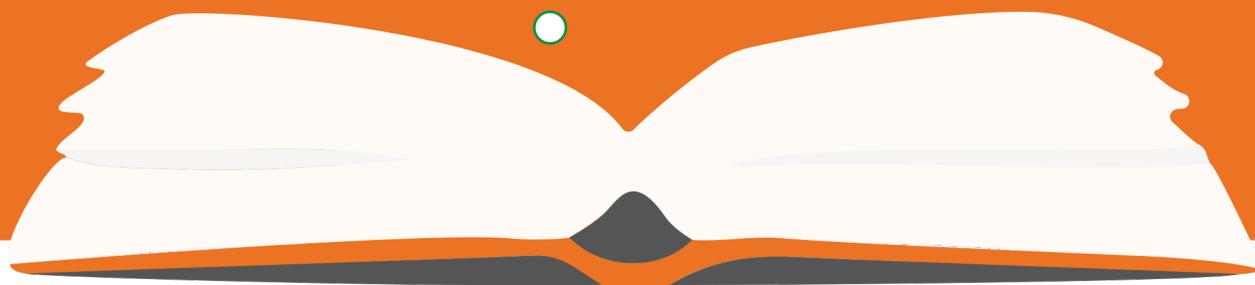


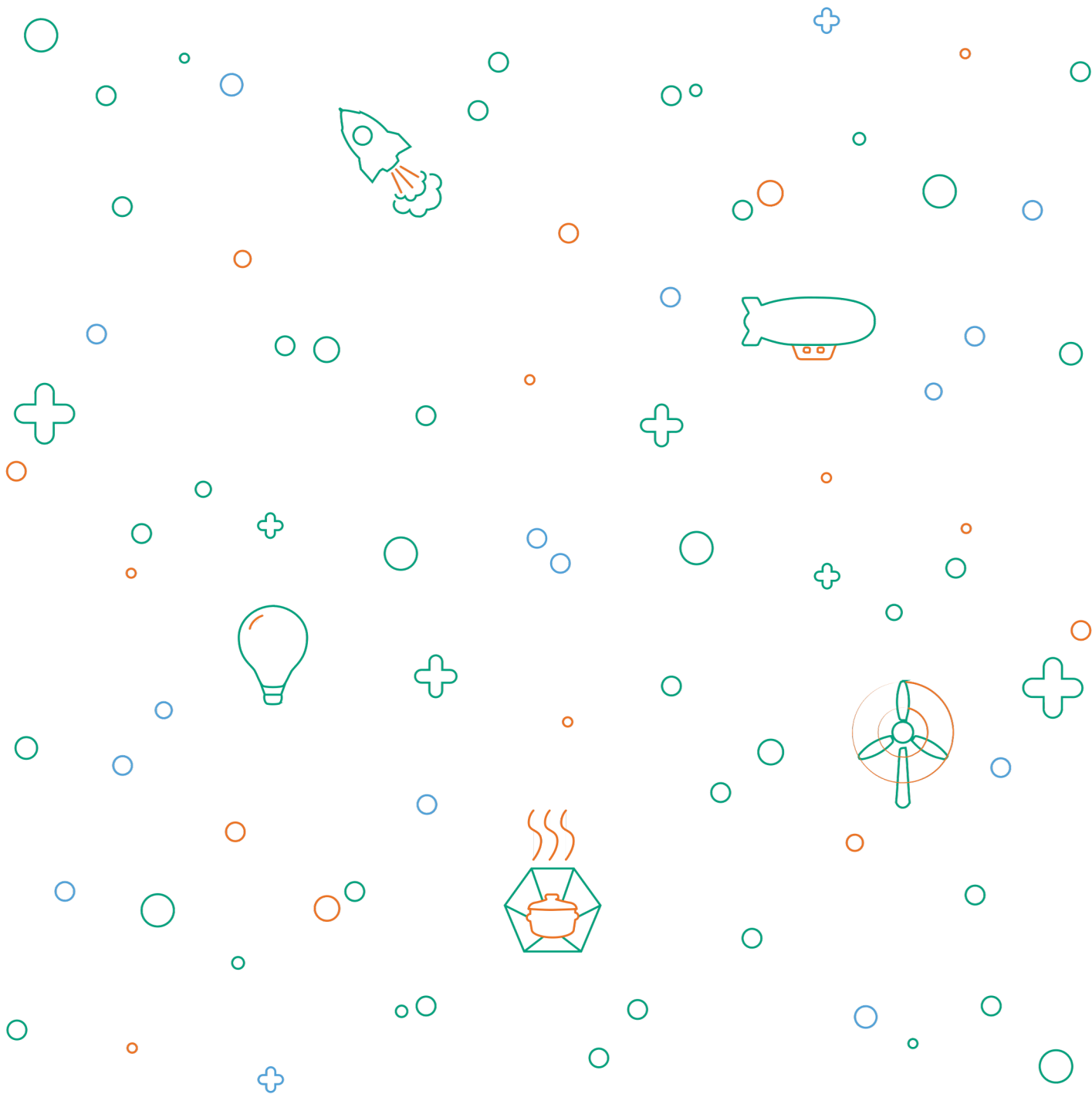
partenariat avec



VOYAGER SUR MARS !



GUIDE D'UTILISATION
ROBOT MARTIEN





LIVRET PÉDAGOGIQUE

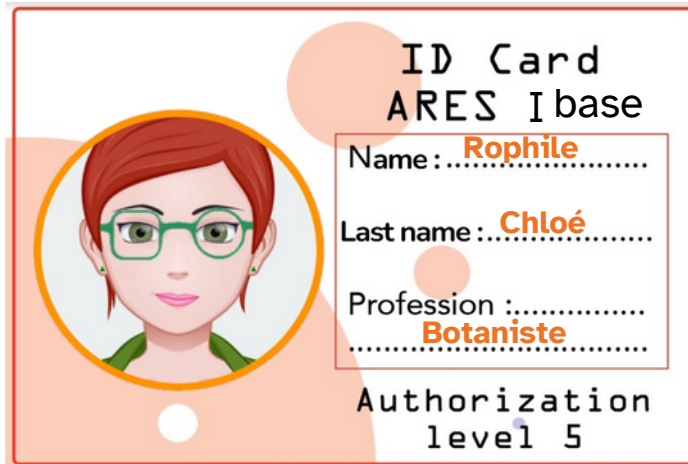
ROBOT MARTIEN

Édition originale rédigée par @P7i7Plum3
et Damien Vallot avec le soutien de l'équipe
"Survive on Mars"

Mise à jour du livret 04-2025

Mise en page et illustration par
Laura Venezia et Diana Khalipina
Tous droits d'auteurs réservés

• Edition 2025 •
Imprimé en Italie



**NOS DEUX COLONS
MARTIENS PRÊTS À
RELEVER TOUS LES DÉFIS !**



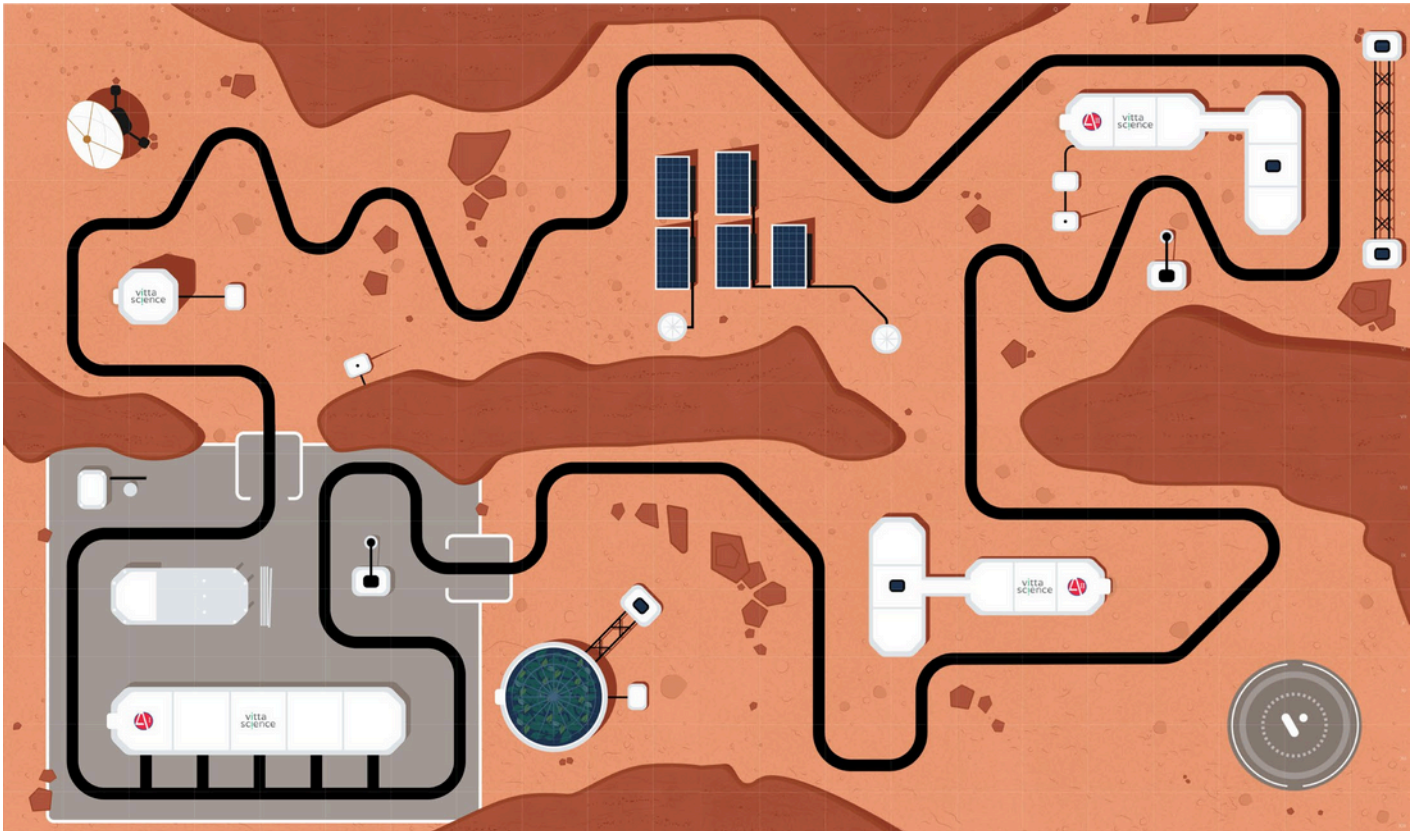
Nous sommes en 2041.

Le succès de la mission sur Mars d'Ingenuity en 2021 est encore dans toutes les mémoires. Ce robot volant de la NASA, déployé aux côtés du rover Perseverance, avait validé le concept de robot attaché à un centre de pilotage sur Mars.

Suite à cette réussite, de nombreuses missions ont été réalisées avec succès. Depuis l'installation sur Mars de la base souterraine Arès I en 2039, nous cherchons à coloniser sa surface. Pour cela, une nuée de robots y a été déployée en 2040 afin de patrouiller, analyser et construire la future base Arès II prévue pour 2042. L'ASM (Agence Spatiale Martienne), basée sur Terre, pilote l'ensemble du projet avec l'aide des premiers colons martiens sur place.

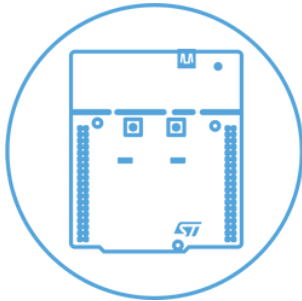
Dans les missions, nous aurons affaire à deux de nos colons les plus émérites de la base Arès I. En effet, suite à un incident survenu sur la base, ceux-ci font face à plusieurs défis !

