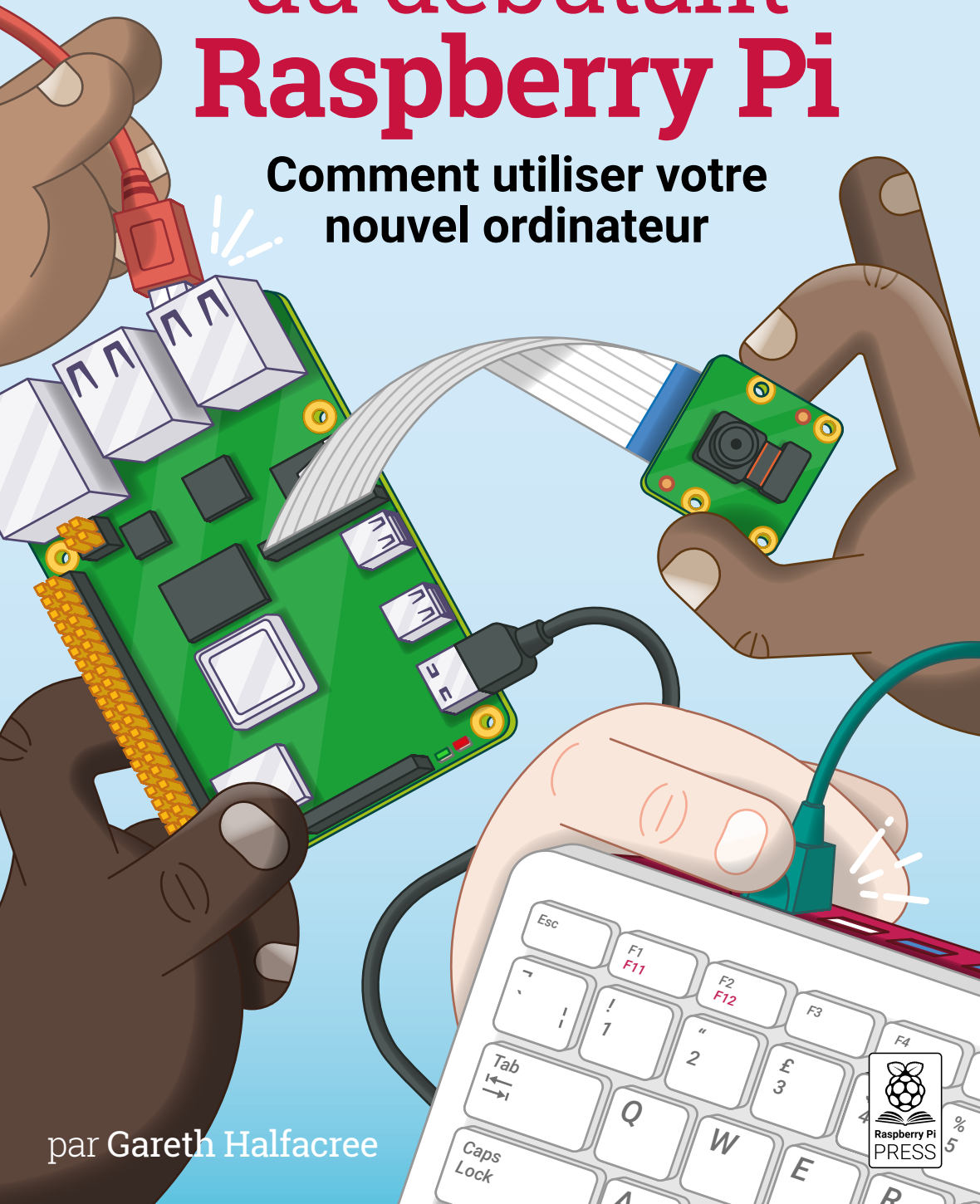
4e Édition

ENTIÈREMENT MIS À JOUR POUR **RASPBERRY PI 400**

LE GUIDE OFFICIEL du débutant Raspberry Pi

Comment utiliser votre nouvel ordinateur



par Gareth Halfacree

LE GUIDE OFFICIEL du débutant Raspberry Pi

Comment utiliser votre nouvel ordinateur

Première édition en 2020 par Raspberry Pi Trading Ltd, Maurice Wilkes Building,
St. John's Innovation Park, Cowley Road, Cambridge, CB4 0DS

Responsable de publication : Russell Barnes • Rédacteur en chef : Phil King
Design : Critical Media • Illustrations : Sam Alder
PDG : Eben Upton

ISBN : 978-1-912047-93-2

Bienvenue dans le Guide officiel Raspberry Pi du débutant

Nous en sommes certains, vous allez adorer Raspberry Pi. Cet ordinateur ultra petit et abordable coûte moins cher que la plupart des jeux vidéo, mais peut être utilisé pour apprendre à coder, construire des robots et créer toutes sortes de projets bizarres et merveilleux.

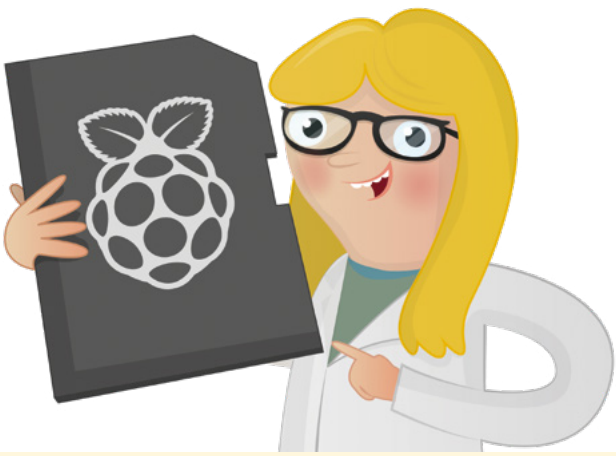
Raspberry Pi est capable de faire tout ce que l'on peut demander à un ordinateur : naviguer sur Internet, jouer à des jeux, regarder des films et écouter de la musique. Mais Raspberry Pi est bien plus qu'un ordinateur moderne.

Avec Raspberry Pi, vous pouvez pénétrer dans le cœur d'un ordinateur. Vous pouvez configurer votre propre système d'exploitation et connecter des fils et des circuits directement aux points de sa carte mère. Il a été conçu pour enseigner aux jeunes comment programmer dans des langages comme Scratch et Python. Tous les principaux langages de programmation sont inclus dans le système d'exploitation officiel.

Plus que jamais, le monde a besoin de programmeurs, et Raspberry Pi a contribué à éveiller la passion de l'informatique et de la technologie auprès des nouvelles générations.

Avec Raspberry Pi, des personnes de tous âges créent des projets passionnants, des consoles de jeu rétro aux stations météo connectées à Internet.

Que vous rêviez de créer des jeux, construire des robots ou réaliser vos projets les plus étonnants, ce livre vous aide à partir du bon pied.



À propos de l'auteur

Gareth Halfacree est un journaliste indépendant spécialisé en technologie, écrivain et ancien informaticien dans le secteur de l'éducation. Passionné par les logiciels et le matériel open source, il a été l'un des premiers à adopter la plateforme Raspberry Pi et a publié plusieurs ouvrages sur les capacités et la flexibilité de celle-ci. Vous pouvez le suivre sur Twitter en tant que **@ghalfacree** ou consulter son site Web à l'adresse **freelance.halfacree.co.uk**.

Table des matières

ANNEXES

Chapitre 1

Premiers pas avec votre Raspberry Pi

Familiarisez-vous avec votre nouvel ordinateur de la taille d'une carte de crédit en explorant les profondeurs de votre Raspberry Pi. Découvrez tous ses composants et leurs fonctions

Raspberry Pi est un appareil remarquable : c'est un ordinateur entièrement fonctionnel dans un tout petit emballage peu coûteux. Que vous recherchiez un appareil pour naviguer sur le Web ou pour jouer, que vous souhaitiez apprendre à écrire vos propres programmes ou que vous cherchiez à créer vos propres circuits et dispositifs physiques, Raspberry Pi et sa merveilleuse communauté vous soutiendront à tout moment.

Raspberry Pi est ce qu'il est convenu d'appeler un *ordinateur monocarte*, ce qui signifie exactement ce que son nom indique : il s'agit d'un ordinateur, exactement comme n'importe quel autre ordinateur de bureau, ordinateur portable ou smartphone, mais construit sur une seule *carte de circuit imprimé*. Comme la plupart des ordinateurs monocarte, Raspberry Pi est de petites dimensions, à peu près comme carte de crédit, ce qui ne veut pas dire qu'il n'est pas puissant : un Raspberry Pi dispose des mêmes capacités qu'un ordinateur plus grand et plus énergivore, mais pas nécessairement aussi rapidement.

La famille Raspberry Pi est née du désir d'appuyer une éducation informatique plus pratique dans le monde entier. Ses créateurs, qui ont créé ensemble la Fondation Raspberry Pi, à but non lucratif, étaient loin de se douter de l'ampleur de l'aventure : les quelques milliers d'appareils fabriqués en 2012 en guise d'essai ont immédiatement été vendus, et des millions ont été expédiés dans le monde entier au cours des années qui ont suivi. Ces cartes mères ont creusé leur nid dans des maisons, des salles de classe, des bureaux, des centres de données, des usines, voire dans des bateaux autopilotés et des ballons aérostatiques.

Depuis le modèle B d'origine, différents modèles de Raspberry Pi ont vu le jour, chacun apportant soit des spécifications améliorées, soit des caractéristiques spécifiques à certaines utilisations particulières. La gamme Raspberry Pi Zero, par exemple, est une version miniature du Raspberry Pi à quelques fonctionnalités près (par exemple, elle ne dispose pas de multiples ports USB et de port

réseau câblé), au profit d'un encombrement nettement réduit et de besoins en énergie moindres.

Tous les modèles Raspberry Pi partagent cependant une caractéristique commune : ils sont tous *compatibles*, ce qui signifie que les logiciels conçus pour un modèle donné fonctionneront sur n'importe quel autre modèle. Il est par ailleurs possible d'installer la toute dernière version du système d'exploitation Raspberry Pi sur un prototype original de pré-lancement du modèle B. Certes, il sera plus lent, mais il fonctionnera quand même.

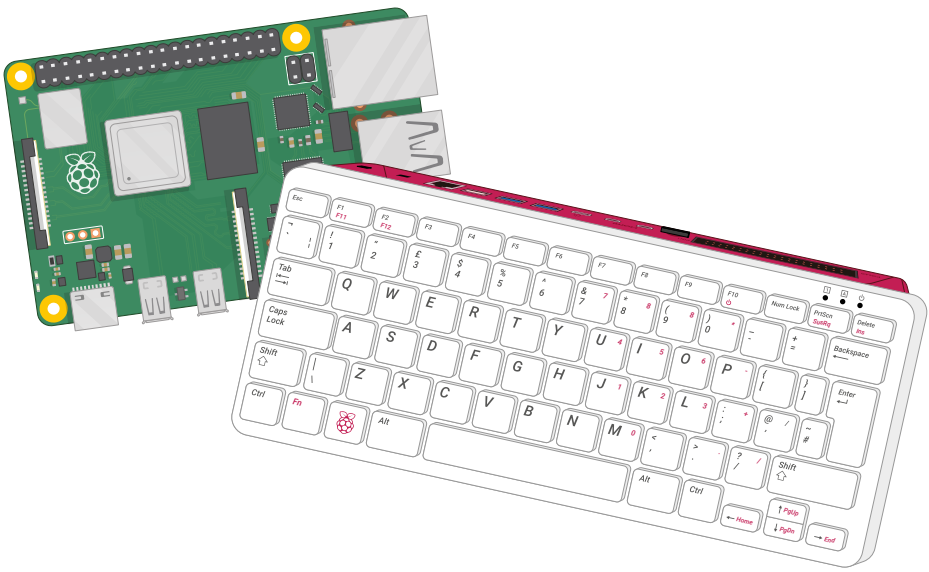
Dans ce livre, vous découvrirez les versions Raspberry Pi 4 Model B et Raspberry Pi 400, les versions les plus récentes et les plus puissantes de Raspberry Pi. Ce que vous apprenez, cependant, peut facilement être appliqué à d'autres modèles de la famille Raspberry Pi, ne vous inquiétez surtout pas si vous utilisez une version différente.

Raspberry Pi 400

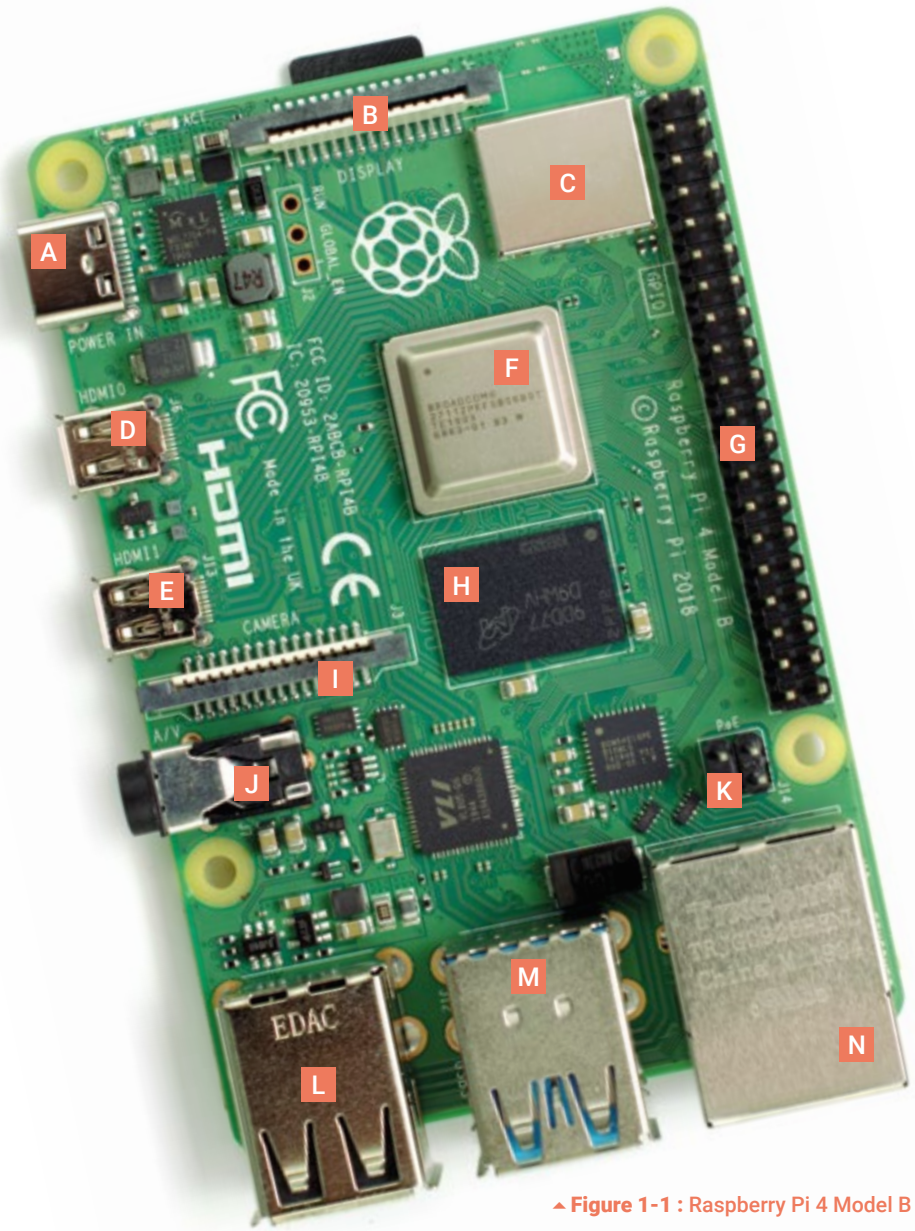
Si vous avez un Raspberry Pi 400, la carte circuit est intégrée dans le boîtier du clavier. Continuez à lire pour découvrir tous les composants qui donnent vie à votre Raspberry Pi, ou passez à la page 20 pour plonger dans les profondeurs de votre appareil.

Une visite guidée de Raspberry Pi

Contrairement à un ordinateur traditionnel, dont les circuits internes sont dissimulés dans un boîtier, un Raspberry Pi standard dispose de tous ses composants, ports et fonctionnalités. Vous pouvez bien entendu acheter un boîtier pour assurer une protection supplémentaire, si vous préférez. Ainsi, il s'agit d'un excellent outil pour apprendre quel est le rôle des différentes parties d'un ordinateur, ainsi que pour se familiariser avec le branchement des différentes parties, désignées par le terme de *périphériques* lorsque vous devrez vous y mettre.



- A Alimentation USB Type-C
- B Connecteur d'affichage DSI
- C Sans fil / Bluetooth
- D Micro-HDMI 0
- E Micro-HDMI 1
- F Système sur puce
- G GPIO
- H RAM
- I Port caméra CSI
- J Prise AV 3,5 mm
- K PoE (Alimentation par Ethernet)
- L USB 2.0
- M USB 3.0
- N Port Ethernet



▲ **Figure 1-1 :** Raspberry Pi 4 Model B

Figure 1-1 Vue aérienne d'un Raspberry Pi 4 Model B. Si vous utilisez un Raspberry Pi pendant que vous lisez le présent guide, essayez de le positionner de la même façon que sur l'image ; si vous le placez dans l'autre sens, il pourrait être compliqué d'utiliser des éléments comme les connecteurs GPIO (qui vous sont expliqués en détail dans le **Chapitre 6, L'informatique physique avec Scratch et Python**).

On pourrait penser qu'une toute petite carte renferme bien des secrets, mais dans les faits le fonctionnement d'un Raspberry Pi est très simple à comprendre, à commencer par ses *composants*, les parties internes qui en assurent le fonctionnement.

Composants d'un Raspberry Pi

Comme tout autre ordinateur, un Raspberry Pi est composé de différents éléments, chacun remplissant un rôle bien défini dans le cadre de son fonctionnement. Le premier, et sans doute le plus important, se trouve juste au-dessus du point central sur la face supérieure de la carte (**Figure 1-2**), recouvert d'un capuchon métallique : le *système sur puce* (SoC).

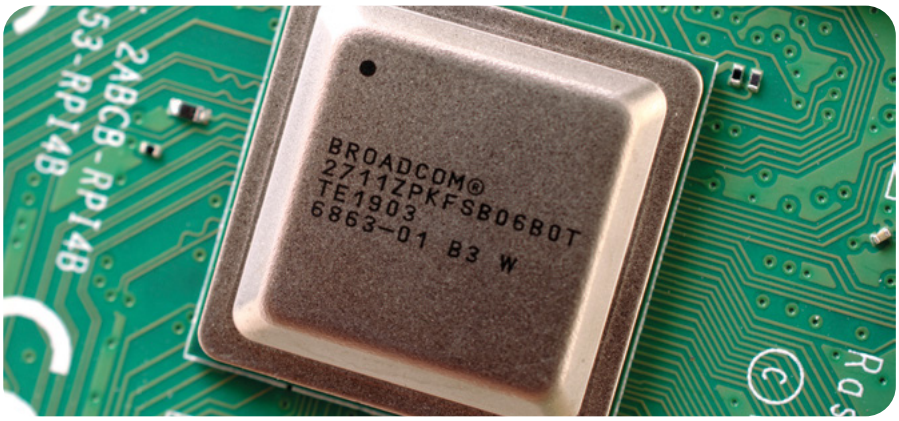


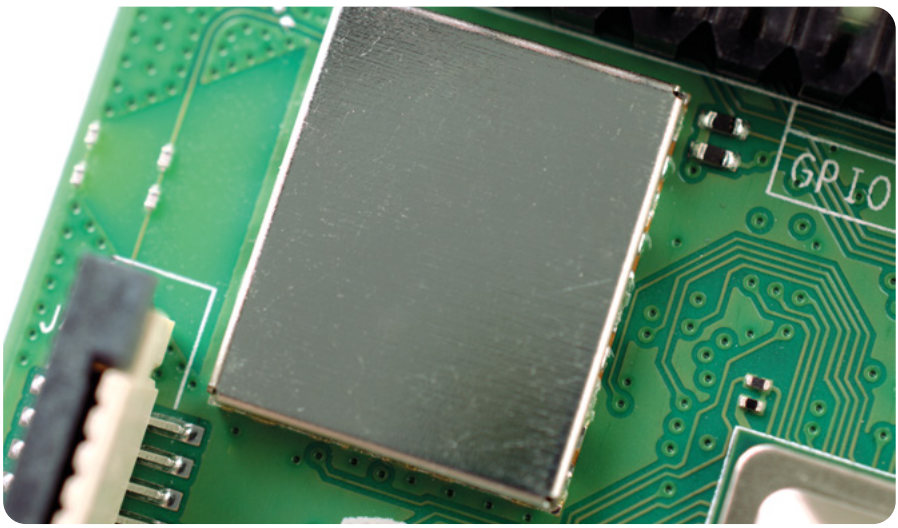
Figure 1-2 : Le système sur puce de Raspberry Pi (SoC)

Le terme « système sur puce » (system-on-chip en anglais) est un bon indicateur de ce qui se cache en dessous du couvercle métallique : une puce en silicium, appelée *circuit intégré,* qui contient l'essentiel du système de Raspberry Pi. Il comprend l'*unité centrale de traitement* (CPU), généralement considérée comme le « cerveau » d'un ordinateur, et l'*unité de traitement graphique* (GPU), qui gère l'aspect visuel.

Un cerveau sans mémoire ne servant pas à grand-chose, juste à côté du SoC se trouve une autre puce, qui ressemble à un petit carré de plastique noir (**Figure 1-3**, au verso). Il s'agit de la *mémoire vive (RAM)* de Raspberry Pi. Lorsque vous travaillez avec un Raspberry Pi, c'est la mémoire vive qui contient tout ce que vous faites ; ce n'est que lorsque vous enregistrez votre travail qu'il s'inscrit sur la carte microSD. Ensemble, ces composants forment la mémoire volatile et non volatile de Raspberry Pi : la RAM volatile se vide de son contenu chaque fois que Raspberry Pi s'éteint, tandis que la carte microSD non volatile sauvegarde son contenu.

▲ **Figure 1-3 :** Mémoire vive (RAM) de Raspberry Pi

Sur la partie supérieure droite de la carte, vous trouverez un autre couvercle métallique (**Figure 1-4**) couvrant la *radio*, le composant qui permet à Raspberry Pi de communiquer sans fil avec d'autres appareils. En soi, la radio remplit un double rôle : celui de *radio WiFi* pour la connexion aux réseaux informatiques, et celui de *Bluetooth radio* pour se connecter à des périphériques tels que des souris et pour envoyer ou recevoir des données de dispositifs intelligents situés à proximité, tels que des capteurs ou des smartphones.



▲ **Figure 1-4 :** Module radio d'un Raspberry Pi

Une autre puce noire, recouverte de plastique, est visible sur le bord inférieur de la carte juste derrière les ports USB du milieu. Il s'agit du *contrôleur USB*, qui est responsable du fonctionnement des quatre ports USB. À ses côtés se trouve une puce encore plus petite, le *contrôleur de réseau* qui gère le port réseau Ethernet de Raspberry Pi. Une dernière puce noire, plus petite que les autres, se trouve un peu au-dessus du connecteur d'alimentation USB de type C, en haut à gauche de la carte (**Figure 1-5**) ; il s'agit d'un *circuit intégré de gestion de l'alimentation (PMIC)* et des poignées qui transforment l'énergie provenant du port micro USB pour alimenter le Raspberry Pi.

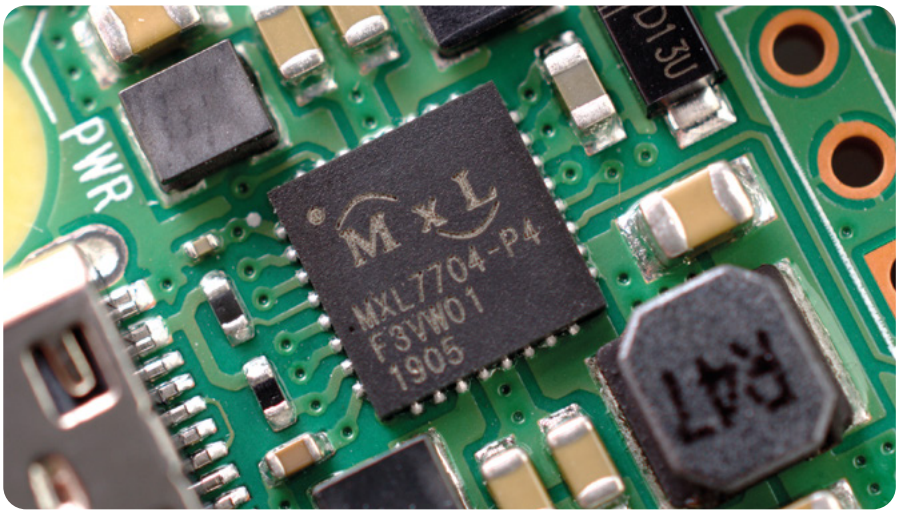


Figure 1-5 : Le circuit intégré de gestion de l'alimentation de Raspberry Pi (PMIC)

Ne vous inquiétez pas si tout cela vous semble trop compliqué : vous n'avez pas besoin de savoir quel est le rôle de chaque composant ni sa position sur la carte pour utiliser Raspberry Pi.

Les ports d'un Raspberry Pi

Tout Raspberry Pi dispose d'un certain nombre de ports, à commencer par quatre *ports USB (Universal Serial Bus)* (**Figure 1-6**) au centre et à droite du bord inférieur. Ces ports vous permettent de connecter n'importe quel périphérique compatible USB à Raspberry Pi, des claviers et souris aux appareils photo numériques et clés USB. D'un point de vue technique, il existe deux types de ports USB : ceux qui comportent des parties noires sont des ports USB 2.0, basés sur la deuxième version de la norme Universal Serial Bus ; ceux qui comportent des parties bleues sont des ports USB 3.0, plus rapides, basés sur la troisième version, plus récente.

▲ **Figure 1-6 :** Les ports USB d'un Raspberry Pi

À droite des ports USB se trouve un *port Ethernet*, également désigné par le terme *port réseau* (**Figure 1-7**). Vous pouvez utiliser ce port pour connecter votre Raspberry Pi à un réseau informatique câblé à l'aide d'un câble muni d'un connecteur RJ45 à son extrémité. Si vous regardez attentivement le port Ethernet, vous verrez deux diodes électroluminescentes (LED) sur la partie inférieure ; ce sont des LED d'état qui vous indiquent que la connexion fonctionne.

Figure 1-7 : Port Ethernet d'un Raspberry Pi

Juste au-dessus du port Ethernet, sur le bord gauche du Raspberry Pi, se trouve une *prise audio-visuelle (AV) de 3,5 mm* (**Figure 1-8**). Cette prise est également désignée par le terme *prise casque* et vous pouvez effectivement y brancher votre casque, mais vous obtiendrez un meilleur son en connectant des haut-parleurs amplifiés plutôt qu'un casque. Elle remplit cependant une fonction supplémentaire peu connue : outre l'audio, la prise AV 3,5 mm transmet un signal vidéo qui peut être connecté à un écran TV, un projecteur ou tout autre écran prenant en charge un *signal vidéo composite* à l'aide d'un câble spécial appelé adaptateur *TRRS (ti-ring-ring-sleeve)*.

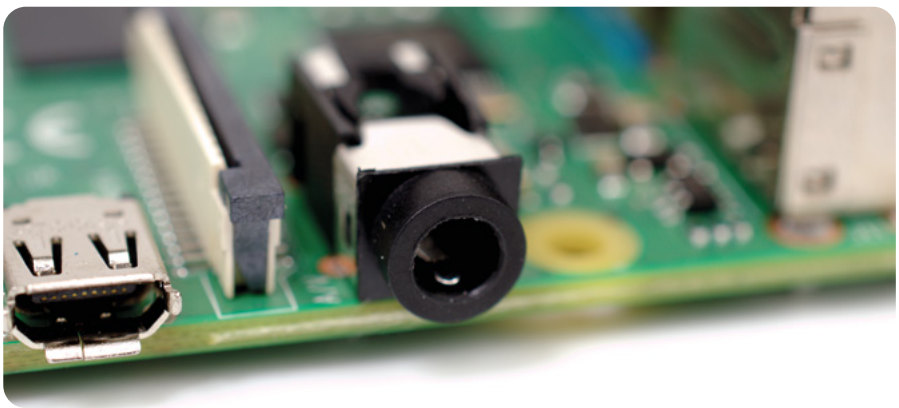
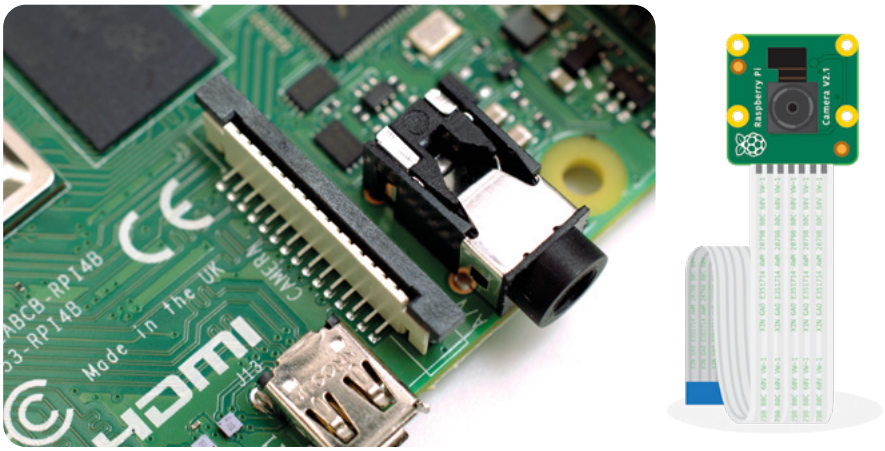


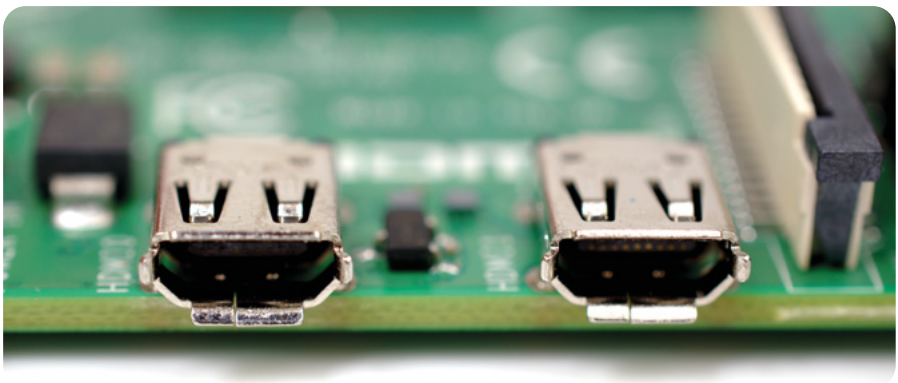
Figure 1-8 : Prise AV 3,5 mm du Raspberry Pi

Directement au-dessus de la prise AV 3,5 mm se trouve un connecteur d'apparence étrange avec un rabat en plastique qui peut être tiré vers le haut ; il s'agit du *connecteur de caméra*, également connu sous le nom de *Interface série de la caméra (CSI)* (**Figure 1-9**). Cela vous permet d'utiliser le Raspberry Pi Camera Module spécialement conçu à cet effet (sur lequel vous obtiendrez plus d'informations dans le **Chapitre 8, Raspberry Pi Camera Module**.)



▲ **Figure 1-9 :** Connecteur caméra d'un Raspberry Pi

Au-dessus, toujours sur le bord gauche de la carte, se trouvent les *micro ports d'interface multimédia haute définition (micro-HDMI)*, qui sont une version réduite des connecteurs que vous trouverez sur une console de jeux, un décodeur ou un téléviseur (**Figure 1-10**). L'élément multimédia qui compose la seconde partie de leur nom vous indique qu'ils transmettent à la fois des signaux audio et vidéo, tandis que « haute définition » est un indicateur d'excellente qualité. Ils servent à connecter votre Raspberry Pi à un ou deux dispositifs d'affichage : un moniteur d'ordinateur, un écran TV ou un projecteur.



▲ **Figure 1-10 :** Ports micro-HDMI d'un Raspberry Pi

Au-dessus des ports HDMI se trouve un *port d'alimentation USB de type C* (**Figure 1-11**), que vous utiliserez pour connecter votre Raspberry Pi à une source d'énergie. Les smartphones, tablettes et autres appareils portables sont dotés d'un port USB de type C. Vous pouvez alimenter votre Raspberry Pi à l'aide d'un chargeur mobile standard, mais pour de meilleurs résultats, il convient plutôt d'utiliser le bloc d'alimentation officiel USB de type C de Raspberry Pi.

Figure 1-11 : Port d'alimentation de type C USB d'un Raspberry Pi

Sur le bord supérieur de la carte se trouve un autre connecteur d'apparence bizarre (**Figure 1-12**), qui à première vue semble identique au connecteur de la caméra. Il s'agit pourtant exactement du contraire : un *connecteur d'affichage* ou *Interface série d'affichage (DSI)* conçu pour être utilisé avec un écran tactile Raspberry Pi (**Figure 1-13** au verso).

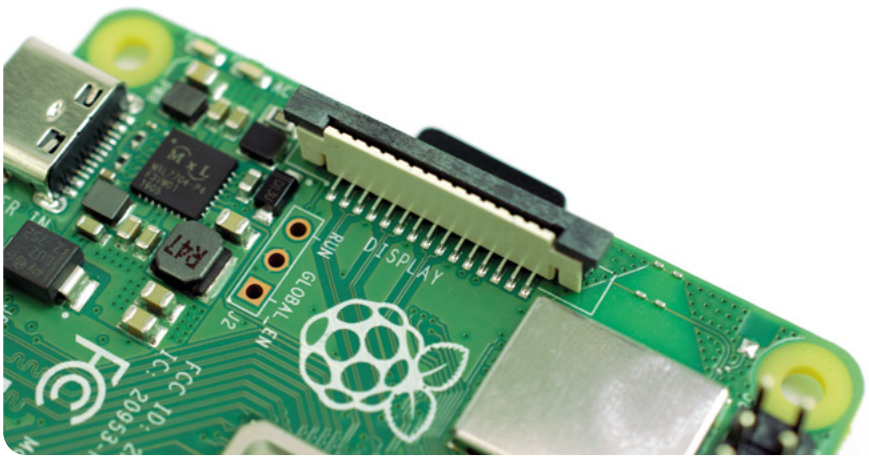
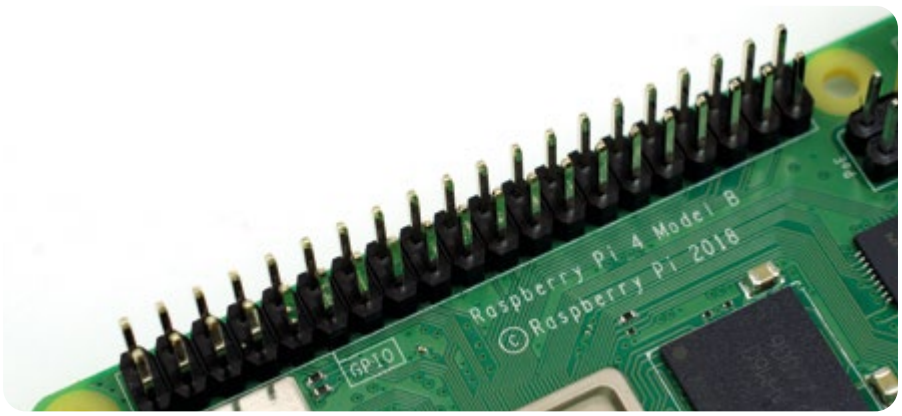


Figure 1-12 : Connecteur d'affichage de Raspberry Pi (DSI)

▲ **Figure 1-13 :** Écran tactile Raspberry Pi

Sur le bord droit de la carte, vous trouverez 40 broches métalliques, réparties en deux rangées de 20 points (**Figure 1-14**). Il s'agit du *connecteur GPIO (general-purpose input/output)*, une fonctionnalité qui permettait à Raspberry Pi de communiquer avec du matériel supplémentaire, qu'il s'agisse de LED et de boutons jusqu'aux capteurs de température, aux joysticks ou aux moniteurs de fréquence cardiaque. Vous en apprendrez plus sur le connecteur GPIO dans le **Chapitre 6, L'informatique physique avec Scratch et Python**. Juste en dessous et à gauche de ce connecteur se trouve un autre connecteur, plus petit, à quatre broches : il sert à connecter l'alimentation électrique par Ethernet (PoE) HAT, un module complémentaire optionnel qui permet à Raspberry Pi d'être alimenté par une connexion réseau plutôt que par le port USB de type C.



▲ **Figure 1-14 :** Connecteur GPIO du Raspberry Pi

Votre Raspberry Pi comporte un tout dernier port, mais il n'est pas visible sur la partie supérieure. Retournez la carte et vous trouverez un *connecteur de carte microSD* sur l'envers, face au connecteur d'affichage (**Figure 1-15**). Il s'agit du stockage de Raspberry Pi : la carte microSD insérée ici contient tous les fichiers que vous enregistrez, tous les logiciels que vous installez et le système d'exploitation qui assure le fonctionnement du Raspberry Pi.

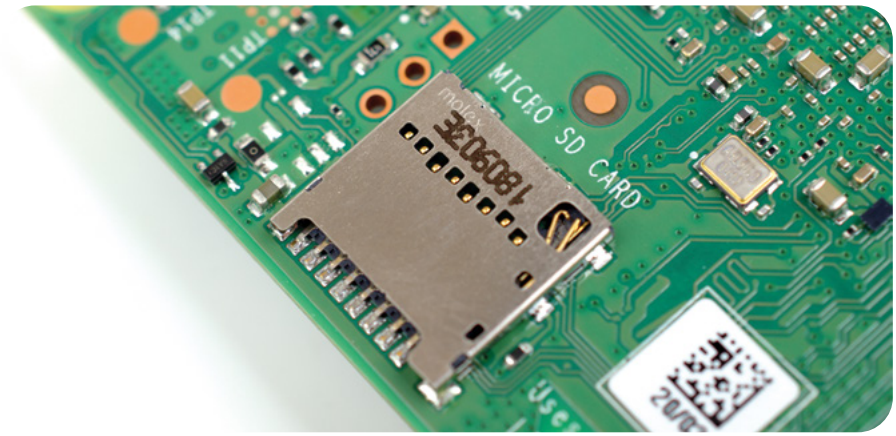


Figure 1-15 : Connecteur carte microSD d'un Raspberry Pi

▲ **Figure 1-16 :** Le modèle Raspberry Pi 400 possède un clavier intégré

Raspberry Pi 400

Le modèle Raspberry Pi 400 reprend les mêmes composants que le Raspberry Pi 4 et les places à l'intérieur d'un boîtier de clavier. En plus de les protéger, le boîtier du clavier prend moins de place sur votre bureau et permet de garder les câbles en ordre.

Le Raspberry Pi 400 comporte les mêmes composants de base que le Raspberry Pi 4, y compris le système sur puce et la mémoire. Vous ne les voyez pas, mais ils sont toujours là. Vous pouvez voir les parties externes, en commençant par le clavier (**Figure 1-16**). Dans le coin droit se trouvent trois diodes électroluminescentes (LED) : la première s'allume lorsque vous appuyez sur la touche Verr Num, qui fait en sorte que certaines touches fonctionnent comme un pavé numérique sur un clavier de taille normale ; la deuxième s'allume lorsque vous appuyez sur Verr Maj, qui fait en sorte que les touches de lettres soient en majuscules plutôt qu'en minuscules ; et la dernière s'allume lorsque le Raspberry Pi 400 est allumé.

Les ports se trouvent à l'arrière du Raspberry Pi 400 (**Figure 1-17**). Le port le plus à gauche, vu de l'arrière, est le connecteur GPIO (General-Purpose Input/Output). Il s'agit du même connecteur qui est décrit à la page 17, mais inversé : la première broche, la broche 1, se trouve en haut à droite, tandis que la dernière broche, la broche 40, se trouve en bas à gauche. Vous aurez l'occasion d'en savoir plus sur le connecteur GPIO dans le **Chapitre 6, L'informatique physique avec Scratch et Python**.

Figure 1-17 : Les ports se trouvent à l'arrière du Raspberry Pi 400

La fente pour la carte microSD est placée juste aux côtés du connecteur GPIO. Elle permet d'insérer la carte microSD sur laquelle sont stockés le système d'exploitation, les applications et autres données du Raspberry Pi 400. La carte microSD est préinstallée dans le Raspberry Pi 400 ; vous pouvez la retirer en appuyant doucement sur la carte jusqu'à ce qu'elle émette un clic et qu'elle sorte, puis en la tirant. Lorsque vous remettez la carte en place, assurez-vous que les contacts métalliques brillants sont tournés vers le bas ; la carte doit s'insérer d'un léger clic.

Les deux ports suivants sont les ports micro-HDMI, pour la connexion à un moniteur, un écran TV ou un autre écran. Tout comme le Raspberry Pi 4, le Raspberry Pi 400 prend en charge jusqu'à deux écrans. À côté de ces deux ports se trouve le port d'alimentation USB de type C, pour la connexion au bloc d'alimentation Raspberry Pi ou à un bloc d'alimentation compatible USB.

Les deux ports bleus sont des ports USB 3.0, qui permettent une connexion à haut débit à des appareils tels que des disques durs, des clés USB, des imprimantes, etc. Le port blanc à droite est un port USB 2.0 à faible débit, auquel vous pouvez connecter la souris Raspberry Pi.

Le dernier port est un port réseau Gigabit Ethernet, qui vous permet de connecter votre Raspberry Pi 400 à votre réseau à l'aide d'un câble réseau, au lieu d'utiliser le réseau sans fil Wi-Fi intégré. Nous reviendrons sur la connexion de votre Raspberry Pi 400 à un réseau dans le **Chapitre 2, Prise en main de Raspberry Pi**.

Chapitre 2

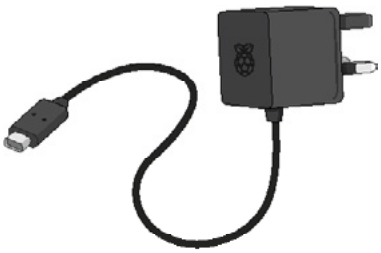
Prise en main de Raspberry Pi

Découvrez quels sont les éléments essentiels dont votre Raspberry Pi a besoin, comment les connecter et les mettre en marche pour un fonctionnement optimal

Votre Raspberry Pi a été conçu pour être aussi rapide et simple d'installation et d'utilisation que possible, mais, comme tout ordinateur, il dépend d'autres composants externes, appelés *périphériques*. Il est vrai qu'en observant la carte circuit de votre Raspberry Pi, qui ne ressemble à aucun des ordinateurs fermés que nous sommes habitués à voir, on pourrait ressentir une vague inquiétude à l'idée que les choses vont forcément se corser. Rassurez-vous, ce n'est pas le cas. Votre Raspberry Pi peut être parfaitement opérationnel en moins de dix minutes en suivant simplement les étapes décrites dans ce guide.

Si vous recevez ce livre dans un kit de bureau Raspberry Pi ou accompagnant un Raspberry Pi 400, vous avez déjà presque tout ce dont vous avez besoin pour commencer : il vous suffit de vous procurer un écran d'ordinateur ou de télévision doté d'une connexion HDMI (le même type de connexion utilisé par les décodeurs, les lecteurs Blu-ray ou les consoles de jeux) pour commencer à découvrir les pouvoirs de votre Raspberry Pi.

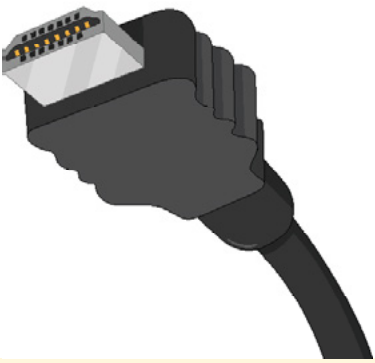
Si votre Raspberry Pi n'est doté d'aucun accessoire, vous devez donc vous procurer également :



- **Alimentation USB** : alimentation 5 V d'une intensité de 3 ampères (3 A) dotée d'un connecteur USB de type C. Le bloc d'alimentation officiel Raspberry Pi est toujours recommandé, car il s'adapte aux demandes de puissance changeantes du Raspberry Pi.

- **carte microSD avec NOOBS** : la carte microSD fait office de stockage permanent de votre Raspberry Pi ; tous les fichiers que vous créez et les logiciels que vous installez, ainsi que le système d'exploitation lui-même, y sont stockés. Une carte de 8 Go suffit pour commencer, mais vous aurez besoin d'une carte de 16 Go pour vous développer. L'utilisation d'une carte dotée du logiciel NOOBS (New Out-Of-Box Software) préinstallé vous fera gagner du temps ; dans le cas contraire, consultez l'**Annexe A** pour obtenir des instructions sur l'installation d'un système d'exploitation sur une carte vierge.

- **Clavier et souris USB** : le clavier et la souris vous permettent de contrôler votre Raspberry Pi. Presque tous les claviers et souris câblés ou sans fil dotés d'un connecteur USB sont compatibles avec le Raspberry Pi, bien que certains claviers gaming avec des lumières colorées sont trop énergivores pour être utilisés de manière fiable.



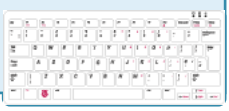
- **Câble micro-HDMI** : ce câble transmet le son et les images du Raspberry Pi vers votre écran TV ou votre moniteur. Une extrémité du câble est dotée d'un connecteur micro-HDMI pour Raspberry Pi, l'autre d'un connecteur HDMI qui se connecte à votre écran. Vous pouvez également utiliser un adaptateur micro-HDMI vers HDMI et un câble HDMI standard de taille normale. Si vous utilisez un moniteur sans prise HDMI, vous pouvez acheter des adaptateurs micro-HDMI vers DVI-D, DisplayPort ou VGA. Pour le connecter à un téléviseur plus ancien qui utilise la vidéo composite ou possède une prise Péritel, utilisez un câble audio/vidéo 3,5 mm TRRS.

Votre Raspberry Pi peut être utilisé en toute sécurité sans boîtier, à condition de ne pas le placer sur une surface métallique conductrice d'électricité qui pourrait provoquer un court-circuit. Un boîtier optionnel peut toutefois offrir une protection supplémentaire ; le Desktop Kit comprend le boîtier officiel Raspberry Pi, tandis que des boîtiers de tiers sont disponibles chez tous les bons revendeurs.

Si vous souhaitez utiliser votre Raspberry Pi sur un réseau câblé, plutôt que sur un réseau sans fil (WiFi), vous aurez également besoin d'un câble réseau. Celui-ci doit être connecté à une extrémité au commutateur ou au routeur de votre réseau. Si vous prévoyez d'utiliser le système sans fil intégrée de Raspberry Pi, vous n'aurez pas besoin de câble ; vous devrez cependant connaître le nom et la clé ou le mot de passe de votre réseau sans fil.

INSTALLATION DU RASPBERRY PI 400

Les instructions ci-après concernent l'installation d'un Raspberry Pi 4 ou d'une autre carte mère de la famille Raspberry Pi. Les instructions pour l'installation du Raspberry Pi 400 se trouvent à la page 32.



Installation du matériel

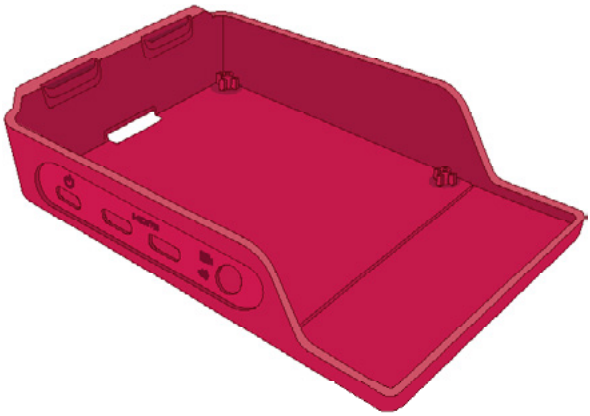
Commencez par déballer votre Raspberry Pi. Si le matériel de votre Raspberry Pi est robuste, il n'est pas indestructible : prenez l'habitude de tenir la carte par les bords, plutôt que de toucher les côtés plats, et touchez les tiges métalliques en relief avec délicatesse. Si ces tiges se plient, au mieux, cela rendra difficile l'utilisation de cartes d'extension et d'autres matériels supplémentaires ; au pire, cela peut provoquer un court-circuit qui endommagera votre Raspberry Pi.

Si vous ne l'avez pas encore fait, consultez le **Chapitre 1, Premier pas avec Raspberry Pi** pour savoir exactement où se trouvent les différents ports et quelle est leur fonction.

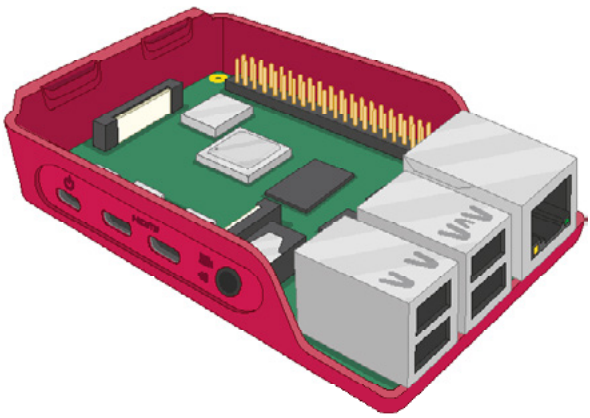
Montage du boîtier

Si vous installez Raspberry Pi dans un boîtier, effectuez cette étape en premier. Si vous utilisez un boîtier Raspberry Pi, commencez par le diviser en deux parties distinctes : le socle (rouge) et le couvercle (blanc).

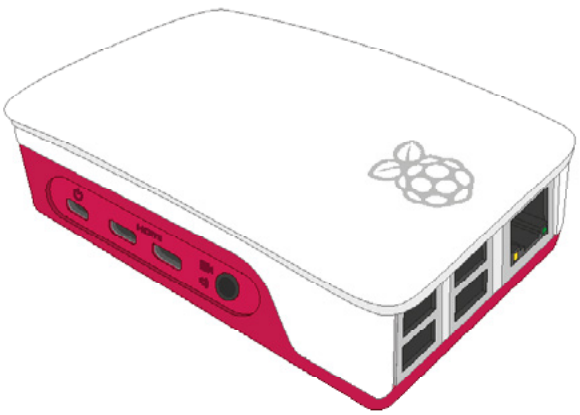
1 **Prenez le socle et tenez-le de manière à ce que l'extrémité surélevée soit à votre gauche et l'extrémité plate à votre droite.**



2 **En soutenant votre Raspberry Pi (sans carte microSD insérée) par ses ports USB et Ethernet, en formant un angle léger, insérez ses connecteurs (USB Type-C, 2 micro-HDMI, et AV 3,5 mm) dans leurs orifices respectifs sur le côté du socle, puis abaissez délicatement l'autre côté jusqu'à ce qu'il soit à plat.**

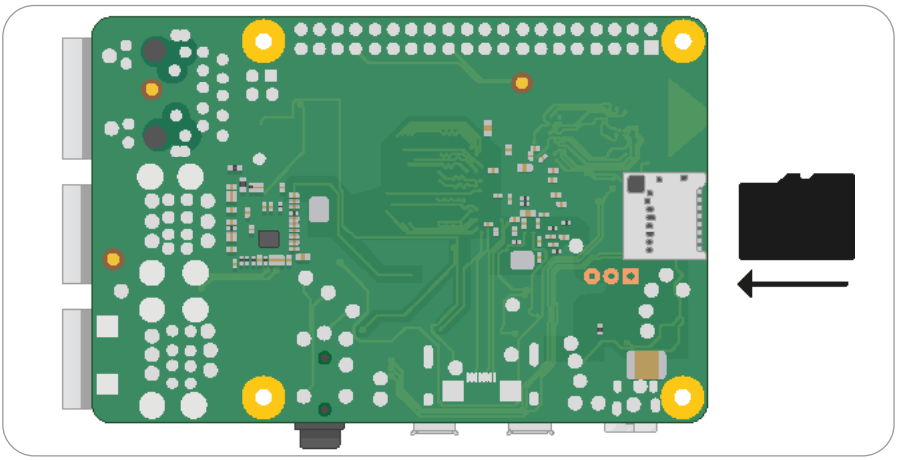


3 **Prenez le couvercle blanc et placez les deux clips sur la gauche dans les orifices correspondants sur la gauche du socle, au-dessus de la fente pour carte microSD. Lorsqu'ils sont en place, appuyez sur le côté droit (au-dessus des ports USB) jusqu'à ce que vous entendiez un clic.**

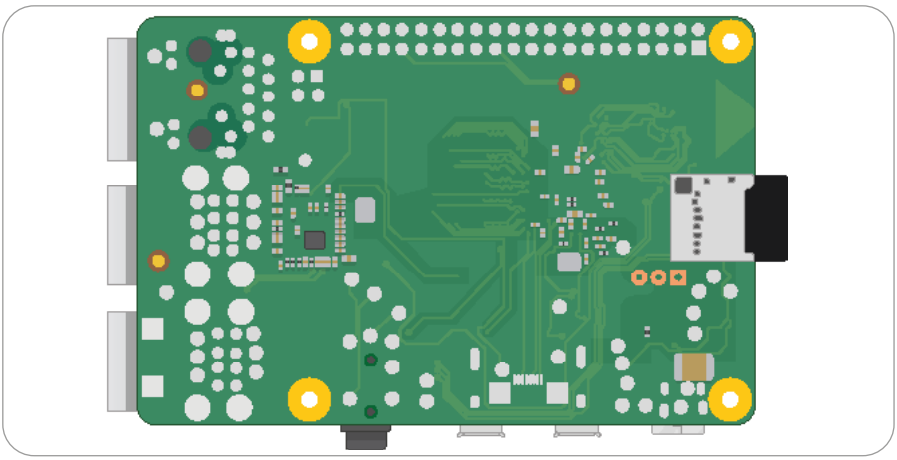


Connexion de la carte microSD

Pour installer la carte microSD, qui constitue l'unité de *stockage* de votre Raspberry Pi, retournez-le (dans son boîtier, le cas échéant) et insérez la carte dans la fente microSD, l'étiquette étant tournée dans la direction opposée du Raspberry Pi. La carte ne rentre que dans un sens et sans besoin d'exercer une pression excessive.



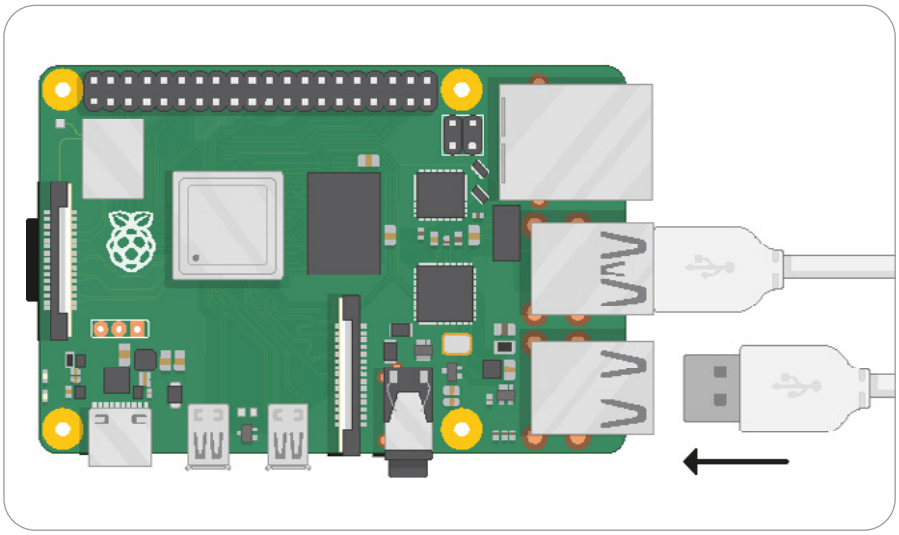
La carte microSD s'insère dans le connecteur, puis s'arrête sans clic.



Si vous souhaitez la retirer à l'avenir, il vous suffira de saisir l'extrémité de la carte en tirant délicatement. Si vous utilisez un ancien modèle de Raspberry Pi, vous devrez d'abord appuyer légèrement sur la carte pour la déverrouiller ; cette opération n'est pas nécessaire avec un Raspberry Pi 3 ou 4.

Connexion d'un clavier et d'une souris

Connectez le câble USB du clavier à l'un des quatre ports USB (2.0 ou 3.0) de votre Raspberry Pi. Si vous utilisez le clavier officiel de Raspberry Pi, il possède un port USB à l'arrière destiné à la souris ; dans le cas contraire, connectez simplement le câble USB de votre souris à un autre port USB du Raspberry Pi.



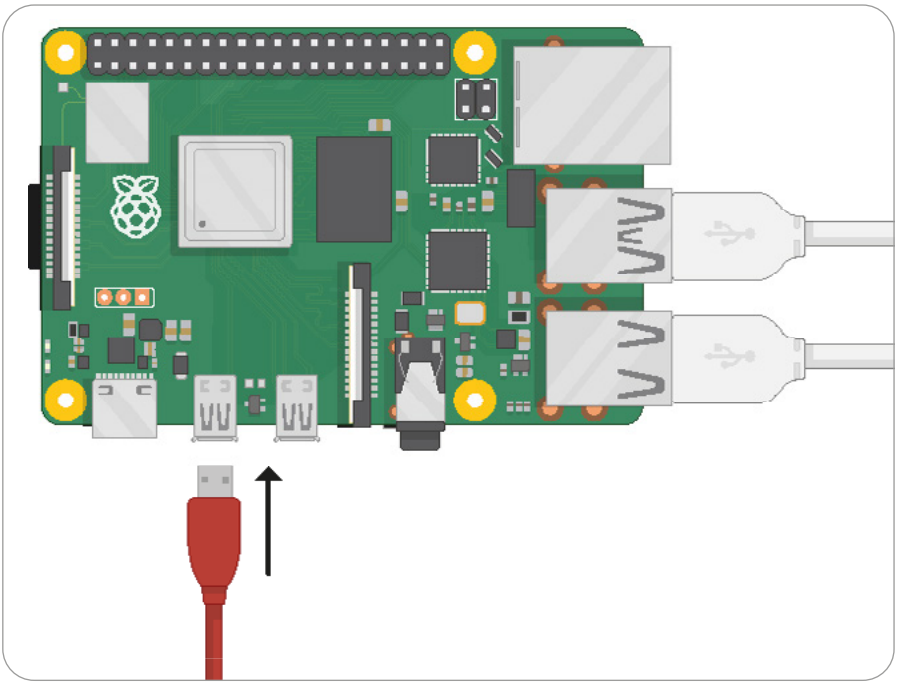
Les connecteurs USB du clavier et de la souris doivent s'insérer sans exercer de pression excessive ; si vous devez forcer pour insérer le connecteur, cela signifie qu'il y a quelque chose d'anormal. Vérifiez que le connecteur USB est dans le bon sens !

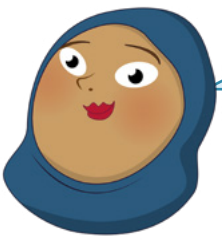
CLAVIER ET SOURIS

Le clavier et la souris sont les principaux moyens de transmettre des instructions à votre Raspberry Pi ; en informatique, ils sont désignés par le terme de *dispositifs d'entrée* pour les différencier de l'écran qui est un *dispositif de sortie*.

Connexion d'un écran

Prenez le câble micro-HDMI et connectez la plus petite extrémité au port micro-HDMI le plus proche du port USB de type C de votre Raspberry Pi, et l'autre extrémité à votre écran. Si votre écran dispose de plus d'un port HDMI, cherchez un numéro de port à côté du connecteur lui-même ; vous devrez basculer le téléviseur sur cette entrée pour voir l'écran de Raspberry Pi. Si vous ne voyez pas de numéro de port, ne vous inquiétez pas : vous pouvez simplement passer d'une entrée à l'autre jusqu'à ce que vous trouviez l'entrée du Raspberry Pi.



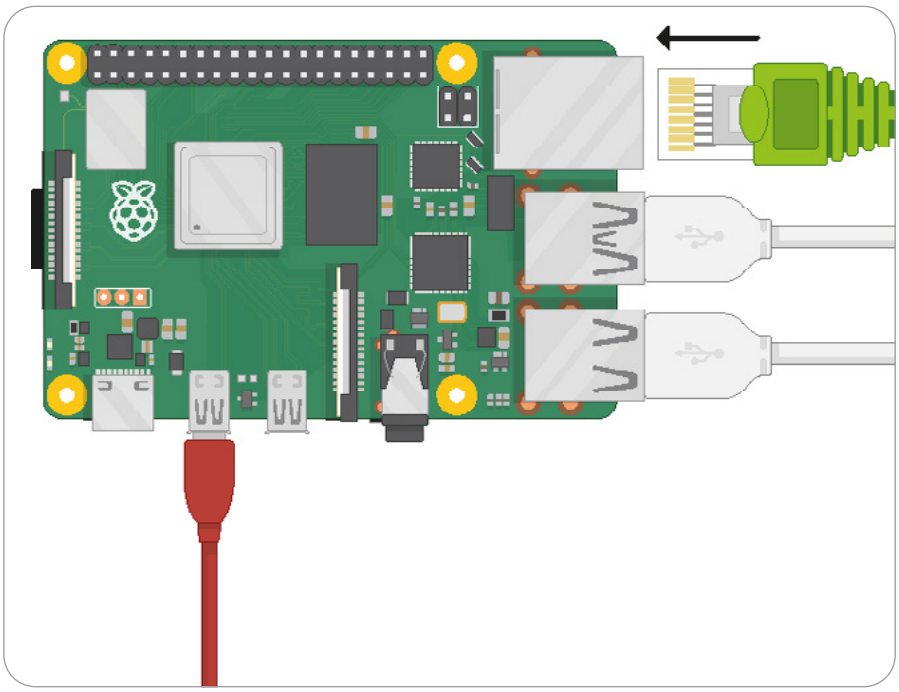


CONNEXION TV

Si votre écran TV ou votre moniteur ne dispose pas d'un port HDMI, cela ne signifie pas que vous ne pouvez pas utiliser Raspberry Pi. Des câbles adaptateurs, disponibles chez tous les revendeurs d'électronique, vous permettront de convertir le port micro-HDMI de votre Raspberry Pi en port DVI-D, DisplayPort, ou VGA pour une utilisation avec des moniteurs plus anciens ; il suffit de les connecter au port micro-HDMI du Raspberry Pi, puis d'utiliser un câble approprié pour relier le câble adaptateur au moniteur. Si votre écran TV ne dispose que d'une entrée vidéo composite ou Péritel, vous pouvez acheter des câbles adaptateurs 3,5 mm TRRS et des adaptateurs Péritel vers RCA qui se connectent à la prise AV 3,5 mm.

Connexion d'un câble réseau (facultatif)

Pour connecter votre Raspberry Pi à un réseau câblé, prenez un câble réseau (désigné par le terme câble Ethernet) et branchez-le au port Ethernet du Raspberry Pi, en ayant soin de placer le clip en plastique vers le bas, jusqu'à ce que vous entendiez un clic. Si vous devez retirer le câble, il vous suffit de presser le clip en plastique vers l'intérieur, en direction de la prise, et de faire glisser doucement le câble pour le libérer.

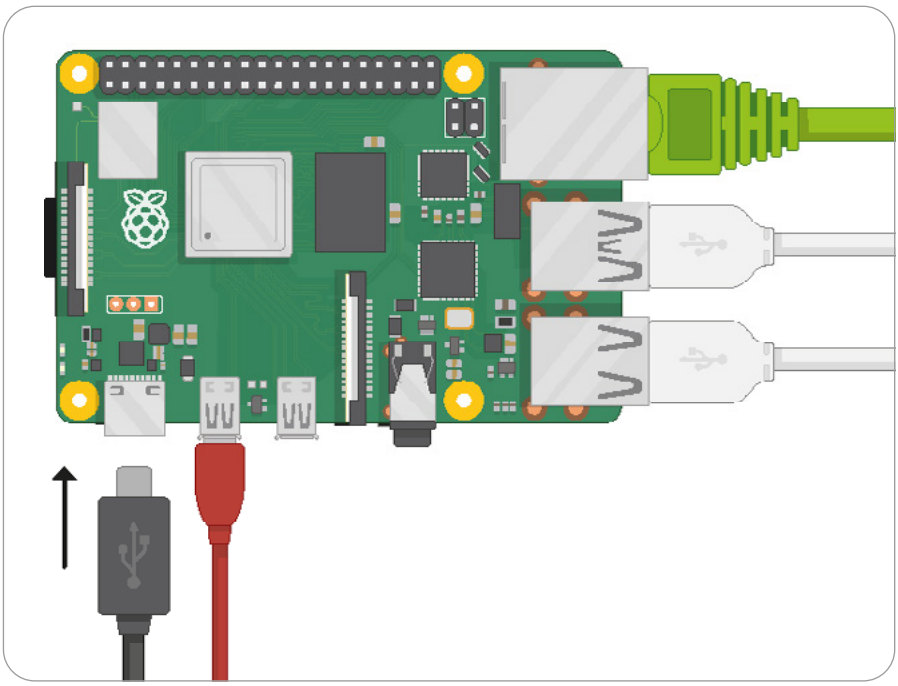


L'autre extrémité de votre câble réseau doit être connectée en suivant la même procédure à n'importe quel port libre de votre concentrateur, commutateur ou routeur de réseau.

Connexion de l'alimentation

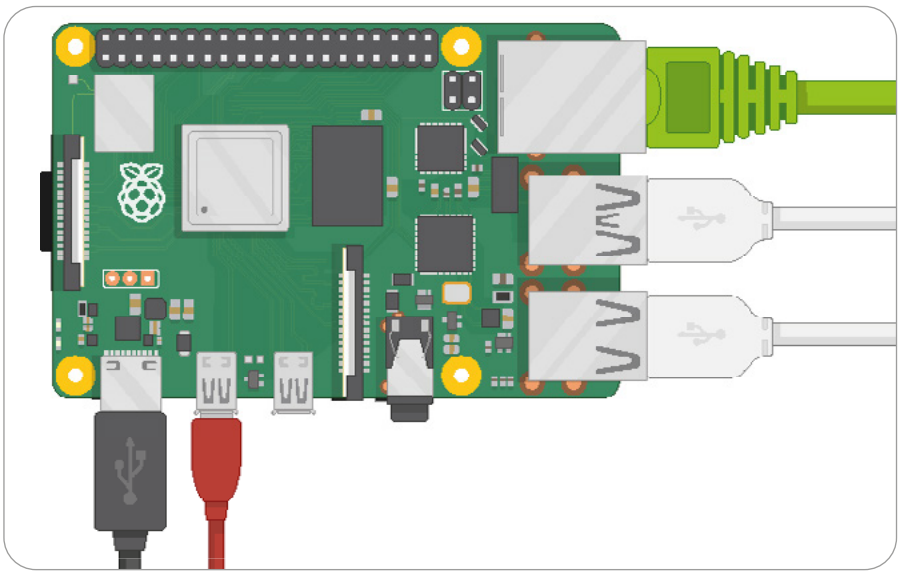
La toute dernière étape du processus d'installation de votre Raspberry Pi consiste à le connecter à une source d'alimentation électrique, et vous ne devez le faire que lorsque vous êtes prêt à installer son logiciel : le Raspberry Pi n'est pas doté d'un interrupteur et se met en tension dès qu'il est connecté à une source d'alimentation électrique.

En premier lieu, connectez l'extrémité USB Type-C du câble d'alimentation au connecteur d'alimentation USB Type-C sur votre Raspberry Pi. Il peut entrer dans n'importe quel sens et doit pouvoir s'insérer facilement. Si votre alimentation électrique est équipée d'une rallonge détachable, assurez-vous que l'autre extrémité est branchée sur le bloc d'alimentation.



Ensuite, connectez le câble d'alimentation à une prise et mettez la prise en tension ; votre Raspberry Pi commencera immédiatement à fonctionner.

Félicitations : vous avez monté votre Raspberry Pi !



Quatre logos Raspberry Pi s'afficheront brièvement sur la partie supérieure gauche d'un écran noir et il est possible qu'un écran bleu apparaisse pendant que le logiciel se redimensionne afin de tirer pleinement parti de votre carte microSD. Si vous voyez un écran noir, attendez quelques minutes ; à sonpremier démarrage, votre Raspberry Pi doit faire un peu de ménage en arrière-plan. Après un certain temps, le bureau et l'assistant d'installation de Raspberry Pi OS s'affichent, tel que représenté dans la **Figure 2-1**. Votre système d'exploitation est maintenant prêt à être configuré, ce qui vous sera expliqué dans le **Chapitre 3, Utiliser votre Raspberry Pi**.

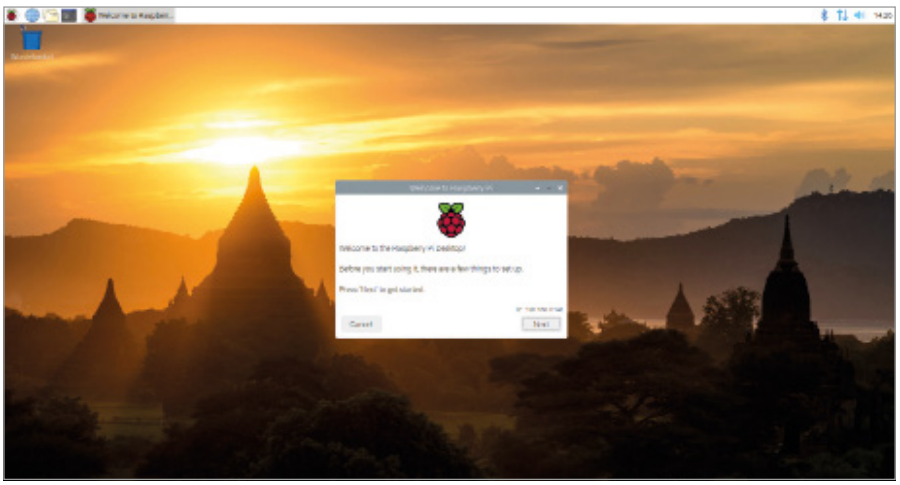
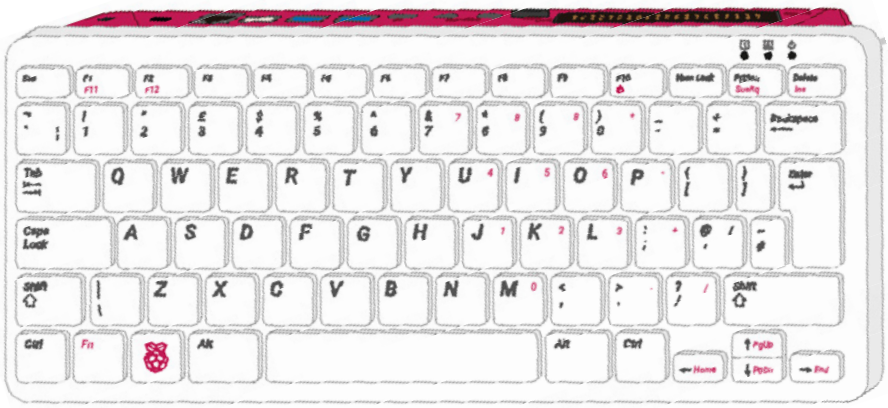


Figure 2-1 : Le bureau et l'assistant de configuration de Raspberry Pi OS

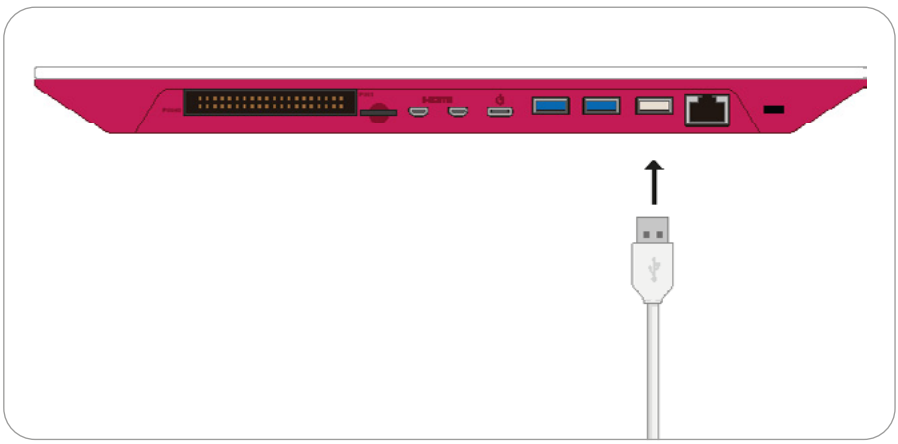
Configuration du Raspberry Pi 400

Contrairement au Raspberry Pi 4, le Raspberry Pi 400 est livré avec un clavier intégré et une carte microSD préalablement installée. Il vous faudra toutefois brancher quelques câbles pour commencer, mais cela ne devrait vous prendre que quelques minutes.



Connexion de la souris

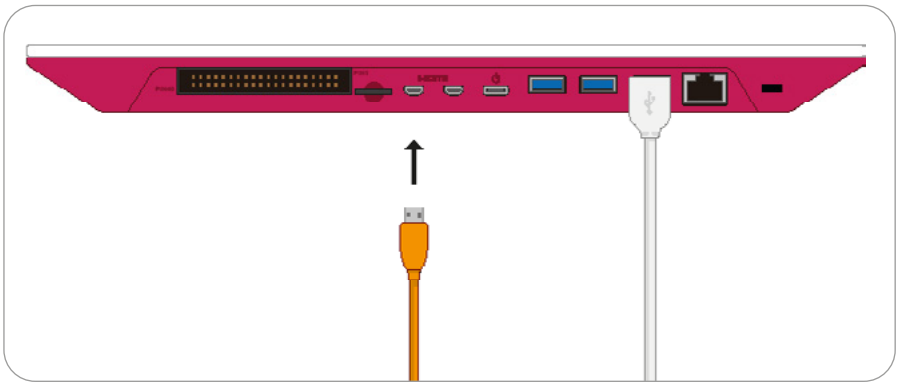
Le clavier du Raspberry Pi 400 étant déjà connecté, il ne vous reste plus qu'à connecter la souris. Prenez le câble USB à l'extrémité de la souris et insérez-le dans l'un des trois ports USB (2.0 ou 3.0) à l'arrière du Raspberry Pi 400. Si vous souhaitez conserver les deux ports USB 3.0 à haut débit pour d'autres accessoires, utilisez le port blanc.



Le connecteur USB doit s'insérer sans exercer de pression excessive ; si vous devez forcer pour insérer le connecteur, cela signifie qu'il y a quelque chose d'anormal. Vérifiez que le connecteur USB est dans le bon sens !

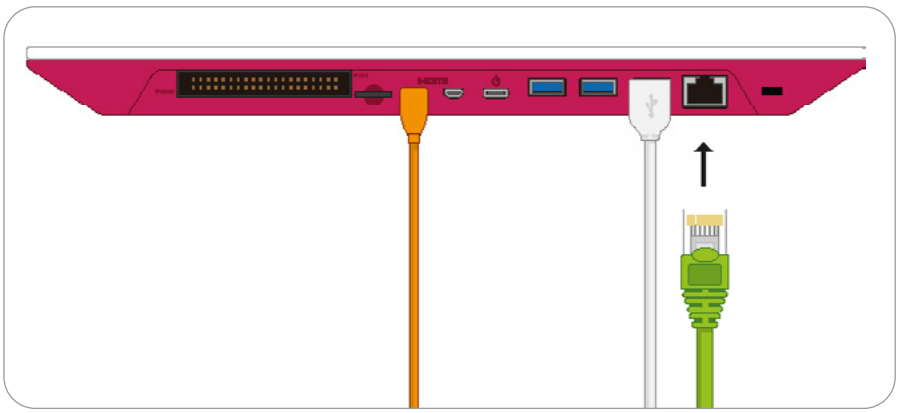
Connexion d'un écran

Prenez le câble micro-HDMI et connectez la plus petite extrémité au port micro-HDMI le plus proche de la fente de la carte microSD de votre Raspberry Pi 400, et l'autre extrémité à votre écran. Si votre écran dispose de plus d'un port HDMI, cherchez un numéro de port à côté du connecteur lui-même ; vous devrez basculer le téléviseur sur cette entrée pour voir l'écran de Raspberry Pi. Si vous ne voyez pas de numéro de port, ne vous inquiétez pas : vous pouvez simplement passer d'une entrée à l'autre jusqu'à ce que vous trouviez l'entrée du Raspberry Pi.



Connexion d'un câble réseau (facultatif)

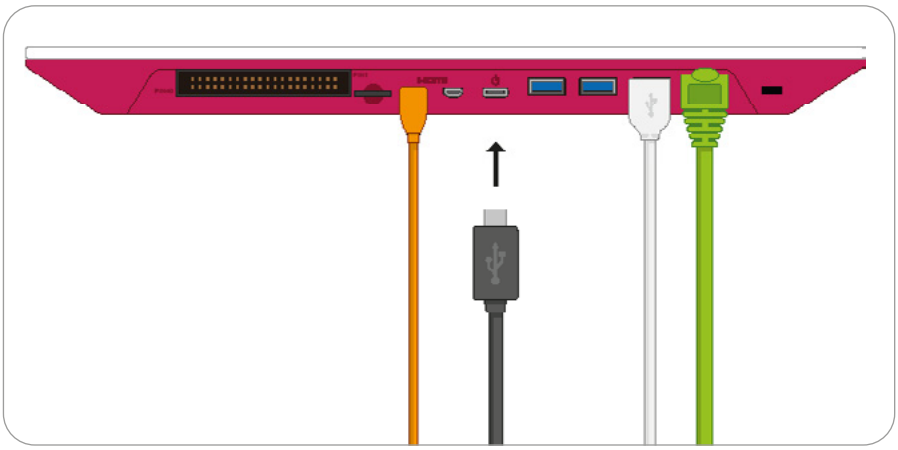
Pour connecter votre Raspberry Pi 400 à un réseau câblé, prenez un câble réseau (désigné par le terme câble Ethernet) et branchez-le au port Ethernet du Raspberry Pi 400, en ayant soin de placer le clip en plastique vers le haut, jusqu'à ce que vous entendiez un clic. Si vous devez retirer le câble, il vous suffit de presser le clip en plastique vers l'intérieur, en direction de la prise, et de faire glisser doucement le câble pour le libérer.



L'autre extrémité de votre câble réseau doit être connectée en suivant la même procédure à n'importe quel port libre de votre concentrateur, commutateur ou routeur de réseau.

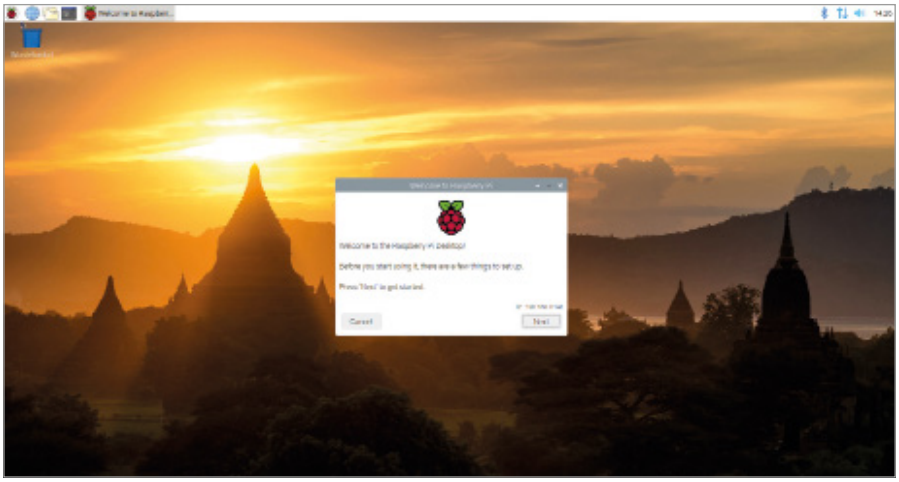
Connexion de l'alimentation

La toute dernière étape du processus d'installation de votre Raspberry Pi 400 consiste à le connecter à une source d'alimentation électrique, et vous ne devez le faire que lorsque vous êtes prêt à installer son logiciel : Le Raspberry Pi 400 n'est pas doté d'un interrupteur et se met en tension dès qu'il est connecté à une source d'alimentation électrique. En premier lieu, connectez l'extrémité USB Type-C du câble d'alimentation au connecteur d'alimentation USB Type-C sur votre Raspberry Pi. Il peut entrer dans n'importe quel sens et doit pouvoir s'insérer facilement. Si votre alimentation électrique est équipée d'une rallonge détachable, assurez-vous que l'autre extrémité est branchée sur le bloc d'alimentation.



