

Premiers pas en robotique avec le robot **Thymio-II** et l'environnement **Aseba/VPL**

Moti Ben-Ari et autres contributeurs
voir `authors.txt` pour plus de détails

Version 1.3~pre1 pour **Aseba 1.3.1**

© 2013–14 par **Moti Ben-Ari** et autres contributeurs.

Cette œuvre est mise à disposition selon les termes de la Licence Creative Commons Attribution - Partage dans les Mêmes Conditions 3.0 non transposé. Pour voir une copie, de la license, visitez <http://creativecommons.org/licenses/by-sa/3.0/deed.fr>.



Préface

Qu'est-ce qu'un robot ?

Imaginez-vous sur votre vélo, pédalant sur la route, lorsque vous vous trouvez face à une colline. Vous décidez donc de pédaler plus vite pour ne pas perdre trop de vitesse. Une fois en haut, vous vous trouvez face à une descente plutôt raide. Vous allez donc freiner pour éviter de prendre trop de vitesse et de perdre le contrôle de votre vélo. Lorsque vous êtes sur votre vélo, vos yeux sont vos *capteurs* qui mesure ce qu'il se passe dans le monde. Lorsque ces capteurs — vos yeux — détectent un *événement* tel qu'une courbe de la route, vous effectuez une *action*, telle que bouger le guidon à gauche ou à droite.

Dans une voiture, nous pouvons trouver de nombreux *capteurs*. Le tachymètre, ce cadran à aiguille ou digital derrière le volant, vous permet de savoir à quelle vitesse votre voiture avance. Si vous remarquez que vous allez trop vite par rapport à la limitation, vous allez freiner et, au contraire, vous accélérerez si vous vous trouvez en dessous de la limite. Un autre cadran vous indique la quantité d'essence se trouvant dans votre réservoir. Si vous voyez qu'il n'y en a presque plus, vous déciderez d'entreprendre l'action d'aller remplir votre réservoir.

Autant sur votre vélo que dans votre voiture, vous recevez des informations, des données, depuis les capteurs et vous décidez d'entreprendre des actions vis-à-vis de ces données. Un *robot* est un système dans lequel le processus — recevoir des données, décider d'une action, réaliser l'action — est effectué par un système informatique, en général sans la participation d'un être humain.

Le robot Thymio II et l'environnement Aseba VPL

Thymio II est un petit robot conçu à but éducatif (Figure 1.1). Il comprend différents capteurs qui peuvent mesurer la lumière, les distances, la température, etc. Il possède également des boutons tactiles, il peut se rendre compte qu'on lui donne une petite tape et peut entendre quelqu'un qui frappe dans ses mains. Il possède deux roues, chacune reliée à un moteur, lui permettant de se déplacer sur une table, ou sur le sol. Il peut encore s'illuminer de toutes les couleurs et jouer de la musique ! Dans le reste de ce document, le robot Thymio II sera souvent nommé simplement Thymio. Il s'agira toujours de la version II du robot.

Aseba est un environnement de programmation pour petits robots comme le Thymio. VPL est un composant d'Aseba qui permet à toutes et tous de programmer *graphiquement* Thymio en utilisant des blocs d'événement et d'action très simples d'emploi. Ce tutoriel suppose qu'Aseba est installé sur votre ordinateur ; si ce n'est

pas le cas, allez sur <https://aseba.wikidot.com/fr:downloadinstall>, sélectionnez votre système d'exploitation, téléchargez et installez.

Table des matières

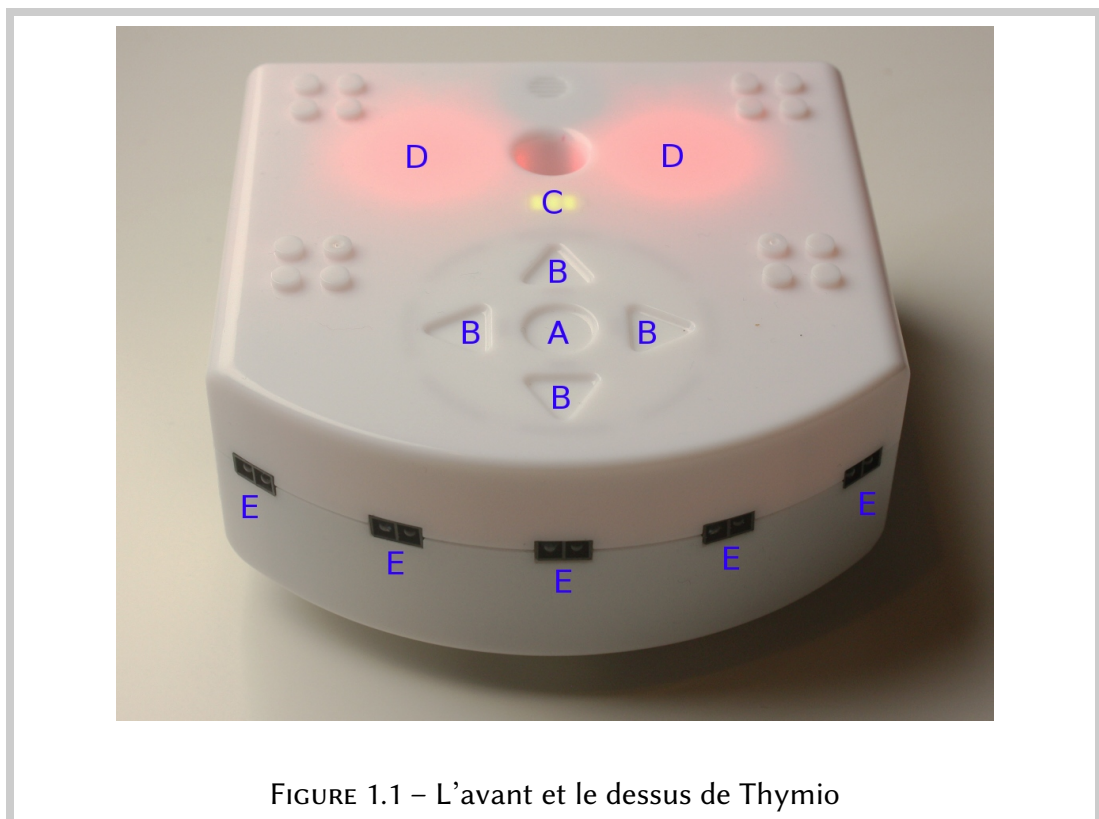
1	Votre premier projet de robotique	2
2	Changer les couleurs	9
3	Thymio en mouvement	12
4	Un robot de compagnie	16
5	Thymio est sur une piste	22
6	Sons et chocs	26
7	Aimer pour un temps	29
8	États : Pour ne pas toujours faire la même chose (avancé)	31
9	Thymio apprend à compter (avancé)	38
10	Et après ?	43

Chapitre 1

Votre premier projet de robotique

Faire connaissance avec Thymio

La Figure 1.1 montre l'avant et le dessus de Thymio. Vous pouvez voir le bouton central rond (A), entouré de quatre boutons triangulaires (B). Ce sont des boutons tactiles, un simple effleurement suffit à les activer. Juste derrière ces boutons se trouve un indicateur en forme de pile (C). Une, deux ou trois petites barres de lumière verte affichent l'état de charge du robot. Sur l'arrière du robot, nous voyons les lumières du haut (D), allumées en rouge sur cette photo. Il y a des lumières similaires sur le bas du robot (allumées en vert dans la Figure 3.2). Enfin, les petits rectangles noirs à l'avant du robot (E) sont des capteurs infrarouge de distance, vous en saurez plus dans le Chapitre 4.



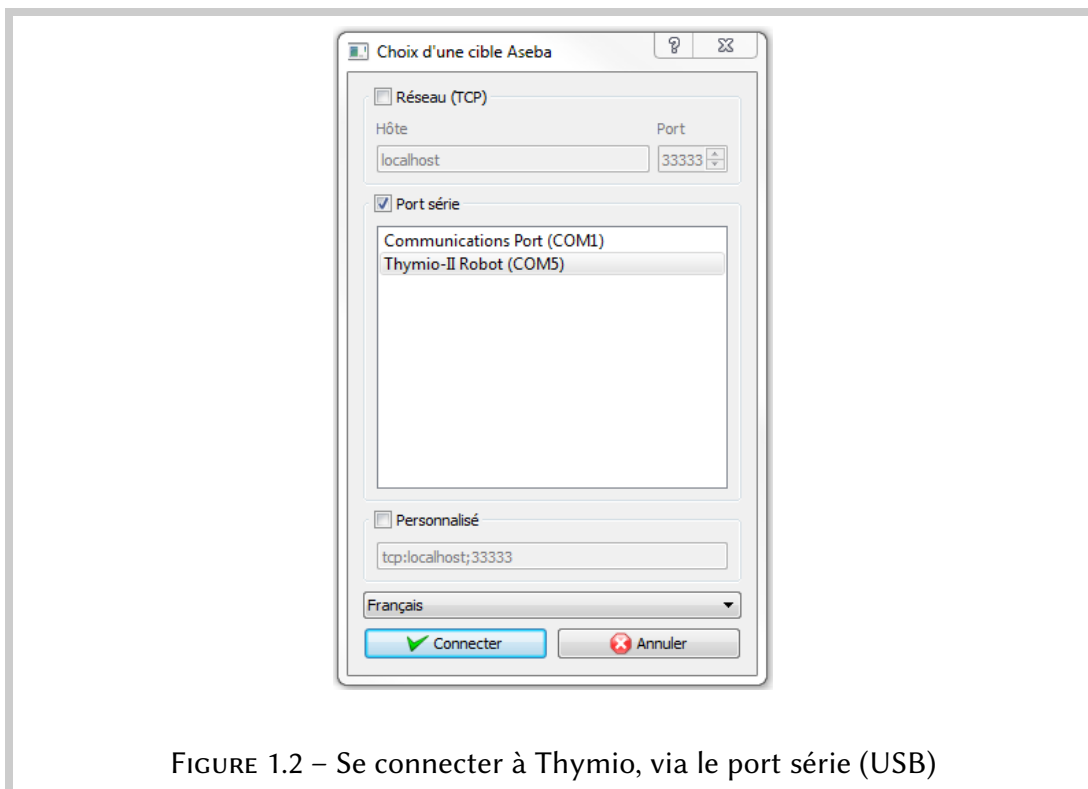

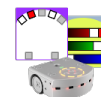


FIGURE 1.2 – Se connecter à Thymio, via le port série (USB)

Connecter le robot, démarrer Aseba et lancer VPL

Pour commencer, connectez Thymio à votre ordinateur à l'aide du câble USB fourni avec le robot. Si la connexion est réussie, le robot jouera quelques notes. S'il est éteint, touchez simplement son bouton central pendant cinq secondes. Lancez VPL en double-cliquant sur l'icône  (visible en grand dans la marge à droite).



Il se peut que VPL se connecte directement à votre robot. Si ce n'est pas le cas, la fenêtre montrée dans la Figure 1.2 devrait apparaître. Cochez la case **Port série**, cliquez sur **Thymio-II Robot**, sélectionnez **Français** et cliquez sur **Connecter**. En fonction de la configuration de votre ordinateur, il peut y avoir plusieurs entrées dans la liste des ports séries. Dans tous les cas, il faut choisir Thymio-II.

Truc

Il est aussi possible d'accéder à VPL depuis Aseba Studio, l'environnement de programmation textuelle, à travers le *plugin* VPL qui se trouve dans la zone *Outils* en bas à gauche de l'écran.

L'interface VPL

L'interface VPL est illustrée ci-dessous. Elle est composée de six zones :

1. Une barre d'outils avec les icônes pour créer un nouveau programme, en ouvrir un existant, sauvegarder, lancer le programme, etc.
2. La zone de programmation pour accueillir le programme qui contrôlera Thymio.
3. Une indication si le programme que vous êtes en train de construire est formulé correctement ou pas.
4. Les blocs d'événement disponibles pour construire votre programme.
5. Les blocs d'action disponibles pour construire votre programme.
6. La traduction textuelle du programme (voir en bas de page).

Les blocs d'événement et d'action seront décrits petit à petit au long de ce document.

