trouvée
500 *(Server error)*	Erreur interne du serveur

Pour visualiser les données renvoyées par le serveur, tapez la commande suivante :

```
>>> r.text
```

Cette commande affichera normalement le code HTML de la page web demandée :

```
u'<!doctype html><html itemscope="" itemtype="http://
     schema.org/WebPage"><head><meta itemprop="
     image" content="/images/google_favicon_128.png">
     <title>Google</title><script>(function(){
window.google={kEI:"4mIRUoGOBIm-0QWmx4GQDw",
     getEI:function(a){for(var b;a&&(!a.getAttribute||!
     (b=a.getAttribute("eid")));)a=a.parentNode;return
     b||google.kEI},https:function(){return"https:"==
     window.location.protocol},kEXPI:"17259,4000116,
     4002854, […]
```

Cette page web se présente sous la forme d'un gros bloc de texte, avec des parties lisibles en langage naturel, et d'autres parties moins lisibles, qui correspondent à du langage HTML ou JavaScript et qui seront décodées par le navigateur web.

Cependant, toutes les requêtes HTTP ne sont pas destinées à obtenir des pages à afficher dans un navigateur. Certaines contiennent seulement des données, sans aucune information sur leur affichage. Beaucoup de sites mettent à disposition du public ces protocoles de données, afin qu'il puisse lire ou envoyer des données à leurs serveurs sans passer par un navigateur web. Ces spécifications du protocole et du formatage des données, qui peuvent être publiques ou privées, sont appelées API *(Application Programming Interface)*. Elles permettent aux différents éléments d'un logiciel de communiquer entre eux et sont très utilisées pour envoyer des données d'un site web à un autre.

Par exemple, imaginons que vous vouliez réaliser un projet qui vous informera, avant de sortir de chez vous, qu'il va bientôt pleuvoir et qu'il faut prendre votre parapluie. Au lieu d'installer une station météorologique et d'apprendre à s'en servir et interpréter ses informations, vous pouvez obtenir les prévisions pour la journée en passant par l'API d'un site météorologique.

Charger les prévisions météo

Afin de déterminer s'il va pleuvoir ou non aujourd'hui, voyons à présent comment utiliser l'API du site Weather Underground (http://www.wunderground.com).

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Gardez à l'esprit que toutes les API ne se valent pas et qu'il est recommandé de consulter leur documentation afin de choisir l'API la mieux adaptée à votre projet. De plus, la plupart des API limitent le nombre de requêtes que vous pouvez exécuter et certaines d'entre elles font même payer l'accès à leurs services. Mais en règle générale, si vous effectuez peu de requêtes, l'accès sera gratuit, ce qui permet d'expérimenter avec des projets amateurs.

1. Dans la barre d'adresse de votre navigateur, saisissez l'URL de la page de documentation de l'API de Weather Underground (http://www.wunderground.com/weather/api).
2. Cliquez sur le bouton Sign Up for FREE!, complétez les informations demandées pour ouvrir un compte et cliquez sur le bouton Sign Up.
3. Un e-mail de confirmation de demande d'inscription vous est alors adressé. Validez votre inscription en cliquant sur le lien contenu dans cet e-mail.
4. Saisissez votre login et mot de passe précédemment définis, puis cliquez sur Login afin d'accéder à votre compte.
5. Cliquez sur l'onglet Pricing, puis sur la case à cocher Developer en bas de la page. Appuyez ensuite sur le bouton Purchase Key.
6. Renseignez les informations demandées afin d'accéder enfin à la page Get Your API Key.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

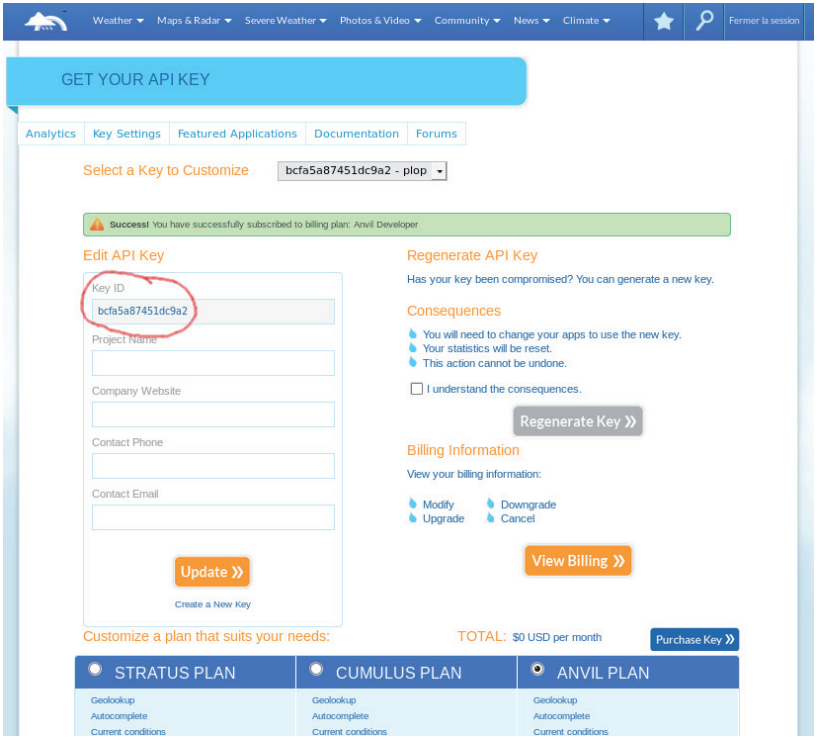


Figure 10-1. Capture d'écran du site Weather Underground montrant la clé générée pour un nouveau compte.

7. Le champ Key ID (voir figure 10-1) contient une longue chaîne de caractères. Il s'agit de la clé, c'est-à-dire un identifiant numérique et unique, associée à votre compte et que vous devrez fournir à chaque requête.



Notez que si vous effectuez des requêtes abusives ou si vous en envoyez trop, le site peut désactiver la clé et refuser de vous servir, à moins que vous achetiez des services supplémentaires. La version gratuite de Weather Underground permet jusqu'à 500 requêtes par jour, ou 10 par minute au maximum.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

8. Cliquez sur Documentation, puis sur Forecast (http://www.wunderground.com/weather/api/d/docs?d=data/forecast) pour afficher toutes les informations qu'il est possible d'obtenir avec une requête. En bas de la page, vous trouverez l'exemple d'un lien effectuant une requête sur le temps qu'il fait à San Francisco, en Californie. Vous remarquerez que le lien contient la clé API, soit `bcfa5a87451dc9a2` pour notre exemple (http://api.wunderground.com/api/bcfa5a87451dc9a2/forecast/q/CA/San_Francisco.json).
9. Pour vérifier que votre compte fonctionne correctement, saisissez cette URL dans la barre d'adresse de votre navigateur. Vous devriez alors voir apparaître des données sous forme de texte au format JSON. Dans l'exemple 10-1 suivant, remarquez comment les données sont structurées hiérarchiquement.



Le format JSON *(JavaScript Object Notation)* est conçu pour être portable et facilement lisible par de nombreux langages (pas seulement JavaScript). Ceci permet la communication entre différentes applications.

Pour plus de renseignements, consultez la page suivante : http://www.json.org/json-fr.html.

10. Triez à présent les données. Dans cette structure, essayez d'identifier et d'obtenir le texte correspondant à la prévision. Modifiez l'URL en y ajoutant le nom de votre ville, puis enregistrez-la dans un nouveau script Python intitulé `texte-prevision.py` (exemple 10-2).
11. Comme vous pouvez le voir sur l'exemple 10-1, pour obtenir le texte de la prévision pour la journée, vous devez aller le chercher dans le champ approprié de la structure de données hiérarchisée. Pour cela, suivez le chemin `forecast>txt_forecast>forecastday>0` (le premier élément, qui correspond à la prévision du jour)`>fcttext`.
12. Si vous lancez le script de l'exemple 10-2 en ligne de commande, la prévision du jour (en anglais) devrait alors s'afficher à l'écran.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Exemple 10-1. Extrait de la réponse envoyée au format JSON par l'API de Weather Underground

```
"response": {
  "version": "0.1",
 "termsofService": "http://www.wunderground.com/weather/
     api/d/terms.html",
  "features": {
    "forecast": 1
  }
},
"forecast":{ ❶
  "txt_forecast": { ❷
    "date":"8:00 PM PDT",
    "forecastday": [ ❸
      {
      "period":0,
      "icon":"partlycloudy",
      "icon_url":"http://icons-ak.wxug.com/i/c/k/partly
     cloudy.gif",
      "title":"Sunday",
      "fcttext":"Partly cloudy. Fog early. High of 75F.
     Winds from the WSW at 5 to 15 mph.", ❹
      "fcttext_metric":"Partly cloudy. Fog early. High of
     24C. Windy. Winds from the WSW at 5 to 25 km/h.",
      "pop":"0"
      },
```

❶ `forecast` est le parent de niveau supérieur de la donnée que nous voulons accéder.

❷ Dans l'élément `forecast`, on trouve les textes dans le sous-élément `txt_forecast`.

❸ Parmi les textes, nous désirons ceux qui concernent la journée en cours.

❹ `fcttext` est le texte qui décrit la prévision pour la journée.