LE THÉÂTRE DES OPÉRATIONS

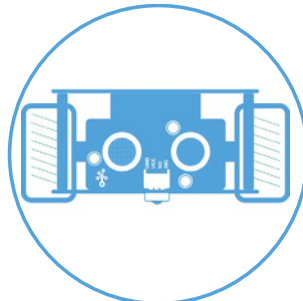


MATÉRIEL NÉCESSAIRE À LA CONSTRUCTION ET À L'UTILISATION DU KIT ROBOT MARTIEN

Contenu du KIT :



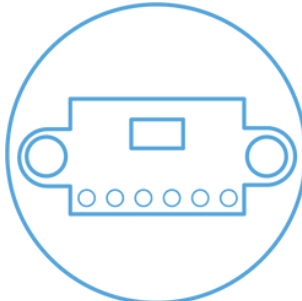
Carte NUCLEO-L476RG



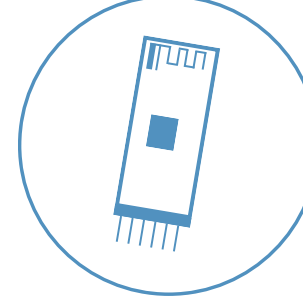
Robot DonutBot



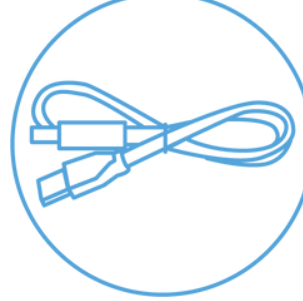
Piste pour robot



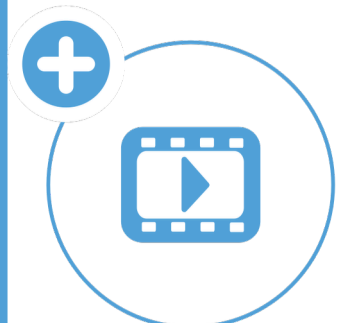
Capteur de distance VL53L0X



Module Bluetooth HM10



Éléments pour le montage



Ressources numériques en ligne



Prévoir un ordinateur

AVERTISSEMENTS CONCERNANT L'UTILISATION DU KIT



ATTENTION !

Présence de petits éléments, ne pas ingérer (risque d'étouffement).



Age minimum

Ne convient pas aux enfants de moins de 7 ans.

CONSIGNES POUR BIEN TRIER

L'infographie suivante détaille les consignes de tri des différents éléments du kit.
Pour plus d'informations, rendez-vous sur le site www.consignesdetri.fr



SOMMAIRE

PAGE
14

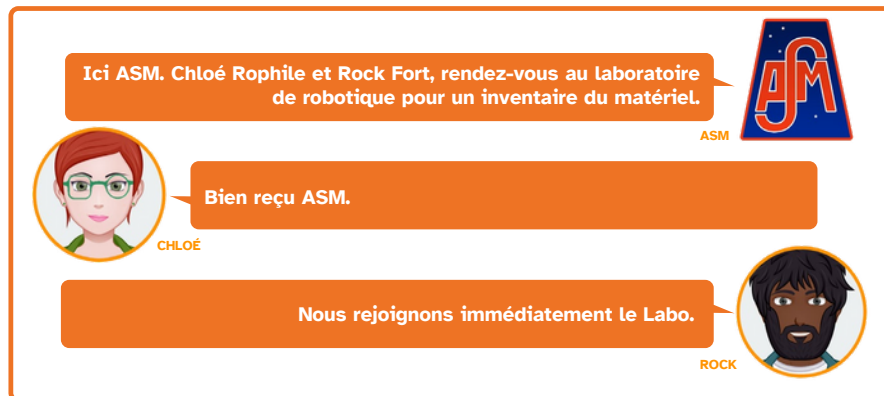
Le kit robot martien et l'environnement de programmation

- Présentation du matériel et de l'interface Vittascience
- Vue d'ensemble du robot DonutBot
- Vue d'ensemble de la carte NUCLEO-L476RG
- Instructions de montage du robot DonutBot
- Programmation du robot DonutBot
- Chargement des batteries du robot DonutBot

PAGE
26

Mission 1 : Allumer les LED et interagir avec le robot

- Mission 1-1 : Allumer les LED du robot
- Mission 1-2 : Utilisation des boutons





PAGE
28

PAGE
33

Mission 2 : Pilotage des moteurs

- **Mission 2-1 : Faire avancer le robot**
- **Mission 2-2 : Apprendre à pivoter**
- **Mission 2-3 : Suivre une trajectoire**

Mission 3 : Tester le détecteur d'obstacles

- **Mission 3-1 : Découverte du capteur de distance Time of Flight (ToF)**
- **Mission 3-2 : Découverte du capteur de distance à ultrasons (en option)**
- **Mission 3-3 : Capteurs de distance et moteurs**

PAGE
36



PAGE
39

Mission 4 : Pilotage temps réel avec le module bluetooth

- Mission 4-1 : Vérification de la communication
- Mission 4-2 : Test en conditions réelles “martiennes” : la télécommande

Mission 5 : Déplacement en mode case

- Mission 5-1 : Déplacement en mode cases
- Mission 5-2 : Détection d'obstacles avant déplacement

Arès I à l'écoute

ARÈS I

ASM

Chloé Rophile et Rock Fort viennent d'être désignés pour remplacer votre programmeur-roboticien actuellement en réanimation suite à l'incident de dépressurisation. Ils devront mettre de côté leurs missions d'informaticienne et de mécanicien. Qu'ils se tiennent prêts à recevoir nos instructions. Fin de transmission.

Bien reçu. Fin de transmission.

ARÈS I

PAGE
43

PAGE
47

Mission 6 : Suiveur de ligne

- Mission 6-1 : Suivre une ligne
- Mission 6-2 : Suivre une ligne avec 3 capteurs
- Mission 6-3 : Suivre une ligne et éviter les obstacles

Mission 7 : Mise en situation complexe

- Mission 7 : Enfin la routine !

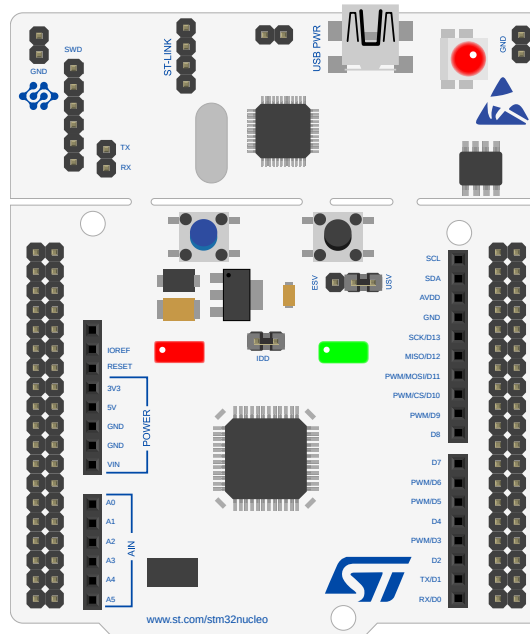
Présentation du matériel et de l'interface Vittascience

🕒 15 min

Le kit "robot martien" comprend tous les éléments nécessaires pour réaliser un robot capable d'explorer Mars de façon autonome. Il est livré avec une carte NUCLEO-L476RG développée par STMicroelectronics, équipée d'un microcontrôleur STM32L476RG.

• Aperçu de la carte NUCLEO-L476RG

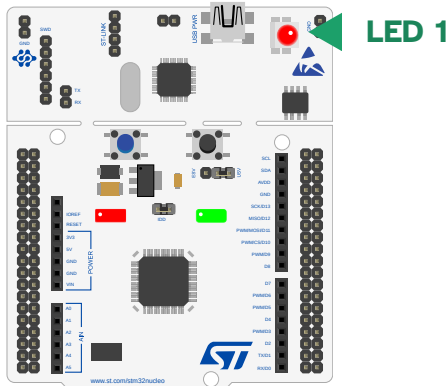
L'illustration suivante montre la carte NUCLEO-L476RG, qui peut aussi servir de support à différents montages avec d'autres composants, notamment des modules Grove.



Elle se programme via l'interface Vittascience en MicroPython. Avant la première utilisation, **il est nécessaire de charger un firmware dans son microcontrôleur afin que la carte puisse exécuter des programmes en Python** (voir encadré).

PROCEDURE DE CHARGEMENT DU FIRMWARE MICROPYTHON

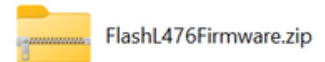
Pour programmer la NUCLEO-L476RG en MicroPython, par code ou par blocs depuis le site Vittascience, il faut charger le firmware adéquat dans son microcontrôleur STM32L476RG. Cette étape doit être réalisée une seule fois, à la réception du kit.



Sur Mac OS



Sur Windows



Voici les étapes à suivre :

1 • Brancher le câble USB sur le port mini USB de la carte. La LED s'allume en rouge.

Sur Mac OS :

2 • Télécharger le firmware (.bin) depuis le site Vittascience puis glisser-déposer celui-ci dans votre carte, qui est apparue comme une clé USB nommée "NOD_L476RG". Attention : ne pas décompresser le fichier.

Sur Windows :

2 • Télécharger l'archive .zip contenant le firmware au format .bin et le script Drive.bat. Avant de le lancer, il faut le modifier (avec le logiciel bloc-note par exemple) **C:\Users\USER_NAME\Downloads\FlashL476Firmware** il faut remplacer **USER_NAME** par votre nom d'utilisateur de votre session Windows. Une fois cela fait, vous pouvez lancer le .bat pour flasher votre carte.

3 • Lorsque le téléchargement est terminé, la LED 1 s'allume en vert.

4 • Débrancher le câble USB.

5 • Connecter la carte à l'ordinateur, la LED 1 est désormais allumée en rouge. La carte est bien alimentée.

6 • Utiliser l'interface Vittascience pour programmer votre carte.

Un problème, une question ? Nous sommes là pour vous répondre : contact@vittascience.com

• Programmation de la carte 🍷 à l'appréciation de l'encadrant

Nous détaillons ici le fonctionnement de l'interface de programmation en ligne [Vittascience.com](https://vittascience.com). Il est également possible de programmer la carte à l'aide des logiciels Arduino (langage C++).

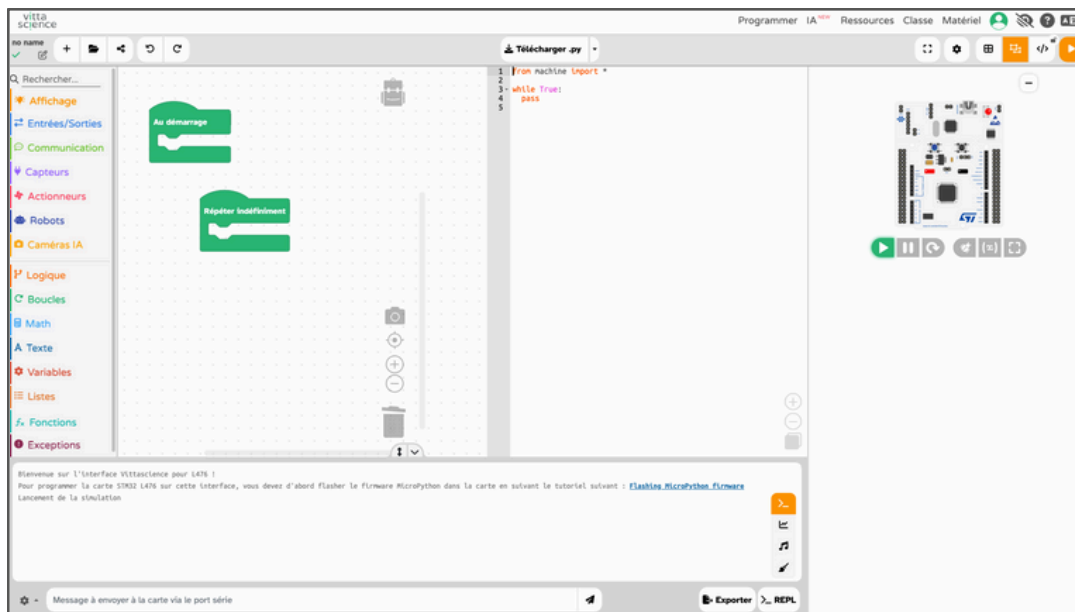
1 • Création d'un compte Vittascience

Nous vous conseillons de vous créer un compte sur notre site. Celui-ci n'est en aucun cas nécessaire pour profiter de votre kit, mais il vous permettra de sauvegarder et de partager vos programmes, ressources et retours d'expériences.

Pour cela, rendez-vous sur [Vittascience.com](https://vittascience.com) et cliquez sur l'icône verte en haut à droite pour vous inscrire.

2 • L'interface

L'interface permet de programmer en bloc avec une transcription en parallèle en langage Python.



Attention : La carte NUCLEO-L476RG est nécessaire pour exécuter le programme sur le robot. Retrouvez sur le site [Vittascience.com](https://vittascience.com) des ressources et des programmes pour apprendre à programmer avec.

Transférer le programme vers la carte : ⚡ **Téléverser** le code est exécuté sur la carte dès la fin du transfert.