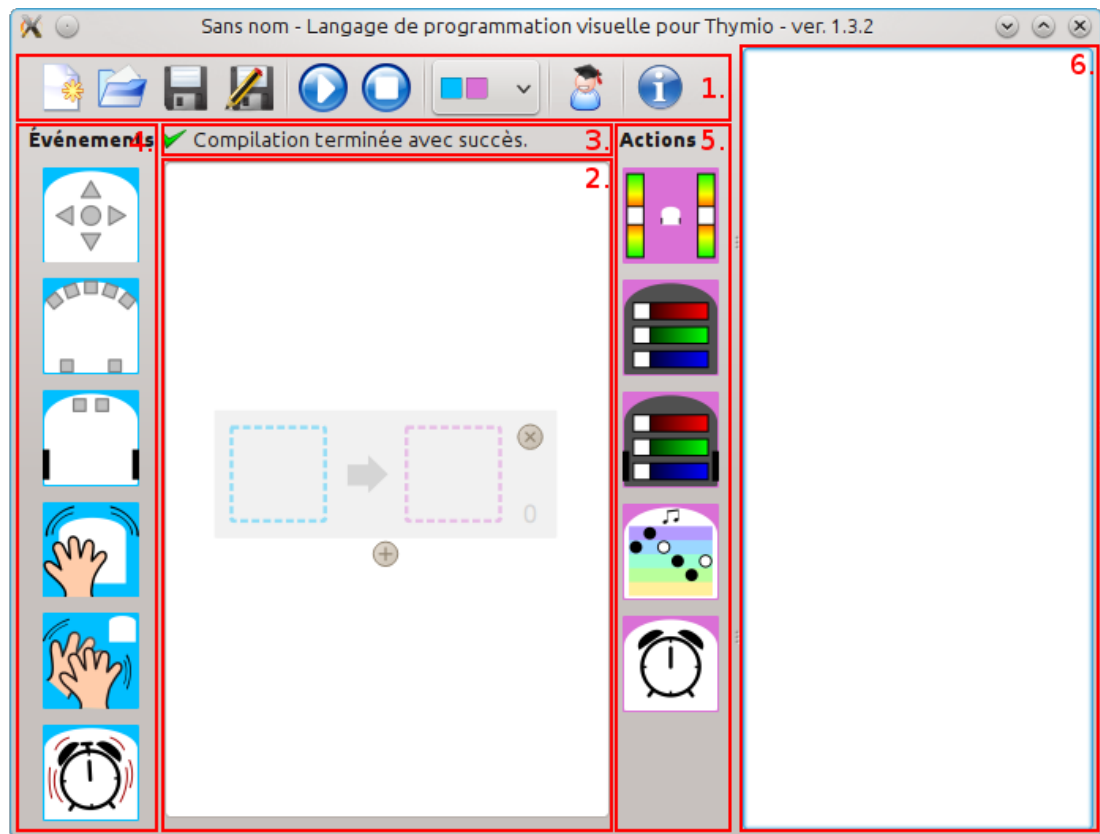



FIGURE 1.3 – La fenêtre de VPL

Pour aller plus loin

Dès que vous créez un programme en utilisant VPL, le programme texte qui sera chargé dans le robot apparaît dans la partie droite de la fenêtre. Si vous êtes curieux et que vous désirez comprendre ce langage, vous pouvez lire le [tutoriel du mode text](http://aseba.wikidot.com/fr:thymiotutoriel) (<http://aseba.wikidot.com/fr:thymiotutoriel>).

Écrire un programme

Quand vous démarrez VPL, une zone de programmation vide est affichée ; si, après avoir construit un bout de programme, vous voulez effacer le contenu de la zone de programmation, vous pouvez cliquer sur . Un programme dans VPL consiste en une ou plusieurs *paires événement-action*, chacune construite en mettant ensemble un bloc événement et un bloc action. Par exemple, la paire :



fait s'allumer la lumière sur le dessus du Thymio en rouge lorsque l'on touche le bouton avant sur le robot.

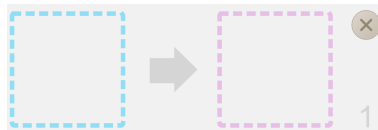


Information importante



La signification d'une paire événement-action est la suivante :

Lorsque l'événement se produit, le robot fait l'action.

Construisons une paire événement-action ensemble. Dans la zone de programmation, vous voyez ceci :



Le carré de gauche, en bleu clair, est l'espace pour l'événement. Le carré de droite, en rose, est l'espace pour l'action. Pour amener un bloc depuis les côtés (zones 4 et 5 de la Figure 1.3) dans la zone de programmation, vous pouvez simplement cliquer dessus ou le glisser jusqu'au carré correspondant en maintenant le bouton gauche de la souris enfoncé.

Commencez par amener le bloc d'événement boutons  sur le carré bleu. Ensuite, amenez le bloc d'action couleur du haut  sur le carré rose. Et voilà ! Vous avez construit une paire événement-action !



Il nous faut maintenant modifier l'événement et l'action pour qu'ils fassent ce que l'on veut. Pour l'événement, cliquez sur le bouton avant (le triangle supérieur) ; il va devenir rouge :



Cela signifie qu'**un événement se produira lorsque le bouton avant de Thymio sera touché.**


Le bloc d'action couleur contient trois barres : la rouge, la verte et la bleue. En mélangeant ces trois couleurs primaires, toutes les couleurs peuvent être créées. Vous

pouvez faire glisser les petits blocs blancs le long des barres de couleurs pour choisir quelle quantité de rouge, de vert et de bleu vous voulez afficher sur Thymio. Ces blocs sont appelés des *sliders*. Essayez de bouger le slider de la ligne rouge tout à droite en laissant les autres tout à gauche. La couleur sera toute rouge sans bleu ni vert :




Sauvegarder le programme



Avant de lancer votre programme, sauvegardez-le sur votre ordinateur. Cliquez sur l'icône  de la barre d'outils. Vous devrez choisir un nom pour votre programme, par exemple **afficher-rouge**. Choisissez l'endroit où vous voulez sauvegarder le programme, sur le bureau par exemple, et cliquez sur Sauvegarder.

Lancer le programme



Pour lancer le programme, cliquez sur  dans la barre d'outils. Essayez maintenant d'appuyer sur le bouton avant de Thymio, il devrait s'être allumé en rouge !

Félicitations !

Vous avez créé et exécuté votre premier programme, qui fait que :
Lorsque l'on touche le bouton avant de Thymio, il devient rouge.

Éteindre le robot

Lorsque vous avez terminé de jouer avec Thymio, vous pouvez l'éteindre en touchant son bouton central et en gardant le contact quatre secondes. Vous entendrez quelques notes et Thymio s'arrêtera. Tant que le robot est connecté à un ordinateur, sa batterie continue à se recharger. Une petite lumière rouge à l'arrière du robot, juste à côté du câble USB, permet de savoir s'il est rechargé ou pas. La lumière passe du rouge au bleu pour indiquer qu'il est complètement rechargé, comme sur la Figure 1.4. Vous pouvez déconnecter le câble quand vous n'utilisez pas le robot.

Truc

Si vous voulez charger le robot plus vite, vous pouvez le connecter à une prise murale avec un chargeur pour téléphone portable fournissant une prise micro-USB.

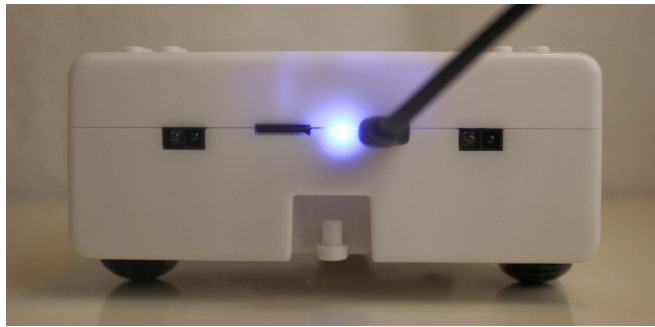




FIGURE 1.4 – L’arrière de Thymio avec le câble microUSB et le témoin de charge


Si le câble USB se déconnecte durant la programmation, VPL attendra une reconnexion. Vérifiez les deux côtés du câble, débranchez et rebranchez le câble, et regarder si VPL fonctionne. Si vous avez un problème, vous pouvez toujours fermer VPL, reconnecter le robot et réouvrir VPL.

Modifier un programme

- Pour effacer une paire événement-action, cliquez sur , en haut à droite de la paire.
- Pour ajouter une paire événement-action, cliquez sur , disponible en dessous de chaque paire.
- Pour déplacer une paire événement-action, maintenez le bouton gauche de la souris enfoncé sur une paire et glissez la à l’endroit désiré.




Ouvrir un programme existant

Si vous voulez continuer un programme que vous aviez commencé précédemment, il suffit de l’ouvrir avec l’interface VPL pour le modifier ou l’améliorer. Cliquez sur l’icône  et sélectionnez le fichier que vous voulez ouvrir, par exemple **afficher-rouge**. Les paires événement-action du programme vont être affichées dans la zone de programmation, et vous pouvez continuer à les modifier.



Les autres possibilités de l’interface VPL


Dans la barre d’outils, vous trouverez les fonctionnalités suivantes :

- **Sauver sous**  : Cliquez sur cet icône si vous voulez enregistrer votre programme sous un *autre nom* ou à un autre endroit sur votre ordinateur. Ce




bouton est utile pour commencer un nouveau programme en partant d'un autre comme base.




- **Stop**  : Ce bouton stoppe l'exécution du programme sur le robot et arrête le robot en réglant la vitesse des moteurs à zéro.




- **Changer de couleurs**  : Vous avez la liberté de choisir une autre paire de couleurs le fond des blocs d'événement et d'action.



- **Mode avancé**  : Le mode avancé permet l'utilisation de variables d'état comme expliqué dans le Chapitre 8.



- **Aide**  : Affiche l'aide de VPL dans votre navigateur web (une connexion internet est nécessaire). Cette aide est disponible sur <https://aseba.wikidot.com/fr:thymio>.



Chapitre 2

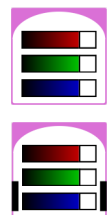
Changer les couleurs

Colorer Thymio

Créons un programme qui affiche deux couleurs différentes sur le dessus de Thymio lorsque les boutons avant ou arrière sont touchés, et deux autres couleurs sous et sur les côtés de Thymio lorsque les boutons gauche ou droite sont touchés.


Programme : **colors.aesl**

Nous avons besoin de quatre paires d'événement-action. Il y a quatre événements — toucher l'un des quatre boutons — et une action couleur est associée avec chaque événement. Notez la différence entre le bloc  et le bloc . Le premier des deux change la couleur affichée sur le dessus de Thymio alors que le deuxième change la couleur affichée sur le dessous et sur les côtés. Le deuxième bloc a deux marques noires qui représentent les roues du robot.



Ce programme est illustré sur la Figure 2.1(a).

Quelles couleurs seront affichées ? Pour les premières trois actions, le slider d'une couleur a été glissé tout à droite alors que les autres sont restés à gauche. Ces actions affichent donc respectivement purement du rouge, du bleu et du vert. L'action associée avec le bouton gauche mixe du rouge et du vert, ce qui produit donc du jaune. Vous pouvez voir que le fond de l'action couleur se colorie en fonction de la position des *sliders*, cela vous montre de quelle couleur sera Thymio !

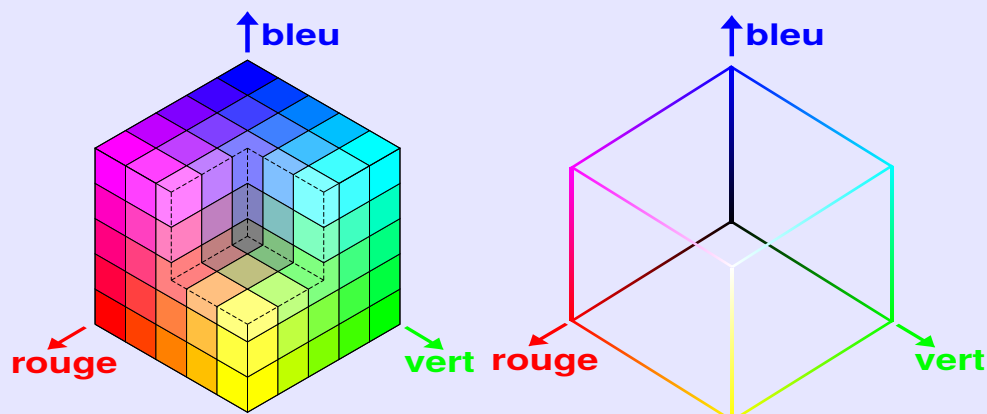
Lancez le programme (icône ) et vérifiez que toucher les boutons change les couleurs du robot. La Figure 1.1 montre Thymio illuminé en rouge sur le dessus et la Figure 3.2 montre Thymio illuminé en vert sur le dessous.



Exercice 2.1


Expérimentez avec les sliders pour voir quelles couleurs peuvent être affichées.