Exemple 10-2. Code source du fichier texte-prevision.py

```
import requests
key = 'bcfa5a87451dc9a2' ❶
ApiUrl = 'http://api.wunderground.com/api/' + key + '/
     forecast/q/FR/Paris.json'

r = requests.get(ApiUrl) ❷
forecast = r.json ❸
print forecast['forecast']['txt_forecast']['fore
```

```
    castday'][0]['fcttext'] ❹
```

❶ Remplacez cette clé API par la vôtre.

❷ Envoie la requête à Weather Underground (n'oubliez pas de modifier le nom du pays et de la ville).

❸ Transforme le texte de la réponse au format JSON en un objet Python.

❹ Explore l'objet pour obtenir et afficher la chaîne de caractères désirée.



NdT : consultez la documentation en ligne pour déterminer le code que Weather Underground a assigné à votre pays. En effet, ce site n'utilise pas les codes ISO internationaux, ce qui lui permet de pouvoir coder en plus les 50 états que comptent les États-Unis.

Par exemple, le code de la France est bien **`Fr`** mais la Belgique est codée **`BX`** au lieu de **`BE`**. Les URL correspondant à Paris et Bruxelles sont alors : http://api.wunderground.com/api/bcfa5a87451dc9a2/forecast/q/FR/Paris.json et http://api.wunderground.com/api/bcfa5a87451dc9a2/forecast/q/BX/Bruxelles.json.

Par ailleurs, le nom de votre ville peut s'écrire différemment en anglais. Ainsi, Paris ne change pas, contrairement à Bruxelles qui devient Brussels.

Vous disposez maintenant d'un script Python qui obtient les prévisions météorologiques à la demande. Mais comment votre Raspberry Pi pourra-t-il déterminer s'il va pleuvoir ou non ? Une première piste serait d'analyser le texte de la prévision à la recherche de mots-clés tels que « rain », « drizzle », « thunderstorms », etc. Mais il existe une solution beaucoup plus simple et fiable. En effet, parmi les données fournies par l'API de Weather Underground, nous trouvons un nombre appelé « pop », pour « probabilité de précipitation ». Sa valeur, comprise entre 0 et 100 %, est une estimation plus facile à traiter avec un programme simple.

Pour notre projet, nous décidons qu'il faudra sortir avec un parapluie si la probabilité de précipitation est supérieure à 30 % pour la journée.

1. Connectez une LED à la broche 25 (voir figure 7-4, page 114).

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

2. Créez un nouveau fichier nommé `PrendsTonParapluie.py` et éditez-le en y saisissant le code de l'exemple 10-3. N'oubliez pas de modifier la clé API et votre ville et/ou pays dans l'URL de Weather Underground.
3. Lancez le script avec les droits `root` en tapant la commande `sudo python PrendsTonParapluie.py`.

Exemple 10-3. Code source du fichier PrendsTonParapluie.py

```
import requests
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
GPIO.setup(25, GPIO.OUT)
key = 'bcfa5a87451dc9a2' ❶
ApiUrl = 'http://api.wunderground.com/api/' + key + '/
     forecast/q/FR/Paris.json'

while True:
    r = requests.get(ApiUrl)
    forecast = r.json
    popValue = forecast['forecast']['txt_forecast']
     ['forecastday'][0]['pop'] ❷
    popValue = int(popValue) ❸
    if popValue >= 30: ❹
        GPIO.output(25, GPIO.HIGH)
    else: ❺
        GPIO.output(25, GPIO.LOW)
    time.sleep(180) # 3 minutes ❻
```

❶ Modifiez cette chaîne de caractères en la remplaçant par votre propre clé API.

❷ Place la probabilité de précipitation dans la variable `popValue`.

❸ Convertit la chaîne de caractères en nombre entier pour faciliter la comparaison.

❹ Si la valeur est supérieure à 30, allume la LED.

❺ Sinon, éteint la LED.

❸ Attend trois minutes entre les requêtes pour ne pas épuiser le quota de l'API. Cela fait 480 requêtes par jour ((24 × 60) / 3), juste un peu moins que les 500 requêtes permises par le compte gratuit de Weather Underground.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Utilisez le raccourci clavier Ctrl + C pour quitter le programme.

Weather Underground est un des nombreux sites qui permettent d'accéder à des données sans navigateur, en passant par une API. Le tableau 10-2 liste d'autres sites populaires qui fournissent une API.

Tableau 10-2. Interfaces de programmation d'applications populaires

Site	Documentation de l'API
Facebook	https://developers.facebook.com
Flickr	http://www.flickr.com/services/api
Foursquare	https://developer.foursquare.com
Reddit	https://github.com/reddit/reddit/wiki/API
Twilio	http://www.twilio.com
Twitter	https://dev.twitter.com
YouTube	https://developers.google.com/youtube

Le Raspberry Pi comme serveur web

Votre Raspberry Pi est capable de chercher des données sur Internet, mais également d'en proposer si vous le transformez en serveur web. Pour ce faire, vous pouvez utiliser l'un des nombreux logiciels conçus à cet effet, précompilés pour Linux et disponibles au moyen de l'outil `apt-get`. Les serveurs traditionnels comme Apache et lighttpd gèrent surtout des fichiers préexistants dans le système (fichiers HTML, images, sons, vidéos, exécutables, etc.) pour constituer des pages web.

Il existe aussi une autre catégorie d'outils qui ajoutent des fonctionnalités aux langages dynamiques tels que Python, Ruby, Perl ou JavaScript. Ces extensions gèrent la connexion au réseau et permettent aux scripts de générer automatiquement des pages HTML, à chaque fois qu'ils reçoivent une requête d'un navigateur web. Ce moyen est idéal pour déclencher des actions physiques, stocker des données ou lire à distance la valeur d'un capteur avec votre navigateur. Vous pouvez même créer votre propre API au format JSON pour vos projets d'électronique !

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

LES BASES DE FLASK

Nous allons maintenant faire appel au framework Flask, qui va faire de votre Raspberry Pi un serveur web dynamique. Ce framework propose de nombreuses fonctionnalités et même des extensions spécialisées (par exemple pour authentifier des utilisateurs, générer des formulaires ou encore accéder à des bases de données). Bien entendu, vous pouvez parfaitement combiner les fonctionnalités de Flask avec celles des autres bibliothèques de Python.

Pour installer Flask, l'outil `pip` doit être installé sur votre Raspberry Pi. Dans le cas contraire, utilisez la commande `apt-get` :

```
pi@raspberrypi ~ $ sudo apt-get install python-pip
```

Une fois `pip` installé, utilisez-le pour charger Flask et ses dépendances au moyen de la commande suivante :

```
pi@raspberrypi ~ $ sudo pip install flask
```

Pour vérifier que tout fonctionne correctement, nous allons écrire un petit programme qui fait office de serveur web minimaliste. Pour cela, nous créons un fichier nommé `hello-flask.py` contenant le code de l'exemple 10-4. Ce code semble un peu abscons mais ne vous inquiétez pas, nous en examinerons chaque ligne par la suite. Pour l'instant, intéressons-nous à la ligne qui contient la chaîne de caractères `"Hello World!"`.

Exemple 10-4. Code source du fichier hello-flask.py

```
from flask import Flask
app = Flask(__name__) ❶

@app.route("/") ❷
def hello():
   return "Hello World!" ❸

if __name__ == "__main__":❹
   app.run(host='0.0.0.0', port=80, debug=True) ❺
```

❶ Crée un objet de type Flask appelé `app`.

❷ Exécute le code qui suit à chaque accès de l'adresse racine du serveur.

❸ Renvoie le texte `"Hello World!"` au client.

❹ Si le script est lancé directement de la ligne de commande,

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

❺ alors lance le serveur pour qu'il écoute le port 80 (celui du protocole HTTP) et affiche toutes les erreurs sur le terminal.



Pour pouvoir lancer ce script, vous aurez besoin de connaître l'adresse IP de votre Raspberry Pi (voir la section « Le réseau » page 38). Vous pouvez l'obtenir en lançant **`ifconfig`** en ligne de commande ou en lisant les derniers messages affichés au démarrage de Linux. Une autre solution consiste à installer **`avahi-daemon`** (tapez **`sudo apt-get install avahi-daemon`** en ligne de commande), qui vous permet d'accéder à votre Raspberry Pi sur votre réseau local à l'adresse http://raspberry.pi.local.

Si vous accédez au serveur web du Raspberry Pi à partir d'un ordinateur sous Windows, essayez d'y installer Bonjour Services.

Le serveur est maintenant prêt, il ne reste plus qu'à le lancer avec les droits de `root` :

```
pi@raspberrypi ~ $ sudo python hello-flask.py
 * Running on http://0.0.0.0:80/
 * Restarting with reloader
```

Depuis un autre ordinateur connecté au même réseau local que le Raspberry Pi, saisissez l'adresse IP du serveur dans la barre d'adresse du navigateur web. Si ce dernier affiche « Hello World! », tout fonctionne correctement. Simultanément, vous verrez apparaître quelques lignes sur le terminal du Raspberry Pi :

```
10.0.1.100 - - [19/Nov/2012 00:31:31] "GET / HTTP/1.1"
     200 -
10.0.1.100 - - [19/Nov/2012 00:31:31] "GET /favicon.ico
     HTTP/1.1" 404 -
```

La première ligne indique qu'un navigateur, à l'adresse 10.0.1.100, a demandé le document à la racine du serveur et que ce dernier l'a trouvé et renvoyé avec le code 200, qui indique que tout est en ordre. La seconde ligne montre une requête que la plupart des navigateurs web envoient pour afficher le favicon du site dans la barre d'adresse. Notre serveur

ne disposant pas d'un fichier `favicon.ico`, il retourne le code 404, qui indique que l'URL n'existe pas.