Sélection du port : Lorsque vous connectez la carte à l'ordinateur, l'interface détecte automatiquement sur quel port elle est connectée. La fenêtre qui s'affiche permet de sélectionner le bon port si plusieurs cartes sont connectées à l'ordinateur.

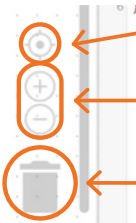
Annuler ou rétablir :  pour revenir en arrière dans les actions exécutées.

Démarrer un nouveau projet :  pour lancer un nouveau projet vierge, cliquez sur ce bouton.

Sauvegarder le projet :  pour enregistrer votre projet afin de le sauvegarder, cliquer sur ce bouton. Grâce à ce bouton, vous pouvez également partager le programme avec la communauté.

Ouvrir un projet existant :  si vous souhaitez ouvrir des programmes déjà réalisés, ou bien faire travailler vos élèves sur une trame que vous avez créée, cliquez sur ce bouton.

Coder en Python :  si vous souhaitez coder directement en Python.

Accès rapide :  Recentrer le programme sur la zone de travail

 Zoom + et -

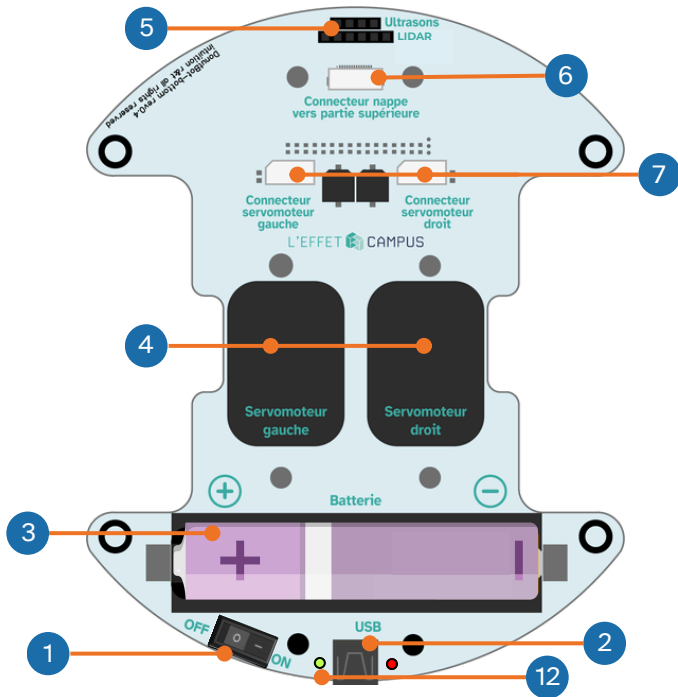
 Supprimer des blocs



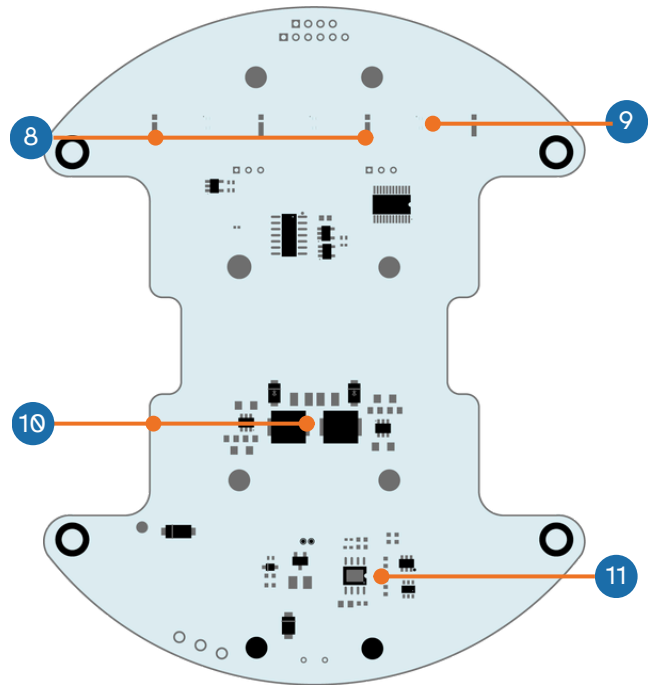
Conseil : Cette interface de programmation est conçue pour être très simple d'utilisation. Vous disposez sur le site de ressources supplémentaires pour la prendre en main.

Vue d'ensemble du robot DonutBot

• Figure 1 : Base inférieure, face avant



• Figure 2 : Base inférieure, face arrière

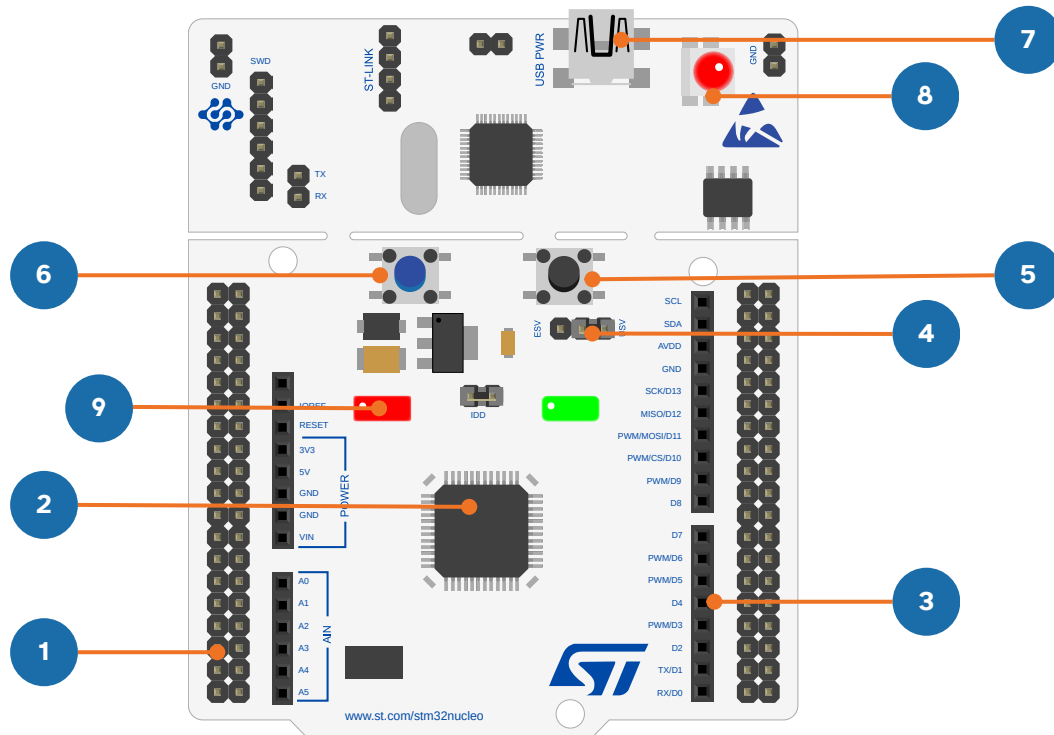


- 1 • Interrupteur d'alimentation On / Off
- 2 • Port USB 5V pour le chargement de la batterie
- 3 • Porte-pile pour pile rechargeable Li-Ion 18650. Attention, l'orientation de ce boîtier peut être inversée selon les versions du châssis.
- 4 • Servomoteur
- 5 • Support pour capteur de distance (à ultrasons ou ToF)
- 6 • Connecteur pour nappe vers la partie supérieure du robot

- 7 • Connecteur pour servomoteur
- 8 • LED blanche pour le suivi de ligne
- 9 • Capteur de couleur
- 10 • Système de gestion de la tension du robot
- 11 • Système de contrôle de charge de la batterie
- 12 • Indicateur d'alimentation du robot (LED verte) et indicateur de charge de la batterie (LED rouge)

Vue d'ensemble de la carte NUCLEO-L476RG

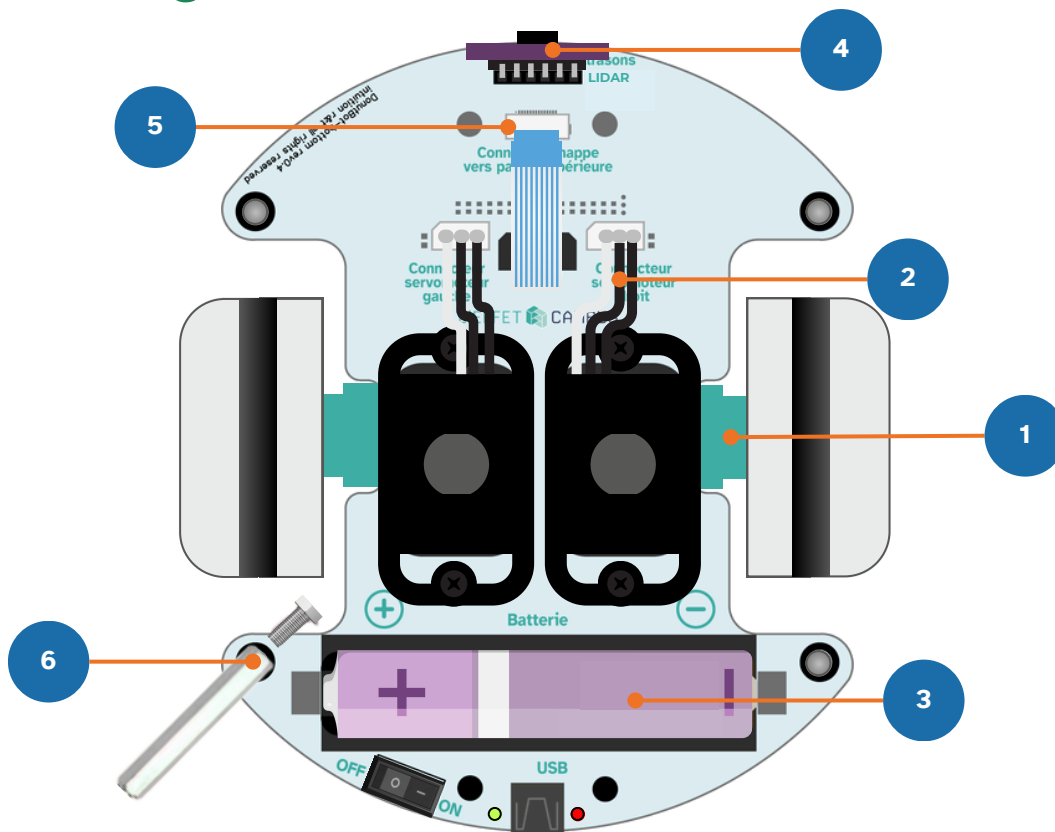
• Figure 5 : NUCLEO-L476RG (face avant)



- 1 • Connecteur pour extension ST-Morpho (mâle)
- 2 • Puce (microcontrôleur) STM32L476RG
- 3 • Connecteur d'extension Arduino (femelle)
- 4 • Connecteur d'alimentation et de programmation (mâle)
- 5 • Bouton de réinitialisation (RESET)

- 6 • Bouton utilisateur (USER)
- 7 • Connecteur mini-USB
- 8 • LED indiquant le statut de l'interface de programmation et débogage ST-LINK
- 9 • LED utilisateur programmable

Assemblage du robot DonutBot ⌚ 15 min



1 • Fixer les roues sur les servomoteur, à l'aide des vis.

2 • Connecter les fiches des servomoteurs en respectant les indications inscrites.

3 • Connecter la batterie si nécessaire.