 **Information**



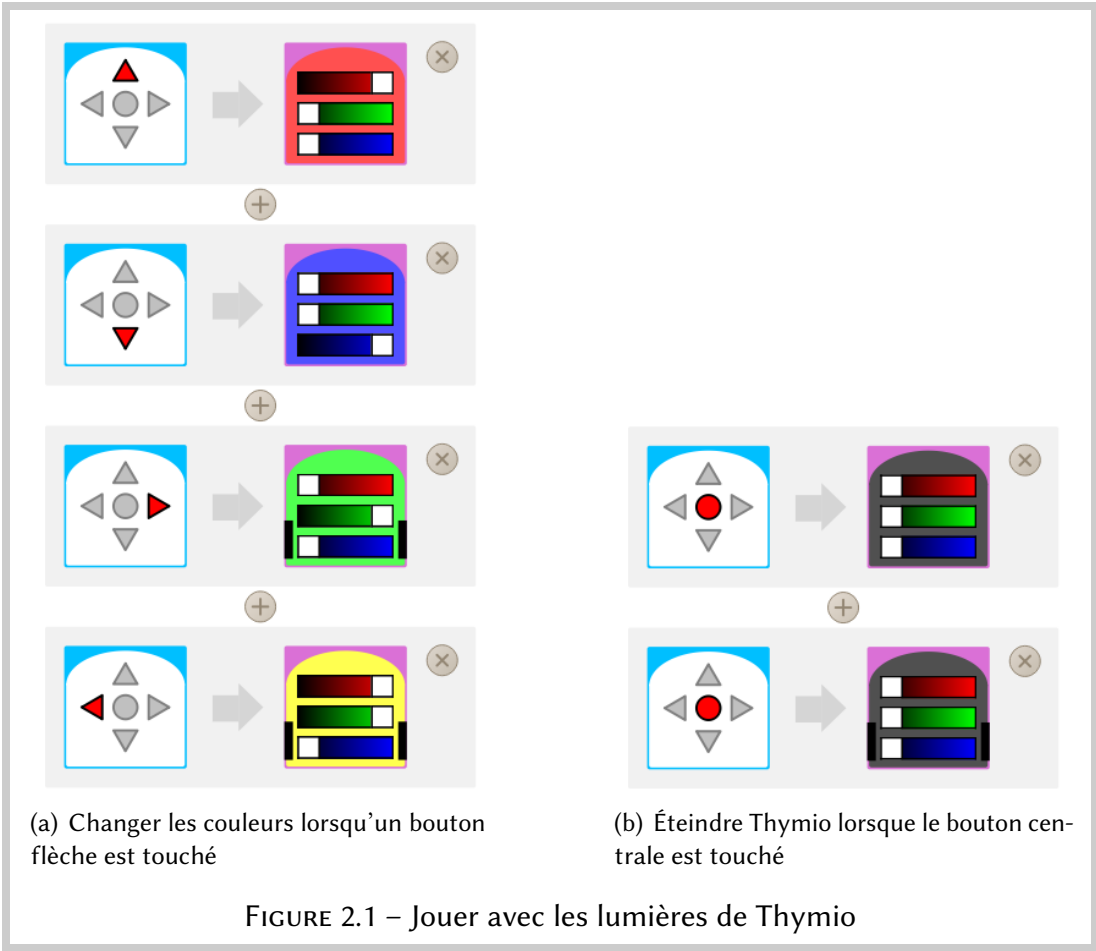
En mixant ensemble du rouge, du vert et du bleu, vous pouvez faire n'importe quelle couleur !

Éteindre les lumières

Modifions maintenant le programme pour que les lumières s'éteignent lorsque le bouton central est touché. Nous allons avoir besoin de deux paires événement-action, une pour éteindre les lumières du dessus de Thymio et une autre pour les lumières du dessous. En faisant glisser tous les *sliders* sur la gauche, comme sur la Figure 2.1(b), la lumière sera éteinte. Vous voyez que l'événement est le même, toucher sur le bouton central, mais l'action associée est différente, éteindre les lumières du haut ou du bas. N'oubliez pas de lancer le programme en cliquant sur l'icône . À l'avenir, il sera implicite qu'il vous faudra lancer chaque programme en cliquant sur cet icône, nous ne vous le dirons plus.

 **De multiples paires événement-action**

- Lorsqu'un programme est lancé, toutes les paires d'événement-action sont actives.
- Il est possible d'avoir plusieurs fois le même événement mais il faut que le bloc d'action associé soit différent.
- Si l'événement et le bloc d'action sont identiques dans plusieurs paires, VPL vous indiquera qu'il y a une erreur (zone 3 dans la Figure 1.3). Vous ne pourrez pas lancer le programme tant qu'il y a des erreurs.

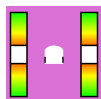



Chapitre 3

Thymio en mouvement

En avant, en arrière

Thymio a deux moteurs, un sur chaque roue. Ils peuvent tourner dans les deux sens, permettant à Thymio d'avancer, de reculer et de tourner. Commençons par un petit programme qui vous apprendra à contrôler les moteurs.



Le bloc d'action des moteurs  représente Thymio entouré de deux *sliders*. Chaque *slider* contrôle un moteur. En les faisant glisser vers l'avant, Thymio avancera, à l'inverse, en les faisant glisser vers l'arrière, Thymio reculera. Pour arrêter les moteurs, il suffit de glisser les *sliders* au centre des barres. Écrivons un programme qui fasse avancer Thymio lorsque l'on touche le bouton avant et qui le fasse reculer l'on touche le bouton arrière.

Programme **moving.aesl**

Nous allons avoir besoin de deux paires événement-action. Une pour faire avancer Thymio, l'autre pour le faire reculer, comme sur la Figure 3.1. Amenez les blocs correspondants dans la zone de programmation, choisissez le bouton que vous souhaitez utiliser et ajustez les *sliders* des moteurs pour faire avancer Thymio dans un cas et pour le faire reculer dans l'autre. Plus vous positionnez les *sliders* vers les extrémités, plus les moteurs tourneront vite. Essayez d'abord avec une vitesse moyenne.

Lancez maintenant le programme et touchez les boutons avant et arrière du robot pour voir si Thymio avance et recule !

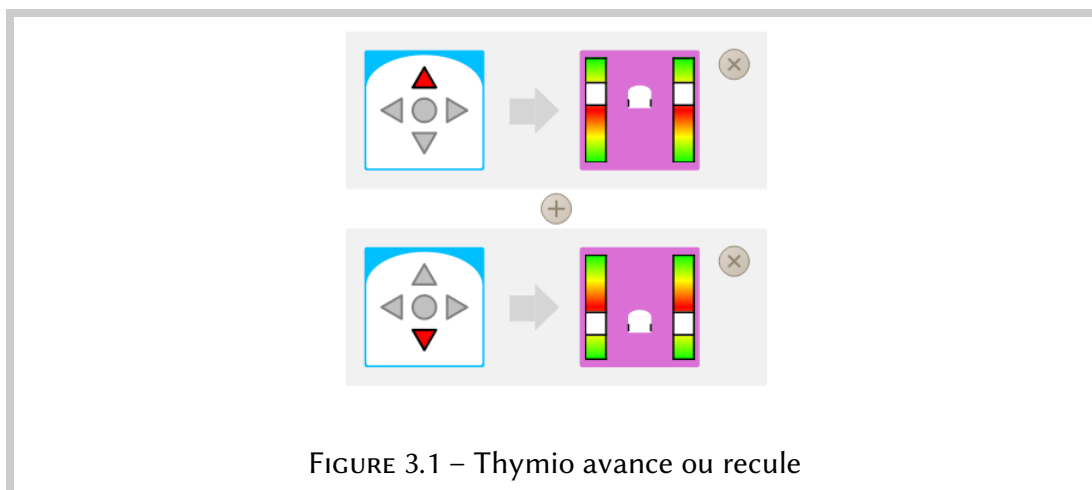



FIGURE 3.1 – Thymio avance ou recule

Arrête-toi !

À l'aide! Thymio ne veut plus s'arrêter!

Cliquez sur l'icône  pour arrêter Thymio. Nous allons corriger ce problème en ajoutant une paire événement-action dans le programme :



Avec ces deux blocs, Thymio s'arrêtera lorsqu'on touche son bouton central. Lorsque vous ajoutez un bloc moteur dans le programme, il est réglé avec les *sliders* en position médiane, de sorte que les moteurs s'arrêtent.

Ne tombe pas de la table

Si Thymio se trouve sur le sol, au pire, il rentrera dans un mur ou se débranchera tout seul de votre ordinateur, mais s'il est sur une table, il risque de tomber! Nous allons créer un petit programme qui lui permettra de s'arrêter s'il arrive au bord de la table.



Attention !


Si Thymio roule sur une table, tenez-vous toujours prêt à le rattraper s'il arrive près du bord de la table!

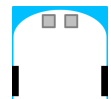
Tournez Thymio sur son dos. Vous verrez deux petits rectangles noirs qui contiennent des éléments optiques, comme nous le voyons sur le haut de la Figure 3.2. Ce sont les *détecteurs de sols*! Ils envoient une impulsion de lumière infrarouge et mesurent la quantité de lumière qui leur est réfléchi. Si Thymio est posé sur une table de couleur claire, beaucoup de lumière sera réfléchi, alors que s'il dépasse le bord de la table, peu de lumière sera réfléchi. Nous allons donc utiliser ces capteurs pour dire à Thymio de s'arrêter lorsqu'il arrive au bord de la table.



Truc

Utilisez une table est de couleur claire et évitez les tables en verre transparent, elles ne réfléchiront probablement pas la lumière et Thymio croira qu'il n'est pas sur une table!

Amenez le bloc détecteur de sols  sur la zone de programmation pour commencer. Les deux petits carrés gris représentent les détecteurs de sols. En cliquant sur ces carrés, ils passent de gris à rouge, à blanc puis à nouveau à gris, etc. Pour ce bloc, ces couleurs signifient :



- **Gris** : Le détecteur n'est pas utilisé.



FIGURE 3.2 – Le dessous de Thymio avec ses détecteurs de sols

- **Rouge** : L'action associée est déclenchée s'il y a beaucoup de lumière réfléchiée.
- **Blanc** : L'action associée est déclenchée s'il y a peu de lumière réfléchiée.

Information

La couleur d'un bloc est arbitraire, celles que nous avons choisie nous paraissent indicatives, mais d'autres pourraient être choisies.

Pour faire que Thymio s'arrête au bord de la table, lorsque qu'il y a peu de lumière réfléchiée, cliquez les carrés jusqu'à ce qu'ils soient blancs et créez la paire événement-action suivante :



Placez Thymio près d'un bord d'une table de façon à ce qu'il soit face au bord et touchez le bouton avant. Thymio devrait avancer jusqu'au bord et s'y arrêter.

Exercice 3.1

Jouez avec le bloc action moteurs de Thymio. À sa vitesse maximale, Thymio est-il toujours capable de s'arrêter avant le bord de la table ? Si non, à partir de quelle vitesse le robot ne peut plus s'arrêter ? Pouvez-vous empêcher le robot de tomber s'il va en arrière ?



Truc

Lorsque j'ai lancé le programme, le robot *est* tombé. La raison était que mon bureau a un bord arrondi ; d'ici à ce que le robot ait détecté un faible niveau de lumière, il n'était déjà plus stable et a basculé. Si vous voulez arrêter Thymio un peu avant le bord de la table, vous pouvez placer une feuille noir, ou du ruban adhésif noir, là où vous souhaitez qu'il s'arrête !

Chapitre 4

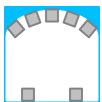
Un robot de compagnie


Un *robot autonome* adopte un comportement spécifique en fonction de la situation dans laquelle il se trouve. Il réussit à réagir grâce au *feedback*, littéralement de l'information en retour. Il faut donc que le robot puisse « voir » le monde qui l'entoure pour pouvoir y réagir.

Thymio vous obéit

Pour commencer, nous allons programmer Thymio pour qu'il vous obéisse. Normalement, le robot restera sur place sans bouger ; quand il détectera votre main devant lui, il bougera en sa direction.

Thymio a cinq capteurs de distance horizontaux à l'avant et deux à l'arrière. Ils sont similaires à ceux qui se trouvent sous Thymio et que nous avons utilisés au Chapitre 3. Avancez votre main en direction des capteurs à l'avant de Thymio, vous verrez une lumière rouge apparaître à côté du capteur qui vous aura détecté, comme sur la Figure 4.1.



Le bloc  sert à utiliser les capteurs horizontaux avant et arrière de Thymio. Les petits carrés gris (cinq sur l'avant et deux sur l'arrière) sont utilisés comme pour les détecteurs situés sous Thymio. En cliquant dessus, ils passent de gris à blanc, à rouge et à nouveau à gris. Pour ce bloc, les significations des couleurs sont :

- **Gris** : Le détecteur n'est pas utilisé.
- **Rouge** : L'événement associé est déclenché si un objet se trouve proche.
- **Blanc** : L'événement associé est déclenché si aucun objet ne se trouve proche.