Ensuite, connectez le câble d'alimentation à une prise et mettez la prise en tension ; votre Raspberry Pi 400 commencera immédiatement à fonctionner. Félicitations : vous avez monté votre Raspberry Pi 400 !

Quatre logos Raspberry Pi s'afficheront brièvement sur la partie supérieure gauche d'un écran noir et il est possible qu'un écran bleu apparaisse pendant que le logiciel se redimensionne afin de tirer pleinement parti de votre carte microSD. Si vous voyez un écran noir, attendez quelques minutes ; à son premier démarrage, votre Raspberry Pi doit faire un peu de ménage en arrière-plan. Après un certain temps, le bureau et l'assistant d'installation de Raspberry Pi OS s'affichent, tel que représenté dans la **Figure 2-2**. Votre système d'exploitation est maintenant prêt à être configuré, ce qui vous sera expliqué dans le **Chapitre 3, Utiliser votre Raspberry Pi**.

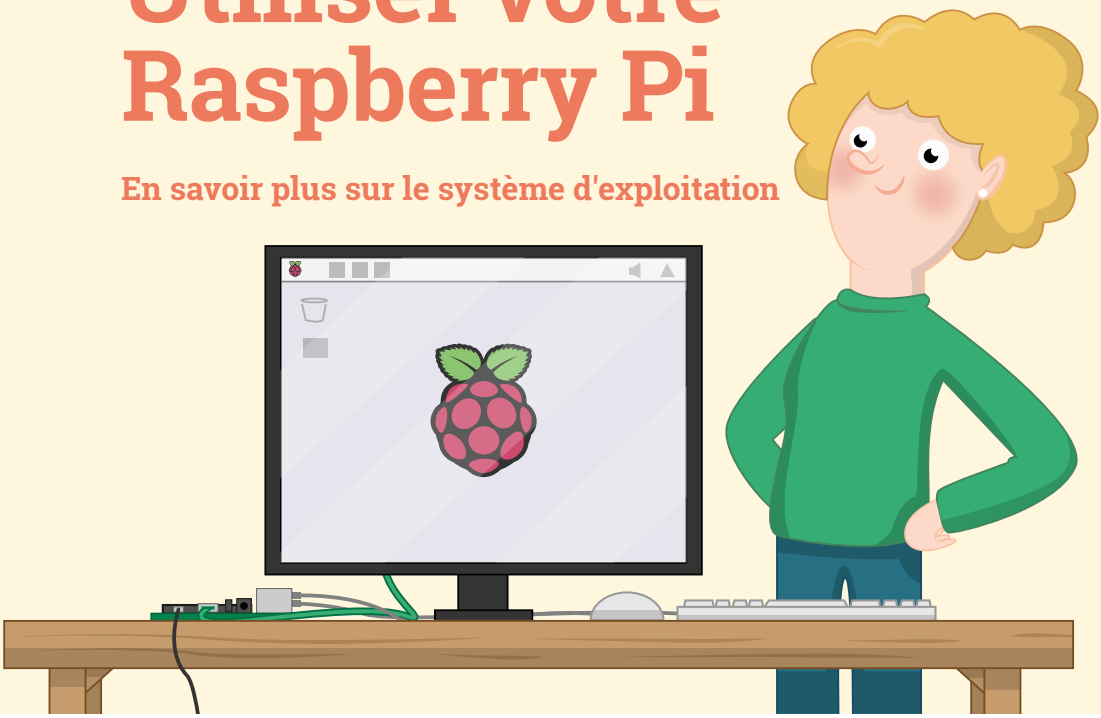


▲ **Figure 2-2 :** Le bureau et l'assistant de configuration de Raspberry Pi OS

Chapitre 3

Utiliser votre Raspberry Pi

En savoir plus sur le système d'exploitation

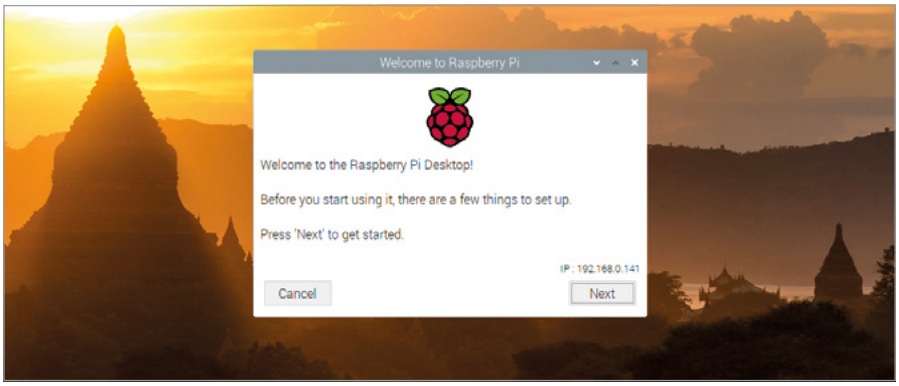


Raspberry Pi est en mesure d'exécuter une large gamme de logiciels, y compris plusieurs systèmes d'exploitation, c'est-à-dire le logiciel de base qui permet à l'ordinateur de fonctionner. Entre les logiciels, le plus connu est le système d'exploitation officiel de la Raspberry Pi Foundation, à savoir Raspberry Pi OS. Basé sur Debian Linux, il a été développé spécialement pour Raspberry Pi et il est doté d'une gamme d'options préinstallées et prêtes à l'emploi.

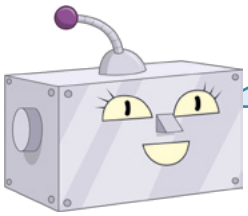
Si vous n'avez jamais utilisé que Microsoft Windows ou Apple MacOS, vous n'avez aucun souci à vous faire : Raspberry Pi OS est basé sur les mêmes principes de fenêtres, icônes, menus et pointeurs (WIMP), et vous devriez pouvoir faire connaissance très rapidement. Le chapitre ci-après vous aidera à démarrer et vous présentera certains des logiciels proposés.

Le Welcome Wizard

La première fois que vous exécutez Raspberry Pi OS, vous ferez la rencontre du Welcome Wizard (**Figure 3-1**). Cet outil utile vous aidera à modifier certains paramètres de Raspberry Pi OS, connus sous le nom de *configuration* afin de définir où et comment vous comptez utiliser votre Raspberry Pi.



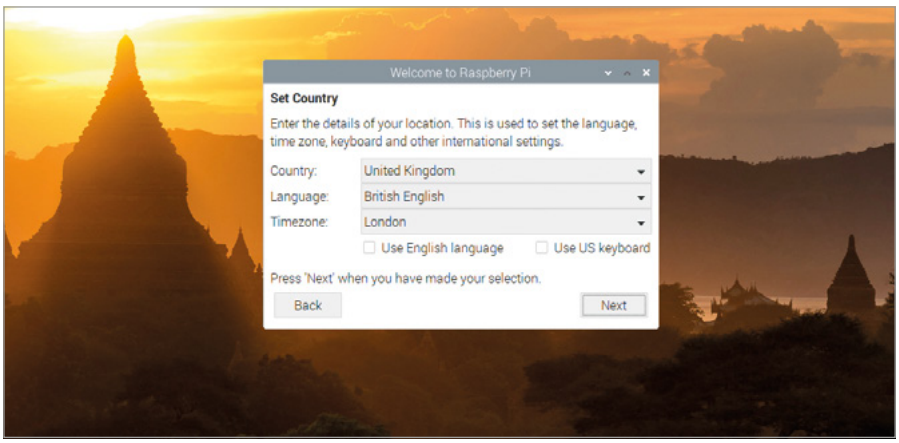
▲ **Figure 3-1 :** Le Welcome Wizard



FERMER LE WIZARD

Vous pouvez choisir de fermer le Welcome Wizard en cliquant sur le bouton Annuler, mais certaines fonctionnalités de Raspberry Pi, comme le réseau sans fil, ne fonctionneront pas tant que vous n'aurez pas répondu au moins à la première série de questions.

Cliquez sur le bouton « Next » puis choisissez votre pays, votre langue et votre fuseau horaire en cliquant tour à tour sur chaque liste déroulante et en sélectionnant votre réponse dans la liste (**Figure 3-2**). Si vous utilisez un clavier de format américain, cliquez sur la case à cocher pour vous assurer que Raspberry Pi OS utilise le bon format de clavier. Si vous souhaitez que les affichages du bureau et des programmes soient en anglais, quelle que soit la langue maternelle de votre pays, cochez la case « Use English language ». Lorsque vous avez terminé, cliquez sur Next.



▲ **Figure 3-2 :** Sélection d'une langue, entre autres options

L'écran suivant vous demandera de modifier le mot de passe de l'utilisateur « pi » (par rapport au mot de passe par défaut « raspberry »). Pour des raisons de sécurité, c'est une excellente idée que d'en créer un nouveau. Saisissez-le dans les cases désignées (**Figure 3-3**). Lorsque vous êtes satisfait, cliquez sur Next.

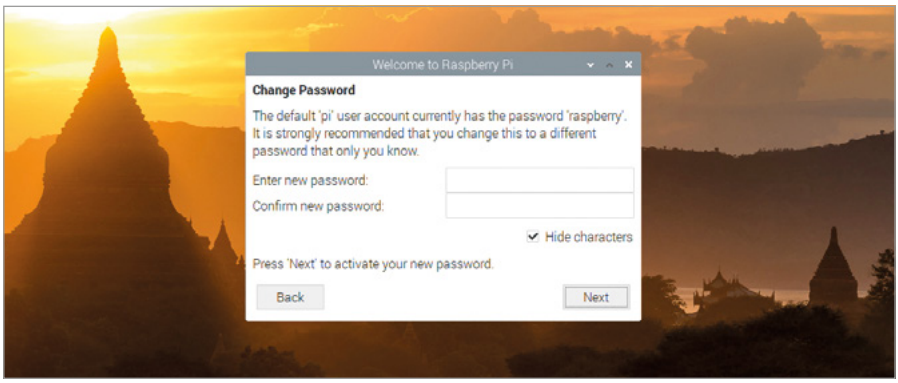


Figure 3-3 : Définition d'un nouveau mot de passe

L'écran suivant vous demande s'il y a une bordure noire tout autour de l'écran (**Figure 3-4**). Si le bureau de la Raspberry Pi remplit la totalité de votre écran TV ou de votre moniteur, ne cochez pas la case ; cochez la case, en revanche, si une bordure noire encadre le bureau et celui-ci est plus étroit que votre écran ou moniteur. Lorsque vous êtes prêt à passer à l'étape suivante, cliquez sur Next.

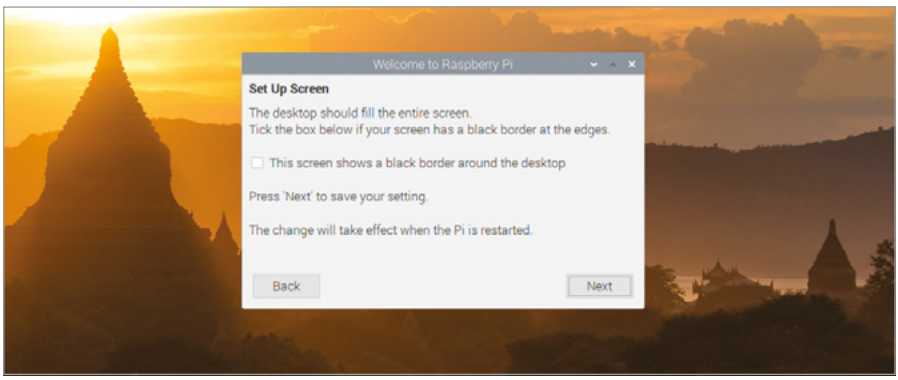


Figure 3-4 : Vérifier qu'il n'y a pas de bordure noire

L'écran suivant vous permettra de sélectionner votre réseau WiFi dans une liste (**Figure 3-5**). Faites défiler la liste des réseaux à l'aide de la souris ou du clavier, trouvez le nom de votre réseau, cliquez dessus, puis cliquez sur Next. En supposant que votre réseau sans fil soit sécurisé (il devrait l'être, à priori), son mot de passe (également appelé clé pré-partagée) vous sera demandé ; celui-ci est normalement inscrit sur une fiche qui accompagne le routeur ou

LES RÉSEAUX SANS FIL

Le réseau sans fil intégré est uniquement disponible sur les familles Raspberry Pi 3, Pi 4 et Pi Zero W. Si vous souhaitez utiliser un autre modèle de Raspberry Pi avec un réseau sans fil, vous aurez besoin d'un adaptateur USB WiFi.

sur le fond du routeur lui-même. Cliquez sur Next pour vous connecter au réseau. Si vous ne souhaitez pas vous connecter à un réseau sans fil, cliquez sur Skip.

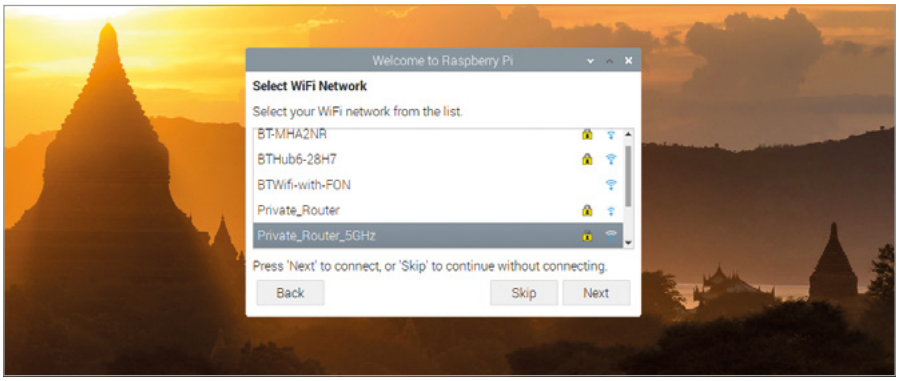


Figure 3-5 : Choisir un réseau sans fil

L'écran suivant vous permettra de vérifier et d'installer des mises à jour pour Raspberry Pi OS et les autres logiciels sur Raspberry Pi (**Figure 3-6**). Raspberry Pi OS est régulièrement mis à jour pour corriger les bugs, ajouter de nouvelles fonctionnalités et améliorer les performances. Pour installer ces mises à jour, cliquez sur Suivant ; sinon, cliquez sur Ignorer. Soyez patient, le téléchargement des mises à jour peut prendre plusieurs minutes. Lorsque les mises à jour sont installées, une fenêtre indiquant que « Le système est à jour » s'affiche ; cliquez sur le bouton OK.

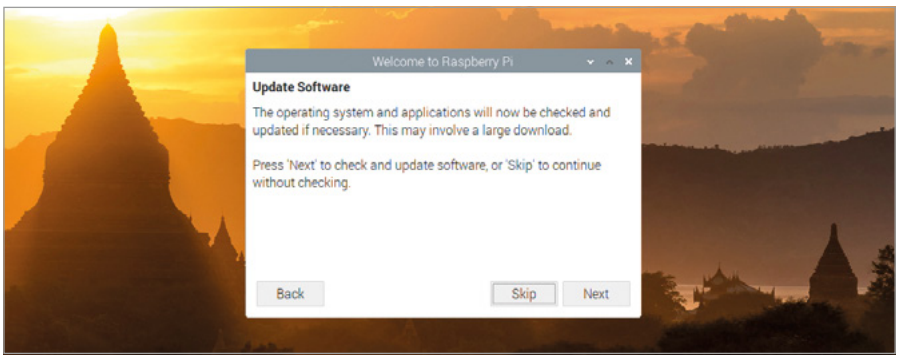
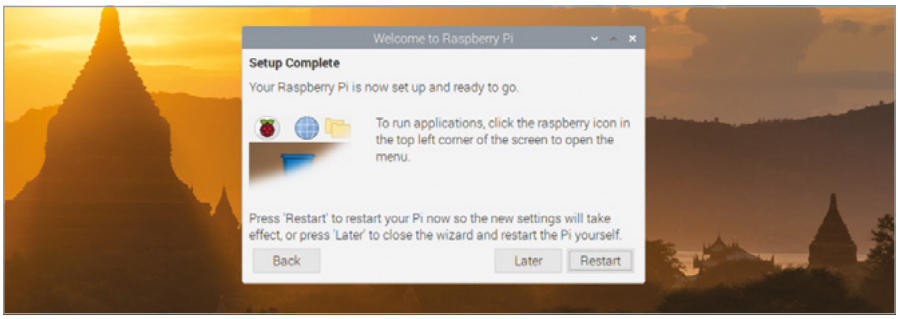


Figure 3-6 : Vérifier la disponibilité de mises à jour

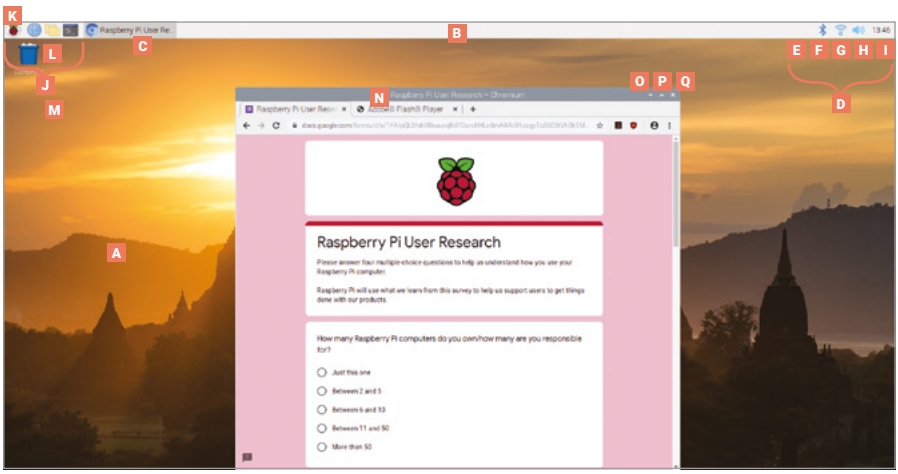
L'écran final du Welcome Wizard (**Figure 3-7**) exécute une tâche très simple : certaines modifications apportées ne prendront effet que lorsque vous redémarrerez votre Raspberry Pi (un processus également appelé redémarrage ou rebooting). Si vous y êtes invité, cliquez sur le bouton Restart et Raspberry Pi redémarrera. À partir de là, le Welcome Wizard n'apparaîtra plus ; son travail est terminé et votre Raspberry Pi est prêt à être utilisé.



▲ **Figure 3-7 :** Redémarrer Raspberry Pi

Explorer le bureau

La version de Raspberry Pi OS installée sur la plupart des cartes Raspberry Pi est connue sous le nom de « Raspberry Pi OS with desktop », en référence à son interface utilisateur graphique principale (**Figure 3-8**). Le bureau est occupé en bonne mesure par une image, appelée fond d'écran (**A** dans **Figure 3-8**), sur laquelle les programmes que vous exécutez apparaîtront. Sur la partie supérieure du bureau se trouve une barre de tâches (**B**), qui vous permet de charger effectivement chacun des programmes ; ceux-ci sont alors indiqués par des tâches (**C**) dans la barre des tâches.



▲ **Figure 3-8 :** Le bureau Raspberry Pi OS

- **A** Fond d'écran
- **B** Barre des tâches
- **C** Tâche
- **D** Barre d'état système
- **E** Éjection des médias
- **F** Icône Bluetooth
- **G** Icône Réseau
- **H** Icône volume
- **I** Horloge
- **J** Lanceur
- **K** Icône menu (ou framboise)
- **L** Icône Corbeille
- **M** Icône lecteur amovible
- **N** Barre de titre de la fenêtre
- **O** Minimiser
- **P** Maximiser
- **Q** Fermer

Le côté droit de la barre de menu contient la *barre d'état système* (**D**). Si une *unité de stockage amovible*, comme une clé USB, est connectée à Raspberry Pi, vous verrez un symbole d'éjection (**E**) ; en cliquant dessus, vous pourrez l'éjecter et la retirer en toute sécurité. À l'extrême droite se trouve l'horloge (**I**) ; cliquez dessus pour afficher un calendrier numérique (**Figure 3-9**).

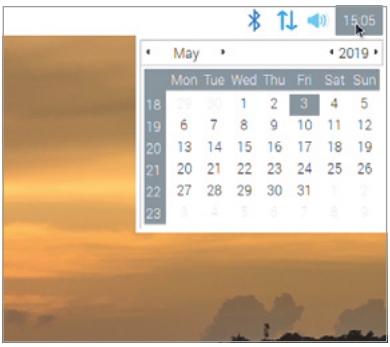


Figure 3-9 : Le calendrier numérique

Juste à côté, vous trouverez le symbole de haut-parleur (**H**) ; cliquez dessus avec le bouton gauche de la souris pour régler le volume audio de Raspberry Pi, ou cliquez avec le bouton droit de la souris pour choisir la sortie que Raspberry Pi doit utiliser. À côté de cela se trouve le symbole de réseau (**G**) ; si vous êtes connecté à un réseau sans fil, la puissance du signal est affichée sous la forme d'une série de barres, tandis que si vous êtes connecté à un réseau câblé, vous ne verrez que deux flèches. En cliquant sur l'icône du réseau, une liste des réseaux sans fil disponibles dans les environs s'affiche (**Figure 3-10**). En cliquant sur l'icône Bluetooth (**F**) juste à côté, vous pourrez vous connecter à un appareil Bluetooth à proximité.

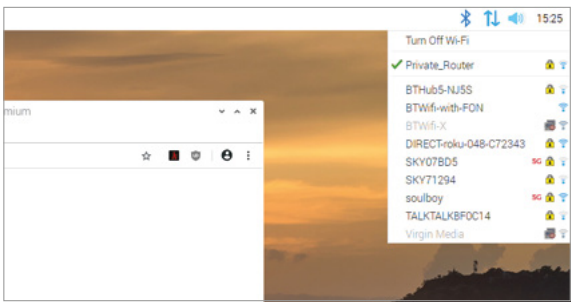


Figure 3-10 : Liste des réseaux sans fil des alentours

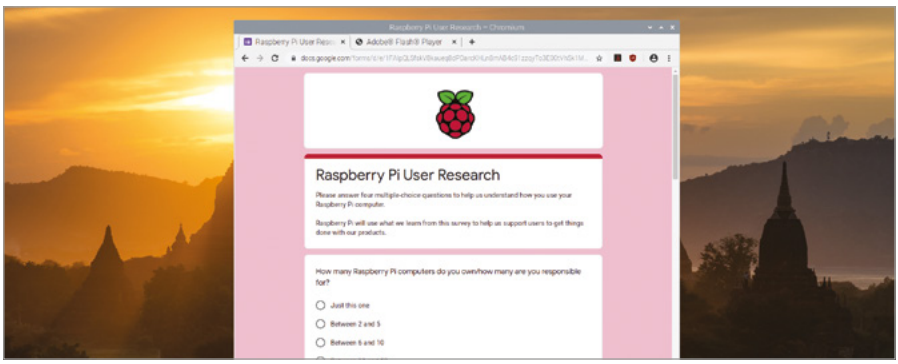
Dans la partie gauche de la barre de menu se trouve le *lanceur* (**J**), où se trouvent les programmes installés parallèlement à Raspberry PiOS. Certains de ces programmes sont visibles sous forme d'icônes de raccourcis, d'autres sont enfouis dans le menu mais vous pouvez les faire ressortir en cliquant sur l'icône en forme de framboise (**K**) tout à gauche (**Figure 3-11**, au verso).

▲ **Figure 3-11 :** Le menu de Raspberry Pi OS

Les programmes figurant au menu sont divisés en catégories, aux dénominations claires et précises : la catégorie Programmation, par exemple, contient des logiciels conçus pour vous aider à écrire vos propres programmes, comme expliqué à partir du **Chapitre 4, Programmation avec Scratch** alors que la catégorie Jeux vous aidera à vous divertir. Tous les programmes ne seront pas détaillés dans ce guide ; n'hésitez pas à les explorer pour en savoir plus.

Le navigateur Web Chromium

Pour vous entraîner à utiliser votre Raspberry Pi, commencez par charger le navigateur Web Chromium : cliquez sur l'icône en forme de framboise en haut à gauche pour afficher le menu, sélectionnez la catégorie Internet à l'aide la souris et chargez le navigateur Web Chromium en cliquant dessus (**Figure 3-12**).



▲ **Figure 3-12 :** Le navigateur Web Chromium

Si vous avez utilisé le navigateur Chrome de Google sur un autre ordinateur, vous serez immédiatement à l'aise avec Chromium. Avec le navigateur Web Chromium, vous pouvez visiter des sites Web, regarder des vidéos et jouer, et même communiquer avec des personnes du monde entier sur des forums et des sites de discussion.

Commencez à utiliser Chromium en maximisant sa fenêtre sur l'écran : recherchez les trois symboles en haut à droite de la barre de titre de la fenêtre Chromium (**N**) et cliquez sur le symbole du milieu de flèche vers le haut (**P**). Il s'agit du bouton *maximiser*, qui agrandit une fenêtre de sorte à ce qu'elle occupe la totalité de l'écran. À gauche du bouton maximiser se trouve *minimiser* (**O**), qui masque la fenêtre jusqu'à ce que vous cliquiez dessus dans la barre des tâches en haut de l'écran. La croix à droite du symbole de maximisation est *fermer* (**Q**) : aucune ambiguïté possible, elle ferme la fenêtre.

FERMER ET ENREGISTRER

Il n'est jamais recommandable de refermer une fenêtre sans avoir enregistré votre travail. La plupart des programmes vous demanderont si vous souhaitez enregistrer avant de cliquer sur fermer, mais d'autres ne vous le demanderont pas.

Cliquez sur la barre d'adresse en haut de la fenêtre Chromium (la grande barre blanche avec une loupe sur le côté gauche) et saisissez **www.raspberrypi.org**, puis appuyez sur la touche **ENTRÉE** sur votre clavier. Le site web de Raspberry Pi s'affiche (**Figure 3-13**). Vous pouvez également saisir des recherches dans la barre d'adresse : essayez de rechercher « Raspberry Pi », « Raspberry Pi OS » ou « Informatique et éducation ».

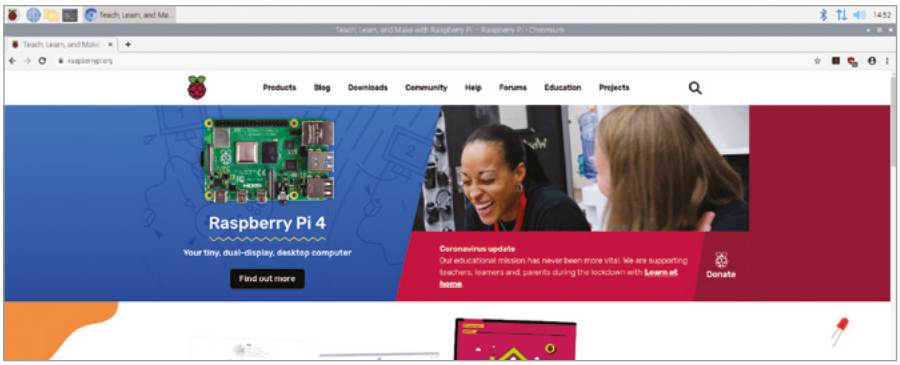


Figure 3-13 : Chargement du site Web Raspberry Pi sur Chromium

La première fois que vous chargez Chromium, il est possible que plusieurs *onglets* s'affichent dans la partie supérieure de la fenêtre. Pour passer d'un onglet à l'autre, cliquez dessus ; pour fermer un onglet sans fermer Chromium, cliquez sur la croix située sur le bord droit de l'onglet que vous souhaitez fermer. Pour ouvrir un nouvel onglet, ce qui est un moyen pratique d'ouvrir plusieurs sites Web sans avoir à gérer plusieurs fenêtres Chromium, cliquez sur le bouton d'onglet sur la droite du dernier onglet de la liste ou maintenez enfoncé en même temps le bouton **CTRL** et la touche **T**, sans lâcher **CTRL**.

Lorsque vous avez terminé avec Chromium, cliquez sur le bouton fermer dans la partie supérieure droite de la fenêtre.

Le gestionnaire de fichiers

Tous les fichiers que vous enregistrez, qu'il s'agisse de programmes, de vidéos ou d'images, finissent dans votre *répertoire personnel*. Pour visualiser le répertoire personnel, cliquez sur l'icône de framboise pour afficher le menu, sélectionnez Accessoires à l'aide de la souris et cliquez sur Gestionnaire de fichiers (**Figure 3-14**).

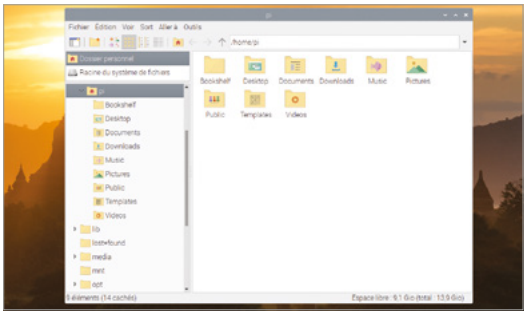


Figure 3-14 : Le gestionnaire de fichiers

Grâce au gestionnaire de fichiers, vous pouvez parcourir les fichiers et les dossiers, qu'on appelle également *répertoires* présents sur la carte microSD de Raspberry Pi, ainsi que sur tous les dispositifs de stockage amovibles (comme les clés USB) que vous connectez aux ports USB de Raspberry Pi. Lorsque vous l'ouvrez pour la première fois, il affiche automatiquement votre répertoire personnel. Vous y trouverez une série d'autres dossiers, appelés *sous-répertoires* classés, comme le menu, par catégories. Les principaux sous-répertoires sont :

- **Bookshelf (Bibliothèque) :** Il contient des copies numériques de livres, de magazines et d'autres publications de Raspberry Pi Press, y compris une copie de ce *Guide du débutant*. Vous pouvez les lire et en télécharger d'autres grâce à l'application Bookshelf, que vous trouverez dans la section Aide du menu.

- **Desktop (Bureau) :** Ce dossier s'affiche lorsque vous chargez Raspberry Pi OS pour la première fois ; si vous y enregistrez un fichier, il s'affichera sur le bureau, ce qui vous facilitera la tâche pour le retrouver et le charger.

- **Documents :** Vous y trouverez la plupart des fichiers que vous créez, des histoires aux recettes de cuisine.

- **Downloads (Téléchargements) :** Lorsque vous téléchargez un fichier depuis Internet à l'aide du navigateur web Chromium, il est automatiquement enregistré dans la rubrique Téléchargements.

- **Music (Musique) :** Toute musique que vous créez ou que vous chargez sur Raspberry Pi peut être stockée ici.

- **Pictures (Images) :** Ce dossier est réservé essentiellement aux images, désignées par le terme technique de *fichiers d'images*.
- **Public :** La plupart de vos fichiers sont privés, mais tout ce que vous rangez dans le dossier Public sera visible à d'autres utilisateurs de Raspberry Pi, même s'ils disposent de leurs propres nom d'utilisateur et mot de passe.
- **Templates (Modèles) :** Ce dossier contient tous les modèles, documents vierges dotés d'une mise en page ou d'une structure de base prédéfinies, que vous avez créés vous-même ou qui sont installés par vos applications.
- **Videos :** Un dossier contenant des vidéos, le premier endroit où la plupart des programmes de lecture vidéo vont chercher.

La fenêtre du gestionnaire de fichiers est elle-même divisée en deux volets : le volet de gauche affiche les répertoires de votre Raspberry Pi, alors que le volet de droite affiche les fichiers et sous-répertoires figurant dans le répertoire sélectionné dans le volet de gauche. Si vous branchez un dispositif de stockage amovible sur le port USB de Raspberry Pi, une fenêtre contextuelle vous demande si vous souhaitez l'ouvrir dans le gestionnaire de fichiers (**Figure 3-15**) ; cliquez sur OK et vous pourrez voir les fichiers et répertoires qu'il contient.

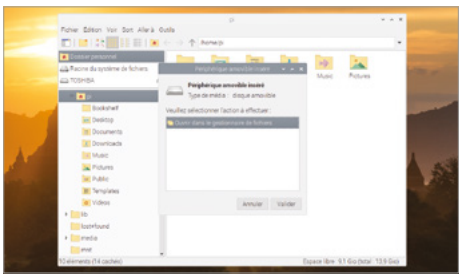
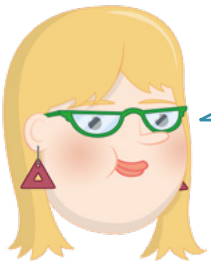


Figure 3-15 : Insertion d'un périphérique de stockage amovible

Il est très simple de copier des fichiers de la carte microSD de Raspberry Pi vers un périphérique amovible : votre répertoire personnel et le périphérique amovible s'affichent dans des fenêtres séparées du gestionnaire de fichiers, placez le curseur de la souris sur le fichier que vous souhaitez copier, cliquez et maintenez enfoncé le bouton gauche de la souris, puis faites-la glisser vers l'autre fenêtre et relâchez le bouton (**Figure 3-16**, au verso). C'est ce que l'on appelle une méthode *glisser-déposer*.

Une autre méthode consiste à cliquer une fois sur le fichier, puis sur le menu Édition, ensuite sur Copier, puis sur l'autre fenêtre, sur le menu Édition et enfin sur Coller.

L'option Couper, également disponible dans le menu Édition, est très similaire mais elle consiste à supprimer le fichier de son emplacement d'origine après avoir effectué la copie. Les deux options peuvent également être utilisées par le biais de raccourcis clavier **CTRL+C** (copier) ou **CTRL+X** (couper), et coller via **CTRL+V**.



LES RACCOURCIS CLAVIER

Lorsqu'un raccourci clavier est précisé, comme **CTRL+C**, il s'agit de maintenir enfoncée la première touche du clavier (**CTRL**), d'appuyer sur la deuxième touche (**C**), puis de relâcher les deux.

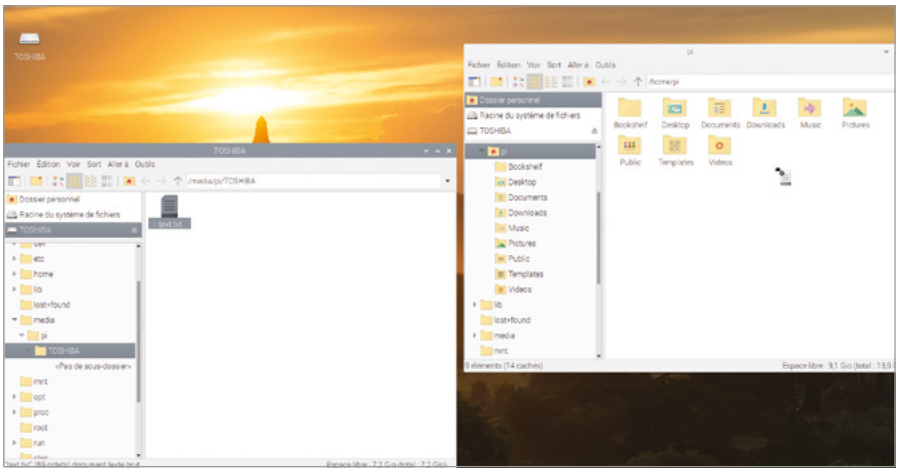


Figure 3-16 : Glisser-déposer un fichier

Lorsque vous avez terminé, fermez le gestionnaire de fichiers en cliquant sur le bouton fermeture en haut à gauche de la fenêtre. Si plusieurs fenêtres sont ouvertes, fermez-les toutes. Si vous avez connecté un périphérique de stockage amovible à votre Raspberry Pi, éjectez-le en cliquant sur le bouton d'éjection en haut à droite de l'écran, recherchez-le dans la liste, puis cliquez dessus avant de le retirer.

ÉJECTION DE PÉRIPHÉRIQUES

Utilisez toujours le bouton d'éjection avant de retirer un périphérique de stockage externe ; dans le cas contraire, les fichiers qui y sont stockés peuvent devenir corrompus et inutilisables.

La suite bureautique LibreOffice

Pour vous familiariser avec d'autres possibilités de Raspberry Pi, cliquez sur l'icône en forme de framboise, placez le curseur de votre souris sur Bureautique, et cliquez sur

LibreOffice Writer. La fonction de traitement de texte de LibreOffice (**Figure 3-17**), une *suite bureautique* très populaire, s'ouvre. Si vous avez utilisé Microsoft Office ou Google Docs, vous avez déjà utilisé une suite bureautique.

Remarque : Il est possible que LibreOffice ne soit pas installé par défaut sur toutes les images de Raspberry Pi OS ; dans ce cas, utilisez l'outil Recommended Software (voir page 43) pour l'installer.

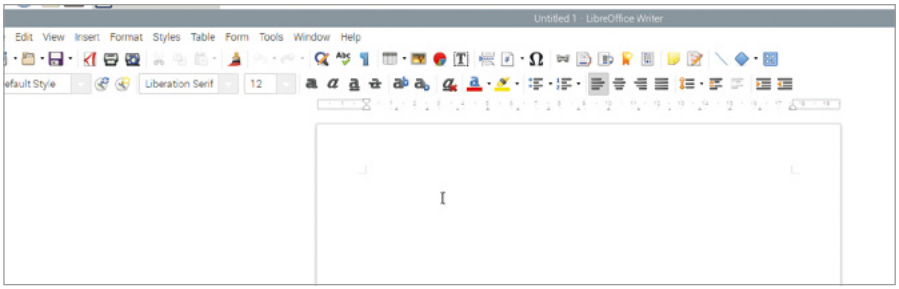


Figure 3-17 : Le programme LibreOffice Writer

Un logiciel de traitement de texte ne se limite pas à la rédaction de documents, mais se charge également de leur mise en page et vous permet de modifier le style, la couleur, la taille de la police, d'ajouter des effets, et même d'insérer des images, des graphiques, des tableaux et d'autres contenus. Le logiciel de traitement de texte vous permet également de vérifier si votre texte comporte des erreurs, en mettant en évidence les fautes d'orthographe et de grammaire, respectivement en rouge et en vert, pendant que vous tapez.

Commencez, par exemple, par rédiger un paragraphe sur ce que vous avez appris sur Raspberry Pi et son logiciel jusqu'à présent. Essayez les différentes icônes sur la partie supérieure de la fenêtre pour comprendre leur rôle : essayez d'agrandir votre texte et d'en modifier la couleur. Si vous n'êtes pas sûr de savoir comment faire, il vous suffit de passer le curseur de la souris sur chacune des icônes pour qu'une « info-bulle » s'affiche et vous indique son rôle. Quand vous avez terminé, cliquez sur le menu File et sélectionnez l'option Save pour sauvegarder votre travail (**Figure 3-18**). Attribuez un nom à votre fichier et cliquez sur le bouton Save.

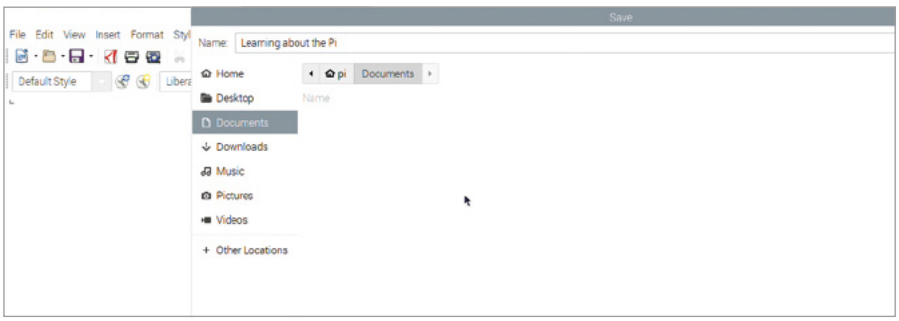


Figure 3-18 : Enregistrer un document

ENREGISTRER VOTRE TRAVAIL

Prenez l'habitude d'enregistrer votre travail, même si vous ne l'avez pas encore terminé. Cela vous évitera bien des ennuis en cas de coupure de courant en plein milieu d'un document !

LibreOffice Writer n'est qu'un élément de la suite bureautique générale LibreOffice. Les autres éléments, que vous trouverez dans la même catégorie de menu Bureautique que LibreOffice Writer, sont les suivants :

- **LibreOffice Base :** Il s'agit d'une base de données, c'est-à-dire un outil permettant de stocker, consulter et analyser des informations.
- **LibreOffice Calc** : Il s'agit d'un tableur, un outil permettant d'effectuer des calculs, de créer des tableaux et des graphiques.
- **LibreOffice Draw** : Il s'agit d'un programme d'illustration, un outil pour créer des images et des diagrammes.
- **LibreOffice Impress** : Il s'agit d'un programme de présentation, pour la création de diapositives et de diaporamas.
- **LibreOffice Math** : Il s'agit d'un éditeur de formules, un outil permettant de créer des formules mathématiques correctement formatées qui peuvent ensuite être utilisées dans d'autres documents.

LibreOffice est également disponible pour d'autres ordinateurs et systèmes d'exploitation. Si vous aimez l'utiliser sur votre Raspberry Pi, vous pouvez le télécharger gratuitement à l'adresse suivante **libreoffice.org** et l'installer sur n'importe quel ordinateur Microsoft Windows, Apple macOS ou Linux.

Si vous voulez en savoir plus sur l'utilisation de LibreOffice, cliquez sur le menu Aide. Dans le cas contraire, fermez LibreOffice Writer en cliquant sur le bouton fermer dans la partie supérieure droite de la fenêtre.

OBTENIR DE L'AIDE

La plupart des programmes comprennent un menu d'aide très complet, allant des informations sur le programme aux guides d'utilisation. Si vous vous sentez perdu ou qu'un programme vous semble incompréhensible, le menu Aide peut vous aider à retrouver votre chemin.

L'outil Recommended Software

Raspberry Pi OS est doté d'une vaste gamme de logiciels, mais il est compatible avec un grand nombre d'autres logiciels. Une sélection choisie de ces logiciels est disponible dans l'outil Recommended Software.

Notez que l'outil Recommended Software nécessite une connexion Internet pour fonctionner. Une fois que votre Raspberry Pi est connecté, cliquez sur l'icône de framboise, placez le curseur de votre souris sur Préférences et cliquez sur Recommended Software. L'outil se chargera, puis commencera à télécharger des informations sur les logiciels disponibles.

Après quelques secondes, une liste de logiciels compatibles s'affichera (**Figure 3-19**). Tout comme les logiciels du menu framboise, ils sont classés en différentes catégories. Cliquez sur une catégorie dans le volet de gauche pour afficher les logiciels de cette catégorie, ou cliquez sur Tous les programmes pour tout afficher.

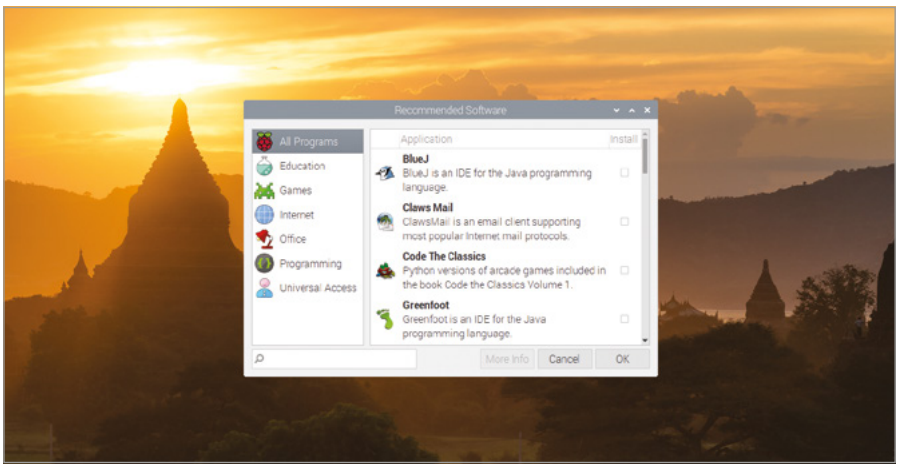
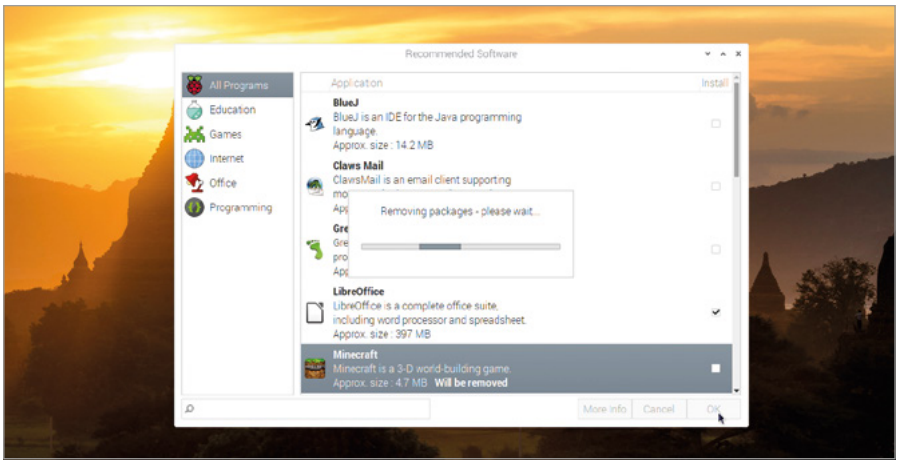


Figure 3-19 : L'outil Recommended Software

Si la case en regard d'un logiciel est cochée, celui-ci est déjà installé sur votre Raspberry Pi. Si ce n'est pas le cas, vous pouvez cocher les cases en regard des logiciels concernés pour les installer. Vous pouvez marquer autant de logiciels que vous le souhaitez avant de les installer tous en même temps, mais si vous utilisez une carte microSD plus petite que celle recommandée, l'espace risque d'être insuffisant pour les charger tous.

Vous pouvez également désinstaller un logiciel de la même manière : décochez la case en regard du logiciel concerné pour le désinstaller. Si vous avez commis une erreur ou si vous changez d'avis, il vous suffit de cliquer à nouveau pour remettre la coche.

Lorsque vous êtes satisfait de votre choix, cliquez sur le bouton OK pour commencer le processus d'installation ou de désinstallation (**Figure 3-20**, au verso). Après le téléchargement et l'installation de tout nouveau logiciel que vous avez choisi, une boîte de dialogue s'affiche ; cliquez sur OK pour fermer l'outil Recommended Software.

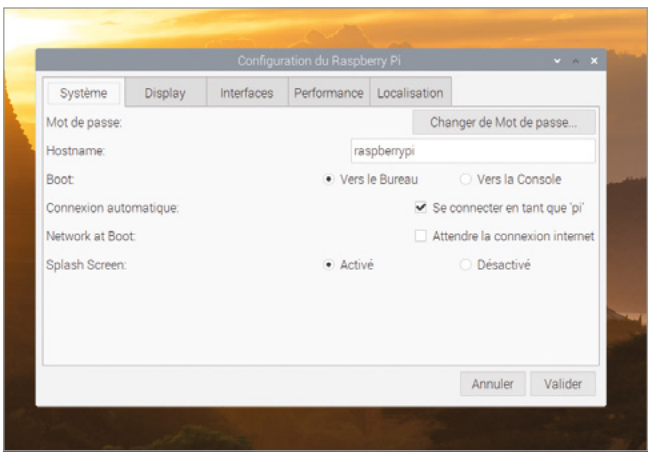


▲ **Figure 3-20 :** Désinstallation de logiciels

Un outil supplémentaire permettant d'installer ou désinstaller un logiciel, l'outil Add/Remove Software, est disponible dans la même catégorie préférences du menu Raspberry Pi OS. Il offre un choix plus large de logiciels, mais qui n'ont pas été validés par la Raspberry Pi Foundation.

Outil de configuration Raspberry Pi

Le dernier programme qui vous sera présenté dans le présent chapitre est connu sous le nom d'outil de Configuration Raspberry Pi. Il ressemble beaucoup au Welcome Wizard que nous avons rencontré au début et vous permet de modifier divers paramètres dans Raspberry Pi OS. Cliquez sur l'icône en forme de framboise, placez le curseur de la souris de sorte à sélectionner la catégorie Préférences, puis cliquez sur Configuration de Raspberry Pi (**Figure 3-21**).



◀ **Figure 3-21 :** Outil de configuration Raspberry Pi

Cet outil comporte cinq onglets. Le premier est l'onglet Système : il vous permet de modifier le mot de passe de votre compte, de définir un nom d'hôte (c'est-à-dire le nom utilisé par Raspberry Pi sur votre réseau sans fil ou câblé local) et de modifier d'autres paramètres. Cependant, il est préférable de ne pas les modifier en règle générale. Cliquez sur l'onglet Display pour afficher la catégorie suivante. Le cas échéant, vous pouvez ici modifier les paramètres d'affichage de l'écran, en fonction de votre écran TV ou moniteur.

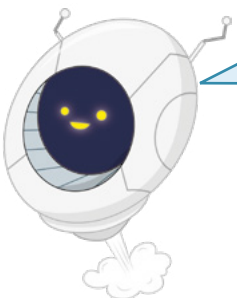
PLUS DE DÉTAILS

Ce bref aperçu vise simplement à vous aider à vous familiariser avec l'outil. Des informations plus détaillées sur chacun de ses paramètres sont disponibles dans l'**Annexe E, Outil de configuration de Raspberry Pi**.

L'onglet Interface propose une série de paramètres, qui sont tous désactivés au départ. Ces paramètres ne doivent être modifiés que si vous ajoutez du nouveau matériel, tel que le module caméra Raspberry Pi, et uniquement à la demande du fabricant du matériel. Il existe certaines exceptions à cette règle : SSH, qui active un « Secure Shell » et vous permet de vous connecter à Raspberry Pi depuis un autre ordinateur de votre réseau à l'aide d'un client SSH ; VNC, qui active un « Virtual Network Computer » et vous permet de visualiser et commander le bureau Raspberry Pi OS depuis un autre ordinateur de votre réseau à l'aide d'un client VNC ; et Remote GPIO, qui vous permet d'utiliser les ports GPIO de Raspberry Pi (nous en parlerons plus en détail au **Chapitre 6, L'informatique physique avec Scratch et Python**) à partir d'un autre ordinateur de votre réseau.

Cliquez sur l'onglet Performance pour afficher la quatrième catégorie. Ici, vous pouvez définir la quantité de mémoire utilisée par l'unité de traitement graphique (GPU) de Raspberry Pi et, pour certains modèles, augmenter les performances de Raspberry Pi grâce à un processus connu sous le nom de *overclocking*. Cependant, tel que nous le disions antérieurement, il est préférable de ne pas modifier ces paramètres, sauf si cela est indispensable.

Finalement, cliquez sur l'onglet Localisation pour afficher la dernière catégorie. Ici, vous pouvez changer votre emplacement, qui définit certains éléments tels que la langue utilisée dans Raspberry Pi OS et le format des chiffres, le fuseau horaire, la disposition du clavier et le pays pour les réseaux WiFi. Pour l'instant, il suffit de cliquer sur Annuler pour fermer l'outil sans ne rien modifier.



ATTENTION !

Les normes applicables aux fréquences WiFi varient en fonction des pays. Si vous réglez le paramètre WiFi dans l'outil de configuration Raspberry Pi sur un pays différent de celui dans lequel vous vous trouvez, vous rencontrerez probablement des difficultés à vous connecter à vos réseaux et vous pourriez enfreindre les lois relatives aux fréquences radio. Évitez donc de procéder de la sorte.

Fermeture

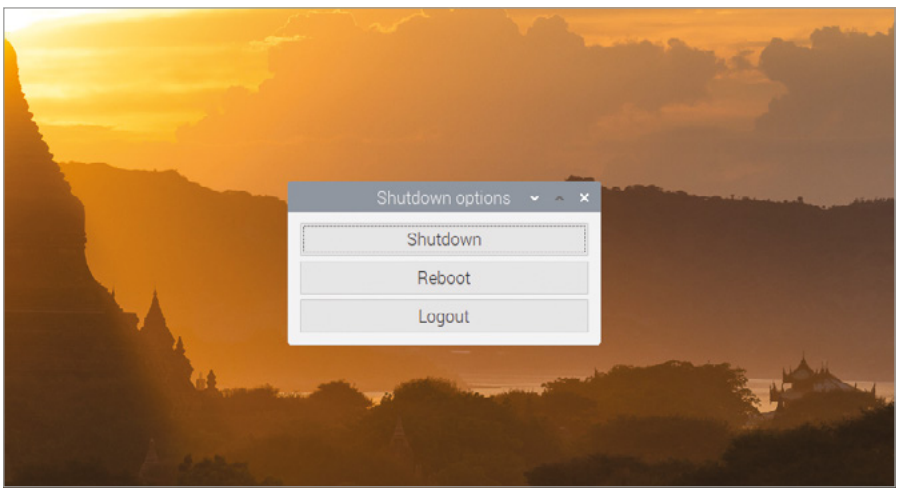
Maintenant que vous avez exploré le bureau de Raspberry Pi OS, il est temps d'acquérir une compétence essentielle : éteindre votre Raspberry Pi en toute sécurité. Comme n'importe quel ordinateur, Raspberry Pi conserve les fichiers sur lesquels vous travaillez dans une *mémoire volatile*, qui se vide à la fermeture du système. Il suffit de sauvegarder au fur et à mesure les documents que vous créez pour les transférer de la mémoire volatile à *la mémoire non-volatile* (la carte microSD) pour s'assurer de ne rien perdre.

Cependant, les documents sur lesquels vous travaillez ne sont pas les seuls dossiers ouverts. Raspberry Pi OS conserve à son tour un certain nombre de fichiers ouverts pendant qu'il est en fonctionnement. Si vous retirez le câble d'alimentation de votre Raspberry Pi pendant que ces fichiers sont encore ouverts, vous risquez de corrompre le système d'exploitation et il sera alors nécessaire de le réinstaller.

Pour éviter que cela ne se produise, vous devez vous assurer de commander à Raspberry Pi OS d'enregistrer tous les fichiers et de se préparer à la fermeture, un processus connu sous le nom d'*arrêt* du système d'exploitation.

Cliquez sur l'icône de la framboise en haut à gauche du bureau, puis cliquez sur Shutdown. Une fenêtre s'affiche en proposant trois options (**Figure 3-22**) : Shutdown, Reboot et Logout. Shutdown est l'option que vous utiliserez le plus : en cliquant sur cette option, Raspberry Pi OS fermera tous les logiciels et fichiers ouverts, puis arrêtera Raspberry Pi. Une fois que l'écran est devenu noir, attendez quelques secondes que le voyant vert clignotant de Raspberry Pi s'éteigne ; vous pourrez alors éteindre l'alimentation électrique en toute sécurité.

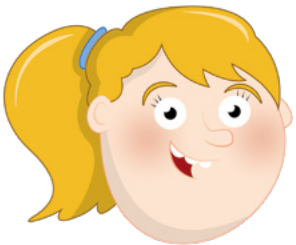
Pour rallumer Raspberry Pi, il suffit de débrancher puis de rebrancher le câble d'alimentation, ou d'activer l'alimentation sur la prise murale.



▲ **Figure 3-22 :** Éteindre votre Raspberry Pi

Le processus de Reboot (redémarrage) est similaire à celui du Shutdown. Il est nécessaire de tout fermer, mais au lieu de couper l'alimentation de Raspberry Pi, il est lancé en suivant le même processus que pour l'arrêt suivi de la déconnexion et reconnexion du câble d'alimentation. La fonction Reboot doit être utilisée si vous apportez certaines modifications nécessitant un redémarrage du système d'exploitation (par exemple des mises à jour du logiciel central) ou en cas d'erreur logicielle (ou *crash*) qui rend Raspberry Pi OS inutilisable.

Enfin, la fonction déconnexion ne vous sera utile que si vous avez plus d'un compte utilisateur sur votre Raspberry Pi : elle ferme tous les programmes que vous avez actuellement ouverts et vous conduit vers un écran de connexion sur lequel vous êtes invité à saisir un nom d'utilisateur et un mot de passe. Si vous vous déconnectez par erreur et que vous souhaitez revenir, il vous suffit de saisir « pi » comme nom d'utilisateur et le mot de passe que vous avez choisi dans le Welcome Wizard au début de ce chapitre.



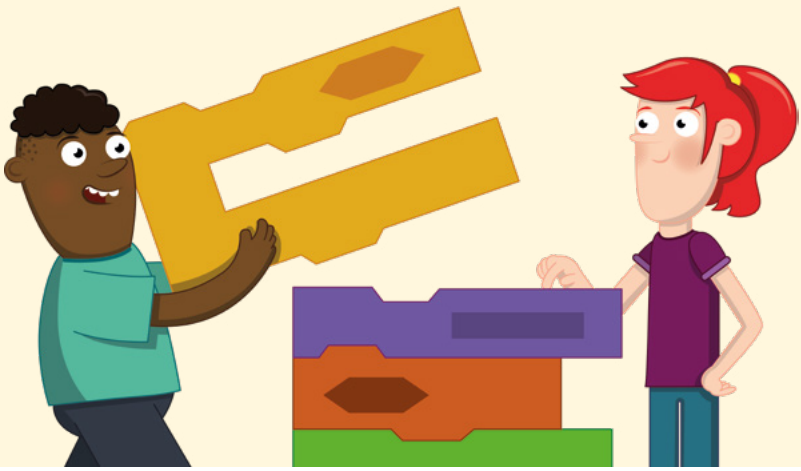
ATTENTION !

Ne retirez jamais le câble d'alimentation de votre Raspberry Pi avant de l'avoir éteint. Ce faisant, vous risquez de corrompre le système d'exploitation et de perdre les fichiers que vous avez créés ou téléchargés.

Chapitre 4

Programmation avec Scratch 3

Apprenez à coder en utilisant Scratch, le langage de programmation par blocs

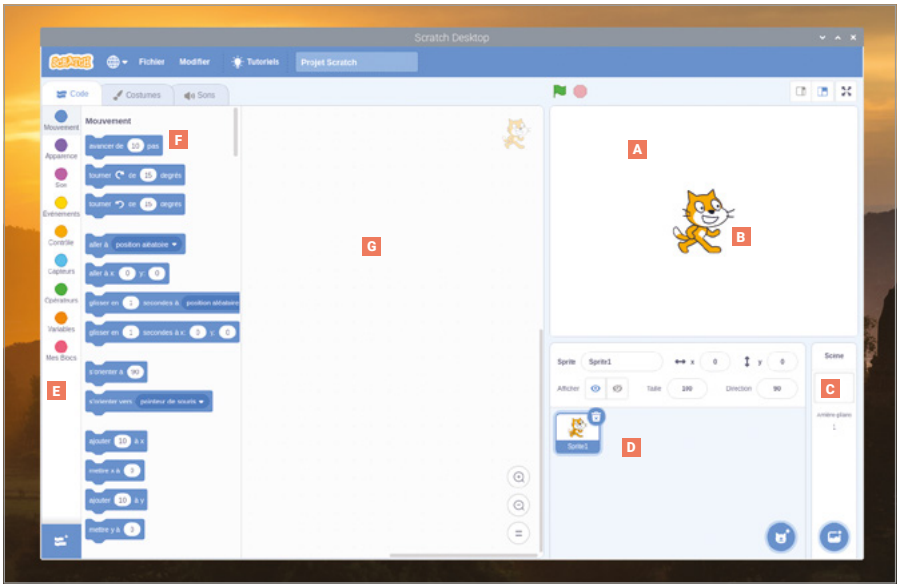


Utiliser Raspberry Pi, ce n'est pas simplement se limiter à utiliser des logiciels créés par d'autres personnes, mais également en créer de nouveaux en laissant libre cours à l'imagination. Que vous ayez ou non de l'expérience dans la création de programmes (un processus connu sous le nom de programmation ou codage), vous trouverez en Raspberry Pi une excellente plateforme de création et d'expérimentation.

Pour apprendre les bases du codage sur Raspberry Pi, Scratch, un langage de programmation visuel développé par le Massachusetts Institute of Technology (MIT), peut se révéler un allié précieux. Dans les langages de programmation traditionnels, vous écrivez des instructions textuelles que l'ordinateur doit exécuter, un peu comme la recette d'un gâteau. Scratch, en revanche, vous permet de construire votre programme étape par étape en utilisant des blocs, des morceaux de code pré-écrits se présentant sous forme de pièces d'un puzzle suivant un code couleur très simple.

Scratch est un excellent moyen de s'initier au langage de programmation pour les codeurs en herbe de tout âge, mais ne vous laissez pas tromper par son aspect convivial : il s'agit bel et bien d'un environnement de programmation puissant et entièrement fonctionnel capable de tout, des jeux et animations simples aux projets de robotique interactifs complexes.

Présentation de l'interface Scratch 3



A **Scène** : exactement comme les acteurs d'une pièce de théâtre, vos sprites se déplacent sur la scène selon les indications de votre programme.

B **Sprites** : les personnages ou objets que vous insérez dans un programme Scratch sont appelés des sprites et ils sont placés sur la scène.

C **Contrôles de scène** : vous pouvez modifier votre scène et y ajouter par exemple vos propres images en arrière-plan, en utilisant les contrôles de scène.

D **Liste des sprites** : tous les sprites que vous avez créés ou chargés dans Scratch apparaîtront dans cette partie de la fenêtre.

E **Palette de blocs** : tous les blocs disponibles pour votre programme apparaissent dans la palette de blocs, qui comporte différentes catégories suivant un code couleur.

VERSIONS DE SCRATCH

Au moment de la rédaction de ce guide, Raspberry Pi OS était doté de trois versions de Scratch : Scratch 1, 2 et 3, tous inclus dans la section Programmation du menu Raspberry Pi OS. Ce chapitre concerne la version Scratch 3. Notez que Scratch 3 ne peut fonctionner que sur Raspberry Pi 4. Si vous préférez utiliser Scratch 2, cette version ne fonctionnera pas sur Raspberry Pi Zero, Modèle A, A+, B ou B+.

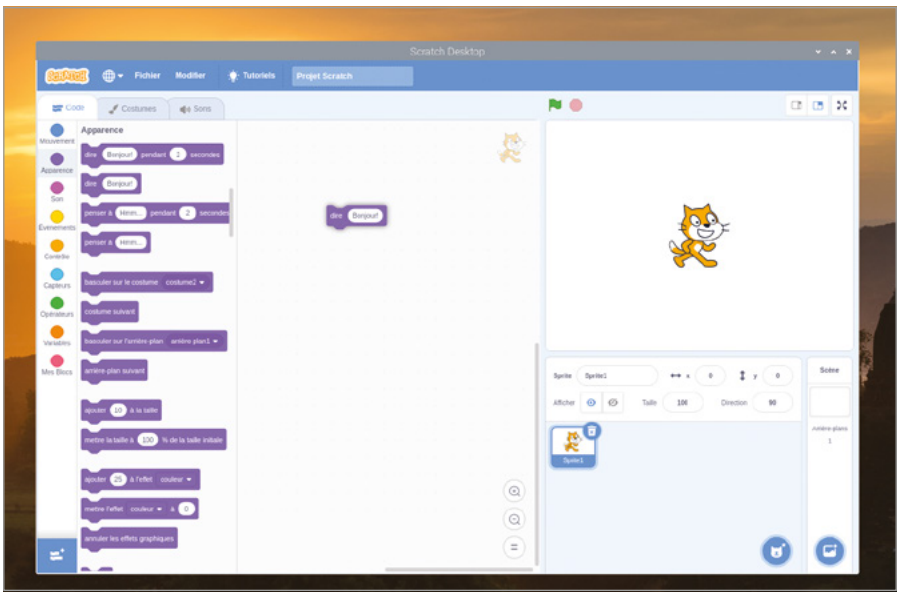
F **Blocs** : les blocs, des morceaux de code de programmation pré-écrits, vous permettent de développer votre programme étape par étape.

G **Zone de codage** : la zone de codage est l'endroit où votre programme est construit selon la méthode glisser-déposer de blocs depuis la palette de blocs pour former des scripts.

Vote tout premier programme Scratch : Bonjour, tout le monde !

Scratch 3 se charge exactement comme n'importe quel autre programme sur Raspberry Pi : cliquez sur le symbole représentant une framboise pour ouvrir le menu de Raspberry Pi OS, déplacez le curseur sur la section Programmation et cliquez sur Scratch 3. Après quelques secondes, l'interface utilisateur de Scratch 3 sera chargée.

Généralement, avec la plupart des langages de programmation, il faut donner des instructions écrites à l'ordinateur. Ce n'est pas le cas avec Scratch. Commencez par cliquer sur la catégorie Apparence dans la palette de blocs, située à gauche de la fenêtre Scratch. Cela fera apparaître les blocs de cette catégorie, en violet. Cherchez le bloc dire Bonjour! , cliquez dessus en maintenant appuyé le bouton gauche de la souris, puis faites-le glisser jusqu'à la zone de codage au centre de la fenêtre de Scratch avant de relâcher le bouton de la souris (**Figure 4-1**).



▲ **Figure 4-1 :** Glissez et déposez le bloc dans la zone de codage

Regardez bien la forme du bloc que vous venez de faire glisser : il a une sorte d'encoche sur le haut, et cette même forme sort du bas. Comme une pièce de puzzle, ce bloc indique qu'il faudra placer quelque chose au-dessus et quelque chose au-dessous. Le programme appelle ce « quelque chose » un *déclencheur*.

Cliquez sur la catégorie Événements dans la palette des blocs, en doré, puis cliquez sur le bloc quand est cliqué, qu'on appelle un *bloc de tête*, et faites-le glisser dans la zone de codage. Placez-le de sorte à ce que la forme qui pointe vers le bas s'emboîte dans l'encoche en haut du bloc dire Bonjour! jusqu'à ce qu'un contour blanc apparaisse, puis relâchez le

bouton de la souris. Vous ne devez pas nécessairement faire preuve de précision, le bloc s'emboîte comme une pièce de puzzle dès qu'il est suffisamment proche. Si ce n'est pas le cas, cliquez en maintenant enfoncé le bouton de la souris et ajustez sa position jusqu'à ce qu'il s'emboîte.

Votre programme est maintenant terminé. Pour le faire fonctionner ou pour *exécuter* le programme, cliquez sur le drapeau vert en haut à gauche de la scène. Si tout fonctionne normalement, le chat sur la scène vous accueillera d'un joyeux « Bonjour ! » (**Figure 4-2**). Votre premier programme est un succès !

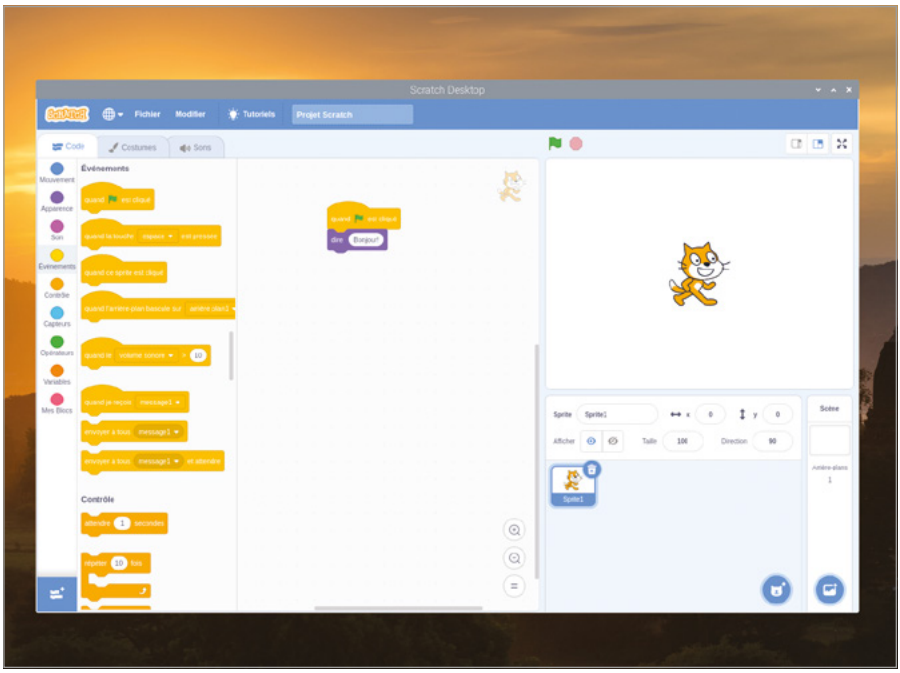
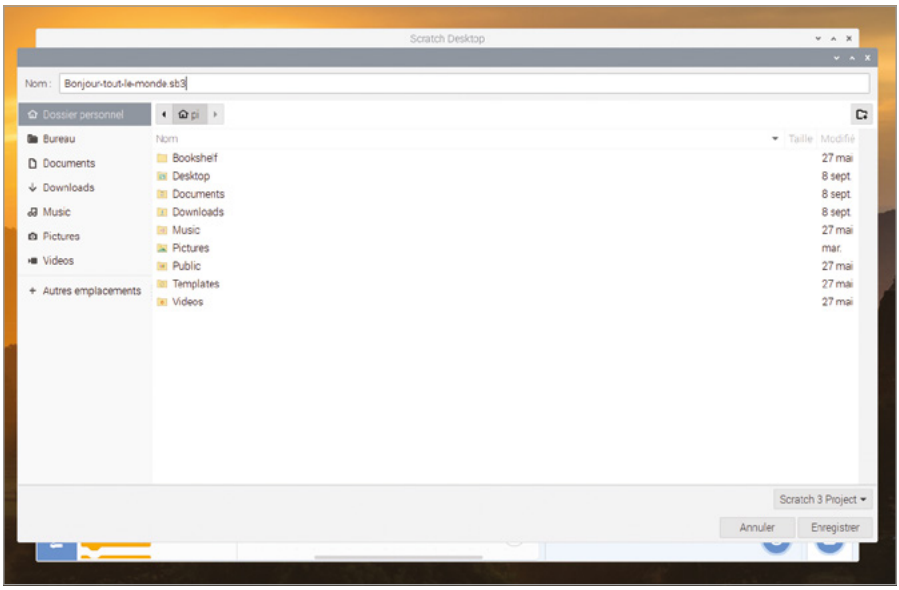
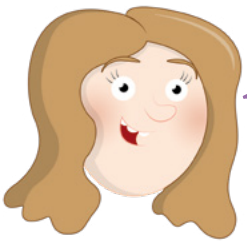


Figure 4-2 : Cliquez sur le drapeau vert au-dessus de la scène et le chat dira Bonjour

Avant de continuer, nommez et enregistrez votre programme. Cliquez sur le menu Fichier, puis sur « Sauvegarder sur votre ordinateur ». Saisissez un nom et cliquez sur le bouton Enregistrer (**Figure 4-3**).



▲ **Figure 4-3 :** Enregistrez votre programme avec un nom mémorable



QUE PEUT-IL DIRE ?

Certains blocs de Scratch peuvent être modifiés. Essayez de cliquer sur le mot « Bonjour! » et de saisir autre chose, puis cliquez à nouveau sur le drapeau vert. Que se passe-t-il sur la scène ?

Prochaines étapes : les séquences

Si votre programme ne comporte que deux blocs, il ne contient qu'une seule instruction : celle de dire « Bonjour! » à chaque fois que vous cliquez sur le drapeau et que le programme s'exécute. Pour en faire un peu plus, vous devez connaître les *séquences*. Dans leur forme la plus simple, les programmes informatiques consistent en une liste d'instructions, tout comme une recette de cuisine. Chaque instruction fait suite à celle qui lui précède dans une progression logique connue sous le nom de *séquence linéaire*.

Commencez par cliquer sur le bloc dire Bonjour! et faites-le glisser de la zone de codage vers la palette de blocs (**Figure 4-4**). Cela permet de supprimer le bloc, de le retirer de votre programme et de ne laisser que le bloc déclencheur quand est cliqué.

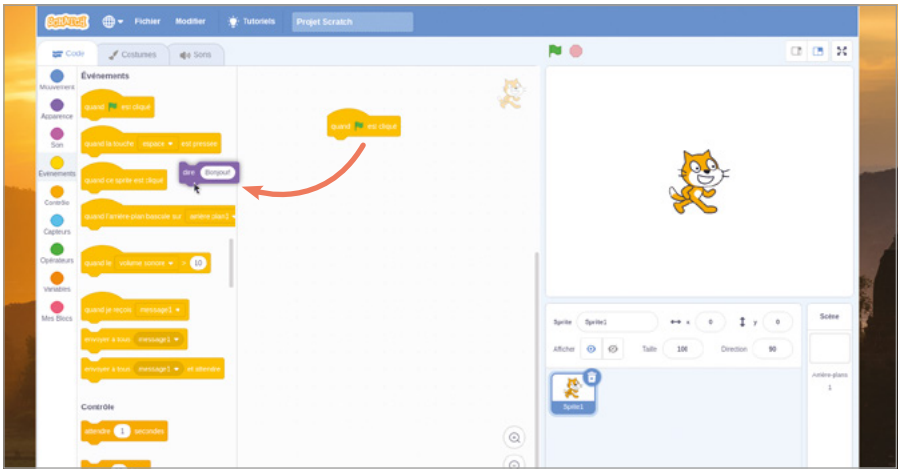
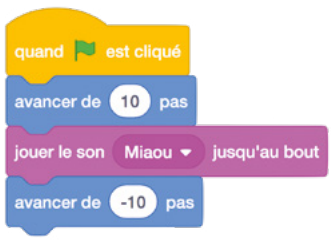


Figure 4-4 : Pour supprimer un bloc, il suffit de le faire glisser hors de la zone de codage

Cliquez sur la catégorie Mouvement dans la palette des blocs, puis cliquez sur le bloc avancer de 10 pas et faites-le glisser de sorte à ce qu'il s'emboîte au-dessous du bloc déclencheur dans la zone de codage. Cela indique à votre sprite (le chat) qu'il doit avancer d'un certain nombre de pas dans la direction où il se trouve actuellement.

Ajoutez des instructions supplémentaires à votre programme pour créer une séquence. Cliquez sur la catégorie Son, en rose, puis cliquez sur le bloc jouer le son Miaou jusqu'au bout, faites-le glisser et emboîtez-le au-dessous du bloc avancer de 10 pas. Continuez de la sorte : cliquez à nouveau sur la catégorie Mouvement et faites glisser un autre bloc avancer de 10 pas au-dessous de votre bloc Son, mais cette fois-ci, cliquez sur le « 10 » pour le sélectionner et saisissez « -10 » pour créer un bloc avancer de -10 pas.

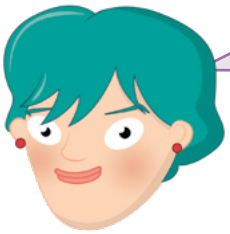


Cliquez sur le drapeau vert au-dessus de la scène pour exécuter le programme. Vous verrez le chat se déplacer vers la droite en miaulant (veillez à disposer de haut-parleurs ou d'écouteurs branchés pour l'entendre) puis revenir au point de départ. Cliquez à nouveau sur le drapeau, et le chat recommencera depuis le début.

Félicitations : vous avez créé une séquence d'instructions et Scratch les exécutera une après l'autre, de haut en bas. Scratch exécute une seule instruction de la séquence à la fois, mais très rapidement : essayez de supprimer le bloc **jouer le son Miaou jusqu'au bout** en cliquant sur le bloc **avancer de -10 pas** en dessous et en le faisant glisser pour les séparer, puis en ramenant le bloc **jouer le son Miaou jusqu'au bout** vers la palette de blocs, et remplacez-le par le bloc plus simple **jouer le son Miaou** avant de ramener votre bloc **avancer de -10 pas** tout en bas de votre programme.

Cliquez sur le drapeau vert pour relancer votre programme, vous aurez l'impression que le chat ne bouge pas. En réalité il se déplace, mais il recule si vite qu'il semble immobile. En effet, le bloc **jouer le son Miaou** n'attend pas la fin du miaulement pour passer à l'étape suivante, car Raspberry Pi « réfléchit » à une rapidité telle que l'instruction suivante s'exécute avant de percevoir le mouvement du chat. Pour régler ce problème, vous pouvez soit utiliser le bloc **jouer le son Miaou jusqu'au bout**, soit cliquer sur la catégorie Contrôle, en orange clair dans la palette de blocs, puis cliquer sur le bloc **attendre 1 secondes** et le glisser entre les blocs **jouer le son Miaou** et **avancer -10 pas** situé en bas.

Si vous cliquez à nouveau sur le drapeau vert pour lancer le programme, vous verrez que le chat attend une seconde, après s'être déplacé vers la droite, avant de revenir vers la gauche. C'est ce qu'on appelle un *délai*, qui est essentiel pour contrôler la durée de votre séquence d'instructions.



DÉFI : AJOUTER DES ÉTAPES

Essayez d'ajouter des étapes à votre séquence et de modifier les valeurs des étapes existantes. Que se passe-t-il lorsque le nombre de pas d'un bloc de mouvement ne correspond pas au nombre de pas d'un autre bloc ? Que se passe-t-il si vous essayez de jouer un son alors qu'un autre son est encore en train de jouer ?

Boucler la boucle

La séquence que vous avez créée jusqu'à présent ne s'exécute qu'une seule fois : vous cliquez sur le drapeau vert, le chat bouge et miaule, puis le programme s'arrête jusqu'à ce que vous cliquiez à nouveau sur le drapeau vert. Pour remédier à cette situation, Scratch a prévu un bloc de contrôle connu sous le nom de *boucle*.

Cliquez sur la catégorie Contrôle dans la palette des blocs et recherchez le bloc répéter indéfiniment. Cliquez dessus et faites-le glisser dans la zone de codage, puis déposez-le au-dessous du bloc quand est cliqué et au-dessus du premier bloc avancer de 10 pas.

Vous remarquerez que ce bloc, en forme de C, s'adapte automatiquement pour entourer les autres blocs de votre séquence. Si vous cliquez maintenant sur le drapeau vert, vous verrez immédiatement les effets du bloc répéter indéfiniment : au lieu de s'exécuter une seule fois, votre programme s'exécute encore et encore : pour l'éternité, donc. En programmation, cela s'appelle une *boucle infinie*, c'est-à-dire une boucle qui ne se termine jamais.

Dès que ces miaulements incessants deviennent lassants, cliquez sur l'octogone rouge à côté du drapeau vert au-dessus de la scène pour interrompre le programme. Pour modifier le type de boucle, cliquez et faites glisser le premier bloc avancer de 10 pas et retirez-le, ainsi que les blocs qui se trouvent en dessous, du bloc répéter indéfiniment, puis placez-les en

dessous du bloc quand est cliqué. Cliquez et faites glisser le bloc répéter indéfiniment dans la palette des blocs pour le retirer, puis cliquez et faites glisser le bloc répéter 10 fois sous le bloc quand est cliqué pour qu'il entoure les autres blocs.

Cliquez sur le drapeau vert pour exécuter votre nouveau programme. Au début, il semblerait que la première version se répète : votre séquence d'instructions s'exécute encore et encore. Mais cette fois, la boucle s'interrompt après dix répétitions et ne continue pas indéfiniment. C'est ce qu'on appelle une *boucle définie* : c'est vous qui décidez à quel moment elle se termine. Les boucles sont des outils puissants, et la plupart des programmes (en particulier les jeux et les programmes de détection) en font un usage intensif (aussi bien des boucles infinies que définies).