Si vous voulez renvoyer des pages HTML complètes, inutile de mettre l'ensemble dans le code source du script Python. À cet effet, le framework Flask dispose de Jinja2, un système de templates qui permet d'utiliser des fichiers externes qui font office de gabarits fixes et parsemés de symboles qui seront remplacés par des données dynamiques.

Si le fichier `hello-flask.py` tourne encore, arrêtez-le en appuyant sur les touches Ctrl + C.

Nous allons maintenant tester Jinja2 à travers un petit exemple. Créez d'abord le fichier `hello-template.py` qui contient le code de l'exemple 10-5. Dans le même répertoire que ce fichier, créez ensuite un sous-répertoire nommé `templates` et ajoutez-y un nouveau fichier `main.html` contenant le code de l'exemple 10-6. C'est principalement du code HTML, sauf que la variable `time` située dans les doubles accolades sera ensuite remplacée par la fonction `render_template`.

Exemple 10-5. Code source du fichier hello-template.py

```
from flask import Flask, render_template
import datetime
app = Flask(__name__)

@app.route("/")
def hello():
    now = datetime.datetime.now() ❶
    timeString = now.strftime("%Y-%m-%d %H:%M") ❷
    templateData = {
        'title' : 'HELLO!',
        'time': timeString
        } ❸
   return render_template('main.html', **templateData) ❹

if __name__ == "__main__":
   app.run(host='0.0.0.0', port=80, debug=True)
```

❶ Place la date et l'heure actuelles dans la variable `now`.

❷ Crée une chaîne de caractères pour représenter la date stockée dans la variable `now`.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

❸ Crée un dictionnaire de variables (un ensemble de clés à des valeurs de valeurs, par exemple, `title` correspond à `HELLO!`) pour l'envoyer au moteur de templates.

❹ Renvoie la page `main.html` en substituant les variables contenues dans le dictionnaire `templateData`.

Exemple 10-6. Code source du fichier templates/main.html

```
<!DOCTYPE html>
    <head>
       <title>{{title}}</title> ❶     </head>
    <body>
       <h1>Hello, World!</h1>
      <h2>La date et l'heure du serveur sont : {{time}}</
     h2> ❷
    </body>
</html>
```

❶ Utilise la variable `title` comme titre de la page HTML.

❷ Insère la variable `time` dans la page.

Il ne reste plus qu'à lancer le script `hello-template.py` avec `sudo`, toujours avec les droits `root`. Saisissez alors l'adresse de votre Raspberry Pi dans la barre d'adresse de votre navigateur. Vous devriez voir apparaître une page HTML ayant pour titre « HELLO! » et affichant la date et l'heure de votre nouveau serveur.



Même si cela dépend de la configuration de votre réseau local, il est peu probable que cette page soit accessible en dehors de ce réseau sur Internet. Si vous souhaitez la rendre publique, vous devrez configurer votre routeur pour qu'il laisse passer certains ports. Consultez la documentation de votre routeur pour plus d'informations à ce sujet.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

CONNECTER LE WEB AU MONDE RÉEL

Vous pouvez combiner Flask avec d'autres bibliothèques Python pour étendre les fonctionnalités de votre site web. Par exemple, en utilisant le module RPi.GPIO (voir chapitre 8), vous pouvez créer un site web qui s'interface au monde physique. Nous allons voir comment faire cela en modifiant le code source précédent, mais pour commencer, connectez trois boutons-poussoir ou interrupteurs aux broches 23, 24 et 25 en suivant les instructions de la boîte à bruitages (voir figure 8-2 page 140).

Le code qui suit est basé sur celui du fichier `hello-template.py`. Reprenez donc le code de ce fichier dans un nouveau fichier nommé `hello-gpio.py`. Ajoutez le module RPi.GPIO ainsi qu'une nouvelle route pour lire l'état des entrées. Cette route va reconnaître une nouvelle URL et en extraire une valeur qui sera placée dans une variable.

Il faudra aussi créer un nouveau template, nommé ici `pin.html`. Celui-ci étant identique à `main.html`, recopiez ce dernier dans le fichier `pin.html` et modifiez-le pour obtenir le code de l'exemple 10-8.

Exemple 10-7. Code source du fichier hello-gpio.py

```
from flask import Flask, render_template
import datetime
import RPi.GPIO as GPIO
app = Flask(__name__)

GPIO.setmode(GPIO.BCM)

@app.route("/")
def hello():
    now = datetime.datetime.now()
    timeString = now.strftime("%Y-%m-%d %H:%M")
    templateData = {
        'title' : 'HELLO!',
        'time': timeString
        }
    return render_template('main.html', **templateData)

@app.route("/readPin/<pin>") ❶
def readPin(pin):
    try: ❷
        GPIO.setup(int(pin), GPIO.IN) ❸
        if GPIO.input(int(pin)) == True: ❹
            reponse = "La broche " + pin + " est à l'état
     bas"
```

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

```
        else: ❺
            reponse = "La broche " + pin + " est à l'état
     haut"
    except: ❻
        reponse = "Erreur lors de la lecture de la broche
     " + pin + "."

    templateData = {
        'title' : 'État de la broche ' + pin,
        'response' : reponse
        }
    return render_template('pin.html', **templateData)

if __name__ == "__main__":
    app.run(host='0.0.0.0', port=80, debug=True)
```

❶ Ajoute une route dynamique qui crée la nouvelle variable `pin` à partir de l'URL.

❷ Si le code de ce bloc déclenche une erreur, exécute le bloc `except`.

❸ Récupère le numéro de la broche dans l'URL, convertit la chaîne en nombre entier et configure la broche en entrée.

❹ Si la broche est à l'état haut, envoie le texte correspondant.

❺ Sinon, envoie le texte indiquant que la broche est à l'état bas.

❻ En cas d'erreur, envoie un message spécial.

Exemple 10-8. Code source du fichier templates/pin.html

```
<!DOCTYPE html>
   <head>
      <title>{{title}}</title> ❶
   </head>
   <body>
      <h1>État de la broche</h1>
      <h2>{{response}}</h2> ❷
   </body>
</html>
```

❶ Remplace la chaîne `{{title}}` par le titre fourni par la variable `title`.

❷ Insère la réponse générée par `hello-gpio.py` dans cette balise HTML.

Lorsque vous lancez ce script modifié, la même page que celle obtenue précédemment devrait s'ouvrir dans votre navigateur. Elle affiche le

message « Hello World ! » du premier exemple de template. Mais si vous ajoutez `/readPin/24` à la fin de l'URL, afin de créer une adresse complète qui ressemble à http://10.0.1.103/readPin/24, une nouvelle page apparaît. Elle indique que la broche 24 est à l'état bas. Maintenez enfoncé le bouton connecté à la broche 24 et rechargez la page. Elle mentionne maintenant que la broche est à l'état haut !

Essayez les autres boutons en changeant le nombre final dans l'URL. Cette technique présente l'avantage de n'avoir à écrire qu'une seule fonction qui fonctionne pour toutes les broches. De plus, elle crée une multitude de pages indépendantes, comme s'il y avait une page par broche.

Projet : une lampe contrôlée par Internet

Grâce au projet du chapitre 7, vous avez pu constater que votre Raspberry Pi peut servir de séquenceur pour contrôler une lampe branchée sur secteur (voir la section « Projet : séquenceur de lampe avec cron » page 121). Maintenant que vous savez utiliser Python et Flask, vous allez découvrir comment contrôler cette lampe en utilisant Internet. Le projet suivant vous servira de base et de référence pour tous les objets que vous souhaiterez connecter au Web via le Raspberry Pi.

Le code précédent permet de gérer plusieurs broches à l'aide d'une seule fonction. Pour ce projet, nous allons le modifier quelque peu de manière à ajouter autant de sorties que nécessaire pour contrôler plusieurs appareils et les configurer selon nos besoins.

Le matériel nécessaire est le même que celui du projet de séquenceur de lampe (voir page 121). Reportez-vous à ce précédent projet pour un rappel des pièces, des composants et des connexions.

1. Connectez le relais PowerSwitch Tail II à la broche 25, comme dans le projet de séquenceur de lampe.
2. Si vous avez un autre PowerSwitch Tail II, connectez-le à la broche 24 pour contrôler un autre appareil. Si vous n'en avez pas, connectez une LED.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28



Nous allons simplement montrer ici comment plusieurs appareils peuvent être contrôlés par le même code.

3. Dans votre répertoire personnel, créez un nouveau répertoire nommé `WebLamp`.
4. Dans ce nouveau répertoire, créez le fichier `weblamp.py` et éditez-le en y insérant le code de l'exemple 10-9.
5. Dans le répertoire `WebLamp`, créez ensuite un sous-répertoire `templates`.
6. Dans ce sous-répertoire `templates`, créez le fichier `main.html` et éditez-le en y insérant le code de l'exemple 10-10.