FIGURE 4.1 – L'avant de Thymio. Deux doigts sont détectés par les capteurs avant.

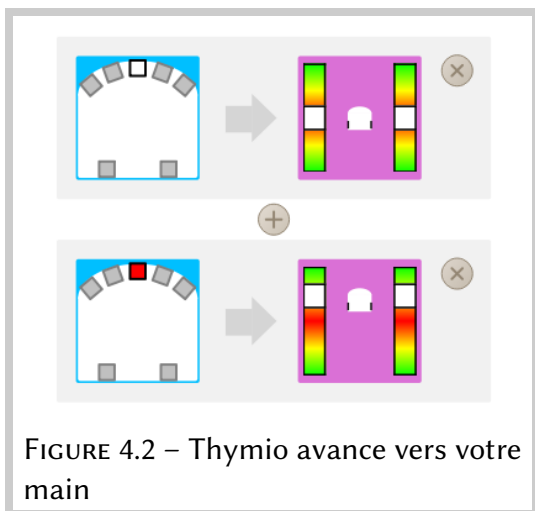


FIGURE 4.2 – Thymio avance vers votre main

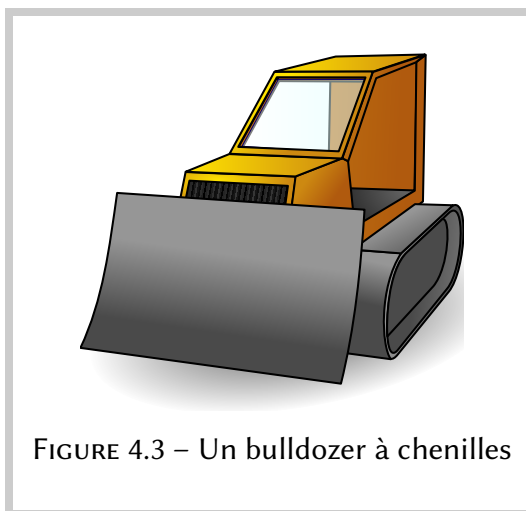


FIGURE 4.3 – Un bulldozer à chenilles

ⓘ Détecteurs de sols et capteurs de distance horizontaux

Attention à ne pas confondre le comportement des capteurs de distance horizontaux avec celui des détecteurs de sols.

- Pour les capteurs horizontaux, le carré blanc spécifie qu'un événement se produira s'il n'y a *rien à proximité*, alors qu'un carré rouge spécifie qu'un événement se produira s'il y a *quelque chose à proximité*.
- Pour les détecteurs de sols, le carré blanc spécifie qu'un événement se produira s'il y a *peu de lumière réfléchi par la surface* alors qu'un carré rouge spécifie qu'un événement se produira s'il y a *beaucoup de lumière réfléchi par la surface*.


Le principe physique de ces deux types de capteurs est similaire, mais parce qu'ils sont placés différemment, leur comportement est différent.

Pour construire le comportement, il nous faut deux paires événement-action, comme sur la Figure 4.2. Dans ce programme, vous voyez que dans la première paire, le carré central est blanc et l'action associée est que les moteurs sont arrêtés. Ainsi, lorsque le robot ne voit rien, il ne bougera pas ; et s'il bougeait, il s'arrêtera. Dans la deuxième paire, le carré central est rouge et les sliders du bloc action moteur sont glissés vers le haut. Ainsi, lorsque vous amenez votre main près de l'avant du robot, un événement se produit qui fait tourner les deux moteurs assez vite et fait avancer le robot en avant.

Faire tourner Thymio


Thymio n'a pas un volant comme une voiture ou un guidon comme un vélo. Comment tourne-t-il donc ? Pour tourner, le robot utilise une *direction différentielle*, ou *differential drive*, qui est utilisée par de nombreux véhicules à chenilles comme le bulldozer sur la Figure 4.3. À la place de tourner un guidon dans la direction désirée, les chenilles ou roues gauches et droites sont commandées par des moteurs individuels à des vitesses

différentes. Si la roue droite tourne plus vite que la roue gauche, alors le véhicule tournera à gauche, tandis que si sa roue gauche tourne plus vite que sa roue droite, il tournera à droite.

Dans VPL, en réglant le bloc action moteur avec les *sliders* à des endroits différents, les roues de Thymio ne tourneront pas à la même vitesse, ce qui fera tourner le robot. Plus la différence de vitesse est élevée, plus le virage sera serré. Pour arriver à une grande différence de vitesses, vous pouvez faire tourner une roue en avant et l'autre en arrière. En fait, pour tourner sur lui-même, il suffit à Thymio de faire tourner ses deux roues à la même vitesse mais dans des sens opposés ! Par exemple, dans ce bloc action moteur , le *slider* gauche indique une grande vitesse en arrière, alors que le *slider* droite indique une grande vitesse en avant. Le résultat est que le robot tournera sur lui-même en direction de la gauche, comme indiqué par l'image du robot.

Expérimentez avec une paire événement-action telle que celle ci :



Si vous chargez ensuite ce programme et appuyez sur le bouton central, Thymio devrait tourner sur lui-même. Vous pouvez toujours l'arrêter en appuyant sur .

Truc

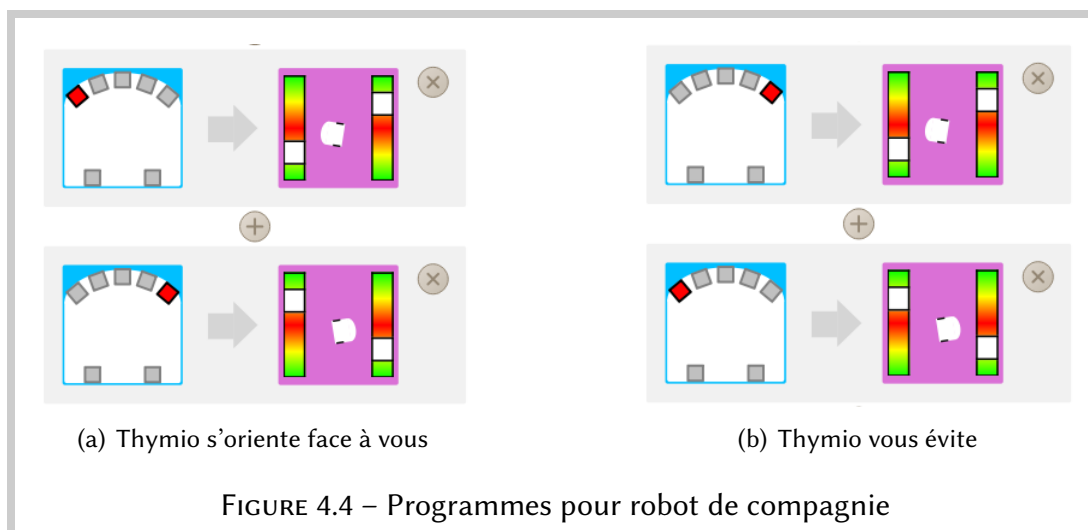
L'icône du Thymio au centre du bloc action moteur s'anime dès que vous réglez les *sliders* pour vous donner une idée du mouvement de Thymio !

Thymio vous aime

Un vrai animal de compagnie ne se contente pas de s'approcher ou de s'éloigner de vous, il vous suit un peu partout ! Pour que Thymio puisse vous suivre le plus fidèlement possible, il faudra ajouter deux paires événement-action au programme précédant. Si Thymio vous détecte avec son capteur avant-droit, il doit tourner à droite et s'il vous détecte avec son capteur avant-gauche, il doit tourner à gauche.

Programme **likes.aesl**

Une façon de faire est illustrée sur la Figure 4.4(a). Vous pouvez essayer différentes vitesses, le faire tourner sur lui-même ou non, afin de trouver le meilleur comportement !



Exercice 4.1

Modifiez le comportement du robot de compagnie pour qu'il bouge en avant quand le programme est exécuté et qu'il s'arrête lorsqu'il détecte le bord de la table (ou une bande de ruban adhésif noir).

Comme vu dans le Chapitre 3, beaucoup de lumière sera réfléchiée par une surface blanche, et peu par une surface noire. Vous allez devoir expérimenter avec le bloc capteurs horizontaux pour déterminer quand choisir un carré blanc et quand choisir un carré rouge, en fonction du sol ou de la table sur lequel vous placez le robot.

Exercice 4.2

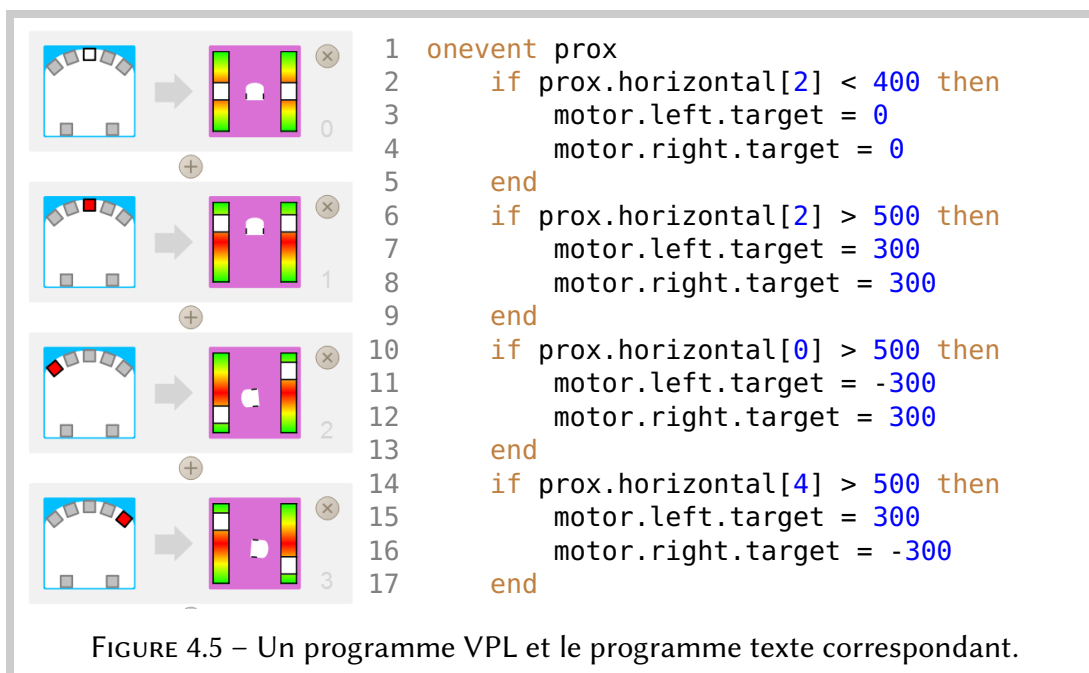
Qu'arrive-t-il si vous changez l'ordre des paires d'événement-action utilisées à l'exercice précédent ?

Thymio ne vous aime pas

Parfois, même le plus fidèle animal de compagnie n'a pas envie de vous suivre. Écrivez un programme qui génère ce comportement.

Programme **does-not-like.aesl**

Ouvrez le programme du Thymio qui vous aime et inversez l'association des événements avec les actions, comme montré sur la Figure 4.4(b). Détecter un obstacle avec le capteur gauche fait tourner le robot à droite, et détecter un obstacle avec le capteur droite fait tourner le robot à gauche.



Exercice 4.3

Jouez avec les différents capteurs avant de Thymio. Les capteurs horizontaux avant sont numérotés 0, 1, 2, 3, 4 de gauche à droite. Les capteurs arrière sont numérotés 5 pour le gauche et 6 pour le droite. À la place d'utiliser les capteurs 0 et 4 comme jusqu'à maintenant :

- Utilisez les capteur 1 pour tourner le robot à gauche et le 3 pour à droite.
- Utilisez à la fois les capteurs 0 et 1 pour tourner le robot à gauche et à la fois les capteurs 3 et 4 pour tourner le robot à droite.
- Ajouter une paire événement-action pour les capteurs arrières 5 et 6.

Régler les sliders plus précisément (avancé)

Ce n'est pas très facile de régler les *sliders* précisément pour, par exemple, faire avancer Thymio tout droit. En regardant la traduction textuelle des paires événement-action, il est possible d'ajuster plus précisément les vitesses des roues. Le Figure 4.5 montre le programme du Thymio qui vous suit, avec la traduction texte à droite. Ce texte est écrit automatiquement quand vous éditez les paires événement-action.

La ligne `onevent prox` signifie que les lignes qui suivent seront exécutées lorsque la lecture des capteurs de distance (appelés *proximity* et abrégés *prox*) se produit (elle se produit 10 fois par seconde).