QUE SE PASSE-T-IL MAINTENANT ?

Que se passe-t-il si vous mettez un nombre plus élevé dans le bloc boucle ? Que se passe-t-il si ce nombre est plus petit ? Que se passe-t-il si vous mettez le numéro 0 dans le bloc boucle ?

Variables et conditions

Les derniers concepts à comprendre avant de se lancer sérieusement dans le codage de programmes Scratch sont étroitement liés l'un à l'autre : les *variables*et les *conditions*. Une variable est, comme son nom l'indique, une valeur susceptible de varier(en d'autres termes, de changer) dans le temps et sous le contrôle du programme. Une variable possède deux propriétés principales : son nom et la valeur qu'elle représente. Cette valeur ne doit pas nécessairement être un chiffre : il peut s'agir de numéros, de texte, de condition vrai/faux, ou encore d'une valeur vide, appelée *valeur nulle*.

Les variables sont des outils puissants. Pensez à tout ce que vous devez contrôler dans un jeu par exemple : la santé d'un personnage, la vitesse d'un objet en mouvement, le niveau en cours et le score. Tous ces éléments sont des variables.

Tout d'abord, cliquez sur le menu Fichier et enregistrez votre programme existant en cliquant sur « Sauvegarder sur votre ordinateur ». Si vous aviez enregistré le programme auparavant, il vous sera demandé si vous souhaitez remplacer l'ancienne copie sauvegardée par votre nouvelle version actualisée. Ensuite, cliquez sur Fichier, puis sur Nouveau pour lancer un nouveau projet vierge (cliquez sur OK lorsqu'on vous demande si vous souhaitez remplacer le contenu du projet en cours). Cliquez sur la catégorie Variables en orange foncé dans la palette des blocs, puis sur le bouton « Créer une variable ». Tapez « boucles » comme nom de variable (**Figure 4-5**), puis cliquez sur OK pour faire apparaître une série de blocs dans la palette de blocs.

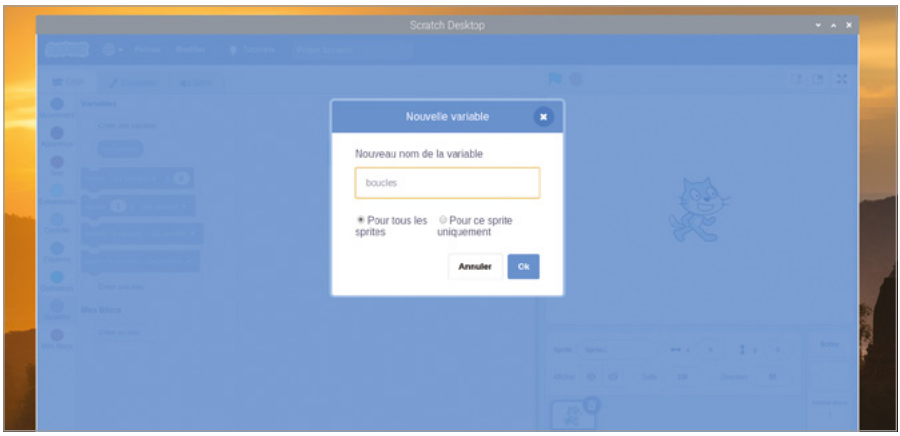
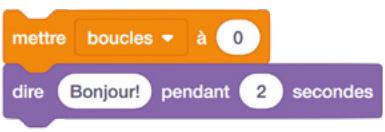
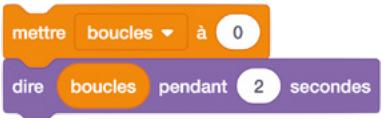


Figure 4-5 : Attribuez un nom à votre nouvelle variable

Cliquez sur le bloc mettre boucles à 0 et faites-le glisser vers la zone de codage. Cela indique à votre programme qu'il doit *initialiser* la variable ayant une valeur de 0. Ensuite, cliquez sur la catégorie Apparence de la palette de blocs et faites glisser le bloc dire Bonjour! pendant 2 secondes sous votre bloc mettre boucles à 0.



Comme vous l'avez constaté antérieurement, les blocs dire Bonjour! font dire au chat tout ce que vous y écrivez. Cependant, plutôt que d'écrire vous-même le message dans le bloc, vous pouvez utiliser une variable à la place. Cliquez à nouveau sur la catégorie Variables dans la palette de blocs, puis cliquez et faites glisser le bloc arrondi boucles (connu sous le nom de *bloc rapporteur*) qui se trouve en haut de la liste et qui est assorti d'une case à cocher, en le superposant au mot « Bonjour ! » dans le bloc dire Bonjour! pendant 2 secondes. Ce faisant, vous créez un nouveau bloc combiné : dire boucles pendant 2 secondes.



Cliquez sur la catégorie Événements dans la palette des blocs, puis cliquez et faites glisser le bloc quand est cliqué et placez-le en tête de votre séquence de blocs. Cliquez sur le drapeau vert au-dessus de la scène et vous verrez le chat dire « 0 » (**Figure 4-6**), c'est-à-dire la valeur que vous avez attribuée à la variable « boucles ».

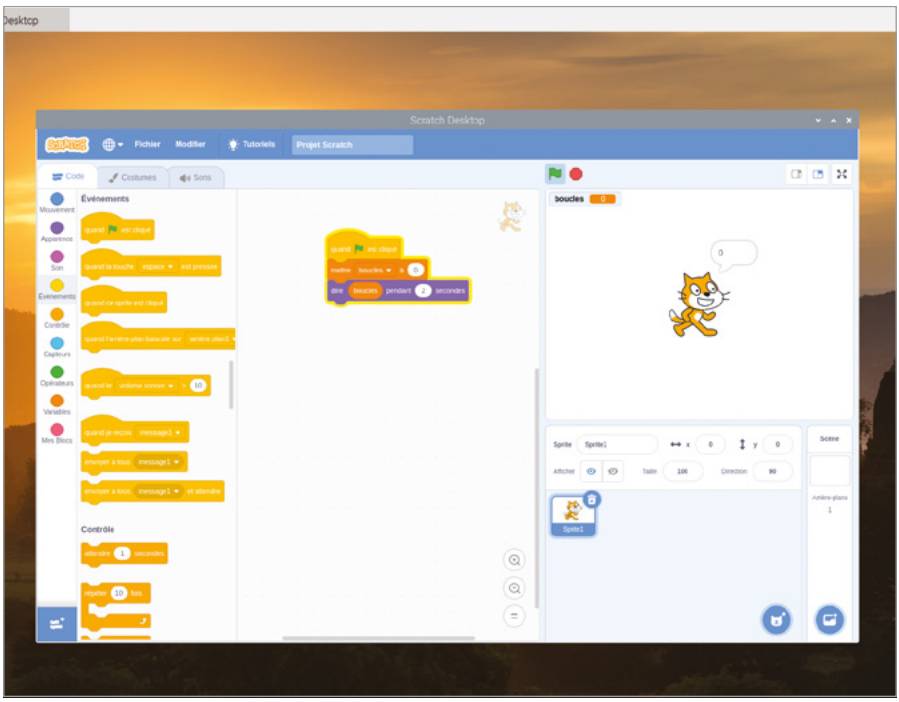
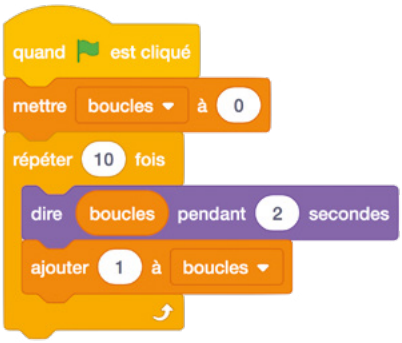


Figure 4-6 : Cette fois, le chat prononcera la valeur de la variable

Mais les variables ne sont pas immuables. Cliquez sur la catégorie Variables dans la palette des blocs, puis cliquez et faites glisser le bloc ajouter 1 à boucles et placez-le à la fin de votre séquence de blocs. Ensuite, cliquez sur la catégorie Contrôle, puis cliquez et faites glisser un bloc de répéter 10 fois et placez-le de manière à ce qu'il se situe immédiatement sous votre bloc mettre boucles à 0 et qu'il entoure les autres blocs de votre séquence.



Cliquez à nouveau sur le drapeau vert. Cette fois, le chat comptera de 0 à 9. Cette séquence fonctionne parce que votre programme change, ou *modifie* la variable : chaque fois que la boucle s'exécute, le programme ajoute un à la valeur de la variable « boucles » (**Figure 4-7**).

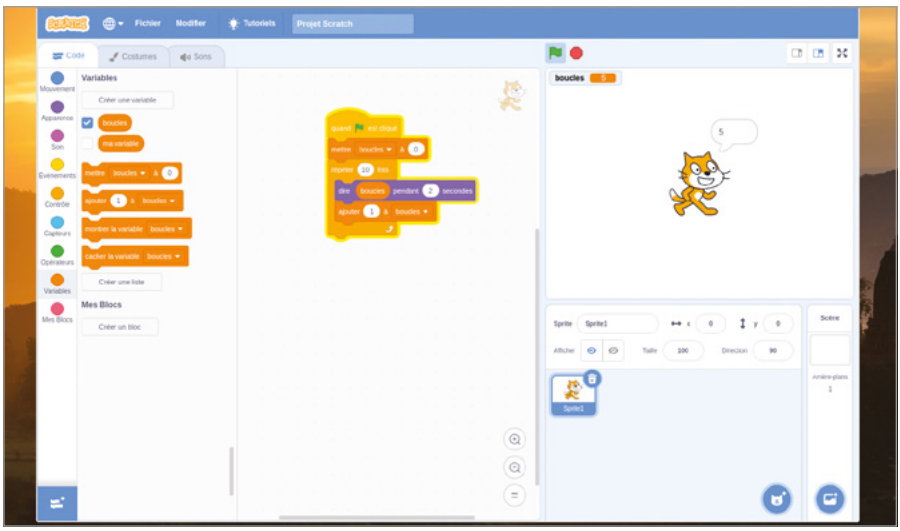
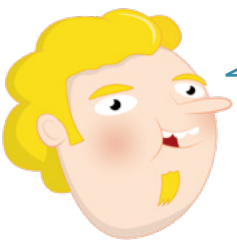


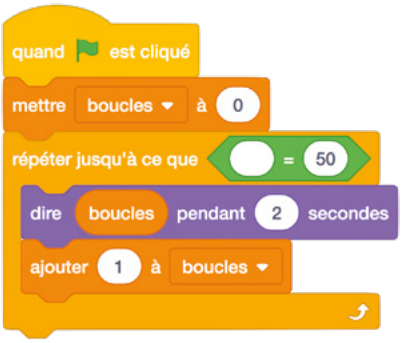
Figure 4-7 : Grâce à la boucle, le chat peut compter de manière croissante



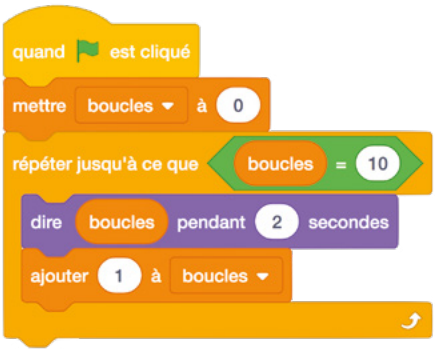
COMPTER À PARTIR DE ZÉRO

La boucle que vous avez créée s'exécute dix fois, mais le sprite de chat ne compte que jusqu'à neuf. Cela est dû au fait que la valeur de départ de notre variable est zéro. En comptant le zéro et le neuf, il y a dix numéros entre les deux, de sorte que le programme s'arrête avant que le chat ne prononce le mot « 10 ». Pour changer cela, vous pouvez définir la valeur initiale de la variable sur 1 au lieu de 0.

Vous pouvez faire bien plus avec une variable, outre la modifier. Cliquez et faites glisser le bloc dire boucles pendant 2 secondes pour le faire sortir du bloc répéter 10 fois et placez-le en dessous du bloc répéter 10 fois. Cliquez et faites glisser le bloc répéter 10 fois vers la palette de blocs pour le retirer, puis remplacez-le par un bloc répéter jusqu'à ce que, en s'assurant que le bloc est connecté au bas du bloc mettre boucles à 0 et qu'il entoure les deux autres blocs de votre séquence. Cliquez sur la catégorie Opérateurs dans la palette des blocs, en vert, puis cliquez et faites glisser le losange (=) et placez-le dans l'espace en forme de losange qui se trouve au cœur du bloc répéter jusqu'à ce que.



Ce bloc Opérateurs vous permet de comparer deux valeurs, y compris entre variables. Cliquez sur la catégorie Variables, faites glisser le bloc rapporteur boucles dans l'espace vide du bloc Opérateurs (=), puis cliquez sur l'espace contenant « 50 » et saisissez le chiffre « 10 ».



Cliquez sur le drapeau vert au-dessus de la scène, et vous constaterez que le programme fonctionne exactement comme avant : le sprite du chat compte de 0 à 9 (**Figure 4-8**), puis le programme s'arrête. En effet, le bloc répéter jusqu'à ce que fonctionne exactement de

la même manière que le bloc répéter 10 fois, mais plutôt que de compter le nombre de boucles, il compare la valeur de la variable « boucles » à la valeur que vous avez saisie du côté droit du bloc. Lorsque la valeur de la variable « boucles » atteint 10, le programme s'arrête.

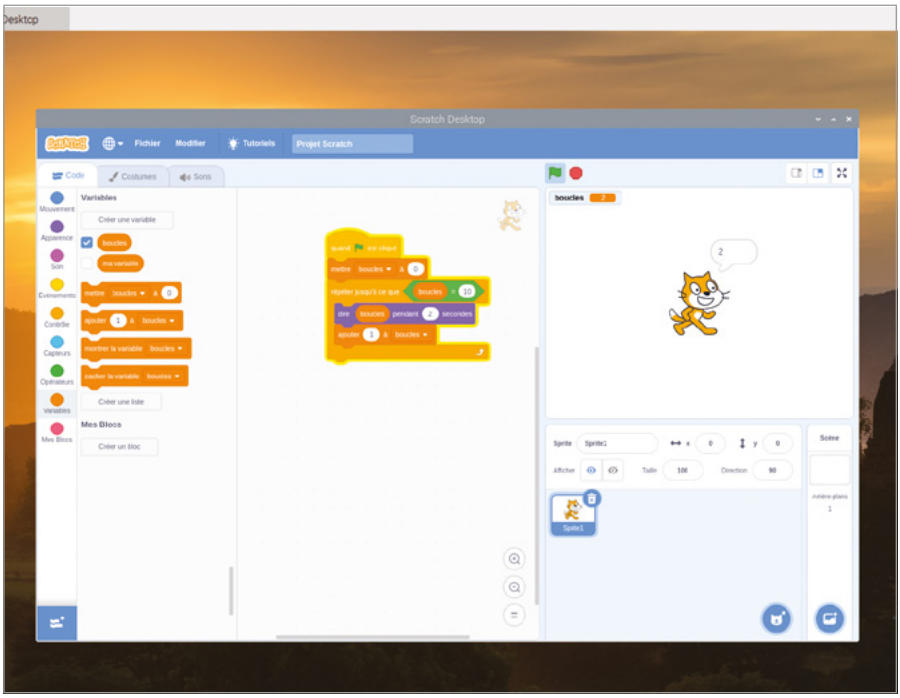
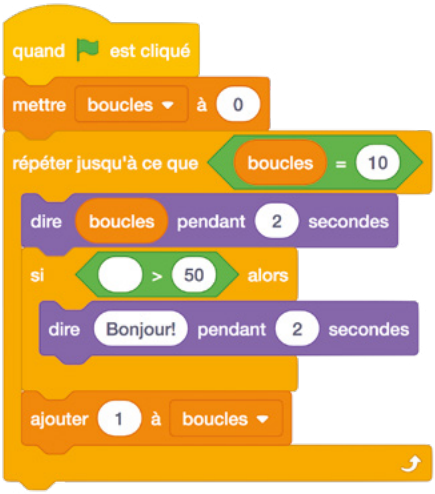


Figure 4-8 : Utilisation d'un bloc « répéter jusqu'à ce que » avec un opérateur comparatif

Comme son nom l'indique, un *opérateur comparatif* compare effectivement deux valeurs. Cliquez sur la catégorie Opérateurs de la palette de blocs, et recherchez les deux autres blocs en forme de losange au-dessus et au-dessous de celui qui contient le symbole « = ». Il s'agit également d'opérateurs comparatifs : « < » compare deux valeurs et il est valide lorsque la valeur de gauche est inférieure à la droite, et « > » est valide lorsque la valeur de gauche est supérieure à celle de droite.

Cliquez sur la catégorie Contrôle de la palette de blocs, cherchez le bloc si alors, puis cliquez et faites-le glisser vers la zone de codage avant de le déposer directement sous le bloc dire boucles pendant 2 secondes. Il entourera automatiquement le bloc ajouter 1 à boucles, puis cliquez et faites glisser ce bloc pour le déplacer jusqu'à ce qu'il s'emboîte au bas du bloc si alors. Cliquez sur la catégorie Apparence de la palette de blocs et faites glisser le bloc dire Bonjour! pendant 2 secondes à l'intérieur de votre bloc si alors. Cliquez sur la catégorie Opérateurs de la palette de blocs et faites glisser le bloc ( ) > ( ) à l'intérieur de l'espace en forme de losange du bloc si alors.



Le bloc **si alors** est un bloc conditionnel, ce qui signifie que les blocs qu'il contient ne seront exécutés que si une certaine condition est remplie. Cliquez sur la catégorie Variables dans la palette de blocs, faites glisser le bloc rapporteur **boucles** dans l'espace vide de votre bloc **> ?** , puis cliquez sur l'espace contenant « 50 » et saisissez le chiffre « 5 ». Enfin, cliquez sur le mot Bonjour! dans votre bloc **dire Bonjour! pendant 2 secondes** et saisissez « C'est trop ! ».

Cliquez sur le drapeau vert. Le programme fonctionnera d'abord exactement comme avant, avec le chat qui compte à partir de zéro en avant. Quand il arrive à 6, le premier nombre au-dessus de 5, le bloc si alors commence à s'activer et le chat fait remarquer que le chiffre devient trop important (**Figure 4-9**). Félicitations : vous savez utiliser des variables et des conditions !

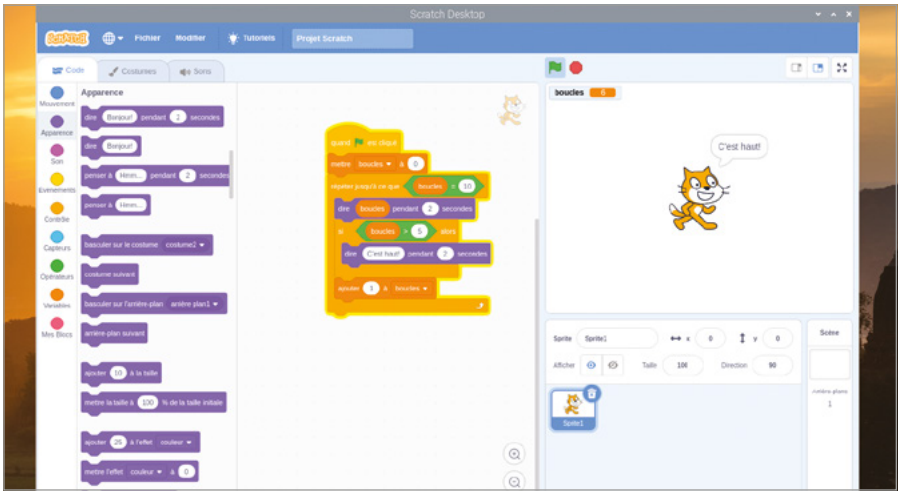
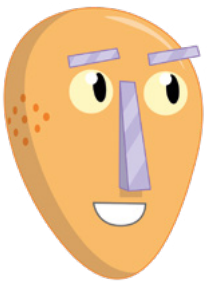


Figure 4-9 : Le chat fait remarquer lorsqu'il arrive à 6



DÉFI : TROP OU PAS ASSEZ

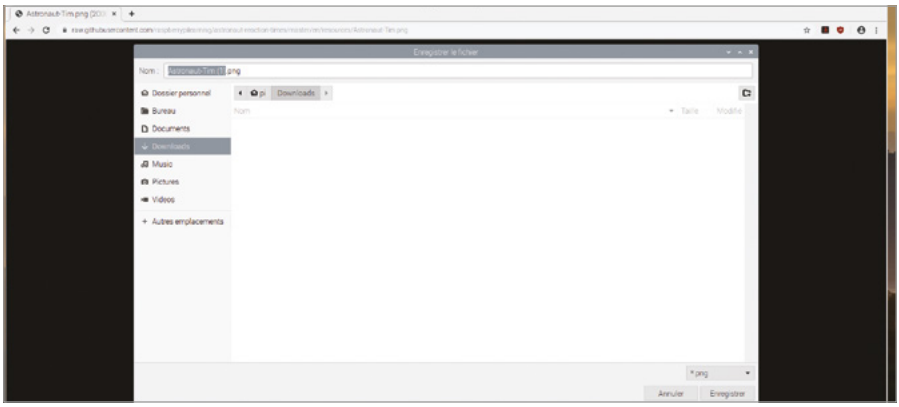
Comment pourriez-vous modifier le programme pour que le chat commente les numéros en dessous de 5 ? Pouvez-vous le modifier de manière à ce que le chat signale les nombres trop bas et les nombres trop élevés ? Essayez le bloc si alors sinon pour faciliter les choses !

Projet 1 : Chronomètre de la réaction de l'astronaute

Maintenant, vous savez tout du fonctionnement de Scratch. L'heure est venue d'essayer quelque chose de plus interactif : un chronomètre du temps de réaction, conçu en honneur de Tim Peake, astronaute britannique de l'Agence spatiale européenne et de ses missions à bord de la Station spatiale internationale.

Si vous souhaitez le conserver, sauvegardez votre programme actuel puis ouvrez un nouveau projet en cliquant sur Fichier et Nouveau. Avant de commencer, attribuez-lui un nom en cliquant sur « Fichier » et « Sauvegarder sur votre ordinateur » : appelez-le « Chronomètre de la réaction de l'astronaute ».

Ce projet s'appuie sur deux images, un arrière-plan et un sprite, qui ne sont pas incluses dans les ressources intégrées de Scratch. Pour les télécharger, cliquez sur l'icône représentant une framboise pour ouvrir le menu de Raspberry Pi OS, déplacez le curseur sur Internet et cliquez sur navigateur Web Chromium. Lorsque le navigateur est chargé, saisissez **rpf.io/astronaut-backdrop** dans la barre d'adresse, puis appuyez sur la touche **ENTRÉE**. Cliquez avec le bouton droit de la souris sur l'image de l'espace et cliquez sur Enregistrer l'image sous..., puis cliquez sur le bouton Enregistrer (**Figure 4-10**). Cliquez à nouveau dans la barre d'adresse, et saisissez **rpf.io/astronaut-sprite** puis appuyez sur la touche **ENTRÉE**.



▲ **Figure 4-10 :** Enregistrer l'image en arrière-plan

Cliquez avec le bouton droit de la souris sur le portrait de Tim Peake et cliquez sur Enregistrer l'image sous..., puis sélectionnez le dossier Téléchargements et cliquez sur le bouton Enregistrer. Une fois ces deux images sauvegardées, vous pouvez fermer Chromium ou le laisser ouvert et utiliser la barre des tâches pour revenir à Scratch 3.

INTERFACE UTILISATEUR

Si vous avez lu ce chapitre depuis le début, vous connaissez désormais l'interface utilisateur de Scratch 3. Les instructions ci-dessous relatives au projet partent du principe que vous savez où trouver les différents éléments ; si vous avez oublié où se trouvent certains objets, revenez vers l'image de l'interface utilisateur au début du présent chapitre.

Cliquez avec le bouton droit de la souris sur le sprite du chat dans la liste et cliquez sur « supprimer ». Passez la souris sur l'icône Choisir un arrière-plan, puis cliquez sur l'icône Importer un arrière-plan dans la liste qui s'affiche. Récupérez le fichier **Space-background.png** dans le dossier Téléchargements, sélectionnez-le en cliquant dessus, puis cliquez sur OK. L'image de l'espace couvre alors le fond blanc de la scène et la zone de codage sera remplacée par l'arrière-plan (**Figure 4-11**). Vous pouvez dessiner sur l'image d'arrière-plan, mais pour l'instant, il suffit de cliquer sur l'onglet Code sur la partie supérieure de la fenêtre de Scratch 3.

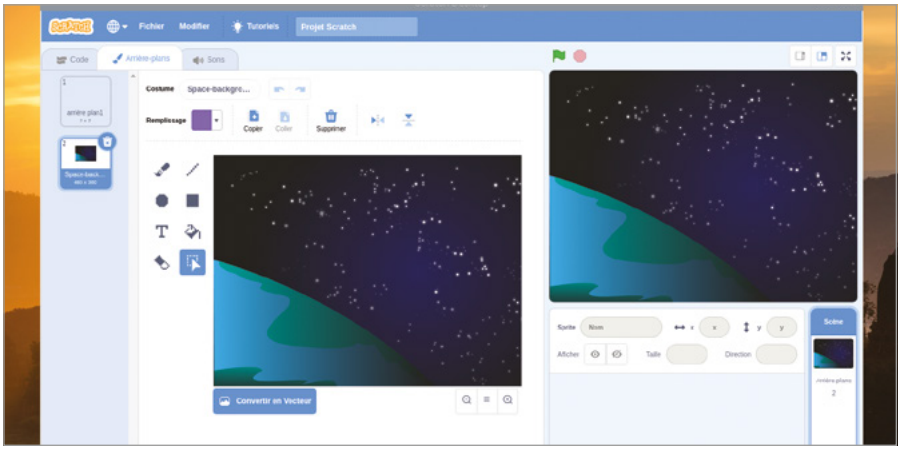


Figure 4-11 : L'image d'arrière-plan de l'espace s'affiche sur la scène

Téléchargez votre nouveau sprite en passant le curseur de la souris sur l'icône Choisir un sprite, puis cliquez sur l'icône Importer un sprite dans la liste qui s'affiche. Récupérez le fichier **Astronaut-Tim.png** dans le dossier Téléchargements, sélectionnez-le en cliquant dessus, puis cliquez sur OK. Le sprite apparaît automatiquement sur la scène, mais pas forcément en plein milieu : cliquez et faites-le glisser à l'aide de la souris et placez-le près du centre inférieur (**Figure 4-12**).

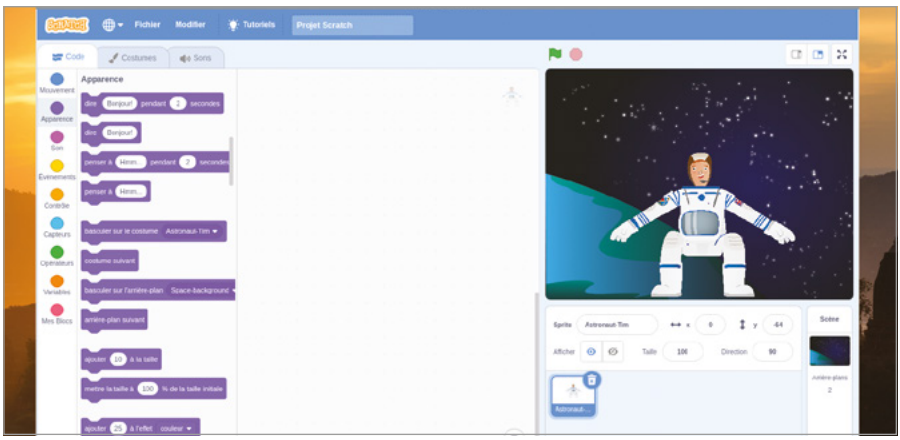


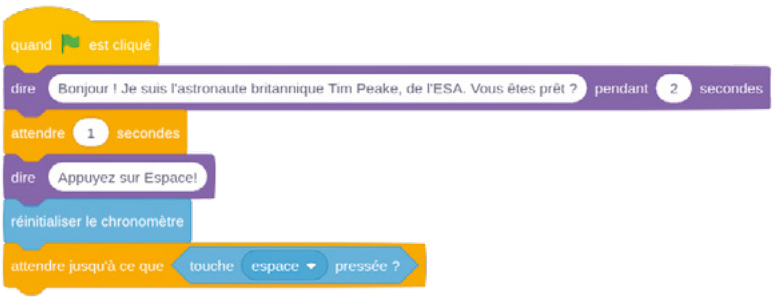
Figure 4-12 : Faites glisser le sprite de l'astronaute vers le centre inférieur de la scène

Une fois votre nouveau contexte et votre nouveau sprite en place, tout est prêt pour créer votre programme. Commencez par créer une nouvelle variable appelée « temps », en vous assurant que « pour tous les sprites » est bien sélectionné avant de cliquer sur OK. Cliquez sur votre sprite (soit sur la scène, soit dans le panneau des sprites) pour le sélectionner, puis ajoutez un bloc quand est cliqué de la catégorie Événements dans la zone de codage.

Ensuite, ajoutez un bloc **dire Bonjour! pendant 2 secondes** de la catégorie Apparence, puis cliquez dessus pour le modifier et lui faire dire « Bonjour ! Je suis l'astronaute britannique Tim Peake, de l'ESA. Vous êtes prêt ? »

Ensuite, ajoutez un bloc **attendre 1 secondes** de la catégorie Contrôle, puis un bloc **dire Bonjour!**. Modifiez ce bloc pour lui faire dire « Cliquez sur Espace ! », puis ajoutez un bloc **réinitialisater le chronomètre** dans la catégorie Capteurs. Ce bloc contrôle une variable spéciale intégrée dans Scratch pour mesurer le temps, et sera utilisé pour chronométrer votre vitesse de réaction dans le jeu.

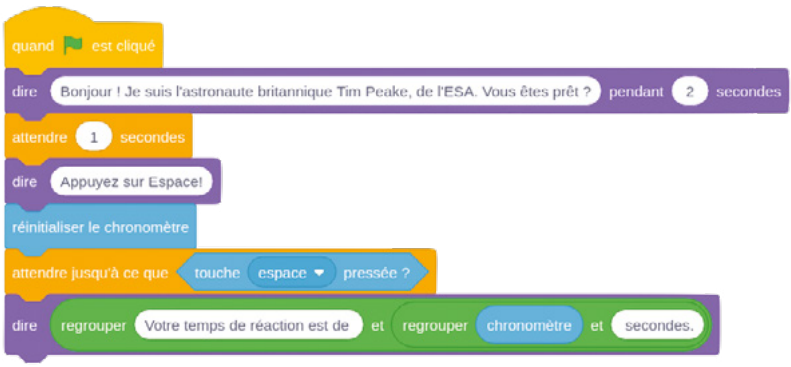
Ajoutez un bloc **attendre jusqu'à ce que** de la catégorie Contrôle, puis faites glisser un bloc Capteurs **touche espace pressée ?** dans l'espace vide. Ce faisant, le programme sera mis en pause jusqu'à ce que vous appuyiez sur la touche **ESPACE** du clavier, sans toutefois arrêter le chronomètre qui mesurera exactement le temps qui s'écoule entre le message « Appuyez sur Espace ! » et le moment où vous appuyez sur la touche **ESPACE**.



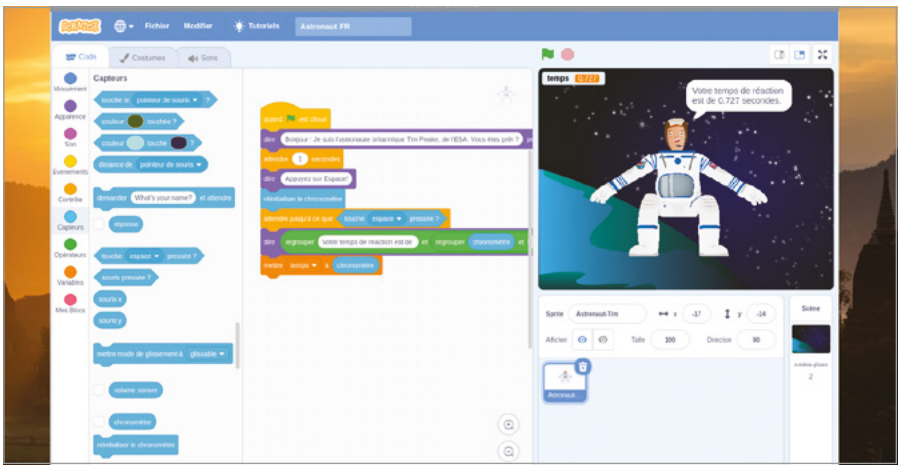
Vous devez maintenant demander à Tim, d'une manière très simple à lire, combien de temps il vous a fallu pour appuyer sur la touche **ESPACE**. Pour ce faire, vous aurez besoin d'un bloc **regrouper** de la catégorie Opérateurs. Ce bloc regroupe deux valeurs l'une après l'autre,

y compris des variables, en une opération dénommée *concaténation*.

Commencez par un bloc dire Bonjour!, puis faites glisser un bloc Opérateurs regrouper sur le mot « Bonjour! ». Cliquez sur « pomme » et saisissez « Votre temps de réaction a été de », en veillant à ajouter un espace vide à la fin, puis faites glisser un autre bloc regrouper sur « banane » dans la deuxième case. Faites glisser un bloc chronomètre de la catégorie Capteurs dans la case qui se trouve désormais au milieu, et saisissez le mot « secondes » dans la dernière case, en veillant à laisser un espace vide au début.



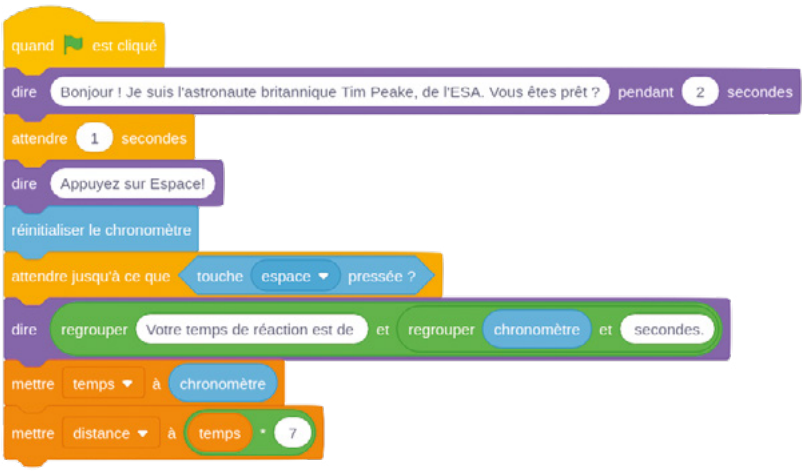
Enfin, faites glisser le bloc de la catégorie Variables mettre ma variable à 0 à la fin de votre séquence. Cliquez sur la flèche pointée vers le bas en regard de « ma variable » et cliquez sur « temps » dans la liste, puis remplacez le « 0 » par un bloc chronomètre de la catégorie Capteurs. Vous pouvez maintenant tester votre programme en cliquant sur le drapeau vert au-dessus de la scène. Préparez-vous, et dès que vous voyez le message « appuyez sur la barre **ESPACE** », appuyez sur la barre **ESPACE** le plus rapidement possible (**Figure 4-13**) : tentez de battre notre meilleur score !



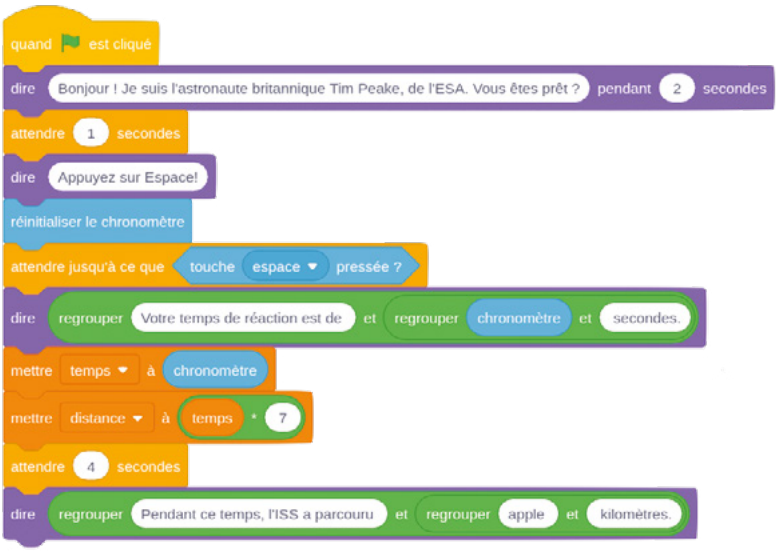
▲ **Figure 4-13 :** Et maintenant, jouons !

Vous pouvez compléter ce projet en lui demandant de calculer approximativement la distance parcourue par la Station spatiale internationale pendant le temps qu'il vous a fallu pour appuyer sur la touche **ESPACE**, sur la base de la vitesse officielle de la station, soit sept kilomètres par seconde. Tout d'abord, créez une nouvelle variable appelée « distance ». Vous remarquerez que les blocs de la catégorie Variables affichent automatiquement la nouvelle variable, mais les blocs de variables `temps` existants dans votre programme restent les mêmes.

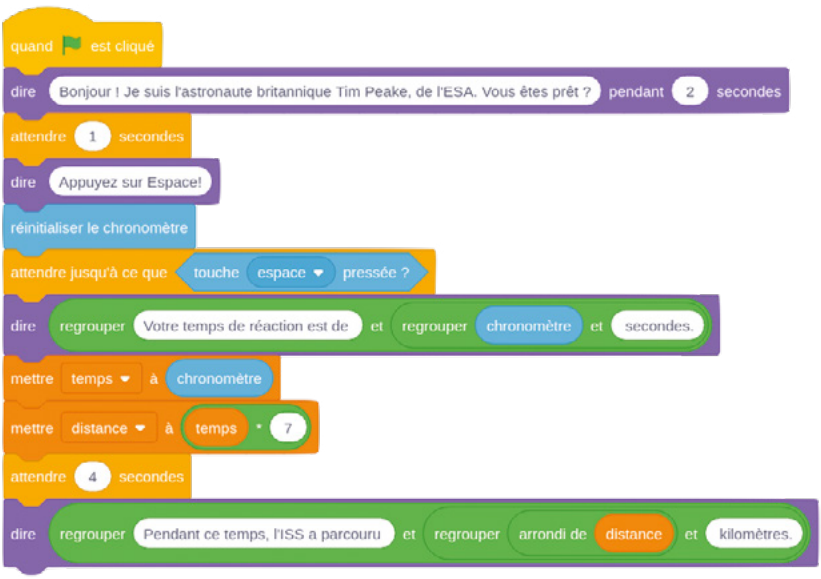
Ajoutez un bloc `mettre distance à 0` , puis faites glisser un bloc Opérateurs `( ) * ( )` , indiquant une multiplication, au-dessus du « 0 ». Faites glisser un bloc `temps` sur le premier espace vide, puis saisissez le chiffre « 7 » dans la deuxième case. Une fois terminé, votre bloc combiné indique `mettre distance à temps * 7` . Ce faisant, vous pouvez mesurer le temps qu'il vous a fallu pour appuyer sur la barre **ESPACE** et le multiplier par sept, pour obtenir la distance en kilomètres parcourue par l'ISS pendant ce temps.



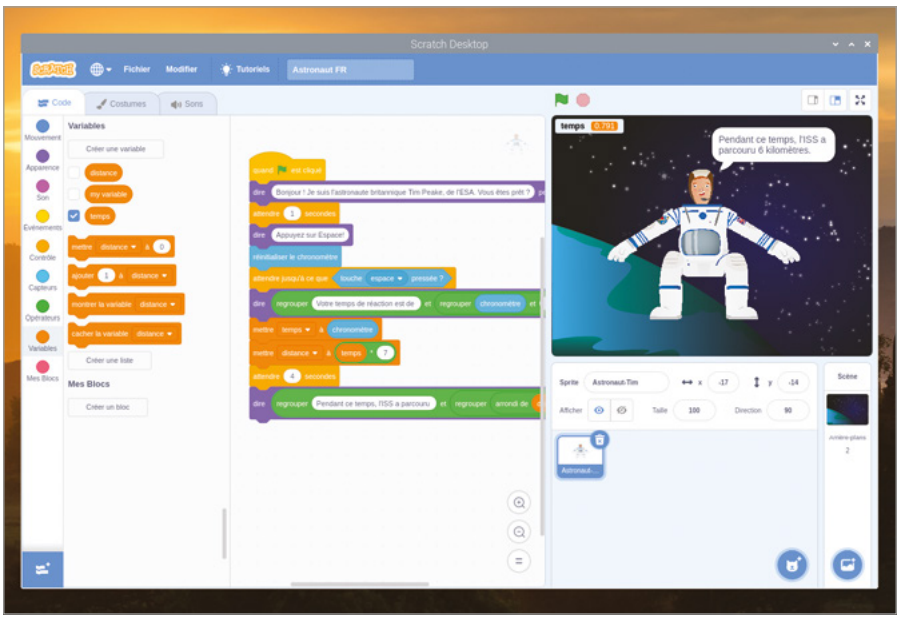
Ensuite, ajoutez un bloc `attendre 1 secondes` et modifiez-le sur « 4 ». Enfin, glissez un autre bloc `dire Bonjour!` à la fin de votre séquence et ajoutez deux blocs `regrouper`, comme vous l'avez fait auparavant. Dans le premier espace, dans « pomme », saisissez « Pendant ce temps, l'ISS a parcouru », sans oublier de laisser un espace à la fin ; dans « banane », saisissez « kilomètres », sans oublier de laisser un espace au début.



Enfin, faites glisser un bloc Opérateurs **arrondi de** dans l'espace vide au milieu, puis faites glisser un bloc de **distance** dans le nouvel espace vide qui s'est créé. Le bloc **arrondi de** sert à arrondir les chiffres à leur nombre entier le plus proche, donc au lieu d'un nombre de kilomètres hyper-précis mais difficile à lire, vous obtiendrez un nombre entier facile à lire.



Cliquez sur le drapeau vert pour lancer votre programme et voir la distance parcourue par l'ISS dans le temps qu'il vous faut pour appuyer sur la barre **ESPACE**. N'oubliez pas de sauvegarder votre programme lorsque vous l'avez terminé, pour le retrouver facilement à l'avenir sans avoir à recommencer depuis le début !



▲ **Figure 4-14 :** Tim vous indique le chemin parcouru par l'ISS

DÉFI : QUI EST LE PLUS RAPIDE ?

À part les astronautes, quelles autres professions doivent avoir des réflexes plus rapides que l'éclair ? Pouvez-vous dessiner vos propres sprites et arrière-plans pour représenter l'une de ces professions ?

Projet 2 : Natation synchronisée

La plupart des jeux utilisent plus d'un bouton, et ce projet vous l'explique en vous proposant un contrôle à deux boutons à l'aide des touches ← et → du clavier.

PROJET EN LIGNE

Ce projet est également disponible en ligne en cliquant sur **rpf.io/synchro-swimming**

Créez un nouveau projet et enregistrez-le sous le nom de « Natation synchronisée ». Cliquez sur la scène dans la section de contrôle de la scène, puis sur l'onglet Arrière-plan en haut à gauche. Cliquez sur le bouton Convertir en Bitmap en dessous de l'arrière-plan. Choisissez une couleur bleue semblable à l'eau dans la palette Remplissage, cliquez sur l'icône Remplissage, puis sur le fond à damiers pour le colorer en bleu (**Figure 4-15**).

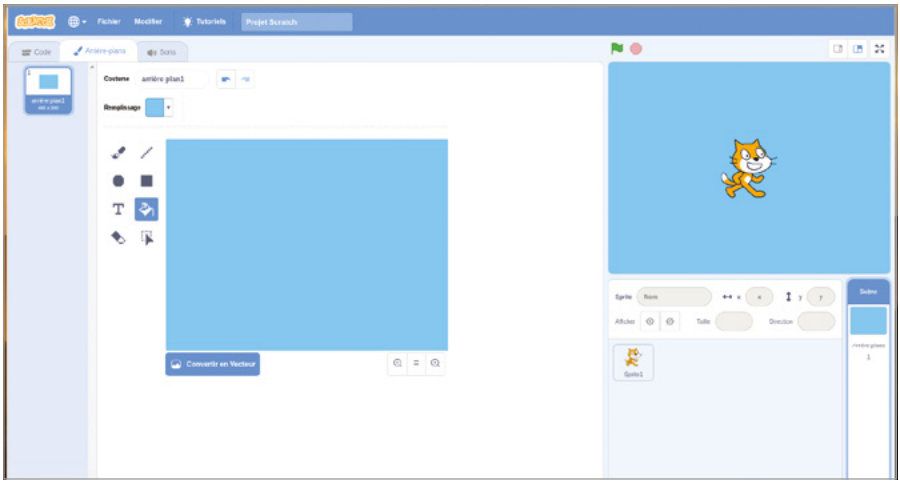


Figure 4-15 : Remplissage du fond en bleu

Cliquez avec le bouton droit de la souris sur le sprite du chat dans la liste et cliquez sur « supprimer ». Cliquez sur l'icône « Choisir un Sprite » pour afficher une liste des sprites intégrés. Cliquez sur la catégorie « Animaux », puis sur «Cat Flying » (**Figure 4-16**), et enfin sur OK. Ce sprite est parfaitement adapté à des projets sur la natation. Cliquez sur le nouveau sprite, puis faites glisser deux blocs Événements quand la touche espace est pressée dans la zone de codage. Cliquez sur la petite flèche vers le bas à côté du mot « espace » dans le premier bloc et choisissez « flèche gauche » dans la liste des options possibles. Faites glisser un bloc Mouvement tourner de 15 degrés sous votre bloc quand la flèche gauche est pressée, puis faites de même avec votre deuxième bloc Événements, mais en choisissant « flèche droite » dans la liste et en utilisant un bloc Mouvement tourner de 15 degrés.

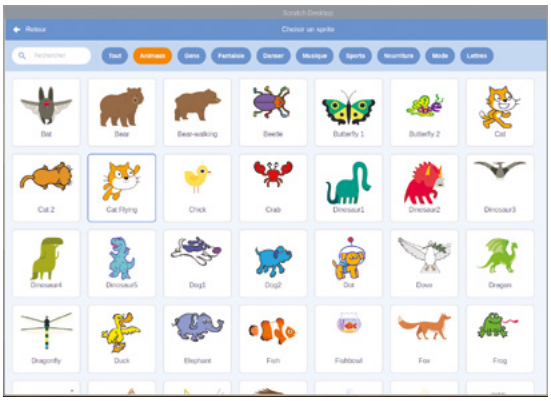


Figure 4-16 : Choisissez un sprite de la bibliothèque

Appuyez sur les touches ← ou → pour tester votre programme. Le chat commencera à se retourner dans la direction que vous lui indiquez sur le clavier. Vous remarquerez que, cette fois, vous n'avez pas eu besoin de cliquer sur le drapeau vert, car les blocs Événement déclencheurs que vous avez choisis sont toujours actifs, même lorsque le programme ne « s'exécute » pas à proprement parler.

Répétez les mêmes étapes deux fois, mais cette fois en choisissant « flèche haut » et « flèche bas » pour les blocs Événement déclencheurs, puis attachez les blocs Mouvement avancer de 10 pas et avancer de -10 pas. Appuyez maintenant sur les touches fléchées et vous verrez que votre chat peut se retourner et nager aussi bien en avant qu'en arrière !

Pour rendre le mouvement du sprite de chat plus réaliste, vous pouvez modifier son apparence, soit, en langage Scratch, son *costume*. Cliquez sur le sprite de chat, puis cliquez sur l'onglet Costumes au-dessus de la palette des blocs. Cliquez sur le costume de « cat-flying-a » et cliquez sur l'icône « X dans la corbeille », dans le coin supérieur droit, pour le

supprimer. Ensuite, cliquez sur le costume « cat flying-b » et, dans la case du nom en haut, attribuez-lui le nom de « droite » (**Figure 4-17**).

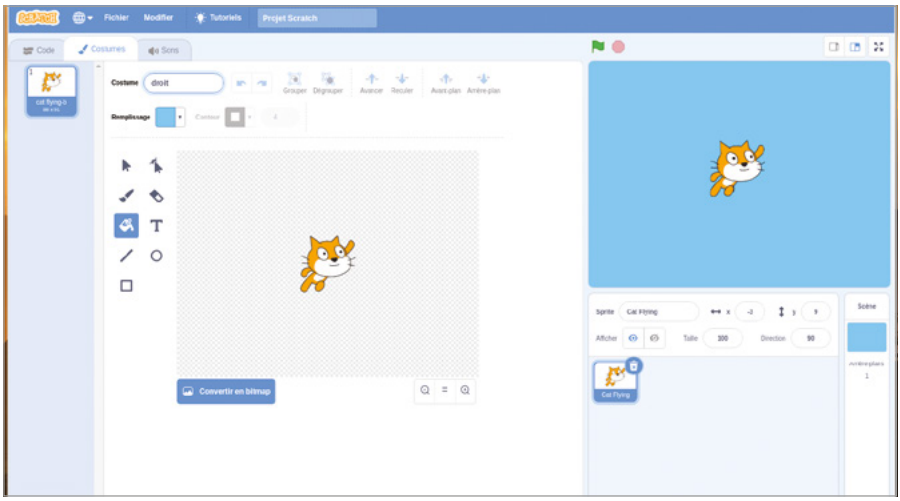


Figure 4-17 : Renommer le costume « droite »

Cliquez avec le bouton droit de la souris sur le costume « droite », que vous venez de renommer, et cliquez sur « dupliquer » pour créer une copie. Cliquez sur cette copie pour la sélectionner, cliquez sur l'icône Sélectionner, cliquez sur Retourner horizontalement , puis renommez-le « gauche » (**Figure 4-18**). Vous avec maintenant deux « costumes » pour votre sprite, qui sont des images en miroir identiques : le premier appelé « droite », avec le chat tourné vers la droite, l'autre appelé « gauche » avec le chat tourné vers la gauche.

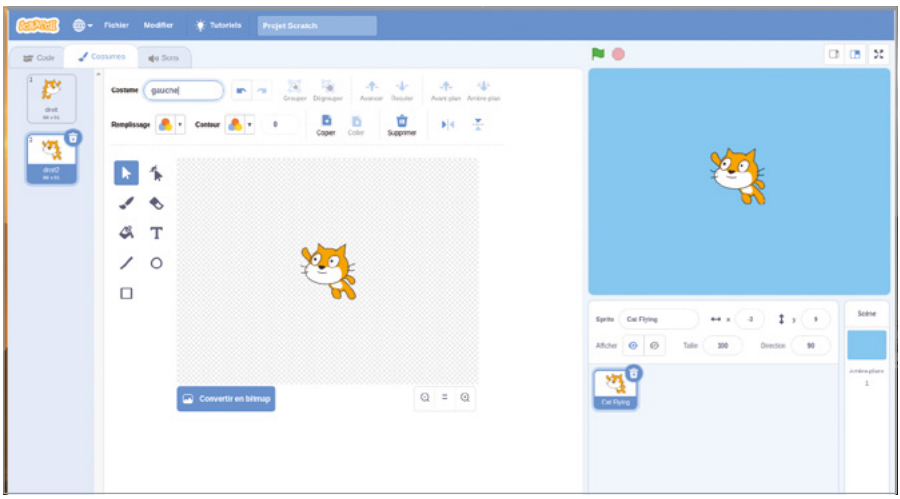
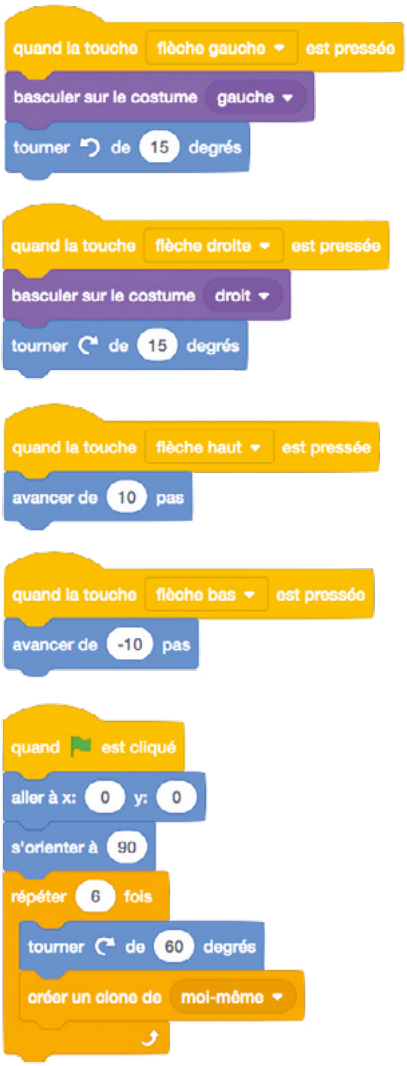


Figure 4-18 : Dupliquer le costume, le retourner et l'appeler « gauche »

Cliquez sur l'onglet Code au-dessus de la zone des costumes, puis faites glisser deux blocs Apparence **basculer sur le costume gauche** sous les blocs Événement flèche gauche et flèche droite, en modifiant celui que vous placez au-dessous du bloc flèche droite en **basculer sur le costume droit** . Essayez à nouveau les touches fléchées ; le chat se retourne pour se placer dans direction dans laquelle il nage.

En revanche, pour la natation synchronisée olympique, nous avons besoin de plus de nageurs, et nous devons trouver le moyen de réinitialiser la position du chat. Ajoutez un bloc Événements quand est cliqué, puis ajoutez en dessous un bloc Mouvement aller à x : 0 y : 0, en modifiant les valeurs au besoin, ainsi qu'un bloc Mouvement s'orienter à 90. Maintenant, lorsque vous cliquez sur le drapeau vert, le chat se déplace au milieu de la scène en pointant vers la droite.

Pour créer de nouveaux nageurs, ajoutez un bloc répéter 6 fois, en modifiant la valeur par défaut de 10, et ajoutez un bloc Contrôle créer un clone de moi-même à l'intérieur. Pour que les nageurs ne nagent pas tous dans la même direction, ajoutez un bloc tourner de 60 degrés au-dessus du bloc créer un clone mais toujours à l'intérieur du bloc répéter 6 fois. Cliquez sur le drapeau vert, et essayez les touches fléchées dès maintenant pour voir vos nageurs prendre vie !



Pour compléter l'ambiance olympique, il faut ajouter de la musique. Cliquez sur l'onglet Sons au-dessus de la palette des blocs, puis cliquez sur le symbole « Choisir un son » . Cliquez sur la catégorie Boucles, puis parcourez la liste (**Figure 4-19**) jusqu'à ce que vous trouviez de la musique que vous aimez : pour notre part, nous avons choisi « Dance Around ». Cliquez sur le bouton OK pour choisir la musique, puis cliquez sur l'onglet Code pour ouvrir à nouveau la zone de codage.

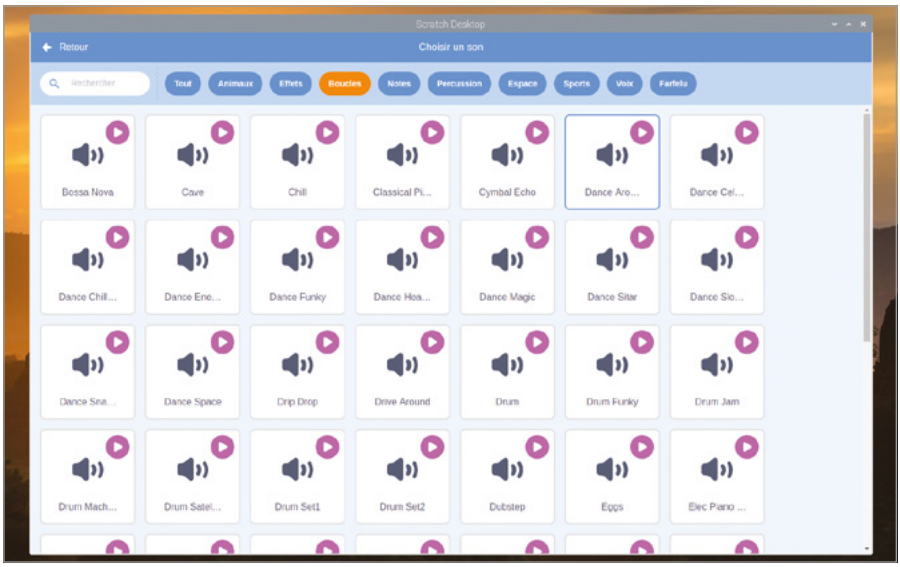
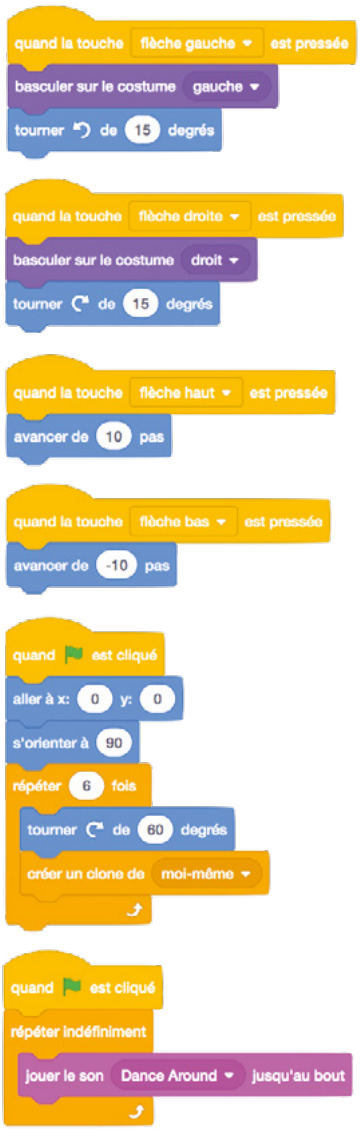
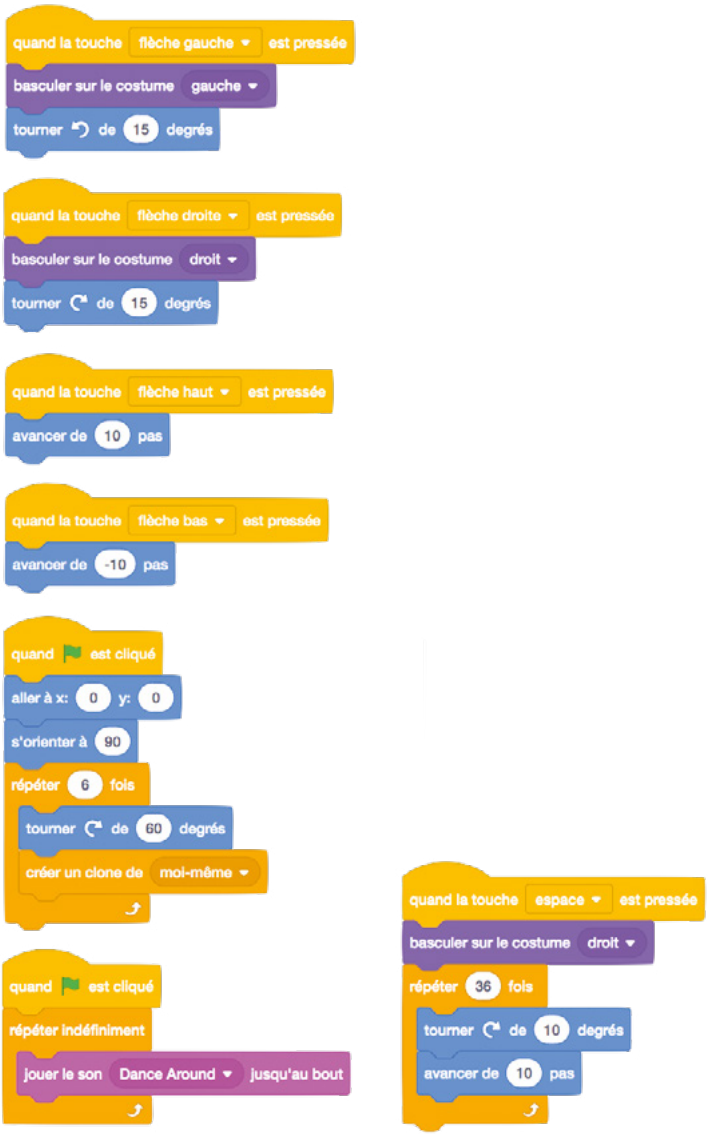


Figure 4-19 : Sélectionnez une boucle musicale dans la bibliothèque de sons

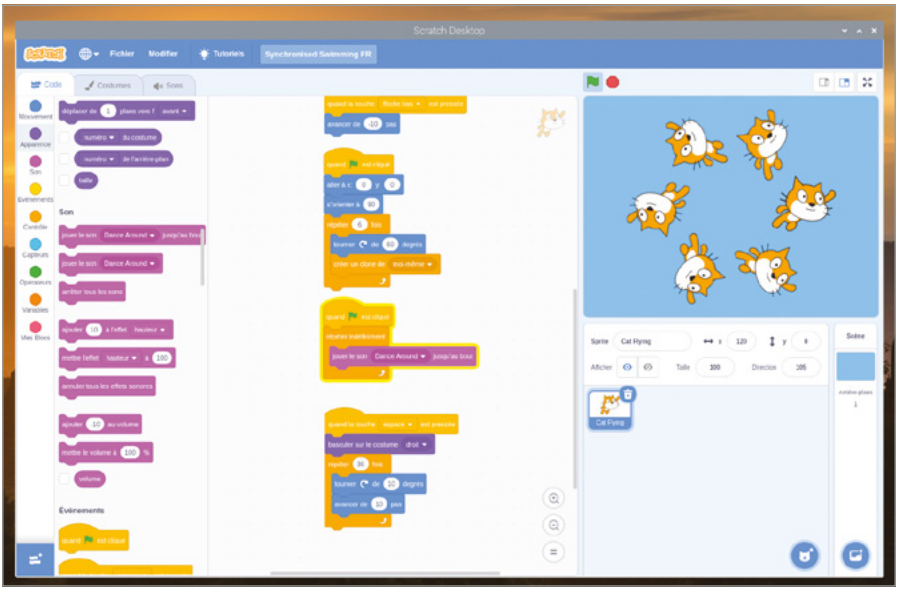
Ajoutez à votre zone de codage un autre bloc Événements quand est cliqué, puis ajoutez un bloc Contrôle répéter indéfiniment . Dans ce bloc Contrôle, ajoutez un bloc jouer le son Dance Around jusqu'au bout, sans oublier de rechercher le nom du morceau de musique que vous avez choisi, et cliquez sur le drapeau vert pour tester votre nouveau programme. Si vous voulez arrêter la musique, cliquez sur l'octogone rouge pour arrêter le programme et arrêter le son !



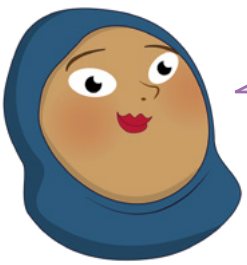
Enfin, vous pouvez simuler une routine de danse complète en ajoutant un nouvel événement déclencheur à votre programme. Ajoutez un bloc Événements **quand la touche espace est pressée**, puis un bloc **basculer sur le costume droit**. En dessous, ajoutez un bloc **répéter 36 fois** (sans oublier de modifier la valeur par défaut) et à l'intérieur de celui-ci un bloc **tourner de 10 degrés** et un bloc **avancer 10 pas**.



Cliquez sur le drapeau vert pour lancer le programme, puis appuyez sur la barre **ESPACE** pour tester la nouvelle routine (**Figure 4-20**, au verso) ! N'oubliez pas de sauvegarder votre programme lorsque vous avez terminé.



▲ **Figure 4-20 :** La routine de natation synchronisée mise au point



DÉFI : LA ROUTINE HABITUELLE

Pouvez-vous créer votre propre routine de natation synchronisée en utilisant des boucles ? Que devriez-vous modifier si vous vouliez ajouter ou retirer des nageurs ? Pouvez-vous ajouter plusieurs routines de natation qui peuvent être lancées en appuyant sur différentes touches du clavier ?

Projet 3 : Jeu de tir à l'arc

Maintenant que vous maîtrisez un peu Scratch, il est temps de passer à quelque chose de plus difficile : un jeu de tir à l'arc, où le joueur doit atteindre une cible à l'aide d'un arc et de flèches qui se déplacent de façon aléatoire.

PROJET EN LIGNE

Ce projet est également disponible en ligne en cliquant sur **rpf.io/archery**

Ouvrez le navigateur Web Chromium et saisissez **rpf.io/p/en/archery-go**, puis appuyez sur la touche **ENTRÉE**. Les ressources du jeu se téléchargent sous forme d'un fichier compressé, que

vous devez décompresser (cliquez avec le bouton droit de la souris et sélectionnez Extraire ici). Revenez à Scratch 3 et cliquez sur le menu Fichier, puis sur « Charger depuis votre ordinateur (Load from Computer) ». Cliquez sur **ArcheryResources.sb3**, puis sur le bouton Ouvrir. Il vous sera demandé si vous souhaitez remplacer le contenu de votre projet en cours : si vous n'avez pas enregistré vos modifications, cliquez sur Annuler et enregistrez-les, sinon cliquez sur OK.

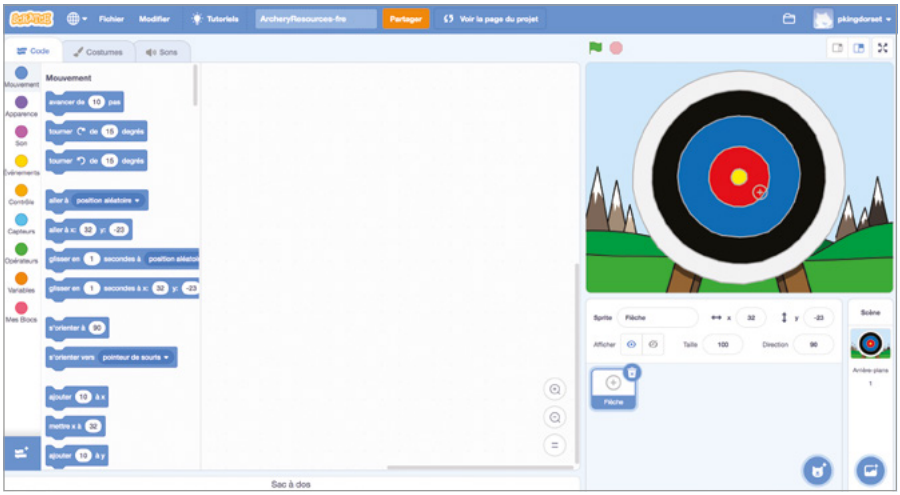


Figure 4-21 : Ressources du projet du jeu de tir à l'arc chargées

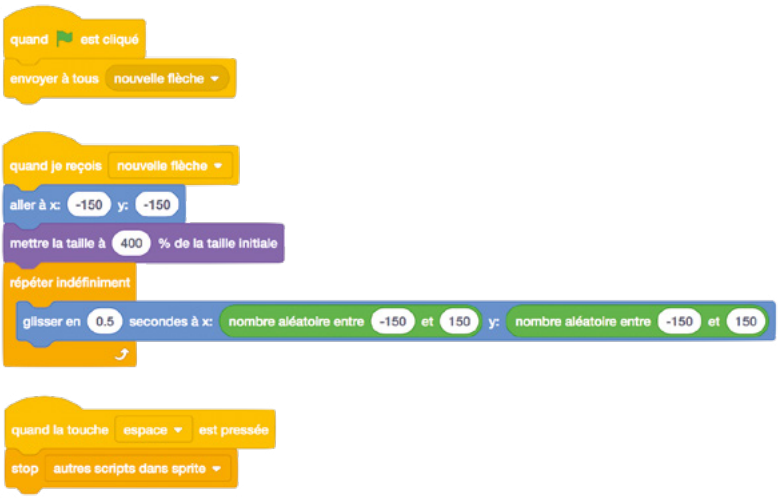
Le projet que vous venez de charger contient un arrière-plan et un sprite (**Figure 4-21**), mais il ne contient pas le code nécessaire à la réalisation d'un jeu : ce sera donc à vous de vous en charger. Commencez par un bloc quand est cliqué, puis ajoutez un bloc envoyer à tous message1. Cliquez sur la flèche vers le bas à la fin du bloc, puis sur « Nouveau message », et saisissez « nouvelle flèche » avant de cliquer sur le bouton OK. Votre bloc indique maintenant envoyer à tous nouvelle flèche.

Un envoi est un message d'une partie de votre programme qui peut être reçue par toute autre partie. Pour que cet envoi ait un effet quelconque, ajoutez un bloc quand je reçois le message1 et modifiez-le de sorte qu'il indique quand je reçois nouvelle flèche. Cette fois-ci, il suffit de cliquer sur la flèche vers le bas et de choisir « Nouvelle flèche » dans la liste ; vous n'avez pas à créer de nouveau le message.

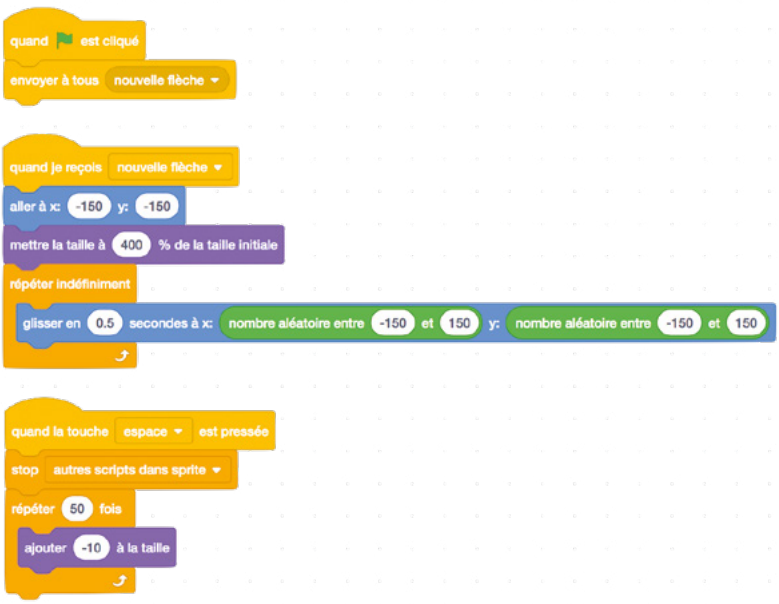
Sous votre bloc quand je reçois nouvelle flèche, ajoutez un bloc aller à x: -150 y: -150 et un bloc mettre la taille à 400 % de la taille initiale. N'oubliez pas que ce ne sont pas les valeurs par défaut de ces blocs, vous devrez donc les modifier une fois que vous les aurez fait glisser sur la zone de code. Cliquez sur le drapeau vert pour voir ce que vous avez créé jusqu'à présent : le sprite flèche, que le joueur utilise pour viser la cible, se positionne en bas à gauche de la scène et sa taille est multipliée par quatre.

Pour indiquer un défi au joueur, ajoutez un mouvement de balancement lorsque l'arc est mis en position et l'archer vise la cible. Faites glisser un bloc répéter indéfiniment, puis un bloc glisser en 1 secondes à x: -150 y: -150. Modifiez la première case blanche pour qu'elle indique « 0,5 » au lieu de « 1 », puis placez un bloc Opérateur nombre aléatoire entre -150 et 150 dans chacune des deux autres cases blanches. Ainsi, la flèche va dériver autour de la scène dans une direction et sur une distance aléatoires, ce qui rend la cible beaucoup plus difficile à atteindre !

Cliquez à nouveau sur le drapeau vert, et vous verrez quel est l'effet de ce bloc : votre sprite de flèche se déplace maintenant autour de la scène, couvrant différentes parties de la cible. Pour l'instant, cependant, vous ne pouvez absolument pas tirer la flèche vers la cible. Faites glisser un bloc quand la touche espace est pressée dans votre zone de codage, suivi d'un bloc Contrôle stop tout. Cliquez sur la flèche vers le bas à la fin du bloc et modifiez-le de sorte qu'il indique stop autres scripts dans sprite.

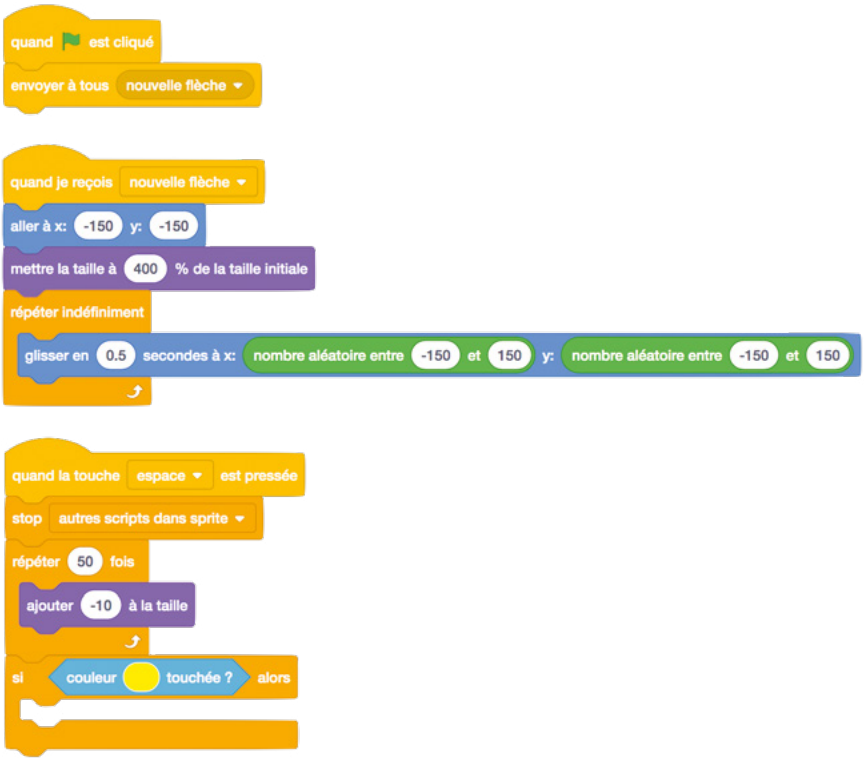


Si vous aviez arrêté votre programme pour ajouter les nouveaux blocs, cliquez sur le drapeau vert pour le relancer, puis appuyez sur la touche **ESPACE** : le sprite flèche s'arrêtera de bouger. C'est un début, mais il faudra bien que la flèche atteigne la cible. Ajoutez un bloc répéter 50 fois, puis un bloc ajouter -10 à la taille, puis cliquez sur le drapeau vert pour tester à nouveau votre jeu. Cette fois, la flèche semble s'éloigner de vous en direction de la cible.

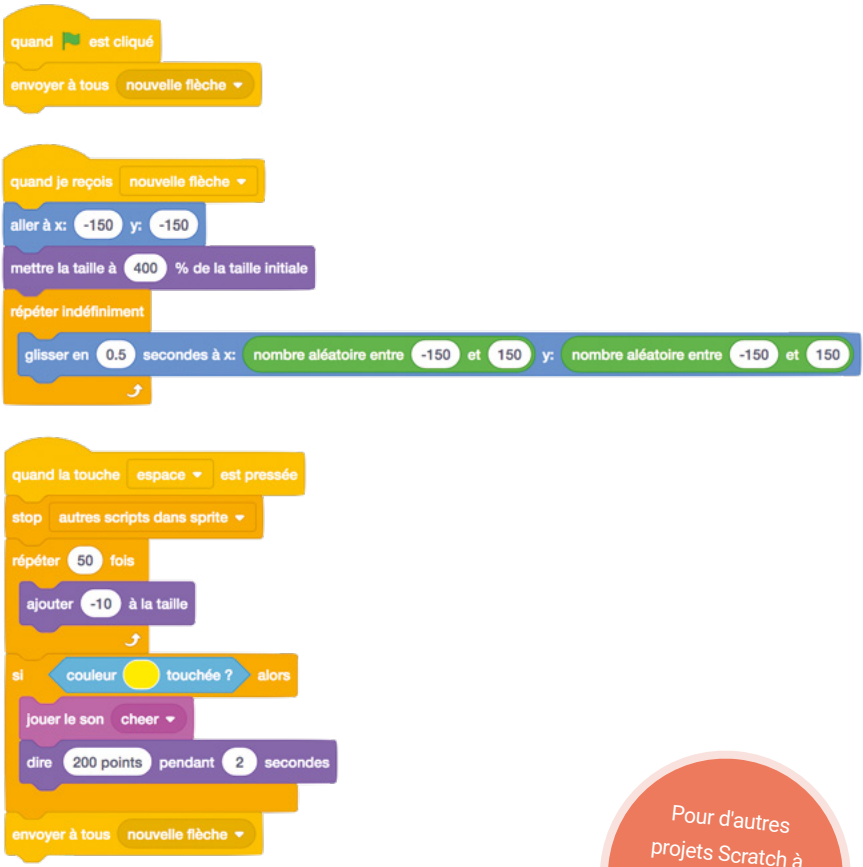


Pour que le jeu soit amusant, il faut trouver un moyen de gagner des points. Toujours dans la même pile de blocs, ajoutez un bloc si alors en s'assurant qu'il est en dessous du bloc répéter 50 fois et non à l'intérieur, avec un bloc Capteurs couleur touchée ? bloc détecteur

dans son espace vide en forme de losange. Pour choisir la bonne couleur, cliquez sur la case colorée à la fin du bloc Capteurs, puis sur l'icône de pipette, et enfin sur le centre jaune de votre cible sur la scène.



Pour que le joueur marque des points, ajoutez un bloc **jouer le son cheer** et un bloc **dire 200 points pendant 2 secondes** à l'intérieur du bloc **si alors**. Enfin, ajoutez un bloc **envoyer à tous nouvelle flèche** tout au fond de la pile de blocs, au-dessous et en dehors du bloc **si alors**, pour donner une nouvelle flèche au joueur chaque fois qu'il en tire une. Cliquez sur le drapeau vert pour commencer votre jeu, et essayez d'atteindre le centre de la cible jaune : quand vous y parviendrez, vous serez récompensé par les acclamations de la foule et un score de 200 points !



Pour d'autres projets Scratch à essayer, consultez l'**Annexe D**

Le jeu fonctionne, mais il est un peu difficile. En utilisant ce que vous avez appris dans ce chapitre, essayez de le rendre plus complexe, attribuant des points à chaque fois que vous atteignez la cible, même si ce n'est pas en plein dans le mille : 100 points pour le rouge, 50 points pour le bleu, et ainsi de suite.

DÉFI : POUVEZ-VOUS L'AMÉLIORER ?

Comment simplifieriez-vous le jeu ? Comment le rendre plus complexe ? Pouvez-vous utiliser des variables pour que le score du joueur augmente à mesure qu'il tire plus de flèches ? Pouvez-vous ajouter un compte à rebours pour mettre plus de pression sur le joueur ?