Si d'autres programmes serveur sont en cours d'exécution, ouvrez un terminal et appuyez sur les touches Ctrl + C pour les arrêter. Allez ensuite dans le répertoire `WebLamp` pour démarrer le nouveau serveur.

```
pi@raspberrypi ~/WebLamp $ sudo python weblamp.py
```

Exemple 10-9. Code source du fichier weblamp.py

```
import RPi.GPIO as GPIO
from flask import Flask, render_template, request
app = Flask(__name__)

GPIO.setmode(GPIO.BCM)

broches = {
    24 : {'name' : 'coffee maker', 'state' : GPIO.LOW},
    25 : {'name' : 'lamp',         'state' : GPIO.LOW}
    } ❶

for pin in pins: ❷
    GPIO.setup(pin, GPIO.OUT)
    GPIO.output(pin, GPIO.LOW)

@app.route("/")
def main():
    for pin in pins:
        pins[pin]['state'] = GPIO.input(pin) ❸
```

```
    templateData = {
        'pins' : pins ❹
        }
    return render_template('main.html', **templateData) ❺

@app.route("/<changePin>/<action>") ❻
def action(changePin, action):
    changePin = int(changePin) ❼
    deviceName = pins[changePin]['nom'] ❽
    if action == "on": ❾
        GPIO.output(changePin, GPIO.HIGH) ❿        message
     = deviceName + " allumée." ⓫
    if action == "off":
        GPIO.output(changePin, GPIO.LOW)
        message = deviceName + " éteinte."
    if action == "toggle":
       GPIO.output(changePin, not GPIO.input(changePin)) ⓬
        message = deviceName + " modifiée."

    for pin in pins:
        pins[pin]['etate'] = GPIO.input(pin) ⓭

    templateData = {
        'message' : message,
        'pins' : pins
    } ⓮

    return render_template('main.html', **templateData)

if __name__ == "__main__":
    app.run(host='0.0.0.0', port=80, debug=True)
```

❶ Crée un dictionnaire `pins` pour mémoriser le numéro de la broche, son nom et son état.

❷ Configure chaque broche en sortie et à l'état bas.

❸ Lit l'état de toutes les broches pour mettre à jour les valeurs du dictionnaire.

❹ La liste des broches est fournie au template.

❺ Traite le template `main.html` avec la liste des broches et envoie la page modifiée au navigateur.

❻ La fonction qui suit est exécutée chaque fois qu'une URL contient un nombre et une action.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

❼ Convertit le numéro de broche dans l'URL en un nombre entier.

❽ Consulte le dictionnaire pour obtenir le nom de la broche correspondante.

❾ Si l'action à effectuer est on, alors exécute le code qui suit.

❿ Met la broche à l'état haut.

⓫ Compose le message d'état à insérer dans le template.

⓬ Change l'état de la broche en écrivant le contraire de la valeur initiale.

⓭ Lit l'état de toutes les broches pour mettre à jour le dictionnaire.

⓮ Envoie le numéro de broche ainsi que le message au template.

Exemple 10-10. Code source du fichier templates/main.html

```
<!DOCTYPE html>
<head>
   <title>État des appareils</title>
</head>
<body>
   <h1>Liste des appareils et de leurs états</h1>
   {% for pin in pins %} ❶
   <p>La {{pins[pin].name}} ❷
   {% if pins[pin].state == true %} ❸
      est actuellement allumée. (<a href="/{{pin}}/
     off">Éteindre</a>)
   {% else %} ❹
      est actuellement éteinte. (<a href="/{{pin}}/
     on">Allumer</a>)
   {% endif %}
   </p>
   {% endfor %}
   {% if message %} ❺
   <h2>{{message}}</h2>
   {% endif %}
</body>
</html>
```

❶ Examine toutes les broches du dictionnaire.

❷ Affiche le nom de la broche.

❸ Si la broche est à l'état haut, affiche le nom de l'appareil et crée un lien pour l'éteindre.

❹ Sinon, indique que l'appareil est éteint et affiche un lien pour l'allumer.

❺ Affiche un éventuel message fourni au template.



Figure 10-2. L'interface web des appareils, telle qu'elle apparaît dans le navigateur d'un téléphone portable.

Ce code est écrit de telle manière qu'il est très facile d'ajouter d'autres appareils supportés par le Raspberry Pi. Pour cela, il vous suffit d'ajouter l'information relative à cet appareil dans le dictionnaire des broches, au début du code. Lorsque vous relancerez le serveur, le nouvel appareil apparaîtra dans la liste des états et les URL de contrôle fonctionneront automatiquement.

Autre fonctionnalité très utile : si vous souhaitez pouvoir changer l'état d'un appareil par un simple toucher sur votre téléphone, il suffit d'ajouter dans vos signets l'adresse http://adresseIP/numérobroche/toggle. Cette URL vérifiera l'état actuel de la broche et le fera basculer.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Pour aller plus loin

- Requests
  La page du projet Requests (http://docs.python-requests.org/en/latest) propose une documentation très complète, agrémentée de nombreux exemples faciles à suivre.

- Flask
  Les possibilités de Flask (http://flask.pocoo.org) sont très vastes et nous n'en avons présenté ici qu'un rapide aperçu. Consultez son site web pour connaître toutes les fonctions disponibles.

- Extensions de Flask
  Les extensions de Flask (http://flask.pocoo.org/extensions) vous permettront d'ajouter facilement des fonctionnalités puissantes à votre site.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

ANNEXE A

ENREGISTRER UNE IMAGE DU SYSTÈME D'EXPLOITATION SUR UNE CARTE SD

Dans ce livre, nous avons principalement utilisé la distribution Raspbian. Sachez toutefois que le Raspberry Pi fonctionne également très bien sous d'autres systèmes d'exploitation. Leurs caractéristiques et fonctionnalités sont très diverses, mais tous présentent néanmoins un point commun : il faut copier leur image sur une carte SD après l'avoir téléchargée sur le Web.

Dans cette annexe, nous allons vous présenter la marche à suivre pour Mac OS X, Windows et Linux.

Écrire une carte SD à partir de Mac OS X

1. Ouvrez l'utilitaire de terminal (dans `/Applications/Utilitaires`) pour accéder à la ligne de commande.
2. Saisissez la commande `df -h`. Le programme `df` affiche l'espace libre de toutes les partitions montées.
3. Insérez la carte SD dans son lecteur, attendez quelques secondes et relancez la commande `df -h`.
4. Examinez la liste des partitions afin de trouver celle qui a été assignée à la carte SD. Pour ce faire, comparez la nouvelle liste avec l'ancienne de manière à détecter la ligne qui a été ajoutée. Pour notre exemple, la carte SD est montée sur le répertoire `/Volumes/Untitled` et le nom de la partition est `/dev/disk3s1`. Les noms sont assignés dynamique-

ment et peuvent changer d'une insertion à une autre. Par conséquent des nombres différents peuvent apparaître si d'autres périphériques sont disponibles dans le Finder. Notez le nom de la partition.

5. Lors de l'insertion de la carte SD, l'ordinateur a automatiquement « monté » cette dernière pour examiner ses fichiers. Nous allons maintenant changer radicalement le système de fichiers, il faut donc « démonter » la carte SD avant de pouvoir installer le nouveau système d'exploitation. Pour ce faire, utilisez la commande suivante : `sudo diskutil unmount /dev/disk3s1` (remplacez les derniers chiffres par ceux que vous avez notés précédemment). Vous pouvez également utiliser les utilitaires de disque pour démonter la carte. Si vous l'éjectez avec le Finder, vous devrez la retirer, la réinsérer, puis exécuter à nouveau toutes les commandes depuis le début. Si la commande de démontage échoue, vérifiez qu'aucun logiciel (dont le Finder) n'explore la carte.
6. Vous devez maintenant déterminer le nom du périphérique correspondant à la carte SD. Prenez le nom de la partition, remplacez `disk` par `rdisk`, et enlevez le `s1` à la fin (qui correspond au numéro de la partition). Par exemple, notre partition `/dev/disk3s1` se trouve sur le périphérique `/dev/rdisk3`.



Faites bien attention de ne pas vous tromper dans le nom du périphérique ! En effet, votre système d'exploitation pourrait être effacé si le nom est incorrect et vous perdrez vos données si vous commencez à écrire dessus. Vérifiez encore une fois avec `df` avant de continuer.

7. Décompressez l'image du système d'exploitation, puis utilisez la commande `dd` pour la recopier bit par bit sur la carte SD avec la commande suivante. Remplacez le nom du fichier de l'image par celui que vous avez téléchargé et substituez `/dev/rdisk3` avec le nom du périphérique déterminé dans l'étape 6.

   Le chapitre 3 présente les outils de base de la ligne de commande. La ligne suivante indique de lancer le programme `dd` avec les droits de

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

l'administrateur (`root`), pour qu'il puisse recopier le fichier source (`if`) dans le fichier destination (`of`) par blocs d'un mégaoctet (`bs=1m`).

```
sudo dd bs=1m if=2013-07-26-wheezy-raspbian.img of=/dev/rdisk3
```

8. La copie va prendre plusieurs minutes. Malheureusement, la commande `dd` ne fournit pas d'information sur la progression donc vous devrez patienter. Une fois la copie terminée, la commande `dd` affichera quelques statistiques. Éjectez alors la carte SD de son lecteur et insérez-la dans le Raspberry Pi. Votre système est désormais prêt à démarrer.