Lorsque l'événement se produit, Thymio teste la valeur des capteurs en utilisant une condition de type `if ... then ... end`, c'est à dire `si ... alors ... fin`. Il commence par tester le capteur numéro 2 (avant centre) comme nous le voyons avec

`prox.horizontal[2]`. Si cette valeur est inférieure à 400, alors Thymio règle la vitesse des moteurs gauche et droite à 0 avec les instructions `motor.left.target = 0` et `motor.right.target = 0`. Chaque bloc `if ... then ... end` teste un capteur spécifique et effectue ou non l'action associée, en fonction du résultat du test. Il correspond donc à une paire événement-action :

0. teste si rien ne se trouve devant ; si c'est le cas, Thymio s'arrête.
1. teste s'il y a quelque chose devant ; si c'est le cas, Thymio avance.
2. teste s'il y a quelque chose à gauche ; si c'est le cas, Thymio tourne à gauche.
3. teste s'il y a quelque chose à droite ; si c'est le cas, Thymio tourne à droite.

Finalement, une fois que Thymio a testé tous ces capteurs, il attend le prochain événement `PROX` et recommence ces tests, indéfiniment. Pour apprendre comment éditer le mode texte, voir le Chapitre 10.



Truc

En bougeant les *sliders* des blocs d'action moteur, vous verrez les vitesses visées pour les moteurs (`motor.X.target`) changer par pas de 50 dans l'intervalle -500 à 500 . En bougeant les *sliders* avec soin, vous pouvez choisir n'importe laquelle de ces valeurs.

Chapitre 5



Thymio est sur une piste

Une promenade en montagne peut sembler être une activité très simple. Il suffit d'enfiler une paire de chaussures de marche et de suivre le chemin. Pour un robot, suivre un chemin peut être très utile. Considérez un entrepôt avec des charriots robotiques qui transportent des objets. Il y a des lignes peintes sur le sol de l'entrepôt et le robot reçoit comme instructions de suivre certaines lignes jusqu'à ce qu'il arrive à la zone de stockage de l'objet qu'il transporte. Écrivons un programme qui permette à Thymio de suivre une ligne noire sur une table blanche.

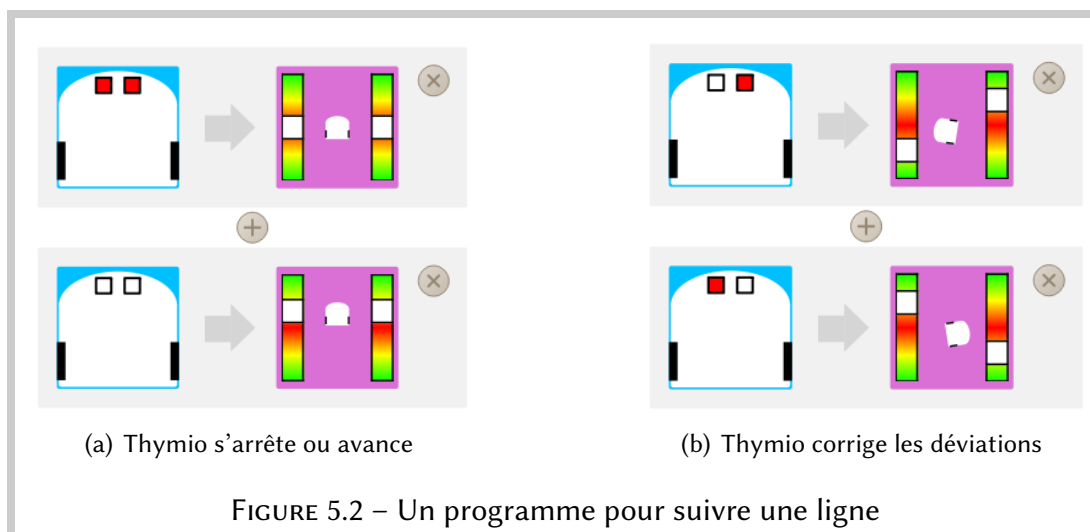
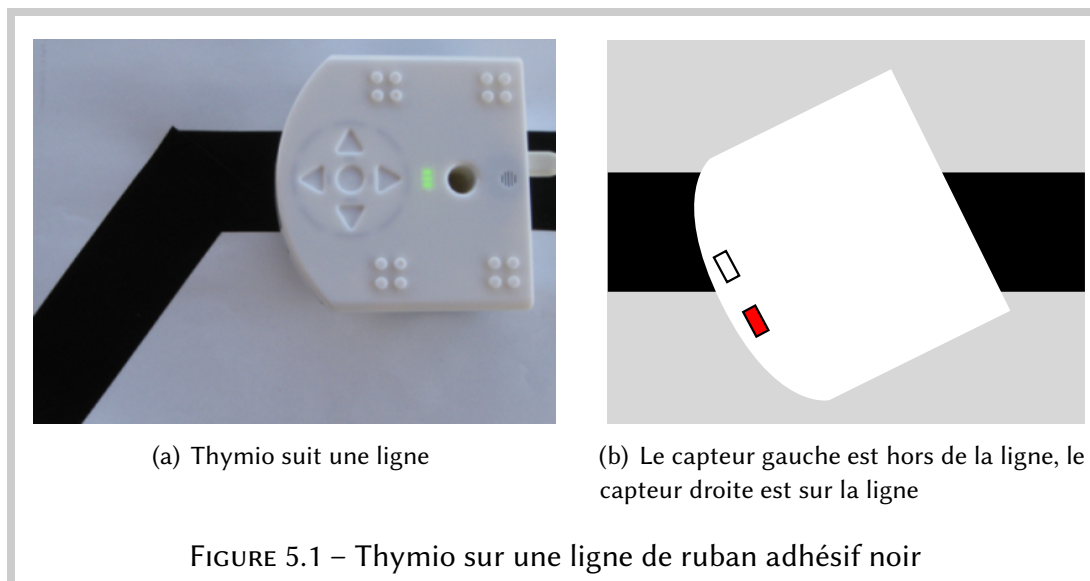
Programme **follow-line.aesl**


Suivre une ligne sur le sol illustre le genre de difficultés rencontrées en robotique, dues à l'incertitude de la perception du monde et de la méconnaissance de l'action du robot dans le monde. Par exemple, la ligne peut être mal dessinée, de la saleté peut la recouvrir, ou de la poussière peut faire tourner une des roues du robot plus vite que l'autre. Pour suivre une ligne malgré ces difficultés, le robot doit utiliser un *contrôleur*, qui aura pour tâche de définir la puissance à appliquer à chaque moteur en fonction des données reçues par les capteurs.

La ligne et le robot

Pour suivre une ligne, nous allons utiliser les détecteurs de sol que nous avons déjà utilisés dans le Chapitre 3. Rappelons qu'ils fonctionnent en envoyant de la lumière infrarouge (invisible pour l'œil humain) et en mesurant la quantité de lumière renvoyée. Si vous posez Thymio sur une couleur claire, beaucoup de lumière sera réfléchi, et donc l'événement  sera déclenché. Nous avons donc besoin d'une ligne qui déclenchera un événement quand peu de lumière sera réfléchi . Ceci est facile à réaliser en imprimant une bande noire, en la peignant ou en utilisant du ruban adhésif noir comme sur la Figure 5.1(a). La ligne doit être assez large pour que les deux capteurs de sol voient du noir quand le robot suit la ligne correctement. Une largeur de 5 centimètres est suffisante pour que le robot suive la ligne même s'il y a de petites déviations.

Le premier pas pour faire suivre une ligne à Thymio est de le faire avancer s'il est sur la ligne (les *deux* capteurs de sol détectent du foncé), et de le faire s'arrêter s'il ne se trouve pas sur la ligne (les *deux* capteurs de sol détectent du clair). Les paires événement-action qui sont illustrées sur Figure 5.2(a).



 **Truc**

Assurez-vous d'avoir un câble USB assez long (disons, deux mètres) pour que le robot reste connecté même quand il bouge. Vous trouverez des rallonges dans les magasins d'informatique.

Votre premier contrôleur

Le prochain pas consiste à programmer le contrôleur qui suit la ligne :

- Si le robot sort de la ligne à *gauche*, comme sur la Figure 5.1(b), le capteur *gauche* détectera le sol alors que le capteur *droite* continuera de détecter la ligne ; dans ce cas le robot doit tourner légèrement à *droite*.

- Si le robot sort de la ligne à *droite*, le capteur *droite* détectera le sol alors que le capteur *gauche* continuera de détecter la ligne ; dans ce cas le robot doit tourner légèrement à *gauche*.

Comme nous avons deux conditions, il nous faut deux paires événement-action, comme illustré sur la Figure 5.2(b).

Régler les paramètres

Il est assez facile de comprendre que si Thymio sort de la ligne à gauche, il doit tourner à droite, comme dans la Figure 5.1(b). La vraie question est plutôt : de combien doit-il tourner ? Si la rotation est trop douce, le capteur droit pourrait aussi sortir de la ligne avant que le robot ne tourne assez ; si au contraire le tour est trop serré, le robot pourrait sortir de la ligne de l'autre côté. Dans tous les cas, des tours trop forts peuvent être dangereux pour le robot et ce qu'il transporte.

Dans ce programme, vous pouvez définir la vitesse des roues gauche et droite dans chaque bloc d'action moteur. Vous devrez expérimenter avec ces valeurs jusqu'à ce que le robot bouge de manière *fiable*. Par fiable, nous entendons que le robot peut réussir plusieurs fois l'expérience de suivi de ligne. Comme à chaque expérience vous placez le robot sur la ligne à différentes positions et pointant dans différentes directions, il faut tester plusieurs fois pour être convaincu que le programme fonctionne.

Il y a plein de façons différentes de configurer les blocs d'action moteur. Thymio doit-il aller vite lorsqu'il est sur la ligne ou, au contraire, lentement ? En allant vite, vous améliorerez son efficacité pour se déplacer d'un point à un autre mais il risquera de sortir de la ligne avant de pouvoir corriger sa direction. À l'inverse, s'il va trop lentement, personne n'achètera votre robot pour l'utiliser dans un entrepôt. De plus, une fois que Thymio remarque qu'il quitte la ligne, que doit-il faire ? S'il fait tourner un moteur dans un sens et l'autre dans le sens opposé, vous vous assurez qu'il ne quittera pas la ligne mais ses mouvements seront très saccadés. S'il corrige simplement sa trajectoire en diminuant la vitesse d'une roue, il se déplacera avec fluidité, mais il risquera de quitter complètement la ligne. Ainsi, il faudra trouver de bons compromis.

Exercice 5.1

Thymio s'arrête complètement s'il ne détecte plus du tout la ligne. Modifiez le programme pour qu'il tourne lentement sur lui-même vers la gauche jusqu'à ce qu'il retrouve la ligne. Essayez-le sur une ligne avec un virage à gauche comme sur la Figure 5.1(a). Essayez d'augmenter la vitesse en avant du robot. Que se passe-t-il lorsque le robot arrive à la fin de la ligne ?

Exercice 5.2

Modifiez le programme de l'exercice précédent pour que le robot tourne à droite quand il arrive au bout de la ligne. Que se passe-t-il ?

Il serait bien de pouvoir se *rappeller* quel capteur était le dernier à perdre le contact avec la ligne afin de faire tourner le robot dans la bonne direction pour retrouver la ligne. Dans le Chapitre 8, nous apprendrons à Thymio à se rappeler ces informations.

Exercice 5.3

Jouez avec différentes formes de parcours :

- Virages doux
- Virages serrés
- Zig-zag
- Ligne plus large
- Ligne plus étroite

Faites la course avec vos amis : Quel robot suit le plus de lignes différentes avec succès ? Pour chaque ligne, quel robot la suit dans le moins de temps ?

Exercice 5.4

Discutez les effets que les modifications suivantes au Thymio auraient sur les capacités du robot à suivre une ligne :

- L'événement de lecture des capteurs de sol arrive plus ou moins souvent que 10 fois par seconde ;
- Les capteurs sont plus éloignés ou proche l'un de l'autre ;
- Il y a plus que deux capteurs de sol sous le robot.

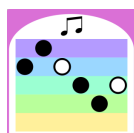
Chapitre 6

Sons et chocs

Faisons une pause avec les tâches compliquées et jouons un peu avec Thymio. Dans ce chapitre, nous vous montrerons que Thymio peut jouer de la musique, répondre à un son ou à un petite tape amicale !

Thymio mélomane

Thymio possède un synthétiseur de sons qui lui permet de jouer des notes de musique ! Vous pouvez le programmer simplement en utilisant le bloc action :



Programme **bells.aesl**

Vous ne deviendrez pas le nouveau Beethoven — on ne peut jouer qu'un ensemble de six notes, sur cinq hauteurs et deux longueurs différentes — mais vous pouvez composer une petite musique qui donnera de la personnalité à votre robot. La Figure 6.1 illustre deux chansons différentes jouées par Thymio lorsque vous touchez les boutons avant ou arrière.