Chapitre 5

Programmation avec Python

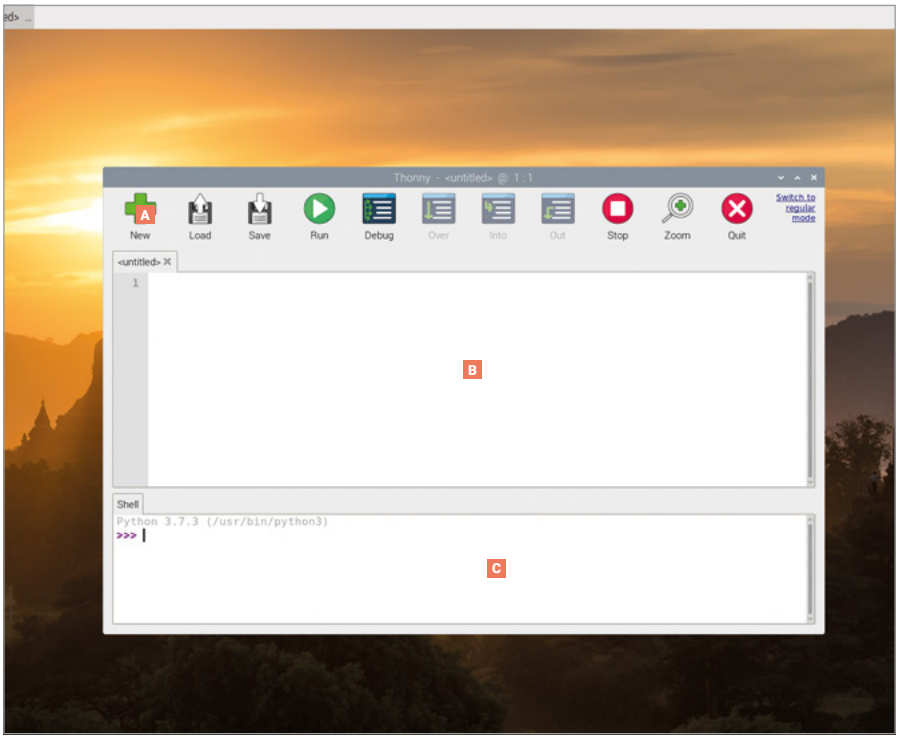
Vous êtes venu à bout de Scratch : nous allons maintenant passer au codage en utilisant le langage Python

Tirant son nom de la troupe de comédiens Monty Python, Python a été inventé par Guido van Rossum, au départ comme un passe-temps, avant de faire son apparition publique en 1991 et de devenir un langage de programmation très apprécié qui alimente un large éventail de projets. Contrairement à l'environnement visuel de Scratch, Python est basé sur le texte : vous écrivez des instructions, en utilisant un langage simplifié et un format spécifique, que l'ordinateur exécute ensuite.

Python est une nouvelle étape importante pour ceux qui ont déjà utilisé Scratch, offrant une plus grande flexibilité et un environnement de programmation plus « traditionnel ». Mais cela ne veut pas dire qu'il est difficile à apprendre : avec un peu de pratique, n'importe qui peut écrire des programmes Python pour tout faire, depuis des calculs très simples jusqu'à des jeux extrêmement compliqués.

Le présent chapitre reprend certains des termes et des concepts introduits dans le **Chapitre 4 Programmation avec Scratch 3**. Si vous ne les avez pas encore faits, il serait préférable de revenir en arrière et faire ces exercices pour que la lecture de ce chapitre soit beaucoup plus simple.

Présentation de l'IDE Thonny Python



A **Barre d'outils** : l'interface Simple Mode de Thonny utilise une barre d'icônes conviviales en guise de menu, vous permettant de créer, sauvegarder, charger et exécuter vos programmes Python et de les tester de plusieurs manières différentes.

B **Zone de script** : la zone de script est le lieu où vous rédigez vos programmes Python. Elle se compose de la zone principale, dédiée à votre programme, et d'une marge latérale étroite affichant les numéros de ligne.

C **Shell Python** : le shell Python vous permet de saisir des instructions individuelles qui sont ensuite exécutées dès que vous appuyez sur la touche **ENTRÉE** et fournit également des informations sur les programmes en cours d'exécution.

VERSIONS DE THONNY

Thonny possède deux versions d'interface : Un « Normal Mode » et un « Simple Mode », idéal pour les débutants. Ce chapitre utilise le Simple Mode, qui est chargé par défaut lorsque vous ouvrez Thonny depuis la section Programmation du menu obtenu en cliquant sur la framboise.

Votre tout premier programme Python : Bonjour tout le monde !

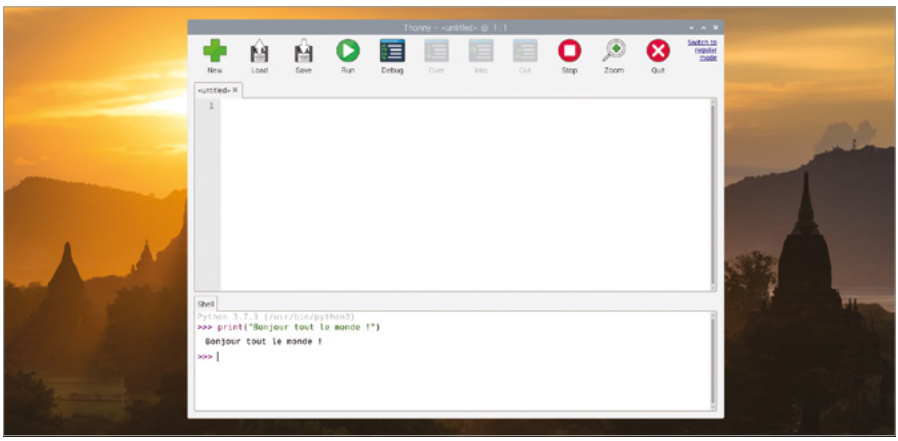
Tout comme n'importe quel autre programme préinstallé sur Raspberry Pi , Thonny est disponible à partir du menu : cliquez sur l'icône représentant une framboise, déplacez le curseur sur la section Programmation et cliquez sur IDE Thonny Python. Après quelques secondes, l'interface utilisateur de Thonny (Simple Mode par défaut) sera chargée.

Thonny est un package connu sous le nom de *environnement de développement intégré (IDE)*, terme rébarbatif mais qui s'explique très simplement : il rassemble, ou *intègre*, les différents outils dont vous avez besoin pour écrire, ou *développer* des logiciels dans une interface utilisateur unique, ou *environnement*. Il existe de nombreux IDE, dont certains prennent en charge de nombreux langages de programmation différents, tandis que d'autres, comme Thonny, se concentrent sur la prise en charge d'un seul langage.

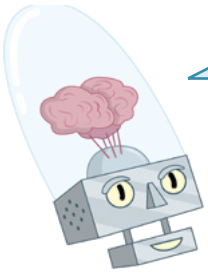
Contrairement à Scratch, qui vous propose des blocs de construction visuels pour créer votre programme, Python est un langage de programmation plus traditionnel basé sur l'écriture. Créez votre premier programme en cliquant sur la zone du shell Python en bas à gauche de la fenêtre de Thonny, puis tapez l'instruction suivante avant d'appuyer sur la touche **ENTRÉE** :

```
print("Bonjour tout le monde !")
```

Lorsque vous appuyez sur **ENTRÉE**, vous verrez que votre programme commence à fonctionner instantanément : Python répondra, dans la même zone du shell, avec le message « Bonjour tout le monde ! » (**Figure 5-1**), comme vous l'avez demandé. C'est parce que le shell constitue une ligne directe avec l'*interprète* Python, dont le rôle consiste à examiner vos instructions et à *interpréter* leur signification. C'est ce que l'on appelle un *mode interactif*, qui peut être apparenté à une conversation en face à face avec quelqu'un : vous dites quelque chose, l'autre personne vous répond et attend ce que vous direz ensuite.



▲ **Figure 5-1 :** Python imprime le message « Bonjour tout le monde ! » dans la zone du shell



ERREUR DE SYNTAXE

Si votre programme ne s'exécute pas mais imprime un message « syntax error » dans la zone du shell, cela signifie que ce que vous avez écrit contient une erreur quelque part. L'écriture des instructions de Python doit suivre des normes très spécifiques : si vous oubliez une parenthèse ou un guillemet, si vous écrivez mal « print » ou si vous mettez un P majuscule, ou si vous ajoutez des symboles supplémentaires quelque part dans l'instruction, celle-ci ne fonctionnera pas. Essayez de saisir à nouveau les instructions et assurez-vous qu'elles correspondent exactement à la version ici décrite avant d'appuyer sur la touche **ENTRÉE** !

Vous n'êtes pas forcé d'utiliser Python en mode interactif, cependant. Cliquez sur la zone de script située à gauche de la fenêtre de Thonny, puis saisissez à nouveau votre code :

```
print("Bonjour tout le monde !")
```

Lorsque vous appuyez sur la touche **ENTRÉE**, cette fois-ci, il ne se passe rien, sauf l'apparition d'une ligne en blanc dans la zone de script. Pour que cette version de votre programme fonctionne, vous devez cliquer sur l'icône Run dans la barre d'outils de Thonny. Ensuite, il vous sera demandé de sauvegarder d'abord votre programme ; entrez un nom descriptif, comme « Bonjour tout le monde » et cliquez sur le bouton « Save ». Une fois votre programme enregistré, vous verrez deux messages s'afficher dans la zone du shell Python (**Figure 5-2**) :

```
>>> %Run 'Bonjour tout le monde.py'
    Bonjour tout le monde !
```

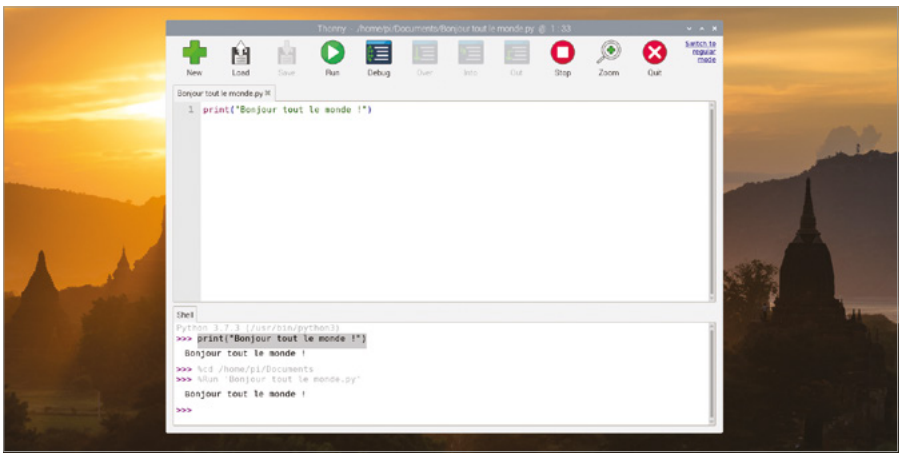


Figure 5-2 : Exécuter votre programme tout simple

La première de ces lignes est une instruction de Thonny disant à l'interprète Python d'exécuter le programme que vous venez de sauvegarder. La seconde est la sortie du programme, c'est-à-dire le message que vous avez demandé à Python d'imprimer. Félicitations : vous avez écrit et exécuté votre premier programme Python en mode interactif et en mode script !

DÉFI : NOUVEAU MESSAGE

Pouvez-vous modifier le message que le programme Python imprime en sortie ? Si vous vouliez ajouter des messages, utiliseriez-vous le mode interactif ou le mode script ? Que se passe-t-il si vous retirez les parenthèses ou les guillemets du programme et que vous essayez de le relancer ?.

Prochaines étapes : boucles et indentation de code

Tout comme Scratch utilise des piles de blocs ressemblant à des pièces de puzzle pour contrôler quelles parties du programme se connectent à quelles autres parties, Python a sa propre façon de contrôler l'ordre dans lequel ses programmes s'exécutent : l'*indentation*. Créez un nouveau programme en cliquant sur l'icône ✚ New dans la barre d'outils Thonny. Vous ne perdrez pas votre programme existant ; Thonny créera un nouvel onglet au-dessus de la zone de script. Saisissez le code suivant :

```
print("Début de boucle")
for i in range(10):
```

La première ligne imprime un message simple dans le shell, tout comme votre programme Bonjour tout le monde. La deuxième commence une boucle *définie*, qui fonctionne exactement comme dans Scratch : un compteur, **`i`**, est affecté à la boucle et reçoit une série de nombres, à savoir les instructions relatives à la **`range`**, dont le but est de commencer à compter à partir de 0 pour remonter vers le nombre 10, mais sans jamais l'atteindre. Le symbole des deux points (**`:`**) indique à Python que l'instruction suivante doit être incluse dans la boucle.

Dans Scratch, les instructions à inclure dans la boucle sont incluses à l'intérieur du bloc en forme de C. Python utilise une approche différente : l'indentation du code. La ligne suivante s'ouvre par quatre espaces vides, que Thonny aurait dû ajouter lorsque vous avez appuyé sur **ENTRÉE** après la ligne 2 :

```
    print("Numéro de boucle", i)
```

Les espaces vides repoussent la ligne vers l'intérieur par rapport aux autres lignes. C'est l'indentation qui permet à Python de faire la différence entre les instructions à l'extérieur de la boucle et les instructions à l'intérieur de la boucle ; le code indenté est forcément imbriqué.

Vous remarquerez que, lorsque vous avez appuyé sur **ENTRÉE** à la fin de la troisième ligne, Thonny a automatiquement indenté la ligne suivante, en supposant qu'elle fait partie de la boucle. Pour supprimer l'indentation, il suffit d'appuyer sur la touche **RETOUR ARRIÈRE** avant de saisir la quatrième ligne :

```
print("Fin de boucle !")
```

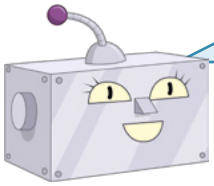
Votre code de quatre lignes est terminé. La première ligne se trouve à l'extérieur de la boucle et ne sera exécutée qu'une seule fois ; la deuxième ligne met en place la boucle ; la troisième se trouve à l'intérieur de la boucle et sera exécutée une fois pour chaque boucle ; la quatrième ligne se trouve à nouveau à l'extérieur de la boucle.

```
print("Début de boucle")
for i in range (10):
    print("Numéro de boucle", i)
print("Fin de boucle !")
```

Cliquez sur l'icône Run, enregistrez le programme sous **Indentation** et affichez la zone du shell pour sa sortie (**Figure 5-3**) :

```
Début de boucle
Numéro de boucle 0
Numéro de boucle 1
Numéro de boucle 2
Numéro de boucle 3
Numéro de boucle 4
Numéro de boucle 5
Numéro de boucle 6
Numéro de boucle 7
Numéro de boucle 8
Numéro de boucle 9
Fin de boucle !
```

Figure 5-3 : Exécution d'une boucle



COMPTER À PARTIR DE ZÉRO

Python est un langage indexé à zéro, ce qui signifie qu'il commence à compter à partir de 0, et non de 1. C'est pourquoi votre programme imprime les chiffres de 0 à 9 plutôt que de 1 à 10. Si vous le souhaitiez, vous pourriez modifier ce comportement en passant de l'instruction range (10) à range (1, 11), ou à tout autre nombre de votre choix.

L'indentation est un élément essentiel de Python et l'une des raisons fréquentes pour lesquelles un programme ne fonctionne pas comme vous l'espériez. Lorsque vous recherchez des problèmes dans un programme, un processus connu sous le nom de *débogage*, vous devez toujours vérifier l'indentation, surtout lorsque vous commencez à imbriquer des boucles les unes dans les autres.

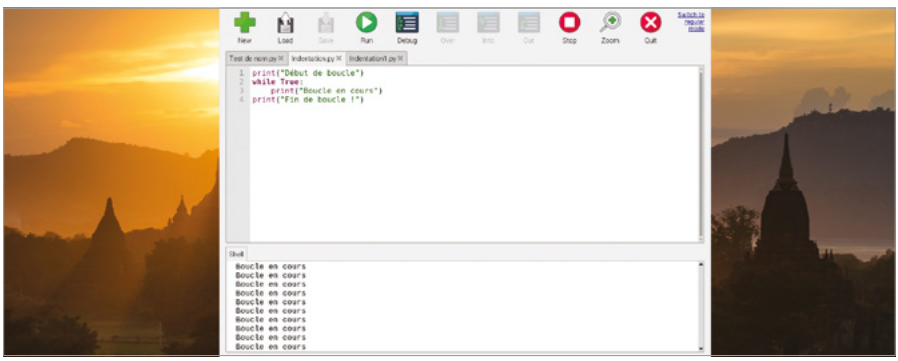
Python prend en charge le boucles *infinies*, qui s'exécutent sans fin. Pour changer votre programme d'un programme à boucle définie à un programme à boucle infinie, modifiez la ligne 2, comme suit :

```
while True:
```

Si vous cliquez maintenant sur l'icône Run, vous obtiendrez une erreur **`name 'i' is not defined`**. Cela est dû au fait que vous avez supprimé la ligne qui a créé et attribué une valeur à la variable **`i`**. Pour y remédier, il suffit de modifier la ligne 3 pour qu'elle n'utilise plus la variable :

```
    print("Boucle en cours")
```

Cliquez sur l'icône Run et, si vous êtes rapide, vous verrez le message « Début de boucle » suivi d'une chaîne ininterrompue de messages « Boucle en cours » (**Figure 5-4**). Le message « Fin de boucle » ne s'imprime jamais car la boucle est infinie : chaque fois que Python termine d'imprimer le message « Boucle en cours », il revient au point de départ de la boucle et l'imprime une nouvelle fois.



▲ **Figure 5-4 :** Une boucle infinie continue jusqu'à ce que vous arrêtiez le programme

Cliquez sur l'icône Stop de la barre d'outils Thonny pour indiquer au programme de s'interrompre, une action connue sous le nom d'interruption du programme. Un message s'affichera alors dans la zone du shell Python, et le programme s'arrêtera, sans ne jamais atteindre la ligne 4.

DÉFI : BOUCLER LA BOUCLE

Pouvez-vous revenir en arrière vers une boucle définie ? Pouvez-vous ajouter une deuxième boucle définie au programme ? Comment ajouter une boucle à l'intérieur d'une boucle, et comment pensez-vous que cela fonctionnera ?

Conditions et variables

Comme dans tous les langages de programmation, le rôle des variables va bien au-delà du contrôle des boucles. Commencez un nouveau programme en cliquant sur l'icône New dans le menu de Thonny, puis saisissez le code suivant dans la zone de script :

```
userName = input ("Comment vous appelez-vous ?")
```

Cliquez sur l'icône Run, enregistrez votre programme sous **Test de nom** et observez ce qui se passe dans la zone du shell : on vous demandera votre nom. Saisissez votre nom dans la zone du shell, puis appuyez sur **ENTRÉE**. Comme c'est la seule instruction de votre programme, il ne se passera rien d'autre (**Figure 5-5**). Si vous souhaitez exploiter les données que vous avez placées dans la variable, vous aurez besoin d'ajouter des lignes à votre programme.

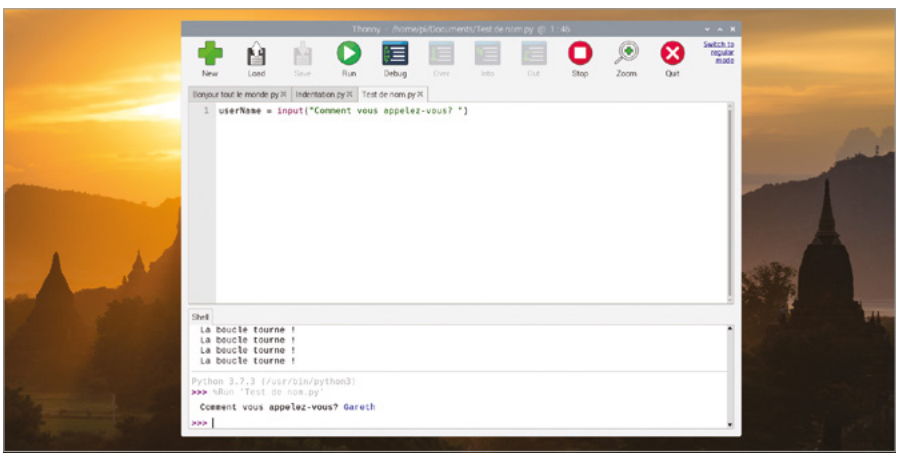


Figure 5-5 : La fonction `input` permet de demander à un utilisateur de saisir du texte

Pour que votre programme exploite la donnée du nom, ajoutez une *déclaration conditionnelle* en saisissant ce qui suit :

```
if userName == "Clark Kent":
    print("Vous êtes Superman !")
else:
    print("Vous n'êtes pas Superman !")
```

N'oubliez pas que lorsque Thonny verra que votre code doit être indenté, il le fera automatiquement, mais il ne sait pas à quel moment l'indentation doit cesser et vous devrez donc supprimer les espaces vous-même.

Cliquez sur l'icône Run et saisissez votre nom dans la zone du shell. Si vous n'êtes pas Clark Kent, vous verrez le message « Vous n'êtes pas Superman ! ». Cliquez à nouveau sur Run et cette fois, entrez le nom « Clark Kent », en veillant à l'écrire exactement comme dans le programme, avec un C et un K majuscules. Cette fois, le programme reconnaît que vous êtes Superman en personne (**Figure 5-6**).

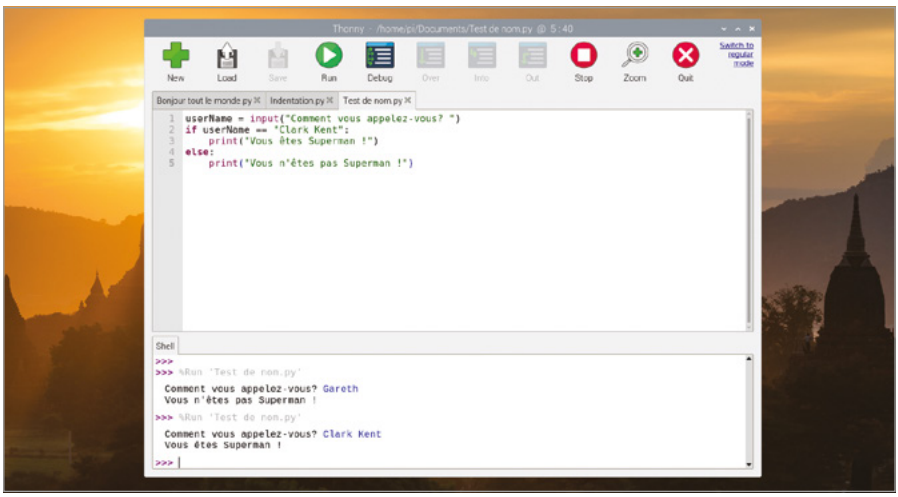


Figure 5-6 : Pourquoi n'êtes-vous pas en train de sauver la planète ?

Les symboles **==** indiquent à Python qu'il doit effectuer une comparaison directe, en cherchant à savoir si la variable **userName** correspond au texte (connu sous le nom de *chaîne*) dans votre programme. Si vous travaillez avec des chiffres, vous pouvez effectuer d'autres comparaisons : **>** pour déterminer si un nombre est supérieur à un autre, **<** pour déterminer s'il est inférieur, **=>** pour définir si un nombre est égal ou supérieur à, **=<** pour définir s'il est égal ou inférieur à. Le symbole **!=**, en revanche, signifie différent de, donc l'exact opposé de **==**. Ces symboles sont techniquement connus sous le nom d'*opérateurs de comparaison*.

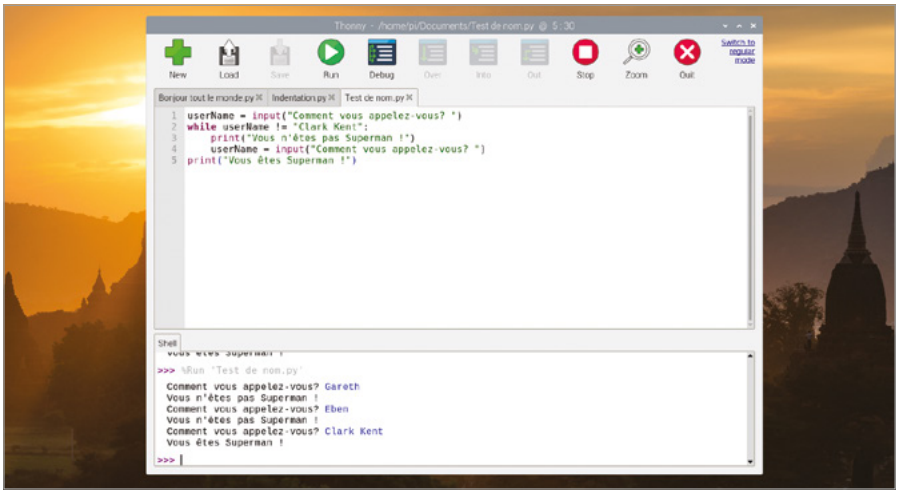
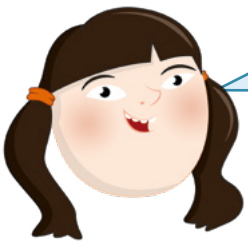


Figure 5-7 : Il vous demandera votre nom jusqu'à ce que vous disiez « Clark Kent »



UTILISATION DE = ET DE ==

En ce qui concerne l'utilisation des variables, il est essentiel d'apprendre la différence entre = et ==. Pour rappel, = signifie « rendre cette variable égale à cette valeur », alors que == signifie « vérifier si la variable est égale à cette valeur ». Les confondre, c'est se retrouver avec un programme qui ne fonctionne pas sur les bras !

Des opérateurs de comparaison peuvent également être utilisés dans les boucles. Supprimez les lignes 2 à 5, puis saisissez ce qui suit à la place :

```
while userName != "Clark Kent":
    print("Vous n'êtes pas Superman : réessayez !")
    userName = input ("Comment vous appelez-vous ?")
print("Vous êtes Superman !")
```

Cliquez à nouveau sur l'icône Run. Cette fois, au lieu de s'interrompre, le programme vous demandera votre nom jusqu'à confirmer que vous êtes bien Superman (**Figure 5-7**), un peu comme un mot de passe très simple. Pour sortir de la boucle, saisissez « Clark Kent » ou cliquez sur l'icône Stop de la barre d'outils de Thonny. Félicitations : vous savez maintenant utiliser les conditions et les opérateurs de comparaison !

DÉFI : AJOUTER DES QUESTIONS

Pouvez-vous modifier le programme afin de poser plus d'une question, en stockant les réponses dans plusieurs variables ? Pouvez-vous créer un programme qui utilise des conditions et des opérateurs de comparaison qui signalent, lorsqu'un utilisateur saisit un nombre, si celui-ci est supérieur ou inférieur à 5, comme le programme que vous avez créé dans le **Chapitre 4, Programmation avec Scratch** ?

Projet 1 : Flocons de neige Turtle (Tortue)

Maintenant que vous comprenez comment fonctionne Python, il est temps de jouer avec les graphiques et de créer un flocon de neige à l'aide d'un outil appelé *Turtle*.

PROJET EN LIGNE

Ce projet est également disponible en ligne en cliquant sur **rpf.io/turtle-snowflakes**

À l'origine, les turtles sont des robots dont la forme rappelle celle de l'animal dont ils portent le nom, conçus pour se déplacer en ligne droite, se retourner et pour soulever et abaisser un crayon. Dans leur version numérique, cela signifie simplement commencer ou arrêter de dessiner une ligne pendant leur déplacement. Contrairement à d'autres langages, notamment Logo et ses nombreuses variantes, Python ne dispose pas d'un outil turtle intégré, mais il dispose d'une *bibliothèque* de code supplémentaire qui lui donne le pouvoir d'un turtle. Les bibliothèques sont des paquets de code qui ajoutent de nouvelles instructions afin d'étendre les capacités de Python, et qui sont introduites dans vos propres programmes à l'aide d'une commande importer.

Créez un nouveau programme en cliquant sur l'icône ✚ New et saisissez ce qui suit :

```
import turtle
```

Lorsque vous utilisez des instructions incluses dans une bibliothèque, vous devez utiliser le nom de la bibliothèque suivi d'un point, puis le nom de l'instruction. Cela peut être ennuyeux à taper à chaque fois, alors vous pouvez attribuer un nom de variable plus court à la place : une seule lettre suffit, mais nous avons pensé que ce serait bien que la variable porte le petit nom de la tortue. Saisissez donc :

```
pat = turtle.Turtle()
```

Pour tester votre programme, vous devrez donner à votre tortue quelque chose à faire. Saisissez :

```
pat.forward(100)
```

Cliquez sur l'icône Run et enregistrez votre programme sous le nom **Flocons de neige Turtle**. Une fois le programme enregistré, une nouvelle fenêtre appelée « Turtle Graphics » s'ouvrira et vous verrez le résultat de votre programme : votre tortue, Pat, avance de 100 unités sur une ligne droite (**Figure 5-8**).

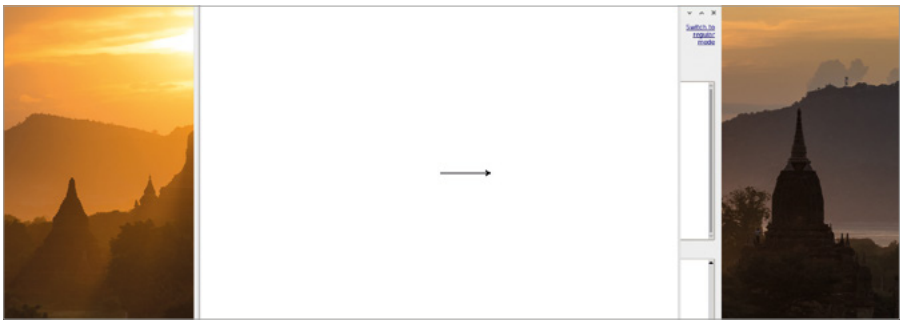


Figure 5-8 : La tortue avance en ligne droite

Revenez à la fenêtre principale de Thonny : si elle est dissimulée derrière la fenêtre Turtle Graphics, cliquez sur le bouton minimiser de la fenêtre Turtle Graphics, ou cliquez sur l'entrée Thonny dans la barre des tâches en haut de l'écran et cliquez sur le bouton Stop pour fermer la fenêtre Turtle Graphics.

Il serait fastidieux de taper à la main chaque instruction de mouvement, alors supprimez la ligne 3 et créez une boucle pour effectuer cette tâche difficile de création de formes :

```
for i in range(2):
    pat.forward(100)
    pat.right(60)
    pat.forward(100)
    pat.right(120)
```

Exécutez votre programme, et Pat dessinera un seul parallélogramme (**Figure 5-9**).

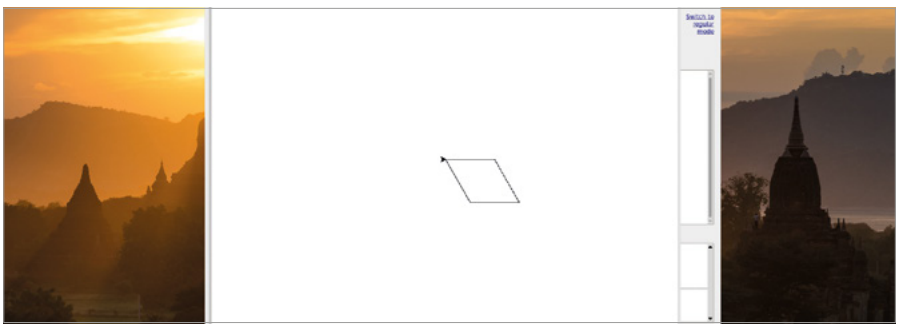


Figure 5-9 : En combinant des virages et des mouvements, vous pouvez dessiner des formes

Pour transformer votre forme géométrique en flocon de neige, cliquez sur l'icône Stop dans la fenêtre principale de Thonny et créez une boucle autour de votre boucle en ajoutant la ligne suivante comme ligne 3 :

```
for i in range(10):
```

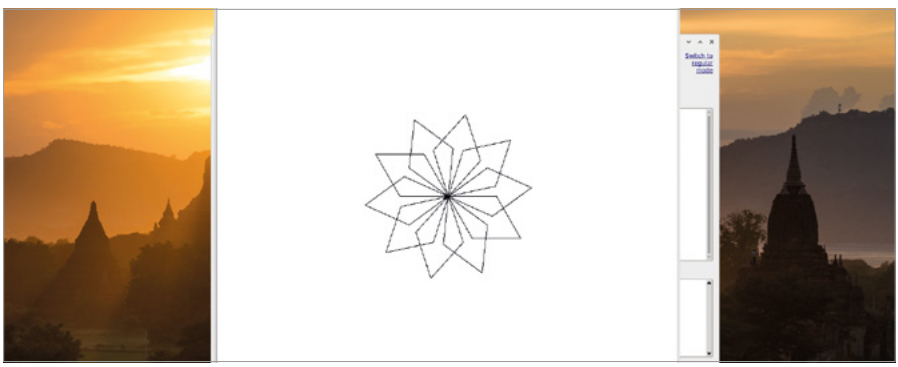
...et ce qui suit au bas de votre programme :

```
    pat.right(36)
```

Votre programme ne fonctionnera pas tel quel, car la boucle existante n'est pas correctement indentée. Pour y remédier, cliquez sur le début de chaque ligne de la boucle existante (lignes 4 à 8) et appuyez sur la touche **ESPACE** pour corriger l'indentation. Votre programme devrait maintenant ressembler à ceci :

```
import turtle
pat = turtle.Turtle()
for i in range(10):
    for i in range(2):
        pat.forward(100)
        pat.right(60)
        pat.forward(100)
        pat.right(120)
    pat.right(36)
```

Cliquez sur l'icône Run et observez la tortue : elle dessinera un parallélogramme, comme auparavant, mais quand elle aura terminé, elle tournera de 36 degrés et en dessinera un autre, puis un autre encore, et ainsi de suite jusqu'à ce qu'il y ait dix parallélogrammes qui se chevauchent sur l'écran, rappelant un flocon de neige (**Figure 5-10**).



▲ **Figure 5-10 :** Répéter une forme pour en faire une plus complexe

Alors que les dessins d'une tortue robotisée ne comportent qu'une seule couleur sur un grand morceau de papier, la tortue simulée de Python dispose de toute une palette de couleurs. Ajoutez une nouvelle ligne 3 et 4, en poussant les lignes existantes vers le bas :

```
turtle.Screen().bgcolor("blue")
pat.color("cyan")
```

Exécutez une nouvelle fois votre programme et vous verrez l'effet de votre nouveau code : la couleur de fond de la fenêtre Turtle Graphics est passée au bleu, et le flocon de neige est maintenant cyan (**Figure 5-11**).

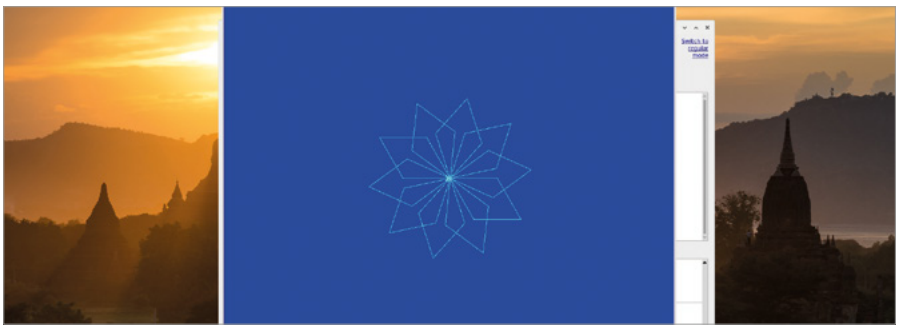


Figure 5-11 : Changement de la couleur de fond et des flocons de neige

Vous pouvez également faire choisir des couleurs au hasard dans une liste, en utilisant la bibliothèque `random`. Retournez au début de votre programme et insérez ce qui suit comme ligne 2 :

```
import random
```

Changez la couleur de fond de ce qui est maintenant la ligne 4 de « blue » à « grey », puis créez une nouvelle variable appelée « couleurs » en insérant une nouvelle ligne 5 :

```
couleurs = ["cyan", "purple", "white", "blue"]
```

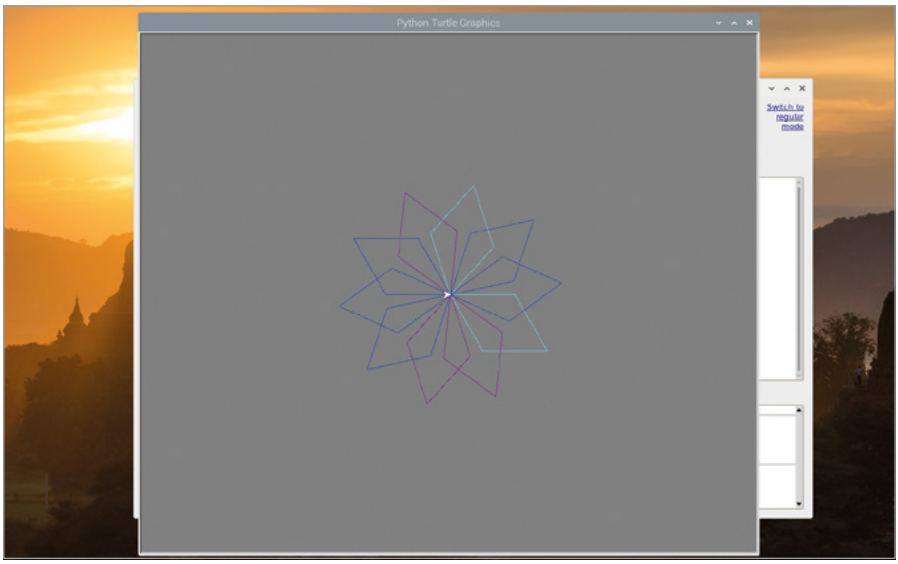
ORTHOGRAPHE AMÉRICAINE

De nombreux langages de programmation sont basés sur l'orthographe de l'anglais américain et Python ne fait pas exception : la commande de changement de couleur du crayon de la tortue s'écrit *color* et si vous l'épelez en suivant l'orthographe britannique comme *colour*, votre programme ne fonctionnera pas. Pour les variables, en revanche, vous pouvez choisir l'orthographe que vous souhaitez : vous pouvez donc appeler votre nouvelle variable *couleurs*, et Python comprendra.

Ce type de variable est connu sous le nom de liste, et il est toujours entre crochets. Dans ce cas, la liste est constituée de possibilités de couleurs pour les segments du flocon de neige, mais vous devez tout de même demander à Python d'en choisir une à chaque fois que la boucle se répète. À la fin du programme, saisissez ce qui suit, en vous assurant que la ligne est indentée de quatre espaces pour qu'elle fasse partie de la boucle extérieure, tout comme la ligne au-dessus :

```
    pat.color(random.choice(couleurs))
```

Cliquez sur l'icône Run pour dessiner encore une fois le flocon de neige-étoile Ninja. Cette fois, Python choisira cependant une couleur aléatoire de votre liste au moment de dessiner chaque pétale, ce qui donnera au flocon de neige une touche agréable et multicolore (**Figure 5-12**).



▲ **Figure 5-12 :** Utilisation de couleurs aléatoires pour les « pétales »

Pour que le flocon de neige ressemble plus à un vrai flocon de neige qu'à une étoile ninja, ajoutez une nouvelle ligne 6, directement sous votre liste **`couleurs`** et saisissez ce qui suit :

```
pat.penup()
pat.forward(90)
pat.left(45)
pat.pendown()
```

En présence d'un vrai robot de tortue, les instructions **`penup`** et **`pendown`** lui indiquent de poser le crayon sur le papier puis de le lever, mais dans le monde virtuel elles donnent à votre tortue l'ordre d'arrêter ou de commencer à dessiner des lignes. Mais cette fois, plutôt que

d'utiliser une boucle, vous allez créer une *fonction*, c'est-à-dire un segment de code que vous pouvez rappeler à tout moment, comme pour créer votre propre instruction Python.

Commencez par supprimer le code pour dessiner vos flocons de neige à partir d'un parallélogramme, c'est-à-dire tout ce qui se trouve entre l'instruction **`pat.color("cyan")`** sur la ligne 10 jusqu'à **`pat.right(36)`** sur la ligne 17. Laissez l'instruction **`pat.color(random.choice(couleurs))`** mais ajoutez un dièse (**#**) au début de la ligne. Ce faisant, vous allez *commenter* une instruction, que Python va ignorer. Les commentaires sont utiles pour ajouter des explications à votre code, ce qui le rendra beaucoup plus facile à comprendre lorsque vous y reviendrez quelques mois plus tard ou que vous l'envoyez à quelqu'un d'autre !

Créez votre fonction, qui sera appelée branche, en saisissant l'instruction suivante sur la ligne 10, comme suit : **`pat.pendown():`**

```
def branche():
```

Cette formulation *définit* votre fonction, **`branche`**. Lorsque vous appuyez sur la touche **ENTRÉE**, Thonny ajoutera automatiquement une indentation pour les instructions de la fonction. Saisissez ce qui suit, en faisant bien attention à l'indentation : à un moment donné, vous allez imbriquer votre code sur trois niveaux d'indentation !

```
    for i in range(3):
        for i in range(3):
            pat.forward(30)
            pat.backward(30)
            pat.right(45)
        pat.left(90)
        pat.backward(30)
        pat.left(45)
    pat.right(90)
    pat.forward(90)
```

Enfin, créez une nouvelle boucle au bas de votre programme (mais au-dessus de la ligne de couleur commentée) pour exécuter, ou *appeler* votre nouvelle fonction :

```
for i in range(8):
    branche()
    pat.left(45)
```

Une fois terminé, votre programme se présente comme suit :

```
import turtle
import random
pat = turtle.Turtle()
turtle.Screen().bgcolor("grey")
couleurs = ["cyan", "purple", "white", "blue"]
pat.penup()
pat.forward(90)
pat.left(45)
pat.pendown()
def branche():
    for i in range(3):
        for i in range(3):
            pat.forward(30)
            pat.backward(30)
            pat.right(45)
        pat.left(90)
        pat.backward(30)
        pat.left(45)
    pat.right(90)
    pat.forward(90)
for i in range(8):
    branche()
    pat.left(45)
#    pat.color(random.choice(couleurs))
```

Cliquez sur Run et observez Pat qui dessine dans la fenêtre graphique en suivant vos instructions. Félicitations : votre flocon de neige ressemble beaucoup plus à un vrai flocon de neige (**Figure 5-13**) !

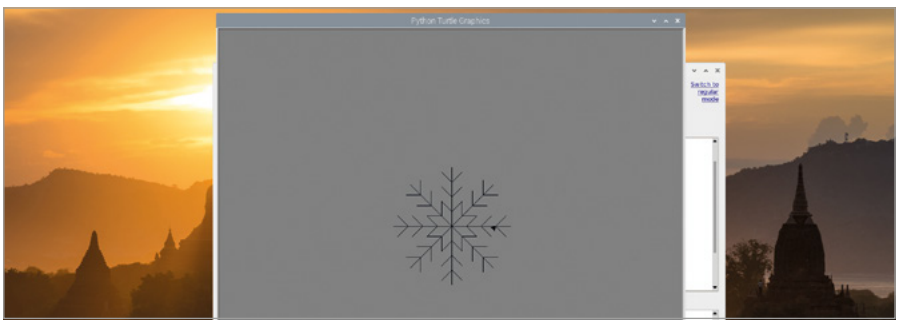


Figure 5-13 : Des branches supplémentaires lui donnent l'apparence d'un flocon de neige

DÉFI : QUOI ENCORE ?

Pouvez-vous utiliser votre instruction commentée pour que les branches du flocon de neige soient dessinées en différentes couleurs ? Pouvez-vous créer une fonction « flocon de neige » et l'utiliser pour dessiner de nombreux flocons de neige à l'écran ? Pouvez-vous faire en sorte que votre programme change la taille et la couleur des flocons de manière aléatoire ?

Projet 2 : Trouvez les différences en frissonnant

Python prend en charge des sons et des images, en plus des graphiques basés sur les turtles, qui peuvent être utilisés pour faire des farces à effet à vos amis : un jeu consistant à trouver des différences dissimulant un secret ténébreux, idéal pour Halloween !

PROJET EN LIGNE

Ce projet est également disponible en ligne en cliquant sur **rpf.io/scary-spot**

Pour ce projet, vous avez besoin de deux images : votre image servant à identifier les différences et une image « effrayante », plus un fichier audio. Cliquez sur l'icône en forme de framboise pour charger le menu de Raspberry Pi OS, sélectionnez la catégorie Internet et cliquez sur le navigateur Web Chromium. Lorsque le navigateur est chargé, saisissez **rpf.io/spot-pic** dans la barre d'adresse, puis appuyez sur la touche **ENTRÉE**. Cliquez avec le bouton droit de la souris et cliquez sur Enregistrer l'image sous..., sélectionnez le dossier Accueil dans la liste sur la gauche puis cliquez sur Enregistrer. Cliquez à nouveau sur la barre d'adresse de Chromium, puis saisissez **rpf.io/scary-pic** et appuyez sur la touche **ENTRÉE**. Cliquez avec le bouton droit de la souris et cliquez sur Enregistrer l'image sous..., sélectionnez le dossier Accueil dans la liste sur la gauche puis cliquez sur Enregistrer.

Pour le fichier audio, cliquez sur la barre d'adresse et saisissez **rpf.io/scream** puis appuyez sur la touche **ENTRÉE**. Ce fichier, le son d'un cri qui terrorisera les joueurs, sera lu automatiquement mais vous devrez le sauvegarder avant de pouvoir l'utiliser. Cliquez avec le bouton droit de la souris sur le petit lecteur audio, cliquez sur Enregistrer sous..., sélectionnez le dossier Accueil et cliquez sur Enregistrer. Vous pouvez maintenant fermer la fenêtre Chromium.

Cliquez sur l'icône New dans la barre d'outils Thonny pour commencer un nouveau projet. Comme auparavant, vous allez avoir recours à une bibliothèque pour étendre les capacités de Python : la bibliothèque Pygame, qui, comme son nom l'indique, a été conçue en pensant à des jeux. Saisissez donc :

```
import pygame
```

Vous aurez besoin de certaines parties d'autres bibliothèques, ainsi que d'une sous-section de la bibliothèque de Pygame. Pour les importer, saisissez :

```
from pygame.locals import *
from time import sleep
from random import randrange
```

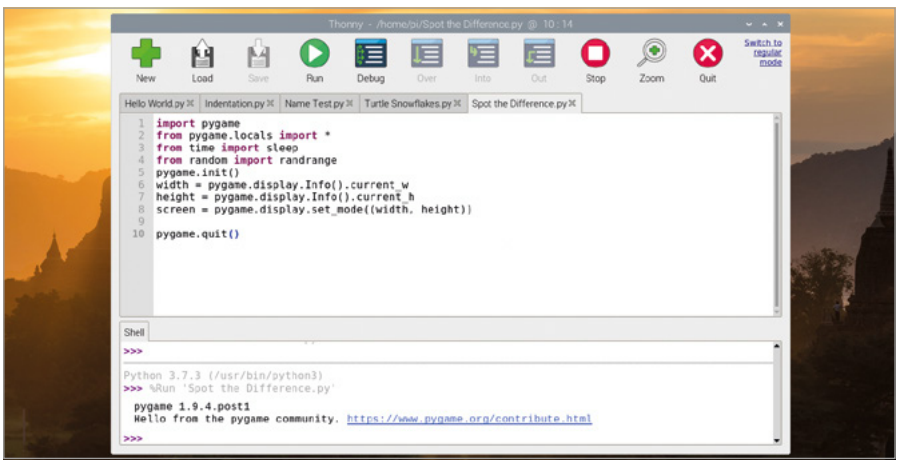
Les instructions **`from`** fonctionnent différemment des instructions **`import`** et vous permettent d'importer uniquement les parties d'une bibliothèque qui vous intéressent plutôt que la bibliothèque entière. Ensuite, vous devez mettre en place Pygame via un processus dit d'*initialisation*. Pygame doit connaître la largeur et la hauteur du moniteur ou de l'écran TV du joueur, c'est-à-dire sa *résolution*. Saisissez donc :

```
pygame.init()
width = pygame.display.Info().current_w
height = pygame.display.Info().current_h
```

La dernière étape de la mise en place de Pygame consiste à créer sa fenêtre, ou écran selon le terme utilisé par Pygame. Saisissez donc :

```
screen = pygame.display.set_mode((width, height))

pygame.quit()
```

Observez la ligne blanche au milieu ; c'est là que vous allez placer votre programme. Pour l'instant, cependant, cliquez sur l'icône Run, enregistrez votre programme sous **Trouvez la différence** et observez : Pygame crée une fenêtre sur fond noir, qui disparaîtra ensuite presque immédiatement dès qu'il atteindra l'instruction d'arrêter. À part un court message dans le shell (**Figure 5-14**), le programme n'a pas obtenu de résultats spectaculaires jusqu'à présent.



▲ **Figure 5-14 :** Votre programme est fonctionnel, mais ne fait pas encore grand-chose

Pour afficher votre image « Trouvez la différence », saisissez la ligne suivante dans l'espace au-dessus de **`pygame.quit()`** :

```
difference = pygame.image.load('spot_the_diff.png')
```

Pour que l'image remplisse l'écran, vous devez l'adapter à la résolution de votre moniteur ou de votre téléviseur. Saisissez donc :

```
difference = pygame.transform.scale(difference, (width, height))
```

L'image est maintenant en mémoire et vous devez dire à Pygame de l'afficher effectivement à l'écran, en suivant unprocessus connu sous le nom de *bit blit*, ou *transfert de blocs de bits*. Saisissez donc :

```
screen.blit(difference, (0, 0))
pygame.display.update()
```

La première de ces lignes copie l'image sur l'écran, en commençant par le coin supérieur gauche ; la seconde indique à Pygame qu'il faut redessiner l'écran. Sans cette deuxième ligne, l'image serait à la bonne place dans la mémoire mais vous ne pourriez pas la voir !

Cliquez sur l'icône Run et l'image apparaîtra rapidement à l'écran (**Figure 5-15**).

Figure 5-15 : Votre image « Trouvez la différence »

Pour voir l'image plus longtemps, ajoutez la ligne suivante juste au-dessus de **`pygame.quit()`** :

```
sleep(3)
```

Cliquez à nouveau sur Run, et l'image restera à l'écran plus longtemps. Ajoutez votre image surprise en saisissant ce qui suit juste en dessous de la ligne **`pygame.display.update():`**

```
zombie = pygame.image.load('scary_face.png')
zombie = pygame.transform.scale(zombie, (width, height))
```

Définissez un délai, pour que l'image du zombie n'apparaisse pas tout de suite :

```
sleep(3)
```

Ensuite, vous pouvez faire apparaître l'image à l'écran et mettre à jour pour qu'elle soit visible au joueur :

```
screen.blit(zombie, (0,0))
pygame.display.update()
```

Cliquez à nouveau sur l'icône Run et observez ce qui se passe : Pygame chargera votre image « Trouvez la différence », mais, lorsque trois secondes se seront écoulées, celle-ci sera remplacée par l'effrayant zombie (**Figure 5-16**) !

Figure 5-16 : Voilà de quoi en terroriser plus d'un

Cependant, l'effet est un peu mitigé si le délai est fixé à trois secondes. Modifiez la ligne **`sleep(3)`** au-dessus de **`screen.blit(zombie, (0,0))`** comme suit :

```
sleep(randrange(5, 15))
```

Ce faisant, un nombre aléatoire sera sélectionné entre 5 et 15, à savoir le nombre de secondes qui constituent le délai. Ensuite, ajoutez la ligne suivante juste au-dessus de votre instruction **`sleep`** pour charger le fichier audio du cri :

```
scream = pygame.mixer.Sound('scream.wav')
```

Déplacez-vous en dessous de votre instruction **`sleep`** et saisissez ce qui suit sur une nouvelle ligne pour lancer la lecture du son, de sorte qu'il se déclenche juste avant que l'image effrayante ne soit montrée au joueur :

```
scream.play()
```

Enfin, demandez à Pygame d'arrêter de jouer le son en saisissant la ligne suivante juste au-dessus de **`pygame.quit()`**:

```
scream.stop()
```

Cliquez sur l'icône Run et admirez votre œuvre : après quelques secondes de plaisir innocent passées à chercher la différence, un cri qui vous glace le sang se fera entendre et votre zombie effrayant s'affichera, de quoi terrifier vos amis ! Si l'image du zombie apparaît avant le son, vous pouvez ajuster en ajoutant un petit délai juste après votre instruction **`scream.play()`** et avant votre instruction **`screen.blit`** :

```
sleep(0.4)
```

Une fois terminé, votre programme se présente comme suit :

```
import pygame
from pygame.locals import *
from time import sleep
from random import randrange
pygame.init()
width = pygame.display.Info().current_w
height = pygame.display.Info().current_h
screen = pygame.display.set_mode((width, height))
difference = pygame.image.load('spot_the_diff.png')
difference = pygame.transform.scale(difference, (width, height))
screen.blit(difference, (0, 0))
pygame.display.update()
zombie = pygame.image.load('scary_face.png')
zombie = pygame.transform.scale (zombie, (width, height))
scream = pygame.mixer.Sound('scream.wav')
sleep(randrange(5, 15))
scream.play()
screen.blit(zombie, (0,0))
pygame.display.update()
sleep(3)
scream.stop()
pygame.quit()
```

Il ne reste plus qu'à inviter vos amis à trouver la différence, en veillant bien entendu à ce que les haut-parleurs soient allumés !

DÉFI : CHANGER D'APPARENCE

Pouvez-vous modifier les images pour que la farce soit plus adaptée à d'autres événements, par exemple Noël ? Pouvez-vous dessiner vos propres images « trouvez la différence » ainsi que l'image effrayante (en utilisant un éditeur graphique tel que GIMP) ? Pourriez-vous détecter lorsque l'utilisateur clique sur une différence, pour rendre le jeu plus convaincant ?

Projet 3 : Le labyrinthe RPG

Maintenant que vous maîtrisez Python, il est temps d'utiliser Pygame pour compliquer un peu les choses : un jeu de labyrinthe parfaitement fonctionnel en texte intégral, basé sur des jeux de rôle classiques. Connus sous le nom d'aventures textuelles ou de fiction interactive, ces jeux remontent à l'époque où les ordinateurs ne pouvaient pas prendre en charge les graphiques, mais leurs fans soutiennent toujours qu'aucun graphique ne sera jamais aussi vivant que ceux que vous avez dans votre imagination !

PROJET EN LIGNE

Ce projet est également disponible en ligne en cliquant sur **rpf.io/python-rpg**

Ce programme est un peu plus complexe que les autres de ce chapitre, donc pour faciliter les choses, vous commencerez avec une version déjà partiellement écrite. Ouvrez le navigateur Web Chromium et accédez à l'adresse suivante : **rpf.io/p/fr-FR/rpg-go**.

Le navigateur Web Chromium téléchargera automatiquement le code du programme dans votre dossier Téléchargements, mais vous avertira que le type de fichier (un programme Python) pourrait nuire à votre ordinateur. Comme vous avez téléchargé le fichier de la Raspberry Pi Foundation, une source de confiance, cliquez sur le bouton « Conserver » du message d'avertissement qui apparaît en bas de l'écran. Retournez à Thonny, puis cliquez sur l'icône « Load » . Recherchez le fichier, **rpg.py** dans votre dossier de téléchargements et cliquez sur le bouton Load.

Commencez par cliquer sur l'icône Run pour vous familiariser avec le fonctionnement d'une aventure textuelle. La sortie du jeu apparaît dans la zone du shell en bas de la fenêtre de Thonny. Agrandissez la fenêtre de Thonny en cliquant sur le bouton maximiser pour la rendre plus lisible.

Tel qu'il se présente actuellement, le jeu est très simple : il consiste en deux pièces et aucun objet. Le joueur commence dans l'entrée, la première des deux salles. Pour se rendre à la cuisine, il suffit de taper « go south » et d'appuyer sur la touche **ENTRÉE** (**Figure 5-17**). Lorsque vous êtes dans la cuisine, vous pouvez saisir « go north » pour retourner dans le hall d'entrée. Vous pouvez également essayer de saisir « go west » et « go east », mais comme il n'y a pas de pièces dans ces directions, le jeu vous renverra un message d'erreur.

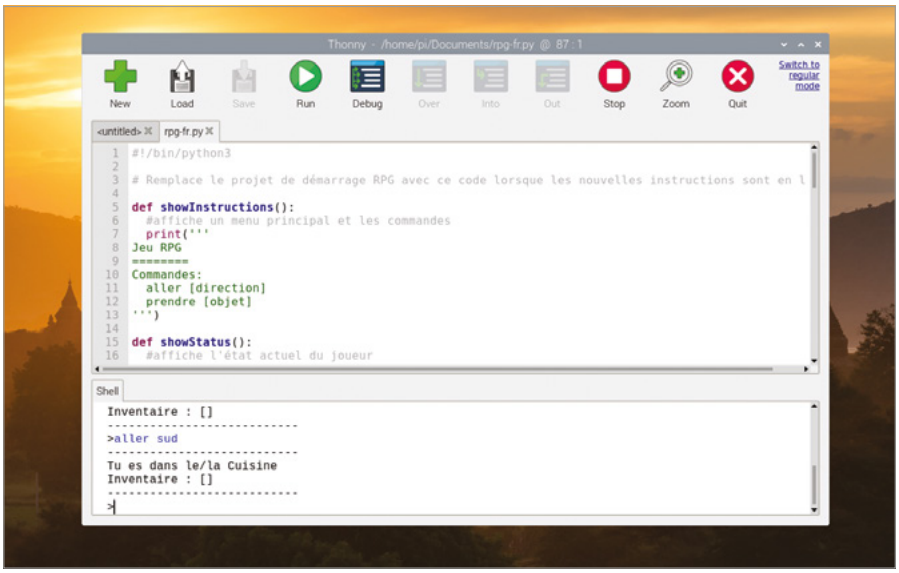


Figure 5-17 : Il n'y a que deux pièces pour l'instant

Faites défiler jusqu'à la ligne 29 du programme dans la zone de script pour trouver une variable appelée **rooms**. Ce type de variable est connu sous le nom de *dictionnaire* et indique au jeu quelles sont les pièces, leurs sorties et la pièce vers laquelle mène une sortie donnée.

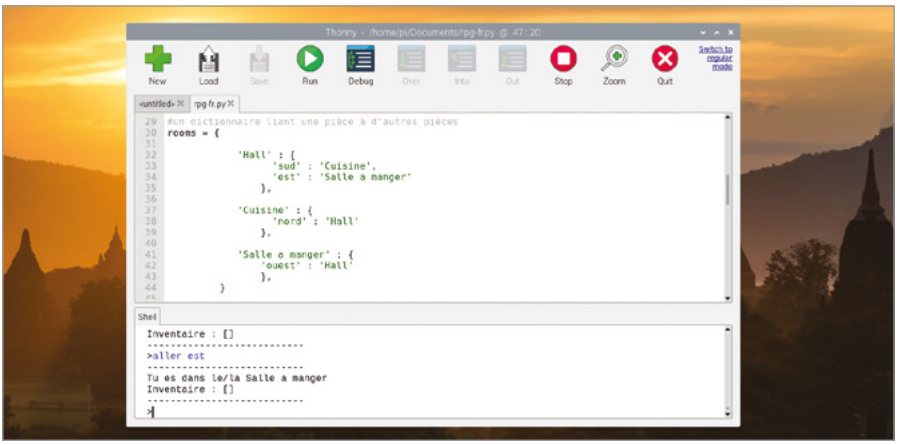
Pour rendre le jeu plus intéressant, ajoutez une autre pièce : une salle à manger, à l'est de l'entrée. Cherchez la variable **rooms** dans la zone de script, et étendez-la en ajoutant un symbole de virgule (**,**) après **}** à la ligne 38, en saisissant ensuite ce qui suit (l'indentation exacte n'est pas indispensable dans un dictionnaire) :

```
'Salle a manger' : {
      'ouest' : 'Hall'
   }
```

Vous aurez également besoin d'une nouvelle sortie dans le hall, car le programme ne crée pas automatiquement les portes de sortie. Rendez-vous à la fin de la ligne 33, ajoutez une virgule, puis ajoutez la ligne suivante :

```
'est' : 'Salle a manger'
```

Cliquez sur l'icône Run et essayez de vous rendre dans votre nouvelle pièce : tapez « aller est » dans le hall pour vous rendre dans la salle à manger (**Figure 5-18**), puis « aller ouest » dans la salle à manger pour revenir dans le hall. Félicitations : vous avez créé une nouvelle pièce !

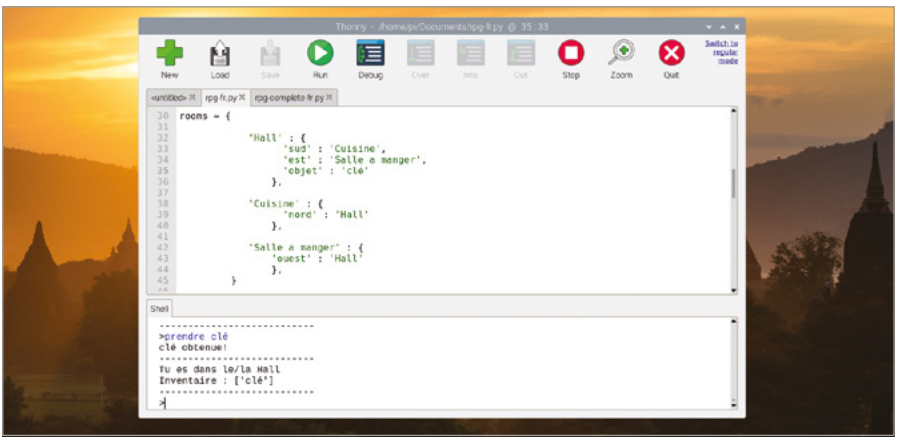


▲ **Figure 5-18 :** Vous avez ajouté une nouvelle pièce

Mais les pièces vides ne sont pas très intéressantes. Pour ajouter un élément à une pièce, vous devez modifier le dictionnaire de cette pièce. Arrêtez le programme en cliquant sur l'icône Stop. Recherchez le dictionnaire **`Hall`** dans la zone des scripts, puis ajoutez une virgule à la fin de la ligne **`'est' : 'Salle a manger'`** avant d'appuyer sur **ENTRÉE** et de saisir la ligne suivante :

```
'objet' : 'clé'
```

Cliquez à nouveau sur Run. Cette fois, le jeu vous dira que vous pouvez voir votre nouvel objet : une clé. Tapez « prendre clé » (**Figure 5-19**) pour la récupérer, en l'ajoutant à la liste des articles que vous transportez, appelée votre *inventaire*. Votre inventaire vous accompagne dans vos déplacements d'une pièce à l'autre.



▲ **Figure 5-19 :** La clé collectée est ajoutée à votre inventaire

Cliquez sur l'icône Stop et rendez le jeu plus intéressant en ajoutant un monstre à éviter. Cherchez le dictionnaire **Cuisine** et ajoutez un élément « monstre » de la même manière que vous avez ajouté l'élément « clé », sans oublier d'ajouter une virgule à la fin de la ligne ci-dessus :

```
'objet' : 'monstre'
```

Pour que le monstre puisse attaquer le joueur, vous devez ajouter de la logique au jeu. Faites défiler le programme jusqu'en bas dans la zone de script et ajoutez les lignes suivantes (y compris le commentaire, marqué d'un symbole dièse, qui vous aidera à comprendre le programme si vous y revenez ultérieurement) en veillant à bien les indenter :

```
    # le joueur perd s'il rentre dans une pièce où se cache le
monstre
    if 'objet' in rooms[currentRoom] and 'monstre' in
rooms[currentRoom]['objet']:
        print("Un monstre t'a eu... TU AS PERDU !")
        break
```

Cliquez sur Run, et essayez d'aller dans la cuisine (**Figure 5-20**) : c'est le monstre qui va être content en vous voyant !

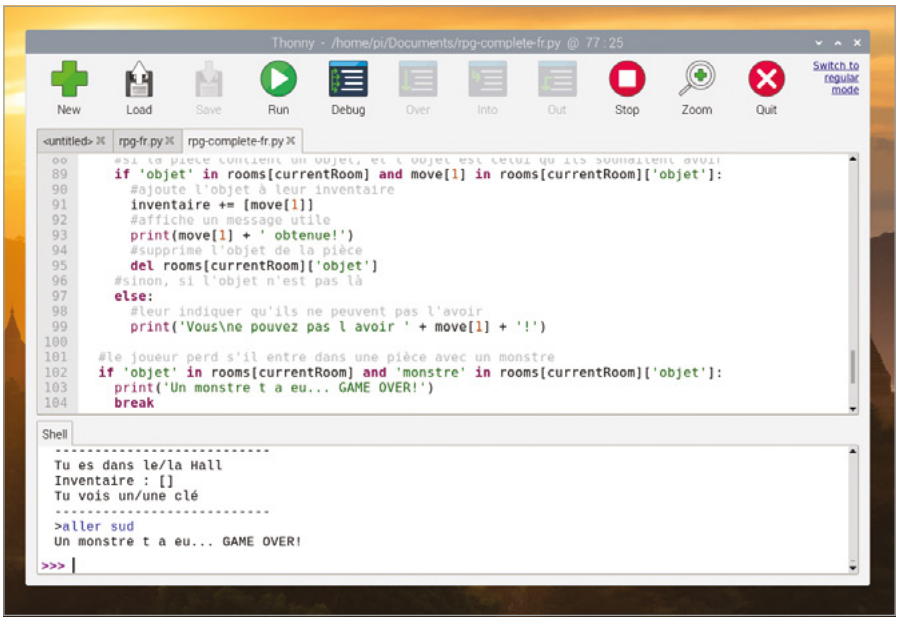


Figure 5-20 : Oubliez les rats, il y a un monstre dans la cuisine

Pour transformer cette aventure en un véritable jeu, vous aurez besoin de plus d'objets, d'une autre pièce et de la possibilité de « gagner » en quittant la maison en ayant mis en sécurité tous les objets dans votre inventaire. Commencez par ajouter une autre pièce, comme vous l'avez fait pour la salle à manger : cette fois, un jardin. Ajoutez une porte de sortie du dictionnaire de la salle à manger, en n'oubliant pas d'ajouter une virgule à la fin de la ligne au-dessus :

```
        'sud' : 'Jardin'
```

Ensuite, ajoutez votre nouvelle pièce au dictionnaire principal **`rooms`**, en n'oubliant pas d'ajouter une virgule après le **`}`** sur la ligne au-dessus, exactement comme avant :

```
    'Jardin' : {
          'nord' : 'Salle a manger'
        }
```

Ajoutez un objet « potion » au dictionnaire de la salle à manger, en n'oubliant pas d'ajouter la virgule nécessaire à la ligne au-dessus :

```
        'objet' : 'potion'
```

Enfin, faites défiler le programme jusqu'en bas et ajoutez la logique nécessaire pour vérifier si le joueur possède tous les objets et, si c'est le cas, lui indiquer qu'il a gagné la partie :

```
   # le joueur gagne s'il atteint le jardin avec une clé et la
potion
      if currentRoom == 'Jardin' and 'clé' in inventaire and
'potion' in inventaire:
          print("Tu l'as échappée belle... TU AS GAGNÉ !")
          break
```

Cliquez sur Run, et essayez de finir le jeu en récupérant la clé et la potion avant d'aller au jardin. N'oubliez pas de ne pas entrer dans la cuisine, car c'est là que se cache le monstre !

Pour terminer de définir le jeu, ajoutez quelques instructions indiquant au joueur comment terminer. Faites défiler l'écran jusqu'en haut du programme, où la fonction **`showInstructions()`** est définie, et ajoutez ce qui suit :

```
  Trouvez la clé et la potion et allez dans le jardin
  Attention aux monstres !
```

Exécutez le jeu une dernière fois, et vous verrez apparaître vos nouvelles instructions tout au début (**Figure 5-21**). Félicitations : vous avez créé un jeu textuel interactif de labyrinthe !

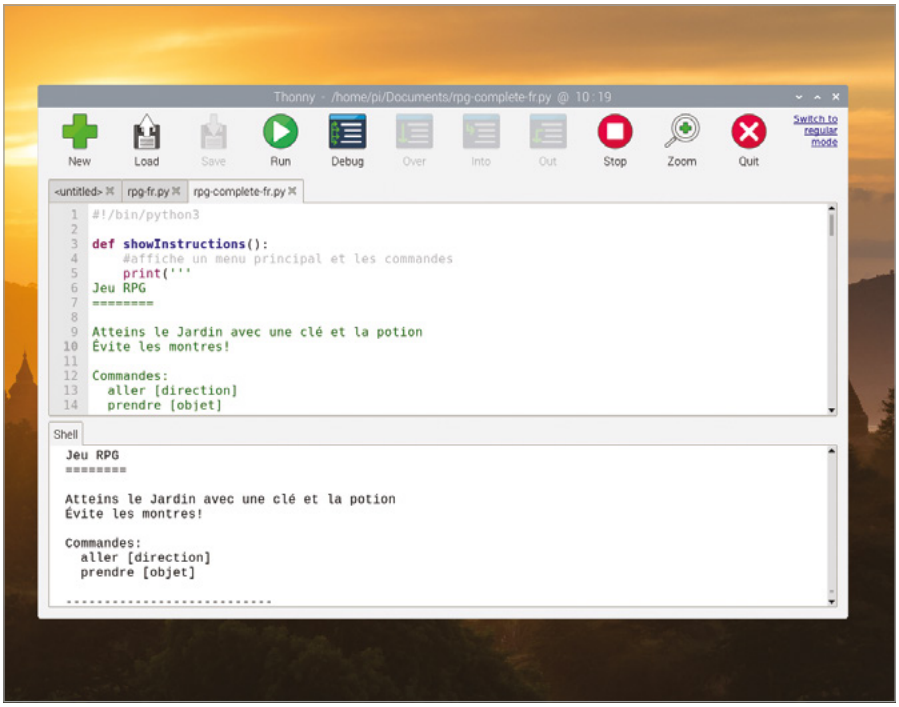


Figure 5-21 : Le joueur sait maintenant ce qu'il doit faire

DÉFI : ÉTOFFER LE JEU

Pouvez-vous ajouter des pièces supplémentaires pour que le jeu dure plus longtemps ? Pouvez-vous ajouter un objet pour vous protéger du monstre ? Comment ajouteriez-vous une arme pour tuer le monstre ? Pouvez-vous ajouter des pièces qui se trouvent au-dessus et en dessous des pièces existantes, accessibles par des escaliers ?

Chapitre 6

L'informatique physique avec Scratch et Python

Maîtriser le codage, ce n'est pas uniquement réaliser des projets à l'écran : vous pourrez également contrôler tous les composants électroniques rattachés aux broches du port GPIO de votre Raspberry Pi

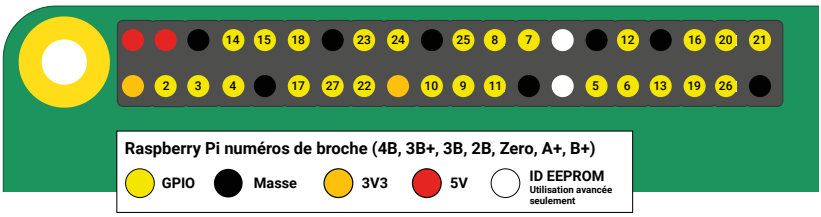
Normalement, quand on parle de « programmation » ou de « codage », on pense naturellement à des logiciels. Mais le codage va bien au-delà des logiciels : il peut rentrer dans la vraie vie grâce à la maîtrise d'éléments matériels. C'est ce qu'il est convenu d'appeler l'*informatique physique*.

Comme son nom l'indique, l'informatique physique consiste à contrôler des objets dans le monde réel grâce à des programmes : du matériel, donc, plutôt que des logiciels. Lorsque vous réglez le programme de votre machine à laver, que vous changez la température de votre thermostat programmable ou que vous appuyez sur un bouton aux feux de circulation pour traverser la route en toute sécurité, vous utilisez l'informatique physique.

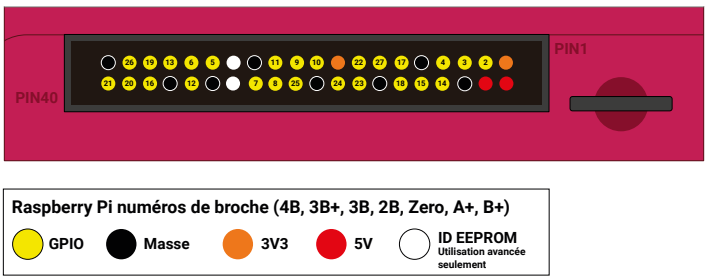
Votre Raspberry Pi est un excellent outil pour apprendre l'informatique physique grâce à une de ses fonctions clés : le port *general purpose input/output* (*GPIO*).

Présentation du port GPIO

Situé sur le bord supérieur de la carte du Raspberry Pi, ou à l'arrière dans le cas du Raspberry Pi 400, et consistant en deux longues rangées de broches métalliques, le port GPIO (general-purpose input/output) vous permet de connecter des éléments matériels comme des LED et des interrupteurs au Raspberry Pi pour commander les programmes que vous créez. Les broches peuvent être utilisées aussi bien en entrée qu'en sortie.



Le port (ou connecteur) GPIO est constitué de 40 broches mâles. Certaines broches sont disponibles pour vos projets d'informatique physique, d'autres sont consacrées à l'alimentation, d'autres encore sont réservées pour communiquer à l'aide d'éléments complémentaires comme la Sense HAT (consulter le **Chapitre 7**).



Le port GPIO du modèle Raspberry Pi 400 dispose exactement des mêmes broches, mais il est installé à l'envers par rapport aux autres modèles de Raspberry Pi. Le présent schéma suppose que vous faites face au port GPIO depuis l'arrière de votre Raspberry Pi 400. Vérifiez toujours attentivement votre câblage lorsque vous connectez quoi que ce soit au port GPIO de votre Raspberry Pi 400 (il est très facile d'oublier, malgré les étiquettes « Pin 40 » et « Pin 1 » sur le boîtier !)

EXTENSIONS GPIO

Il est parfaitement possible d'utiliser le connecteur GPIO du Raspberry Pi 400 tel quel, mais une extension pourrait s'avérer extrêmement utile. À l'aide d'une extension, les broches sont ramenées sur le côté de votre Raspberry Pi 400, ce qui signifie que vous pouvez vérifier et ajuster votre câblage sans avoir à passer par l'arrière.

Les extensions compatibles comprennent la gamme Black HAT Hack3r de **pimoroni.com** et la Pi T-Cobbler Plus de **adafruit.com**.

Si vous achetez une extension, vérifiez toujours son câblage : certaines, comme la Pi T-Cobbler Plus, modifient la disposition des broches GPIO. En cas de doute, suivez toujours les instructions du fabricant.

Il existe plusieurs catégories de types de broches, chacune ayant une fonction particulière :

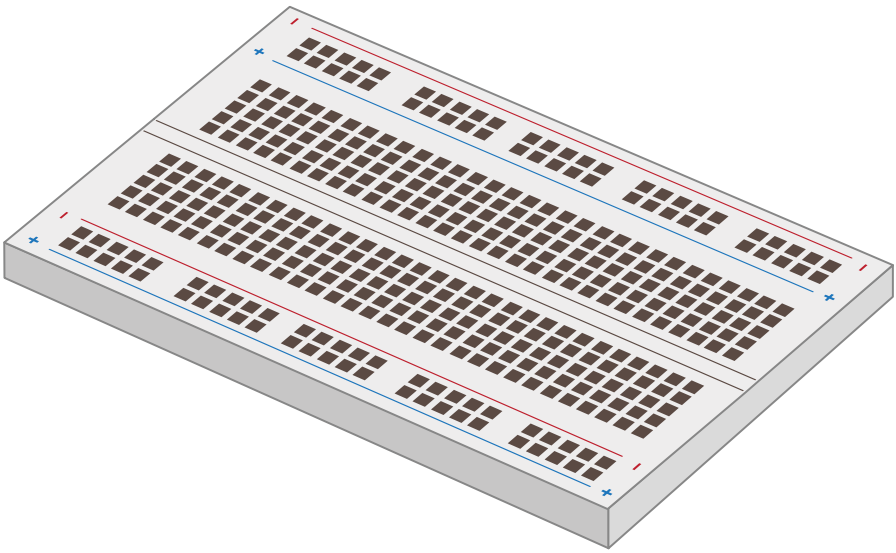
3V3	**Alimentation 3,3 volts**	Une source d'alimentation permanente de 3,3 V, la même tension alimentant votre Raspberry Pi en interne
5V	**Alimentation 5 volts**	Une source d'alimentation permanente de 5 V, la même tension alimentant votre Raspberry Pi via le connecteur micro-USB
Masse (GND)	**Masse 0 volts**	Une prise de masse, utilisée pour compléter un circuit relié à une source d'énergie
GPIO XX	**Numéro de broche d'entrée/sortie à usage général « XX »**	Les broches GPIO disponibles pour vos programmes, identifiées par un numéro compris entre 2 et 27
ID EEPROM	**Broches réservées à des fins particulières**	Broches réservées à Hardware Attached on Top (HAT) et autres accessoires

ATTENTION !

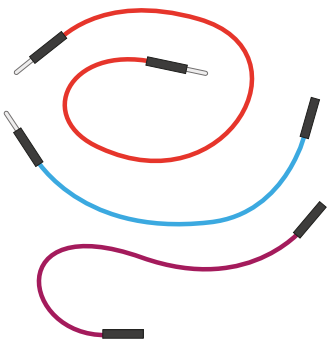
Le connecteur GPIO du Raspberry Pi est un moyen amusant et sûr d'expérimenter l'informatique physique, mais il doit être traité avec précaution. Faites attention à ne pas plier les broches en connectant et déconnectant du matériel. Ne connectez jamais, ni accidentellement ni délibérément, deux broches directement ensemble, sauf si cela est expressément indiqué dans les instructions d'un projet : cela risque de provoquer un court-circuit en fonction des broches susceptible d'endommager de façon permanente votre Raspberry Pi.

Composants électroniques

Le port GPIO n'est que l'une des parties dont vous aurez besoin pour explorer le domaine de l'informatique physique, l'autre partie étant constituée de composants électroniques, les dispositifs que vous pourrez contrôler à partir du port GPIO. Il existe des milliers de composants, mais la plupart des projets GPIO sont réalisés à partir des parties communes ci-après.

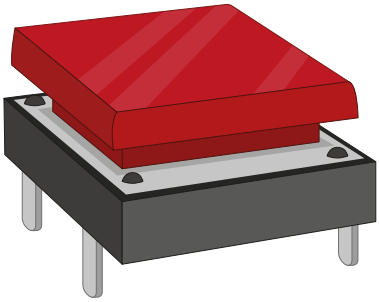


Une *platine*, ou *platine d'expérimentation*, peut faciliter considérablement tous vos projets d'informatique physique. Plutôt que de travailler avec différents composants séparés qui doivent être reliés par des fils, une platine vous permet d'insérer des composants et de les connecter par des circuits métalliques cachés sous la surface. De nombreuses platines comportent également des sections pour la distribution de l'alimentation, ce qui facilite la mise en place de vos circuits. Une platine n'est pas indispensable pour vous lancer dans l'informatique physique, mais elle vous sera certainement très utile.

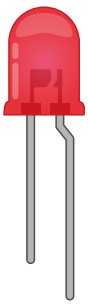


Les fils de raccordement, ou *cordons de raccordement*, servent à connecter les différents composants à votre Raspberry Pi et, si vous n'utilisez pas de platine, les uns aux autres. Il en existe trois types : mâle-femelle (M2F), qui servent à connecter la platine aux broches GPIO ; femelle-femelle (F2F), nécessaires à connecter des composants individuels entre eux si vous n'utilisez pas de platine ; et mâle-mâle (M2M), utilisés pour effectuer des connexions des différentes parties de la platine entre elles. En fonction de votre projet, vous pourriez avoir besoin des trois types de fils de raccordement ; si vous utilisez une platine, des fils de raccordement M2F et M2M devraient suffire.

Un *interrupteur à bouton-poussoir*, ou *interrupteur instantané*, est le type d'interrupteur utilisé pour commander une console de jeux. Disponible en deux ou quatre bornes (les deux types fonctionnent avec le Raspberry Pi), le bouton-poussoir est un dispositif d'entrée, qui vous permet de transmettre des ordres pour que votre dispositif effectue une tâche. Un autre type d'interrupteur est le *commutateur* : alors qu'un bouton-poussoir n'est actif que lorsque vous le maintenez enfoncé, un commutateur (par exemple un interrupteur d'éclairage) s'active en l'appuyant une fois, puis reste actif jusqu'à ce que vous l'appuyiez à nouveau.