Écrire une carte SD à partir de Windows

1. Téléchargez et installez le programme Win32DiskImager (https://launchpad.net/win32-image-writer).
2. Insérez la carte SD dans son lecteur et notez la lettre du lecteur qui lui a été assignée par l'explorateur de fichiers.
3. Lancez le programme Win32DiskImager et sélectionnez le fichier contenant l'image à installer.
4. Choisissez la lettre correspondant à votre carte SD, puis cliquez sur Write. Si Win32DiskImager n'arrive pas à écrire sur la carte, essayez de la reformater avec l'explorateur de fichiers.
5. Une fois la copie terminée, éjectez la carte et insérez-la dans votre Raspberry Pi.

Écrire une carte SD à partir de Linux

La procédure est très similaire à celle décrite pour Mac OS X car ces deux systèmes d'exploitation sont compatibles Unix.

1. Lancez un terminal pour accéder à la ligne de commande et saisissez la commande `df -h`.

2. Insérez la carte SD dans son lecteur, attendez quelques secondes et relancez la commande `df -h`.
3. Examinez la liste des partitions afin de trouver celle qui a été assignée à la carte SD. Pour ce faire, comparez la nouvelle liste avec l'ancienne pour déterminer quelle ligne a été ajoutée. La carte SD est habituellement assignée à une partition dont le nom est `/dev/sdd1`. Les noms sont attribués dynamiquement et peuvent changer d'une insertion à une autre. Des caractères différents peuvent donc apparaître. Notez le nom de la partition.



Il est aussi possible d'obtenir ces informations avec la commande **`dmesg`**, qui affiche les messages du kernel, notamment les noms des périphériques qui viennent d'être connectés.

4. Il est fort probable que l'ordinateur ait monté automatiquement la carte SD lors de son insertion afin d'examiner son contenu. Nous allons maintenant changer radicalement le système de fichiers. Par conséquent, avant de pouvoir installer le nouveau système d'exploitation, il faut démonter la carte SD grâce à la commande suivante : `umount /dev/sdd1` (remplacez les derniers caractères par ceux que vous avez notés précédemment). Si la commande de démontage échoue, vérifiez que vous n'êtes pas en train d'utiliser ou d'explorer le contenu de la carte.
5. Vous allez maintenant déterminer le nom du périphérique correspondant à la carte SD. Pour ce faire, prenez le nom de la partition et enlevez le dernier chiffre à la fin (qui correspond au numéro de la partition). Par exemple, notre partition `/dev/sdd1` se trouve sur le périphérique `/dev/sdd`.



Comme mentionné précédemment pour Mac OS X, veillez à ne pas vous tromper dans le nom du périphérique, sous peine d'effacer votre système d'exploitation et de perdre vos données. Vérifiez encore une fois avec **`df`** avant de continuer.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

6. Décompressez l'image du système à installer, puis utilisez la commande `dd` pour la recopier bit par bit sur la carte SD grâce à la commande suivante. Remplacez le nom du fichier de l'image par celui que vous avez téléchargé, et substituez `/dev/sdd` avec le nom du périphérique déterminé à l'étape 5.

```
pi@raspberrypi ~ $ sudo dd bs=1m if=2013-07-26-wheezy-
    raspbian.img of=/dev/sdd
```

   Cette commande indique de lancer le programme `dd` avec les droits de l'administrateur (`root`) pour qu'il puisse recopier le fichier source (`if`) dans le fichier destination (`of`) par blocs d'un mégaoctet (`bs=1m`).

7. La copie va durer quelques minutes, puis la commande `dd` affichera un certain nombre de statistiques. Par sécurité, exécutez la commande `sync` pour vider les tampons d'écriture et vous assurer que les données ont été intégralement copiées.

8. Éjectez la carte SD de son lecteur et insérez-la dans le support du Raspberry Pi. Votre système est prêt à démarrer !

BerryBoot

Pour installer un système d'exploitation sur une carte SD, vous pouvez également utiliser l'utilitaire BerryBoot, qui ne nécessite pas de reformatage de la carte SD ni d'écriture de bas niveau.

BerryBoot fait partie du projet de client léger BerryTerminal. Il permet d'installer plusieurs systèmes d'exploitation sur une même carte SD. Pour ce faire, copiez les fichiers de BerryBoot sur une carte SD formatée en FAT standard, puis utilisez-la pour démarrer votre Raspberry Pi. Un utilitaire d'installation va alors vous demander de choisir l'image que vous souhaitez installer. Vous devrez être connecté à Internet pour charger l'image.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

ANNEXE B

LE JEU ASTRAL TRESPASSERS AU COMPLET

Cette annexe contient l'intégralité des scripts en Scratch pour tous les sprites du jeu Astral Trespassers décrit au chapitre 5. Comme Scratch est un langage graphique, il est plus pratique de tout regrouper au même endroit. Les onglets des costumes pour chaque sprite sont aussi présentés.

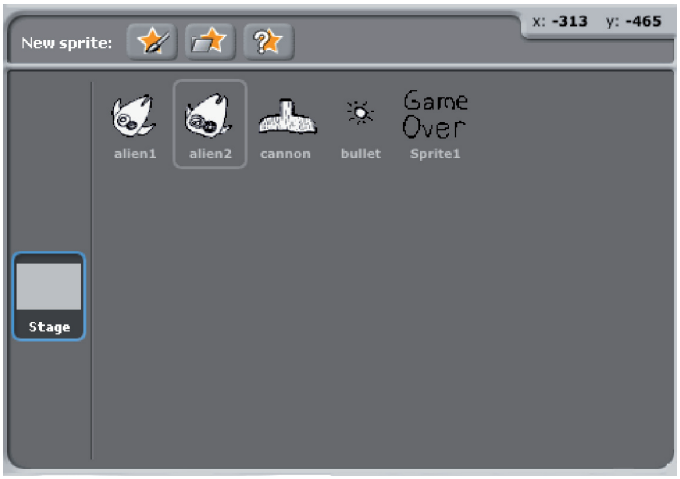


Figure B-1. Les cinq sprites

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

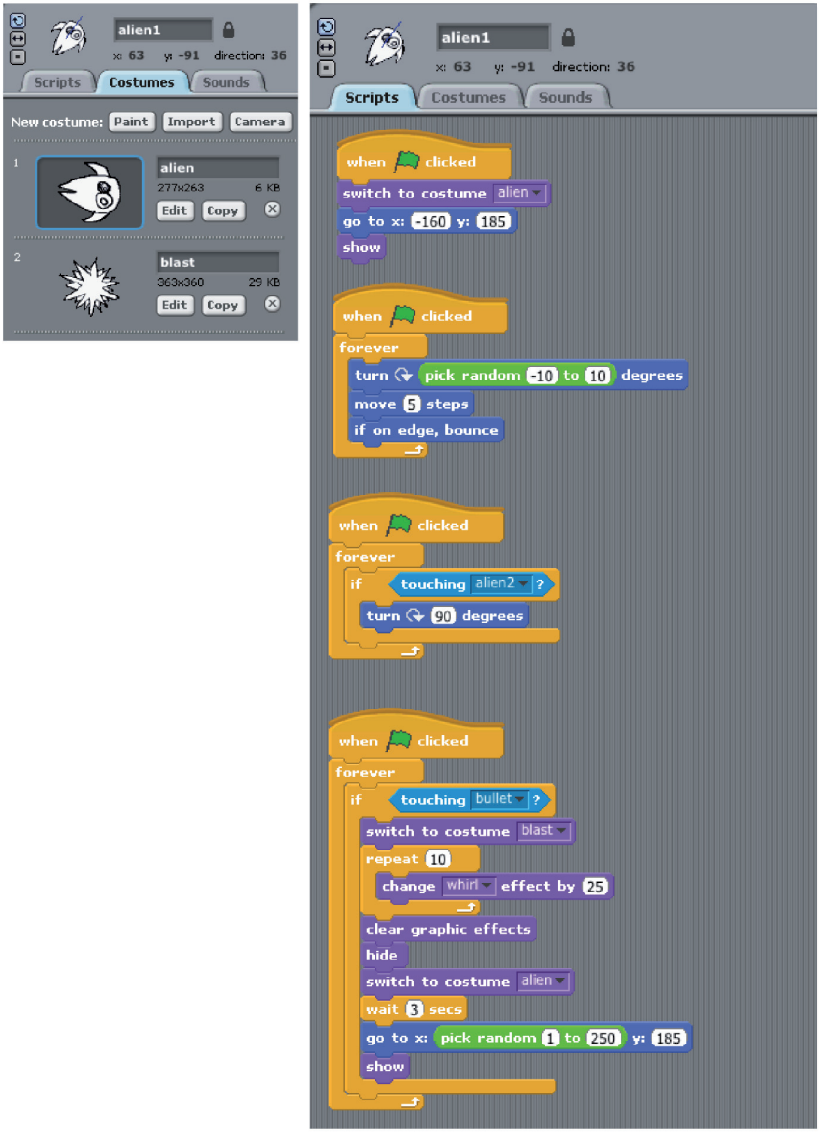


Figure B-2. À gauche, l'onglet des costumes du sprite alien1 ; à droite, son script complet