Les petits cercles sont les six notes. Chaque note doit être placée sur une barre de couleur représentant une hauteur. Il suffit de cliquer avec votre souris sur la barre de couleur de votre choix au niveau de la note que vous souhaitez modifier et voilà !

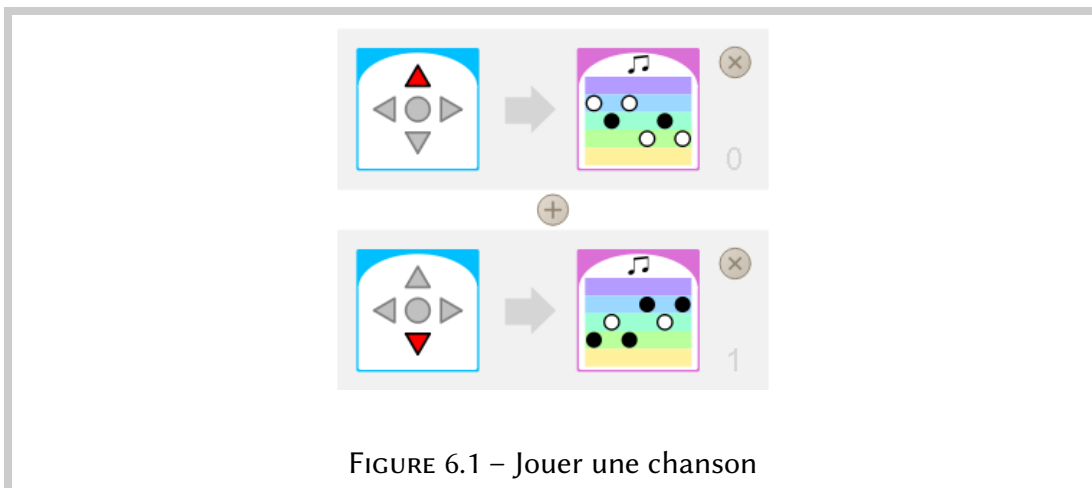



FIGURE 6.1 – Jouer une chanson

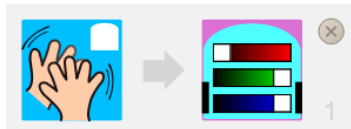
Ensuite, vous pouvez choisir entre jouer une noire (courte), et une blanche (deux fois plus longue), en cliquant sur la note que vous souhaitez modifier. N'essayez pas de glisser/déplacer une note, ça ne fonctionnera pas.

Exercice 6.1

Écrivez un programme qui vous permet d'envoyer un [code Morse](#). Les lettres dans les codes Morse sont encodées par des séquences de longs sons (*traits*) et de courts sons (*points*). Par exemple, la lettre *V* est encodée par trois points suivis par un trait.

Contrôlez votre robot par le son

Thymio a un microphone, il peut donc réagir à un son ! L'événement  se déclenche si Thymio entend un son fort, comme par exemple quelqu'un qui tape dans ses mains. La paire événement-action suivante allumera les lumières en dessous du robot lorsque vous tapez dans vos mains :



Information

Si vous vous trouvez dans un environnement bruyant, vous ne pourrez pas utiliser cet événement, car le niveau sonore sera toujours haut et l'événement s'activera à répétition.


Exercice 6.2

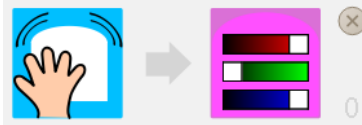
Écrivez un programme qui fasse démarrer le robot quand vous tapez dans vos mains et le fasse s'arrêter lorsque vous touchez un bouton.

Puis écrivez un programme qui fasse l'opposé : démarrer lorsque vous touchez un bouton et s'arrêter lorsque vous tapez dans vos mains.

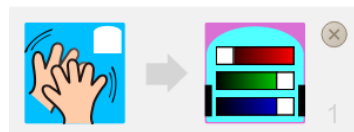
C'est bien Thymio !



Il est important de récompenser votre animal de compagnie quand il est gentil, c'est pareil avec Thymio ! Il peut vous détecter si vous lui donner une petite tape sur la tête grâce à l'événement . Par exemple, la paire événement-action suivante allume les lumières du haut lorsque vous lui donner une petite tape :



Construisez un programme avec cette paire événement-action et la paire suivante qui allume les lumières du bas du robot lorsque vous tapez dans vos mains :



Programme **whistles.aesl**

Arrivez-vous à n'allumer que les lumières du haut ? C'est difficile à faire : une tape génère un son qui est assez fort pour faire aussi s'allumer les lumières du bas. Avec un peu de pratique, on arrive à taper le robot assez gentilement pour que le son résultant ne soit pas suffisant pour être considéré.

Exercice 6.3

Écrivez un programme qui fasse avancer le robot jusqu'à ce qu'il touche un mur.

Faites attention à ce que le robot **bouge lentement** afin qu'il ne s'endommage pas.

Chapitre 7

Aimer pour un temps

Dans le Chapitre 4, nous avons programmé un robot de compagnie qui nous aimait ou nous fuyait. Considérons un comportement plus avancé : un compagnon timide qui n'arrive pas à se décider s'il nous aime ou pas. Initialement, le robot s'approchera de notre main tendue, puis il s'en éloignera. Après un moment, il changera d'avis et reviendra en direction de notre main.

Programme **shy.aesl**

Le comportement du robot est le suivant. Lorsque le bouton droite est touché, le robot tourne à droite. Quand il détecte votre main, il tourne à gauche mais après un moment il regrette sa décision et se retourne. Nous savons comment construire les paires événement-action pour le premier mouvement :




et pour se détourner quand votre main est détectée :



Le comportement de se retourner « après un moment » peut être décomposé en deux paires événement-action :

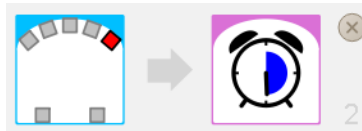
- *Quand* le robot commence à se détourner → *démarrer un minuteur* pour deux secondes.
- *Quand* le temps du minuteur est écoulé → *tourner* à droite.

Nous avons besoin d'une nouvelle *action* pour la première partie du comportement et d'un nouvel *événement* pour la seconde partie.

L'action démarre un *minuteur*, qui ressemble à un réveil . Ce minuteur est, en fait, un compte à rebours qui peut aller jusqu'à 4 secondes. Pour régler la durée du compte à rebours, vous pouvez cliquer n'importe où dans le cadran de l'alarme. Une petite animation vous montre combien de temps le compte à rebours va durer.




La paire événement-action pour cette première partie du comportement est :



Le compte à rebours est réglé sur deux secondes. Lorsque l'événement détectant votre main se produit, il y aura donc deux actions : tourner le robot à gauche et lancer le compte à rebours.



La seconde partie du comportement nécessite un événement se produisant lorsque le compte à rebours arrive à zéro, c'est le bloc d'événement *temps écoulé*  qui montre un réveil sonnante.

La paire événement-action pour faire tourner à nouveau le robot vers la droite est donc :



Exercice 7.1

Écrivez un programme qui fasse avancer le robot à vitesse maximale pour trois seconde lorsque que le bouton avant est touché ; puis qui fasse reculer le robot. Ajoutez une paire événement-action qui arrête le mouvement en touchant le bouton central.


Information

En robotique, ce genre de compte à rebours s'appelle des *timers*. Ils sont extrêmement utiles dans de nombreuses situations et vous vous en apercevrez très rapidement en créant vos propres comportements pour Thymio.

Chapitre 8

États : Pour ne pas toujours faire la même chose (avancé)

Un programme VPL est composé d'une série de paires événement-action. Tous les événements sont vérifiés périodiquement et les actions appropriées sont effectuées. Ceci limite les programmes que nous pouvons créer ; pour aller plus loin nous avons besoin d'une façon de spécifier que certaines paires événement-action sont actives à un certain moment, alors que d'autres ne le sont pas. Par exemple, dans le Chapitre 5, lorsque le robot sort de la ligne, nous aurions aimé qu'il tourne à gauche ou à droite afin de rechercher la ligne dans une direction qui dépend de quel côté il est sorti.

Les états sont disponibles dans le mode *avancé* de VPL. Cliquez sur  avant de travailler sur les projets de ce chapitre.



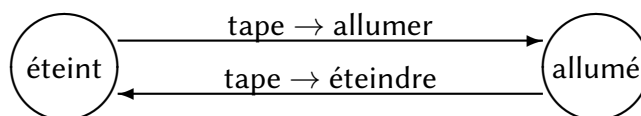
Tape, tape

Dans les programmes que nous avons réalisé jusqu'ici, nous avons souvent *démarré* Thymio en appuyant sur un de ces boutons et *arrêté* Thymio en appuyant sur un autre. Mais regardez votre ordinateur, normalement, il n'a qu'un seul bouton pour l'allumer ou l'éteindre. Le bouton se *rappelle* s'il est dans l'état **allumé** ou l'état **éteint**. Le bouton inclut souvent une petite lumière qui indique son état courant.

Écrivons un programme qui allume les lumières du robot si vous lui donnez une petite tape et qui les éteigne si vous lui donnez une seconde tape.

Programme **tap-allumé-éteint.aesl**

Il est pratique de décrire ce comportement en utilisant un *diagramme d'états* :



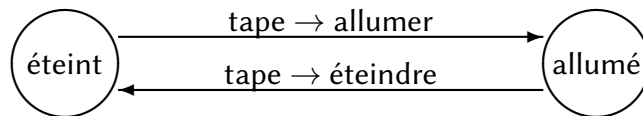
Ce diagramme comprend deux états indiqués par des cercles, **allumé** et **éteint**. Depuis l'état **éteint**, le robot peut aller dans l'état **allumé** et revenir, mais seulement en suivant les instructions sur les flèches. Les instructions décrivent quand une transition d'un état à l'autre peut se produire et comment :

- **Quand** Thymio est dans l'état **éteint** **et** que l'événement *tape* se produit → *allumer* le robot **et** aller dans l'état **allumé**.

- **Quand** Thymio est dans l'état **allumé et** que l'événement *tape* se produit → *éteindre* le robot **et** aller dans l'état **éteint**.

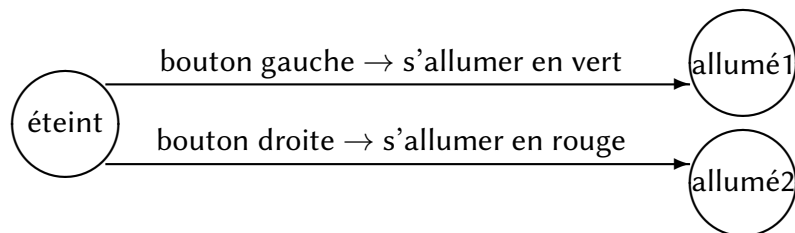
L'accent mis sur le mot « **et** » avant la flèche → signifie que deux conditions doivent être remplies pour que la transition se fasse : (a) Le robot doit être dans un certain état et (b) l'événement doit se produire. Lorsque les deux conditions sont remplies, alors la transition est prise ce qui fait à la fois changer l'état et exécute l'action écrite après la flèche →.

Il est important de réaliser que les deux parties de la condition sont indépendantes. Dans le diagramme ci-dessus (répété ici) :



l'événement *tape* apparaît deux fois, mais l'action causée par cet événement *dépend* de l'état dans lequel le robot se trouve.

De façon similaire, dans un même état, différents événements peuvent causer différentes actions et des transitions vers de nouveaux états différents. Dans le diagramme suivant :




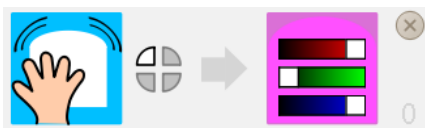
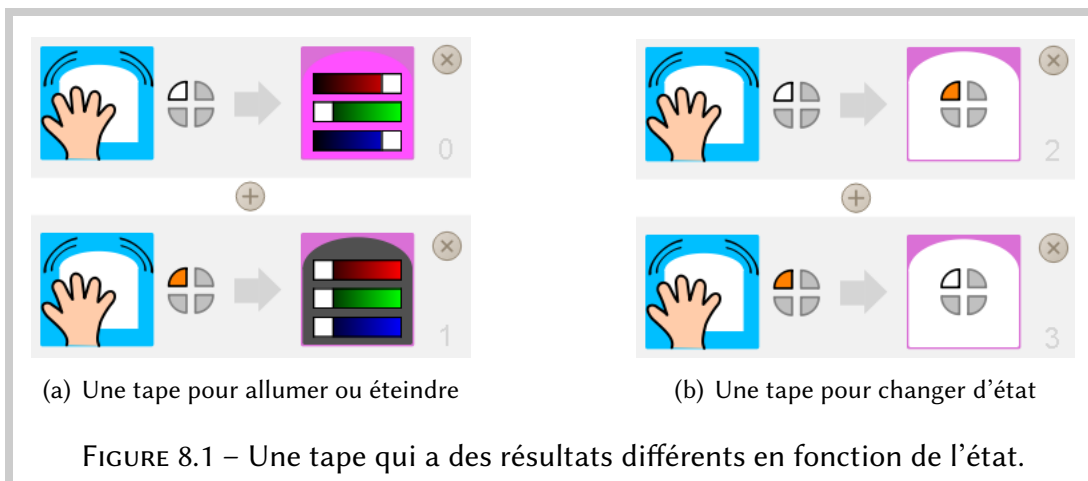
toucher le bouton gauche dans l'état **éteint** allumer le robot en vert et l'amène dans l'état **allumé1**, alors que toucher le bouton droite, *dans le même état*, génère une action différente, allumer le robot en rouge, et amène le robot dans un état différent, **allumé2**.