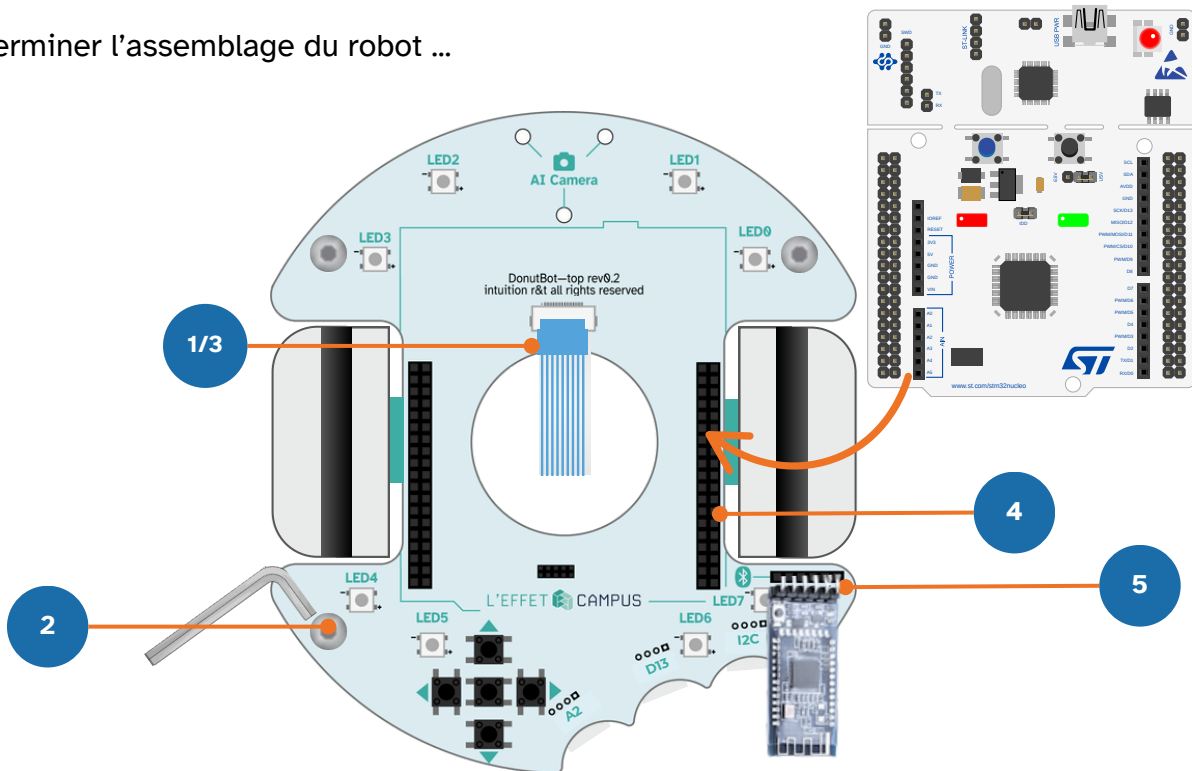
Attention, à la polarité de la batterie !

4 • Positionner le capteur de distance dans son emplacement dédié. Le module de mesure est à orienter vers l'extérieur.

5 • Fixer une extrémité de la nappe dans son connecteur (il faut soulever le loquet puis le rabaisser pour verrouiller la nappe).

6 • Commencer à fixer une partie des entretoises (1 vis + entretoise) dans les trous dédiés.

Pour terminer l'assemblage du robot ...



1 • Faire passer la nappe dans la zone évidée centrale de la partie supérieure et la poser sur les entretoises. La face supérieure avec les connecteurs de la carte NUCLEO-L476RG doit être positionnée vers le haut.

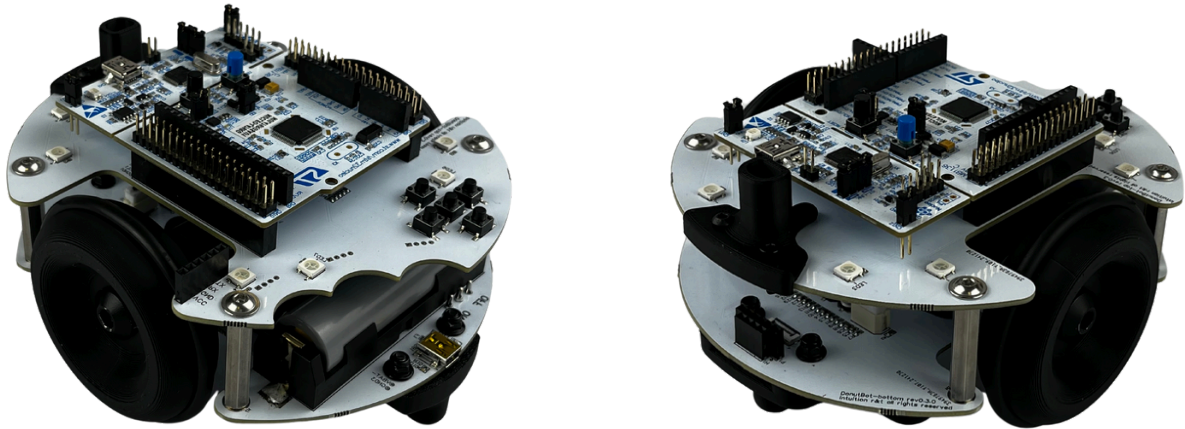
2 • Visser la partie supérieure sur les entretoises à l'aide des 4 vis restantes.

3 • Connecter la nappe dans son logement dédié sur la partie supérieure du robot (il faut soulever le loquet puis le rabaisser pour verrouiller la nappe).

4 • Positionner la carte NUCLEO-L476RG sur son emplacement en respectant le sens (s'aider du tracé dessiné sur le châssis du robot). Attention à bien vérifier le bon alignement des connecteurs !

5 • Fixer le module bluetooth, si nécessaire, dans son connecteur. Attention à respecter le sens ! Les inscriptions VCC et GND doivent correspondre deux à deux entre le module et le châssis.

Photos du robot martien assemblé



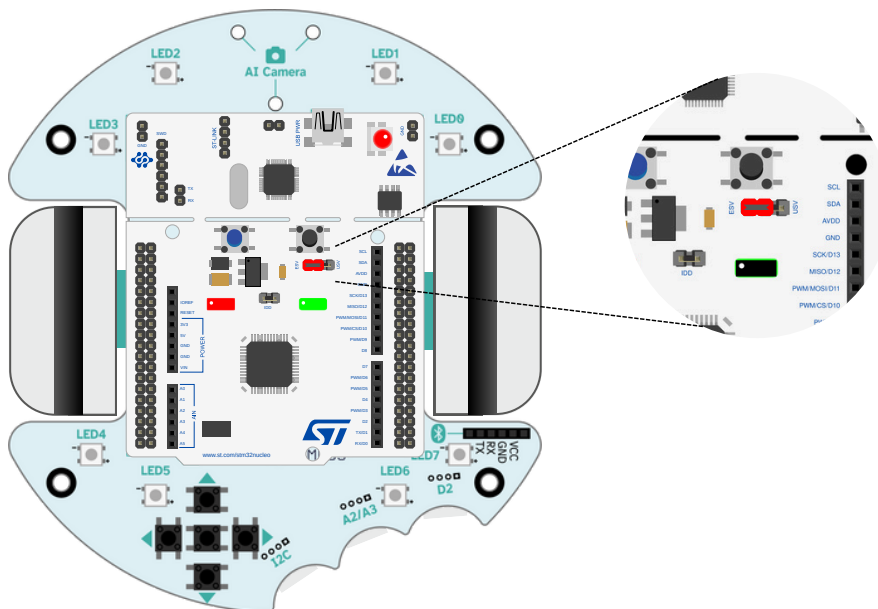
Les photos ci-dessus vous donneront la confirmation visuelle que vous n'avez pas fait d'erreur en suivant les instructions de montage.

Maintenant, il est temps de passer à la programmation du robot !

Programmation du robot DonutBot

Configuration pour la programmation en MicroPython avec l'interface Vittascience

- 1 • Mettre l'interrupteur d'alimentation du robot sur « ON ».
- 2 • Positionner le cavalier (dessiné en rouge sur la figure ci-dessous) sur la position « E5V ».
- 3 • Connecter la carte à l'ordinateur à l'aide du câble USB.
- 4 • Ouvrir l'interface de programmation de Vittascience : vittascience.com/l476.
- 5 • Cliquer sur le bouton « Téléverser » pour envoyer le programme dans la carte.



Démarrage du robot

- 1 • Débrancher tout câble USB connecté à votre robot.
- 2 • Vérifier que le cavalier (en rouge) sur la figure ci-dessus est sur la position « E5V ». La carte NUCLEO-L476RG est alimentée directement par la batterie du robot.
- 3 • Placer l'interrupteur d'alimentation du robot sur « ON », la LED sur sa base doit s'allumer en vert.

Le robot commencera à exécuter le dernier programme que vous avez téléchargé dans la carte NUCLEO-L476RG en utilisant l'interface de programmation de Vittascience.

Si le comportement de votre robot n'est pas celui que vous attendiez, vérifiez votre programme et assurez-vous que la batterie est assez chargée...

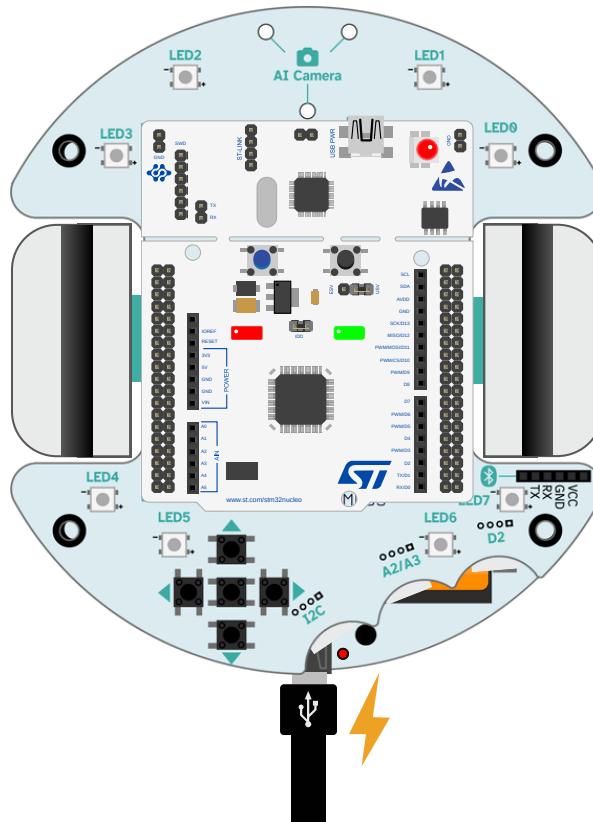
Chargement de la batterie du robot DonutBot

1 • Connectez le câble USB sur le robot, sur le port USB 5V sur la partie inférieure du châssis du DonutBot.

2 • La LED de chargement s'allume en rouge et reste allumée jusqu'à ce que la batterie soit complètement chargée.

3 • Le voyant est éteint lorsque la batterie est rechargée.

Votre robot est maintenant prêt à démarrer. Débrancher le câble et mettre l'interrupteur d'alimentation situé sur le robot sur « ON ».



Mission • 1 20 min

Allumer les LED et interagir avec le robot

Pour chacune des missions, vous trouverez le code de programmation par blocs correspondant au programme. La programmation en code Python est également disponible sur l'interface Vittascience. Par gain de place, dans ce livret, nous avons fait le choix de ne pas l'afficher pour toutes les missions.

• Mission 1-1 : Allumer les LED du robot

Pour débiter cette première mission, nous allons allumer les LED du robot. Cette mission très simple permet de maîtriser le transfert d'un programme et l'interface de la plateforme Vittascience.

Votre planning sera très chargé afin de maîtriser l'ensemble des technologies utiles à ce projet. Il faudra peut-être faire preuve d'initiative pour développer de nouvelles compétences qui n'étaient pas prévues dans votre formation initiale pour venir sur Mars. Commençons par quelque chose de simple. Vous allez être confrontés à quelques petits exercices de programmation pour vous familiariser avec les procédures d'utilisation de la carte à microcontrôleur présente dans tous les robots radioguidés de la surface.


ASM



Ho-la-la , il va falloir qu'on manipule ces petits trucs ? Mais je vais les écraser entre mes gros doigts !

ROCK



Pfff , ça fait plus de 30 ans qu'on utilise des microcontrôleurs dans nos rovers et ça fonctionne toujours ! Imagine, il y en avait déjà dans Perseverance ! C'est du solide !

CHLOE



Procédure en cours d'envoi. Message terminé.

Les blocs à utiliser pour cette mission sont dans le menu de gauche, pour commencer seules les deux catégories suivantes seront utiles :



Au démarrage

[Donutbot] contrôler la LED 0 ▼ à 

attendre 1 seconde(s) ▼

[Donutbot] contrôler la LED 0 ▼ à 

```

1 from machine import *
2 import neopixel
3 import utime
4
5 """ DonutBot robot """
6
7 # Neopixel on D12
8 d12 = Pin('D12', Pin.OUT)
9 npDonutbot = neopixel.NeoPixel(d12, 8)
10
11 npDonutbot[0] = (51, 51, 255)
12 npDonutbot.write()
13 utime.sleep(1)
14 npDonutbot[0] = (0, 0, 0)
15 npDonutbot.write()
16
17 while True:
18     pass
19 
```



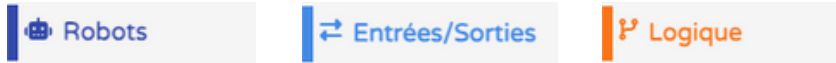
Accédez au programme en ligne :

<https://fr.vittascience.com/l476/?link=6807e6d554b1d&mode=blocks>

• Mission 1-2 : Utilisation des boutons

Nous venons d'apprendre à allumer une LED sur le robot avec la carte. Étudions à présent comment interagir avec l'aide des boutons. Nous allons faire en sorte que les LED s'allument lorsque nous appuyons sur les boutons du robot.