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

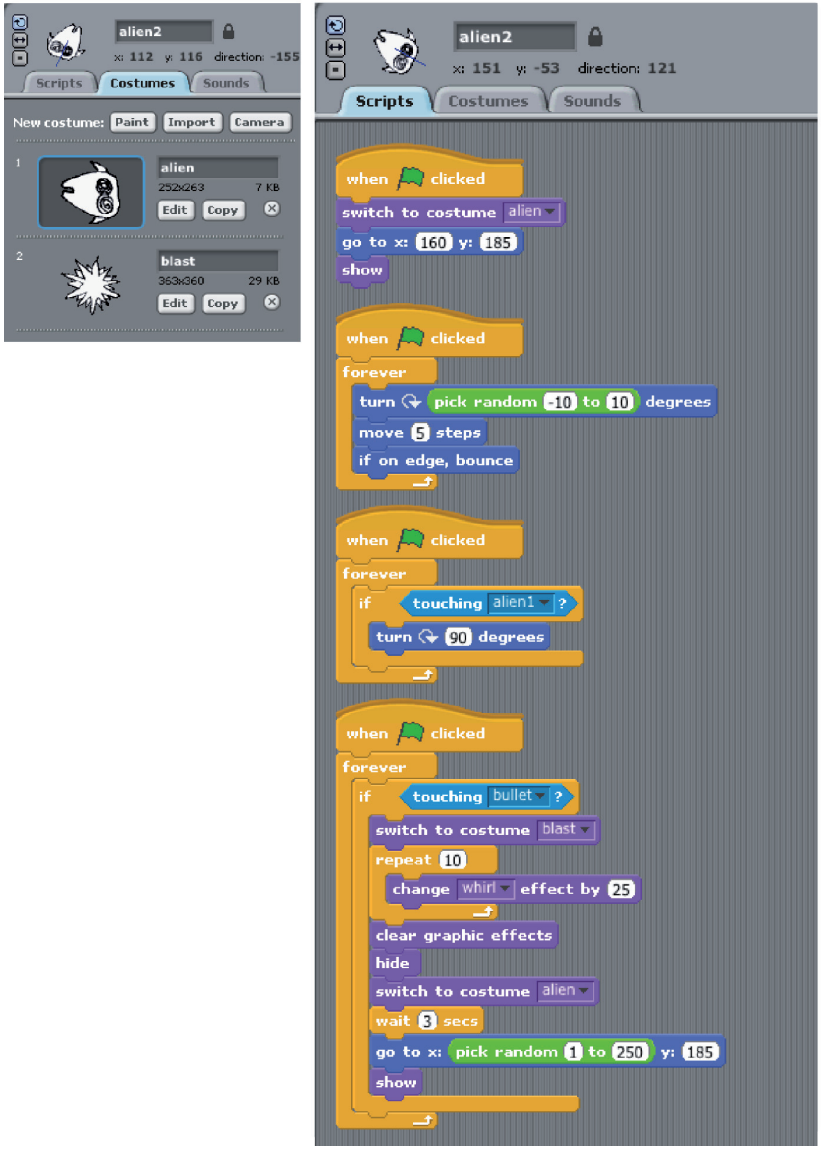


Figure B-3. Onglet des costumes du sprite alien2 et son script complet

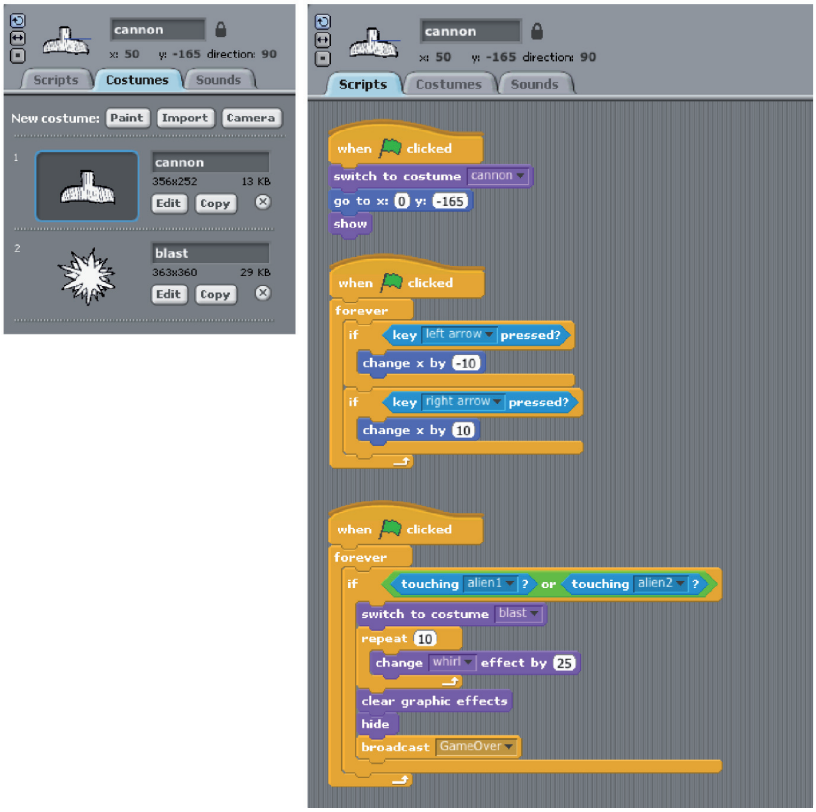


Figure B-4. Onglet des costumes du sprite cannon et son script complet

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

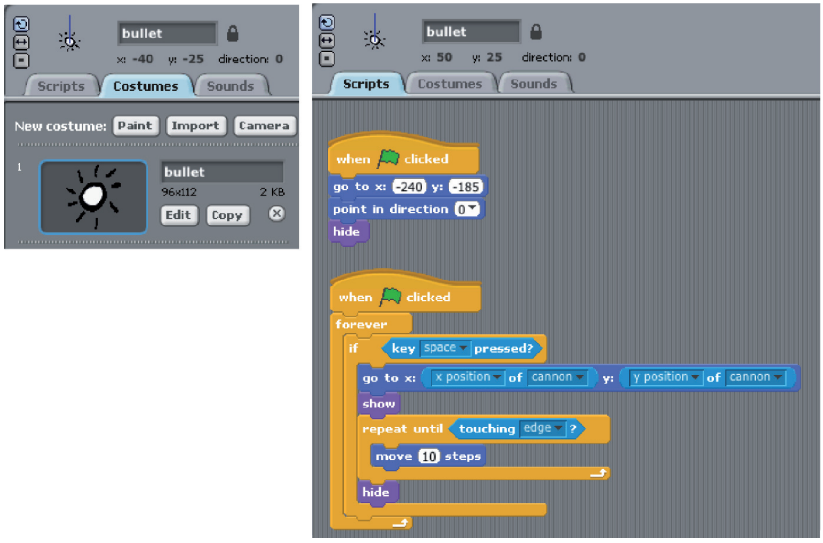


Figure B-5. Onglet des costumes du sprite bullet et son script complet

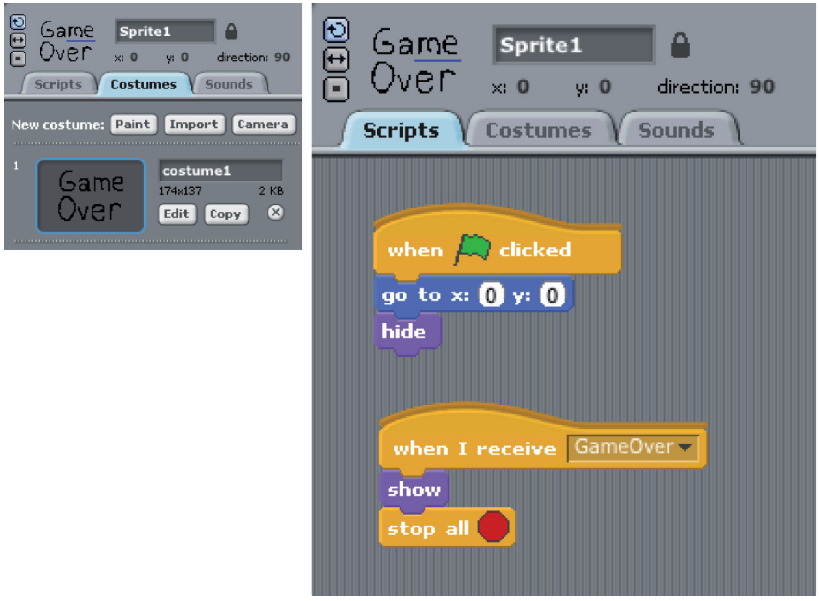


Figure B-6. Onglet des costumes du sprite gameover et son script complet

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

ANNEXE C

ENTRÉES ANALOGIQUES

Dans cet ouvrage, vous avez appris à interfacer le Raspberry Pi avec des boutons-poussoir, des interrupteurs, des LED ou encore des relais, composants qui peuvent être soit allumés, soit éteints. Mais vous voudrez peut-être mesurer des valeurs continues du monde physique, comme une température, une distance, une intensité lumineuse ou un angle.

Malheureusement, vous ne pouvez pas connecter directement de tels capteurs au Raspberry Pi car il ne dispose que d'entrées numériques. Cependant, il est possible d'interfacer ces capteurs à l'aide de circuits électroniques supplémentaires, qui vont convertir les grandeurs analogiques en nombres lisibles par le Raspberry Pi.