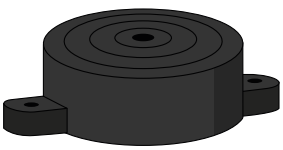


Une *diode électroluminescente*(*LED*) est un *périphérique de sortie* que vous contrôlez directement à partir de votre programme. Une LED s'éclaire quand elle est allumée, et vous en trouverez partout dans votre maison, des petites qui vous indiquent que vous avez laissé votre machine à laver allumée, aux grandes qui éclairent vos chambres. Les LED sont disponibles dans une large gamme de formes, de couleurs et de tailles, mais toutes ne sont pas compatibles avec le Raspberry Pi : évitez les LED conçues pour une alimentation de 5 ou 12 V.



Les résistances sont des éléments qui contrôlent le flux de *courant électrique* et sont disponibles en différentes valeurs mesurées à l'aide d'une unité de mesure appelée *ohm*(Ω). Plus le nombre d'ohms est élevé, plus la résistance est importante. Pour les projets d'informatique physique Raspberry Pi, elles sont surtout utiles à protéger les LED contre une consommation de courant trop importante et contre les dommages qu'elles pourraient causer à elles-mêmes ou à votre Raspberry Pi ; pour cela, vous aurez besoin de résistances d'environ 330 Ω, bien que de nombreux fournisseurs d'électricité vendent des packs pratiques contenant des résistances de valeurs différentes pour permettre une flexibilité accrue.

Le *buzzer piézoélectrique*, généralement dénommé buzzer ou avertisseur, est un autre dispositif de sortie. Alors qu'une LED produit de la lumière, un buzzer produit un son, ou plutôt un bourdonnement. Le boîtier en plastique du buzzer contient une paire de plaques métalliques qui, lorsqu'elles sont activées, vibrent l'une contre l'autre pour produire le bourdonnement. Il existe deux types de buzzer : les *buzzers actifs* et les *buzzers passifs*. Procurez-vous un buzzer actif, car ce sont les plus simples à utiliser.



Parmi les autres composants électriques courants, citons les moteurs, qui nécessitent une carte de contrôle spéciale avant de pouvoir être connectés au Raspberry Pi, les capteurs infrarouges qui détectent les mouvements, les capteurs de température et d'humidité qui peuvent être utilisés pour prévoir le temps, et les photorésistances (LDR), des dispositifs d'entrée qui fonctionnent comme des LED inversées en détectant la lumière.

Les composants d'informatique physique compatibles avec les Raspberry Pi sont vendus dans le monde entier, soit séparément soit en tant que kits qui vous dotent de tout le nécessaire pour démarrer. Certains des détaillants les plus populaires sont :

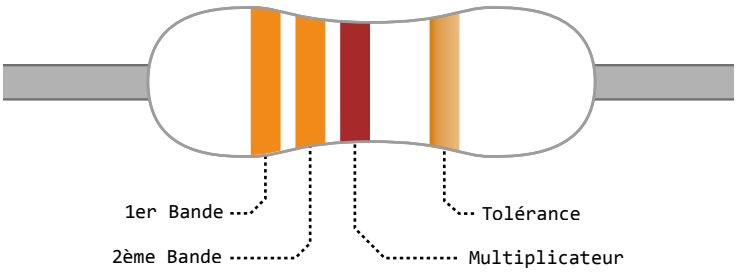
- **RS Components** – uk.rs-online.com
- **CPC** - cpc.farnell.com
- **Pimoroni** - pimoroni.com
- **Pi Hut** - thepihut.com
- **ModMyPi** - modmypi.com
- **PiSupply** - uk.pi-supply.com
- **Adafruit** - adafruit.com

Pour réaliser les projets de ce chapitre, vous devez au minimum posséder :

- 3 LED : rouge, verte, et jaune ou ambrée
- 2 interrupteurs à bouton-poussoir
- 1 buzzer actif
- des fils de raccordement mâle-femelle (M2F) et femelle-femelle (F2F)
- en option, une platine et des fils de connexion mâle-mâle (M2M)

Lecture des codes de couleur des résistances

Les résistances existent dans un large éventail de valeurs, des versions à résistance nulle qui ne sont en fait que des morceaux de fil aux versions à haute résistance de la taille de votre jambe. Cependant, très peu de ces résistances indiquent leurs valeurs en chiffres : elles utilisent plutôt un code spécial sous forme de bandes colorées autour du corps de la résistance.



	1er/2ème Bande	Multiplicateur	Tolérance
Noir	0	$\times 10^{0}$	-
Brun	1	$\times 10^{1}$	±1%
Rouge	2	$\times 10^{2}$	±2%
Orange	3	$\times 10^{3}$	-
Jaune	4	$\times 10^{4}$	-
Vert	5	$\times 10^{5}$	±0.5%
Bleu	6	$\times 10^{6}$	±0.25%
Violet	7	$\times 10^{7}$	±0.1%
Gris	8	$\times 10^{8}$	±0.05%
Blanc	9	$\times 10^{9}$	-
Or	-	$\times 10^{-1}$	±5%
Argent	-	$\times 10^{-2}$	±10%
Aucune	-	-	±20%

Pour comprendre la valeur d'une résistance, placez-la de sorte que les bandes groupées se trouvent sur la gauche et la bande isolée sur la droite. En partant de la première bande, cherchez sa couleur dans la colonne « 1ère/2ème bande » du tableau pour obtenir le premier et le deuxième chiffre. Dans l'exemple, cette résistance comporte deux bandes oranges, représentant chacune une valeur de 3 pour un total de 33. Si votre résistance comporte quatre bandes groupées au lieu de trois, notez également la valeur de la troisième bande (pour les résistances à cinq/six bandes, voir **rpf.io/5-6band**).

Passez ensuite à la dernière bande groupée (la troisième ou la quatrième) et observez sa couleur dans la colonne « Multiplier ». Cette colonne vous indique le nombre par lequel vous devez multiplier votre nombre pour obtenir la valeur réelle de la résistance. Cet exemple comporte une bande brune, qui signifie « $\times 10^{1}$ ». Cela peut prêter à confusion, mais il s'agit tout simplement d'une *notation scientifique* : « $\times 10^{1}$ » signifie « ajoutez un zéro à la fin de votre nombre ». Si la bande était

bleue, correspondant à $\times 10^6$, la colonne vous indiquerait « ajouter six zéros derrière votre nombre ».

Le nombre 33, qui nous est indiqué par les bandes oranges, plus le zéro ajouté d'après la bande brune nous donnent une valeur de 330, à savoir la valeur de la résistance, mesurée en ohms. La dernière bande sur la droite représente la *tolérance* de la résistance. Cette valeur indique la probabilité à laquelle le chiffre obtenu correspond à sa valeur nominale. Les résistances d'entrée de gamme peuvent comporter une bande argentée, indiquant que leur valeur peut être 10 % plus ou moins élevée que leur valeur nominale, ou ne pas comporter de dernière bande, indiquant que leur valeur réelle peut être 20 % plus ou moins élevée ; les résistances haut-de-gamme comportent une bande grise, indiquant qu'elles se situent à plus ou moins 0,05 % de leur valeur nominale. Pour les projets d'amateurs, la précision n'est pas essentielle : toute tolérance fonctionnera généralement très bien.

Si la valeur de votre résistance dépasse 1 000 ohms (1 000 Ω), elle est généralement exprimée en kilohms (kΩ) ; si elle dépasse un million d'ohms, il s'agit de mégohms (MΩ). Une résistance 2 200 Ω s'exprime donc comme 2,2 kΩ ; une résistance de 2 200 000 Ω s'écrirait comme 2,2 MΩ.

AVEZ-VOUS LA RÉPONSE ?

De quelle couleur sont les bandes d'une résistance 100 Ω ?
De quelle couleur sont les bandes d'une résistance 2,2 MΩ ?
Si vous cherchez des résistances d'entrée de gamme, quelle couleur de bande représentant la tolérance rechercheriez-vous ?

Votre tout premier programme d'informatique physique : Hello, LED !

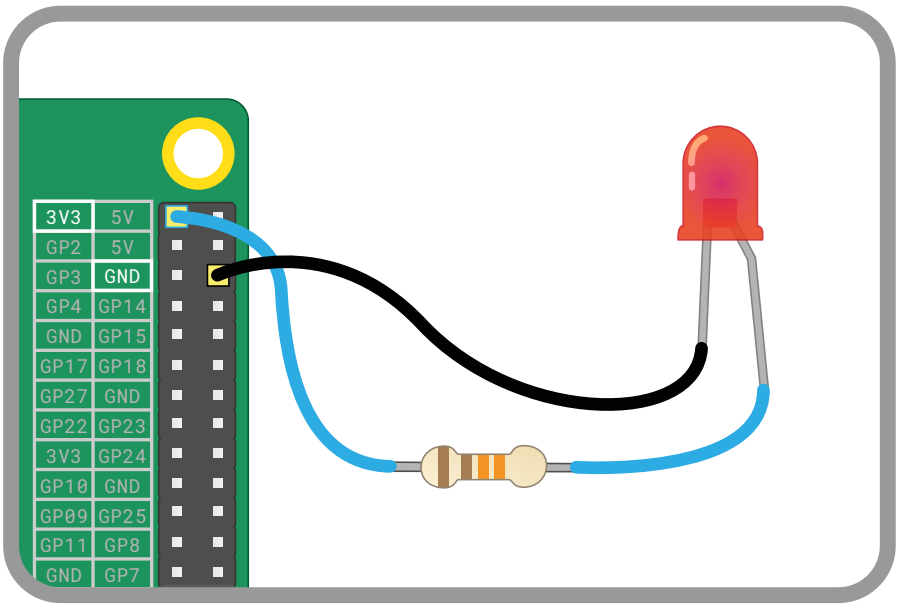
Si parvenir à afficher « Hello, World » à l'écran était une première étape fantastique dans l'apprentissage d'un langage de programmation, le fait d'arriver à allumer un voyant LED constitue le premier pas d'usage dans l'apprentissage de l'informatique physique. Pour ce projet, vous aurez besoin d'une LED et d'une résistance de 330 ohms (330 Ω), ou aussi près que possible de 330 Ω, ainsi que de fils de connexion femelle-femelle (F2F).

LA RÉSISTANCE EST ESSENTIELLE

La résistance est un composant essentiel de ce circuit : elle protège vote Raspberry Pi et la LED en limitant la quantité de courant électrique que la LED peut utiliser. En absence de résistance, la LED risque d'utiliser trop de courant et de brûler (ou de brûler le Raspberry Pi). Lorsqu'elle est utilisée de cette manière, la résistance est désignée par le terme de *résistance de limitation de courant*. La valeur exacte de la résistance dont vous avez besoin dépend de la LED que vous utilisez, mais une résistance de 330 Ω fonctionne pour la plupart des LED courantes. Plus la valeur est élevée, plus la LED s'assombrit ; plus la valeur est faible, plus la LED est brillante.

Ne connectez jamais une LED à un Raspberry Pi sans résistance de limitation de courant, à moins d'avoir la certitude que la LED possède une résistance intégrée de valeur appropriée.

Commencez par vérifier que votre LED fonctionne. Tournez votre Raspberry Pi de manière à ce que le port GPIO se situe sur la droite en deux bandes verticales. Connectez une extrémité de votre résistance de 330 Ω à la première broche 3,3 V (étiquetée 3V3 sur la **Figure 6-1**) à l'aide d'un fil de liaison femelle-femelle, puis connectez l'autre extrémité à la patte plus longue (positif, ou anode) de votre LED à l'aide de l'autre fil de connexion femelle-femelle. Prenez un dernier fil de raccordement femelle-femelle et connectez la patte plus courte (négatif ou cathode) de votre LED à la première broche de masse (marquée GND sur la **Figure 6-1**).



▲ **Figure 6-1 :** Câblez votre LED à ces broches, sans oublier la résistance !

Si votre Raspberry Pi est sous tension, la LED devrait s'allumer. Si elle ne s'allume pas, vérifiez votre circuit : assurez-vous que vous n'avez pas utilisé une valeur de résistance trop élevée, que tous les fils sont correctement connectés et que vous avez bien choisi les bonnes broches GPIO telles qu'elles sont illustrées sur le schéma. Vérifiez également les pattes de la LED, car les LED ne fonctionnent que dans un sens : la patte la plus longue doit être connectée au pôle positif du circuit et la patte la plus courte au pôle négatif.

Une fois que votre LED fonctionne, il est temps de la programmer. Débranchez le fil de raccordement de la broche 3,3 V (étiquetée 3V3 sur la **Figure 6-2**) et connectez-le à la broche GPIO 25 (étiquetée GP25 sur la **Figure 6-2**). La LED s'éteint, ce qui est parfaitement normal, rassurez-vous.

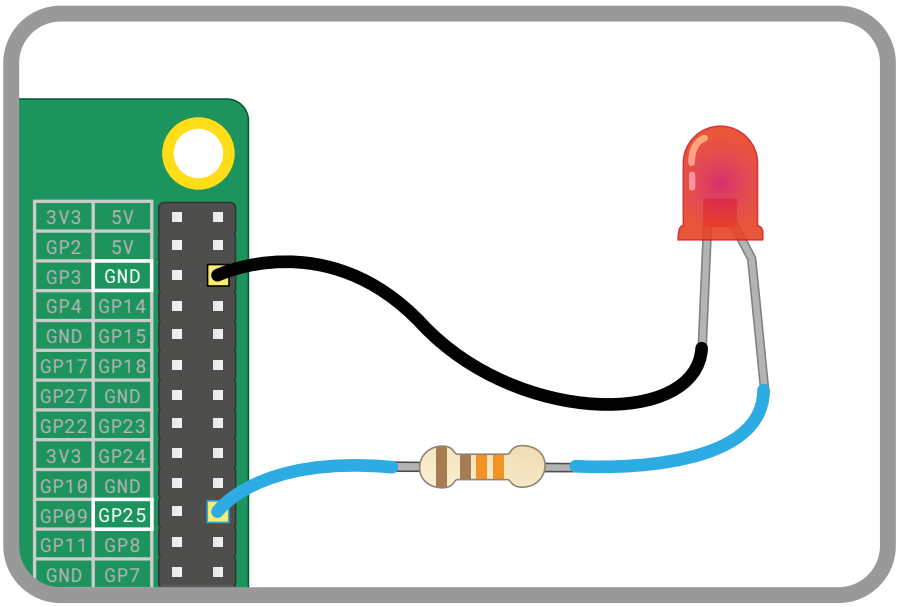


Figure 6-2 : Débranchez le fil de raccordement de la broche 3,3 V et connectez-le à la broche 25 du GPIO

Vous êtes maintenant prêt à créer un programme Scratch ou Python pour allumer et éteindre votre LED.

CONNAISSANCES EN MATIÈRE DE CODAGE

Les projets présentés dans ce chapitre supposent une certaine aisance d'utilisation de Scratch 3 et de l'environnement de développement intégré (IDE) Thonny Python. Si vous ne l'avez pas encore fait, lisez le **Chapitre 4, Programmation avec Scratch 3** ou le **Chapitre 5, Programmation avec Python** et réalisez d'abord les projets décrits dans ces chapitres.

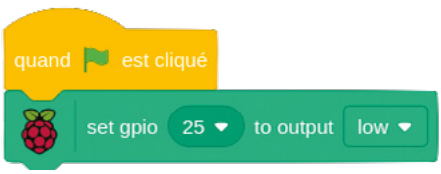
Commande d'une LED dans Scratch

Chargez Scratch 3 et cliquez sur l'icône Ajouter une extension . Faites défiler la page vers le bas pour rechercher l'extension « Raspberry Pi GPIO » (**Figure 6-3**), puis cliquez dessus. Ce faisant, vous chargez les blocs dont vous avez besoin pour commander le port GPIO de votre Raspberry Pi depuis Scratch 3. Vous verrez les nouveaux blocs apparaître dans la palette de blocs ; lorsque vous en aurez besoin, ils seront disponibles dans la catégorie Raspberry Pi GPIO.

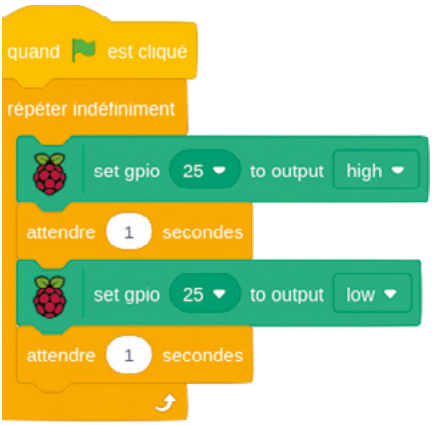
Figure 6-3 : Ajouter l'extension Raspberry Pi GPIO dans Scratch 3

Commencez par faire glisser un bloc d'événement quand est cliqué sur la zone de code, puis faites glisser un bloc set gpio to output high directement en dessous. Vous devez choisir le numéro de la broche à utiliser : cliquez sur la petite flèche pour ouvrir la sélection déroulante et cliquez sur « 25 » pour indiquer à Scratch que vous souhaitez contrôler la broche 25 du GPIO.

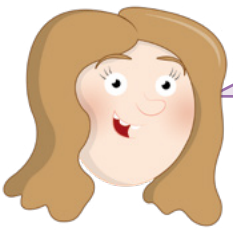
Cliquez sur le drapeau vert pour exécuter votre programme. Votre LED s'allume : vous avez programmé votre premier projet d'informatique physique ! Cliquez sur l'octogone rouge pour arrêter votre programme : vous remarquerez que votre LED reste allumée ! C'est parce que votre programme a commandé au Raspberry Pi uniquement d'allumer la LED : c'est ce que signifie la partie « output high » de votre bloc set gpio 25 to output high. Pour l'éteindre, cliquez sur la flèche vers le bas à l'extrémité du bloc et sélectionnez « low » dans la liste.



Cliquez à nouveau sur le drapeau vert et votre programme éteindra la LED. Pour rendre les choses plus intéressantes, ajoutez un bloc de contrôle répéter indéfiniment et quelques blocs attendre 1 seconde pour créer un programme permettant d'allumer et d'éteindre la LED chaque seconde.



Cliquez sur le drapeau vert et observez votre LED : elle s'allumera pendant une seconde, s'éteindra pendant une seconde, s'allumera pendant une seconde, et continuera à répéter cette séquence jusqu'à ce que vous cliquiez sur l'octogone rouge pour l'arrêter. Voyez ce qui se passe lorsque vous cliquez sur l'octogone alors que la LED est allumée ou éteinte.



DÉFI : POUVEZ-VOUS LE CHANGER ?

Comment modifieriez-vous le programme pour que la LED reste allumée plus longtemps ? Ou pour qu'elle reste éteinte plus longtemps ? Quel est le délai le plus court que vous pouvez utiliser tout en laissant la LED s'allumer et s'éteindre ?

Commande d'une LED dans Python

Chargez Thonny à partir de la section Programmation du menu framboise, puis cliquez sur le bouton New pour commencer un nouveau projet et sur Enregistrer pour l'enregistrer sous **Hello LED**. Pour utiliser les broches du GPIO à partir de Python, vous aurez besoin d'une bibliothèque appelée GPIO Zero. Pour ce projet, vous n'avez besoin que de la partie de la bibliothèque qui vous permet de travailler avec les LED. Importez uniquement cette section de la bibliothèque en saisissant ce qui suit dans la zone du shell Python :

```
from gpiozero import LED
```

Ensuite, vous devez indiquer à GPIO Zero à quelle broche du GPIO la LED est connectée. Saisissez donc :

```
led = LED(25)
```

Ensemble, ces deux lignes permettent à Python de contrôler les LED connectées aux broches du GPIO de Raspberry Pi et de lui indiquer quelle broche (ou quelles broches, si vous avez plus d'une LED dans votre circuit) commander. Pour commander effectivement les LED, saisissez :

```
led.on()
```

Pour éteindre à nouveau la LED, saisissez :

```
led.off()
```

Félicitations, vous contrôlez les broches GPIO de votre Raspberry Pi dans Python ! Essayez de saisir ces deux instructions à nouveau. Si la LED est déjà éteinte, **`led.off()`** n'aura aucun effet ; il en va de même si la LED est déjà allumée et que vous tapez **`led.on()`**.

Pour réaliser un véritable programme, saisissez ce qui suit dans la zone de script :

```
from gpiozero import LED
from time import sleep

led = LED(25)

while True:
    led.on()
    sleep(1)
    led.off()
    sleep(1)
```

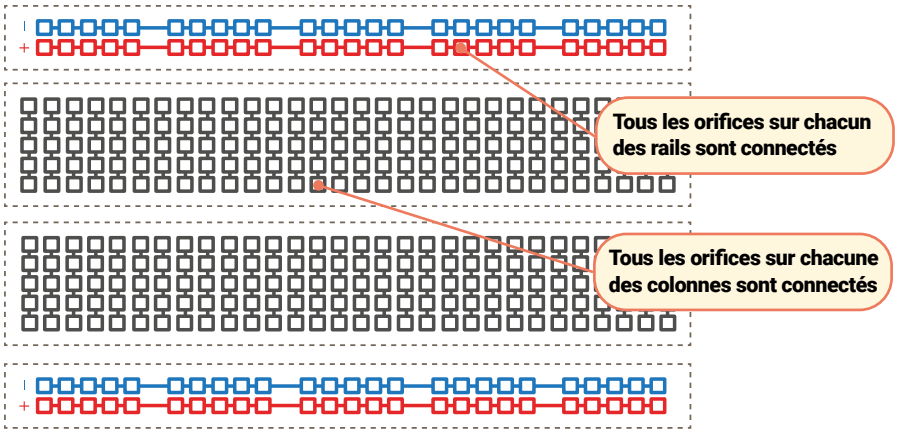
Ce programme importe la fonction LED de la bibliothèque **`gpiozero`** (GPIO Zero) et la fonction **`sleep`** de la librairie **`time`**, puis crée une boucle infinie pour allumer la LED pendant une seconde, l'éteindre pendant une seconde et ainsi de suite. Cliquez sur le bouton Exécuter pour le voir en action : votre LED se met à clignoter. Comme pour le programme Scratch, observez le comportement lorsque vous cliquez sur le bouton Stop alors que la LED est allumée, par rapport à la réaction lorsque la LED est éteinte.

DÉFI : UN ÉCLAIRAGE PLUS LONG

Comment modifieriez-vous le programme pour que la LED reste allumée plus longtemps ? Ou pour qu'elle reste éteinte plus longtemps ? Quel est le délai le plus court que vous pouvez utiliser tout en laissant la LED s'allumer et s'éteindre ?

Utilisation d'une platine

Les prochains projets présentés dans ce chapitre seront beaucoup plus faciles à réaliser si vous utilisez une platine pour rassembler tous les composants et effectuer les connexions électriques.



Une platine est composée d'orifices placés à une distance de 2,54 mm l'un de l'autre pour accueillir les différents composants. Au-dessous desdits orifices se trouvent des bandes de métal qui remplissent la même fonction que les fils de connexion que vous avez utilisés jusqu'à présent. Elles sont disposées en rangées et sur la plupart des platines un espace au milieu les sépare en deux moitiés. De nombreuses platines présentent des lettres en haut et des chiffres sur les côtés. Leur rôle est celui de vous permettre de repérer facilement un orifice en particulier : A1 est le coin supérieur gauche, B1 est l'orifice à la droite de celui-ci, tandis que B2 est un orifice plus bas. A1 est connecté à B1 par les bandes métalliques cachées, mais aucun orifice n'est connecté à deux autres orifices, sauf si vous utilisez vous-même un fil de connexion.

Les plus grandes platines disposent également de bandes trouées sur les côtés, généralement marquées de bandes rouges et noires ou rouges et bleues. Il s'agit des *rails d'alimentation* conçus pour faciliter le câblage : vous pouvez connecter un seul fil de la broche de masse du Raspberry Pi à l'un des rails d'alimentation (généralement marqué d'une bande bleue ou noire et d'un symbole moins) pour obtenir une *base commune* pour de nombreux composants sur la platine, et vous pouvez faire de même si votre circuit requiert une alimentation 3,3 V ou 5 V.

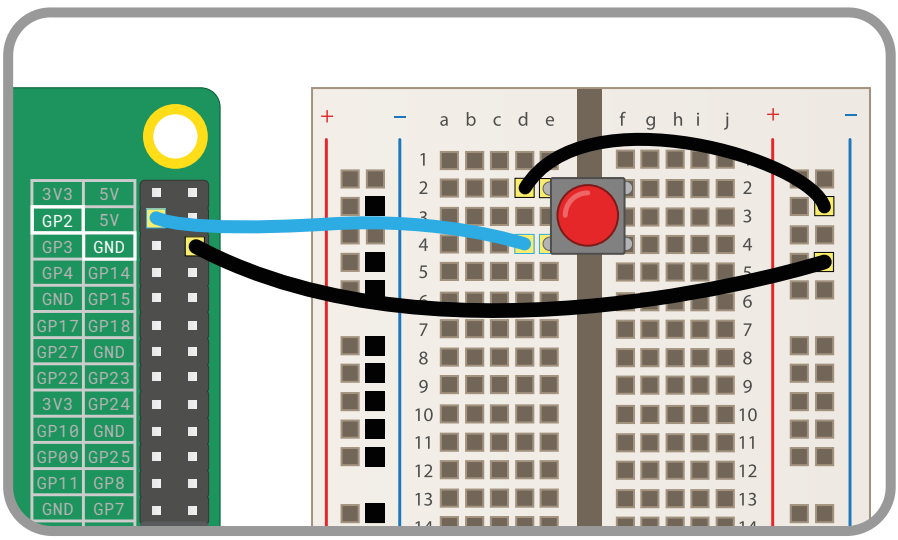
Il est très simple d'ajouter des composants électroniques à une platine : il suffit d'aligner leurs broches (les parties métalliques qui ressortent) avec les trous et de les insérer doucement jusqu'à ce que le composant soit en place. Pour les connexions qui sont nécessaires au-delà des connexions réalisées grâce à la platine, vous pouvez utiliser des fils de connexion mâle-mâle (M2M) ; pour les connexions entre la platine et le Raspberry Pi, utilisez des fils de connexion mâle-femelle (M2F).

N'essayez jamais de faire passer plus d'un fil de composant ou de raccordement dans le même orifice de la platine. N'oubliez pas : les orifices sont connectés en colonnes, à l'exception de la séparation au milieu, de sorte qu'un fil de composant dans A1 est électriquement connecté à tout ce que vous ajouterez à B1, C1, D1 et E1.

Prochaines étapes : connexion d'un bouton

Utiliser des éléments de sortie comme les LED est une chose, mais comme l'indique la partie « input/output (entrée/sortie) » de « GPIO », il est également possible d'utiliser des broches comme entrées. Pour ce projet, vous aurez besoin d'une platine, d'un câble de connexion mâle-mâle (M2M), d'une paire de fils de connexion mâle-femelle (M2F) et d'un interrupteur à bouton-poussoir. Si vous ne disposez pas d'une platine, vous pouvez utiliser des fils de connexion femelle-femelle (F2F), mais le risque d'interrompre accidentellement le circuit en appuyant sur le bouton sera beaucoup plus élevé.

Commencez par ajouter le bouton-poussoir à votre platine. Si votre bouton-poussoir n'a que deux bornes, assurez-vous qu'elles se trouvent dans des rangées de numéros différents sur la platine ; s'il a quatre bornes, tournez-le de manière à ce que les côtés où se trouvent les bornes soient alignés sur les rangées de la platine et que les côtés plats non pourvus de bornes se trouvent en haut et en bas. Connectez le pôle masse de votre platine à une broche de masse du Raspberry Pi (désignée par le sigle GND sur la **Figure 6-4**) via un câble mâle-femelle, puis connectez une borne de votre interrupteur à bouton poussoir au pôle masse à l'aide d'un fil mâle-mâle. Pour finir, connectez l'autre borne (qui se trouve du même côté que la borne que vous venez de connecter, si vous utilisez un interrupteur à quatre bornes) à la broche GPIO 2 (marquée GP2 sur la **Figure 6-4**) du Raspberry Pi à l'aide d'un fil de raccordement mâle-femelle.

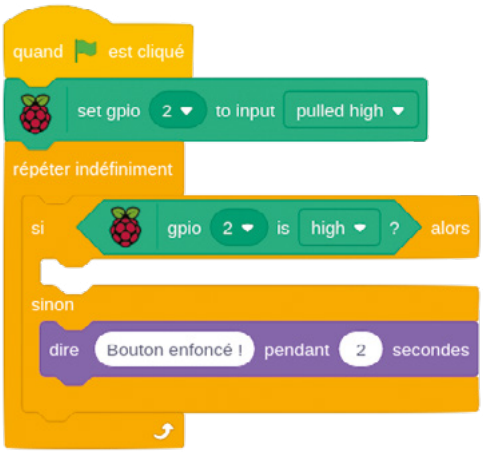


▲ **Figure 6-4 :** Connexion d'un bouton-poussoir aux broches du connecteur GPIO

Lecture d'un bouton dans Scratch

Lancez un nouveau programme Scratch et faites glisser un bloc quand est cliqué sur la zone de codage. Connectez un bloc set gpio to input pulled high et sélectionnez le chiffre 2 dans la liste déroulante pour qu'il corresponde à la broche GPIO utilisée pour le bouton-poussoir.

Si vous cliquez maintenant sur le drapeau vert, vous n'obtiendrez aucune réaction. C'est parce que vous avez indiqué à Scratch d'utiliser la broche comme entrée, sans spécifier d'action. Faites glisser un bloc répéter indéfiniment en bas de votre séquence, puis faites glisser un bloc si alors sinon à son intérieur. Cherchez le bloc gpio is high?, faites-le glisser dans l'espace en forme de losange à l'intérieur du bloc si alors et sélectionnez le chiffre 2 dans la liste déroulante afin de lui indiquer la broche GPIO à vérifier. Faites glisser un bloc dire Bonjour ! pendant 2 secondes dans la partie sinon du bloc et modifiez-le de sorte à lui faire dire « Bouton appuyé ! ». Laissez la partie « si alors » du bloc vide pour l'instant.



L'ensemble est relativement complexe, mais faites le test : cliquez sur le drapeau vert et appuyez sur le bouton sur votre platine. Votre sprite doit vous indiquer que le bouton a été appuyé : vous avez réussi à lire un message transmis par la broche GPIO !

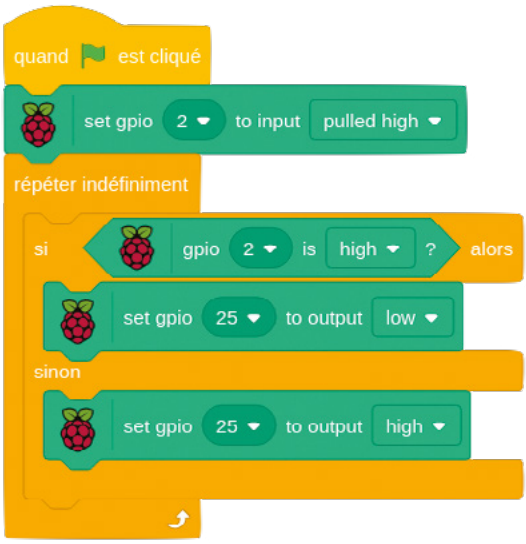
Vous avez peut-être remarqué que la partie si gpio 2 is high? alors du bloc est vide. Le code qui s'exécute lorsque le bouton est appuyé se trouve pour sa part dans la partie sinon du bloc. Cela vous semble confus, car on aurait tendance à croire qu'en appuyant sur le bouton le réglage High se déclenche ? En réalité, le contraire se produit : les broches GPIO du Raspberry Pi sont normalement réglées sur High, ou activé, lorsqu'elles sont réglées en entrée, et le fait d'appuyer sur le bouton modifie le réglage sur Low, ou désactivé.

Observez à nouveau votre circuit : le bouton est connecté à la broche GPIO 2, qui fournit la partie positive du circuit, et au pôle de masse. Lorsque le bouton est appuyé, la tension sur la broche GPIO est ramenée à zéro via le pôle de masse, et votre programme Scratch interrompt son exécution du code dans votre bloc si gpio 2 is high? alors et exécute à la place le code qui se trouve dans la partie sinon du bloc.

Si tout cela vous semble incompréhensible, souvenez-vous simplement de ceci : en appuyant sur un bouton connecté à une broche GPIO de votre Raspberry Pi, la broche passe à zéro, pas quand elle est activée !

Pour étendre encore votre programme, ajoutez la LED et la résistance dans le circuit : n'oubliez pas de connecter la résistance à la broche GPIO 25 et à la patte longue de la LED, et la patte plus courte de la LED au rail de masse de votre platine.

Faites glisser le bloc **dire Bouton appuyé ! pendant 2 secondes** de la zone de code vers la palette de blocs pour le supprimer, puis remplacez-le par un bloc **set gpio 25 to output high**, sans oublier que vous devrez modifier le numéro GPIO à l'aide de la flèche déroulante. Ajoutez un bloc **set gpio 25 to output low**, sans oublier de modifier les valeurs, dans la partie **si gpio 2 is high? alors** encore vide du bloc.



Cliquez sur le drapeau vert et appuyez sur le bouton. La LED s'allumera tant que vous maintiendrez le bouton enfoncé ; relâchez le bouton, et la LED s'éteint. Félicitations : vous contrôlez une broche GPIO à partir d'un message transmis par une autre broche !

DÉFI : POUR QUE LA LED RESTE ALLUMÉE

Comment modifieriez-vous le programme pour que la LED reste allumée pendant plusieurs secondes, même après avoir relâché le bouton ? Que faudrait-il changer pour que la LED s'allume quand on n'appuie pas sur le bouton et s'éteigne quand on l'appuie ?

Lecture d'un bouton dans Python

Cliquez sur le bouton New dans Thonny pour commencer un nouveau projet et sur Save pour l'enregistrer. L'utilisation d'une broche GPIO en entrée pour un bouton est très similaire à l'utilisation d'une broche en sortie pour une LED, mais vous devez importer une autre partie de la bibliothèque GPIO Zero. Saisissez ce qui suit dans la zone de script :

```
from gpiozero import Button
button = Button(2)
```

Pour que le code soit exécuté lorsque le bouton est enfoncé, GPIO Zero dispose de la fonction **wait_for_press**. Saisissez donc :

```
button.wait_for_press()
print("Vous avez appuyé!")
```

Cliquez sur le bouton Exécuter, puis appuyez sur le bouton-poussoir. Votre message s'affichera dans le shell de Python en bas de la fenêtre Thonny : vous avez réussi à lire un message transmis par une broche GPIO ! Si vous voulez réessayer votre programme, vous devrez cliquer à nouveau sur le bouton Exécuter ; du moment que le programme ne comporte pas de boucle, il s'arrête dès qu'il a fini d'afficher le message dans le shell.

Pour étendre encore votre programme, ajoutez la LED et la résistance dans le circuit si vous ne l'avez pas encore fait : n'oubliez pas de connecter la résistance à la broche GPIO 25 et à la patte longue de la LED, et la patte plus courte de la LED au rail de masse de votre platine.

Pour contrôler une LED et lire un bouton, vous devez importer les deux fonctions **Button** et **LED** de la bibliothèque GPIO Zero. Vous aurez également besoin de la fonction **sleep** de la bibliothèque **time**. Retournez au début de votre programme et saisissez ce qui suit sur les deux premières lignes :

```
from gpiozero import LED
from time import sleep
```

Sous la ligne **button = Button(2)**, saisissez :

```
led = LED(25)
```

Effacez la ligne **print("Vous m'avez appuyé!")** et remplacez-la par :

```
led.on()
sleep(3)
led.off()
```

Une fois terminé, votre programme se présente comme suit :

```
from gpiozero import LED
from time import sleep
from gpiozero import Button

button = Button(2)
led = LED(25)
button.wait_for_press()
led.on()
sleep(3)
led.off()
```

Cliquez sur le bouton Exécuter, puis appuyez sur le bouton-poussoir : la LED s'allume pendant trois secondes, puis s'éteint et le programme s'arrête. Félicitations : vous pouvez contrôler une LED en utilisant une entrée de bouton dans Python !

DÉFI : AJOUTER UNE BOUCLE

Comment ajouteriez-vous une boucle pour que le programme se répète au lieu de s'arrêter après avoir appuyé une seule fois sur le bouton ? Que faudrait-il changer pour que la LED s'allume quand on n'appuie pas sur le bouton et s'éteigne quand on l'appuie ?

Faites-vous entendre : contrôler un buzzer

Les LED sont un excellent dispositif de sortie, mais elles ne sont pas très utiles si vous regardez ailleurs. La solution : un buzzer, qui émet un bruit audible partout dans la pièce. Pour ce projet, vous aurez besoin d'une platine, d'un câble de connexion mâle-femelle (M2F) et d'un buzzer. Si vous n'avez pas de platine, vous pouvez connecter le buzzer en utilisant plutôt des câbles femelles-femelles (F2F).

En termes de circuits et de programmation, un buzzer peut être traité exactement comme une LED. Répétez le circuit que vous avez mis en place pour la LED, mais remplacez la LED par le buzzer actif et laissez la résistance en dehors, car le buzzer aura besoin de plus de courant. Connectez une patte du buzzer à la broche GPIO 15 (marquée GP15 dans le schéma) illustré dans la **Figure 6-5**) et l'autre à la broche de masse (marquée GND sur le schéma) à l'aide de votre platine et des fils de raccordement mâle-femelle.

Si votre buzzer a trois pattes, assurez-vous que la patte marquée d'un symbole moins (-) est reliée à la broche de masse et que la branche marquée d'un S ou du mot SIGNAL est connectée à GPIO 15, puis connectez la troisième patte (généralement celle du milieu) à la broche 3,3 V (3V3.)

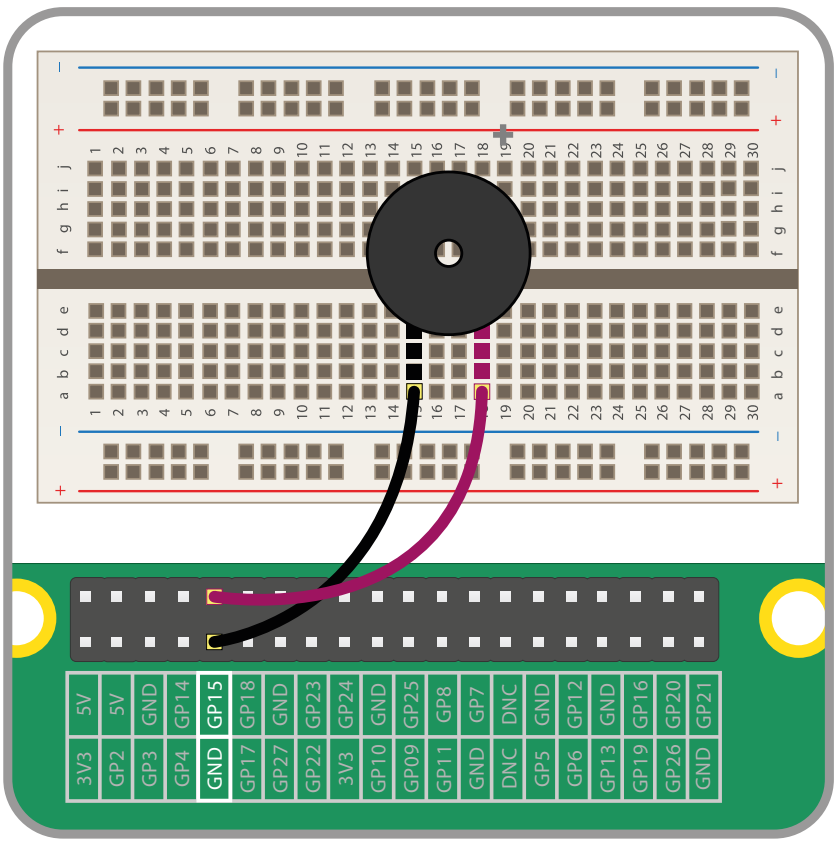


Figure 6-5 : Connexion d'un buzzer aux broches du connecteur GPIO

Commander un buzzer dans Scratch

Recréez le même programme que celui que vous aviez créé pour faire clignoter la LED ou chargez-le, si vous l'avez enregistré avant de créer le projet du bouton. À l'aide de la liste déroulante dans les blocs **set gpio to output high**, sélectionnez le numéro 15, de sorte que Scratch contrôle la broche GPIO qui convient.

Cliquez sur le drapeau vert, et votre buzzer se mettra à sonner, une seconde en marche, une seconde en arrêt. Si vous n'entendez le buzzer qu'une fois par seconde, vous utilisez un buzzer passif plutôt qu'un buzzer actif. Alors qu'un buzzer actif génère un signal qui change rapidement, appelé *oscillation* pour faire vibrer les plaques métalliques, un buzzer passif a besoin d'un signal oscillant. Lorsque vous vous limitez à l'activer en utilisant Scratch, les plaques ne bougent qu'une fois et s'arrêtent, ce qui fait que le son « clic » retentit jusqu'à la prochaine fois que votre programme active ou désactive la broche.

Cliquez sur l'octogone rouge pour arrêter votre buzzer, mais assurez-vous de le faire lorsqu'il n'émet pas de son, sinon le buzzer continuera de sonner jusqu'à ce que vous relanciez votre programme !

DÉFI : CHANGER LE BUZZ

Comment modifieriez-vous le programme pour que le buzzer sonne moins longtemps ? Pouvez-vous construire un circuit pour que le buzzer soit commandé par un bouton ?

Commander un buzzer dans Python

Commander un buzzer actif via la bibliothèque GPIO Zero équivaut à commander une LED, en ce sens qu'un buzzer peut s'allumer et s'éteindre. Il vous faudra cependant une autre fonction : **`buzzer`**. Créez un nouveau projet dans Thonny et enregistrez-le sous **Buzzer**, puis saisissez :

```
from gpiozero import Buzzer
from time import sleep
```

Comme pour les LED, GPIO Zero doit savoir à quelle broche est connecté votre buzzer pour le commander. Saisissez donc :

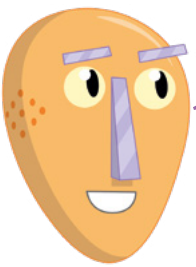
```
buzzer = Buzzer(15)
```

A partir de là, votre programme est presque identique à celui que vous avez écrit pour la LED ; la seule différence (à part un numéro de broche GPIO différent) est que vous utilisez la fonction **`buzzer`** au lieu de la fonction **`led`**. Saisissez donc :

```
while True:
    buzzer.on()
    sleep(1)
    buzzer.off()
    sleep(1)
```

Cliquez sur le bouton Exécuter et votre buzzer se mettra à vibrer, en passant une seconde en marche, une seconde en arrêt. Si vous utilisez un buzzer passif plutôt qu'un buzzer actif, vous n'entendrez qu'un bref clic par seconde au lieu d'un buzz continu : c'est parce qu'un buzzer passif ne possède pas d'*oscillateur* pour créer un signal rapide qui fait vibrer les plaques à l'intérieur du buzzer.

Cliquez sur le bouton Stop pour quitter le programme, mais assurez-vous que le buzzer ne sonne à ce moment-là, sinon il continuera à sonner jusqu'à ce que vous relanciez votre programme !



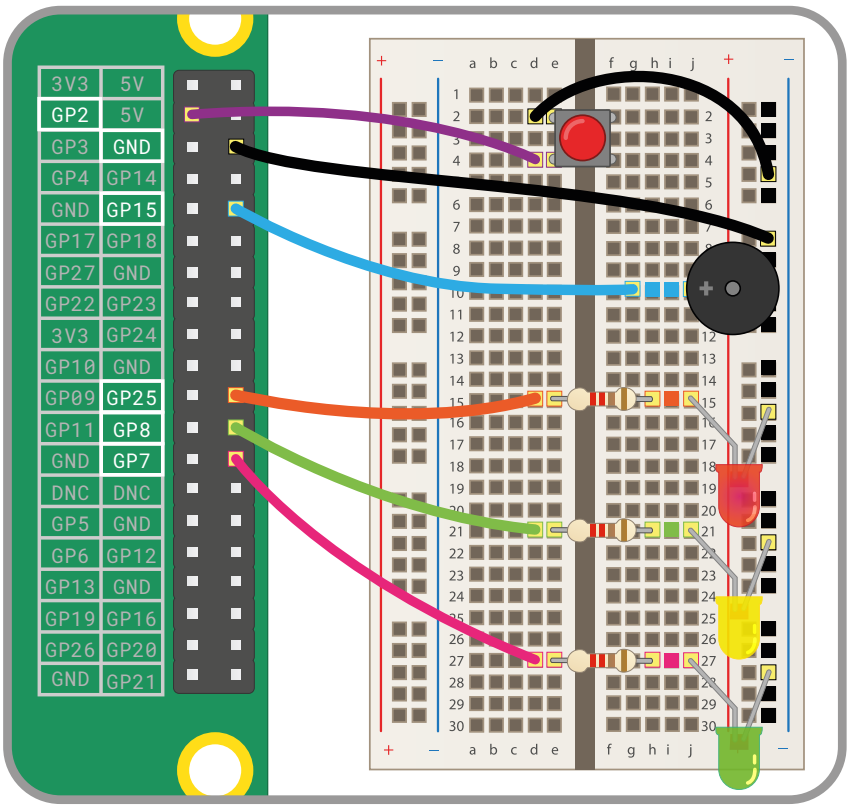
DÉFI : UN MEILLEUR BUZZ

Comment modifieriez-vous le programme pour que le buzzer sonne moins longtemps ? Pouvez-vous construire un circuit pour que le buzzer soit commandé par un bouton ?

Projet Scratch : Feux de circulation

Maintenant que vous savez comment utiliser les boutons, les buzzers et les LED en entrée et en sortie, vous êtes prêt pour vous lancer dans l'informatique du monde réel : les feux de circulation et le bouton sur lequel vous pouvez appuyer pour traverser la route. Pour ce projet, vous aurez besoin d'une platine ; une LED rouge, une verte et une orange ; trois résistances 330 Ω ; un buzzer ; un interrupteur à bouton-poussoir ; plusieurs câbles de raccordement mâle-mâle (M2M) et mâle-femelle (M2F).

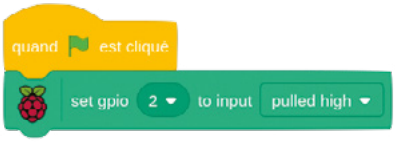
Commencez par construire le circuit (**Figure 6-6**), reliant le buzzer à la broche GPIO 15 (GP15 dans la **Figure 6-6**), la LED rouge à la broche GPIO 25 (GP25), la LED jaune à la GPIO 8 (GP8), la LED verte à la GPIO 7 (GP7) et l'interrupteur à la GPIO 2 (GP2). N'oubliez pas de connecter les résistances 330 Ω entre les broches GPIO et les pattes longues des LED, et de relier les secondes pattes de tous vos composants au rail de masse de votre platine. Pour finir, connectez le rail de masse à une broche de masse (étiquetée GND) sur le Raspberry Pi pour compléter le circuit.



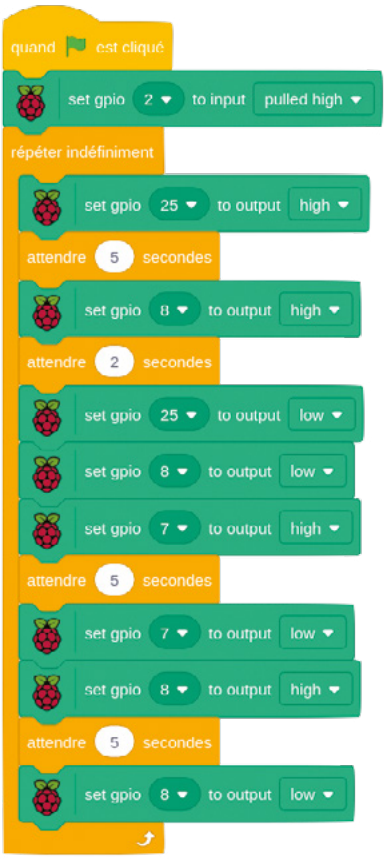
▲ **Figure 6-6 :** Schéma de câblage pour le projet Feux de circulation

Lancez un nouveau projet dans Scratch 3 et faites glisser un bloc quand est cliqué sur la zone de codage. Ensuite, vous devrez indiquer à Scratch que la broche GPIO 2, qui est

connectée au bouton-poussoir de votre circuit, est une entrée et non une sortie : faites glisser un bloc **set gpio to input pulled high** de la catégorie Raspberry Pi GPIO de la palette des blocs sous votre bloc **quand est cliqué**. Cliquez sur la flèche vers le bas à côté de 0 et sélectionnez le numéro 2.



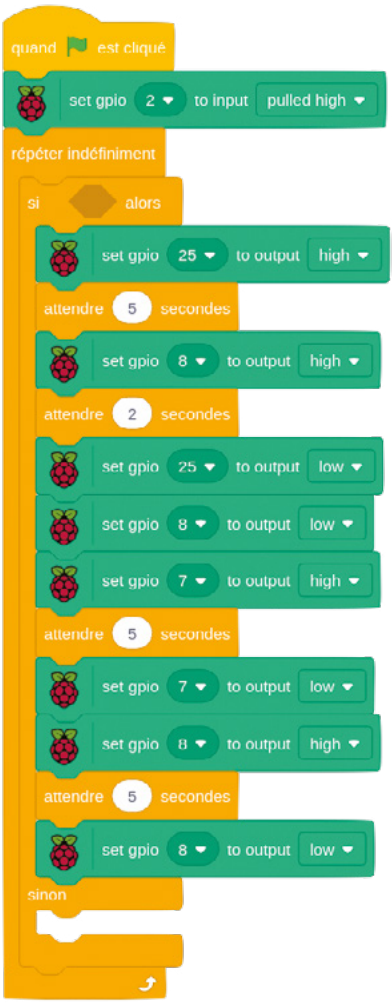
Ensuite, vous devez créer votre séquence de feux de circulation. Faites glisser un bloc **répéter indéfiniment** dans votre programme, puis ajoutez les blocs pour allumer et éteindre les feux de circulation en suivant une séquence précise. Rappelez-vous quelles broches GPIO sont reliées à quel composant : lorsque vous utilisez la broche 25, vous utilisez la LED rouge, la broche 8 est reliée à la LED jaune et la broche 7 à la LED verte.



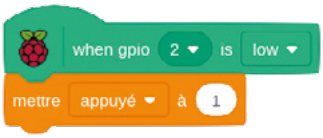
Cliquez sur le drapeau vert, et observez vos LED : d'abord le rouge s'allume, puis le rouge et l'orange, puis le vert, puis l'orange, et pour finir la séquence recommence avec le feu rouge.

Cette séquence correspond à celle utilisée par les feux de circulation au Royaume-Uni ; si vous le souhaitez, vous pouvez modifier la séquence pour l'adapter à celle d'autres pays.

Pour simuler un passage piéton, vous avez besoin de votre programme qui surveille à quel moment le bouton est appuyé. Cliquez sur l'octogone rouge pour arrêter votre programme, s'il est encore en exécution. Faites glisser un bloc **si alors sinon** et connectez-le de manière à ce qu'il soit directement sous votre bloc **répéter indéfiniment**, avec votre séquence de feux de circulation dans la section « si alors ». Laissez l'espace en forme de losange vide pour l'instant.

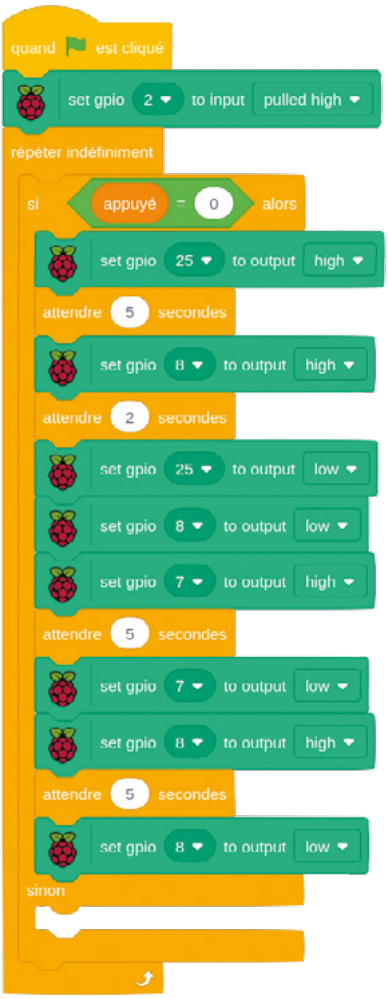


Dans un vrai passage piéton, le feu ne passe pas au rouge dès que le bouton est appuyé : il faut attendre le prochain passage au rouge dans la séquence. Pour intégrer cette logique dans votre propre programme, faites glisser un bloc **when gpio is low** sur la zone de code et sélectionnez « 2 » dans la liste déroulante. Ensuite, faites glisser un bloc **mettre poussé à 1** en dessous.



Cette pile de blocs attend que le bouton soit appuyé, puis définit la variable « appuyé » sur 1. En réglant une variable de la sorte, vous pouvez mémorisez le fait que le bouton a été enfoncé, même si l'action ne se déclenche pas immédiatement.

Retournez à votre pile de blocs d'origine et cherchez le bloc si alors. Faites glisser un bloc Opérateur = dans l'espace en forme de losange à l'intérieur du bloc si alors, puis faites glisser un bloc rapporteur poussé dans le premier espace vide. Saisissez 0 au lieu de 50 à droite du bloc.



Cliquez sur le drapeau vert, et observez les feux de circulation suivre leur séquence. Appuyez sur le bouton-poussoir : au début, vous aurez l'impression qu'il ne se passe rien, mais une fois la séquence terminée (lorsque seule la LED jaune est allumée), les feux de circulation s'éteindront et resteront éteints, grâce à votre variable « appuyé ».

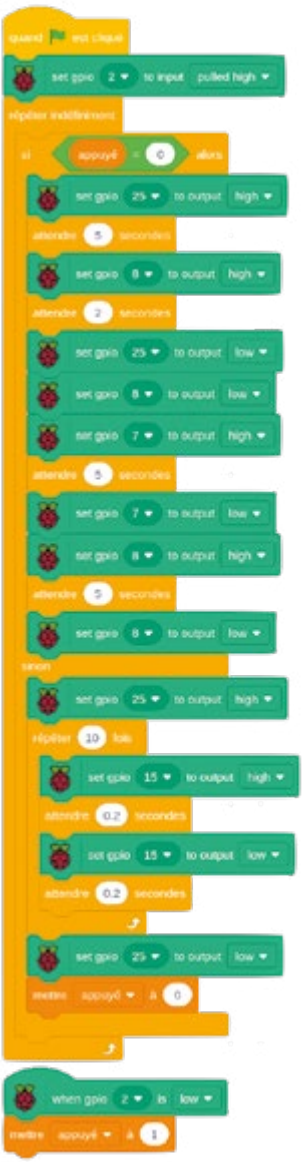
Il ne reste plus qu'à faire en sorte que le bouton du passage pour piétons obtienne un effet autre que l'extinction des feux. Dans la pile de blocs principale, cherchez le bloc **sinon** et faites glisser un bloc **set gpio 25 to output high** à son intérieur (sans oublier de modifier le numéro de broche GPIO par défaut pour qu'il corresponde à la broche sur laquelle votre LED rouge est connectée.

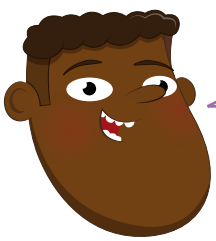
En dessous, toujours dans le bloc **sinon**, créez une séquence pour le buzzer : faites glisser un bloc **répéter 10 fois**, puis remplissez-le avec des blocs **set gpio 15 to output high**, **attendre 0,2 seconde**, **set gpio 15 to output low** et **attendre 0,2 seconde**, en modifiant les valeurs de la broche GPIO pour qu'elles correspondent à la broche du composant du buzzer.

Enfin, au-dessous de votre bloc **répéter 10 fois** mais toujours dans le bloc **sinon**, ajoutez un bloc **set gpio 25 to output low** et un **mettre poussé à 0** (ce dernier bloc réinitialisant la variable qui mémorise la pression du bouton, afin que la séquence de buzzer ne se répète pas indéfiniment).

Cliquez sur le drapeau vert et appuyez sur l'interrupteur sur votre platine. Une fois la séquence terminée, vous verrez le feu rouge s'allumer et le buzzer sonner pour indiquer aux piétons qu'ils peuvent traverser en toute sécurité. Après quelques secondes, le buzzer s'arrête et la séquence des feux de circulation recommence et se poursuit jusqu'à la prochaine pression sur le bouton.

Félicitations : vous avez programmé votre propre schéma de feux de circulation entièrement fonctionnel, comprenant le passage pour piétons !





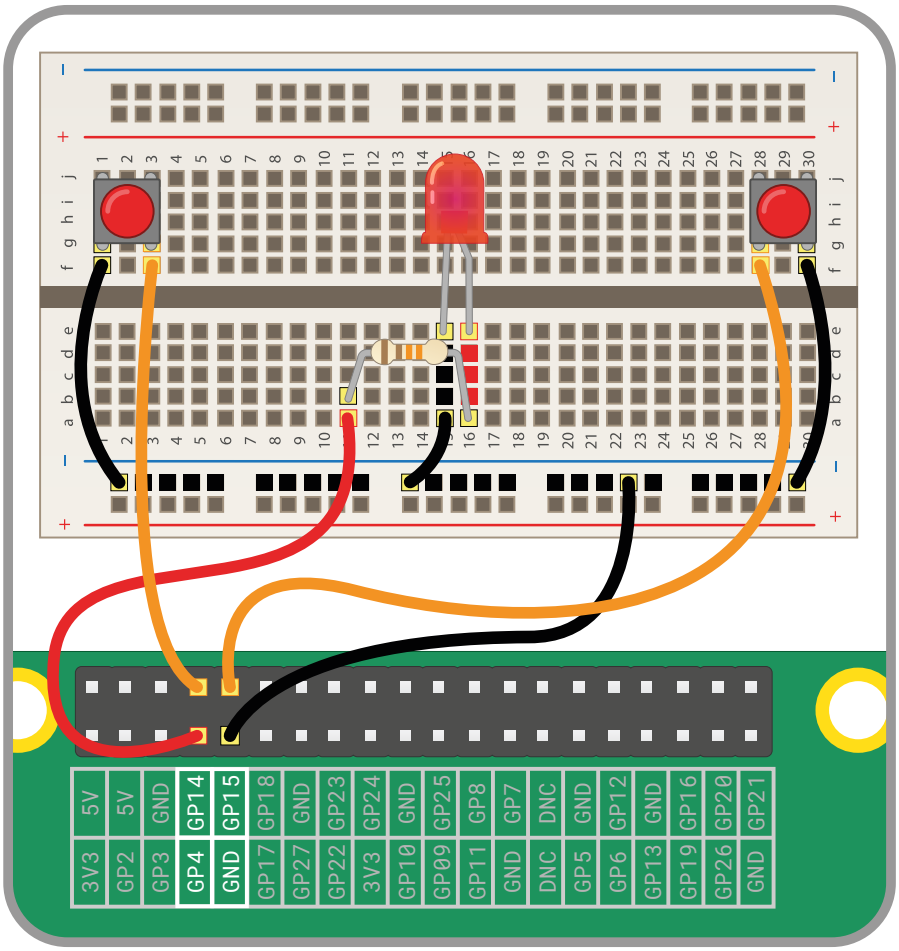
DÉFI : POUVEZ-VOUS L'AMÉLIORER ?

Pouvez-vous modifier le programme afin d'accorder aux piétons plus de temps pour traverser ? Pouvez-vous trouver des informations sur les séquences des feux de circulation dans d'autres pays et reprogrammer vos feux en conséquence ? Comment pourriez-vous réduire la luminosité des LED ?

Projet Python : Jeu de vitesse

Maintenant que vous savez comment utiliser les boutons et les LED en entrée et en sortie, vous êtes prêt pour vous lancer dans l'informatique du monde réel : un jeu de vitesse pour deux joueurs pour calculer lequel des deux a de meilleurs réflexes ! Pour ce projet, vous aurez besoin d'une platine, d'une LED et d'une résistance 330 Ω, de deux interrupteurs à bouton-poussoir, de plusieurs fils de connexion mâle-femelle (M2F) et de quelques fils mâle-mâle (M2M).

Commencez par construire le circuit (**Figure 6-7**) : connectez le premier interrupteur situé à gauche de votre platine à la broche GPIO 14 (GP14 dans la **Figure 6-7**), le deuxième interrupteur sur le côté droit de votre PLATINE à la broche GPIO 15 (GP15), la patte longue de la LED à la résistance 330 Ω qui se connecte ensuite à la broche GPIO 4 (GP4) du Raspberry Pi, et les deuxièmes branches de tous vos composants au rail de masse de la platine. Pour finir, connectez le rail de masse à une broche de masse (GND) du Raspberry Pi.



▲ **Figure 6-7 :** Schéma de câblage pour le Jeu de vitesse

Créez un nouveau projet dans Thonny et enregistrez-le sous **Jeu de réactivité**. Vous allez utiliser les fonctions **LED** et **button** de la bibliothèque GPIO Zero, ainsi que la fonction **sleep** de la bibliothèque « time ». Plutôt que d'importer les deux fonctions GPIO Zero sur deux lignes distinctes, vous pouvez gagner du temps et les importer ensemble en utilisant un symbole de virgule (**,**) pour les séparer. Saisissez ce qui suit dans la zone de script :

```
from gpiozero import LED, Button
from time import sleep
```

Comme auparavant, vous devrez indiquer à GPIO Zero à quelles broches les deux boutons et la LED sont connectés. Saisissez donc :

```
led = LED(4)
right_button = Button(15)
left_button = Button(14)
```

Ajoutez maintenant des instructions pour allumer et éteindre la LED, afin de pouvoir vérifier qu'elle fonctionne correctement :

```
led.on()
sleep(5)
led.off()
```

Cliquez sur le bouton Exécuter : la LED s'allume pendant cinq secondes, puis s'éteint et le programme s'arrête. Pour les besoins d'un jeu de vitesse, cependant, le fait que la LED s'éteigne après exactement 5 secondes à chaque fois est un peu prévisible. Ajoutez ce qui suit au-dessous de la ligne **from time import sleep** :

```
from random import uniform
```

La bibliothèque « random » (aléatoire), comme son nom l'indique, vous permet de générer des nombres aléatoires (ici avec une distribution uniforme : consulter **rpf.io/uniforme**). Cherchez la ligne **sleep(5)** et modifiez-la comme suit :

```
sleep(uniform(5, 10))
```

Cliquez à nouveau sur le bouton Exécuter : cette fois, la LED restera allumée pendant un nombre aléatoire de secondes (entre 5 et 10). Calculez combien de temps il faut à la LED pour s'éteindre, puis cliquez sur le bouton Exécuter plusieurs fois : vous verrez que le temps est différent pour chaque exécution, ce qui rend le programme moins prévisible.

Pour transformer les boutons en déclencheurs pour chaque joueur, vous devrez ajouter une

fonction. Tout en bas de votre programme, saisissez :

```
def pressed(button):
    print(str(button.pin.number) + " a gagné le jeu")
```

Rappelez-vous que Python utilise l'indentation pour indiquer quelles lignes font partie de votre fonction ; Thonny indentera automatiquement la deuxième ligne pour vous. Enfin, ajoutez les deux lignes suivantes pour détecter quel joueur appuie sur le bouton (sans oublier qu'elles ne doivent pas être indentées, sinon Python les traitera comme faisant partie de votre fonction).

```
right_button.when_pressed = pressed
left_button.when_pressed = pressed
```

Exécutez votre programme, et cette fois, essayez d'appuyer sur l'un des deux boutons dès que la LED s'éteint. Un message s'affiche annonçant le premier bouton à avoir été appuyé sur le shell Python en bas de la fenêtre de Thonny. Malheureusement, vous verrez également des messages à chaque fois que vous appuierez sur l'un ou l'autre des boutons, utilisant un code pin plutôt qu'un nom convivial pour le bouton.

Pour remédier à cela, commencez par demander aux joueurs de vous donner leurs noms. Au-dessous de la ligne **`from random import uniform`**, saisissez :

```
left_name = input("Le joueur de gauche s'appelle ")
right_name = input("Le joueur de droite s'appelle ")
```

Retournez à votre fonction et remplacez la ligne **`print(str(button.pin.number) + " a gagné ")`** par :

```
    if button.pin.number == 14:
        print (left_name + " a gagné")
    else:
        print(right_name + " a gagné")
```

Cliquez sur le bouton Exécuter, puis saisissez les noms des deux joueurs dans la zone du shell Python. Maintenant, lorsque vous appuyez sur le bouton (n'oubliez pas, aussi vite que possible dès que la LED s'éteint), vous verrez que le nom du joueur s'imprime à la place du numéro d'identification.

Pour éviter que chaque pression de bouton soit annoncée comme gagnante, vous devez ajouter une nouvelle fonction à partir de la bibliothèque sys (abréviation de *système*), à savoir la fonction **`exit (sortir)`**. Sous la dernière ligne **`import`**, saisissez ce qui suit :

```
from os import _exit
```

Ensuite, à la fin de votre fonction, sous la ligne **`print(right_name + " a gagné le jeu ")`**, saisissez :

```
    _exit(0)
```

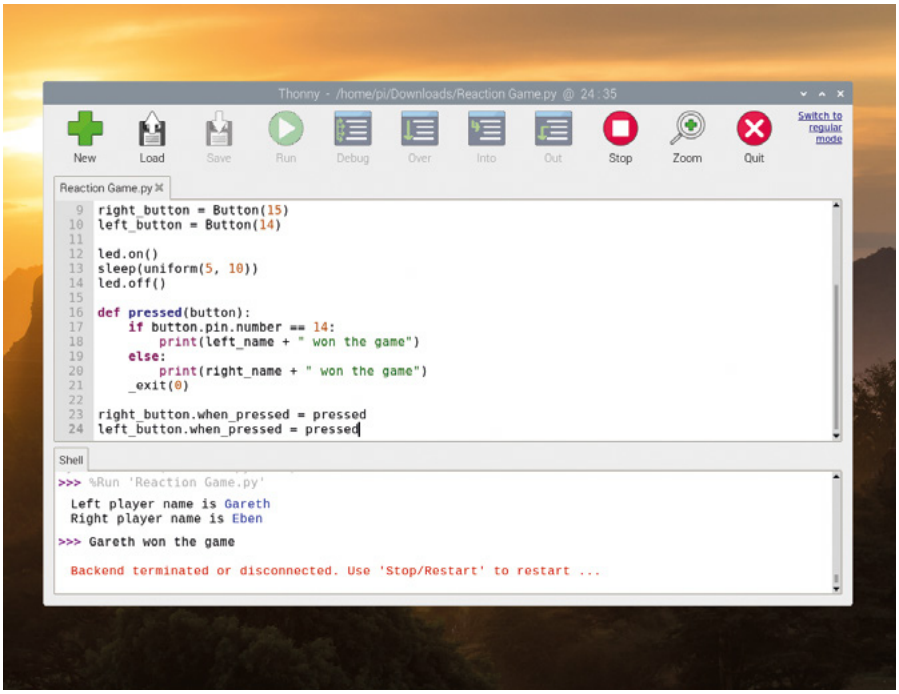
L'indentation est essentielle ici : **`_exit(0)`** doit être indenté de quatre espaces, s'alignant sur **`else:`** deux lignes au-dessus et sur **`if`** encore deux lignes au-dessus. Cette instruction indique à Python d'arrêter le programme après que le premier bouton a été appuyé, ce qui signifie que le joueur dont le bouton est appuyé en second n'obtient aucune récompense pour avoir perdu !

Une fois terminé, votre programme se présente comme suit :

```
from gpiozero import LED, Button
from time import sleep
from random import uniform
from os import _exit

left_name = input("Le joueur de gauche s'appelle ")
right_name = input ("Le joueur de droit s'appelle ")
led = LED(4)
right_button = Button(15)
left_button = Button(14)

led.on()
sleep(uniform(5, 10))
led.off()

def pressed(button):
    if button.pin.number == 14:
        print(left_name + " a gagné")
    else:
        print(right_name + " a gagné")
    _exit(0)

right_button.when_pressed = pressed
left_button.when_pressed = pressed
```

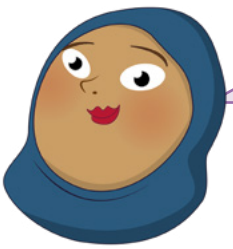
Cliquez sur le bouton Exécuter, saisissez les noms des joueurs, attendez que la LED s'éteigne et le nom du joueur gagnant s'affichera. Vous verrez également un message de Python : **`Backend terminated or disconnected . Use 'Stop/Restart' to restart`**

`...` Cela signifie simplement que Python a reçu votre commande **`_exit(0)`** et a arrêté le programme, mais que vous devrez cliquer sur l'icône Stop pour l'arrêter complètement et préparer votre programme pour un autre tour (**Figure 6-8**).



▲ **Figure 6-8 :** Une fois le gagnant désigné, vous devrez arrêter le programme

Félicitations : vous avez créé votre propre jeu !



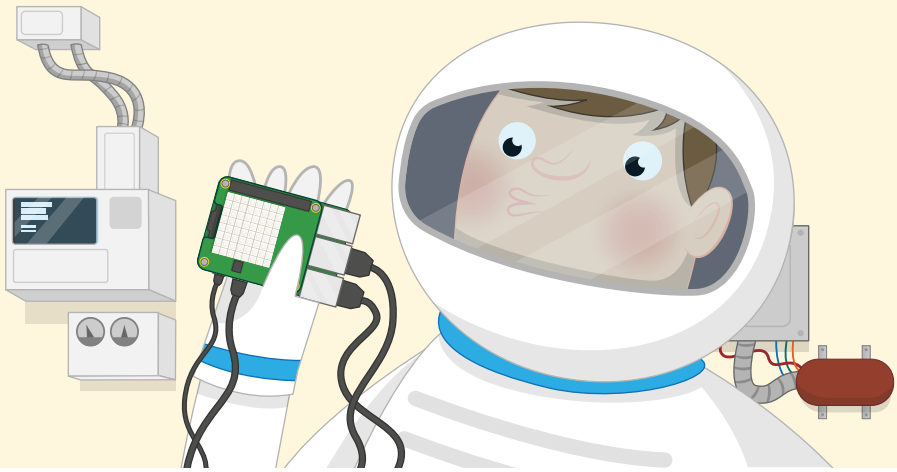
DÉFI : AMÉLIORER LE JEU

Pouvez-vous ajouter une boucle, pour que le jeu fonctionne en continu ? N'oubliez pas de supprimer l'instruction **`_exit(0)`** d'abord ! Pouvez-vous ajouter un compteur de points, afin de désigner le gagnant sur plusieurs tours ? Et pourquoi pas une minuterie, pour calculer le temps de réaction lorsque la LED s'éteint ?

Chapitre 7

L'informatique physique avec la Sense HAT

Utilisée sur la Station spatiale internationale, la Sense HAT est une carte complémentaire multifonctionnelle pour Raspberry Pi, équipée de capteurs et d'un écran matriciel à LED



Le Raspberry Pi est accompagné d'un type spécial de carte complémentaire appelée *HAT (Hardware Attached on Top ou matériel ajouté)*. Les HAT peuvent tout faire : des microphones et des lumières aux relais et écrans électroniques, en passant par le Raspberry Pi, mais un HAT nous intéresse tout particulièrement : la Sense HAT.

La Sense HAT a été conçue spécialement pour la mission spatiale Astro Pi. Dans le cadre de ce projet commun porté par la Raspberry Pi Foundation, l'Agence spatiale du Royaume-Uni et l'Agence spatiale européenne, des cartes Raspberry Pi et Sense HAT ont atteint la Station spatiale internationale à bord d'une fusée cargo Cygnus d'Orbital Sciences. Une fois placés en orbite en toute sécurité au-dessus de la Terre, les Sense Hat (surnommés Ed et Izzy par les astronautes) ont été utilisées pour exécuter des codes et réaliser des expériences scientifiques auxquelles ont participé des écoliers de toute l'Europe.

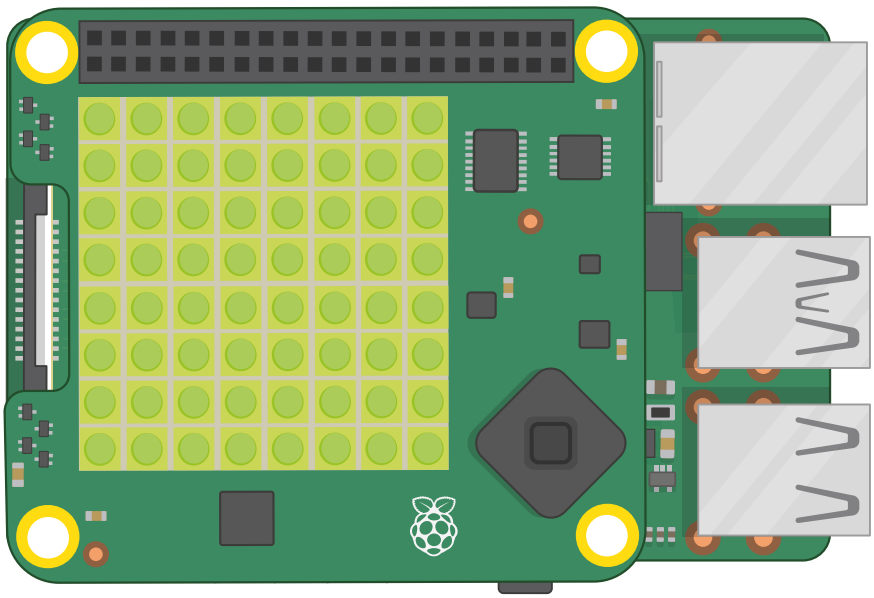
Vous n'allez probablement pas utiliser vos Sense HAT comme Ed et Izzy, mais ce même matériel se trouve également ici sur Terre, chez tous les vendeurs de Raspberry Pi. D'ailleurs, si vous ne voulez pas acheter une Sense HAT tout de suite, vous pouvez en simuler un via un logiciel !

RÉEL OU SIMULÉ

Ce chapitre a été conçu en ayant en tête une Sense HAT physique rattachée au connecteur GPIO d'un Raspberry Pi, mais si vous n'en possédez pas une, il est possible d'ignorer la section « Installation de la Sense HAT » et de simplement essayer les projets dans l'émulateur Sense HAT : ils fonctionneront tout aussi bien !

Présentation de la Sense HAT

La Sense HAT constitue un complément puissant et multifonctionnel pour Raspberry Pi. Outre une matrice 8×8 de 64 LED rouges, vertes et bleues (RGB) programmables qui peuvent être commandées pour produire n'importe quelle couleur parmi des millions, la Sense HAT comprend un contrôleur à cinq manettes et six capteurs sur carte.



Capteur gyroscope : Utilisé pour détecter les changements d'angle au fil du temps, désignés par le terme technique de *vitesse angulaire* en suivant la direction du champ de gravité de la Terre (la force qui attire les objets vers le centre de la planète). Pour faire simple, le capteur gyroscopique vous indique à quel moment vous faites pivoter la Sense HAT par rapport à la surface de la Terre et à quelle vitesse.

Accéléromètre : Semblable au capteur gyroscope, mais plutôt qu'une mesure d'angle par rapport à la gravité de la Terre, il mesure la force d'accélération dans plusieurs directions. Combinées, les lectures (données) des deux capteurs peuvent vous aider à suivre la direction vers laquelle se dirige la Sense Hat et comment elle se déplace.

Magnétomètre : Le magnétomètre mesure la force d'un champ magnétique et peut aider à suivre les mouvements de la Sense HAT : en mesurant le champ magnétique naturel de la Terre, le magnétomètre peut déterminer la direction du nord magnétique. Ce même capteur peut également être utilisé pour détecter des objets métalliques et des champs électriques. Ces trois capteurs sont intégrés dans une seule puce, appelée « ACCEL/GYRO/MAG » sur la carte de la Sense HAT.

Capteur d'humidité : Mesure la teneur en vapeur d'eau contenue dans l'air, connue sous le nom d'*humidité relative*. L'humidité relative peut varier de 0 % (absence totale d'eau), à 100 % (saturation totale de l'air). Les données relatives à l'humidité peuvent être utiles pour comprendre la probabilité de précipitations !

Capteur de pression barométrique : également connu sous le nom de *baromètre*, il mesure la pression atmosphérique. Pour la plupart des personnes, le baromètre est en lien avec les prévisions météo, mais il a également une seconde utilité secrète : lors de l'ascension ou de la descente d'une colline ou d'une montagne, il peut suivre votre progression en sachant que l'air se raréfie et la pression baisse à mesure que vous vous éloignez du niveau de la mer.

Capteur de température : Mesure la température ambiante, bien que ce capteur soit influencé par la température de la Sense HAT : si vous utilisez un boîtier, vos valeurs seront probablement plus élevées que prévu. La Sense HAT ne dispose pas de capteur de température séparé : elle utilise plutôt des capteurs de température intégrés aux capteurs d'humidité et de pression barométrique. Un programme peut utiliser l'un de ces capteurs ou les deux, ce sera à vous de faire votre choix.

SENSE HAT SUR LE RASPBERRY PI 400

La Sense HAT est entièrement compatible avec le Raspberry Pi 400, et peut être insérée directement dans le connecteur GPIO à l'arrière de celui-ci. Cependant, en procédant de la sorte les LED seront à l'arrière et le haut de la carte sera cette fois en bas.

Pour y remédier, vous aurez besoin d'un câble ou d'une carte d'extension GPIO. Les extensions compatibles comprennent la gamme Black HAT Hack3r de Pimoroni : vous pouvez utiliser la Sense HAT avec la carte Black HAT Hack3r elle-même, ou simplement utiliser la limande à 40 points incluse en guise d'extension. Vérifiez toujours les instructions du fabricant pour vous assurer de brancher le câble et la Sense HAT dans le bon sens !

Installation de la Sense HAT

Si vous avez une Sense HAT physique, commencez par la déballer et assurez-vous que toutes les pièces sont présentes : vous devez avoir la Sense HAT en soi, quatre montants en métal ou en plastique appelés *entretoises*, et huit vis. Il est possible que des broches métalliques disposées sur une bande en plastique noir soient également dans le lot, similaire aux connecteurs GPIO du Raspberry Pi ; si c'est le cas, appuyez la bande contre la partie basse de la Sense HAT avec les broches vers le haut jusqu'à ce que vous entendiez un clic.

Les entretoises sont conçues pour éviter que la Sense HAT se plie ou se torde en utilisant le joystick. Votre Sense HAT fonctionnera même si elles ne sont pas installées, leur utilisation permet de protéger votre Sense HAT, votre Raspberry Pi et votre connecteur GPIO de tout risque d'endommagement.

ATTENTION !

Les modules Hardware Attached on Top (matériel ajouté) (HAT) ne doivent être branchés et débranchés du connecteur GPIO que lorsque votre Raspberry Pi est éteint et déconnecté de sa source d'alimentation électrique. Veillez toujours à ce que le module HAT reste bien à plat lors de son installation et vérifiez qu'il est bien aligné avec les broches du collecteur GPIO avant de le pousser vers le bas.

Installez les entretoises en insérant quatre vis en dessous de votre de Raspberry Pi à travers les quatre orifices de montage à chaque coin, puis vissez les entretoises. Appuyez la Sense HAT sur le connecteur GPIO du Raspberry Pi, en veillant à l'aligner correctement avec les broches et à la maintenir aussi horizontale que possible. Enfin, fixez la Sense HAT aux entretoises que vous avez installées précédemment en insérant les quatre dernières vis dans les orifices prévus à cet effet. Si elle est installée correctement, la Sense HAT doit être parfaitement à plat et ne doit pas se plier ou vibrer lorsque vous appuyez sur son joystick.