Implémenter des diagrammes d'états avec des paires événement-action

Nous montrons comment *implémenter* le comportement décrit par le diagramme d'état avec des paires événement-action. Implémenter signifie construire un programme qui fera ce que le diagramme d'états décrit. La Figure 8.1 montre le programme. Regardons maintenant les paires événement-action unes à unes.

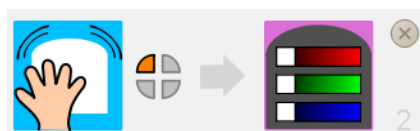




Dans la première paire événement-action, l'événement est composé du bloc détection de choc avec une indication d'état  :



Un état est indiqué par quatre quartiers d'un cercle, chacun pouvant être soit allumé (orange) ou éteint (blanc). Dans ce programme, nous utiliserons le quartier en haut à gauche pour indiquer si la lumière du haut du robot est éteinte ou allumée. Dans cette paire, ce quartier est coloré en blanc, ce qui veut dire que la lumière du robot est éteinte. Ainsi, cette paire veut dire : si le robot est tapé et que le robot est éteint, allumer le robot.

La seconde paire événement-action veut dire : si le robot est tapé et qu'il est allumé, alors l'éteindre :



Si vous regardez à nouveau le diagramme d'états, vous verrez que seulement la moitié du travail est fait. En effet, en allumant et éteignant le robot, nous devons aussi changer son état d'**éteint** à **allumé** ou d'**allumé** à **éteint**. Pour cela nous créons en plus deux paires événement-actions en utilisant le bloc d'action *état* , comme montré sur la Figure 8.1(b). 

Le sens de la première paire est : *quand* le robot est tapé *et* que l'état est *éteint*, alors changer l'état à **allumé** :



De même, le sens de la seconde paire est : *quand* le robot est tapé *et* que l'état est **allumé**, alors changer l'état à **éteint** :





La Figure 8.1 montre le programme complet composé de quatre paires ; nous voyons que chaque événement cause à la fois une action sur la lumière et un changement de l'état du robot. Tant l'action que le changement d'état dépendent de l'état dans lequel le robot se trouve, appelé *état courant*.

Dans combien d'états différents Thymio peut-il être ?

L'état est indiqué par un cercle divisé en quatre quartiers. Quand utilisé avec un événement ou dans le bloc d'action état, chaque quartier peut être :

- **Blanc** : le quartier est *éteint* ;
- **Orange** : le quartier est *allumé* ;
- **Gris** : le quartier n'est pas pris en compte.



Par exemple, dans , les quartiers en haut à gauche et en bas à droite sont allumés, le quartier en haut à droite est éteint, et le quartier en bas à gauche n'est pas pris en compte. Ceci veut dire que si  est associé à un bloc événement, l'événement se produira si l'état est défini soit par :



Comme chacun des quatre quartiers peut être soit allumé soit éteint, il y a $2 \times 2 \times 2 \times 2 = 16$ états possibles :

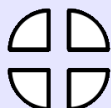
- (éteint, éteint, éteint, éteint)
- (éteint, éteint, éteint, allumé)
- (éteint, éteint, allumé, éteint)
- ...
- (allumé, allumé, allumé, éteint)
- (allumé, allumé, allumé, allumé).

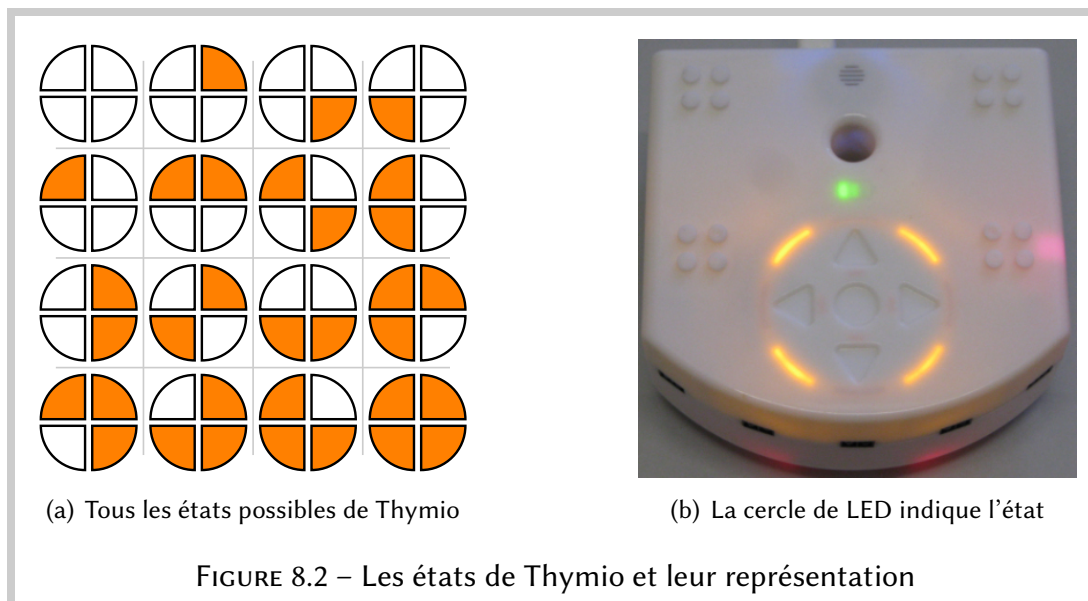
La Figure 8.2(a) énumère graphiquement tous ces états. L'état courant du robot est toujours affiché sur le haut du robot par des arcs de cercles lumineux, par exemple, la Figure 8.2(b) montre le robot dans l'état **(allumé, allumé, allumé, allumé)**.

Information

Lorsqu'un programme démarre, l'état initial est toujours **(éteint, éteint,**

éteint, éteint) :





Truc

Si vous n'utilisez pas tous les 16 états possibles, mais par exemple que 2 ou 4, vous êtes libre de décider quel quartier vous utiliser pour représenter votre état. Aussi, si par exemple vous avez deux choses différentes à encoder dans l'état, et que chacune d'elle a deux valeurs possibles, vous pouvez utiliser deux quartiers indépendamment. C'est pourquoi la capacité d'*ignorer* un quartier est très utile ! Essayez toujours de rester le plus simple possible.

Attraper la souris

Écrivons un programme qui fasse tourner le robot de droite à gauche à la recherche d'une souris (ou d'un autre objet). Si le robot détecte une souris avec son capteur avant-gauche, il continue la recherche jusqu'à ce que la souris soit détectée avec son capteur avant-droite. Puis, il se positionne en face de la souris, comme sur la Figure 8.3(a).

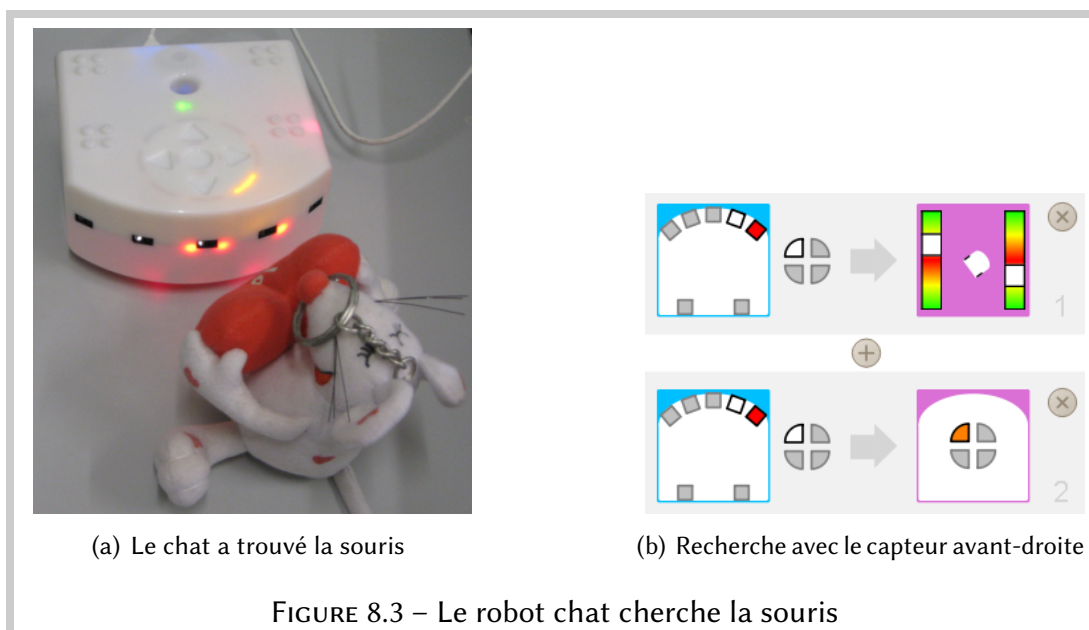
Programme **mouse.aesl**

La paire événement-action suivante fait tourner le robot à gauche :

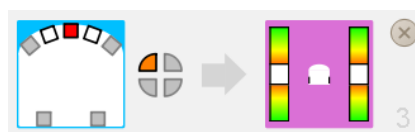


Ceci se produira lorsque le quartier gauche-haut est éteint ; initialement tous les quartiers de l'état sont éteint.

La première paire événement-action dans la Figure 8.3(b) attend que la souris soit détectée par le capteur avant-droite. Notez que le petit carré à côté de ce capteur est



blanc pour que l'événement se produise seulement si le capteur le plus à droite seul détecte la souris. La seconde paire événement-action de la Figure 8.3(b) change l'état. La dernière paire événement-action du programme arrête le robot lorsque la souris est directement en face du capteur avant-centre :




Pourquoi l'événement de cette paire doit-il dépendre de l'état ? La raison est que le capteur central détectera aussi la souris durant le scan initial de droite à gauche. Nous voulons que le robot fasse d'abord un scan complet avant de retourner à la position de la souris ; il est donc nécessaire que cette première détection soit ignorée. Ceci est accompli en arrêtant le scan seulement lorsque l'état est **allumé** et ceci arrive que lorsqu'un scan complet a été effectué.

💡 **Truc**

Il vous faudra expérimenter avec la distance de la souris au robot. Si elle est trop proche du robot, les capteurs à côté du capteur central détecteront aussi la souris, alors que l'événement demande qu'ils ne la détecte *pas*.

📌 **Exercice 8.1**

Écrivez un programme qui fasse danser le robot : il tourne à gauche sur place durant deux secondes, puis tourne à droite sur place durant trois secondes. Ces mouvements se répètent indéfiniment.

 **Exercice 8.2 (Difficile)**

Modifier le programme de suivi de ligne du Chapitre 5 pour que le robot tourne à gauche quand il sort de la ligne par le côté droite, et qu'il tourne à droite quand il sort de la ligne par le côté gauche.

Chapitre 9

Thymio apprend à compter (avancé)

Dans ce chapitre, nous allons montrer comment les états de Thymio peuvent être utilisés pour compter et même pour faire un peu d'arithmétique.

L'implémentation détaillée des projets ne sera pas donnée ici. Nous pensons que vous avez désormais assez d'expérience pour les développer par vous-mêmes. Les codes source des différents programmes de ce chapitre se trouvent dans l'archive, essayez de ne pas les regarder tout de suite mais plutôt de faire un maximum de chemin de votre côté.



Les projets suivants utilisent l'événement  pour changer l'état de Thymio et il est représenté par le cercle de LEDs.

Information importante

L'état du robot est affiché grâce au cercle de LEDs en dessus du robot. La Figure 8.2(b) montre Thymio dans l'état (**allumé, allumé, allumé, allumé**).

Sentez-vous libre de changer n'importe lequel de ces comportements.

Pair et impair

Programme

Choisissez l'un des quartiers du cercle de LEDs. Il sera **éteint** (blanc) si vous tapez dans vos mains un nombre pair de fois et **allumé** (orange) si vous tapez dans vos mains un nombre impair de fois. Presser le bouton central de Thymio le fera retourner au mode pair (puisque zéro est un nombre pair).