Nous aurons besoin pour ce faire de nouveaux blocs qui se trouvent dans les menus suivants :



Accédez au programme en ligne :

<https://fr.vittascience.com/l476/?link=6808a19248558&mode=blocks>

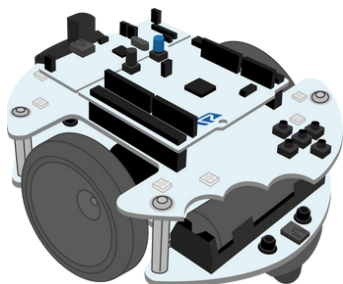


Mission • 2 ⌚ 40 min

Pilotage des moteurs

• Mission 2-1 : Faire avancer le robot

Il est temps de mettre en mouvement notre petit robot. Commençons par les mouvements simples "avancer" et "tourner".



Ici ASM. Les robots constructeurs de surface sont actuellement à l'arrêt et dans des positions inconnues. Vous allez maintenant apprendre à les déplacer et les remettre en position afin de les rapatrier pour une révision.



Cool il est où le joystick ?

À mon avis c'est pas comme avec tes perceuses où tu appuies juste sur un bouton !



Le roues du robot peuvent être contrôlées de plusieurs façons, à l'aide de différents blocs de code :

[Donutbot] contrôler le moteur **droit** direction **v** vitesse **50** (%)

Bloc 1 : Permet de contrôler chaque moteur, droit ou gauche, de manière indépendante, de contrôler le sens de rotation et de contrôler la vitesse de rotation.

[Donutbot] contrôler le robot **avancer** vitesse **50** (%)

Bloc 2 : Permet de contrôler simultanément le sens et la vitesse de rotation des deux moteurs.

[Donutbot] pivoter vers la **droite** vitesse **50** (%)

Bloc 3 : Permet d'activer un moteur pour pivoter à droite ou à gauche et de contrôler sa puissance.

[Donutbot] arrêter le moteur **droit**

Bloc 4 : Permet d'arrêter l'un des deux moteurs seulement, ou bien les deux simultanément.

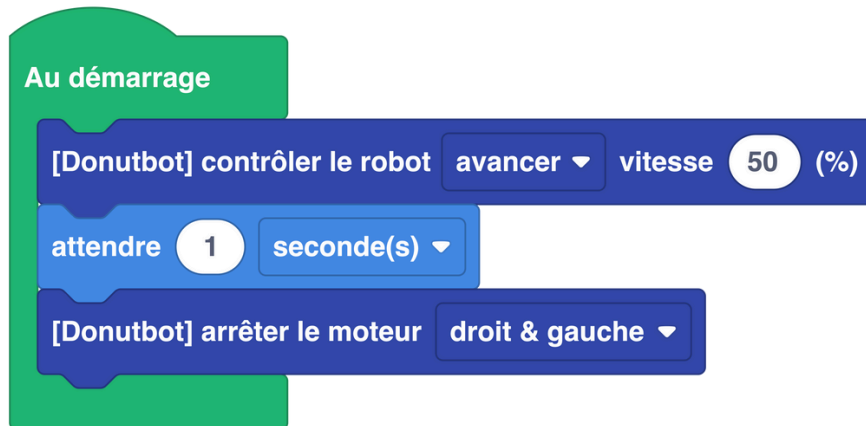
[Donutbot] relâcher le moteur **droit**

Bloc 5 : Permet de relâcher le moteur et de rendre la roue libre de tourner.

Pour notre premier programme nous allons faire avancer le robot en avant pendant 1 seconde. Nous utiliserons des blocs qui se trouvent dans :



Voici le code de programmation par bloc pour ce programme :



Accédez au programme en ligne :
<https://fr.vittascience.com/l476/?link=6808a4886ce1c&mode=blocks>



• Mission 2-2 : Apprendre à pivoter

Un robot qui avance tout droit, c'est bien, mais c'est dangereux, nous devons lui apprendre à tourner ! Pour cela, il est nécessaire contrôler indépendamment chaque moteur.

Deux choix s'offrent à nous pour tourner : la méthode "char d'assaut" ou la méthode "voiture". Le char d'assaut fait tourner ses chenilles droite et gauche en sens opposés, ce qui lui permet d'effectuer une rotation sur place. La voiture avec le système de différentiel a généralement une roue qui tourne plus vite que l'autre pour suivre une trajectoire courbe. Chaque technique présente ses avantages et ses inconvénients. La technique "char d'assaut" permet des virages plus serrés au détriment de la vitesse globale de déplacement. Pour la méthode "voiture", c'est l'inverse.

Dans notre cas nous, allons nous déplacer dans un milieu hostile (Mars) donc nous privilégierons la manœuvrabilité du robot avec la méthode "char d'assaut" pour réaliser des virages serrés à 90°. Pour ce faire, nous utiliserons des blocs qui se trouvent dans :



Voici le code de programmation par bloc pour ce programme :



Accédez au programme en ligne :

<https://fr.vittascience.com/l476/?link=6808ab714a3ce&mode=blocks>



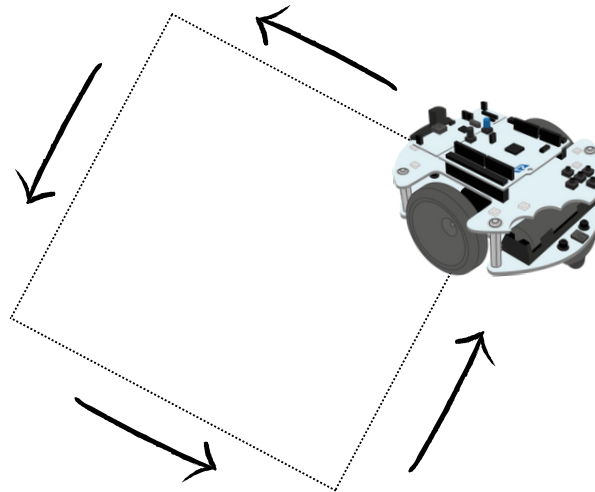
Ce programme ne produira probablement pas un virage à 90° parfait. Pour ce faire vous devrez ajuster les temporisations soit par essais et erreurs, soit à l'aide d'un tableau de proportionnalité. Ne pas hésiter à réduire la vitesse afin de mieux contrôler la direction.

Vous pouvez également utiliser le bloc suivant afin de déterminer un angle de rotation :

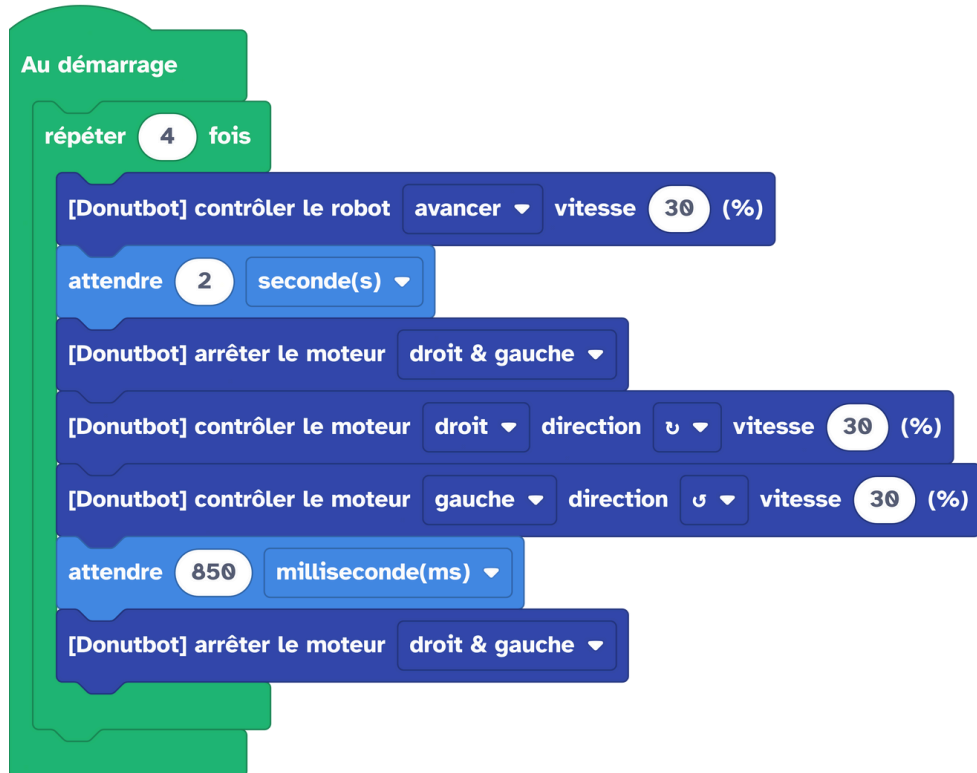
[Donutbot] pivoter vers la gauche ▼ de 90 ° vitesse 50 (%)

• Mission 2-3 : Suivre une trajectoire

Nous contrôlons maintenant parfaitement notre robot et nous pouvons le déplacer où nous voulons. Vous savez avancer, reculer et tourner à 90° . Essayez à présent de suivre une trajectoire carrée ! Pour ce faire, nous utiliserons des blocs qui se trouvent dans :



Voici le code de programmation par bloc pour ce programme :



Accédez au programme en ligne :

<https://fr.vittascience.com/l476/?link=6808abde0c6c2&mode=blocks>



Mission • 3 ⌚ 40 min

Tester le détecteur d'obstacles

• Mission 3-1 : Découverte du capteur de distance Time of Flight (ToF)

ASM, on a un problème !
L'un des robots semble bloqué.



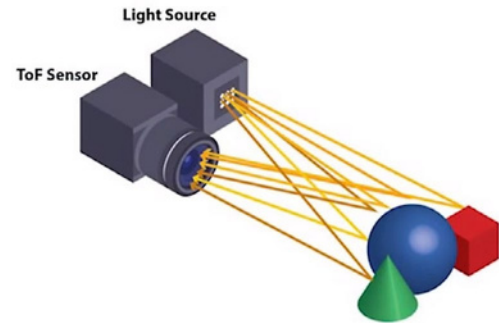


Ça doit sûrement être un problème de détecteur d'obstacles.
À mon avis, il est coincé contre un rocher.

Bien reçu Arès I.
Voici le protocole d'utilisation du capteur TOF.



Un capteur "Time of Flight", abrégé ToF, détecte les obstacles devant lui et détermine leur distance.



Son principe de fonctionnement est le suivant :

Le capteur ToF envoie des ondes lumineuses (laser infrarouge) pour sonder l'espace devant lui. Leur vitesse de propagation (celle de la lumière dans l'air) étant connue il ne reste à son électronique qu'à calculer le temps d'aller-retour de la lumière pour estimer la distance de l'obstacle.

Pour cette mission nous allons afficher la distance d'un objet dans la console série de l'ordinateur. Pour ce faire, nous utiliserons des blocs qui se trouvent dans :

Robots

Entrées/Sorties

Communication

Répéter indéfiniment

écrire dans la console [Donutbot - ToF] distance (cm)

attendre 1 seconde(s)

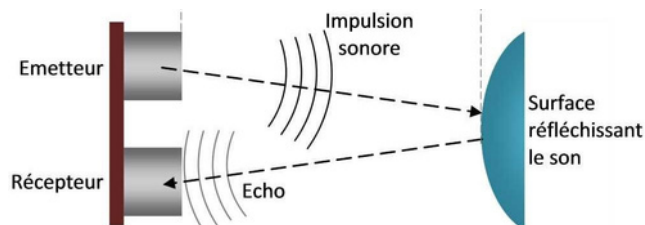
Accédez au programme en ligne :

<https://fr.vittascience.com/l476/?link=6808ac49d58b3&mode=blocks>



• Mission 3-2 : Découverte du capteur de distance à ultrasons (disponible en option)

Un capteur à ultrasons détecte les obstacles devant lui et détermine leur distance. Il fonctionne comme le capteur Time of Flight, mais il est moins rapide et moins précis.



Son principe de fonctionnement est le suivant :

Un émetteur produit une impulsion sonore. Cette impulsion se propage jusqu'à rencontrer un obstacle (à la distance d) qui la réfléchit partiellement vers un récepteur (écho).