Rebranchez l'alimentation sur votre Raspberry Pi, et vous verrez les voyants LED de la Sense HAT s'allumer en arc-en-ciel (**Figure 7-1**), puis s'éteindre à nouveau. Votre Sense Hat est bien installée !

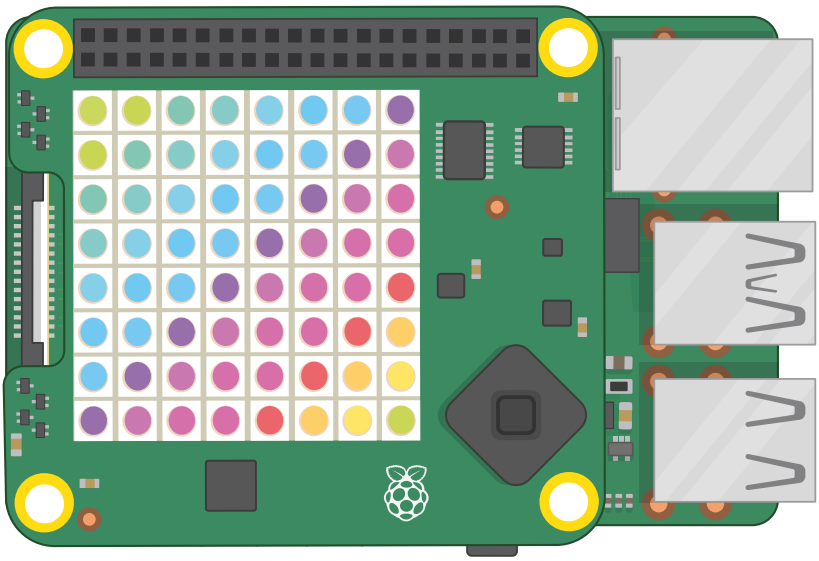
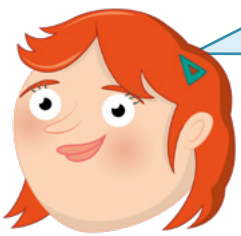


Figure 7-1 : Un effet arc-en-ciel apparaît lors de la première mise sous tension

Si vous souhaitez retirer la Sense Hat, il vous suffit de desserrer les vis du haut, de soulever la HAT (en ayant soin de ne pas plier les broches du connecteur GPIO car la HAT tient généralement fermement, raison pour laquelle un tournevis pourrait vous être utile), puis de retirer les entretoises du Raspberry Pi.

Hello, Sense HAT !

Comme pour tous les projets de programmation, la première activité à réaliser avec votre Sense HAT sera bien entendu celle de faire défiler un message de bienvenue sur son écran LED. Si vous utilisez l'émulateur Sense HAT, chargez-le maintenant en cliquant sur l'icône du menu Raspberry Pi OS, en choisissant la catégorie Programmation, et en cliquant sur Sense HAT Emulator.



EXPÉRIENCE DE PROGRAMMATION

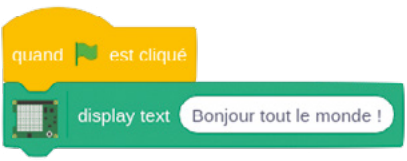
Ce chapitre suppose que vous possédez une certaine expérience de Scratch 3 ou Python et l'environnement de développement intégré (IDE) de Thonny, selon que vous travaillez avec les exemples de code Scratch ou Python (ou les deux !) Si vous ne l'avez pas encore fait, lisez le **Chapitre 4, Programmation avec Scratch** ou le **Chapitre 5, Programmation avec Python** et réalisez les projets décrits dans ces chapitres.

Message de bienvenue de Scratch

Chargez Scratch 3 depuis le menu Raspberry Pi OS. Cliquez sur le bouton « Ajouter une extension » en bas à gauche de la fenêtre « Scratch ». Cliquez sur l'extension Sense HAT de Raspberry Pi (**Figure 7-2**). Ce faisant, vous pourrez charger les blocs dont vous avez besoin pour contrôler les différentes fonctions de la Sense HAT, y compris son écran LED. Quand vous en aurez besoin, vous les retrouverez dans la catégorie Sense HAT du Raspberry Pi.

Figure 7-2 : Ajouter l'extension Sense HAT de Raspberry Pi dans Scratch 3

Commencez par faire glisser un bloc d'événement quand est cliqué sur la zone de script, puis faites glisser un bloc display text Hello! directement en dessous. Modifiez le texte de manière à ce que le bloc indique display text Bonjour tout le monde !.



Cliquez sur le drapeau vert sur la scène et observez votre Sense HAT ou votre simulateur Sense HAT : le message défilera lentement le long de la matrice LED de la Sense HAT, illuminant les pixels à LED pour former chacune des lettres, l'une après l'autre (**Figure 7-3**, au verso). Félicitations : votre programme est un franc succès !

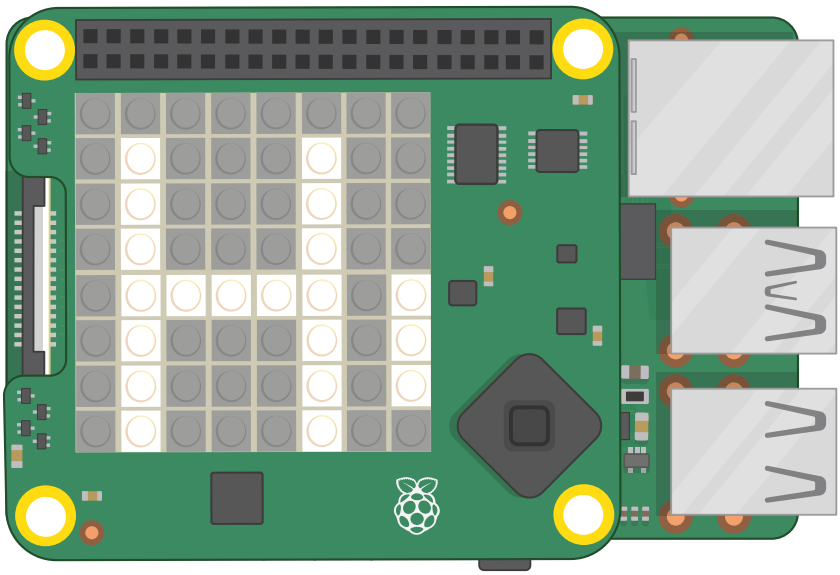


Figure 7-3 : Votre message défile sur la matrice LED

Puisque vous pouvez maintenant faire défiler un message simple, il est désormais temps de définir l'affichage de ce message. En plus de pouvoir modifier le message à afficher, vous pouvez modifier le sens de défilement de votre message sur la Sense Hat. Faites glisser un bloc set rotation to 0 degrees de la palette des blocs et insérez-le en dessous de quand est cliqué et au-dessus de display text Bonjour tout le monde !, puis cliquez sur la flèche vers le bas à côté de 0 et changez-le en 90. Cliquez sur le drapeau vert et vous verrez le même message qu'auparavant, mais au lieu de défiler de gauche à droite, il défilera de bas en haut (**Figure 7-4**) : vous devrez tourner la tête, ou le Sense HAT, pour le lire !

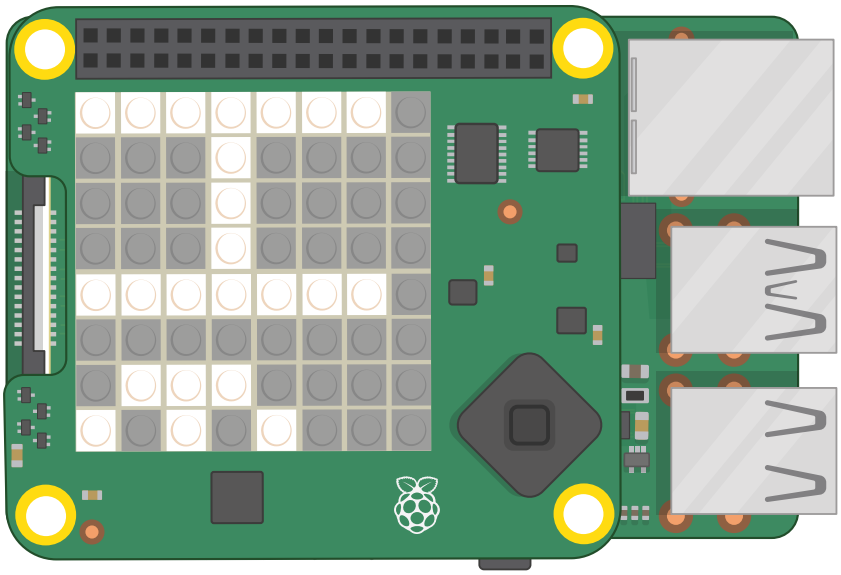


Figure 7-4 : Cette fois, le message défile verticalement

Redéfinissez maintenant la rotation sur 0, puis faites glisser un bloc **set colour** entre **set rotation to 0 degrees** et **dispay text Bonjour tout le monde !**. Cliquez sur la couleur à la fin du bloc pour faire apparaître le sélecteur de couleurs de Scratch et choisissez un beau jaune vif, puis cliquez sur le drapeau vert pour observer les changements dans le rendu de votre programme (**Figure 7-5**).

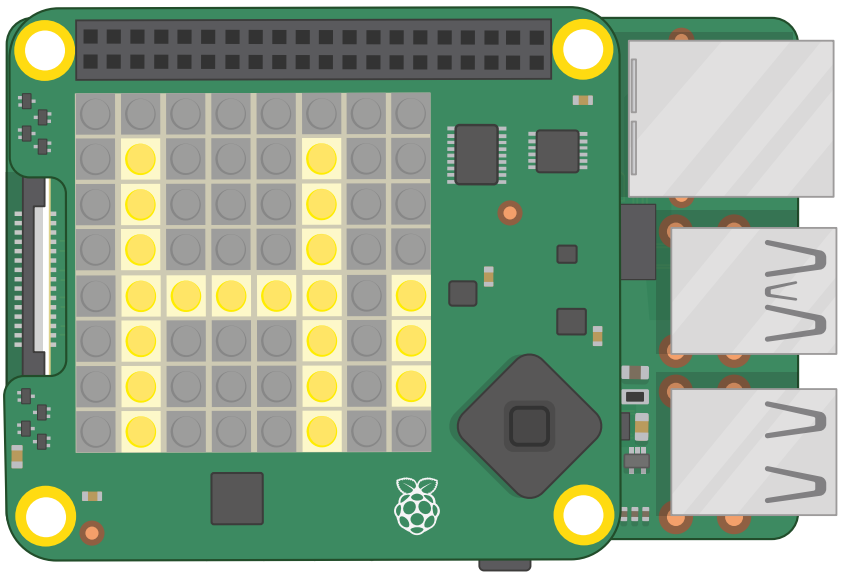


Figure 7-5 : Changer la couleur du texte

Enfin, faites glisser un bloc set background entre set colour to jaune et display text Bonjour tout le monde ! , puis cliquez sur la couleur pour faire réapparaître le sélecteur de couleur. Cette fois, le choix d'une couleur n'affecte pas les LED qui composent le message, mais s'appliquera à tous les autres, c'est-à-dire au fond. Choisissez une belle couleur bleue, puis cliquez à nouveau sur le drapeau vert : cette fois, votre message sera jaune vif sur fond bleu. Essayez plusieurs couleurs pour trouver votre combinaison préférée : toutes les couleurs ne vont pas bien ensemble !

Vous pouvez faire défiler des messages entiers, mais également des lettres individuelles. Faites glisser votre bloc display text hors de la zone de script pour le supprimer, puis faites glisser un bloc display character A sur la zone de script à sa place. Cliquez sur le drapeau vert, et observez la différence : ce bloc n'affiche qu'une seule lettre à la fois, et la lettre reste sur la Sense HAT jusqu'à ce que vous le vouliez, sans défiler et sans disparaître. Les mêmes blocs de couleur s'appliquent à ce bloc en tant que bloc display text : tentez de modifier la couleur de la lettre en rouge (**Figure 7-6**).

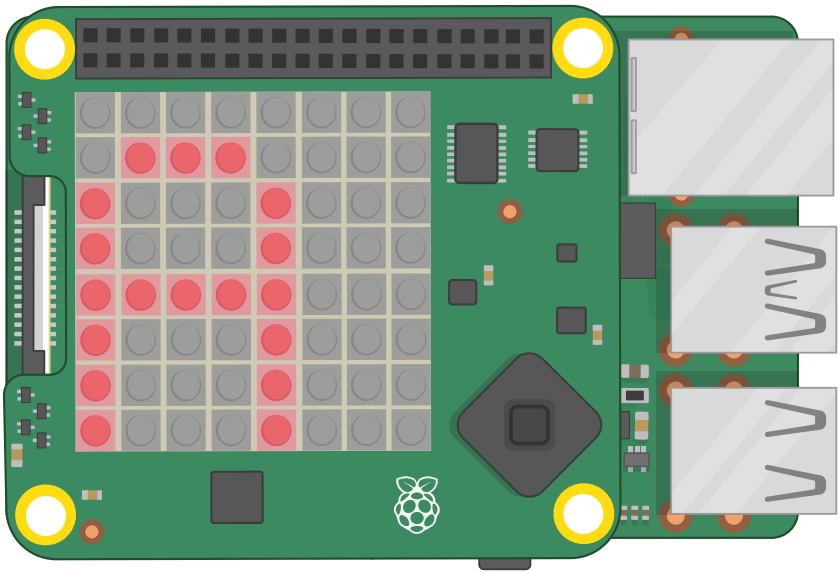
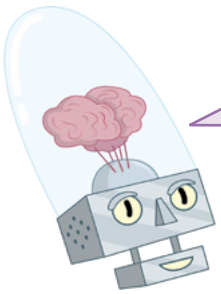


Figure 7-6 : Affichage d'une seule lettre



DÉFI : MESSAGE RÉPÉTÉ

Pouvez-vous mettre à profit vos connaissances en matière de boucles pour faire en sorte qu'un message défilant se répète ? Pouvez-vous réaliser un programme qui épelle un mot lettre par lettre en utilisant différentes couleurs ?

Message de bienvenue de Python

Chargez Thonny en cliquant sur l'icône du menu Raspbian, en choisissant Programmation, et en cliquant sur Thonny. Si vous utilisez l'émulateur Sense HAT et qu'il est couvert par la fenêtre Thonny, cliquez et maintenez le bouton de la souris sur la barre de titre de l'une ou l'autre fenêtre (en haut, en bleu) et faites-la glisser pour la déplacer sur le bureau jusqu'à ce que vous puissiez voir les deux fenêtres.

CHANGEMENT DE LIGNE DANS PYTHON

Le code Python écrit pour une Sense HAT physique fonctionnera sur l'émulateur Sense HAT et vice-versa, avec une seule modification. Si vous utilisez l'émulateur Sense HAT avec Python, vous devrez remplacer la ligne `sense_hat import SenseHat` dans tous les programmes de ce chapitre par `from sense_emu import SenseHat`. Si vous voulez ensuite les exécuter à nouveau sur une Sense HAT physique, il suffit de revenir à la ligne d'origine !

Pour utiliser la Sense HAT, ou l'émulateur Sense HAT, dans un programme Python, vous devez importer la bibliothèque Sense HAT. Saisissez ce qui suit dans la zone de script, en vous souvenant d'utiliser **`sense_emu`** (au lieu de **`sense_hat`**) si vous utilisez l'émulateur Sense HAT :

```
from sense_hat import SenseHat
sense = SenseHat()
```

La fonction de la bibliothèque Sense HAT est de prendre en charge un message, le formater de manière à ce qu'il puisse être affiché sur l'écran LED et le faire défiler de manière fluide. Saisissez donc :

```
sense.show_message("Bonjour tout le monde !")
```

Cliquez sur le bouton Exécuter et enregistrez votre programme sous le nom **Hello Sense HAT**. Vous verrez votre message défiler lentement le long de la matrice LED de la Sense HAT, illuminant les pixels à LED pour former chacune des lettres, l'une après l'autre (**Figure 7-7**, au verso). Félicitations : votre programme est un franc succès !

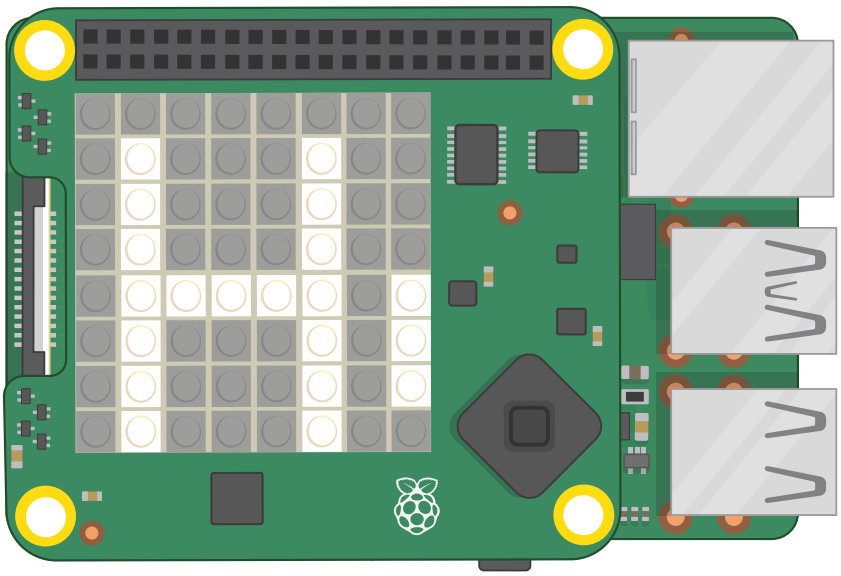


Figure 7-7 : Défilement d'un message sur la matrice LED

La fonction **`show_message()`** a bien plus à offrir. Retournez à votre programme et modifiez la dernière ligne, comme suit :

```
sense.show_message("Bonjour tout le monde !", text_colour=(255,
255, 0), back_colour=(0, 0, 255), scroll_speed=(0.05))
```

Ces instructions supplémentaires, séparées par des virgules, sont connues sous le nom de *paramètres* et contrôlent divers aspects de la fonction **`show_message()`**. Le plus simple est **`scroll_speed=()`**, qui modifie la vitesse de défilement du message sur l'écran. Une valeur de 0,05 fait défiler le message deux fois plus vite que de coutume. Plus la valeur est élevée, plus la vitesse est faible.

Les paramètres **`text_colour=()`** et **`back_colour=()`** (orthographiés à la manière de l'anglais britannique, contrairement à la plupart des instructions de Python) fixent respectivement la couleur du texte et du fond. Cependant, ils ne prennent pas en charge les noms des couleurs en format texte : vous devez indiquer la couleur souhaitée sous la forme d'une série de trois chiffres. Le premier chiffre représente la proportion de rouge présent dans la couleur, 0 indiquant une proportion nulle à 255 le plus de rouge possible ; le deuxième chiffre représente la proportion de vert dans la couleur et le troisième chiffre la quantité de bleu. L'ensemble de ces trois chiffres est connu sous le nom de *RGB* (red, green, blue, ou rouge, vert et bleu en français).

Cliquez sur l'icône Exécuter et observez la Sense HAT : cette fois, le message jaune vif sur fond bleu défilera beaucoup plus rapidement (**Figure 7-8**). Essayez de modifier les paramètres pour trouver une combinaison de vitesse et de couleurs qui vous convienne.

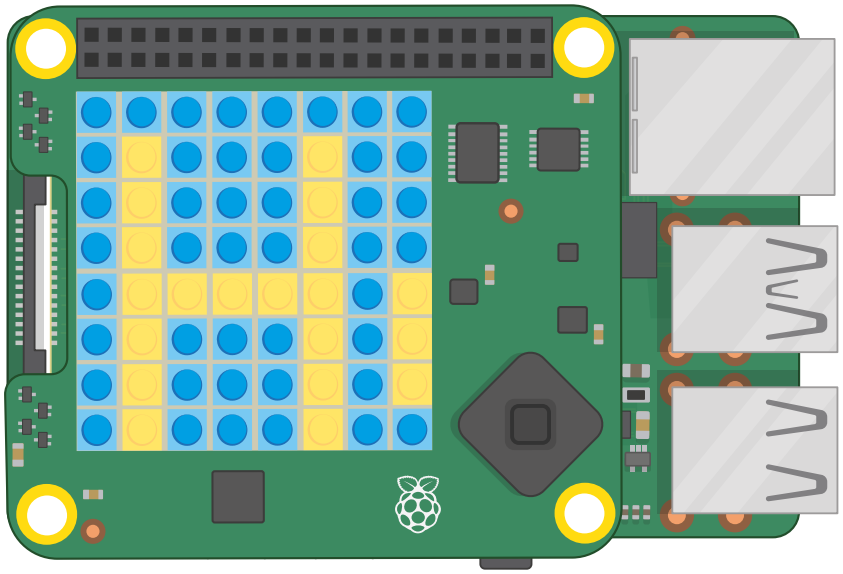


Figure 7-8 : Changer la couleur du message et du fond

Si vous souhaitez utiliser des noms plus conviviaux au lieu de valeurs RGB pour définir vos couleurs, vous devrez créer des variables. Au-dessus de votre ligne **`sense.show_message()`**, ajoutez :

```
jaune = (255, 255, 0)
bleu = (0, 0, 255)
```

Revenez à votre ligne **`sense.show_message()`** et modifiez-la de la sorte :

```
sense.show_message("Bonjour tout le monde !", text_colour=(jaune), back_colour=(bleu), scroll_speed=(0.05))
```

Cliquez à nouveau sur l'icône Exécuter, et vous verrez que rien n'a changé : votre message est toujours en jaune sur fond bleu. Cette fois, cependant, vous avez utilisé les noms de variables pour rendre votre code plus lisible : au lieu d'une chaîne de chiffres, le code indique clairement la couleur souhaitée. Vous pouvez définir autant de couleurs que vous le souhaitez : essayez d'ajouter une variable « rouge » avec des valeurs 255, 0 et 0 ; une variable « blanc » avec des valeurs 255, 255, 255 ; et une variable « noir » avec des valeurs 0, 0 et 0.

Vous pouvez faire défiler des messages entiers, mais également des lettres individuelles. Supprimez votre ligne **`sense.show_message()`** et saisissez ce qui suit à la place :

```
sense.show_letter("Z")
```

Cliquez sur Exécuter et vous verrez la lettre « Z » s'afficher sur l'écran de la Sense HAT. Cette fois, elle restera immobile : les lettres individuelles, contrairement aux messages, ne défilent pas automatiquement. Vous pouvez contrôler `sense.show_letter()` avec les mêmes paramètres de couleur que `sense.show_message()` : essayez de changer la couleur de la lettre en rouge (**Figure 7-9**).

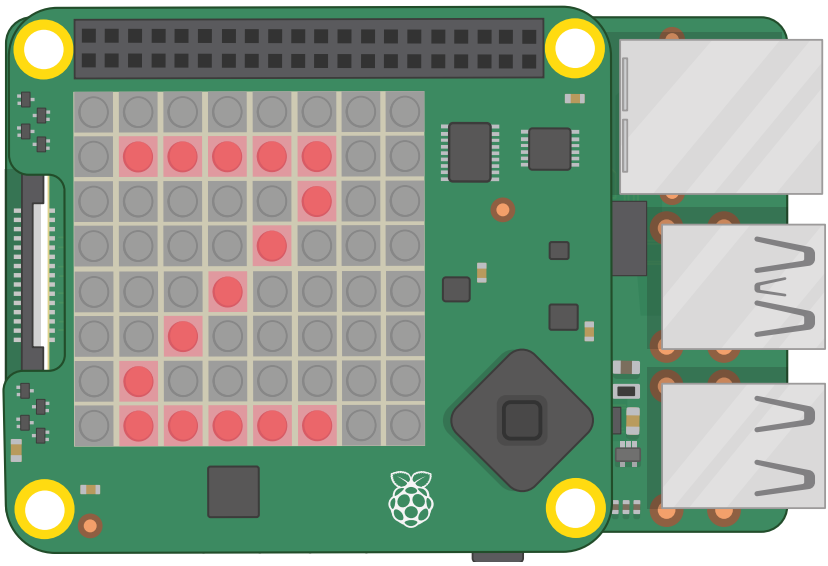
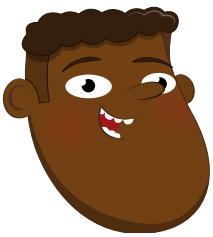


Figure 7-9 : Affichage d'une seule lettre



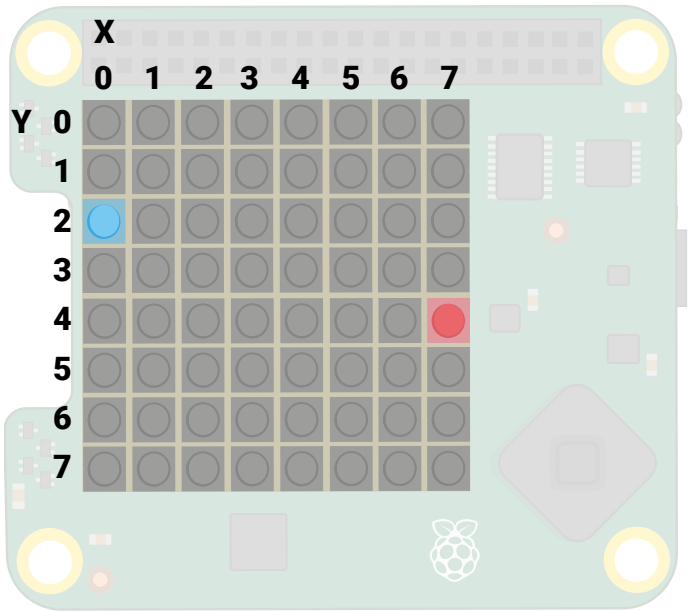
DÉFI : MESSAGE RÉPÉTÉ

Pouvez-vous mettre à profit vos connaissances en matière de boucles pour faire en sorte qu'un message défilant se répète ? Pouvez-vous réaliser un programme qui épelle un mot lettre par lettre en utilisant différentes couleurs ? À quelle vitesse pouvez-vous faire défiler un message ?

Prochaines étapes : Dessins lumineux

L'écran LED de la Sense HAT ne se limite pas à afficher des messages : vous pouvez également afficher des images. Chaque LED peut être traitée comme un pixel (abréviation de *picture element, ou élément d'image*) dans une image de votre choix, ce qui vous permet d'agrémenter vos programmes d'images et même d'animations.

Pour créer des dessins, il faut cependant pouvoir modifier les LED une par une. Pour ce faire, vous devez comprendre la disposition de la matrice LED de la Sense HAT afin d'écrire un programme en mesure d'allumer ou d'éteindre les bonnes LED.



▲ **Figure 7-10 :** Système de coordonnées de la matrice LED

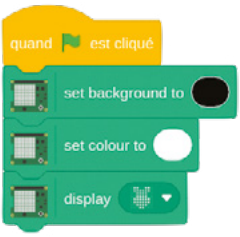
Chaque ligne et chaque colonne de l'écran comportent huit LED (**Figure 7-10**). Cependant, lorsque vous comptez les LED, vous devez, comme la plupart des langages de programmation, compter de 0 à 7. La première LED se trouve dans le coin supérieur gauche, la dernière en bas à droite. En utilisant les chiffres des lignes et des colonnes, vous pouvez indiquer les *coordonnées* de chaque LED sur la matrice. Les coordonnées de la LED bleue dans la matrice illustrée sont 0, 2 ; la LED rouge est aux coordonnées 7, 4. L'axe X, horizontal, est indiquée en premier, suivi de l'axe Y, en vertical.

Lorsque vous planifiez les images que vous souhaitez dessiner sur la Sense HAT, il peut être utile de les dessiner d'abord à la main, sur du papier quadrillé, ou à l'aide d'un tableur tel que LibreOffice Calc.

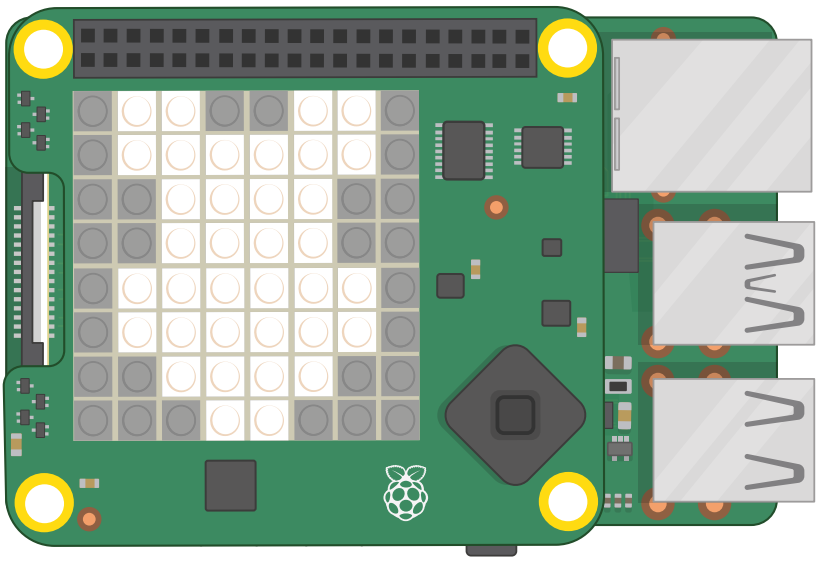
Images dans Scratch

Commencez un nouveau projet dans Scratch, en sauvegardant votre projet en cours si vous souhaitez le conserver. Si vous avez réalisé les projets décrits dans ce chapitre, Scratch 3 conservera l'extension Raspberry Pi Sense HAT que vous avez chargée ; si vous avez fermé et rouvert Scratch 3 depuis votre dernier projet, chargez l'extension en utilisant le bouton Ajouter extension. Commencez par faire glisser un bloc d'événement quand est cliqué sur la zone de script, puis faites glisser un bloc set background et un bloc set colour directement en dessous. Modifiez les deux pour définir la couleur de fond en noir et la couleur du texte en blanc : le noir est obtenu en faisant glisser les curseurs Luminosité et Saturation sur 0 ; pour le blanc, faites glisser la Luminosité sur 100 et la Saturation sur 0. Vous devez y penser au début de chaque

programme Sense HAT, sinon Scratch utilisera simplement les dernières couleurs que vous aurez choisies, même si vous les avez choisies dans un autre programme. Faites glisser ensuite un bloc display framboise en bas de votre programme.



Cliquez sur le drapeau vert : vous verrez les LED de la Sense HAT former une framboise lumineuse (**Figure 7-11**).

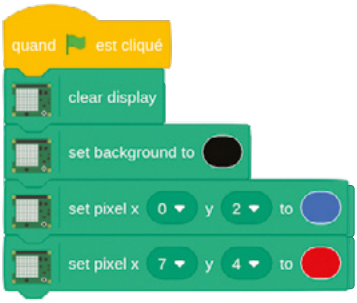


▲ **Figure 7-11 :** Ne regardez pas directement les LED lorsqu'elles sont blanches

Ne vous limitez pas à la forme de framboise prédéfinie. Cliquez sur la flèche vers le bas à côté de la framboise pour activer le mode dessin : vous pouvez cliquer sur n'importe quelle LED du motif pour l'allumer ou l'éteindre, tandis que les deux boutons du bas permettent d'éteindre ou d'allumer toutes les LED. Essayez de dessiner votre propre image, puis cliquez sur la flèche verte pour la voir affichée sur la Sense HAT. Essayez également de changer la couleur et la couleur de fond à l'aide des blocs ci-dessus.

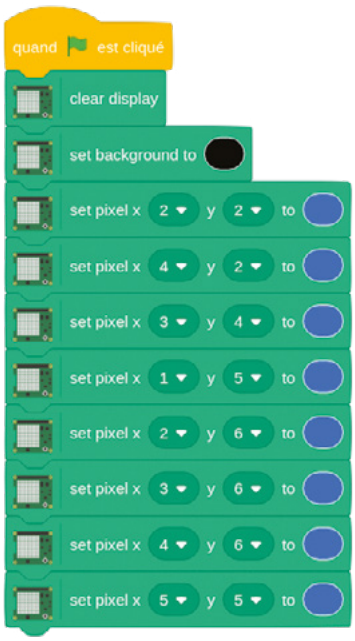
Lorsque vous avez terminé, faites glisser les trois blocs dans la palette des blocs pour les supprimer, et placez un bloc clear display en dessous de quand est cliqué ; cliquez sur le drapeau vert, et toutes les LED s'éteignent.

Pour réaliser une image, vous devez être en mesure de contrôler les différents pixels et leur attribuer des couleurs différentes. Vous pouvez y parvenir en enchaînant des blocs édités display framboise avec des blocs set colour , ou vous pouvez vous charger de chaque pixel individuellement. Pour créer votre propre version de l'exemple de matrice LED présenté au début de cette section, avec deux LED spécifiquement sélectionnées en rouge et bleu, laissez le bloc clear display en haut de votre programme et faites glisser un bloc set background en dessous. Changez le bloc set background en noir, puis faites glisser deux blocs set pixel x 0 y 0 en dessous. Pour finir, modifiez ces blocs comme suit :

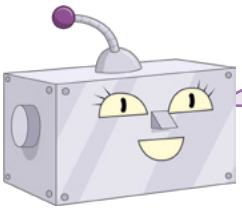


Cliquez sur le drapeau vert, et vous verrez vos LED s'allumer pour former l'image matricielle (**Figure 7-10**) illustrée à la page 165. Félicitations : vous pouvez contrôler les LED une par une !

Modifiez vos blocs de pixels existants comme suit, et faites glisser d'autres blocs vers le bas jusqu'à obtenir le programme suivant :



Avant de cliquer sur le drapeau vert, essayez de deviner l'image qui va apparaître en vous basant sur les coordonnées de la matrice LED que vous avez utilisée, puis lancez votre programme et voyez si vous avez raison !



DÉFI : NOUVEAUX DESIGNS

Pouvez-vous concevoir d'autres images ? Procurez-vous du papier millimétré ou quadrillé et l'utiliser pour planifier votre image manuellement. Pouvez-vous dessiner une image et la faire changer de couleur ?

Images dans Python

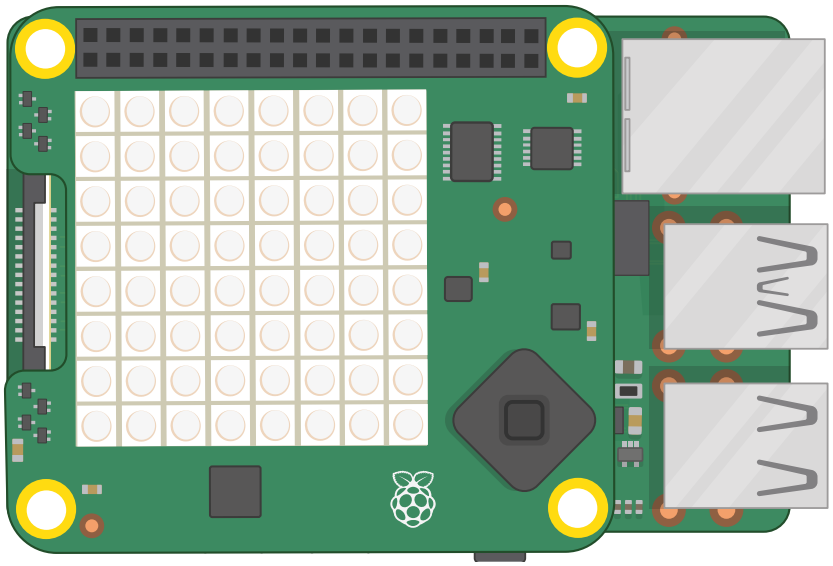
Lancez un nouveau programme dans Thonny et enregistrez-le en tant que Dessin Sense HAT, puis saisissez ce qui suit sans oublier d'utiliser **`sense_emu`** (au lieu de **`sense_hat`**) si vous utilisez l'émulateur Sense HAT :

```
from sense_hat import SenseHat
sense = SenseHat()
```

N'oubliez pas que vous avez besoin de ces deux lignes dans votre programme pour utiliser Sense HAT. Ensuite, saisissez :

```
sense.clear(255, 255, 255)
```

Sans regarder directement les LED de Sense HAT, cliquez sur l'icône Exécuter : vous devez normalement les voir toutes devenir d'un blanc éclatant (**Figure 7-12**) : c'est pourquoi vous ne devez pas les regarder directement lorsque vous exécutez votre programme !



▲ **Figure 7-12 :** Ne regardez pas directement les LED lorsqu'elles sont illuminées en blanc éclatant

Le paramètre **`sense.clear()`** est conçu pour supprimer toute programmation antérieure des LED, mais prend en charge les paramètres de couleur RGB, ce qui signifie que vous pouvez changer l'affichage dans la couleur de votre choix. Essayez de modifier la ligne comme suit :

```
sense.clear(0, 255, 0)
```

Cliquez sur Exécuter, et la Sense HAT prend une couleur vert vif (**Figure 7-13**, au verso). Essayez différentes couleurs ou ajoutez les variables de couleur que vous avez créées pour votre programme Bonjour tout le monde pour en rendre la lecture plus aisée.

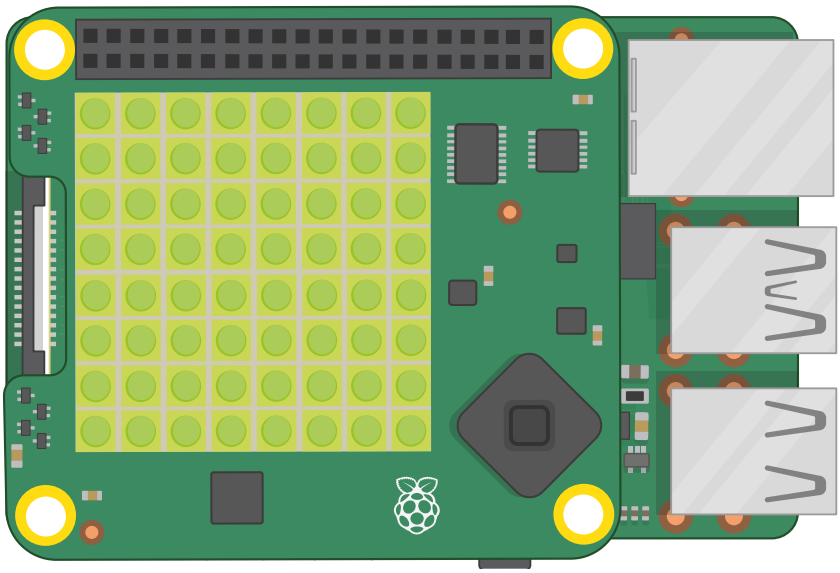


Figure 7-13 : La matrice de LED illuminée en vert vif

Pour effacer les valeurs attribuées aux LED, vous devez utiliser les valeurs RGB pour le noir : 0 rouge, 0 bleu et 0 vert. Il existe cependant un moyen plus facile. Modifiez la ligne de votre programme, comme suit :

```
sense.clear()
```

La Sense HAT s'assombrit car pour la fonction **`sense.clear()`**, le fait de ne rien mettre entre parenthèses équivaut à lui dire de les colorer en noir, donc de les éteindre (**Figure 7-14**). Lorsque vous avez besoin de réinitialiser toutes les valeurs LED de vos programmes, cette fonction est extrêmement utile.

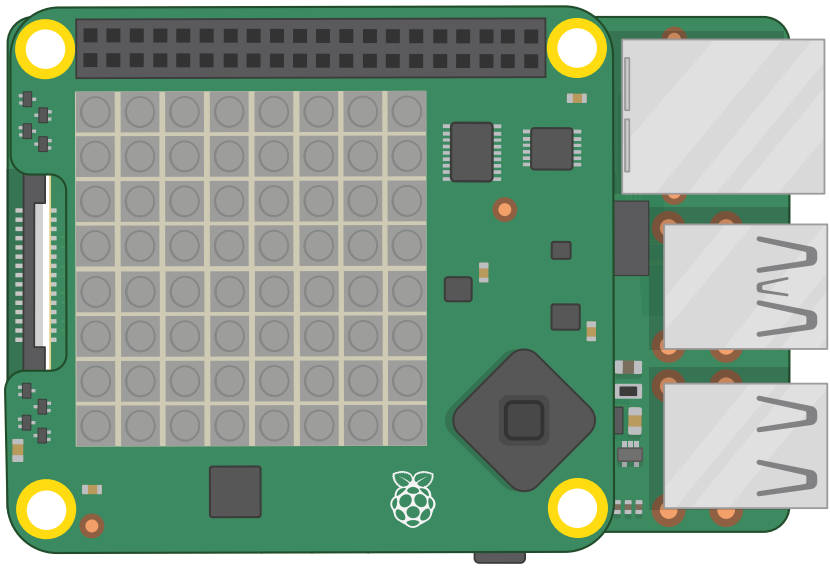


Figure 7-14 : Utilisez la fonction `sense.clear` pour éteindre toutes les LED

Pour créer votre propre version de la matrice LED illustrée au début de cette section, avec deux LED spécifiquement sélectionnées en rouge et bleu, laissez le bloc en haut de votre programme et ajoutez les lignes suivantes à votre programme après **`sense.clear()`** :

```
sense.set_pixel(0, 2, (0, 0, 255))
sense.set_pixel(7, 4, (255, 0, 0))
```

Les deux premiers chiffres correspondent à l'emplacement du pixel sur la matrice, l'axe X (horizontal) suivi de l'axe Y (vertical). Ensuite, entre parenthèses, se trouvent les valeurs RGB de la couleur des pixels. Cliquez sur le bouton Exécuter et observez l'effet : deux LED de votre Sense HAT s'allument, comme dans la **figure 7-10** à la page 165.

Supprimez ces deux lignes et saisissez ce qui suit :

```
sense.set_pixel(2, 2, (0, 0, 255))
sense.set_pixel(4, 2, (0, 0, 255))
sense.set_pixel(3, 4, (100, 0, 0))
sense.set_pixel(1, 5, (255, 0, 0))
sense.set_pixel(2, 6, (255, 0, 0))
sense.set_pixel(3, 6, (255, 0, 0))
sense.set_pixel(4, 6, (255, 0, 0))
sense.set_pixel(5, 5, (255, 0, 0))
```

Avant de cliquer sur Exécuter, regardez les coordonnées et comparez-les à la matrice : pouvez-vous deviner à quelle image correspondent ces instructions ? Cliquez sur Exécuter pour savoir si vous avez deviné !

Dessiner une image détaillée à l'aide des fonctions **`set_pixel()`** prend du temps. Pour accélérer le processus, vous pouvez modifier plusieurs pixels en même temps. Supprimez toutes vos lignes
`set_pixel()` et saisissez :

```
g = (0, 255, 0)
b = (0, 0, 0)

creeper_pixels = [
    g, g, g, g, g, g, g, g,
    g, g, g, g, g, g, g, g,
    g, b, b, g, g, b, b, g,
    g, b, b, g, g, b, b, g,
    g, g, g, b, b, g, g, g,
    g, g, b, b, b, b, g, g,
    g, g, b, b, b, b, g, g,
    g, g, b, g, g, b, g, g
]

sense.set_pixels(creeper_pixels)
```

Il y en a beaucoup, mais commencez par cliquer sur Exécuter pour voir si vous reconnaissez votre petite créature. Les deux premières lignes créent deux variables pour définir les couleurs : le vert et le noir. Pour faciliter l'écriture et la lecture du code du dessin, les variables sont représentées par des lettres : la lettre « g » représente le vert et la lettre « b » représente le noir.

Le bloc de code suivant crée une variable qui définit les valeurs de couleur pour les 64 pixels de la matrice LED, séparées par des virgules et placées entre crochets. Au lieu de chiffres, ce code utilise les variables de couleur que vous avez créées précédemment : en regardant de près et gardant à l'esprit que la lettre « g » représente le vert et la lettre « b » représente le noir, vous pouvez déjà apercevoir l'image qui va apparaître (**Figure 7-15**).

Enfin, la fonction **`sense.set_pixels(creeper_pixels)`** s'appuie sur cette variable et utilise la fonction **`sense.set_pixels()`** pour dessiner l'entière matrice en une seule fois. Cela est bien plus facile que de tracer des images pixel par pixel !

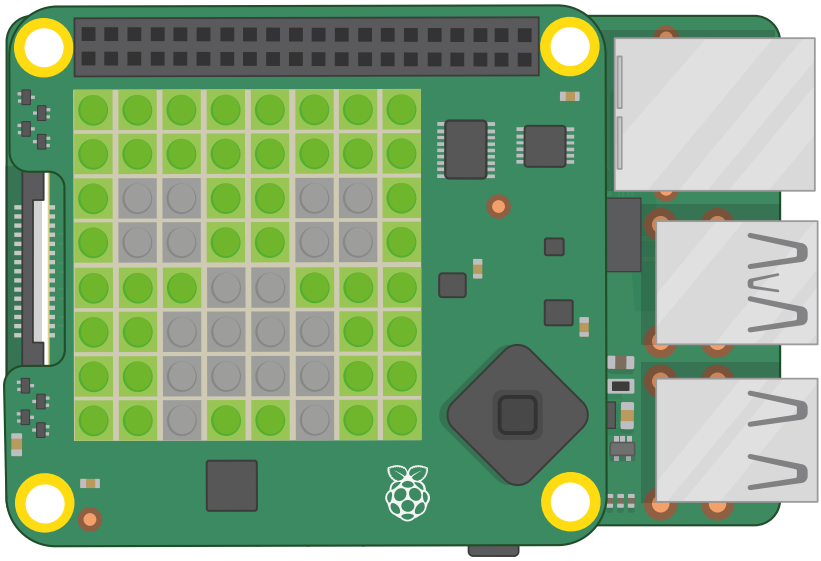


Figure 7-15 : Affichage d'une image sur la matrice

Vous pouvez également faire pivoter et retourner les images, soit pour montrer les images dans le bon sens lorsque vous retournez votre Sense HAT, soit pour créer des animations simples à partir d'une seule image asymétrique.

En premier lieu, modifiez votre variable **`creeper_pixels`** pour fermer son œil gauche, en remplaçant les quatre pixels « b », à partir des deux premiers sur la troisième ligne, suivis des deux premiers sur la quatrième ligne, par « g » :

```
creeper_pixels = [
    g, g, g, g, g, g, g, g,
    g, g, g, g, g, g, g, g,
    g, g, g, g, g, b, b, g,
    g, g, g, g, g, b, b, g,
    g, g, g, b, b, g, g, g,
    g, g, b, b, b, b, g, g,
    g, g, b, b, b, b, g, g,
    g, g, b, g, g, b, g, g
]
```

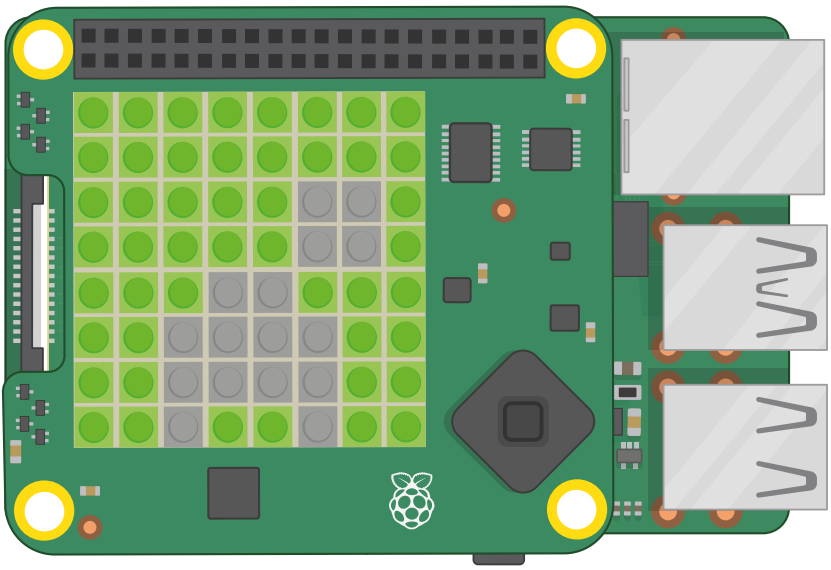
Cliquez sur Exécuter, et vous verrez la créature fermer œil gauche (**Figure 7-16**, au verso). Pour créer une animation, revenez au début de votre programme et ajoutez la ligne :

```
from time import sleep
```

Ensuite, allez tout en bas et saisissez :

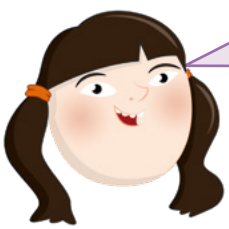
```
while True:
    sleep(1)
    sense.flip_h()
```

Cliquez sur Exécuter, et regardez la créature fermer et ouvrir les yeux, l'un après l'autre !



▲ **Figure 7-16 :** Projection d'une simple animation de deux images

La fonction **`flip_h()`** permet de retourner une image autour de son axe horizontal ; si vous voulez retourner une image autour de son axe vertical, remplacez **`sense.flip_h()`** par **`sense.flip_v()`**. Vous pouvez également faire pivoter une image de 0, 90, 180 ou 270 degrés en utilisant la fonction **`sense.set_rotation(90)`**, en changeant le nombre de degrés en fonction de l'angle auquel vous souhaitez faire pivoter l'image. Essayez de retourner la créature au lieu de lui faire cligner des yeux !

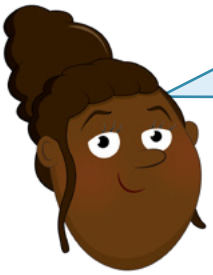


DÉFI : NOUVEAUX DESIGNS

Pouvez-vous concevoir d'autres images et animations ? Procurez-vous du papier millimétré ou quadrillé et l'utiliser pour planifier votre image manuellement, ce qui simplifiera l'écriture de la variable. Pouvez-vous dessiner une image et la faire changer de couleur ? Conseil : vous pouvez modifier les variables après les avoir utilisées une fois.

Sentir le monde qui vous entoure

Le véritable pouvoir du Sense HAT réside dans les différents capteurs dont elle dispose. Ils vous permettent de prendre des mesures de tout ce qui vous entoure, de la température à l'accélération, et de les utiliser dans vos programmes à votre convenance.



L'ÉMULATION DES CAPTEURS

Si vous utilisez un émulateur Sense HAT, vous devrez activer la simulation des capteurs inertiels et environnementaux : dans l'émulateur, cliquez sur Édition, puis sur Préférences en cochant les cases correspondantes. Dans le même menu, sélectionnez « 180°..360°|0°..180° » sous « Échelle d'orientation » pour vous assurer que les chiffres de l'émulateur correspondent aux chiffres indiqués par Scratch et Python, puis cliquez sur le bouton « Close (Fermer) ».

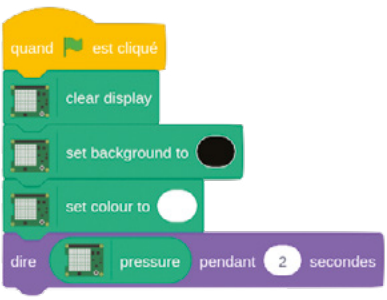
Détection de l'environnement

Le capteur de pression barométrique, le capteur d'humidité et le capteur de température sont tous des capteurs environnementaux qui mesurent différents paramètres dans l'environnement qui entoure la Sense HAT.

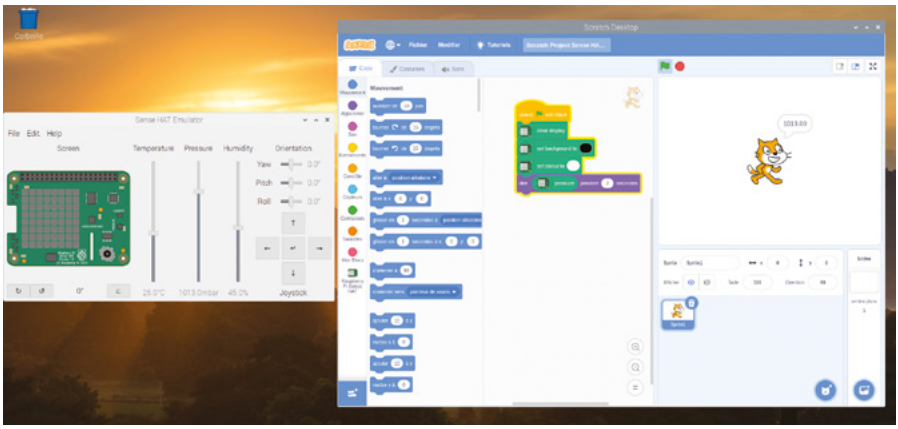
Détection de l'environnement dans Scratch

Démarrez un nouveau programme dans Scratch, en sauvegardant votre ancien programme si vous le souhaitez, et ajoutez l'extension Sense HAT Raspberry Pi si vous ne l'avez pas encore chargée. Commencez par faire glisser un bloc d'événement quand est cliqué dans votre zone de script, puis faites glisser un bloc clear display au-dessous et un bloc set background to noir en dessous de ce dernier. Ensuite, ajoutez un bloc set colour to blanc : utilisez les curseurs de luminosité et de saturation pour choisir la bonne couleur. Il convient toujours de procéder de la sorte au début de vos programmes, car cela permettra d'éviter que la Sense HAT affiche des éléments de programmes précédents tout en garantissant les couleurs de votre choix.

Faites glisser un bloc Apparence dire Bonjour ! pendant 2 secondes directement sous vos blocs existants. Pour effectuer une lecture du capteur de pression, recherchez le bloc pressure dans la catégorie Raspberry Pi Sense HAT et faites-le glisser au-dessus du mot « Bonjour ! » dans votre bloc dire Bonjour ! pendant 2 secondes.



Cliquez sur le drapeau vert et le chat Scratch vous indiquera les valeurs du capteur de pression en *millibars*. Le message s'efface au bout de deux secondes ; essayez de souffler sur le Sense HAT (ou de déplacer le curseur de pression vers le haut dans l'émulateur) et cliquez sur le drapeau vert pour exécuter à nouveau le programme ; cette fois, vous devriez obtenir une valeur plus élevée (**Figure 7-17**, au verso).

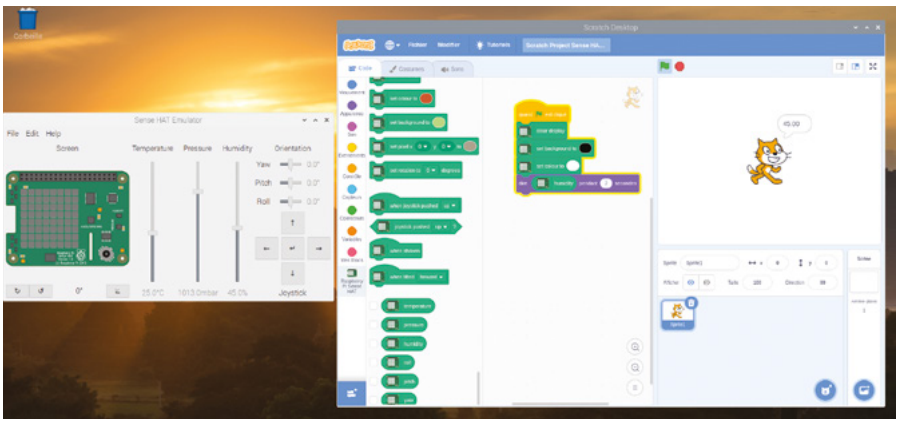


▲ **Figure 7-17 :** Affichage du relevé du capteur de pression

MODIFICATION DES VALEURS

Si vous utilisez l'émulateur Sense HAT, vous pouvez modifier les valeurs signalées par chacun des capteurs émulés à l'aide des curseurs et boutons correspondants. Essayez de faire glisser le curseur du capteur de pression vers le bas, puis cliquez à nouveau sur le drapeau vert.

Pour passer au capteur d'humidité, supprimez le bloc pressure et remplacez-le par le bloc humidity. Relancez votre programme, et l'humidité relative actuelle de la pièce s'affiche. Encore une fois, vous pouvez essayer de l'exécuter à nouveau en soufflant sur le Sense HAT (ou en déplaçant le curseur Humidité de l'émulateur vers le haut) pour observer le changement de lecture (**Figure 7-18**) : votre haleine est étonnamment humide !



▲ **Figure 7-18 :** Affichage du relevé du capteur d'humidité

Pour le capteur de température, il suffit de supprimer le bloc humidity et de le remplacer par temperature, puis de relancer votre programme. La température s'affiche en degrés Celsius (**Figure 7-19**). Il se peut toutefois que la lecture de la température de votre pièce ne soit pas tout à fait exacte : Raspberry Pi génère de la chaleur pendant son temps de fonctionnement, ce qui réchauffe également la Sense HAT et ses capteurs.

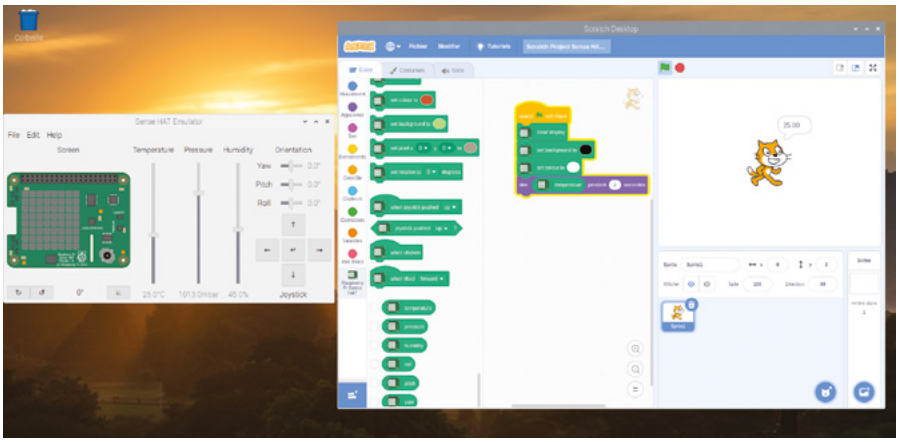
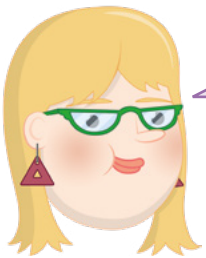


Figure 7-19 : Affichage du relevé du capteur de température



DÉFI : DÉFILEMENT ET BOUCLE

Pouvez-vous modifier votre programme pour afficher la lecture de chacun des capteurs à tour de rôle, puis les faire défiler sur la matrice LED plutôt que de les imprimer sur la scène ? Pouvez-vous faire tourner votre programme en boucle, afin qu'il affiche constamment les conditions environnementales actuelles ?

Détection de l'environnement dans Python

Pour avoir accès à la lecture des capteurs, créez un nouveau programme dans Thonny et enregistrez-le sous **Capteurs Sense HAT**. Saisissez ce qui suit dans la zone de script, comme toujours en utilisant la Sense HAT, et n'oubliez pas de préciser `sense_emu` si vous utilisez l'émulateur :

```
from sense_hat import SenseHat
sense = SenseHat()
sense.clear()
```

Il convient toujours d'inclure la fonction `sense.clear()` au début de vos programmes, pour éviter que la Sense HAT n'affiche des éléments du dernier programma exécuté.

Pour obtenir les valeurs du capteur de pression, saisissez :

```
pression = sense.get_pressure()
print(pression)
```

Cliquez sur Exécuter et vous verrez un numéro imprimé sur le shell Python en bas de la fenêtre de Thonny. Il s'agit de la pression atmosphérique détectée par le capteur de pression barométrique, en *millibars* (**Figure 7-20**). Essayez de souffler sur le Sense HAT (ou de déplacer le curseur de pression vers le haut dans l'émulateur) et cliquez à nouveau sur Exécuter ; cette fois, vous devriez obtenir une valeur plus élevée.

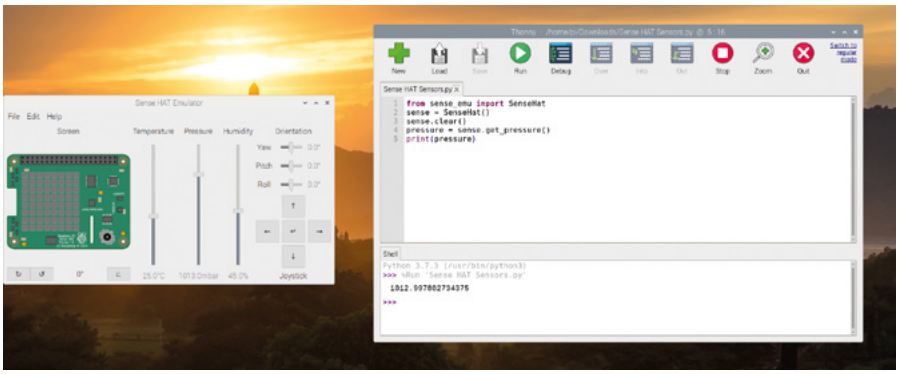


Figure 7-20 : Affichage d'un relevé de pression de la Sense HAT

MODIFICATION DES VALEURS

Si vous utilisez l'émulateur Sense HAT, vous pouvez modifier les valeurs signalées par chacun des capteurs émulés à l'aide des curseurs et boutons correspondants. Essayez de faire glisser le curseur du capteur de pression vers le bas, puis cliquez à nouveau sur Exécuter.

Pour passer au capteur d'humidité, supprimez les deux dernières lignes du code et remplacez-les par :

```
humidité = sense.get_humidity()
print(humidité)
```

Cliquez sur Exécuter et vous verrez un nombre affiché sur le shell Python : il s'agit de l'humidité relative de votre pièce, exprimée en pourcentage. Encore une fois, vous pouvez souffler sur le Sense HAT (ou déplacer le curseur Humidité de l'émulateur vers le haut) pour observer le changement de lecture lorsque vous exécutez à nouveau votre programme (**Figure 7-21**) : votre haleine est étonnamment humide !

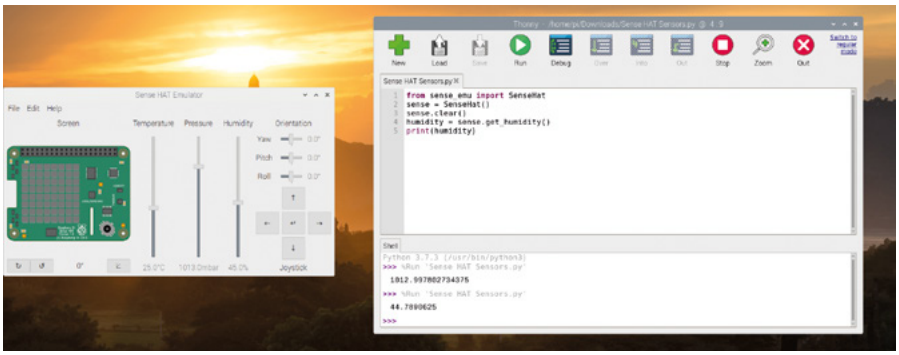


Figure 7-21 : Affichage du relevé du capteur d'humidité

Pour passer au capteur de température, supprimez les deux dernières lignes de votre programme et remplacez-les par :

```
temp = sense.get_temperature()
print(temp)
```

Cliquez sur Exécuter et la température s'affiche en degrés Celsius (**Figure 7-22**). Il se peut toutefois que la lecture de la température de votre pièce ne soit pas tout à fait exacte : Raspberry Pi génère de la chaleur pendant son temps de fonctionnement, ce qui réchauffe également la Sense HAT et ses capteurs.

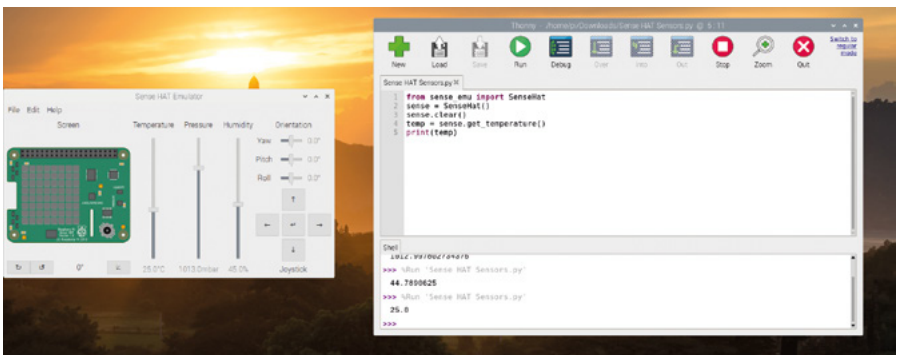


Figure 7-22 : Affichage de la température actuelle

Normalement, la Sense HAT indique la température sur la base d'une lecture du capteur de température intégré au capteur d'humidité ; si vous souhaitez utiliser la lecture du capteur de pression, vous devez utiliser **`sense.get_temperature_from_pressure()`**. Il est également possible de combiner les deux relevés pour obtenir une moyenne, dont le degré de précision est plus élevé qu'en utilisant un seul des deux capteurs. Supprimez les deux dernières lignes de votre programme et saisissez :

```
htemp = sense.get_temperature()
ptemp = sense.get_temperature_from_pressure()
temp = (htemp + ptemp) / 2
print(temp)
```

Cliquez sur Exécuter et vous verrez un nombre affiché sur la console Python (**Figure 7-23**). Cette fois, il est basé sur les relevés des deux capteurs, que vous avez additionnés et divisés par deux (le nombre de relevés) pour obtenir une moyenne. Si vous utilisez l'émulateur, les trois méthodes (humidité, pression et la moyenne des deux) afficheront le même nombre.

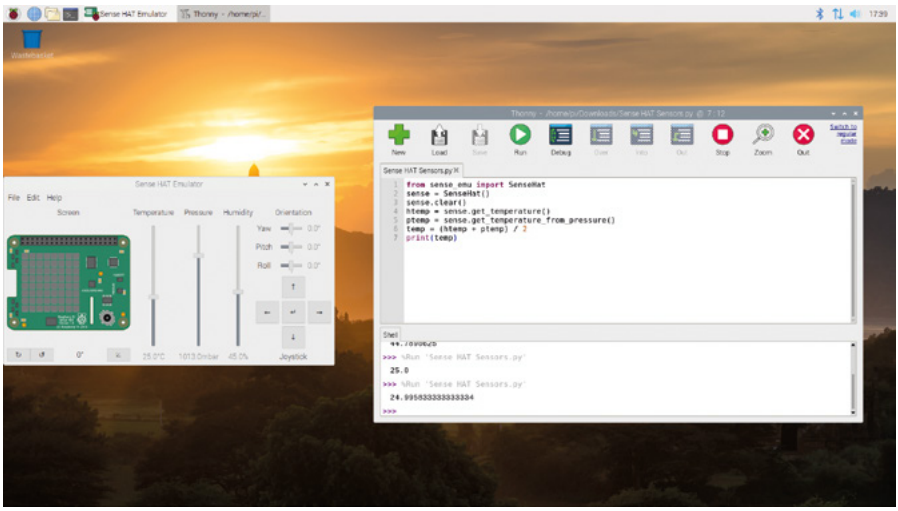


Figure 7-23 : Température basée sur les relevés des deux capteurs

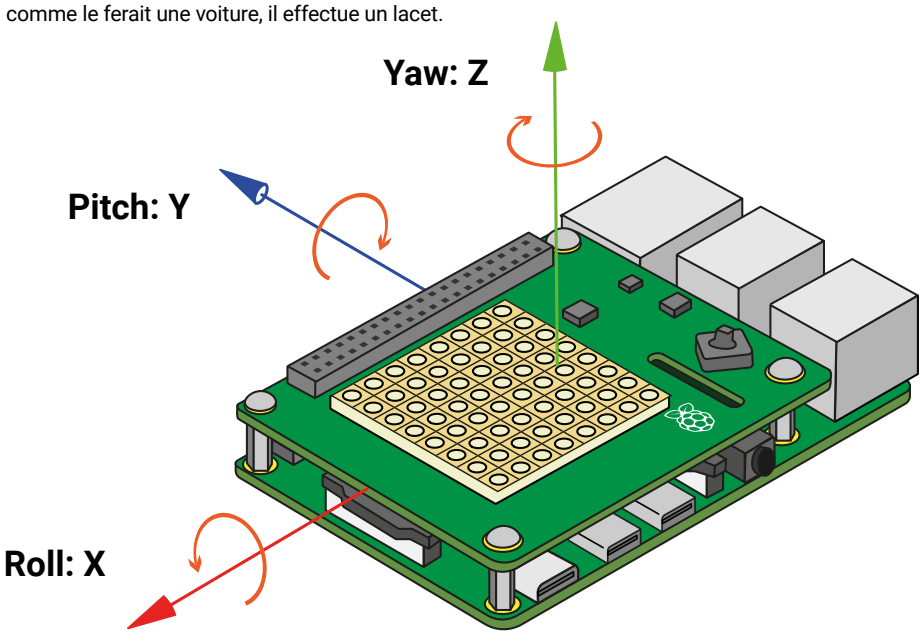
DÉFI : DÉFILEMENT ET BOUCLE

Pouvez-vous modifier votre programme pour afficher la lecture de chacun des capteurs à tour de rôle, puis les faire défiler sur la matrice LED plutôt que de les afficher sur le shell ? Pouvez-vous faire tourner votre programme en boucle, afin qu'il affiche constamment les conditions environnementales actuelles ?

Détection inertielle

Le capteur gyroscopique, l'accéléromètre et le magnétomètre se combinent pour former ce qu'il est convenu d'appeler une *unité de mesure inertielle (UMI)*. Bien que, techniquement parlant, ces capteurs prennent des mesures de l'environnement environnant tout comme les capteurs environnementaux (le magnétomètre, par exemple, mesure l'intensité du champ magnétique), ils sont généralement utilisés pour collecter des données relatives au mouvement de la Sense HAT en soi. L'UMI résulte de la somme de plusieurs capteurs ; certains langages de programmation vous permettent de prendre des lectures de chaque capteur séparément, tandis que d'autres ne vous donneront qu'une lecture combinée.

Pour comprendre exactement ce qu'est l'UMI, il est essentiel de bien comprendre comment s'effectuent les différents mouvements. La Sense HAT, tout comme votre Raspberry Pi auquel elle est rattachée, peut se déplacer le long de trois axes spatiaux : d'un côté à l'autre sur l'axe X ; en avant et en arrière sur l'axe Y ; et de haut en bas sur l'axe Z (**Figure 7-24**). La Sense HAT peut également tourner autour de ces trois axes, mais ces rotations sont désignées par des termes spécifiques : la rotation sur l'axe X est appelée *roulis (roll)*, la rotation sur l'axe Y est dénommée *tangage (pitch)*, alors que toute rotation autour de l'axe Z prend le nom de *lacet (yaw)*. Lorsque vous faites tourner la Sense HAT le long de son axe court, vous ajustez son tangage ; lorsque vous la faites tourner le long de son axe long, il s'agit du roulis ; lorsque vous la faites tourner tout en la maintenant à plat sur la table, ce mouvement s'appelle lacet. Imaginez un avion : quand il décolle, il augmente son tangage pour prendre de l'altitude ; quand il effectue un roulis de victoire, il tourne autour de son axe de roulement ; quand il utilise son gouvernail pour tourner comme le ferait une voiture, il effectue un lacet.



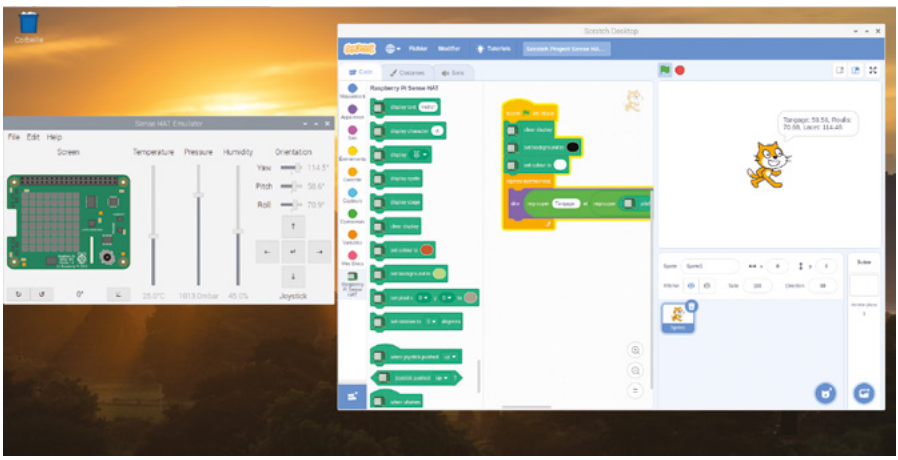
▲ **Figure 7-24 :** Les axes spatiaux de l'UMI de la Sense HAT

Détection inertielle dans Scratch

Démarrez un nouveau programme dans Scratch et ajoutez l'extension Sense HAT Raspberry Pi si vous ne l'avez pas encore chargée. Commencez votre programme comme vous l'avez fait auparavant : faites glisser un bloc Événement quand est cliqué dans votre zone de code, puis faites glisser un bloc clear display au-dessous, suivi des blocs set background to noir et set colour to blanc que vous avez modifiés.

Faites glisser ensuite un bloc répéter indéfiniment en bas de vos blocs existants et ajoutez un bloc dire bonjour !. Pour afficher une lecture pour chacun des trois axes UMI (tangage, roulis et lacet), vous devez ajouter les blocs Opérateur rejoindre en plus des blocs Sense HAT correspondants. N'oubliez pas d'inclure des espaces et des virgules pour faciliter la lecture des résultats.

Cliquez sur le drapeau vert pour lancer votre programme, et essayez de déplacer la Sense HAT le Raspberry Pi dans tous les sens, en faisant attention à ne rien débrancher ! En inclinant la Sense HAT sur ses trois axes, vous verrez que les valeurs de tangage, de roulis et de lacet changent en conséquence (**Figure 7-25**).



▲ **Figure 7-25 :** Affichage des valeurs de tangage, de roulis et de lacet

Détection inertielle dans Python

Créez un nouveau programme dans Thonny et enregistrez-le sous **Mouvements Sense HAT**. Commencez avec les mêmes lignes que de coutume, sans oublier de préciser **`sense_emu`** si vous utilisez l'émulateur :

```
from sense_hat import SenseHat
sense = SenseHat()
sense.clear()
```

Pour utiliser les informations d'UMI afin de déterminer l'orientation actuelle de la Sense HAT sur ses trois axes, saisissez ce qui suit :

```
orientation = sense.get_orientation()
tangage = orientation["pitch"]
roulis = orientation["roll"]
lacet = orientation["yaw"]
print("tangage {0} roulis {1} lacet {2}".format(tangage, roulis,
lacet))
```

Cliquez sur Exécuter pour visualiser les lectures de l'orientation de la Sense HAT réparties sur les trois axes (**Figure 7-26**). Essayez de faire pivoter la Sense HAT et cliquez à nouveau sur Exécuter ; vous devriez voir les valeurs changer pour refléter la nouvelle orientation.

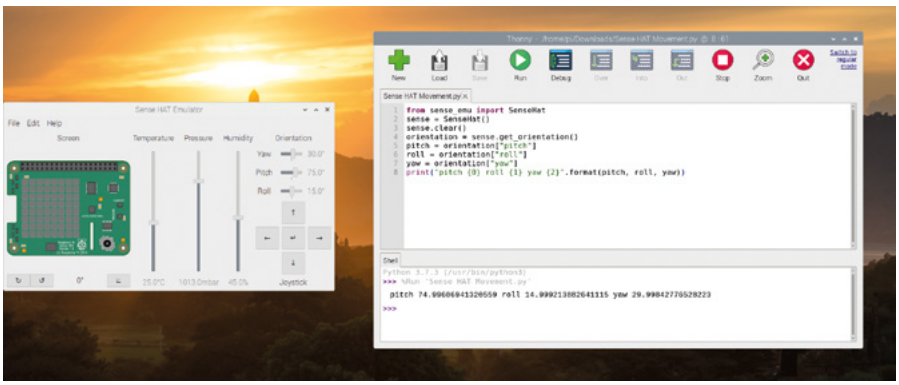


Figure 7-26 : Affichage des valeurs de tangage, de roulis et de lacet de la Sense HAT

Mais l'UMI ne se contente pas de mesurer l'orientation : elle peut également détecter les mouvements. Pour obtenir des relevés de mouvement précis, l'UMI doit être lue fréquemment en boucle : contrairement à l'orientation, un seul relevé ne vous donnera aucune information utile pour détecter un mouvement. Supprimez tout ce qui suit **`sense.clear()`** puis saisissez le code suivant :

```
while True:
    acceleration = sense.get_accelerometer_raw()
    x = acceleration["x"]
    y = acceleration["y"]
    z = acceleration["z"]
```

Vous disposez maintenant de variables contenant les relevés actuels de l'accéléromètre pour les trois axes spatiaux : X (de gauche à droite) ; Y (d'avant en arrière) ; Z (de haut en bas). La lecture des valeurs de l'accéléromètre peut s'avérer compliquée, mais il est possible de la simplifier en arrondissant au nombre entier le plus proche en saisissant :

```
    x = round(x)
    y = round(y)
    z = round(z)
```

Pour finir, imprimez les trois valeurs en saisissant la ligne suivante :

```
    print("x={0}, y={1}, z={2}".format(x, y, z))
```

Cliquez sur Exécuter et vous pourrez visualiser les valeurs de l'accéléromètre affichées dans la zone shell de Python (**Figure 7-27**). Contrairement à votre programme précédent, ces valeurs s'afficheront en continu ; pour les arrêter, cliquez sur le bouton rouge d'arrêt du programme.

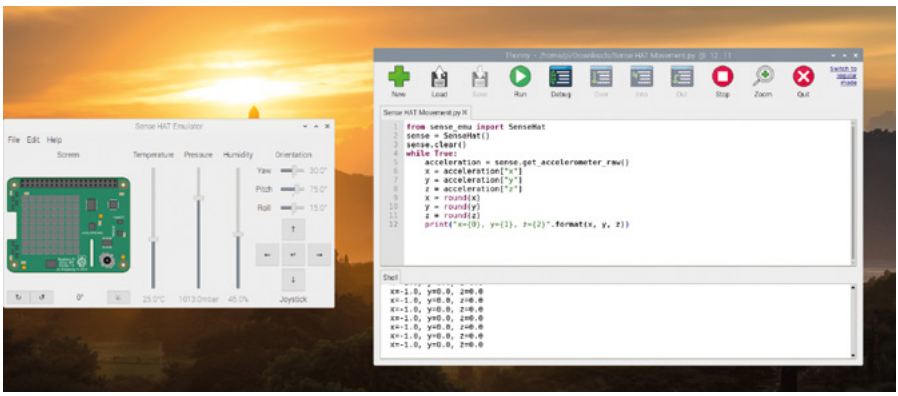


Figure 7-27 : Lectures de l'accéléromètre arrondies au nombre entier le plus proche

Vous avez peut-être remarqué que l'accéléromètre vous indique que l'un des axes (l'axe Z, si votre Raspberry Pi est à plat sur la table) a une valeur d'accélération de 1,0 gravité (1 G), alors que la Sense HAT ne bouge pas. Cela est dû au fait qu'elle détecte l'attraction gravitationnelle de la Terre (la force qui attire la Sense HAT vers le centre de la Terre, et la raison pour laquelle tout ce qui tombe de votre bureau finit inévitablement au sol).

Pendant que votre programme s'exécute, soulevez délicatement la Sense HAT et le Raspberry Pi, et faites-les tourner, en ayant soin de ne débrancher aucun câble ! Avec le réseau de Raspberry Pi et les ports USB pointant vers le sol, vous verrez les valeurs changer, de sorte que l'axe Z indique 0G et l'axe X 1G ; tournez à nouveau pour que les ports HDMI et d'alimentation soient pointés vers le sol et l'axe Y indique 1 G. Si vous faites le contraire et que le port HDMI pointe vers le plafond, l'axe Y indiquera une valeur de -1 G.