Conversion analogique/numérique

Vous allez découvrir dans cette section comment utiliser le composant « convertisseur analogique/numérique » (ADC, *Analog to Digital Converter*). Il existe un grand nombre de circuits remplissant cette fonction, aux caractéristiques multiples et variées. Nous allons utiliser un circuit du fabricant Texas Instruments, dont la référence est ADS1015. Le boîtier de l'ADS1015 étant trop petit pour être piqué dans une platine de prototypage, Adafruit Industries a fabriqué un circuit imprimé (http://www.adafruit.com/products/1083) pour faciliter son utilisation (figure C-1). Une fois que vous aurez soudé les broches des connecteurs, vous pourrez utiliser cette platine dans vos prototypes et vos projets. L'ADS1015 utilise le protocole I^2C pour transférer les valeurs échantillonnées mais il n'est pas nécessaire d'apprendre comment ce protocole fonctionne : Adafruit

fournit aussi une bibliothèque open source en Python pour lire les données sur I²C provenant de l'ADS1015 et de son grand frère, l'ADS1115.

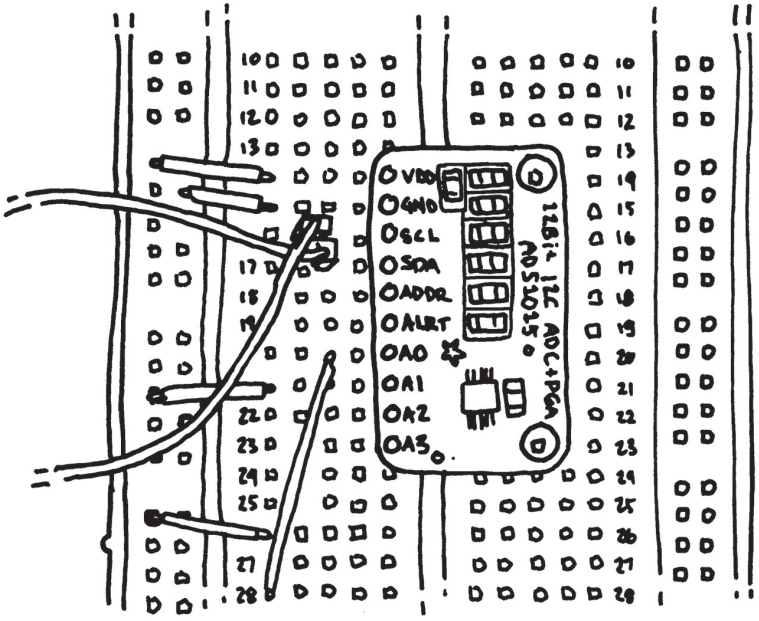


Figure C-1. Le convertisseur analogique/numérique ADS1015 sur sa platine d'adaptation conçue par Adafruit

Voici la marche à suivre pour connecter la platine ADS1015 à votre Raspberry Pi.

1. Connectez la broche 3,3 V du Raspberry Pi au rail d'alimentation positif de la platine de prototypage. Consultez la figure 7-2 page 111 pour localiser les broches de GPIO sur le connecteur du Raspberry Pi.
2. Connectez la broche GND du Raspberry Pi au rail d'alimentation négatif de la platine de prototypage.
3. Piquez le module ADS1015 dans la platine de prototypage et reliez ses broches d'alimentation : la broche VDD est reliée au rail positif et la broche GND au rail négatif.
4. Connectez la broche SCL du module ADS1015 à la broche SCL du Raspberry Pi (près de la broche GND sur le connecteur GPIO).
5. Connectez la broche SDA du module ADS1015 à la broche SDA du Raspberry Pi (située entre la broche SCL et 3,3 V sur le connecteur GPIO).

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Nous allons à présent bricoler un capteur analogique afin de réaliser nos premiers essais. De nombreux types de capteurs peuvent être reliés à l'ADS1015 mais pour ce premier exemple, nous allons utiliser un potentiomètre de 2 kΩ pour mesurer un angle avec le Raspberry Pi.

Un potentiomètre est une sorte de résistance variable, disponible sous forme linéaire ou rotative, qui génère une tension sur une de ses broches, laquelle est proportionnelle à la position du curseur.

Pour connecter un potentiomètre à l'ADS1015, procédez comme suit :

1. Piquez le potentiomètre dans la platine de prototypage.
2. Le potentiomètre est doté de trois broches. Connectez celle du milieu à la broche A0 de l'ADS1015.
3. Une des deux autres broches (peu importe laquelle) doit être connectée au rail d'alimentation positif.
4. La broche restante doit être connectée au rail d'alimentation négatif.

La figure C-2 illustre l'ensemble des connexions que vous devez obtenir.

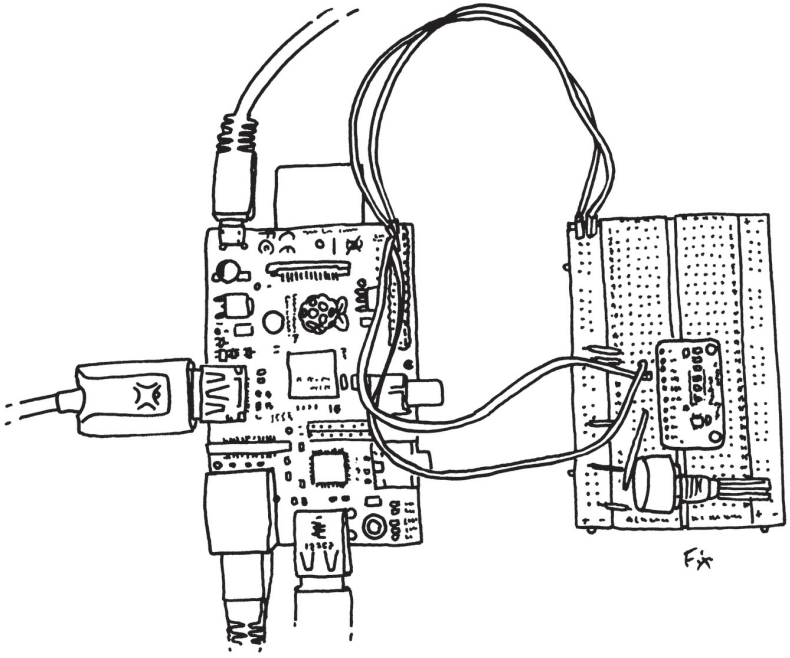


Figure C-2. L'ADS1015 permet de connecter un potentiomètre au Raspberry Pi.

Le matériel étant installé, nous pouvons passer au logiciel. Commençons par configurer le Raspberry Pi en désinhibant l'interface I²C et en installant quelques bibliothèques.

1. Dans un terminal, éditez le fichier `raspi-blacklist.conf` avec les droits `root` :

```
pi@raspberrypi ~ $ sudo nano /etc/modprobe.d/
      raspi-blacklist.conf
```

   Afin de permettre au module I²C d'être chargé dans le noyau, il faut l'enlever de cette liste en ajoutant devant son nom (i2c-bcm2708) le caractère # :

```
# blacklist spi and i2c by default (many users don't need
      them)
blacklist spi-bcm2708
#blacklist i2c-bcm2708
```

2. Quittez l'éditeur nano en appuyant sur les touches Ctrl + X et validez la sauvegarde avec la touche O.

3. Ouvrez ensuite le fichier `/etc/modules` :

```
pi@raspberrypi ~ $ sudo nano /etc/modules
```

4. Ajoutez la ligne `i2c-dev` à la fin du fichier, qui devrait alors ressembler à ceci :

```
# /etc/modules: kernel modules to load at boot time.
#
# This file contains the names of kernel modules that
      should be loaded
# at boot time, one per line. Lines beginning with "#"
      are ignored.
# Parameters can be specified after the module name.
snd-bcm2835
i2c-dev
```

5. Comme à l'étape 2, appuyez sur les touches Ctrl + X pour sortir de nano et validez la sauvegarde avec la touche O.

6. Mettez à jour la liste des paquets de Raspbian :

```
pi@raspberrypi ~ $ sudo apt-get update
```

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

7. Installez les paquets `i2c-tools` et `python-smbus` :

```
pi@raspberrypi ~ $ sudo apt-get install i2c-tools
     python-smbus
```

8. Redémarrez votre Raspberry Pi :

```
pi@raspberrypi ~ $ sudo reboot
```

9. Une fois le système relancé, vérifiez que votre Raspberry Pi peut détecter l'ADS1015. Pour les cartes de première génération, utilisez la commande suivante :

```
pi@raspberrypi ~ $ sudo i2cdetect -y 0
```

   Pour les cartes de deuxième génération, la commande est un peu différente :

```
pi@raspberrypi ~ $ sudo i2cdetect -y 1
```

10. Si la platine est reconnue, vous verrez alors le nombre 48 apparaître dans une grille, comme ceci :

```
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- 48 -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

11. Maintenant que vous savez que la platine est fonctionnelle, vous pouvez passer à l'étape de lecture du potentiomètre. Au préalable, téléchargez la bibliothèque Python d'Adafruit dans votre répertoire personnel. Sur la ligne de commande, recopiez la ligne suivante, en une seule ligne, sans espace dans l'URL :

```
pi@raspberrypi ~ $ wget https://github.com/adafruit/
     Adafruit-Raspberry-Pi-Python-Code/archive/master.zip
```

12. Décompressez cette archive :

```
pi@raspberrypi ~ $ unzip master.zip
```

13. Allez dans le répertoire `ADS1x15` de la bibliothèque :

```
pi@raspberrypi ~ $ cd Adafruit-Raspberry-Pi-Python-Code-
     master/Adafruit_ADS1x15
```

14. Exécutez le script d'exemple :