Lorsque l'écho atteint le récepteur, le son a parcouru approximativement la distance $2d$ (inconnue) pendant la durée t (connue), à la vitesse V (connue) du son dans l'air.

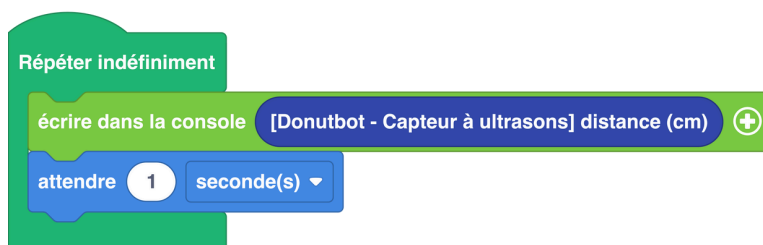
On en déduit d .

Pour cette mission nous allons afficher la distance d'un objet dans la console série de l'ordinateur. Pour ce faire, nous utiliserons des blocs qui se trouvent dans :

Robots

Entrées/Sorties

Communication



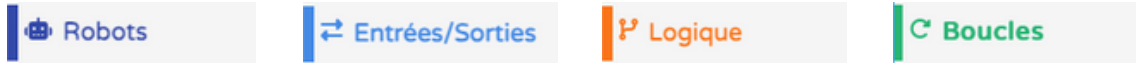
Accédez au programme en ligne :

<https://fr.vittascience.com/l476/?link=6808ac91bc783&mode=blocks>



• Mission 3-3 : Capteurs de distance et moteurs

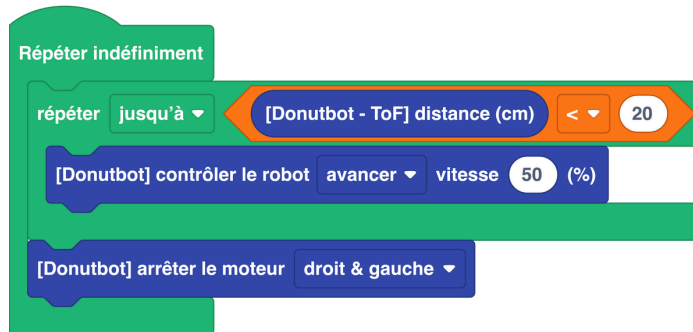
Nous allons exploiter le capteur ToF (ou le capteur à ultrasons) pour le déplacement du robot. Celui-ci devra se déplacer jusqu'à ce que sa distance à un obstacle ait une valeur choisie (par exemple 20 cm au plus). Pour ce faire, nous utiliserons des blocs qui se trouvent dans :



Nous allons utiliser dans cette mission les comparateurs logiques, qui permettent de tester si une information est vraie ou fausse. Selon le résultat, on décidera de réaliser telle ou telle action.

Dans notre cas, la question sera :

“Est ce que la distance entre mon robot et l’obstacle est inférieure à 20 cm ?”



Si vous mesurez à l'aide d'un centimètre la distance obtenue après déplacement du robot, il se peut qu'il y ait une légère différence avec l'indication du capteur, possiblement pour deux raisons :

- Une erreur de protocole : il ne faut pas mesurer au niveau des roues, mais plutôt au niveau du capteur de distance.
- L'inertie du robot qui ne peut pas s'arrêter instantanément ; plus sa vitesse sera importante, plus longue sera sa distance de freinage.
- Le microcontrôleur de la NUCLEO-L476RG a besoin de fractions de secondes pour traiter l'information et envoyer l'ordre de freiner, qui ne sera pas exécuté instantanément non plus.

Ces temps de latence peuvent se traduire par un excès de quelques millimètres ou même centimètres parcourus selon les cas.

NB : Il est tout à fait possible d'utiliser le capteur à ultrasons à la place du capteur ToF, le programme sera identique excepté pour le nom du capteur utilisé.

Accédez au programme en ligne :

<https://fr.vittascience.com/l476/?link=6808acf337cbe&mode=blocks>



Mission • 4 40 min

Pilotage temps réel avec le module bluetooth



ROCK

ASM, ici Arès I. Nous confirmons le problème : le robot bloqué est trop proche d'un obstacle.



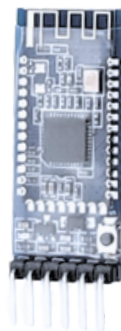
CHLOÉ

On pourrait peut-être mettre nos combinaisons pour aller le décoincer ?



ASM

Hors de question Chloé ! C'est trop dangereux pour vous !
Si l'antenne Bluetooth est installée sur les robots, voici une procédure pour apprendre à communiquer à distance.



Module Bluetooth HM-10 à fixer sur le robot.
Attention à respecter le sens des broches !

Nous allons à présent piloter le robot à l'aide d'une communication sans fil. L'idée est que le robot reçoive des informations envoyées par Bluetooth par l'opérateur et réagisse en fonction des consignes transmises.

• Mission 4-1 : Vérification de la communication

Dans une première mission, nous allons établir la communication avec le robot. Nous avons besoin d'un smartphone pour recevoir le signal émis par le robot. Pour ce faire, nous utiliserons des blocs qui se trouvent dans :

Robots

Entrées/Sorties

Répéter indéfiniment

[DonutBot] envoyer des données

DonutBot - module bluetooth activé

attendre

5

seconde(s)

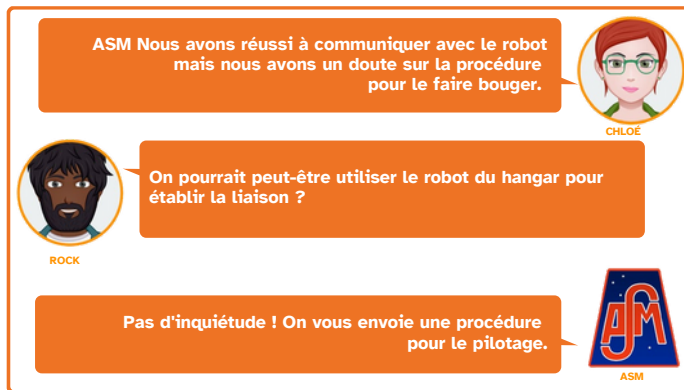
Accédez au programme en ligne :

<https://fr.vittascience.com/l476/?link=6808adb968d54&mode=blocks>

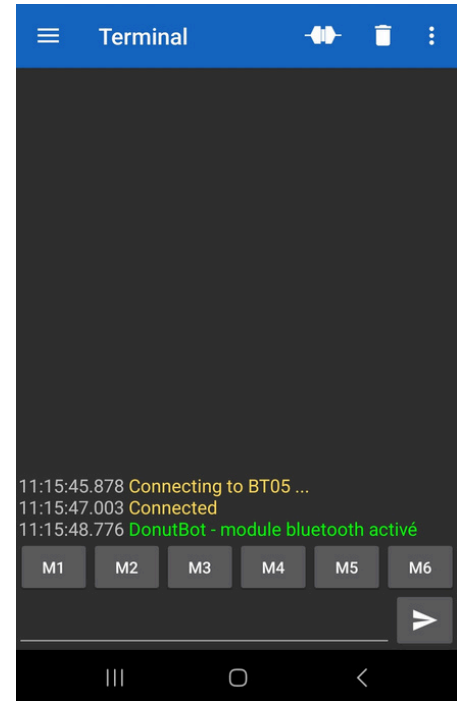


La technologie Bluetooth permet l'échange de données entre deux appareils (ici votre smartphone et le robot) au moyen d'ondes radio de fréquence autour de 2,4 GHz. Cette technologie est très répandue au quotidien : kit mains libres des voitures, casque ou oreillettes sans fil, montre connectée, etc.

Sur le smartphone, nous avons besoin d'une application comme Serial Bluetooth Terminal (Android) ou BLE Tools (iOS).



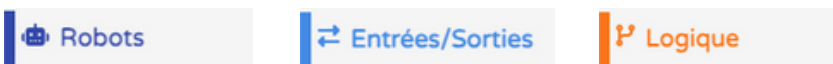
La capture d'écran ci-contre montre l'application Serial Bluetooth Terminal qui reçoit les données envoyées par le robot.



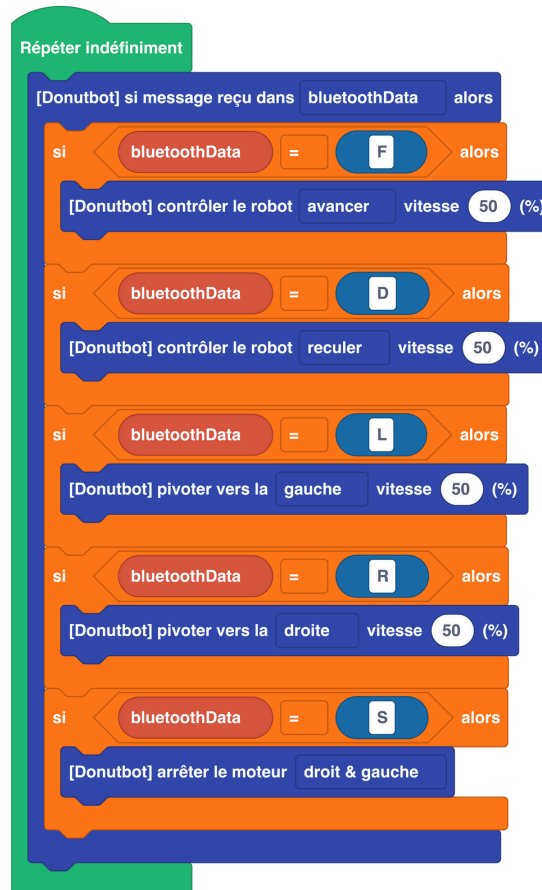
• Mission 4-2 : Test en conditions réelles "martiennes", la télécommande

Ce nouvel exercice va consister à piloter notre petit robot dans le hangar. Pour cela, nous utiliserons les programmes réalisés dans les missions qui précèdent. L'objectif sera de faire avancer le robot, de le faire tourner, de le faire reculer et de l'arrêter avec les différentes commandes envoyées.

Pour ce faire, nous utiliserons des blocs qui se trouvent dans :



Il est également possible de créer sa propre application pour avoir un plus joli rendu. On vous propose d'utiliser l'un des deux sites internet disponibles : [Thunkable](#) ou [App Inventor](#).



On constate qu'il est complexe de piloter le robot avec ce système car il y a un temps de latence.

Une solution plus efficace consiste à le programmer pour qu'il gère ses déplacements de façon autonome sur la base de la vitesse de rotation de ses moteurs mais aussi d'informations sur son environnement, obtenues par les capteurs dont il est équipé.

Sur commande, le robot devra s'adapter pour accomplir au mieux sa tâche. Par exemple, si on lui donne la consigne "Allez à 100 mètres direction Nord !" il devra grâce à sa programmation relier le point de rendez-vous qui se situe 100 mètres au nord.

Accédez au programme en ligne :

<https://fr.vittascience.com/l476/?link=6808ae1b5d57d&mode=blocks>



Mission • 5 50 min

Déplacement en mode cases



ROCK

ASM ! Votre procédure fait que le robot réagit trop tard !



CHLOE

Mais non c'est toi qui n'es pas assez soigneux !



ASM

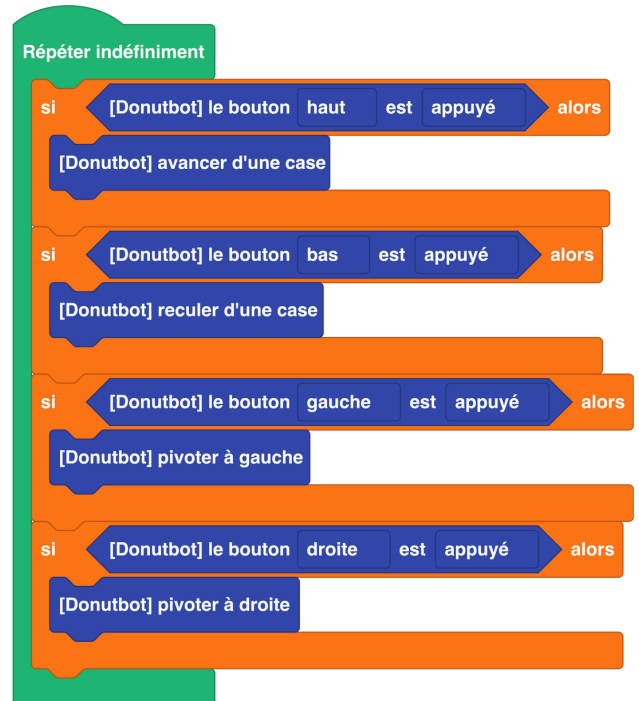
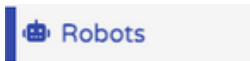
Pas d'inquiétude ! On vous envoie une nouvelle procédure afin de vous faciliter la vie.