En sachant que la gravité de la Terre est d'environ 1 G et, en connaissant les axes spatiaux, vous pouvez déduire quel est le bas et quel est le haut à partir des lectures de l'accéléromètre. Vous pouvez également les utiliser pour détecter des mouvements : essayez de secouer avec précaution la Sense HAT et le Raspberry Pi, et observez les résultats : plus vous secouez, plus l'accélération est importante.

Lorsque vous utilisez la fonction **`sense.get_accelerometer_raw()`**, vous indiquez à la Sense HAT d'éteindre les deux autres capteurs UMI (le capteur gyroscopique et le magnétomètre) et de renvoyer les données émanant exclusivement de l'accéléromètre. Naturellement, vous pouvez faire la même chose avec les autres capteurs.

Cherchez la ligne **`acceleration = sense.get_accelerometer_raw()`** et modifiez-la comme suit :

```
orientation = sense.get_gyroscope_raw()
```

Remplacez le mot **`acceleration`** sur les trois lignes en dessous par **`orientation`**. Cliquez sur Exécuter, et vous verrez l'orientation du Sense HAT sur les trois axes, arrondie au nombre entier le plus proche. Contrairement à la dernière fois où vous avez vérifié l'orientation, cette fois-ci les données proviennent uniquement du gyroscope sans utiliser l'accéléromètre ou le magnétomètre. Cela peut être utile si vous voulez connaître l'orientation d'une Sense HAT en mouvement à l'arrière d'un robot, par exemple, sans que la lecture soit influencée par le mouvement, ou si la Sense HAT se trouve à proximité d'un champ magnétique puissant

Arrêtez votre programme en cliquant sur le bouton rouge d'arrêt. Pour utiliser le magnétomètre, supprimez tout le code de votre programme hormis les quatre premières lignes, puis saisissez ce qui suit sous la ligne **`while True`** :

```
north = sense.get_compass()
print(north)
```

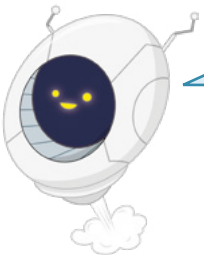
Lancez votre programme et vous verrez la direction du nord magnétique affichée à plusieurs reprises sur la zone du shell Python. Faites tourner la Sense HAT avec précaution et vous verrez le cap varier en fonction de l'orientation de la Sense HAT par rapport au nord : vous avez construit une boussole ! Si vous avez un aimant (un simple aimant du réfrigérateur fera l'affaire), essayez de le déplacer autour du Sense HAT et observez les effets de cette opération sur le magnétomètre.

DÉFI : ROTATION AUTOMATIQUE

En vous appuyant sur ce que vous avez appris sur la matrice LED et les capteurs de l'unité de mesure inertielle, pouvez-vous écrire un programme qui fait tourner une image en fonction de la position de la Sense HAT ?

Contrôle par joystick

En dépit de sa petite taille, le joystick de la Sense HAT, qui se trouve dans le coin inférieur droit, est étonnamment puissant : il peut reconnaître des entrées dans les quatre directions (haut, bas, gauche et droite) et possède également une cinquième entrée à laquelle on accède en l'enfonçant par le haut comme un bouton-poussoir.



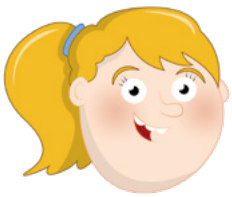
ATTENTION !

Le joystick de la Sense HAT ne doit être utilisé que si vous avez installé les entretoises comme décrit au début de ce chapitre. Si les entretoises ne sont pas installées, le fait d'appuyer sur le joystick risque de plier la carte Sense HAT et endommager le connecteur GPIO de la Sense HAT et du Raspberry Pi.

Contrôle par joystick dans Scratch

Démarrez un nouveau programme dans Scratch avec l'extension Sense HAT Raspberry Pi. Commencez votre programme comme vous l'avez fait auparavant : faites glisser un bloc Événement quand est cliqué dans votre zone de script, puis faites glisser un bloc clear display au-dessous, suivi des blocs set background to noir et set colour to blanc que vous avez modifiés.

Dans Scratch, le joystick de la Sense HAT correspond aux touches de curseur du clavier : pousser le joystick vers le haut équivaut à appuyer sur la touche flèche vers le haut, le pousser vers le bas équivaut à appuyer sur la touche flèche vers le bas, le pousser vers la gauche équivaut à appuyer sur la touche flèche vers la gauche, et le pousser vers la droite équivaut à appuyer sur la touche flèche vers la droite ; la pression sur l'interrupteur à bouton-poussoir équivaut à appuyer sur la touche **ENTRÉE**.



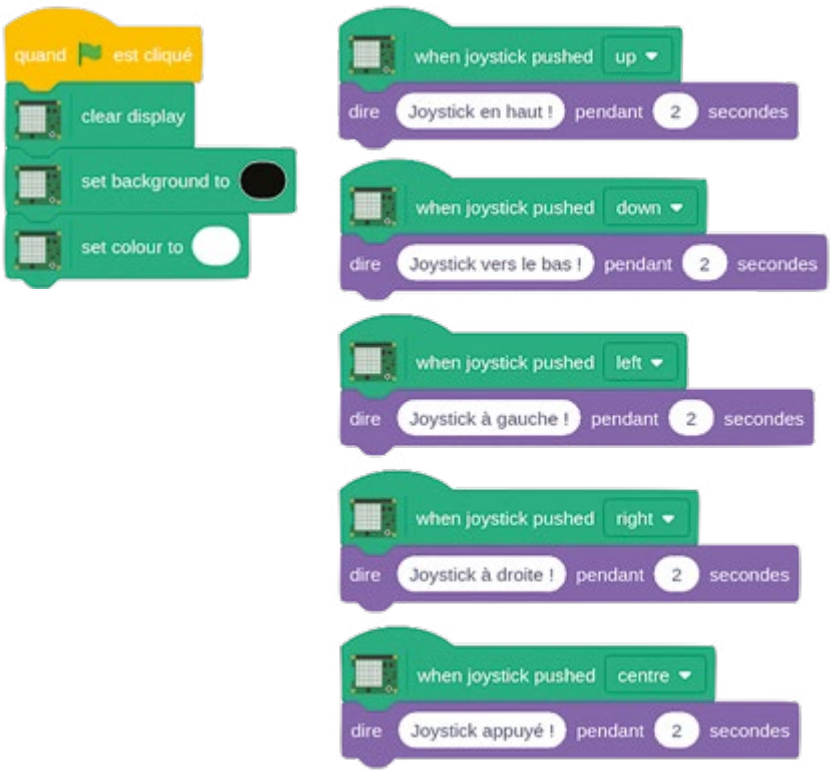
ATTENTION !

Le contrôle par joystick n'est disponible que sur la Sense HAT physique. Lorsque vous utilisez l'émulateur Sense HAT, utilisez plutôt les touches correspondantes de votre clavier pour simuler des pressions de joystick.

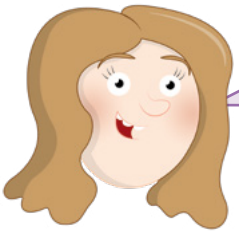
Faites glisser un bloc **when joystick pushed up** sur votre zone de codage. Ensuite, pour lui assigner une activité, faites glisser un bloc **dire Bonjour! pendant 2 secondes** en dessous.

Poussez le joystick vers le haut et vous verrez le chat de Scratch vous accueillir d'une joyeux « Hello ! »

Ensuite, modifiez le bloc **dire Bonjour! pendant 2 secondes** en un bloc **dire Joystick en haut ! pendant 2 secondes**, et continuez à ajouter des blocs Événement et Apparence jusqu'à ce que chacun des cinq modes de pression du joystick soit défini.



Essayez de pousser le joystick dans différentes directions pour voir vos messages apparaître !



DÉFI FINAL

Pouvez-vous contrôler un sprite Scratch sur la scène à l'aide d'un joystick ? Pouvez-vous faire en sorte que si le sprite s'unit à un autre sprite, représentant un objet, les LED de la Sense HAT affichent un message de félicitations ?

Contrôle par joystick dans Python

Créez un nouveau programme dans Thonny et enregistrez-le sous Joystick Sense HAT. Commencez par les trois lignes habituelles qui configurent la Sense HAT et remettez la matrice LED en blanc :

```
from sense_hat import SenseHat
sense = SenseHat()
sense.clear()
```

Ensuite, mettez en place une boucle infinie :

```
while True:
```

Ensuite, indiquez à Python de capter les signaux du joystick Sense HAT avec la ligne suivante, qui sera automatiquement indentée par Thonny :

```
    for event in sense.stick.get_events():
```

Enfin, ajoutez la ligne suivante (également indentée par Thonny) pour définir une réaction à chaque pression du joystick :

```
        print(event.direction, event.action)
```

Cliquez sur Exécuter et essayez de pointer le joystick dans plusieurs directions. Vous verrez la direction que vous avez choisie affichée dans la zone du shell Python : haut, bas, gauche, droite et milieu si vous avez utilisé l'interrupteur à bouton-poussoir du joystick.

Vous verrez également que deux événements se présentent à vous chaque fois que vous appuyez sur le joystick : un événement, **`pressed`** quand vous pointez vers une direction pour la première fois ; l'autre événement, **`released`** quand le joystick est ramené à la position initiale. Vous pouvez exploiter ce facteur dans vos programmes : pensez à un personnage dans un jeu, qui pourrait se mettre en mouvement en pressant le joystick dans une direction, puis s'arrêter dès que le joystick est relâché.

Vous pouvez également utiliser le joystick pour déclencher des fonctions, plutôt que de vous limiter à utiliser une boucle for. Supprimez tout ce qui se trouve au-dessous de **`sense.clear()`** et saisissez :

```
def rouge():
    sense.clear(255, 0, 0)

def bleu():
    sense.clear(0, 0, 255)

def vert():
    sense.clear(0, 255, 0)

def jaune():
    sense.clear(255, 255, 0)
```

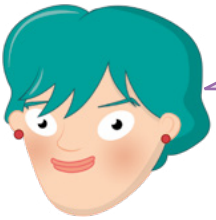
Ces fonctions définissent l'entière matrice LED de la Sense HAT d'une seule couleur : rouge, bleu, vert ou jaune, ce qui va énormément faciliter le fonctionnement de votre programme ! Pour les déclencher effectivement, vous devez indiquer à Python quelle fonction correspond à chaque pression du joystick. Saisissez les lignes ci-dessous :

```
sense.stick.direction_up = rouge
sense.stick.direction_down = bleu
sense.stick.direction_left = vert
sense.stick.direction_right = jaune
sense.stick.direction_middle = sense.clear
```

Enfin, le programme a besoin d'une boucle infinie ou boucle *principale* pour continuer à s'exécuter et, par conséquent, pour continuer à répondre aux pressions du joystick, plutôt que de parcourir le code une seule fois avant de s'arrêter. Saisissez les deux lignes suivantes :

```
while True:
    pass
```

Cliquez sur Exécuter, et essayez de manœuvrer le joystick : vous verrez les LED s'illuminer d'une belle couleur brillante ! Pour éteindre les LED, il suffit d'appuyer sur le joystick comme sur un bouton-poussoir : la direction **`middle`** est définie de sorte à éteindre les LED via la fonction **`sense.clear()`**. Félicitations : vous pouvez capter les signaux transmis par le joystick !



DÉFI FINAL

Pouvez-vous mettre à profit les connaissances que vous avez acquises pour dessiner une image à l'écran, puis la faire tourner dans n'importe quelle direction à l'aide du joystick ? Pouvez-vous faire en sorte que le bouton du milieu vous fasse passer d'une image à l'autre ?

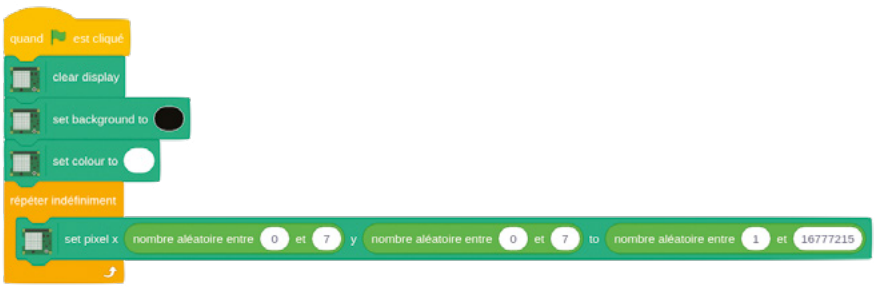
Projet Scratch : Scintillement Sense HAT

Maintenant que vous savez comment fonctionne la Sense HAT, il est temps de mettre à profit tout ce que vous avez appris pour créer un scintillement en fonction de la chaleur, un dispositif qui fonctionne mieux lorsqu'il fait froid et qui ralentit progressivement à mesure que la température s'élève.

Démarrez un nouveau projet dans Scratch et ajoutez l'extension Sense HAT Raspberry Pi si vous ne l'avez pas encore chargée. Comme toujours, commencez par quatre blocs : quand est cliqué, clear display, set background to noir, et set colour to blanc, sans oublier de changer les couleurs par rapport à leurs paramètres par défaut.

Commencez par créer un scintillement tout simple, mais artistique. Faites glisser un bloc répéter indéfiniment sur la zone de codage, puis placez un bloc set pixel x 0 y 0 to colour en son centre. Plutôt que d'utiliser des nombres définis, remplissez chacune des sections x, y et couleur de ce bloc avec un bloc nombre aléatoire entre 1 et 10 Opérateurs.

Les valeurs de 1 à 10 ne sont pas très utiles ici, vous devez donc effectuer quelques modifications. Les deux premiers chiffres du bloc set pixel sont les coordonnées X et Y du pixel sur la matrice LED, ce qui signifie qu'ils doivent être compris entre 0 et 7 : modifiez donc les deux premiers blocs pour obtenir nombre aléatoire entre 0 et 7 . La section suivante concerne la couleur souhaitée pour le pixel. Lorsque vous utilisez le sélecteur de couleur, la couleur que vous choisissez s'affiche directement dans la zone de script ; à l'intérieur, cependant, les couleurs sont représentées par un numéro, que vous pouvez utiliser tel quel. Modifiez le dernier bloc aléatoire pour obtenir nombre aléatoire entre 0 et 16 777 215.



Cliquez sur le drapeau vert et observez les LED de la Sense HAT s'illuminer de couleurs aléatoires (**Figure 7-28**). Félicitations : vous avez créé un scintillement électronique !

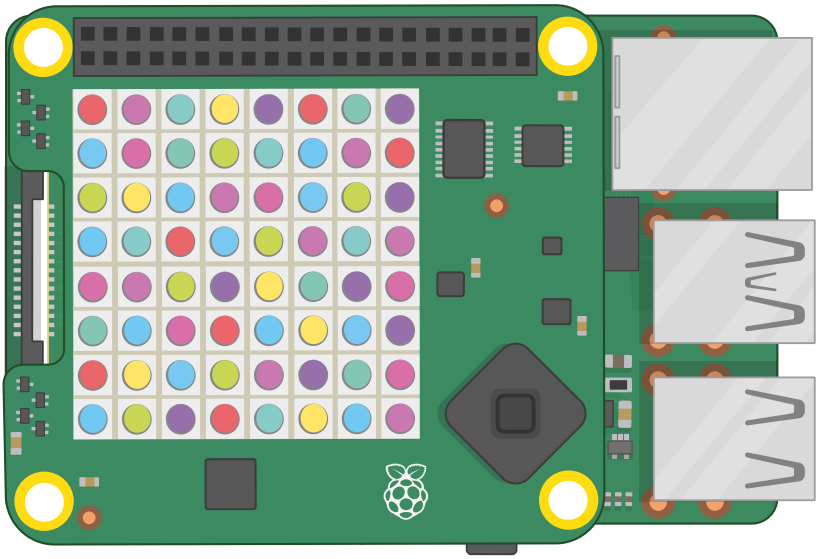
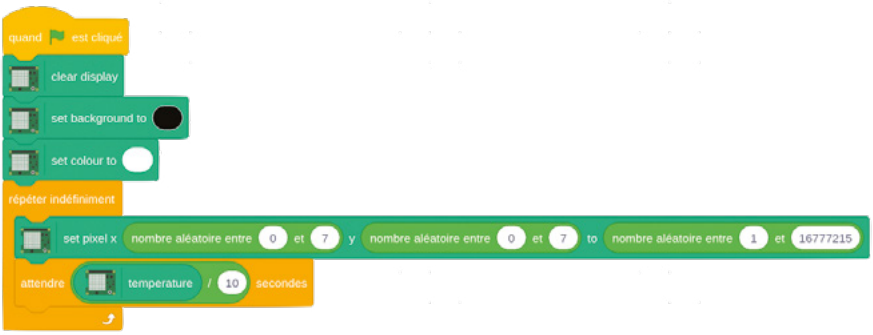


Figure 7-28 : Éclairage des pixels en couleurs aléatoires

Le système de scintillement n'est pas très interactif. Pour changer cela, commencez par faire glisser un bloc **attendre 1 seconde** sous le bloc **set pixel** mais à l'intérieur du bloc **répéter indéfiniment**. Faites glisser un bloc Opérateurs **● / ●** sur le 1, puis saisissez 10 dans le deuxième espace. Pour finir, faites glisser un bloc **temperature** sur le premier espace du bloc diviseur Opérateur.



Cliquez sur le drapeau vert et vous remarquerez (sauf si vous vivez dans un endroit très froid) que le scintillement est considérablement plus lent qu'auparavant. Cela est dû au fait que vous avez créé un délai en fonction de la température : le programme attend un nombre de secondes équivalent à *la température actuelle divisée par 10* avant de recommencer chaque boucle. Si la température de la pièce est de 20° C, le programme attendra 2 secondes avant de recommencer une boucle ; si la température est de 10° C, il attendra 1 seconde ; si elle est inférieure à 10 °C, il attendra moins d'une seconde.

Si votre Sense HAT indique une température négative (inférieure à 0° C, le point de congélation de l'eau), elle tentera d'attendre moins de 0 seconde, ce qui est impossible faute de pouvoir voyager dans le temps : l'effet sera donc le même que si elle attendait 0 seconde. Félicitations : vous pouvez désormais intégrer les différentes fonctionnalités du Sense HAT dans vos propres programmes !

Projet Python : Tricordeur Sense HAT

Maintenant que vous savez comment fonctionne la Sense HAT, il est temps de mettre à profit tout ce que vous avez appris pour construire un tricordeur, un appareil que les fans d'une certaine franchise de science-fiction connaissent à la perfection et qui peut renvoyer des données émanant de plusieurs senseurs intégrés.

Créez un nouveau projet dans Thonny et enregistrez-le sous **Tricordeur**, puis commencez avec les mêmes lignes dont vous avez besoin pour créer un programme Sense HAT :

```
from sense_hat import SenseHat
sense = SenseHat()
sense.clear()
```

Ensuite, vous devez définir les fonctions de chacun des différents capteurs de la Sense HAT. Commencez par l'unité de mesure inertielle en saisissant :

```
def orientation():
    orientation = sense.get_orientation()
    tangage = orientation["pitch"]
    roulis = orientation["roll"]
    lacet = orientation["yaw"]
```

Les résultats émanant de senseurs vont défiler sur l'écran LED, raison pour laquelle il convient de les arrondir pour ne pas avoir à voir défiler des dizaines de décimales. Il convient de les arrondir à une décimale plutôt qu'à un nombre entier en saisissant ce qui suit :

```
    tangage = round(tangage, 1)
    roulis = round(roulis, 1)
    lacet = round(lacet, 1)
```

Enfin, vous devez indiquer à Python de faire défiler les résultats sur les LED, de sorte que le tricordeur fonctionne comme un appareil portatif sans avoir besoin d'être connecté à un moniteur ou à un écran TV :

```
        sense.show_message("Tangage {0}, Roulis {1}, Lacet {2}".
format(tangage, roulis, lacet))
```

Maintenant que vous disposez d'une fonction complète pour lire et afficher l'orientation de l'UMI, vous devez créer des fonctions similaires pour chacun des autres capteurs. Commencez par le capteur de température :

```
def température():
    temp = sense.get_temperature()
    temp = tour(temp, 1)
    sense.show_message("Température: %s degrès Celsius" % temp)
```

Regardez attentivement la ligne qui imprime le résultat sur les LED : la fonction **%s** est désignée par le terme de paramètre fictif, qui est par la suite remplacé par le contenu de la variable **temp**. Vous pouvez ainsi formater la sortie en définissant une étiquette (« Température ») et une unité de mesure (« degrés Celsius »), ce qui rend votre programme beaucoup plus convivial.

Ensuite, définissez une fonction pour le capteur d'humidité :

```
def humidité():
    humidité = sense.get_humidity()
    humidité = round(humidité, 1)
    sense.show_message("Humidité: %s pourcent" % humidité)
```

Ensuite, le capteur de pression :

```
def pression():
    pression = sense.get_pressure()
    pression = round(pression, 1)
    sense.show_message("Pression: %s millibars" % pressure)
```

Et pour finir, la lecture de la boussole du magnétomètre :

```
def boussole():
    for i in range(0, 10):
        nord = sense.get_compass()
    nord = round(nord, 1)
    sense.show_message("Nord: %s degrés" % nord)
```

La boucle courte **for** pour cette fonction prend en charge dix lectures du magnétomètre pour s'assurer de disposer de suffisamment de données pour fournir un résultat précis. Si vous constatez que la valeur déclarée ne cesse de changer, essayez de l'étendre à 20, 30, voire 100 boucles pour améliorer encore la précision.

Votre programme comporte maintenant cinq fonctions, chacune d'entre elles associée à la lecture d'un des capteurs de la Sense HAT et les faisant défiler sur les LED. Il faut cependant pouvoir choisir le capteur que l'on veut utiliser, ce qui est une tâche idéale pour le joystick.

Saisissez donc :

```
sense.stick.direction_up = orientation
sense.stick.direction_right = température
sense.stick.direction_down = boussole
sense.stick.direction_left = humidité
sense.stick.direction_middle = pression
```

Ces lignes attribuent un capteur à chacune des cinq directions possibles sur le joystick : vers le haut, le capteur d'orientation ; vers le bas, le magnétomètre ; vers la gauche, le capteur d'humidité ; vers la droite, le capteur de température ; et en appuyant sur le bouton du milieu, le capteur de pression.

Pour finir, vous avez besoin de définir une boucle principale pour que le programme continue à percevoir les pressions du joystick et ne s'arrête pas immédiatement. Tout en bas de votre programme, saisissez :

```
while True:
    pass
```

Cliquez sur Exécuter, et manœuvrez le joystick afin de prendre une mesure à partir d'un des capteurs (**Figure 7-29**). Lorsque le résultat finit de défiler, appuyez sur une autre direction. Félicitations : vous avez construit un tricordeur portable qui ferait la fierté de la Fédération des planètes unies !

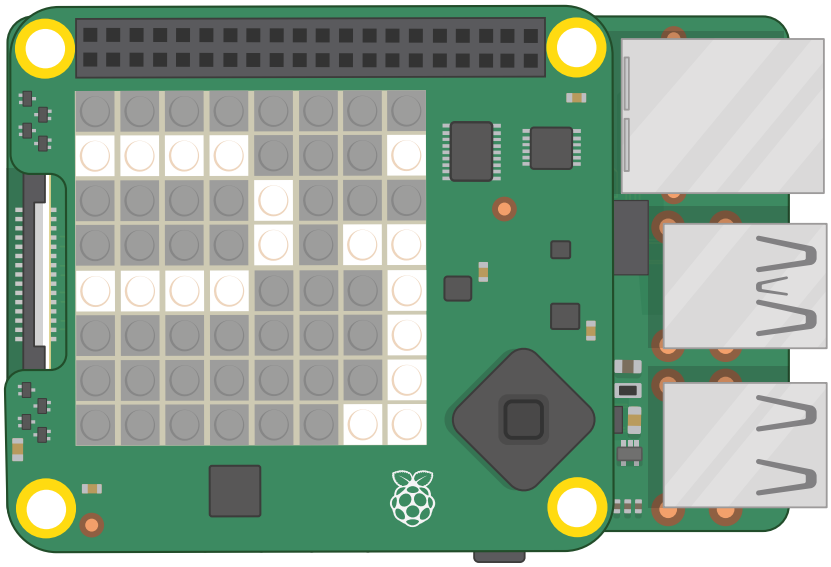


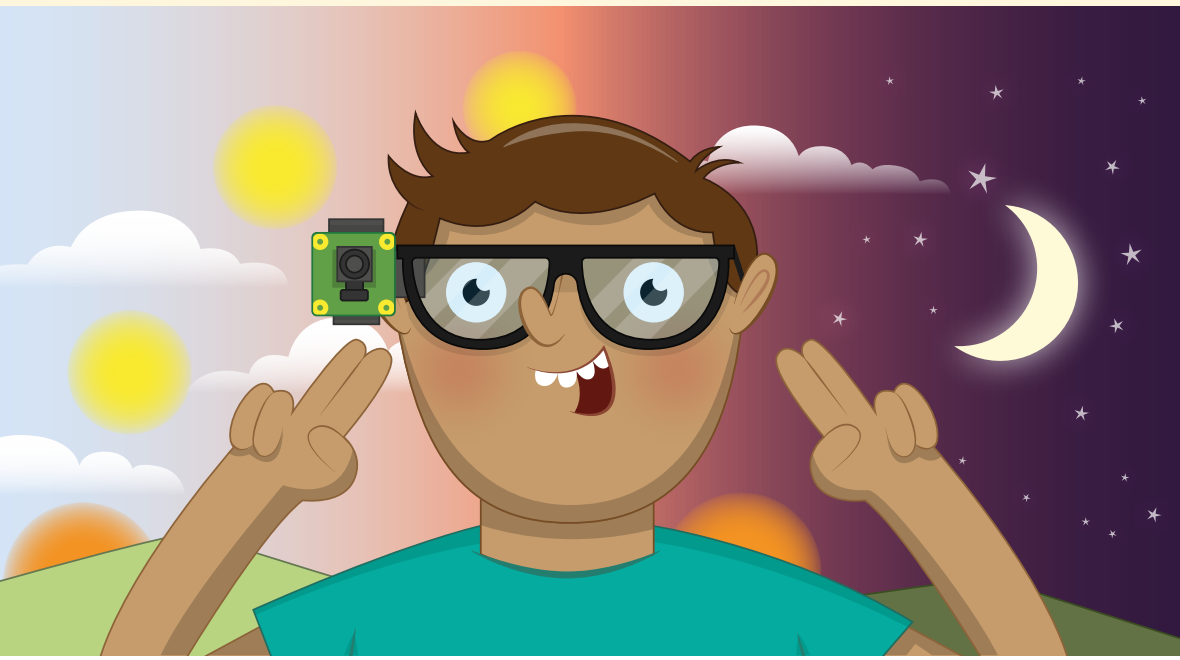
Figure 7-29 : Chaque lecture défile sur l'écran

Pour d'autres projets Sense HAT, cliquez sur les liens figurant dans l'**Annexe D, Lecture complémentaire**.

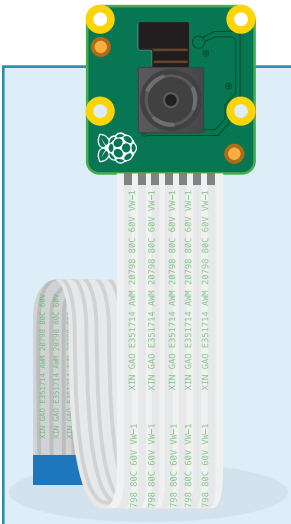
Chapitre 8

Raspberry Pi Camera Module

En connectant un Camera Module (module de caméra) ou une HQ Camera à votre Raspberry Pi, vous pouvez prendre des photos haute résolution et filmer des vidéos, et créer d'excellents projets de vision par ordinateur



Si vous avez toujours rêvé de fabriquer un objet qui vous permet de voir (désigné dans le monde de la robotique sous le nom de *vision par ordinateur*), le Camera Module (module caméra) optionnel de Raspberry Pi, ou la nouvelle High Quality Camera (caméra de haute qualité), constitue un excellent point de départ. Le Camera Module/HQ Camera est un petit circuit imprimé carré doté d'une limande qui se connecte au port CSI (Camera Serial Interface) de votre Raspberry Pi (non disponible sur le Raspberry Pi 400) et fournit des images fixes haute résolution et des signaux vidéo animés qui peuvent être utilisés tels quels ou intégrés dans vos programmes.



TYPES DE CAMERA !

Trois types de caméra Raspberry Pi sont disponibles : le Camera Module standard, la version « NoIR » et la High Quality (HQ) Camera. Si vous voulez prendre des photos et des vidéos normales dans des environnements bien éclairés, le Camera Module standard est celui à utiliser ; si vous voulez utiliser des objectifs spéciaux et que vous recherchez la meilleure qualité d'image, utilisez la HQ Camera. Le Camera Module NoIR (appelé ainsi parce qu'il ne possède pas de filtre infrarouge, ou IR) est conçu pour être utilisé avec des sources de lumière infrarouge afin de prendre des photos et des vidéos dans l'obscurité totale. Si vous construisez une caméra nichoir, une caméra de sécurité ou tout autre projet impliquant la vision nocturne, il vous faut la version NoIR, sans oublier d'acheter une source de lumière infrarouge en même temps !

Les Camera Modules standard et NoIR de Raspberry Pi sont basés sur un capteur d'images Sony IMX219. Il s'agit d'un *Capteur de 8 mégapixels*, qui est en mesure de prendre des photos comportant jusqu'à 8 millions de pixels, c'est-à-dire des images atteignant jusqu'à 3 280 pixels de large sur 2 464 de haut. En plus des images fixes, le Camera Module peut capturer des séquences vidéo en résolution Full HD à une cadence de 30 images par seconde (30 ips). Pour obtenir des mouvements plus fluides ou pour créer un effet de ralenti, la caméra peut être réglée pour capturer des images à une cadence plus élevée tout en diminuant la résolution : 60 ips pour les vidéos en 720p, et jusqu'à 90 ips pour les vidéos en 480p (VGA).

La HQ Camera utilise un capteur Sony IMX477 de 12,3 mégapixels, qui est également plus imposant que celui des Camera Modules standard et NoIR, ce qui signifie qu'il est en mesure de capter plus de lumière et d'obtenir des images de meilleure qualité. Cependant, contrairement aux Camera Modules, la HQ Camera ne comporte pas d'objectif, sans lequel elle ne peut prendre aucune photo ou vidéo. Vous pouvez utiliser n'importe quel objectif avec une monture C ou CS ; d'autres montures peuvent être utilisées à l'aide d'un adaptateur C ou CS approprié. Pour plus de détails sur la fixation de l'objectif, consultez l'**Annexe F : High Quality Caméra**.

RASPBERRY PI 400

Malheureusement, les Raspberry Pi Camera Modules ne sont pas compatibles avec le modèle Raspberry Pi 400. En alternative, vous pouvez utiliser des webcams USB, mais vous ne pourrez pas utiliser les outils logiciels spécifiques au Raspberry Pi Camera Module inclus dans Raspberry Pi OS.

Installation de la caméra

Comme tout autre matériel complémentaire, le Camera Module ou la HQ Camera ne doivent être connectés ou déconnectés du Raspberry Pi que lorsque l'appareil est hors tension et le câble d'alimentation débranché. Si votre Raspberry Pi est sous tension, sélectionnez « Shutdown » dans le menu framboise, attendez qu'il s'éteigne et débranchez-le.

Dans la plupart des cas, la limande fournie sera déjà connectée au Camera Module ou à HQ Camera. Si ce n'est pas le cas, retournez votre carte caméra pour que le capteur soit placé vers le bas et cherchez un connecteur en plastique plat. Saisissez du bout des doigts les bords qui dépassent et tirez délicatement vers vous jusqu'à ce que le connecteur ressorte à moitié. Faites glisser la limande, avec les bords argentés vers le bas et le plastique bleu vers le haut, sous le rabat que vous venez de tirer, puis remettez le rabat en place en le poussant doucement avec un clic (**Figure 8-1**) ; vous pouvez utiliser n'importe quelle extrémité du câble. Si le câble est installé correctement, il sera droit et ne se détache pas en tirant doucement ; dans le cas contraire, tirez sur le rabat et réessayez.

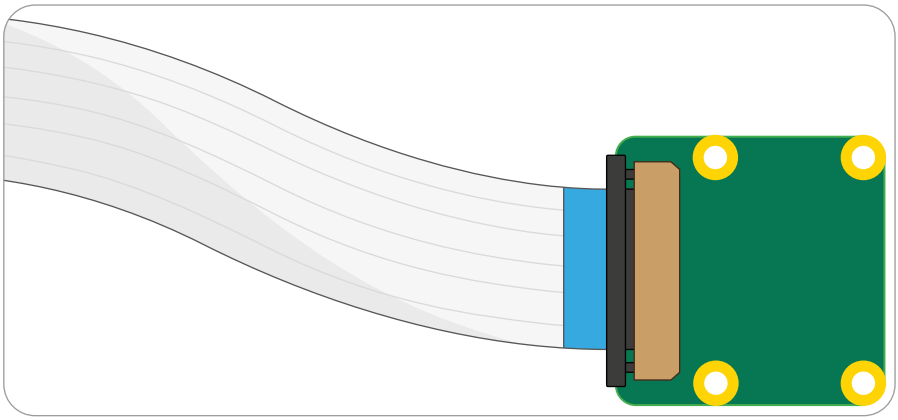


Figure 8-1 : Connexion de la limande au Camera Module

Installez l'autre extrémité du câble de la même façon. Cherchez le port de la caméra (ou CSI) sur le Raspberry Pi et soulevez délicatement le rabat. Si votre Raspberry Pi est installé dans un boîtier, il vous sera peut-être plus simple de le retirer d'abord. En positionnant le Raspberry Pi de manière à ce que le port HDMI se trouve face à vous, connectez la limande de sorte à ce que les bords argentés soient à votre gauche et le plastique bleu à votre droite (**Figure 8-2**), puis remettez doucement le rabat en place. Si le câble est installé correctement, il sera droit et ne se détachera pas si vous tirez dessus doucement ; dans le cas contraire, tirez sur le rabat et réessayez.

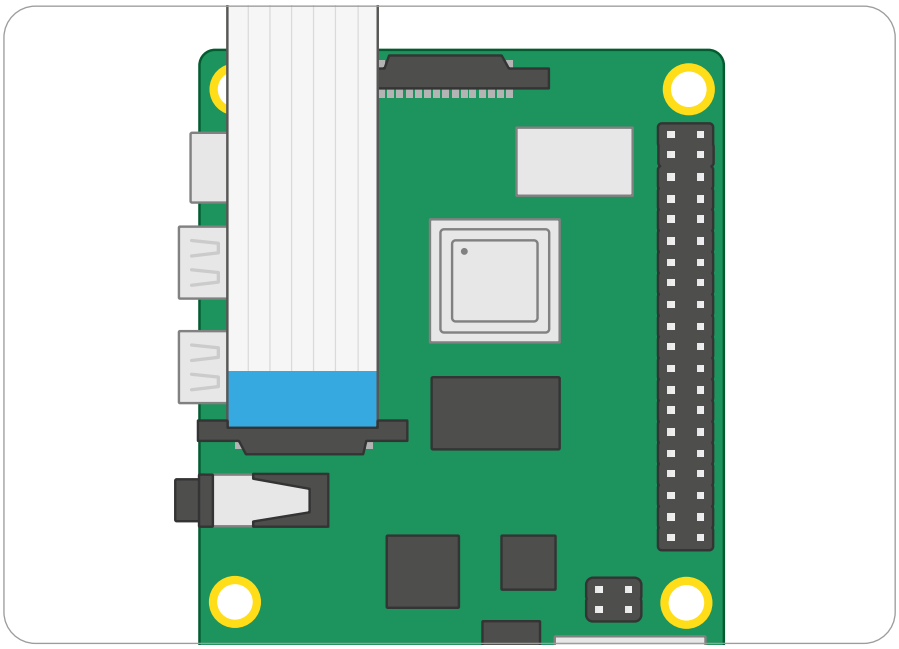


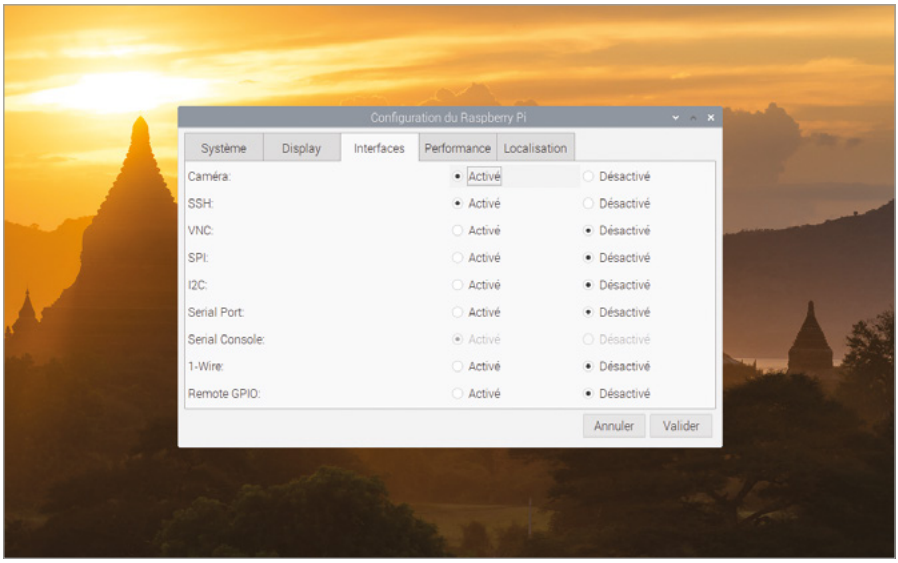
Figure 8-2 : Connexion de la limande au port caméra/CSI du Raspberry Pi

Le Camera Module est livré avec un couvercle en plastique bleu pour protéger l'objectif des rayures pendant la fabrication, l'expédition et l'installation. Retirez-le délicatement de l'objectif en soulevant la languette et votre appareil est prêt à l'emploi.

RÉGLAGE DE LA MISE AU POINT

Le Camera Module est généralement doté d'une petite molette en plastique permettant de régler la mise au point de l'objectif. Généralement, la mise au point d'usine est parfaite, mais si vous voulez photographier des objets de très près, vous pouvez régler la mise au point manuellement en tournant délicatement la molette sur l'objectif. Pour la mise au point de la HQ Camera, consulter l'**Annexe F**.

Remettez le Raspberry Pi sous tension et chargez Raspberry Pi OS. Avant de pouvoir utiliser la caméra, vous devez indiquer au Raspberry Pi qu'elle est connectée : ouvrez le menu de l'icône de framboise, choisissez la catégorie Préférences et cliquez sur Configuration du Raspberry Pi. Une fois l'outil chargé, cliquez sur l'onglet Interfaces, cherchez Caméra dans la liste et cliquez sur le bouton rond à gauche de « Activé » pour l'activer (**Figure 8-3**, au verso). Cliquez sur OK, et l'outil vous demandera de redémarrer votre Raspberry Pi. Votre appareil photo est désormais prêt à l'emploi !

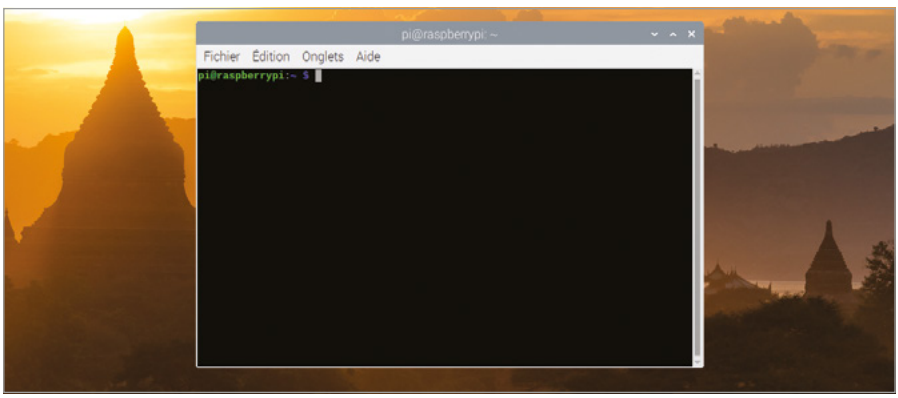


▲ **Figure 8-3 :** Vous devez activer la caméra dans la Configuration du Raspberry Pi

Tester la caméra

Pour confirmer que votre Camera Module ou la HQ Camera sont correctement installés et que vous avez activé l'interface dans l'outil de configuration Raspberry Pi, vous pouvez utiliser l'outil **raspistill**. À l'égal que l'outil **raspivid** pour les vidéos, cet outil est conçu pour capturer des images à partir de la caméra en utilisant l'*interface en ligne de commande (CLI)* du Raspberry Pi.

Contrairement aux programmes que vous avez utilisés jusqu'à présent, raspistill ne figure pas dans le menu. Cliquez sur l'icône framboise pour charger le menu, sélectionnez la catégorie Accessoires et cliquez sur LXTerminal. Une fenêtre noire avec des inscriptions en vert et bleu s'affiche (**Figure 8-4**) : il s'agit du *terminal*, qui vous permet d'accéder à l'interface en ligne de commande.



▲ **Figure 8-4 :** Ouvrir une fenêtre de Terminal pour entrer des commandes

Pour tester la caméra, saisissez les informations suivantes dans le Terminal :

```
raspistill -o test.jpg
```

Dès que vous appuyez sur la touche **ENTRÉE**, s'affiche à l'écran ce que la voit la caméra (**Figure 8-5**). Il s'agit d'un *aperçu en direct* qui restera visible pendant 5 secondes, sauf indications contraire de raspistill. Une fois ces 5 secondes écoulées, l'appareil photo prendra une seule photo et l'enregistrera dans votre dossier personnel sous le nom **test.jpg**. Si vous souhaitez en capturer une autre, saisissez la même commande une seconde fois, en ayant soin de modifier le nom du fichier de sortie, après le **`-o`** pour ne pas écraser votre première photo !

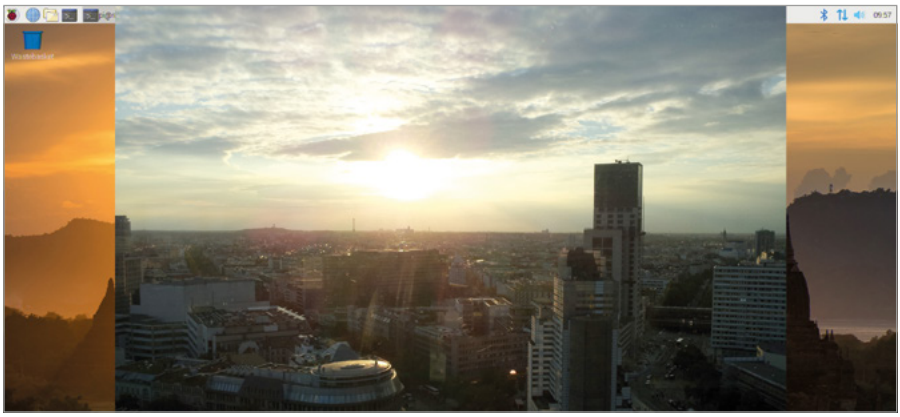


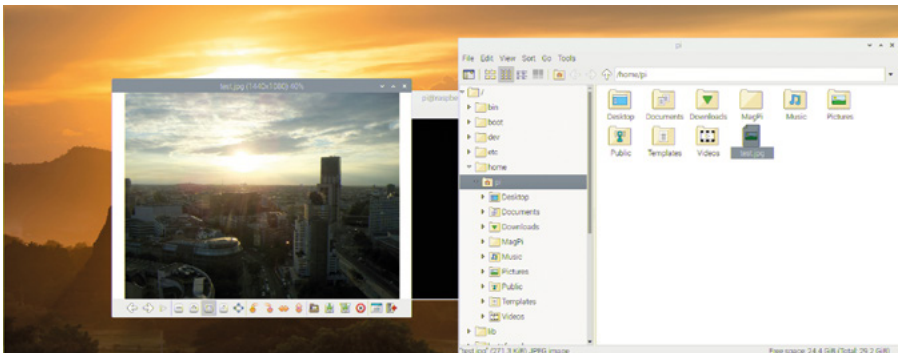
Figure 8-5 : L'aperçu en direct de la caméra

Si l'aperçu en direct était à l'envers, vous devez signaler à raspistill que la caméra est retournée. Le Camera Module est conçu de sorte que la limande sorte par le bord inférieur ; si elle sort par les côtés ou par le haut, comme c'est le cas avec certains accessoires de montage de caméra tiers, vous pouvez faire pivoter l'image de 90, 180 ou 270 degrés en utilisant le commutateur **`-rot`**. Pour une caméra dont le câble sort par le haut, il suffit d'utiliser la commande suivante :

```
raspistill -rot 180 -o test.jpg
```

Si la limande sort par le bord droit, utilisez une valeur de rotation de 90 degrés ; si elle sort par le bord gauche, définissez une valeur de 270 degrés. Si votre prise de vue originale était dans le mauvais angle, corrigez l'angle à l'aide du commutateur **`-rot`**.

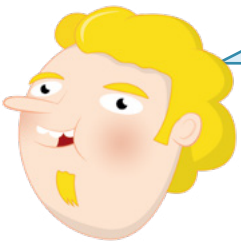
Pour visualiser votre photo, ouvrez le gestionnaire de fichiers de la catégorie Accessoires du menu framboise : l'image que vous avez prise, appelée **test.jpg**, se trouve dans votre dossier **home/pi**. Cherchez-le dans la liste des fichiers, puis double-cliquez dessus pour le charger dans un visionneur d'images (**Figure 8-6**). Vous pouvez envoyer l'image par e-mail en tant que pièce jointe, la télécharger sur des sites web via le navigateur ou la faire glisser sur un périphérique de stockage externe.



▲ **Figure 8-6 :** Ouverture de l'image capturée

Présentation de picamera

La manière la plus simple de contrôler le Camera Module ou la HQ Camera est d'utiliser Python, via la bibliothèque picamera très pratique. Cette fonction permet un contrôle total de la caméra, notamment la prévisualisation, la capture d'images et de vidéos, et vous permet de les intégrer dans vos propres programmes, éventuellement en les combinant avec des programmes qui utilisent le module GPIO via à la bibliothèque GPIO Zero !



PROGRAMMATION PYTHON

Les projets présentés dans ce chapitre supposent que vous possédez une certaine expérience avec le langage de programmation Python, l'IDE Thonny et les broches GPIO du Raspberry Pi. Si vous ne l'avez pas encore fait, lisez le **Chapitre 5, Programmation avec Scratch** ou le **Chapitre 6, Informatique physique avec Scratch et Python** d'abord !

Fermez le Terminal, s'il est toujours ouvert, en cliquant sur le X en haut à droite de la fenêtre, puis chargez Thonny depuis la catégorie Programmation du menu framboise. Enregistrez votre nouveau projet sous **Caméra** puis commencez à importer les bibliothèques dont votre programme a besoin en saisissant ce qui suit dans la zone de script :

```
from picamera import PiCamera
from time import sleep
camera = PiCamera()
```

La dernière ligne vous permet de commander le Camera Module ou la HQ Camera en utilisant la fonction **camera**. Saisissez d'abord :

```
camera.start_preview()
sleep(10)
camera.stop_preview()
```

Cliquez sur Run, et votre bureau disparaîtra, remplacé par un aperçu plein écran de tout ce que voit votre caméra (**Figure 8-7**). Déplacez-la ou agitez la main devant l'objectif et vous verrez l'image à l'écran changer en conséquence. Au bout de 10 secondes, l'aperçu se ferme et votre programme se termine mais, contrairement à l'aperçu de raspistill, aucune image ne sera enregistrée.

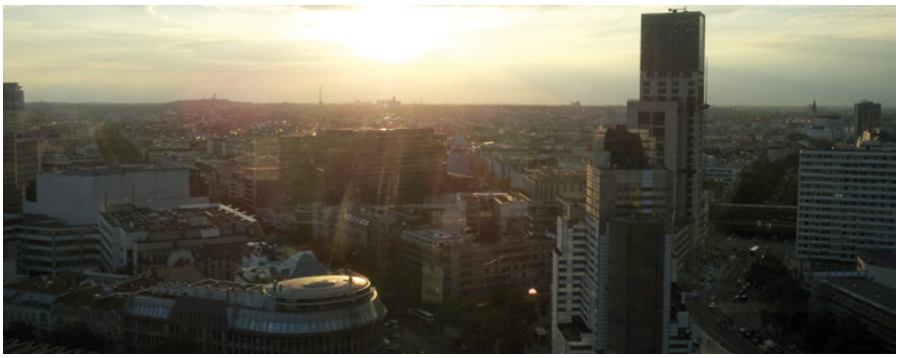


Figure 8-7 : Un aperçu en direct et en plein écran de la vue de la caméra

Si votre prévisualisation était dans le mauvais sens, vous pouvez faire pivoter l'image pour la remettre dans le bon sens. Au-dessous de la ligne **camera = PiCamera()**, saisissez :

```
camera.rotation = 180
```

Si l'aperçu était à l'envers, cette ligne permettra de le remettre à l'endroit. Comme raspistil, la fonction **camera.rotation** vous permet de faire pivoter l'image de 90, 180 ou 270 degrés, selon que le câble sort du côté droit, supérieur ou gauche du Camera Module. N'oubliez pas d'utiliser **camera.rotation** au début de tout programme que vous écrivez, pour éviter de capturer des images ou des vidéos à l'envers !

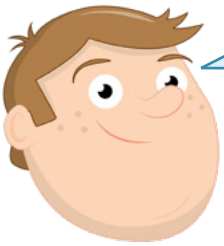
Capture d'images fixes

Pour capturer une image, plutôt que de simplement montrer un aperçu en direct, votre programme doit être modifié. Commencez par réduire le délai de l'aperçu : cherchez la ligne **`sleep(10)`** et modifiez-la comme suit :

```
sleep(5)
```

Au-dessous de cette ligne, ajoutez :

```
camera.capture('/home/pi/Desktop/image.jpg')
```



TEMPS D'AJUSTEMENT

Lorsque la caméra est en mode de prévisualisation, elle analyse la vidéo afin de déterminer si les paramètres doivent être ajustés pour obtenir la meilleure qualité possible. Vous le remarquerez si vous êtes dans un environnement très sombre ou très lumineux, l'aperçu étant d'abord impossible à voir pour devenant rapidement plus clair. Pour donner à la caméra le temps de s'adapter, ajoutez toujours une période de prévisualisation d'au moins 2 secondes à votre programme avant de capturer une image.

La fonction **`camera.capture`** indique à Python d'enregistrer une image fixe, spécifiant le nom souhaité de l'image, et dans quel dossier elle doit être enregistrée. Dans cet exemple, vous l'enregistrez sur le bureau : il se trouve juste au-dessous de la Corbeille. Si la fenêtre de Thonny vous gêne, il vous suffit de cliquer et de faire glisser la barre de titre pour la déplacer. Double-cliquez sur le fichier pour visualiser l'image que vous avez capturée (**Figure 8-8**). Félicitations : vous avez programmé une caméra !

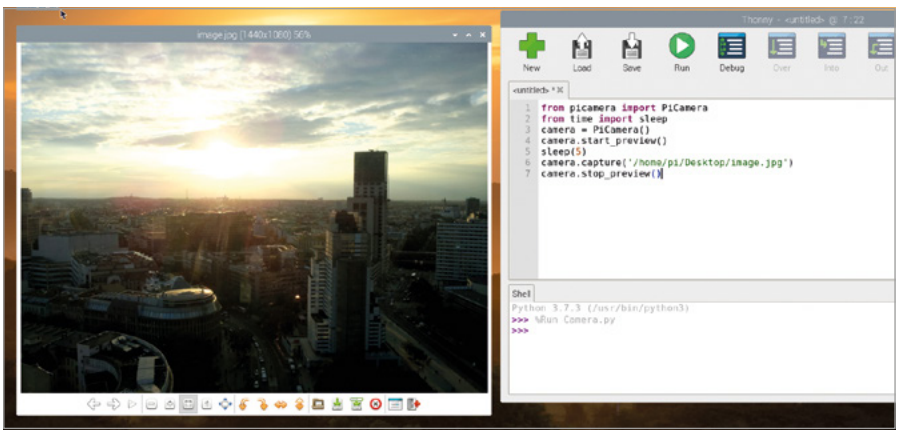


Figure 8-8 : Ouverture de l'image capturée

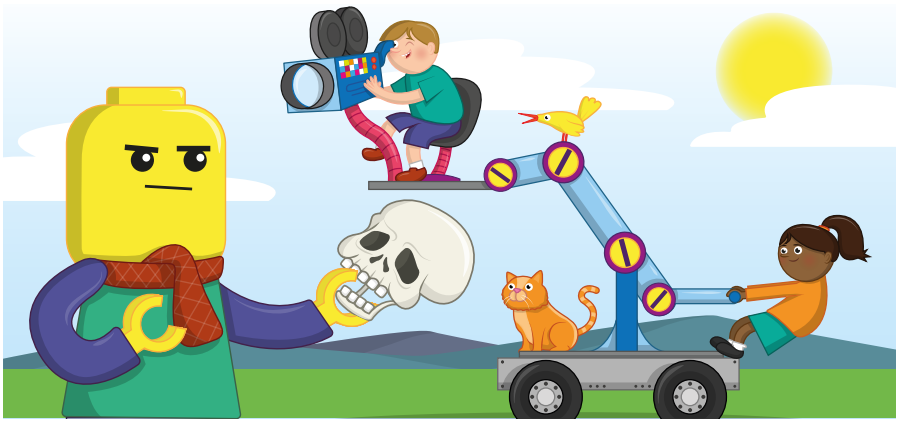
Capture d'images vidéo en mouvement

Outre des photos, vous pouvez également capturer des vidéos. Supprimez tout ce qui se trouve entre les lignes **`camera.start_preview()`** et **`camera.stop_preview()`**, puis saisissez ce qui suit sous **`camera.start_preview() :`**

```
camera.start_recording('/home/pi/Desktop/video.h264')
sleep(10)
camera.stop_recording()
```

L'aperçu de la caméra s'affichera, comme auparavant, mais cette fois, il sera également enregistré dans un fichier sur le bureau. Attendez 10 secondes, le délai que vous avez défini dans Python, faites quelques mouvements de danse devant la caméra pour rendre le tout plus intéressant puis, lorsque la prévisualisation sera terminée, vous retrouverez votre fichier vidéo sur le bureau.

Pour visionner la vidéo, il suffit de double-cliquer sur le fichier **video.h264** sur votre bureau. La lecture de la vidéo démarre et vous vous voyez danser à l'écran ! Une fois la vidéo terminée, le logiciel de lecture s'arrête avec un message amical dans le Terminal. Félicitations : vous pouvez maintenant capturer des vidéos en utilisant votre Raspberry Pi Camera Module ou la HQ Camera !



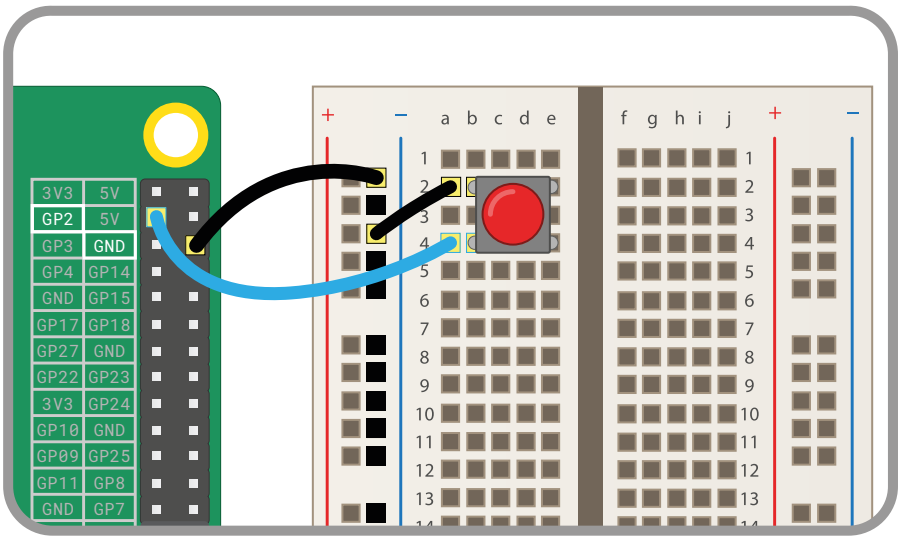
Bouton animation stop-motion (image par image)

En vous appuyant sur les connaissances acquises dans ce chapitre ainsi que dans le **Chapitre 6**, **L'informatique physique** concernant la connexion du matériel au connecteur GPIO du Raspberry Pi, il est temps de construire quelque chose de spécial : votre propre studio d'animation stop-motion.

L'animation stop motion consiste à prendre de nombreuses photos d'objets inanimés, comme des modèles réduits de voitures ou des figurines, et à les déplacer légèrement entre chaque photo. Bien que les objets ne bougent jamais dans les photos, si vous les montrez les uns après les autres rapidement, vous aurez l'impression qu'ils bougent aussi vite ou aussi lentement que vous le souhaitez !

Pour ce projet, vous aurez besoin d'un interrupteur à bouton-poussoir, d'une platine, d'un câble de connexion mâle-mâle (M2M) et d'une paire de fils de connexion mâle-femelle (M2F). Si vous n'avez pas de platine, vous pouvez connecter l'interrupteur en utilisant plutôt des câbles femelles-femelles (F2F), mais il sera plus difficile d'appuyer dessus. Si vous avez besoin d'un rappel sur chacun de ces éléments, consultez le **Chapitre 6, L'informatique physique avec Scratch et Python**. Vous aurez également besoin d'objets à animer : il peut s'agir d'un bloc d'argile, d'une voiture miniature ou d'une figurine.

Commencez par créer le circuit : ajoutez le bouton-poussoir à la platine, puis connectez le pôle de terre de la platine à une broche de terre du Raspberry Pi (désignée par le sigle GND sur la **Figure 8-9**) via un câble mâle-femelle. Utilisez un câble de connexion mâle-mâle pour connecter une borne de l'interrupteur au pôle de terre de la platine, puis un fil de connexion mâle-femelle pour connecter l'autre borne de l'interrupteur à la broche 2 du connecteur GPIO (dénommée GP2 sur la **Figure 8-9**).



▲ **Figure 8-9 :** Schéma de câblage de la connexion d'un bouton-poussoir aux broches du connecteur GPIO

Créez un nouveau projet dans Thonny et enregistrez-le sous le nom de **Stop Motion**. Commencez par importer et configurer les bibliothèques dont vous avez besoin pour utiliser la caméra et le port GPIO :

```
from picamera import PiCamera
from gpiozero import Button
camera = PiCamera()
button = Button(2)
```

Ensuite, saisissez :

```
camera.start_preview()
button.wait_for_press()
camera.capture('/home/pi/Desktop/image.jpg')
camera.stop_preview()
```

Cliquez sur Run pour obtenir un aperçu de ce que votre caméra visualise. L'aperçu s'affichera à l'écran jusqu'à ce que vous appuyiez sur le bouton-poussoir : appuyez dessus maintenant, et l'aperçu se fermera alors que votre programme enregistre une image sur le bureau. Cherchez l'image, appelée **image.jpg** et double-cliquez pour l'ouvrir et confirmer que le programme fonctionne.

L'animation stop-motion consiste à créer de nombreuses images fixes, pour donner une impression de mouvement en les assemblant l'une après l'autre. Pour éviter d'enregistrer toutes ces images sur votre bureau en générant une impression de désordre, créez un dossier pour les stocker. Faites un clic droit sur n'importe quelle partie du bureau ne contenant pas de fichier ou d'icône, puis choisissez New Folder (**Figure 8-10**). Nommez le dossier **animation** en minuscules, puis cliquez sur le bouton OK.

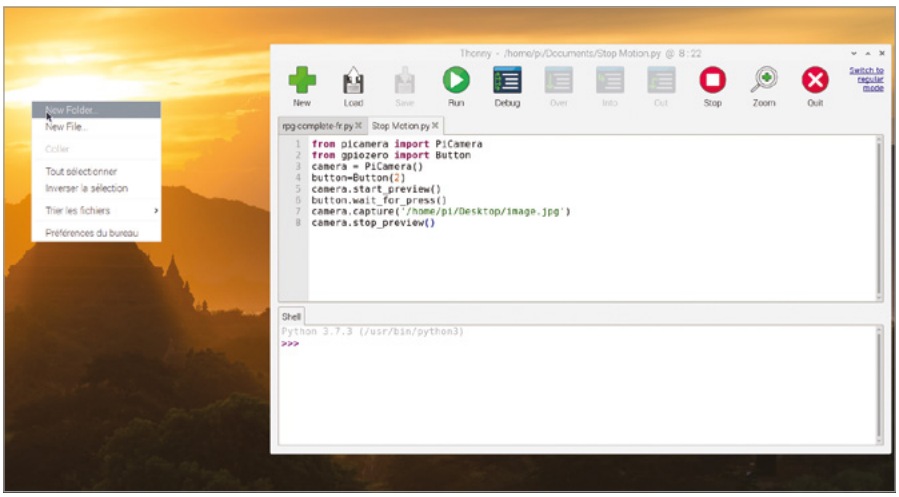


Figure 8-10 : Créez un nouveau dossier pour vos images capturées

Il est fastidieux de devoir redémarrer votre programme chaque fois que vous capturez une image pour votre animation, vous devez donc le modifier pour qu'il fonctionne en boucle. Contrairement aux boucles précédentes que vous avez créées, celle-ci doit pouvoir se fermer avec élégance : sinon, si vous arrêtez le programme pendant l'aperçu de la caméra, vous ne pourrez plus voir le bureau ! Pour ce faire, vous devez utiliser deux instructions spéciales : `try` et `except`.

Commencez par supprimer tout ce qui suit **camera.start_preview(),**, puis saisissez :

```
frame = 1
```

Vous créez une nouvelle variable, **frame**, que votre programme utilisera pour stocker le numéro de l'image en cours. Cette fonction permet d'enregistrer un nouveau fichier à chaque fois ; sans cela, vous écraserez votre première image chaque fois que vous appuyez sur le bouton !

Ensuite, saisissez ce qui suit pour définir votre boucle :

```
while True:
    try:
```

La nouvelle instruction **try** dit à Python d'exécuter le code, quel qu'il soit : dans ce cas, il s'agira de code de la capture d'images. Saisissez :

```
button.wait_for_press()
camera.capture('/home/pi/Desktop/animation/frame%03d.jpg' % frame)
frame += 1
```

Ces trois lignes de code contiennent plusieurs astuces très intelligentes. La première se cache dans le nom du fichier de capture : **%03d** et indique à Python de prendre un nombre et d'y ajouter des zéros à gauche pour atteindre les trois chiffres. Ainsi, « 1 » devient « 001 », « 2 » devient « 002 » et « 10 » devient « 010 ». Vous en avez besoin dans votre programme pour conserver vos fichiers dans le bon ordre et pour vous assurer que vous n'écrasez pas un fichier préalablement enregistré.

Le code **% frame** à la fin de cette ligne indique à Python d'utiliser le numéro de la variable frame dans le nom du fichier. Pour que chaque dossier soit unique, la dernière ligne **frame += 1** incrémente de 1 la variable frame. La première fois que vous appuyez sur le bouton, **frame** sera incrémentée de 1 à 2 ; la fois suivante, de 2 à 3 ; et ainsi de suite.

Mais pour l'instant, votre code ne se ferme pas encore proprement lorsque vous avez fini de prendre des photos. Pour ce faire, il vous faut une variable **except** à associer à votre variable **try**. Tapez ce qui suit, sans oublier de supprimer un niveau d'indentation sur la première ligne pour que Python sache qu'elle ne fait pas partie de la section **try** :

```
    except KeyboardInterrupt:
        camera.stop_preview()
        break
```

Une fois terminé, votre programme se présente comme suit :

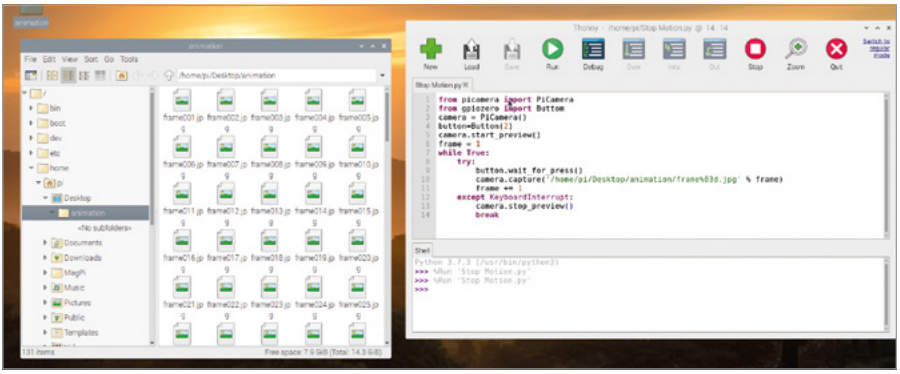
```
from picamera import PiCamera
from time import sleep
from gpiozero import Button
camera = PiCamera()
button = Button(2)
camera.start_preview()
frame = 1
while True:
    try:
        button.wait_for_press()
        camera.capture('/home/pi/Desktop/animation/frame%03d.jpg'
% frame)
        frame += 1
    except KeyboardInterrupt:
        camera.stop_preview()
        break
```

Cliquez sur Run, mais au lieu d'appuyer sur le bouton, appuyez sur les touches **CTRL** et **C** sur votre clavier. Vous n'avez pas besoin d'appuyer sur les deux touches en même temps : il suffit de maintenir la touche **CTRL**, d'appuyer et relâcher la touche **C** puis de relâcher **CTRL**. Ces deux touches agissent comme un interrupteur, indiquant à Python qu'il est temps d'arrêter. Sans la ligne **`except KeyboardInterrupt:`**, Python quitterait immédiatement et laisserait la prévisualisation de la caméra bloquer votre écran ; grâce à cette ligne, Python exécute le code qu'elle contient (dans ce cas, un code lui disant d'arrêter la prévisualisation de la caméra) et d'en sortir proprement.

Vous êtes maintenant prêt à commencer à capturer votre animation stop-motion ! Positionnez le Camera Module ou HQ Camera de manière à ce qu'il soit dirigé vers les objets à animer, et assurez-vous qu'il ne bouge pas : si la caméra bouge, l'effet est gâché. Placez vos objets dans leur position initiale, puis cliquez sur Exécuter pour lancer votre programme. Vérifiez que tout semble correct dans l'aperçu, et appuyez sur le bouton pour capturer votre première image.

Déplacez légèrement les objets (moins vous les déplacez entre une image et l'autre, plus l'animation terminée sera fluide) et appuyez à nouveau sur le bouton pour capturer une autre image. Continuez ainsi jusqu'à ce que votre animation soit terminée : plus vous capturez d'images, plus votre animation sera longue.

Lorsque vous avez terminé, appuyez sur **CTRL+C** pour arrêter votre programme, puis double-cliquez sur le dossier **animation** sur votre bureau pour voir les photos que vous avez prises (**Figure 8-11**, au verso). Double-cliquez sur une image pour l'ouvrir et la voir plus en détail !



▲ **Figure 8-11 :** Les images capturées dans le dossier

Mais pour l'instant, tout ce que vous avez, c'est un dossier rempli d'images. Pour créer une animation, vous devez les transformer en vidéo. Pour ce faire, cliquez sur l'icône framboise pour charger le menu, sélectionnez Accessoires et cliquez sur LXTerminal. Une *interface en ligne de commande* s'affiche, expliquée en détails dans l'**Annexe C**, qui vous permet de saisir des commandes à transmettre à votre Raspberry Pi. Lorsque le Terminal se charge, commencez par accéder au dossier que vous avez créé en saisissant :

```
cd Desktop/animation
```

Il est essentiel que le « D » de « Desktop » soit en majuscules ; Raspberry Pi OS *distingue les majuscules des minuscules* et, si vous ne saisissez pas un nom de commande ou de dossier exactement comme il a été écrit à l'origine, cela ne fonctionnera pas ! Une fois que vous avez accédé au dossier, saisissez :

```
ffmpeg -i frame%03d.jpg -r 10 animation.h264
```

Il s'agit d'un programme appelé **ffmpeg** qui permet de récupérer des images fixes dans le dossier et les convertir en une vidéo appelée **animation.h264**. (Remarque : si ffmpeg n'est pas disponible, vous pouvez l'installer avec `sudo apt-get install ffmpeg`.) Selon le nombre de photos que vous avez prises, ce processus peut prendre quelques minutes ; vous saurez qu'il est terminé lorsque vous verrez réapparaître l'invite du Terminal.

Pour lire la vidéo, cherchez le fichier **animation.h264** dans votre dossier **animation** et double-cliquez dessus pour l'ouvrir. Vous pouvez également le lire à partir du Terminal en saisissant :

```
omxplayer animation.h264
```

Une fois la vidéo chargée, vous verrez votre animation stop-motion prendre vie. Félicitations : vous avez transformé votre Raspberry Pi en un puissant studio d'animation !

Si votre animation se déplace trop rapidement ou trop lentement, modifiez la partie **`-r 10`** de la commande **`ffmpeg`** par un nombre inférieur ou supérieur : il s'agit de la cadence d'images, ou le nombre d'images fixes contenues dans une seconde de vidéo. Un chiffre plus bas rendra l'animation plus lente, mais moins fluide ; un chiffre plus élevé rendra l'animation plus fluide, mais plus rapide.

Si vous souhaitez enregistrer votre vidéo, veillez à la faire glisser du bureau et à la déposer vers votre dossier Videos ; dans le cas contraire, la prochaine fois que vous lancerez votre programme, vous allez écraser le fichier !

Paramètres avancés de la caméra

Si vous avez besoin de plus de commandes sur le Raspberry Pi Camera Module ou la HQ Camera, vous pouvez utiliser la bibliothèque Python picamera pour accéder à divers paramètres. Ces paramètres, ainsi que leurs valeurs par défaut, sont détaillés ci-dessous pour être inclus dans vos propres programmes.

`camera.awb_mode = 'auto'`

Cette fonction définit le mode de balance des blancs automatique de l'appareil photo, et peut être réglée sur l'un des modes suivants : **`off`**, **`auto`**, **`sunlight`** (lumière naturelle), **`cloudy`** (nuageux), **`shade`** (ombre), **`tungsten`** (tungstène), **`fluorescent`**, **`incandescent`**, **`flash`**, or **`horizon`**. Si vous trouvez que vos photos et vidéos sont un peu trop bleues ou trop jaunes, essayez un autre mode.

`camera.brightness = 50`

Cette variable définit la luminosité de l'image de la caméra, de plus sombre (0) à plus claire (100).

`camera.color_effects = None`

Cette variable modifie l'effet de couleur actuellement utilisé par la caméra. Normalement, ce réglage devrait être laissé tel quel, mais si vous fournissez une paire de chiffres, vous pouvez modifier la façon dont la caméra enregistre la couleur : essayez **`(128, 128)`** pour créer une image en noir et blanc.

`camera.contrast = 0`

Cette variable détermine le contraste de l'image. Un chiffre plus élevé génère un effet plus dramatique et violent, alors qu'un nombre plus bas génère un effet plus nuancé. Vous pouvez utiliser n'importe quel nombre entre -100 pour un contraste minimum et 100 pour un contraste maximum.

`camera.crop = (0.0, 0.0, 1.0, 1.0)`

Cela vous permet de recadrer l'image, en rognant les côtés et le haut pour ne capturer que la partie de l'image dont vous avez besoin. Les nombres représentent la coordonnée X, la coordonnée Y, la largeur et la hauteur, et par défaut, ils capturent l'image complète. Essayez de réduire les deux derniers nombres (- 0,5 et 0,5 par exemple) pour observer l'effet de ce réglage.

`camera.exposure_compensation = 0`

Cette variable détermine la *compensation* de l'exposition de l'appareil photo, ce qui vous permet de contrôler manuellement la quantité de lumière capturée pour chaque image. Contrairement au réglage de la luminosité, cette commande permet de contrôler la caméra directement. Les valeurs valides vont de -25 pour une image très sombre à 25 pour une image très claire.

`camera.exposure_mode = 'auto'`

Cette variable définit le *mode d'exposition* ou la logique suivie par la caméra pour déterminer l'exposition idéale d'une image. Les possibilités sont : **`off`**, **`auto`**, **`night`** (nuit), **`backlight`** (éclairage arrière), **`spotlight`**, **`sports`**, **`snow`** (neige), **`beach`** (plage), **`verylong`** (très long), **`fixedfps`**, **`antishake`** (anti-vibrations) et **`fireworks`** (feux d'artifice).

`camera.framerate = 30`

Cette variable définit le nombre d'images capturées pour créer une seconde de vidéo, appelé *cadence d'images*. Une cadence d'images plus élevée crée une vidéo plus fluide, mais occupe plus d'espace de stockage. Des fréquences d'images plus élevées nécessitent l'utilisation d'une résolution plus faible, que vous pouvez régler via **`camera.resolution`**.

`camera.hflip = False`

Cette variable permet de retourner l'image de la caméra sur l'axe horizontal, ou X, lorsqu'elle est réglée sur **`True`**.

`camera.image_effect = 'none'`

Cette variable permet d'appliquer l'un des nombreux effets d'image au flux vidéo, qui sera visible dans l'aperçu ainsi que dans les images et vidéos enregistrées. Les possibilité sont : **`blur`** (flou), **`cartoon`** (dessin animé), **`colorbalance`** (balance des couleurs), **`colorpoint`**, **`colorswap`** (échange de couleur), **`deinterlace1`**, **`deinterlace2`**, **`denoise`** (suppression du bruit), **`emboss`** (relief), **`film`**, **`gpen`**, **`hatch`** (hachuré), **`negative`** (négatif), **`none`** (aucun), **`oilpaint`** (peinture à l'huile), **`pastel`**, **`posterise`**, **`saturation`**, **`sketch`**, **`solarize`** (solariser), **`washedout`** (délavé) et **`watercolor`** (couleur eau).

`camera.ISO = 0`

Cette variable modifie le réglage ISO de l'appareil photo, ce qui affecte sa sensibilité à la lumière. Par défaut, la caméra ajuste automatiquement ce paramètre en fonction de la lumière disponible. Vous pouvez définir vous-même l'ISO en utilisant l'une des valeurs suivantes : 100, 200, 320, 400, 500, 640, 800. Plus l'ISO est élevé, plus la caméra sera performante dans les environnements à faible luminosité, mais la vidéo qu'elle capture sera plus granuleuse.

`camera.meter_mode = 'average'`
Cette variable contrôle la manière dont l'appareil photo définit la quantité de lumière disponible lors du réglage de l'exposition. La valeur par défaut est la moyenne de la quantité de lumière disponible sur l'ensemble de l'image ; les autres modes possibles sont **`backlit`** (éclairé par l'arrière), **`matrix`** (matrice) et **`spot`** (point).

`camera.resolution = (1920, 1080)`
Cette variable définit la résolution de l'image ou de la vidéo capturée, représentée par les valeurs de largeur et de hauteur. Les résolutions plus basses occupent moins d'espace de stockage et permettent d'utiliser une fréquence d'images plus élevée ; les résolutions plus élevées sont de meilleure qualité mais occupent plus d'espace de stockage.

`camera.rotation = 0`
Cette variable contrôle la rotation de l'image, de 0 à 90, 180 et 270 degrés. Utilisez-la si vous ne pouvez pas positionner la caméra de manière à ce que la limande sorte de la partie basse.

`camera.saturation = 0`
Cette variable permet de contrôler la saturation de l'image, ou la vibrance des couleurs. Les valeurs possibles vont de -100 à 100.

`camera.sharpness = 0`
Cette variable permet de contrôler la netteté de l'image. Les valeurs possibles vont de -100 à 100.

`camera.shutter_speed = 0`
Cette variable permet de contrôler la vitesse d'ouverture et de fermeture de l'obturateur lors de la capture d'images et de vidéos. Vous pouvez régler la vitesse d'obturation manuellement en microsecondes. Les vitesses d'obturation plus longues fonctionnent mieux en cas de faible luminosité et les vitesses d'obturation plus rapides lorsque la lumière est plus intense. Il est préférable de conserver les paramètres automatiques par défaut.

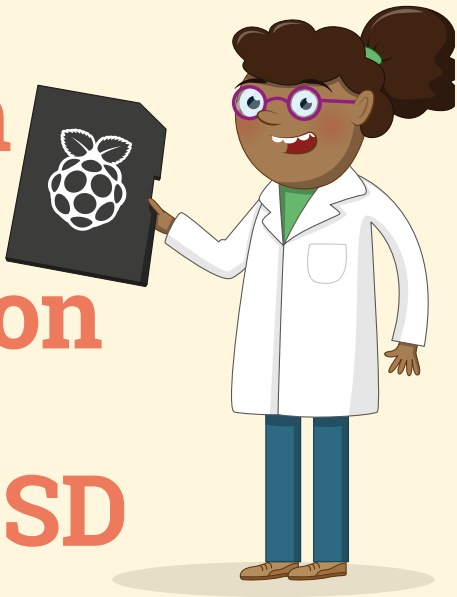
`camera.vflip = False`
Cette variable permet de retourner l'image de la caméra sur son axe vertical, ou Y, lorsqu'elle est réglée sur **`True`**.

`camera.video_stabilization = False`
Lorsque cette variable est réglée sur **`True`**, la stabilisation vidéo est activée. Vous n'en avez besoin que si la caméra est en mouvement pendant que vous enregistrez, par exemple si elle est fixée à un robot ou transportée, afin de réduire les tremblements.

De plus amples informations sur ces paramètres, ainsi que sur d'autres paramètres non mentionnés ici, sont disponibles à l'adresse **picamera.readthedocs.io**.

Annexe A

Installer un système d'exploitation sur une carte microSD



Il est possible d'acheter des cartes microSD à l'aide du logiciel NOOBS (New Out Of the Box Software, ou nouveau logiciel prêt à l'emploi) préinstallé auprès d'un fournisseur Raspberry Pi, ce qui vous permettra d'installer facilement Raspberry Pi OS (qui était connu auparavant sous le nom de Raspbian) pour ton Raspberry Pi. Sinon, suivez les instructions ci-après pour installer un système d'exploitation manuellement sur une carte microSD vierge (ou réutilisable) à l'aide de Raspberry Pi Imager .

ATTENTION !

Si vous avez acheté une carte microSD avec NOOBS préinstallé, vous n'avez rien d'autre à faire que de l'insérer dans votre Raspberry Pi. Ces instructions concernent des cartes microSD vierges ou déjà utilisées sur lesquelles vous souhaitez installer un nouveau système d'exploitation. Si vous suivez ces instructions avec une carte microSD contenant des fichiers, ces derniers seront définitivement perdus : veillez à effectuer d'abord une sauvegarde !

Télécharger Raspberry Pi Imager

Basé sur Debian, Raspberry Pi OS est le système d'exploitation officiel pour Raspberry Pi. La façon la plus simple d'installer Raspberry Pi OS sur une carte microSD pour votre Raspberry Pi est d'utiliser l'outil Raspberry Pi Imager, téléchargeable depuis la page **rpf.io/downloads**. Cette méthode constitue une alternative à l'installation du système d'exploitation via NOOBS, bien que ce dernier soit toujours disponible sur cette même page de téléchargements.

L'application Raspberry Pi Imager est disponible pour les ordinateurs Windows, macOS et Ubuntu Linux, choisissez donc la version appropriée pour votre système.