Programme **count-to-two.aesl**


La méthode de comptage illustrée ici montre le concept de *modulo 2*. Nous comptons de 0 à 1 puis de nouveau à 0. Le terme *modulo* est proche de *reste* : si vous tapez dans vos mains 7 fois, et que nous divisons 7 par 2, nous obtenons 3 avec un reste de 1. En *modulo 2*, nous ne gardons que le reste de 1.

Dans cette arithmétique, 0 et 1 sont souvent appelés *pair et impair*. Nous appelons aussi ce concept *l'arithmétique cyclique*. Au lieu de compter de 0 à 1, puis de 1 à 2, nous *retournons* à 0 : 0, 1, 0, 1 ...

Ce concept est en fait très familier puisqu'il est utilisé dans les horloges. Les secondes et les minutes sont comptées en *modulo 60* et les heures en *modulo 12 ou 24*. Donc, la seconde après 59 n'est pas 60 mais 0. Dans la même logique, l'heure après 23 n'est pas 24 mais 0. S'il est 23 : 00 et que nous nous mettons d'accord pour nous rencontrer 3 heures après, l'heure du rendez-vous est 26 modulo 24, donc 2 heures du matin.

Compter en unaire

Modifiez le programme pour compter en modulo 4. Il y a quatre états possibles : 0, 1, 2, 3. Choisissez trois LEDs du cercle, elles représenteront les états 1, 2 et 3 ; l'état 0 sera représenté en éteignant toutes les LEDs.

Cette méthode de représentation de nombre est appelée *unaire* car différents éléments d'un état représentent des nombres différents. Nous utilisons souvent cette représentation pour compter des points par exemple :  | représente 6.

Programme **count-to-four.aesl**

Exercice 9.1

Jusqu'à combien pouvons-nous compter sur le Thymio en utilisant la représentation unaire ?

Compter en binaire

Nous sommes familiarisés avec *les représentations en base*, en particulier avec la base 10 (décimal). Les symboles 256 en base 10 ne représentent pas trois objets différents mais bien un seul. Le 6 représente le nombre de 1, le 5 représente le nombre de 10, le 2 représente le nombre de 10×10 (donc de 100). Si nous additionnons ces nombres nous obtenons deux cents cinquante-six. Grâce à la base 10, nous pouvons représenter facilement de grands nombres. De plus, l'arithmétique que nous avons apprise à l'école nous aide à manipuler de grands chiffres.

Nous utilisons la base 10 parce que nous avons 10 doigts, ce qui facilite l'apprentissage de cette base. Les ordinateurs n'ont que deux « doigts » (**éteint** et **allumé**), c'est donc cette base qui est utilisée par les ordinateurs. Au début, la base 2 peut paraître étrange mais il est assez facile de s'y faire. Nous utilisons les mêmes symboles qu'en base 10, le 0 et le 1. Au lieu de passer de 1 à 2, nous retournons à 0.

Voici comment un ordinateur compte jusqu'à dix :

0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000, 1001, 1010



Prenons le nombre en base 2 suivant : 1101. Nous calculons sa valeur de droite à gauche, comme avec la base 10. Le chiffre le plus à droite représente le nombre de 1,

celui d'à côté le nombre de 2, le suivant le nombre de 2×2 (donc de 4) et le dernier (le plus à gauche) le nombre de $2 \times 2 \times 2$ (donc de 8). Comme nous avons le nombre en base 2 : 1101, nous avons $8 + 4 + 0 + 1$, ce qui fait treize, représenté en base 10 par 13.

Programme

Modifiez le programme pour que Thymio compte en modulo 4 en utilisant la représentation binaire (base 2).

Programme **count-to-four-binary.aesl**

Nous n'avons besoin que de deux LEDs du cercle pour représenter les nombres 0 à 3 en base 2. Prenons le quartier en haut à droite pour représenter le nombre de 1 (**éteint** (blanc) pour aucun et **allumé** (orange) pour un) et le quartier d'en haut à gauche pour représenter le nombre de 2. Par exemple,  représente le chiffre 1 et  représente le chiffre 2. Si les deux quartiers sont blanc, l'état représente le chiffre 0 et s'ils sont les deux oranges, le chiffre 3.

Il y a quatre transitions $0 \rightarrow 1, 1 \rightarrow 2, 2 \rightarrow 3, 3 \rightarrow 0$, donc quatre paires événements-action sont nécessaires en addition à une paire événement-action pour remettre le compteur à zéro lorsque l'on appuie sur le bouton central.



Truc

Les deux quartiers du bas ne sont pas utilisés. Ils sont donc laissés en gris et sont ignorés par le programme.



Exercice 9.2

Développez le programme pour qu'il puisse compter en modulo 8. Le quartier en bas à gauche représentera le nombre de 4.



Exercice 9.3

Jusqu'à combien pouvons-nous compter avec Thymio en utilisant la représentation binaire (base 2) ?

Additionner et soustraire

Écrire un programme pour compter jusqu'à 8 est plutôt long puisqu'il faut créer 8 paires événement-action, une pour chaque transition de n à $n + 1$ (modulo 8). Au lieu de procéder de cette façon, nous utilisons des méthodes pour additionner les nombres chiffre à chiffre. En base 10, nous faisons ainsi :

$$\begin{array}{r}
 387 \\
 +426 \\
 \hline
 813
 \end{array}$$

Et en base 2 :

$$\begin{array}{r}
 0011 \\
 +1011 \\
 \hline
 1110
 \end{array}$$

En base 2, lorsque nous ajoutons 1 à 1, au lieu de 2, nous obtenons 10. Le 0 est écrit dans la même colonne que le 1 et à côté, à gauche, le nouveau 1 est reporté (c'est la retenue). L'exemple du dessus montre l'addition de 3 (0011) avec 11 (1011), ce qui donne 14 (1110).

Programme

Écrivez un programme qui commence par représenter 0 et qui ajoute 1 à chaque fois que vous tapez dans vos mains. L'addition doit être faite en modulo 16, donc ajouter 1 à 15 retournera à 0.

Conseils :

- Le quartier d'en bas à droite sera utilisé pour représenter les 8.
- Si le quartier d'en haut à droite (représentant le nombre de 1) est éteint (blanc) et que Thymio entend un « clap », changez-le simplement à allumé (orange) quel que soit l'état des autres quartiers.
- Si le quartier d'en haut à droite (représentant le nombre de 1) est allumé (orange) et que Thymio entend un « clap », changez-le à éteint (blanc) ensuite, reportez le 1 en retenue. Il y aura trois paires événement-action, en fonction de la position du *prochain* quartier montrant 0 (blanc).
- Si tous les quartiers sont allumés (orange), la valeur 15 est représentée. Ajouter un à cet état dépasse le modulo et renvoie la valeur à 0. Il vous faut simplement éteindre toutes les LEDs.

Programme **addition.aesl**

Exercice 9.4

Modifiez le programme pour qu'il commence à 15 et soustraie 1 à chaque « clap ». Lorsqu'il arrivera à 0 et qu'il entendra encore un « clap », il devra retourner à 15.

Exercice 9.5

Coller des petites bandes de ruban adhésif noir (ou dessinez simplement) à intervalles réguliers sur une surface blanche. Écrivez ensuite un programme qui fait que Thymio avance et s'arrête lorsqu'il passe sur la quatrième bande.

Cet exercice n'est pas facile. Les bandes de ruban adhésif doivent être assez longue pour que Thymio les détecte mais pas trop pour ne pas qu'il lance plusieurs événements par bande. Il vous faudra faire des essais avec leurs tailles et avec la vitesse de Thymio.

Chapitre 10

Et après ?

Ce tutoriel vous a présenté le robot Thymio et l'environnement Aseba/VPL. Cet environnement de programmation par image est très pratique et simple d'emploi mais il a tout de même des limitations. Il n'est pas conçu pour programmer des comportements complexes. Pour cela, il vous faudra vous familiariser avec l'environnement d'Aseba Studio, comme illustré sur la Figure 10.1.

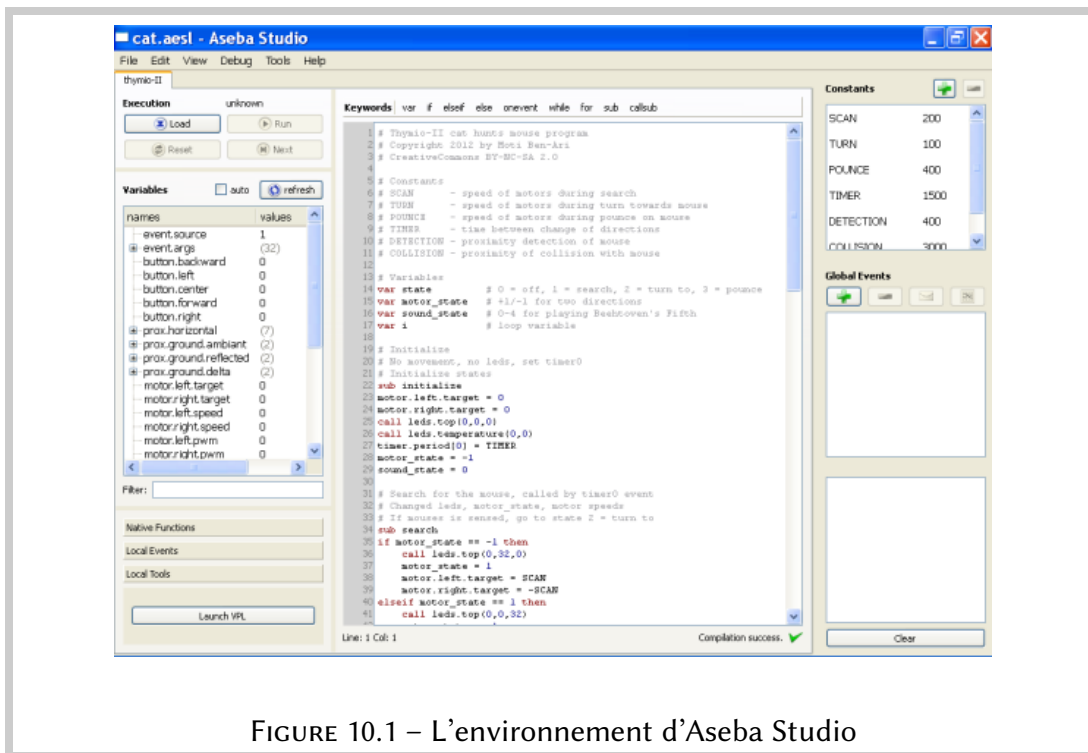


FIGURE 10.1 – L'environnement d'Aseba Studio

La programmation en utilisant Aseba Studio est aussi basée sur les concepts d'événements et d'actions. On retrouve toutes les fonctionnalités que vous avez découvertes en utilisant le VPL. Mais vous avez beaucoup plus de liberté parce que :

- Vous pouvez contrôler précisément quand un événement cause une action, en fonction, par exemple, de la quantité de lumière réfléchiée mesurée par un capteur de sol ou la distance par un capteur horizontal.
- Vous pouvez spécifier qu'une seule action consiste en plusieurs opérations différentes : contrôler les moteurs, changer l'état, définir des seuils, allumer ou éteindre les lumières, etc.
- Vous avez la flexibilité d'un langage de programmation complet avec des variables, des expressions et des structures de contrôle.

Aseba Studio vous donne accès aux fonctionnalités de Thymio qui ne sont pas disponibles dans VPL :

- Vous pouvez contrôler toutes les lumières, pas seulement celles de dessus et de dessous.
- Vous avez plus de flexibilité dans le contrôle du synthétiseur de son.
- Vous avez accès à un capteur de température.
- Vous pouvez voir précisément les valeurs des accéléromètres sur les trois dimensions et plus seulement détecter un choc.
- Vous pouvez utiliser une télécommande pour contrôler Thymio.

Lorsque vous travaillez dans Aseba Studio, vous pouvez ouvrir VPL en cliquant sur le bouton **Lancer VPL** dans l'onglet *Outils* en bas à gauche de la fenêtre. Vous pouvez importer dans Aseba Studio tous les programmes VPL en ouvrant les fichiers correspondants.

Pour utiliser Aseba Studio, commencer depuis la page *Programmation* du site web du Thymio : <https://aseba.wikidot.com/fr:thymioprogram> et suivez les liens correspondants à la *programmation texte*. Si vous manquez d'inspiration, vous trouverez de nombreux nouveaux défis sur la page des exemples : <https://aseba.wikidot.com/fr:thymioexemples>.

★ **Nous vous souhaitons beaucoup de plaisir à apprendre !**
Merci d'avoir lu ce tutoriel :-)