• Mission 5-1 : Déplacement en mode cases

Pour cette mission nous allons programmer les déplacements du robot en mode cases en utilisant les boutons. Le robot fera de petits déplacements comme s'il était sur un échiquier ou un plateau de dames.

Cela permettra de simplifier la programmation. On peut par exemple, se déplacer de 3 cases en avant et de 2 cases vers la droite.

Pour ce faire, nous utiliserons des blocs qui se trouvent dans :



Accédez au programme en ligne :

<https://fr.vittascience.com/l476/?link=6808af91dc594&mode=blocks>

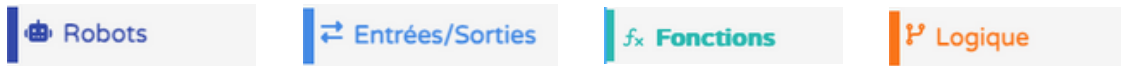
Une autre méthode, plus avancée, consiste à utiliser des fonctions.

Pour faciliter la tâche, nous allons créer trois fonctions : une pour avancer, une pour tourner à droite et une pour tourner à gauche.

Nous allons exploiter les missions précédentes avec la rotation à 90° qui conviendra parfaitement pour un déplacement sur case. Nous allons également mesurer combien de temps il faut, en déplacement rectiligne, pour avancer d'une case.

Il n'y a pas de gros changements par rapport aux précédentes missions sauf qu'au lieu de mettre les blocs dans la boucle "Répéter indéfiniment" ou "Au démarrage", nous allons les mettre dans un bloc "Fonction" auquel nous donnerons un nom reconnaissable afin de ne pas nous perdre dans le code. Nous utilisons toujours un bloc de conditions afin que le robot exécute la fonction lorsque l'on appuie sur le bon bouton, ainsi qu'une seconde de délai afin d'avoir le temps d'appuyer.

Pour ce faire, nous utiliserons des blocs qui se trouvent dans :

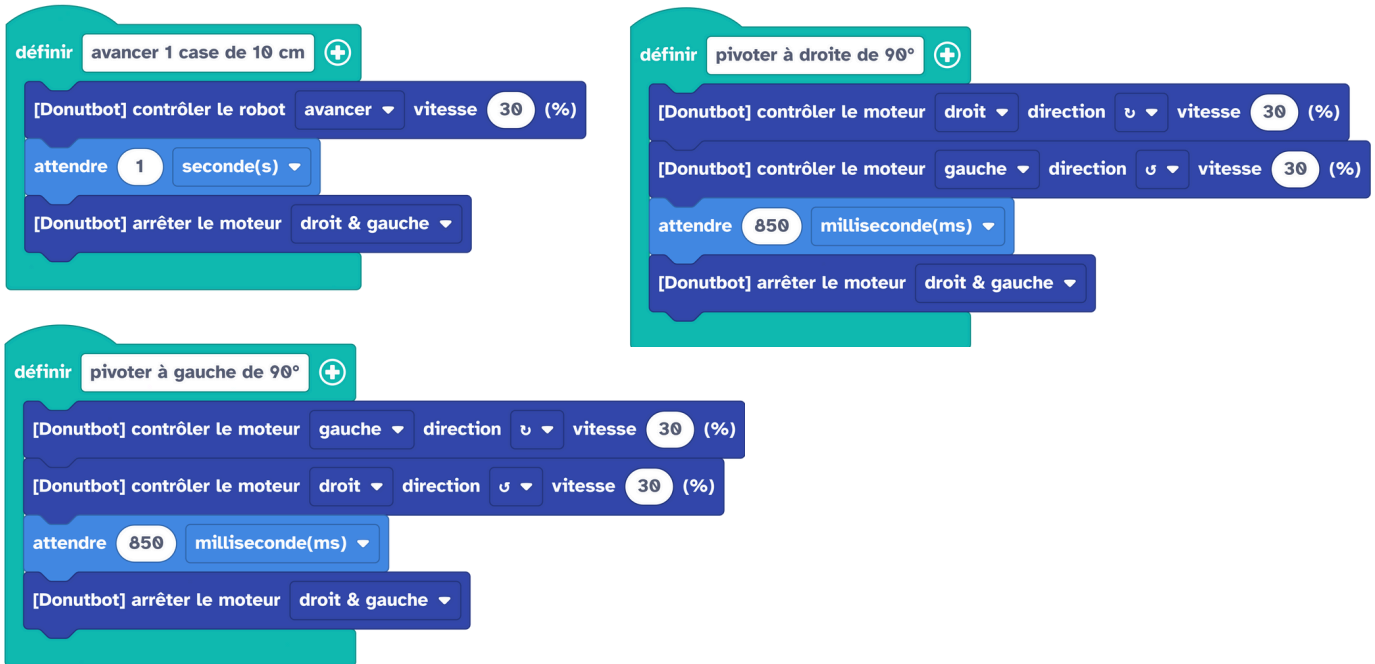


Les fonctions simplifient la programmation. On peut les appeler dans le programme principal pour qu'elles s'exécutent d'une traite. Les fonctions rendent le programme plus lisible en nous évitant de répéter un grand nombre de fois les lignes de code dans le programme principal.

Accédez au programme en ligne :

<https://fr.vittascience.com/l476/?link=6808d81eb0b4f&mode=blocks>

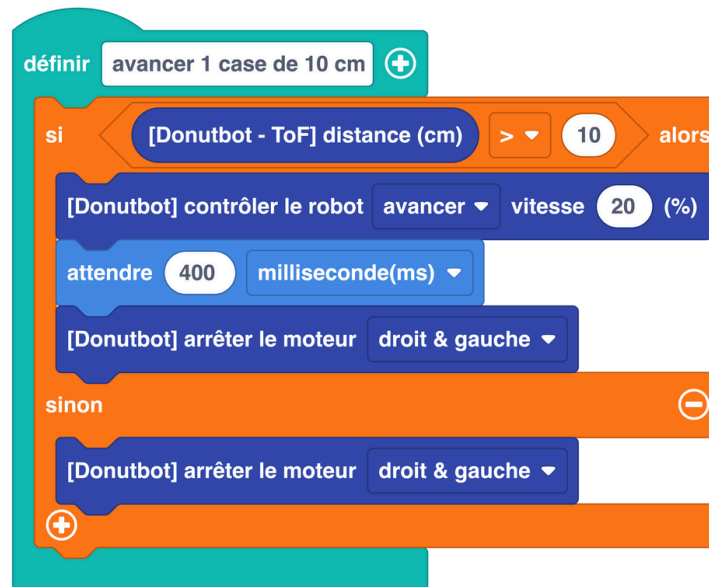




• Mission 5-2 : Détection d'obstacles avant déplacement

Se déplacer sans regarder où l'on va risque de rapidement poser des problèmes. Dans la vie de tous les jours notre vue en 3 dimensions nous permet de repérer les obstacles et d'évaluer leur distance. Pour notre robot, ce seront le capteur Time of Flight ou le capteur à ultrasons qui joueront le rôle de nos yeux. L'objectif de cette mission sera de se déplacer en vérifiant avant chaque mouvement si la case de destination est libre. Dans le cas contraire, on décidera de ne pas avancer sur la case encombrée et d'afficher une alerte clignotante sur les LED du robot. Nous allons uniquement modifier la fonction avancer pour que celle-ci détecte les obstacles. Pour ce faire, nous utiliserons des blocs qui se trouvent dans :

Ici, seuls les changements dans la fonction « Avancer 10 cm » sont présentés :



Nous voilà déjà bien avancés, notre robot est maintenant capable de se déplacer seul en évitant les obstacles ou plutôt en les ignorant !



Mission • 6 ⌚ 50 min

Suiveur de ligne

• Mission 6-1 : Suivre une ligne

Le suivi de ligne est assuré par trois capteurs de couleurs situés sous le châssis du robot.

Quatre LED blanches vont émettre de la lumière et les capteurs de couleurs vont fonctionner comme des phototransistors qui détectent la réflexion de la lumière blanche sur le sol.

Le noir absorbant le rayonnement, le récepteur ne détecte pas de signal émis. C'est le cas, lorsque le capteur est au-dessus de la ligne, celui-ci va retourner la valeur de 1. Dans le cas contraire, si le signal est réfléchi (comme sur du blanc par exemple), le capteur retourne la valeur de 0.

Pour commencer, nous utiliserons un des blocs qui se trouvent dans :

 Robots

Répéter indéfiniment

[Donutbot] suivre la ligne

Accédez au programme en ligne :

<https://fr.vittascience.com/l476/?link=6808d8c24aa31&mode=blocks>





ROCK

ASM ! Super la procédure, cela fonctionne presque bien ! Mais un problème se pose lorsque l'on rentre dans la base. Il n'y a plus de cases à suivre...



CHLOÉ

Peut-être qu'il serait possible de suivre des lignes tracées au sol ?

Bien joué, Chloé c'est l'idée ! On vous envoie les procédures pour régler les problèmes.



ASM

• **Mission 6-2 : Suivre une ligne avec 3 capteurs**

Essayons maintenant de décomposer le programme qui se cache derrière le bloc ‘Suivre la ligne’ Dans un premier temps, pour faciliter la compréhension, voici ce que l’on obtiendrait dans le cas de la présence de deux capteurs pour suivre la ligne avec un robot.

- **Cas 1 :** Les 2 capteurs sont au-dessus de la ligne noire.
→ Retourne 1,1
- **Cas 2 :** Le capteur L est au-dessus de la ligne et le capteur R est à l'extérieur.
→ Retourne 1,0
- **Cas 3 :** Le capteur L est à l'extérieur et le capteur R est au-dessus de la ligne.
→ Retourne 0,1
- **Cas 4 :** Les 2 capteurs sont à l'extérieur de la ligne noire.
→ Retourne 0,0

La figure ci-contre illustre le fonctionnement des capteurs et l'information que l'on pourrait récupérer au niveau de la carte NUCLEO-L476RG.



Pour faciliter la compréhension avec les 3 capteurs du robot DonutBot, nous allons utiliser un tableau logique dans lequel figurent des colonnes pour les capteurs ainsi que des colonnes pour le fonctionnement des moteurs. Nous utiliserons le code 1 pour "l'absence de ligne / couleur = blanc" et 1 pour "présence de ligne / couleur = noir".

Capteur gauche	Capteur central	Capteur droit	Fonctionnement des moteurs
0	0	0	Chercher la ligne
0	0	1	Pivoter à droite
0	1	0	Avancer
1	0	0	Pivoter à gauche

On constate que dans le premier cas, la ligne n'est plus détectée et qu'il faut mettre en place une stratégie pour la retrouver. Ce pourrait être par exemple « avancer de 5 cm puis tourner en rond jusqu'à détection de ligne » ou alors simplement « reculer » car on est allé trop vite et on a perdu la ligne.

Programmons cela ! Pour ce faire, nous utiliserons des blocs qui se trouvent dans :



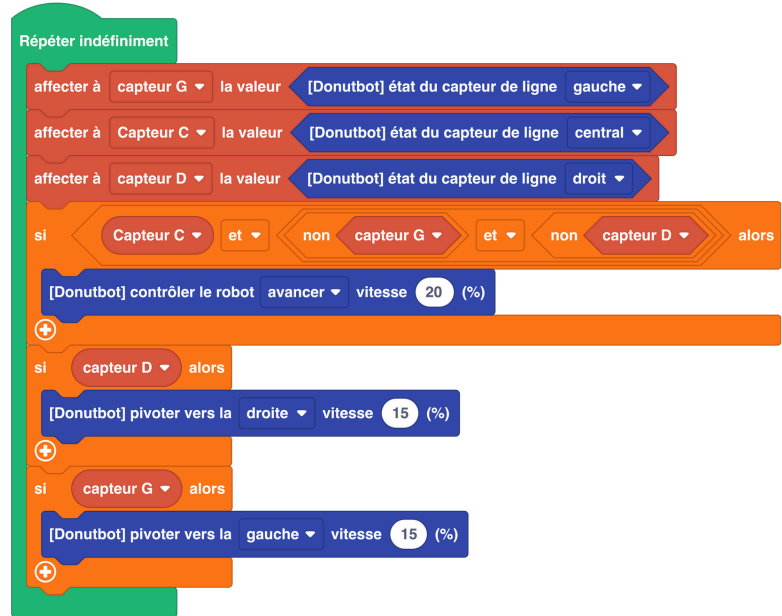
On peut ajouter dans le bloc “Au démarrage” du programme une instruction “relâcher le moteur droit & gauche”. C’est une mini initialisation qui permet d’être sûr que l’on commence moteur éteint. On peut également y ajouter une pause pour avoir le temps de bien positionner le robot avant que celui-ci ne démarre.