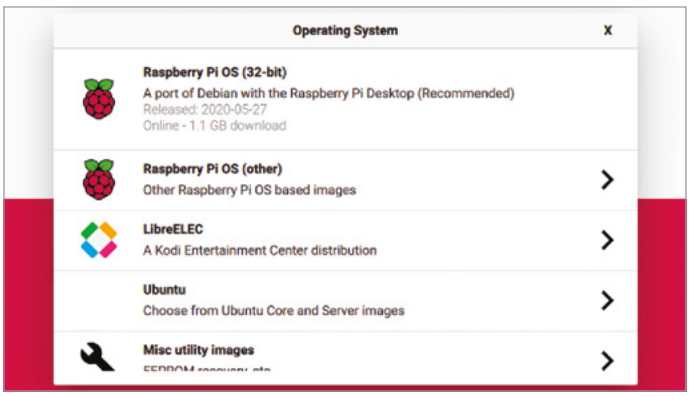
Sur macOS, double-cliquez sur le fichier DMG téléchargé. Il vous sera peut-être demandé de modifier vos paramètres de sécurité et de confidentialité pour permettre aux applications téléchargées depuis « l'App Store et développeurs identifiés » de fonctionner. Vous pouvez alors simplement faire glisser l'icône de Raspberry Pi Imager dans le dossier Applications.

Sur un PC Windows, double-cliquez sur le fichier EXE téléchargé. Lorsque vous y êtes invité, lancez-le en sélectionnant « Oui ». Cliquez ensuite sur le bouton « Installer » pour lancer l'installation.

Inscrire le système d'exploitation sur la carte microSD

Rattachez votre carte microSD à votre ordinateur PC ou Mac : si aucun lecteur de carte n'est intégré, vous aurez besoin d'un adaptateur USB pour carte microSD. La carte ne doit pas nécessairement être pré-formatée.

Lancez Raspberry Pi Imager. Cliquez sur le bouton « Choose OS (Choisir le système d'exploitation) » pour sélectionner le système d'exploitation que vous souhaitez installer. La première option est Raspberry Pi OS : si vous préférez la version allégée Lite ou la version complète (avec tous les logiciels recommandés préinstallés), sélectionnez « Raspberry Pi OS (other) » (Système d'exploitation Raspberry (autre)). Vous pouvez également installer LibreELEC (choisissant ainsi la version de votre modèle Raspberry Pi) et Ubuntu Core ou Server.



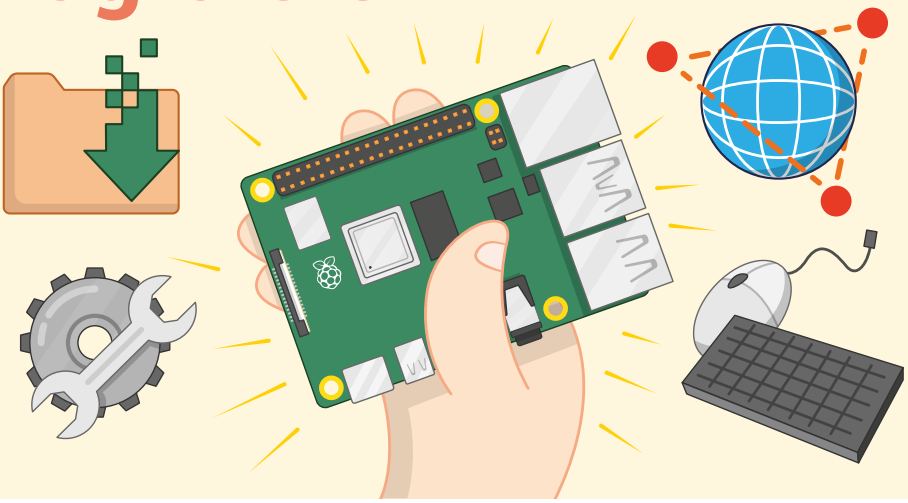
Remarque : Si vous préférez installer un autre système d'exploitation comme Lakka par exemple, il vous suffit de télécharger son fichier image depuis le site Web correspondant, puis sélectionner l'option « Use Custom » dans Raspberry Pi Imager.

Une fois le système d'exploitation sélectionné, cliquez sur le bouton « Choose SD card (Choisir une carte SD) » et sélectionnez votre carte microSD (en général, il n'y a qu'une seule option).

Enfin, cliquez sur le bouton « Write (Écrire) » et attendez pendant que l'utilitaire inscrit le système d'exploitation sélectionné sur votre carte, puis le vérifie. Une fois le processus terminé, vous pouvez retirer la carte microSD. Vous pouvez ensuite l'insérer dans votre Raspberry Pi et démarrer celui-ci sous le système d'exploitation que vous venez d'installer.

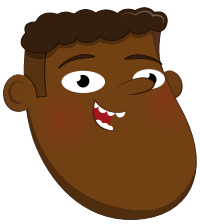
Annexe B

Installation et désinstallation de logiciels



Raspberry Pi OS est fourni avec une sélection de logiciels populaires, sélectionnés par la Raspberry Pi Foundation, mais d'autres logiciels fonctionneront également sur un Raspberry Pi. En suivant les instructions ci-après, vous pouvez découvrir des logiciels supplémentaires, les installer et les désinstaller, afin d'élargir les possibilités de votre Raspberry Pi.

La présente annexe contient des instructions qui s'ajoutent à celles du **Chapitre 3, Utilisation de votre Raspberry Pi**, qui explique comment utiliser l'outil Recommended Software ; si vous ne l'avez pas encore fait, lisez-le avant d'utiliser les méthodes décrites dans cette annexe.

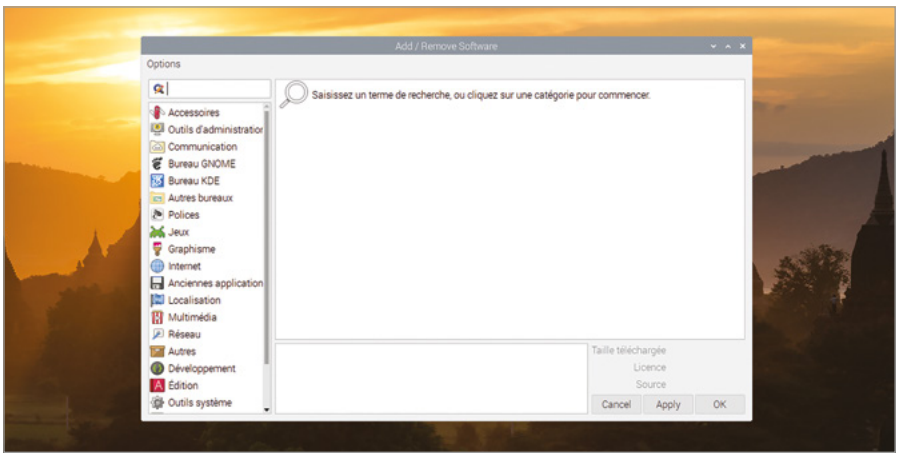


CAPACITÉ DE LA CARTE

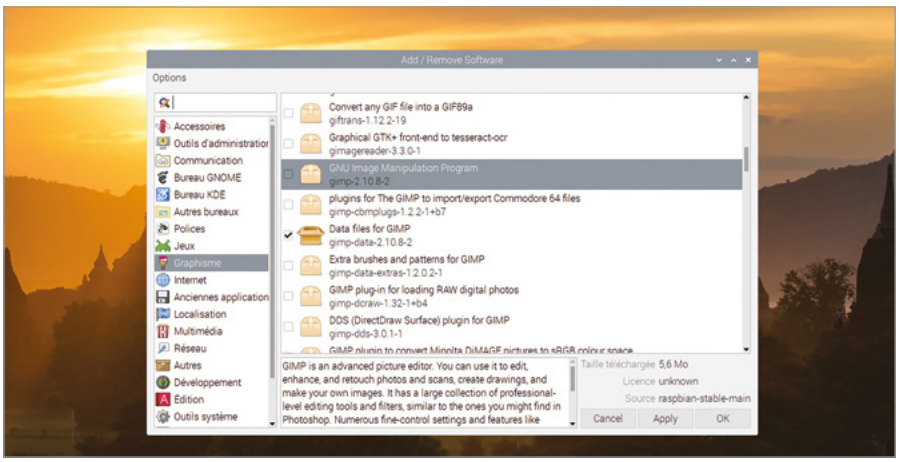
L'ajout de logiciels supplémentaires à votre Raspberry Pi occupera de l'espace sur votre carte microSD. Une carte de 16 Go ou plus vous permettra d'installer plus de logiciels ; pour vérifier si la carte que vous comptez utiliser est compatible avec Raspberry Pi, consultez **rpf.io/sdcardlist**.

Explorer les logiciels disponibles

Pour visualiser et rechercher la liste des logiciels disponibles pour Raspberry Pi OS, en utilisant ce qu'il est convenu d'appeler des *référentiels de logiciels*, cliquez sur l'icône de la framboise pour charger le menu, sélectionnez la catégorie Préférences, puis cliquez sur Add/Remove Software (Ajouter/supprimer un logiciel). Après quelques secondes, la fenêtre de l'outil s'affichera.



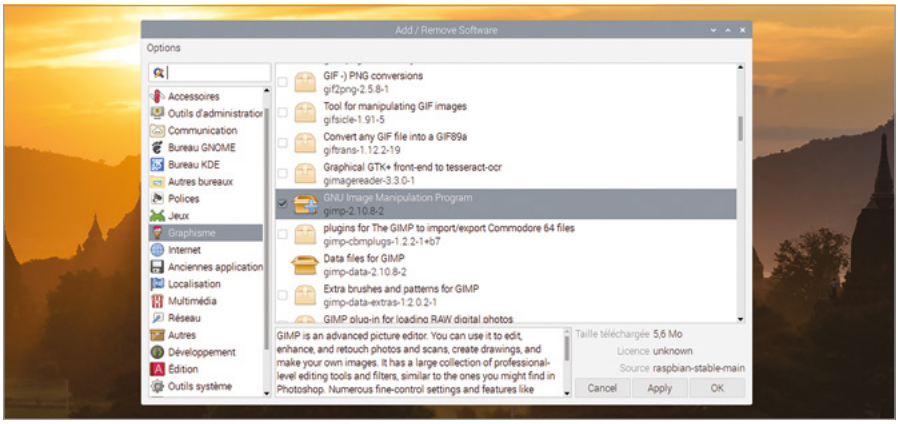
Sur le côté gauche de la fenêtre Add/Remove Software (Ajouter/supprimer un logiciel) s'affiche une liste de catégories, ces mêmes catégories qui figurent dans le menu principal en cliquant sur l'icône framboise. En cliquant sur l'une d'elles, vous obtiendrez la liste des logiciels disponibles dans cette catégorie. Vous pouvez également saisir un terme de recherche dans la case située en haut à gauche de la fenêtre, par exemple « éditeur de texte » ou « jeu », pour que la liste des logiciels correspondants à une catégorie s'affiche. En cliquant sur un package, vous obtiendrez des informations supplémentaires à son sujet dans l'espace situé au bas de la fenêtre.



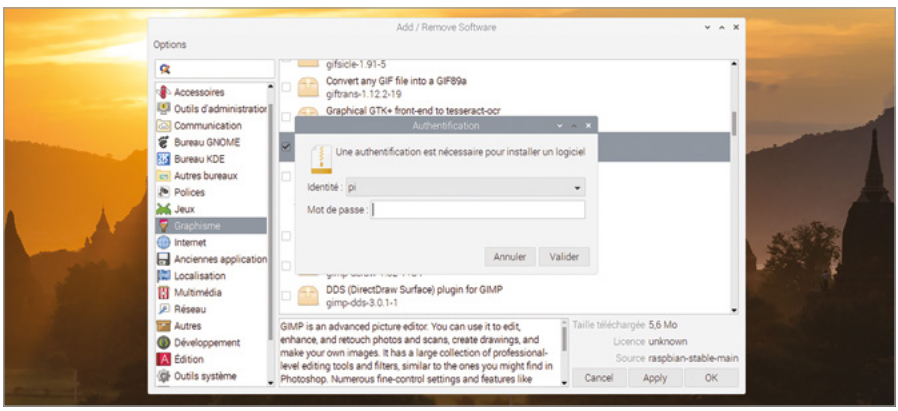
Dans la catégorie que vous avez sélectionnée, de nombreux packages logiciels sont disponibles et il se peut que l'outil Add/Remove Sofware (Ajouter/supprimer un logiciel) mette un certain temps à parcourir la liste entière.

Installation de logiciels

Pour sélectionner un package à installer, cochez la case en regard de celui-ci en cliquant dessus. Vous pouvez installer plus d'un package à la fois : il suffit de cliquer pour en ajouter d'autres. L'icône correspondant au package se transforme en une boîte ouverte avec un symbole « + », pour confirmer qu'il va être installé.

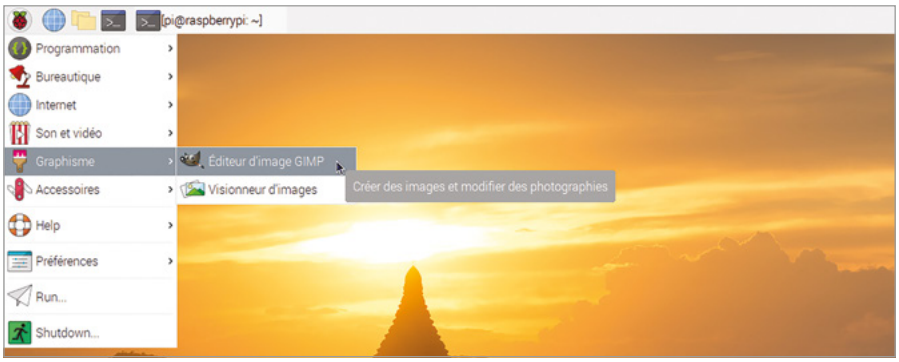


Lorsque vous êtes satisfait de vos choix, cliquez sur le bouton OK ou Appliquer ; la seule différence est que le bouton OK ferme l'outil Add/Remove Software (Ajouter/supprimer un logiciel) lorsque votre logiciel est installé, tandis que le bouton Appliquer le laisse ouvert. Il vous sera demandé d'entrer votre mot de passe pour confirmer votre identité : vous ne voulez certainement pas que n'importe qui puisse ajouter ou supprimer un logiciel dans votre Raspberry Pi !



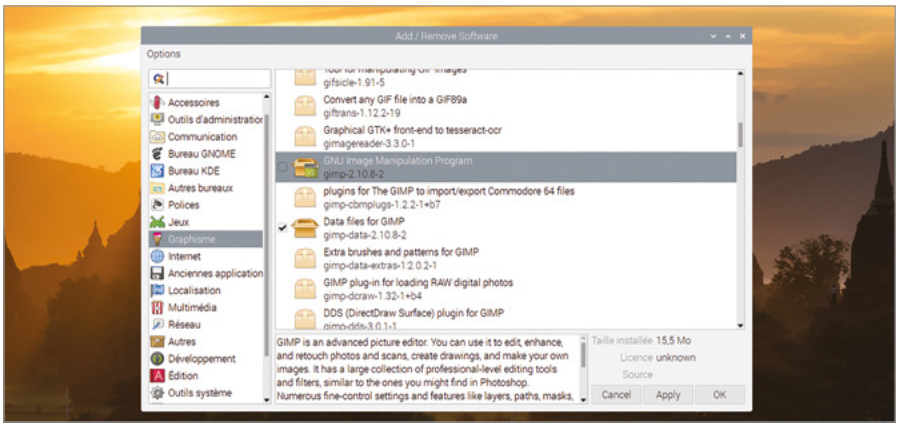
Vous constaterez peut-être que, lorsque vous installez un seul package, d'autres packages sont installés en même temps que celui-ci ; ces packages supplémentaires sont appelés *dépendances*, dont les logiciels que vous avez choisi d'installer ont besoin pour fonctionner : par exemple, des bundles d'effets sonores pour un jeu ou une base de données pour un serveur Web.

Une fois le logiciel installé, vous le retrouvez en cliquant sur l'icône framboise pour charger le menu et trouver la catégorie du logiciel. Il est important de savoir que la catégorie du menu n'est pas toujours la même que celle de l'outil Add/Remove Software (Ajouter/supprimer un logiciel), car certains logiciels ne figurent pas du tout dans le menu ; ces logiciels sont appelés *logiciel en ligne de commande* et doivent être exécutés dans le Terminal. Pour plus d'informations sur la ligne de commande et le terminal, consultez **Annexe C, L'interface de la ligne de commande**.



Désinstallation de logiciels

Pour sélectionner un package à supprimer ou *désinstaller*, recherchez-le dans la liste des packages (la fonction de recherche est très utile) et décochez la case en regard de celui-ci en cliquant dessus. Vous pouvez désinstaller plus d'un package à la fois : il suffit continuer à cliquer pour en supprimer d'autres. L'icône correspondant au package se transforme en une boîte ouverte placée aux côtés d'une corbeille de recyclage, pour confirmer qu'il va être désinstallé.



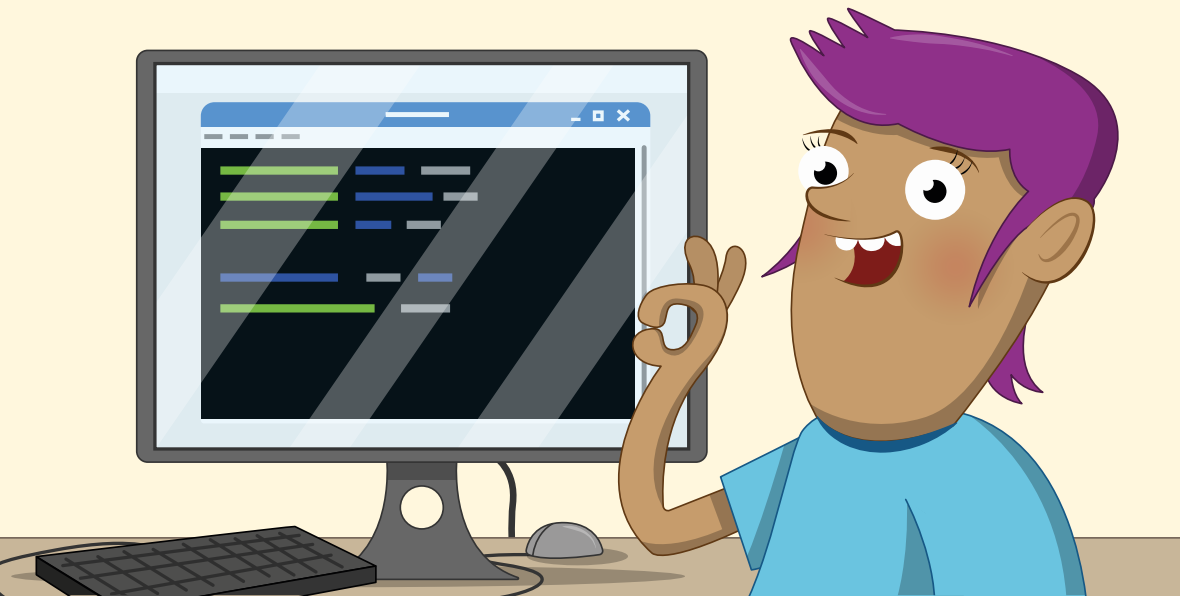
Tout comme vous l'avez fait auparavant, vous pouvez cliquer sur OK ou sur Appliquer pour lancer la désinstallation des packages logiciels sélectionnés. Il vous sera demandé de confirmer votre mot de passe, sauf si vous l'avez fait dans les dernières minutes, et vous pourrez également être invité à confirmer que vous souhaitez supprimer toute dépendance associée à votre package logiciel. Une fois la désinstallation terminée, le logiciel disparaîtra du menu de l'icône framboise, mais les fichiers que vous avez créés à l'aide du logiciel (images pour un pack graphique, par exemple, ou sauvegardes pour un jeu) ne seront pas supprimés.

ATTENTION !

Tous les logiciels installés dans Raspberry Pi OS s'affichent dans l'outil Add/Remove Software (Ajouter/supprimer un logiciel), y compris les logiciels nécessaires au fonctionnement de votre Raspberry Pi. Il est possible de supprimer suffisamment de packages et que cela empêche le bureau de se charger. Pour éviter cela, ne désinstallez aucun logiciel à moins d'être sûr que vous n'en avez plus besoin. Si cela s'est déjà produit, réinstallez Raspberry Pi OS en suivant les instructions du **Chapitre 2, Premiers pas avec Raspberry Pi**ou réinstallez le système d'exploitation comme décrit dans l'**Annexe A**.

Annexe C

L'interface en ligne de commande



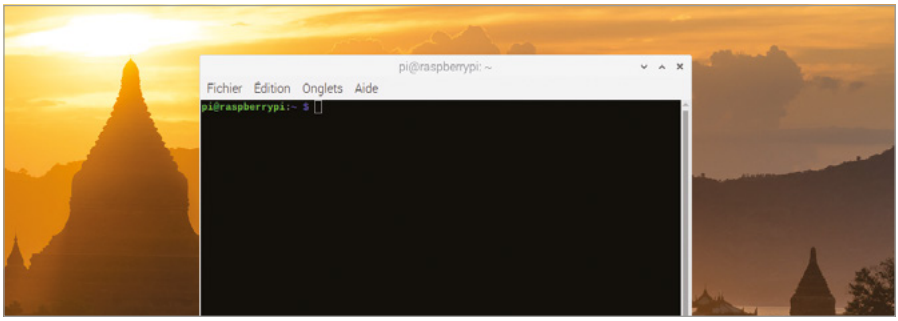
Bien qu'il soit possible de gérer la plupart des logiciels d'un Raspberry Pi depuis le bureau, certains ne sont accessibles qu'en mode texte, connu sous le nom de *interface en ligne de commande (CLI)* dans une application dénommée Terminal. La plupart des utilisateurs n'auront jamais besoin d'utiliser la CLI, mais pour ceux qui veulent en savoir plus, cette annexe propose une introduction de base.

PLUS D'INFOS

Cette annexe n'est pas conçue pour être un guide exhaustif de l'interface en ligne de commande de Linux. Pour un aperçu plus détaillé de l'utilisation de la CLI, consultez le site **rpf.io/terminal** dans un navigateur Web.

Chargement du Terminal

L'accès à la CLI se fait par le Terminal, un logiciel qui charge un élément désigné par le terme technique *terminal de téléscripteur virtuel (VTY)*. Ce terme remonte aux débuts de l'informatique, lorsque les utilisateurs définissaient des commandes au moyen d'une grosse machine à écrire électromécanique et ne disposaient pas d'un clavier et d'un écran. Pour charger le package Terminal, cliquez sur l'icône framboise pour charger le menu, sélectionnez la catégorie Accessoires et cliquez sur Terminal.



Tout comme n'importe quelle autre fenêtre, la fenêtre du Terminal peut être déplacée sur le bureau, redimensionnée, maximisée ou minimisée. Vous pouvez également agrandir les caractères si vous avez du mal à lire, ou les réduire si vous souhaitez mieux les recadrer dans la fenêtre : cliquez sur le menu Édition et choisissez respectivement Zoom avant ou Zoom arrière, ou appuyez et maintenez enfoncée la touche **CTRL** sur votre clavier, puis la touche **+** ou **-**.

L'invite

Le premier élément que vous trouverez dans un Terminal est une *invite* qui attend vos instructions. Sur un Raspberry Pi fonctionnant sous Raspberry Pi OS, l'invite est la suivante :

```
pi@raspberrypi:~ $
```

La première partie de l'invite, **`pi`**, est votre nom d'utilisateur ; la deuxième partie, après le **`@`**, est le nom d'hôte de l'ordinateur que vous utilisez, qui est par défaut votre **`raspberrypi`**. Après les **`:`** se trouve un tilde, le symbole **`~`**, qui est une manière abrégée de se référer à votre répertoire racine et représente votre *répertoire de travail actuel (CWD)*. Enfin, le symbole **`$`** indique que votre utilisateur est un *utilisateur non privilégié*, ce qui signifie que vous avez donc besoin d'un mot de passe pour effectuer des tâches telles que l'ajout ou la suppression de logiciels.

Se déplacer

Tentez de saisir ce qui suit, puis appuyez sur la touche **ENTRÉE** :

```
cd Desktop
```

L'invite devient donc :

```
pi@raspberrypi:~/Desktop $
```

Cela vous montre que votre répertoire de travail actuel a changé : vous étiez auparavant dans votre répertoire racine, indiqué par le symbole **~**, et vous vous trouvez désormais dans le sous-répertoire **Desktop** (Bureau) au-dessous de votre répertoire racine. Pour ce faire, vous avez utilisé la commande **cd**, à savoir *changer de répertoire*.

MAJUSCULES ET MINUSCULES

L'interface en ligne de commande de Raspberry Pi OS distingue les majuscules des minuscules, ce qui signifie qu'il est important que les commandes ou les noms aient des majuscules et des minuscules. Si vous recevez un message « Aucun fichier ou répertoire » lorsque vous avez essayé de changer de répertoire, vérifiez que vous avez écrit Desktop avec un D majuscule.

Il y a quatre moyens de revenir à votre répertoire racine : essayez-les toutes à tour de rôle, en revenant ensuite dans le sous-répertoire **Desktop** à chaque fois. La première est :

```
cd ..
```

Les symboles **..** sont un autre raccourci, cette fois indiquant « le répertoire au-dessus de celui-ci », également connu sous le nom de *répertoire parent*. Du moment que le répertoire au-dessus de **Desktop** est votre répertoire racine, vous allez y être renvoyé. Revenez au sous-répertoire **Desktop** et essayez le deuxième moyen :

```
cd ~
```

Le symbole **~** signifie littéralement « changer en mon répertoire racine ». Contrairement à **cd ..**, qui vous emmène simplement au répertoire parent du répertoire dans lequel vous vous trouvez actuellement, cette commande fonctionne à partir de n'importe où. Il existe cependant un moyen encore plus simple :

```
cd
```

Sans qu'on lui donne le nom d'un répertoire, **cd** retourne par défaut à votre répertoire racine. Le dernier moyen de retourner à votre répertoire racine est de saisir :

```
cd /home/pi
```

Ce faisant, vous empruntez un *chemin absolu*, qui fonctionnera quel que soit le répertoire de travail en cours. Ainsi, à l'égal que **cd** seul ou **cd ~**, le système vous renvoie à votre répertoire racine, où que vous soyez ; contrairement aux autres méthodes, cependant, vous devez nécessairement connaître votre nom d'utilisateur.

Gestion des dossiers

Pour vous entraîner à travailler avec des dossiers, passez au répertoire **Desktop** et saisissez ce qui suit :

```
touch Test
```

Vous verrez un fichier appelé **Test** s'afficher sur le bureau. La commande **touch** est généralement utilisée pour mettre à jour les informations de date et d'heure d'un fichier, mais si (comme c'est le cas) le fichier n'existe pas, elle le crée.

Essayez avec :

```
cp Test Test2
```

Vous verrez un second fichier, appelé **Test2**, s'afficher sur le bureau. Il s'agit d'une *copie* du fichier original, identique en tout point. Supprimez-le en saisissant :

```
rm Test2
```

Ce faisant, vous allez *supprimer* le fichier, et vous le verrez donc disparaître.

> **ATTENTION !**
>
> Lorsque vous supprimez des fichiers à l'aide du Gestionnaire de fichiers graphique, ceux-ci sont stockés dans la Corbeille et vous pouvez les récupérer ultérieurement. En revanche, les fichiers supprimés via la commande **rm** sont supprimés pour toujours. Faites attention à ce que vous saisissez !

Ensuite, essayez :

```
mv Test Test2
```

Cette commande permet de *déplacer* le fichier, et vous verrez votre fichier **Test** disparaître et être remplacé par **Test2**. La commande de déplacement, **mv**, peut être utilisée de cette manière pour renommer des fichiers.

Cependant, lorsque vous n'êtes pas sur le bureau, vous devez quand même être en mesure de voir quels fichiers se trouvent dans un répertoire. Saisissez :

```
ls
```

Cette commande *répertorie* le contenu du répertoire en cours, ou de tout autre répertoire que vous lui indiquez. Pour plus de détails, y compris la liste des fichiers masqués et l'indication de la taille des fichiers, essayez d'ajouter quelques sélecteurs :

```
ls -larth
```

Ces sélecteurs contrôlent la commande **ls** : **l** commute la sortie en une longue liste verticale ; **a** lui commande d'afficher tous les fichiers et répertoires, y compris ceux qui seraient normalement masqués ; **r** inverse l'ordre de tri normal ; **t** trie en fonction de la date de modification, ce qui, combiné avec **r**, renvoie les fichiers les plus anciens en haut de la liste et les fichiers les plus récents en bas ; **h** utilise des tailles de fichiers lisibles par l'homme, ce qui rend la liste plus facile à comprendre.

Exécution des programmes

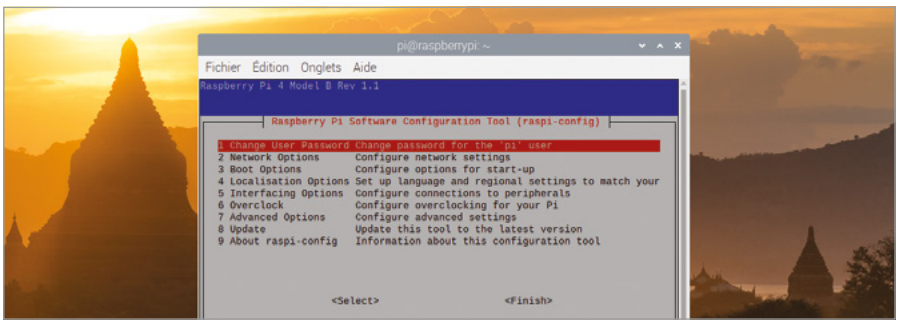
Certains programmes ne peuvent être exécutés qu'en ligne de commande, tandis que d'autres ont une interface graphique et une interface en ligne de commande. Par exemple, l'outil Logiciel de configuration du Raspberry Pi entre dans la seconde catégorie et vous devez normalement le charger à partir du menu de l'icône framboise. Saisissez :

```
raspi-config
```

Un message d' erreur vous indiquera alors que le logiciel ne peut être exécuté qu'en tant que *root*, le compte de super-utilisateur de votre Raspberry Pi. Il vous indiquera également comment y parvenir, en saisissant :

```
sudo raspi-config
```

La partie **sudo** de la commande signifie *switch-user do* et indique à Raspberry Pi OS d'exécuter la commande en tant qu'utilisateur root.



Vous ne devrez utiliser **sudo** que lorsqu'un programme a besoin de *privilèges* élevés, par exemple lorsqu'il s'agit d'installer ou de désinstaller des logiciels ou d'ajuster les paramètres du système. Un jeu, par exemple, ne doit jamais être lancé en utilisant **sudo**.

Appuyez sur la touche **TAB** deux fois pour sélectionner Finish (Terminer) et appuyez sur **ENTRÉE** pour quitter l'outil Logiciel de configuration du Raspberry Pi et revenir à l'interface en ligne de commande. Enfin, saisissez :

```
exit
```

Cela mettra fin à votre session de l'interface en ligne de commande et fermera l'application Terminal.

Utilisation des TTY

L'application Terminal n'est pas la seule façon d'utiliser l'interface en ligne de commande : vous pouvez également passer à l'un des nombreux terminaux en service appelés *télétypes* ou *TTY*. Maintenez enfoncées les touches **CTRL** et **ALT** sur votre clavier et appuyez sur la touche **F2** pour passer à « tty2 ».

Vous devrez vous reconnecter avec votre nom d'utilisateur et votre mot de passe, après quoi vous pourrez utiliser l'interface en ligne de commande comme dans le Terminal. L'utilisation de ces TTY est pratique lorsque, pour une raison quelconque, l'interface principale du bureau ne fonctionne pas.

Pour quitter le TTY, appuyez sur **CTRL+ALT**, puis sur **F7** : le bureau s'affichera à nouveau. Appuyez sur **CTRL+ALT+F2** et vous repasserez en mode « tty2 » : tout ce que vous aviez en cours d'exécution sera toujours là.

Avant de changer à nouveau, saisissez :

```
exit
```

Appuyez ensuite sur **CTRL+ALT+F7** pour revenir au bureau. La raison pour laquelle vous devez sortir avant de passer en mode TTY est que toute personne ayant accès au clavier peut passer en TTY ; si vous êtes toujours connecté, elle pourra accéder à votre compte sans même connaître votre mot de passe !

Félicitations : vous avez fait vos premiers pas dans la maîtrise de l'interface en ligne de commande de Raspberry Pi OS !

Annexe D

Lecture complémentaire

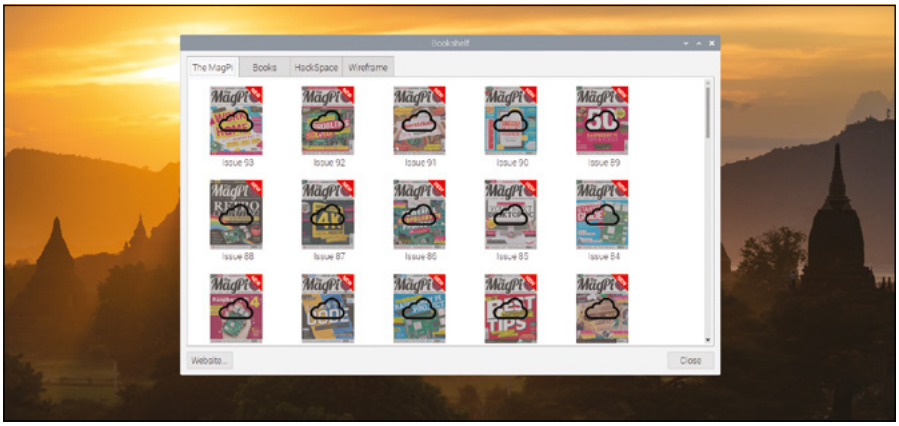
Le Guide officiel du débutant Raspberry Pi est conçu pour vous aider à faire vos premiers pas avec votre Raspberry Pi, mais il ne s'agit en aucun cas d'une présentation exhaustive de tout ce que vous pouvez faire. La communauté Raspberry Pi s'étend sur l'ensemble de la planète et ses membres réalisent toutes sortes d'activités, des jeux aux applications de détection en passant par la robotique et l'intelligence artificielle, ce qui représente une immense source d'inspiration.

La présente annexe met en évidence certaines sources d'idées de projets, de programmes de formation et d'autres documents qui peuvent constituer une excellente étape suivante, maintenant que vous avez franchi le cap du *Guide du débutant*.

Bookshelf (Bibliothèque)

Menu Raspberry > Help > Bookshelf

Bookshelf est une application incluse dans le système d'exploitation Raspberry Pi qui vous aide à naviguer, télécharger, et lire les versions numériques des publications de Raspberry Pi Press - y compris une copie de ce Guide du débutant. Il suffit de le charger en cliquant sur l'icône du menu framboise, en choisissant Help, et en cliquant sur Bookshelf, puis de naviguer parmi une gamme de magazines et de livres, tous gratuits à télécharger et à lire à votre guise.



Le Blog Raspberry Pi

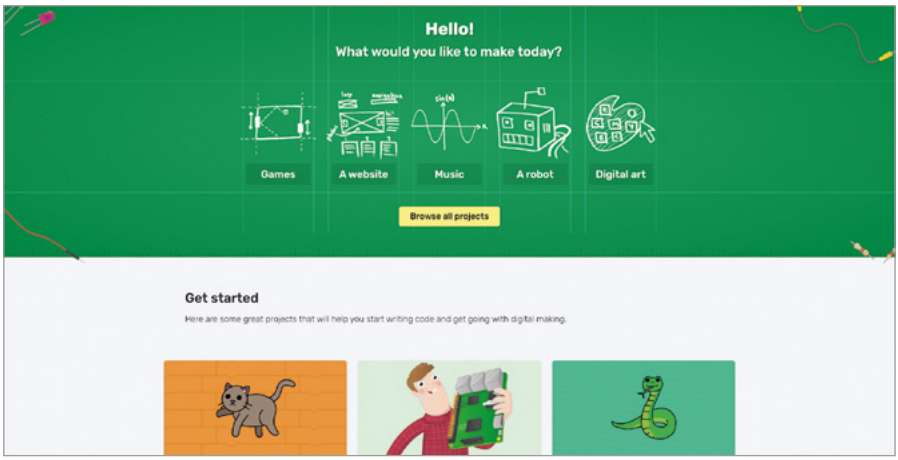
rpf.io/blog

Pour tout savoir sur les nouveautés de Raspberry Pi, votre première étape est notre blog officiel qui englobe tout, des présentations de nouveaux matériels et ressources pédagogiques, à un passage en revue des projets, campagnes et initiatives qui se réalisent au sein de notre communauté. Si vous voulez vous tenir au courant de tout ce qui concerne Raspberry Pi, c'est ici que vous devez être.

Projets Raspberry Pi

rpf.io/projects

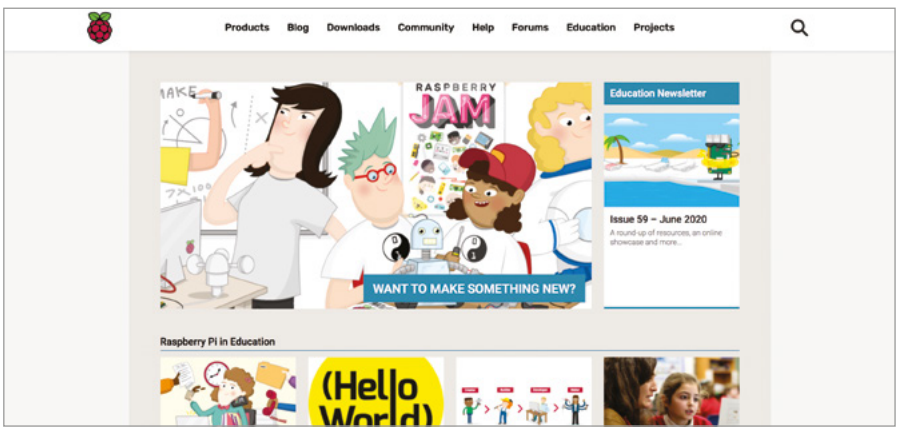
Le site officiel Projets Raspberry Pi propose des tutoriels de projets étape par étape dans différentes catégories, allant de la création de jeux à la musique ou à la construction de votre propre site web ou d'un robot alimenté par Raspberry Pi. La plupart des projets sont également disponibles en plusieurs langues à différents niveaux de difficulté adaptés à tous, des débutants absolus aux créateurs expérimentés.



Raspberry Pi Education

rpf.io/education

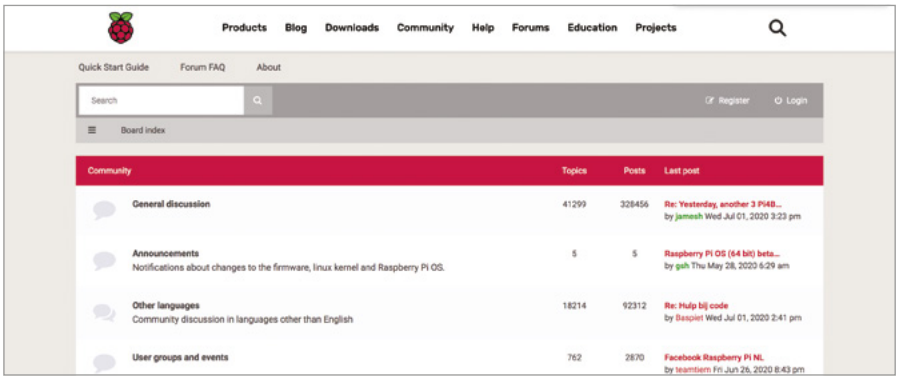
Le site officiel Raspberry Pi Education propose des bulletins d'information, des formations en ligne et des projets destinés aux éducateurs. Le site présente également des liens vers d'autres ressources, notamment le programme de formation Picademy, les programmes de codage du Code Club et du CoderDojo animés par des bénévoles, ainsi que les événements mondiaux Raspberry Jam.



Forums Raspberry Pi

rpf.io/forums

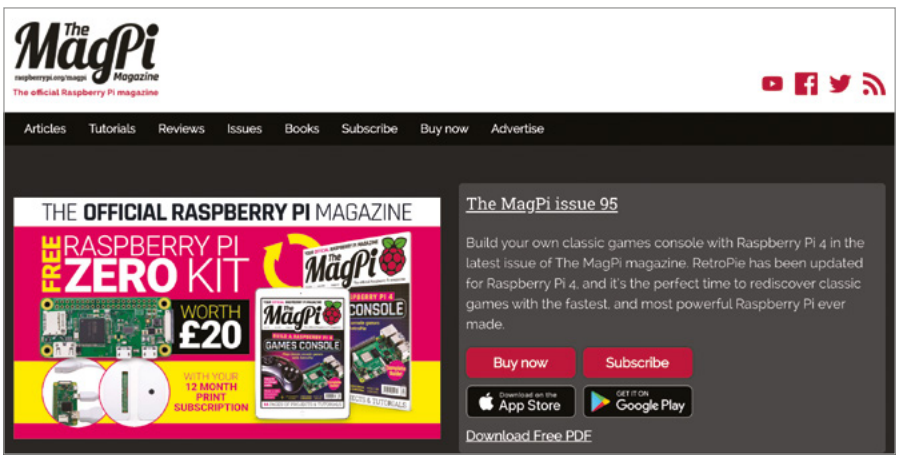
Les forums Raspberry Pi sont un lieu de rencontre pour les fans de Raspberry Pi, où ils peuvent se retrouver et discuter de tout, des problèmes rencontrés par les débutants aux sujets les plus techniques. Il y a même une zone « hors sujet » pour les discussions générales !



Le magazine The MagPi

magpi.cc

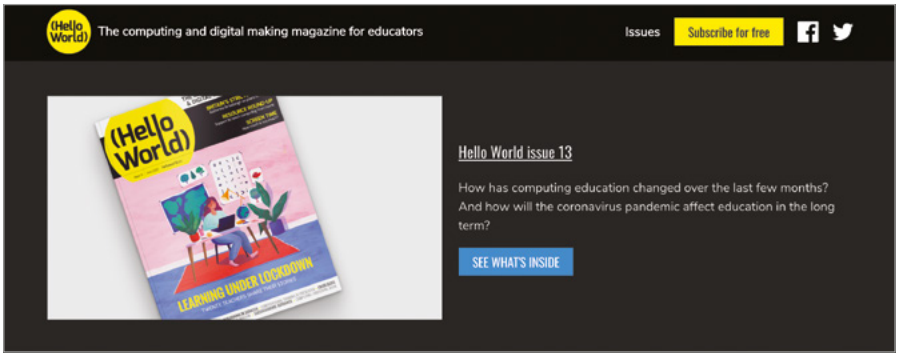
Le magazine officiel de Raspberry Pi, The MagPi, est une publication mensuelle sur papier glacé qui parle de tout, des tutoriels et guides aux critiques et aux nouveautés, et qui est soutenue en grande partie par la communauté mondiale de Raspberry Pi. Des copies sont disponibles dans tous les bons kiosques à journaux et supermarchés, et peuvent également être téléchargées gratuitement sous la licence Creative Commons. The MagPi publie également des livres et des magazines sur des sujets variés, qui peuvent être achetés en format papier ou téléchargés gratuitement.



Magazine Hello World

helloworld.cc

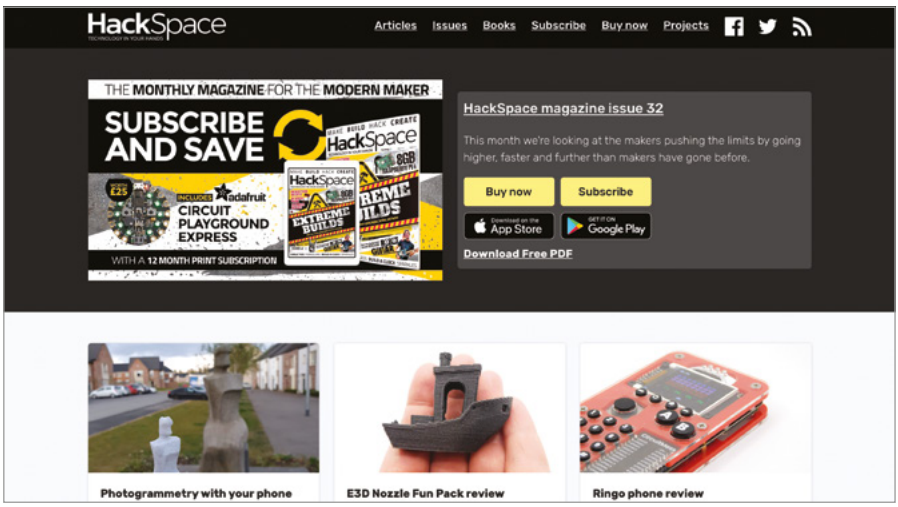
Publié trois fois par an, Hello World est disponible gratuitement pour les enseignants, les bénévoles et les bibliothécaires résidant au Royaume-Uni. Pour tous les autres pays, des copies numériques gratuites peuvent être téléchargées sous la licence Creative Commons, et des abonnements à la version imprimée sont disponibles dans le commerce.



Magazine HackSpace

hsmag.cc

Destiné à un public plus large que celui de The MagPi, le magazine HackSpace offre un regard sur la communauté des fabricants et passe en revue des matériels et logiciels, des tutoriels et des interviews. Si vous souhaitez élargir vos horizons au-delà de Raspberry Pi, le magazine HackSpace est un excellent point de départ. Vous pouvez le trouver en version imprimée dans les supermarchés et les kiosques à journaux ou télécharger gratuitement la version numérique.



Annexe E

L'outil Configuration du Raspberry Pi

L'outil Configuration du Raspberry Pi est un puissant outil qui permet de régler de nombreux paramètres sur votre Raspberry Pi, des interfaces disponibles aux programmes en passant par le contrôle via un réseau. Cependant, cela peut être un peu intimidant pour les nouveaux arrivants, c'est pourquoi cette annexe vous guidera à travers chacun des paramètres en vous expliquant leurs objectifs.

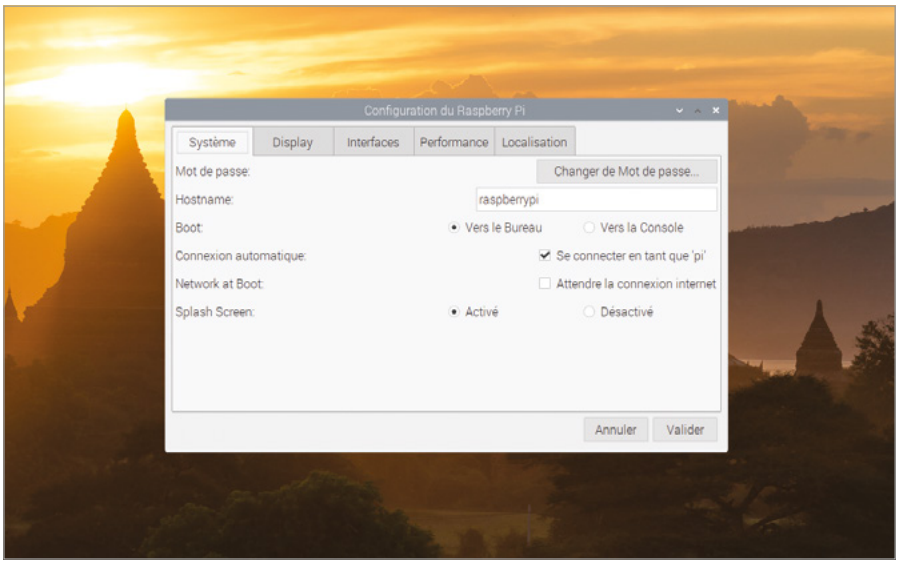
Vous pouvez charger l'outil Configuration du Raspberry Pi à partir du menu de l'icône framboise, sous la catégorie Préférences. Il peut également être exécuté à partir de l'interface en ligne de commande ou du Terminal en utilisant la commande **`raspi-config`**. La présentation de la version en ligne de commande et de la version graphique est différente, les options apparaissant dans différentes catégories, selon la version que vous utilisez ; la présente annexe fait référence à la version graphique.

ATTENTION !

À moins d'avoir la certitude qu'un paramètre doit être modifié, il convient de ne pas manipuler l'outil de configuration Raspberry Pi. Si vous ajoutez du nouveau matériel à votre Raspberry Pi, tel qu'un HAT audio ou un Camera Module, les instructions vous indiqueront quel paramètre modifier ; sinon, les paramètres par défaut doivent généralement être laissés tels quels.

Onglet Système

L'onglet Système contient des options qui contrôlent divers paramètres du système d'exploitation Raspberry Pi OS.



- **Mot de passe :** Cliquez sur le bouton « Changer le mot de passe... » pour définir un nouveau mot de passe pour votre compte utilisateur actuel. Par défaut, il s'agit du compte « pi ».

- **Hostname :** Le nom par lequel un Raspberry Pi s'identifie sur les réseaux. Si vous avez plus d'un Raspberry Pi connectés au même réseau, ils doivent avoir chacun un nom unique pour les distinguer.

- **Boot :** Si vous réglez ce paramètre sur « Vers le bureau » (par défaut), le bureau Raspberry Pi OS sera chargé ; si vous le réglez sur « Vers la console », l'interface de lignes de commande sera chargée comme décrit dans **Annexe C, L'interface en ligne de commande**.

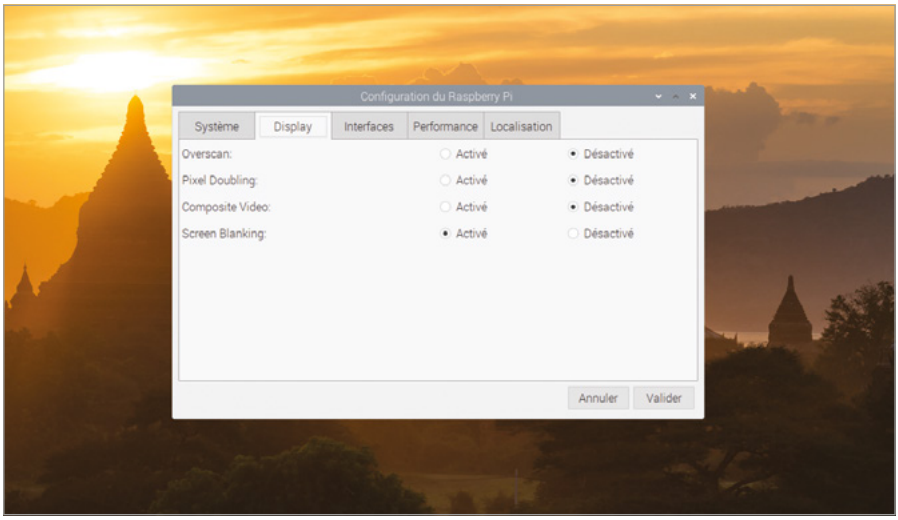
- **Connexion automatique :** Lorsque la case « Se connecter en tant que pi » est cochée (par défaut), Raspberry Pi OS charge le bureau sans que vous ayez à saisir votre nom d'utilisateur et votre mot de passe.

- **Network at Boot :** Lorsque la case « Attendre la connexion réseau » est cochée, Raspberry Pi OS ne se charge pas tant qu'il ne dispose pas de connexion réseau fonctionnelle.

- **Splash Screen :** Lorsque ce paramètre est activé (par défaut), les messages de démarrage de Raspberry Pi OS sont masqués par un écran graphique (Splash Screen).

Onglet Display

L'onglet Display contient des paramètres qui contrôlent l'affichage à l'écran.



- **Overscan :** Ce réglage permet de contrôler si la sortie vidéo de Raspberry Pi comporte ou non des barres noires sur ses bords, pour compenser l'image de nombreux téléviseurs. Si des barres noires apparaissent, réglez sur « Désactivé » ; sinon, laissez sur « Activé ».

- **Pixel Doubling :** Si vous utilisez un écran haute résolution mais de petites dimensions, vous pouvez activer le doublement des pixels pour agrandir tous les éléments afin qu'ils soient plus faciles à voir sur l'écran.

- **Composite Video :** Ce paramètre contrôle la sortie vidéo composite disponible sur la prise jack audio-vidéo (AV) combinée, lorsqu'elle est utilisée avec un adaptateur TRRS (tip-ring-ring-sleeve). Si vous souhaitez utiliser la sortie vidéo composite au lieu d'un câble HDMI, réglez cette option sur « Activé » ; sinon, laissez-la désactivée.

- **Screen Blanking :** Cette option vous permet d'activer et de désactiver la fonction d'effacement d'écran (le délai d'attente qui permet d'éteindre l'écran après quelques minutes).

Onglet Interfaces

L'onglet Interfaces contient les paramètres qui contrôlent les interfaces matérielles disponibles sur Raspberry Pi.

- **Caméra :** Active ou désactive l'interface série de la caméra (CSI), pour une utilisation avec un module caméra de Raspberry Pi.

- **SSH :** Active/désactive l'interface Secure Shell (SSH) ; ce paramètre vous permet d'ouvrir une interface en ligne de commande sur Raspberry Pi depuis un autre ordinateur de votre réseau en utilisant un client SSH.

- **VNC :** Active/désactive l'interface Virtual Network Computing (VNC) ; ce paramètre vous permet d'ouvrir une interface en ligne de commande sur Raspberry Pi depuis un autre ordinateur de votre réseau à l'aide d'un client VNC.

- **SPI :** Active ou désactive la Serial Peripheral Interface (SPI), utilisée pour contrôler certaines extensions matérielles qui se connectent aux points GPIO.

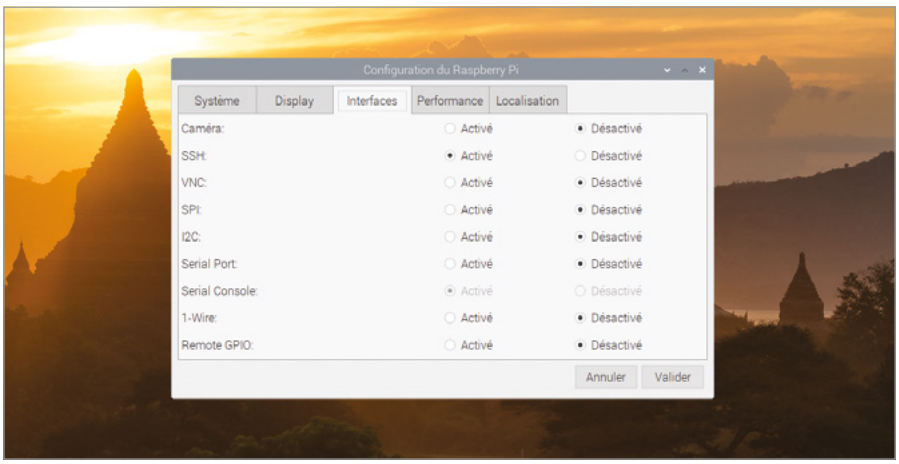
- **I2C :** Active ou désactive l'interface Inter-Integrated Circuit (I²C), utilisée pour contrôler certaines extensions matérielles qui se connectent aux points GPIO.

- **Serial Port :** Active ou désactive le port série de Raspberry Pi, disponible à partir des points GPIO.

- **Serial Console :** Active ou désactive la console série, une interface en ligne de commande disponible sur le port série. Cette option n'est disponible que si le paramètre Serial Port ci-dessus est réglé sur Activé.

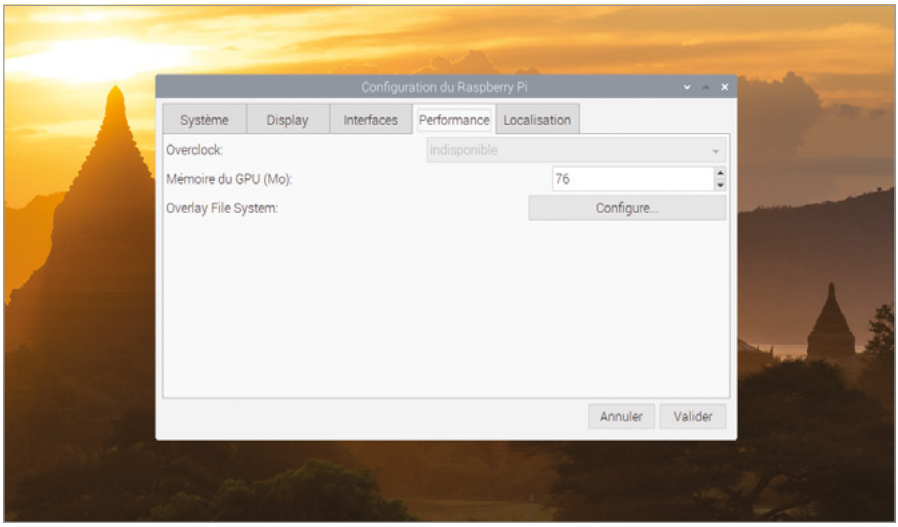
- **1-Wire :** Active ou désactive l'interface 1-Wire, utilisée pour contrôler certaines extensions matérielles qui se connectent aux points GPIO.

- **Remote GPIO :** Active ou désactive un service réseau qui vous permet de contrôler les points GPIO de Raspberry Pi depuis un autre ordinateur de votre réseau à l'aide de la bibliothèque GPIO Zero. Pour plus d'informations sur les GPIO distants, consultez **gpiozero.readthedocs.io**.



Onglet Performance

L'onglet Performance contient des paramètres qui contrôlent la quantité de mémoire disponible et la vitesse d'exécution du processeur de Raspberry Pi.



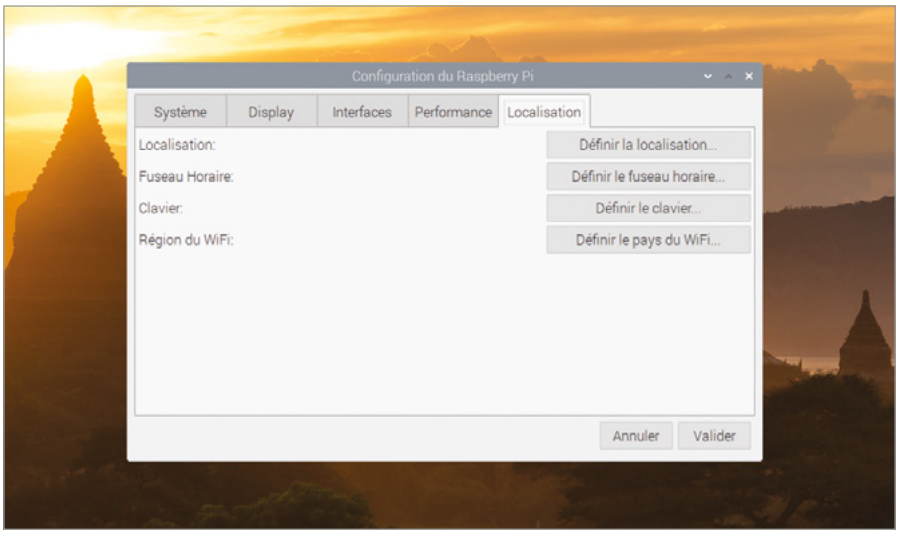
- **Overclock :** Vous permet de choisir parmi une gamme de paramètres qui améliorent les performances de votre Raspberry Pi au prix d'une augmentation de la consommation d'énergie, de la production de chaleur et d'une éventuelle diminution de la durée de vie globale. Cette option n'est pas disponible sur tous les modèles Raspberry Pi.

- **Mémoire du GPU (Mo) :** Permet de définir la quantité de mémoire réservée à l'utilisation du processeur graphique de Raspberry Pi. Des valeurs supérieures à la valeur par défaut peuvent améliorer les performances pour les rendus 3D compliqués et les tâches générales du GPU (GPGPU) au prix d'une réduction de la mémoire disponible pour Raspberry Pi OS ; des valeurs inférieures peuvent améliorer les performances pour les tâches qui nécessitent beaucoup de mémoire au prix d'un ralentissement des performances du rendu 3D, de l'appareil photo et de certaines fonctions de lecture vidéo, voire leur indisponibilité.

- **Overlay File System :** Vous permet de verrouiller le système de fichiers de Raspberry Pi pour que les modifications soient effectuées uniquement sur un disque virtuel en RAM plutôt que d'être inscrites sur la carte mircoSD, de sorte que vous revenez à l'état initial à chaque redémarrage.

Onglet Localisation

L'onglet Localisation contient des paramètres qui contrôlent la région dans laquelle votre Raspberry Pi est appelé à fonctionner, y compris les paramètres de disposition du clavier.



- **Localisation :** Vous permet de choisir votre localisation, un paramètre système qui comprend la langue, le pays et le jeu de caractères. Veuillez noter que le changement de langue ici ne modifiera la langue affichée que dans les applications pour lesquelles une traduction est disponible.

- **Fuseau Horaire :** Vous permet de choisir votre fuseau horaire régional, en sélectionnant une région du monde suivie de la ville la plus proche. Si votre Raspberry Pi est connecté au réseau mais que l'horloge affiche une heure erronée, cela est généralement dû au fait que vous avez sélectionné le mauvais fuseau horaire.

- **Clavier :** Permet de choisir le type de clavier, la langue et la disposition. Si votre clavier ne tape pas les bonnes lettres ou les bons symboles, vous pouvez rectifier la situation ici.

- **Région du WiFi :** Vous permet de définir le pays à des fins de réglementation radio. Veillez à bien sélectionner le pays dans lequel vous utilisez votre Raspberry Pi : si vous sélectionnez un autre pays, il pourrait être impossible de vous connecter aux points d'accès WiFi environnants tout en commettant une infraction aux lois relatives à la radiodiffusion. Le pays doit être défini avant que la radio WiFi puisse être utilisée.

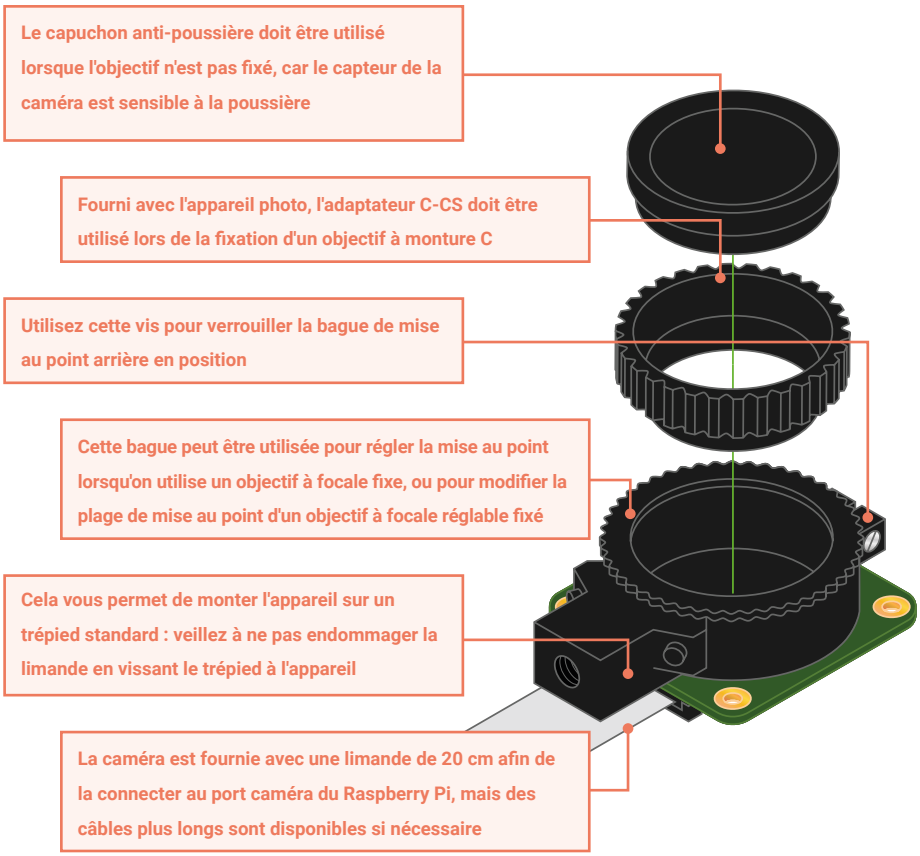
Annexe F

Configuration de la High Quality Camera

La High Quality Camera (caméra haute qualité), abrégé en HQ Camera, peut capturer des images de plus haute résolution que le Camera Module standard. Contrairement à ce dernier, elle ne dispose pas d'un objectif. En revanche, elle peut être utilisée avec n'importe quel objectif standard à monture C ou CS. Les objectifs 6 mm et 16 mm sont disponibles à l'achat avec la caméra pour vous aider à vous lancer.

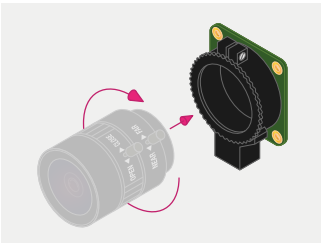
Objectif 6 mm monture CS

Un objectif 6 mm à prix accessible est disponible pour la HQ Camera. Cet objectif convient à des photographies basiques. Il peut également être utilisé pour la macrophotographie car il peut mettre au point des objets à très courte distance.



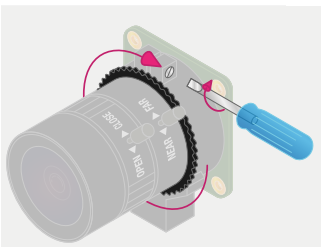
01 Fixation de l'objectif

L'objectif 6 mm est un dispositif à monture CS, aucune bague d'adaptation C-CS n'est donc nécessaire (voir le schéma ci-dessus). Il ne sera pas en mesure d'effectuer une mise au point correcte si l'adaptateur est monté : au besoin, retirez-le. Ensuite, faites tourner l'objectif dans le sens des aiguilles d'une montre jusqu'à la bague de mise au point arrière.

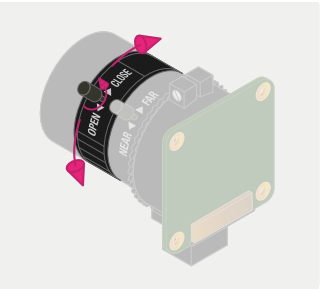


02 Bague de mise au point arrière et vis de blocage

La bague de mise au point arrière doit être vissée à fond pour obtenir une distance focale arrière la plus courte possible. Utilisez la vis de blocage de mise au point arrière pour vous assurer qu'elle se maintienne en position lors du réglage de l'ouverture ou de la mise au point.

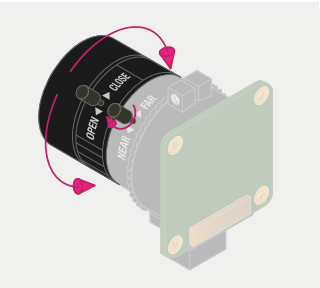


03 Ouverture



Pour régler l'ouverture, soutenez l'appareil photo avec l'objectif tourné vers vous. Tournez l'anneau du milieu tout en maintenant fermement l'anneau extérieur, le plus éloigné de la caméra. Tournez dans le sens des aiguilles d'une montre pour fermer l'ouverture et réduire la luminosité de l'image. Tournez dans le sens inverse des aiguilles d'une montre pour augmenter l'ouverture. Une fois que vous êtes satisfait du niveau de luminosité, serrez la vis sur le côté de l'objectif pour bloquer l'ouverture.

04 Mise au point



Tout d'abord, verrouillez la bague de mise au point intérieure, identifiée par « NEAR (PRÊT) ◀▶ FAR (LOIN) », en position en serrant la vis. Ensuite, soutenez l'appareil photo avec l'objectif éloigné de vous. Saisissez les deux bagues extérieures de l'objectif et tournez-les toutes les deux dans le sens des aiguilles d'une montre jusqu'à ce que l'image soit nette (ce qui correspond à quatre ou cinq tours complets). Pour régler la mise au point, tournez les deux anneaux extérieurs dans le sens des aiguilles d'une montre pour mettre au point un objet proche. Tournez-les dans le sens inverse des aiguilles d'une montre pour mettre au point un objet distant. Il se peut que vous deviez ensuite régler à nouveau l'ouverture.

Objectif 16 mm monture C

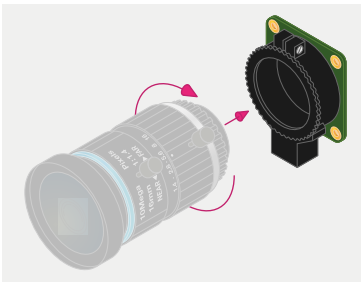
L'objectif 16 mm offre une image de meilleure qualité que l'objectif 6 mm. Son angle de vue étroit est plus adapté à l'observation d'objets éloignés.

01 Fixation de l'adaptateur C-CS

Assurez-vous que l'adaptateur C-CS fourni avec la caméra HQ est bien fixé à l'objectif 16 mm. L'objectif est un dispositif à monture C dont la mise au point arrière est donc plus longue que l'objectif 6 mm et pour lequel un adaptateur est nécessaire.

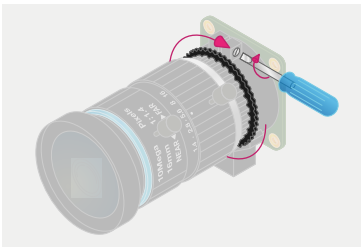
02 Fixation de l'objectif à la caméra

Faites tourner l'objectif 16 mm et l'adaptateur C-CS dans le sens des aiguilles d'une montre jusqu'à la bague de mise au point arrière.



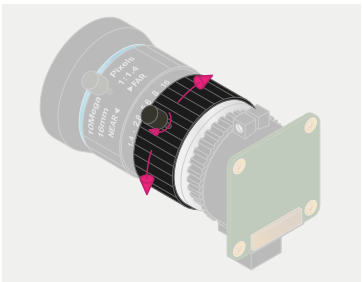
03 Bague de mise au point arrière et vis de blocage

La bague de mise au point arrière doit être vissée à fond. Utilisez la vis de blocage de mise au point arrière pour vous assurer qu'elle se maintienne en position lors du réglage de l'ouverture ou de la mise au point.



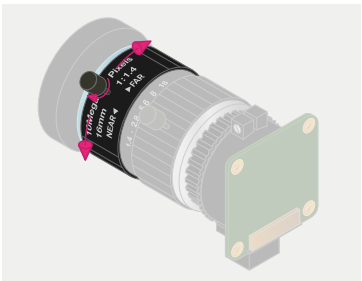
04 Ouverture

Pour régler l'ouverture, soutenez l'appareil photo avec l'objectif tourné vers vous. Tournez l'anneau intérieur, le plus proche de la caméra, tout en maintenant la caméra immobile. Tournez dans le sens des aiguilles d'une montre pour fermer l'ouverture et réduire la luminosité de l'image. Tournez dans le sens inverse des aiguilles d'une montre pour augmenter l'ouverture. Lorsque vous êtes satisfait du niveau de luminosité, serrez la vis sur le côté de l'objectif pour bloquer l'ouverture en position.



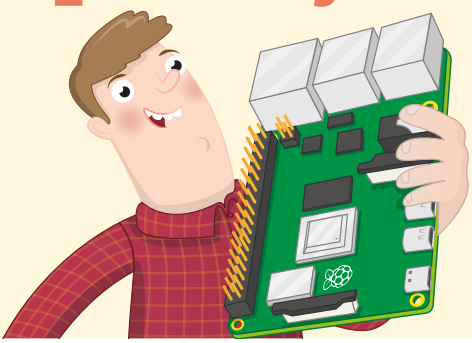
05 Mise au point

Pour régler la mise au point, soutenez l'appareil photo avec l'objectif tourné vers l'extérieur. Pour régler la mise au point, tournez les deux anneaux extérieurs dénommés « NEAR (PRÊT) ◀ ▶ FAR (LOIN) » dans le sens contraire aux aiguilles d'une montre pour mettre au point un objet proche. Tournez-les dans le sens des aiguilles d'une montre pour mettre au point un objet distant. Il se peut que vous deviez ensuite régler à nouveau l'ouverture.



Annexe G

Spécifications de Raspberry Pi

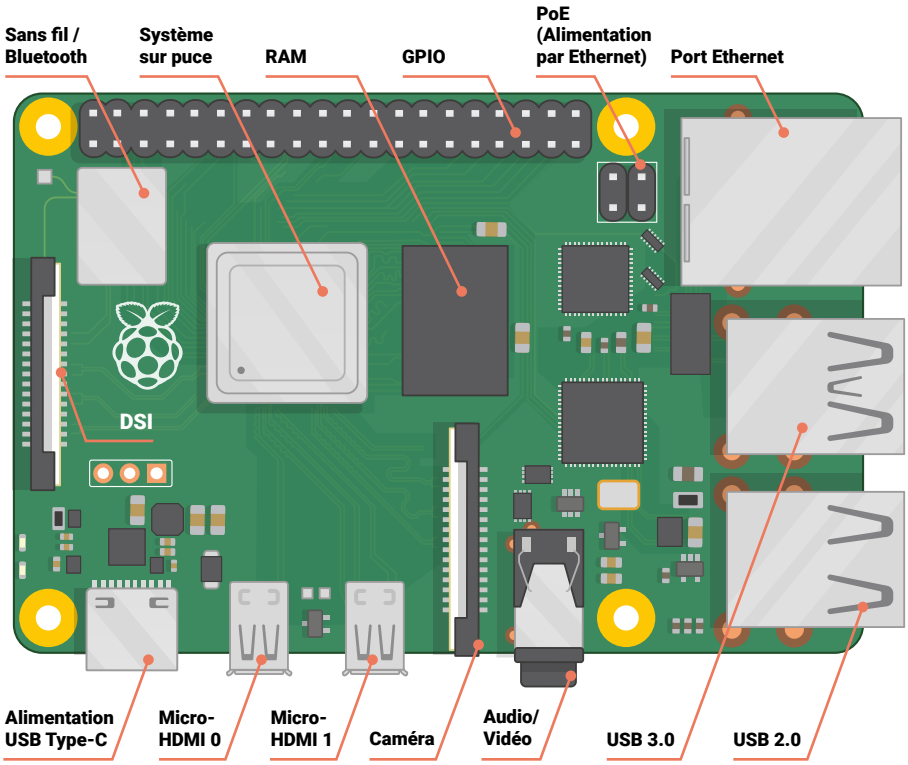


Le terme spécifications désigne les différentes composantes et caractéristiques d'un ordinateur. En analysant les spécifications, vous pouvez obtenir toutes les informations dont vous avez besoin pour comparer deux ordinateurs. Ces spécifications peuvent sembler confuses au premier abord, car elles sont très techniques et il n'est pas forcément nécessaire de bien les connaître pour utiliser un Raspberry Pi, mais nous avons décidé de les inclure ici pour satisfaire la curiosité de nos lecteurs.

Le système sur puce de Raspberry Pi 4 Model B et de Raspberry Pi 400 est un Broadcom BCM2711B0, tel que vous le verrez écrit sur son couvercle métallique si vous regardez d'assez près (sur Raspberry Pi 4). Il comprend une unité centrale (CPU) de quatre cœurs de 64 bits ARM Cortex-A72, chacun fonctionnant à 1,5 GHz ou 1,8 GHz (1,5 ou 1,8 milliard de cycles par seconde), et une unité de traitement graphique (GPU) Broadcom VideoCore VI (Six) fonctionnant à 500 MHz (500 millions de cycles par seconde) pour les tâches vidéo et pour les tâches de rendu 3D telles que les jeux.

Ce système sur puce est connecté à une mémoire RAM (random-access memory ou mémoire vive) LPDDR4 (Low-Power Double-Data-Rate 4) qui fonctionne à 3 200 MHz (trois mille deux cent millions de cycles par seconde) de 2 Go, 4 Go ou 8 Go (deux, quatre ou huit milliards d'octets) (4 Go sur Raspberry Pi 400). Cette mémoire est partagée entre le processeur central et le processeur graphique. La fente pour carte microSD prend en charge jusqu'à 512 Go (512 milliards d'octets) de stockage.

Le port Ethernet prend en charge des connexions pouvant atteindre un gigabit (1 000 Mbps, 1000-Base-T), tandis que la radio prend en charge les réseaux WiFi 802.11ac fonctionnant sur les bandes de fréquences 2,4 GHz et 5 GHz, ainsi que les connexions Bluetooth 5.0 et Bluetooth Low Energy (BLE).



Sous la forme d'une liste, les spécifications de Raspberry Pi 4 se présentent comme suit :

- **CPU :** ARM Cortex-A72 64 bits quadri-cœur à 1,5 GHz
- **GPU :** VideoCore VI à 500 MHz
- **RAM :** 1 Go, 2 Go ou 4 Go de LPDDR4
- **Réseau :** Ethernet Gigabit, WiFi bi-bande 802.11ac, Bluetooth 5.0, Bluetooth Low Energy
- **Sorties audio/vidéo :** Prise AV analogique 3,5 mm, 2 micro-HDMI 2.0
- **Connectivité périphérique :** 2 ports USB 2.0, 2 ports USB 3.0, interface série de la caméra, interface série de l'écran (DSI)
- **Stockage :** microSD, jusqu'à 512 Go
- **Alimentation :** 5 volts à 3 ampères via USB Type-C
- **Fonctionnalités supplémentaires :** Connexion GPIO de 40 points, compatibilité alimentation par Ethernet (matériel supplémentaire nécessaire)

Les spécifications de Raspberry Pi 400 sont :

- **CPU :** ARM Cortex-A72 64 bits quadri-cœur à 1,8 GHz
- **GPU :** VideoCore VI à 500 MHz
- **RAM :** 4 Go de LPDDR4
- **Réseau :** Ethernet Gigabit, WiFi bi-bande 802.11ac, Bluetooth 5.0, Bluetooth Low Energy
- **Sorties audio/vidéo :** 2 micro-HDMI 2.0
- **Connectivité périphérique :** 1 port USB 2.0, 2 ports USB 3.0
- **Stockage :** microSD, jusqu'à 512 Go (16 Go fournis)
- **Alimentation :** 5 volts à 3 ampères via USB Type-C
- **Fonctionnalités supplémentaires :** Connexion GPIO de 40 points

Annexe H

Instructions de sécurité et d'utilisation de Raspberry Pi

Raspberry Pi

Conçu et distribué par
Raspberry Pi Trading Ltd
Maurice Wilkes Building
Cowley Road
Cambridge
CB4 0DS
ROYAUME-UNI
www.raspberrypi.org

Raspberry Pi Conformité réglementaire et informations sur la sécurité

Raspberry Pi 4 Model B
FCC ID : 2ABCB-RPI4B
IC ID : 20953-RPI4B

Raspberry Pi 400
FCC ID : 2ABCB-RPI400
IC ID : 20953-RPI400

IMPORTANT : Consultez les instructions d'installation avant de procéder au raccordement sur **www.raspberrypi.org/safety**

ATTENTION : Cancer et pathologies du système reproductif – www.P65Warnings.ca.gov.

Toutes les informations réglementaires et les certificats peuvent être consultés à l'adresse suivante **www.raspberrypi.org/compliance**

IFETEL : 2019LAB-ANCE4957
Numéro de certificat du Raspberry Pi 4 Model B.

N° d'enregistrement TRA
ER73381/19
Numéro de certificat du Raspberry Pi 4 Model B.

CCAO19LP1120T2
Numéro de certificat du Raspberry Pi 4 Model B.

NTC
Type approuvé :
No : ESD-GEC-1920098C
Numéro de certificat du Raspberry Pi 4 Model B.

TA-2019/750 APPROUVÉ
Numéro de certificat du Raspberry Pi 4 Model B.

LE GUIDE OFFICIEL du débutant Raspberry Pi

Raspberry Pi est un petit ordinateur intelligent de fabrication britannique dont le potentiel est illimité. Réalisé à partir de la même technologie que celle utilisée pour les smartphones, Raspberry Pi est conçu pour vous aider à apprendre le codage, à découvrir le fonctionnement des ordinateurs et à réaliser vos propres créations incroyables. Le but de ce livre est de vous montrer à quel point il est facile de se lancer.

Il vous expliquera comment :

- Configurer votre Raspberry Pi, installer son système d'exploitation et commencer à utiliser cet ordinateur parfaitement fonctionnel.
- Lancer des projets de codage, en suivant des instructions détaillées et utilisant les langages de programmation Scratch 3 et Python.
- Expérimenter la connexion de composants électroniques et s'amuser en créant des projets étonnants.

raspberrypi.org

9 781912 047932

Raspberry Pi PRESS