```
pi@raspberrypi Adafruit_ADS1x15 $ sudo python ads1015_
     example.py
Channel 0 = 2.067 V
Channel 1 = 3.309 V
```

15. Tournez complètement le curseur du potentiomètre dans un sens et relancez le script. Assurez-vous que la valeur du canal 0 a changé :

```
pi@raspberrypi Adafruit_ADS1x15 $ sudo python ads1015_
     example.py
Channel 0 = 3.306 V
Channel 1 = 3.309 V
```

16. Tournez à présent le curseur dans l'autre sens et relancez le script :

```
pi@raspberrypi Adafruit_ADS1x15  $ sudo python ads1015_
     example.py
Channel 0 = 0.000 V
Channel 1 = 3.309 V
```

Comme vous avez pu le constater, la tension du canal 0 de l'ADS1015 change lorsque le curseur est tourné. Le code du script `ads1015_example.py` effectue quelques calculs pour ajuster la valeur fournie par l'ADS1015. Bien sûr, vous devrez modifier ces calculs pour les adapter à vos propres capteurs, selon leur type.

Créons maintenant un nouveau fichier dans le même répertoire, dont le contenu est le suivant :

```
from Adafruit_ADS1x15 import ADS1x15 ❶
from time import sleep

adc = ADS1x15() ❷

while True:
   resultat = adc.readADCSingleEnded(0) ❸
   print resultat
   sleep(.5)
```


Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

❶ Importe la bibliothèque ADS1x15 d'Adafruit.

❷ Crée un nouvel objet de type ADS1x15, appelé `ADC`.

❸ Lit un échantillon du canal 0 de l'ADS1015 et met le résultat dans la variable `resultat`.

Ce code, exécuté avec les droits `root`, va afficher les nombres bruts venant de l'ADS1015 deux fois par seconde. Ces valeurs vont augmenter ou diminuer lorsque vous tournerez le potentiomètre.

La bibliothèque ADS1x15 d'Adafruit effectue tout le travail d'interfaçage pour vous, ce qui facilitera l'emploi de capteurs analogiques dans vos projets. Par exemple, si vous voulez réaliser un jeu de type « Pong », vous pourriez ajouter un deuxième potentiomètre sur le canal 1 et utiliser Pygame (voir chapitre 4) pour gérer l'affichage graphique.

La distribution Educational Linux d'Adafruit

Adafruit Industries a créé une distribution Linux dérivée de Raspbian/Wheezy, préconfigurée avec des pilotes et des applications qui facilitent le prototypage de vos projets. Cette distribution s'appelle Adafruit Raspberry Pi Educational Linux Distro (http://learn.adafruit.com/adafruit-raspberry-pi-educational-linux-distro) et son nom de code est Occidentalis, en référence à une espèce de framboises noires.

Occidentalis est une illustration intéressante de la dynamique du modèle open source : l'écosystème autour de Linux a depuis longtemps vu apparaître des versions adaptées pour des applications particulières. Il n'est pas rare que les distributions dérivées constituent des terrains d'expérimentations fertiles, dont les résultats les plus probants sont souvent réintégrés dans les distributions originales.

Educational Linux Distro apporte de nombreuses améliorations afin de supporter des capteurs et des périphériques courants, ainsi que les produits d'Adafruit, sans avoir à installer d'autres bibliothèques ou reconfigurer le système. Elle comporte même de nouvelles fonctionnalités de bas niveau permettant de contrôler des servomoteurs ou de faire de la modulation de largeur d'impulsion (PWM, *Pulse Width Modulation*) directement avec les broches GPIO du Raspberry Pi.

Consultez le site d'Adafruit pour visualiser la liste des fonctionnalités d'Occidentalis, que vous pourrez mettre en œuvre grâce aux nombreux tutoriels d'Adafruit.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

ANNEXE D

À PROPOS DES CONTRIBUTEURS

Les auteurs

Matt Richardson est un créatif et producteur vidéo installé à Brooklyn, aux États-Unis. Il écrit régulièrement pour le magazine *Make* et le site Makezine.com. Il est également consultant pour Awesome Button Studios.

Parmi ses travaux les plus connus, on notera la Descriptive Camera, un appareil photo qui génère une description textuelle au lieu d'une image. Il a aussi conçu The Enough Already, un appareil qui coupe le son du téléviseur lorsque certains mots-clés sont prononcés. Ses projets ont notamment été présentés dans le *New York Times*, *Wired* et *New York Magazine*, et exposés au Nevada Museum of Art, à la Biennale de Santorini, au Festival international de photographie de Rome et à la Design Week de Milan.

Shawn Wallace est un designer d'open hardware au Modern Device de Providence, aux États-Unis. Il est aussi membre du collectif d'artistes Fluxama, qui a créé de nouveaux instruments de musique électronique sur iOS, tels que Noisemusick et DR-OM. Il a aussi été éditeur aux éditions O'Reilly Media et Maker Media, et a enseigné au Fab Lab de Providence dans le cadre de la Fab Academy. Il a par ailleurs dirigé durant plusieurs années le centre culturel AS220 de cette même ville.

Les traducteurs

Yann Guidon écrit depuis 1999 pour *Linux Magazine France* et s'investit dans les mouvements du logiciel libre et du hardware libre. Sous le label YGDES, il intervient à Paris dans de nombreux projets, où il collabore surtout avec des artistes. Il a conçu et fabriqué de nombreux systèmes à LED, qui ont été exposés en galerie, au Centre Pompidou de Metz, au

Mepic (Musée éclaté de la presqu'île de Caen), à la Fnac Montparnasse et dans l'aéroport Roissy-Charles-de-Gaulle. Après avoir conçu l'architecture du F-CPU, il développe maintenant le YASEP, un microprocesseur embarqué 100 % open source.

Originaire de New York et résidant à Paris, Janet Hoffmann est écrivain, traductrice et consultante. Sa formation en administration de structures éducatives lui permet aussi bien de collaborer avec des artistes et universitaires que d'organiser des activités d'enrichissement pour jeunes surdoués.

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

INDEX

A

ADS1015 199
alimentation 8
allumage d'une LED 112
API 166
Arch Linux 13
Arduino 95
  port série 99
Astral Trespassers 84, 193
audio 25

B

BerryBoot 191
BitTorrent 14
boîte à bruitages 138
boîtier 11
bouton-poussoir 117
breadboard 113
broche d'entrées-sorties généralistes (GPIO) 6 107

C

cabine photographique 158
câble
  Ethernet 9
  HDMI 9
carte 2
  SD 8, 187
code source XII
configuration du Raspberry Pi 17
connecteur
  Camera Serial Interface (CSI) 6
  Display Serial Interface (DSI) 6
connexion Internet 15
conversion analogique/numérique 199
copier-coller 26
cron 125
CSI 146

D

démarrage du Raspberry Pi 15
dissipateur 10
distribution 12
dock d'extension 11
dongle Wi-Fi USB 10

E

éditeur de texte 26
Educational Linux 205
entrée analogique 199

F

faire clignoter une LED 133
fichier 28
  virtuel 116
Flask 174

G

gestionnaire des fichiers 24
GPIO 108, 130

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28

H

horloge temps réel 10
hub USB alimenté 10

I

IDLE 44

J

Jinja2 176

L

lampe contrôlée par Internet 180
Leafpad 26
lecture d'un bouton 117
LibreOffice VIII
ligne de commande 27
Linux 23, 189
lire l'état d'un bouton 135
luvcview 147
LXDE 24
LXTerminal 27

M

Mac OS X 187
Midori 24
mise hors tension
    du Raspberry Pi 20
mode headless 39
module
  caméra 10
  RPi.GPIO 178
moniteur LCD 10

N

nano 26
navigateur web 24

O

omxplayer 25
Openbox 24

P

périphérique 7
permission 36
PicoBoard 91
points d'accès P2 et P3 6
prévisions météo 166
problèmes courants 20
processus 35
puce SoC 108
Pygame 59
  entrée 65
  événement 65
  son 70
  surface 61
  vidéo 73
Python 43
  animation 59
  entrée-sortie 129
  erreur 55
  Internet 163
  module 50
  multimédia 59
  objet 50

Q

QtonPi 13

R

Raspberry Pi
  connecteur d'alimentation 6
  distributeur 2
  fabricant 2
  LED d'état 5
  prise Ethernet 5

prise HDMI 5
prise USB 4
processeur 4
sortie audio analogique 6
sortie vidéo composite 6
support de carte SD 4
Raspberry Pi Linux version éducative d'Adafruit (Occidentalis) 13
Raspbian 13, 24
Arduino 97
Raspod XII
reconnaissance de visage 156
réglage de la date et de l'heure 40
répertoire /etc 40
Requests 164
réseau 38
résistance de rappel 118

S

Scratch 75, 193
scène 81
sprite 82
script shell 121
séquenceur de lampe 121, 180
serveur web 173
shell 27
SimpleCV 148
sprite 68, 193
station de base 11
sudo 36
superordinateur XII
système de fichiers 28

T

téléchargement de données web 164
touche de contrôle 34

V

vidéo 25

W

webcam 145
Windows 189

X

XBian 13
XBMC X

Ce document est la propriété exclusive de Patrick de Maqueville (patrick.f33@gmail.com) - 23 décembre 2014 à 17:28