On observe que le programme va devenir plus conséquent et faire appel à l’état du capteur à chaque instruction. Afin de simplifier la programmation, nous allons utiliser des variables.

Accédez au programme en ligne :

<https://fr.vittascience.com/l476/?link=6808d9184a73b&mode=blocks>



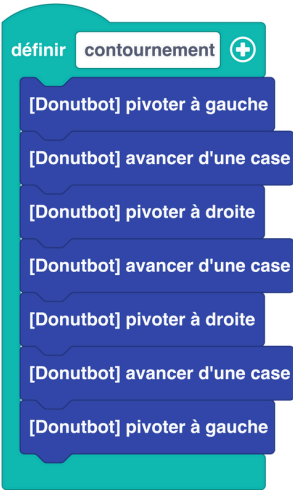
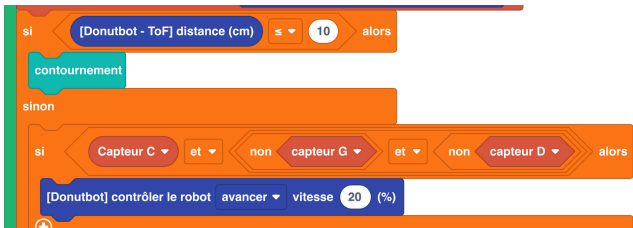


• Mission 6-3 : Suivre une ligne et éviter les obstacles

Pour cette mission, nous allons imaginer que le robot détecte un objet alors qu’il suit sa ligne. Dans ce cas de figure, il devra s’arrêter, le contourner puis retrouver la ligne pour continuer derrière l’obstacle. Nous allons utiliser des fonctions comme pour la mission 5 : “Avancer 10 cm” et “Tourner à droite” ou “Tourner à gauche” pour former une fonction de contournement d’obstacles. Essayons de mettre cela en place ; nous aurons besoins des blocs suivants :



Seule la fonction de contournement est détaillée ci-contre. Le reste de la programmation est identique à la mission précédente.



Accédez au programme en ligne :

<https://fr.vittascience.com/l476/?link=6808d9bdcf2c8&mode=blocks>

Mission • 7 🕒 à l'appréciation de l'encadrant

Enfin la routine !

Mise en situation complexe :

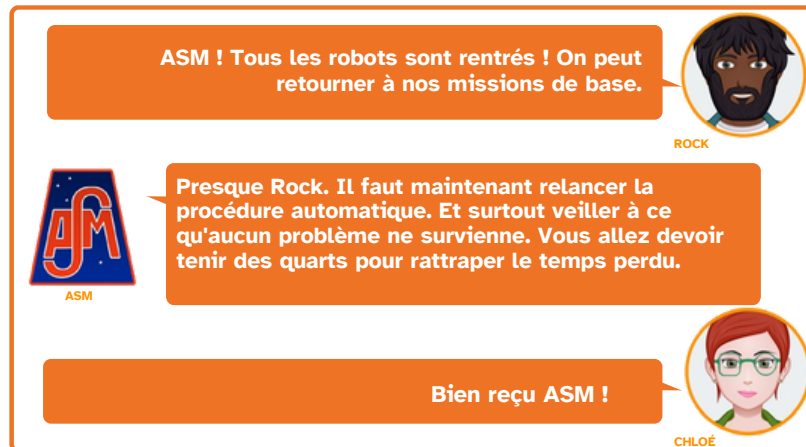
Le robot suit la boucle en permanence, mais une chute de température brutale survient. Cet évènement le contraint à arrêter toute activité pour se mettre à l'abri.

Pour cela, on passe en commande manuelle et on planifie un retour au centre Ares I ou aux coordonnées indiquées.

Les déplacements seront sécurisés en utilisant le capteur ToF pour vérifier que chaque mouvement est valide.

Le pilotage à distance peut se faire via le smartphone et le module Bluetooth. Il faudra prévoir des déplacements de plusieurs natures.

Le code est présenté ici dans son intégralité. Bien qu'il ne soit pas trop complexe, il devient conséquent en taille !



```

Au démarrage
  [Donutbot] clignoter le robot en
  
```

```

Répéter indéfiniment
  [Donutbot] si message reçu dans bluetoothData alors
    si bluetoothData = " F " alors
      [Donutbot] avancer d'une case
    sinon si bluetoothData = " D " alors
      [Donutbot] reculer d'une case
    sinon si bluetoothData = " L " alors
      [Donutbot] pivoter à gauche
    sinon si bluetoothData = " R " alors
      [Donutbot] pivoter à droite
    sinon si bluetoothData = " S " alors
      suiveur de ligne
    sinon
      suiveur de ligne
  
```

```

définir suiveur de ligne
  affecter à capteur G la valeur [Donutbot] état du capteur de ligne gauche
  affecter à Capteur C la valeur [Donutbot] état du capteur de ligne central
  affecter à capteur D la valeur [Donutbot] état du capteur de ligne droit
  si [Donutbot - ToF] distance (cm) ≤ 10 alors
    contournement
  sinon
    si Capteur C et non capteur G et non capteur D alors
      [Donutbot] contrôler le robot avancer vitesse 20 (%)
    si capteur D alors
      [Donutbot] pivoter vers la droite vitesse 15 (%)
    si capteur G alors
      [Donutbot] pivoter vers la gauche vitesse 15 (%)
  
```

```

définir contournement
  [Donutbot] pivoter à gauche
  [Donutbot] avancer d'une case
  [Donutbot] pivoter à droite
  [Donutbot] avancer d'une case
  [Donutbot] pivoter à droite
  [Donutbot] avancer d'une case
  [Donutbot] pivoter à gauche
  
```



Pour aller plus loin

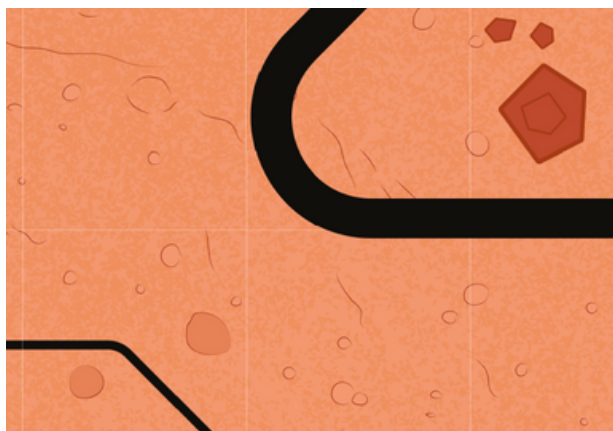
Afin de compléter la programmation de ce livret, il est possible de rajouter les activités suivantes en cours d'année ou sur d'autres niveaux en parallèle.

Réalisation d'objets en 3D :

- Mini cratère qui s'adapte au parcours
- Rocher
- Base Martienne
- Panneau solaire

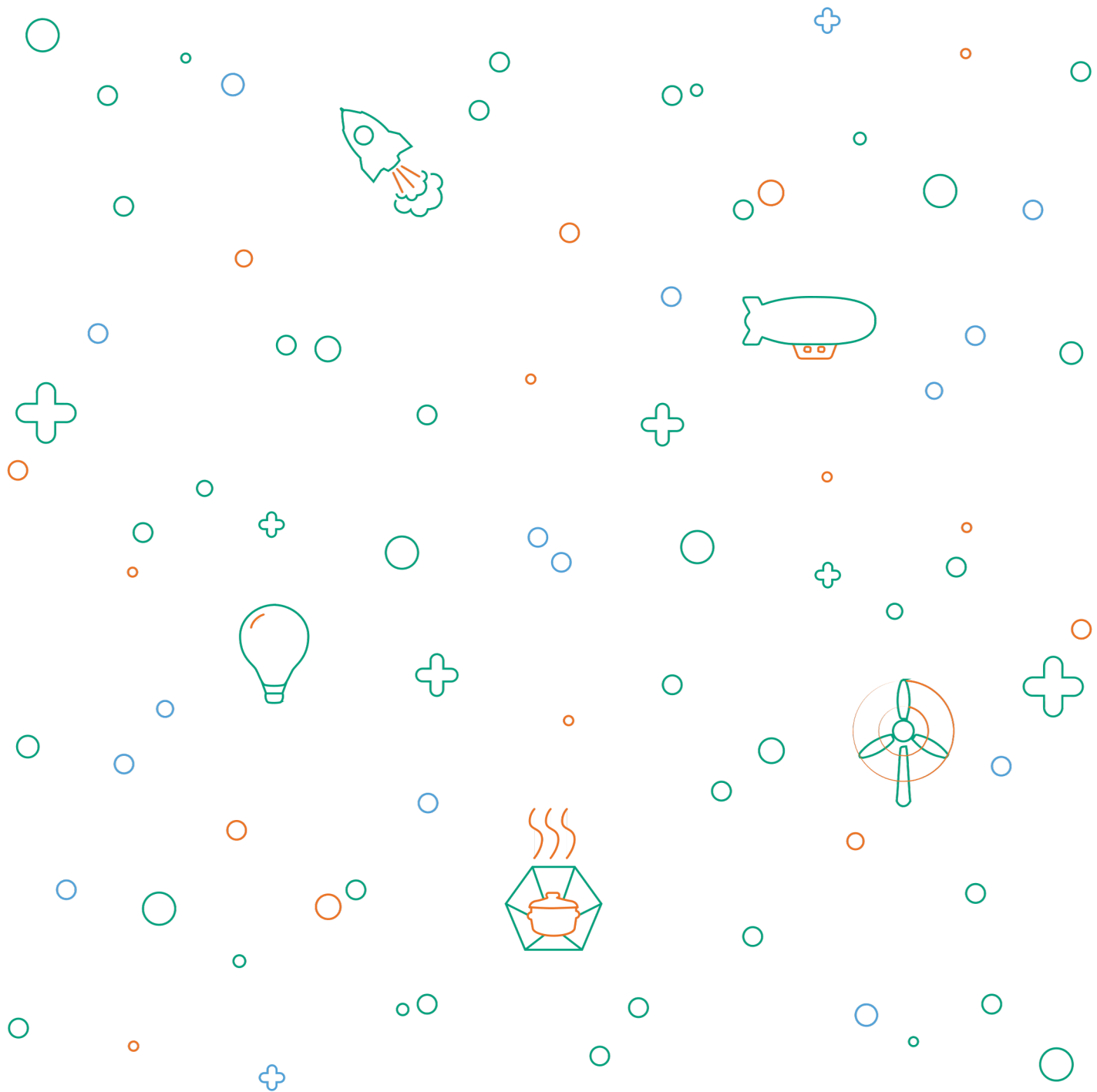
Un projet plus conséquent consisterait à concevoir une base pour accueillir le robot. On pourrait partir de l'idée d'un garage avec une porte automatique, ce qui permettrait d'utiliser un capteur de distance afin de détecter la présence du robot et de commander l'ouverture ou la fermeture du sas d'entrée de la base. Ce projet mettrait en œuvre de la programmation, mais aussi de la conception et de la fabrication avec machine outils, imprimante 3D ou découpe laser.

Pour fermer la piste du robot martien, imprimez cette feuille au format A4 et placez-la sur la piste. Retrouvez-la sur le site <https://fr.vittascience.com/learn/tutorial.php?id=1398/>



Notes

This image shows a single sheet of white paper with horizontal ruling lines. The lines are evenly spaced and extend across the width of the page. There are no margins, text, or other markings on the paper.





Voici le livret d'accompagnement du robot martien.

Il détaille chaque étape et chaque atelier nécessaire à la réalisation de l'expérience. Ce livret n'est pas exhaustif, l'imagination et les ressources disponibles sur le site sont là pour aider à approfondir l'expérience ! Ce livret est susceptible d'évoluer, nous vous invitons à consulter la version disponible en ligne qui est mise à jour régulièrement sur : www.vittascience.com/learn (chercher "Robot martien" pour trouver le guide d'utilisation)



life.augmented



STEM your way
Innovation depends on you



IPCEI
on Microelectronics