Cette page décrit les possibilités de programmation du Thymio. Elle liste les différentes variables et fonctions, et indique à quels éléments du robot elles font référence. Cette page fait référence à la révision 7 et plus récentes du firmware.
Vous pouvez avoir accès aux anciennes API des versions 3 à 6 et de la version 2 du firmware.

Fold

Table des matières

Bibliothèque standard

Le Thymio inclut la bibliothèque standard de fonctions natives Aseba, documentée sur sa propre page.

Boutons

Thymio possède 5 boutons capacitifs correspondant aux flèches et au bouton central. Un tableau de 5 variables, `buttons.binary`, contient l'état de ces boutons (1 = appuyé, 0 = relâché) :

- `button.backward` : flèche arrière
- `button.left` : flèche gauche
- `button.center` : bouton central
- `button.forward` : flèche avant
- `button.right` : flèche droite

Thymio met à jour ce tableau à une fréquence de 20 Hz, et génère l'événement `button` après chaque mise à jour. En outre, pour chacune de ces touches, quand elle est appuyée ou relâchée, un événement correspondant avec le même nom est émis.

Capteurs de distance

Horizontaux

Thymio possède 7 capteurs de distances autour de lui. Un tableau de 7 variables, `prox.horizontal`, contient les valeurs de ces capteurs :

- `prox.horizontal[0]` : avant gauche
- `prox.horizontal[1]` : avant milieu-gauche
- `prox.horizontal[2]` : avant milieu
- `prox.horizontal[3]` : avant milieu-droite
- `prox.horizontal[4]` : avant droite
- `prox.horizontal[5]` : arrière gauche
- `prox.horizontal[6]` : arrière droite

Les valeurs dans ce tableau varient de 0 (le robot ne voit rien) à plusieurs milliers (le robot est très proche d'un obstacle). Thymio met à jour ce tableau à une fréquence de 10 Hz, et génère l'événement `prox` après chaque mise à jour.

Sol

Thymio possède 2 capteurs de distance au sol. Ces capteurs sont situés à l'avant du robot. Comme un sol noir se comporte comme pas de sol (le noir absorbe la lumière infrarouge), ces capteurs permettent de suivre une ligne au sol. Trois tableaux permettent de lire les valeurs de ces capteurs :

- `prox.ground.ambiant` : intensité lumineuse ambiante au sol, varie entre 0 (pas de lumière) et 1023 (lumière maximale)
- `prox.ground.reflected` : quantité de lumière reçue pendant que le capteur émet de l'infrarouge, varie entre 0 (aucune réflexion) et 1023 (lumière maximale)
- `prox.ground.delta` : différence entre la lumière réfléchie et la lumière ambiante, liée à la distance et à la couleur du sol

Pour chaque tableau, l'index 0 correspond au capteur de gauche et l'index 1 à celui de droite. Tout comme pour les capteurs de distance, Thymio met à jour ce tableau à une fréquence de 10 Hz.

Communication locale

Thymio peut utiliser ses capteurs horizontaux de distance infrarouge pour communiquer une valeur à un autre robot à proximité dans un rayon de 15 cm. La valeur est envoyée à 10 Hz lors du traitement des capteurs de distance. Thymio envoie une valeur de 11 bits (mais un futur firmware pourrait utiliser 1 bits pour utilisation interne, il est donc préférable de se limiter à 10 bits). Pour utiliser la communication, appelez la fonction `prox.comm.enable(state)`, avec 1 comme paramètre `state` pour enclencher et 0 pour désactiver. Si la communication est activée, la valeur dans `prox.comm.tx` est transmise toutes les 100 ms. Lorsque Thymio reçoit une valeur, l'événement `prox.comm` est déclenché et la valeur peut être lue dans la variable `prox.comm.rx`.

Accéléromètre

Thymio possède un accéléromètre 3 axes. Un tableau de 3 variables, `acc`, contient la valeur de l'accélération sur ces axes :

- `acc[0]` : axe x (de droite à gauche du robot, positif si vers gauche)
- `acc[1]` : axe y (d'avant vers l'arrière du robot, positif si vers arrière)
- `acc[2]` : axe z (qui rentre du plan horizontal, positif si vers sol)

Les valeurs dans ce tableau varient de -32 à 32, 1 g correspondant à 23, et donc une unité correspond à 0.45 N. Thymio met à jour ce tableau à une fréquence de 16 Hz, et génère l'événement `acc` après chaque mise à jour. En outre, quand un choc est détecté, un événement `tap` est émis.

Capteur de température

La variable `temperature` contient la température actuelle en dixième de degré Celsius. Thymio met à jour cette valeur à 1 Hz, et génère l'événement `temperature` après chaque mise à jour.

LEDs

Thymio possède de nombreuses LEDs réparties sur tout son corps. La plupart sont associées à des capteurs et permettent de mettre en évidence leur activation: l'allumage des LEDs est lié par défaut aux valeurs des capteurs. Cependant, si elles sont utilisées dans le code, cette utilisation supplante celle par défaut.

Des fonctions natives permettent de contrôler les différentes LEDs. Pour toutes les LEDs, la plage de valeurs d'intensité s'étend de 0 (éteint) à 32 (complètement allumé).

Le cercle de LEDs du dessus

8 LEDs jaunes forment un cercle sur le robot, autour des boutons.

Activation par défaut : reflète les valeurs de l'accéléromètre. Eteintes à l'horizontale, leur intensité augmente en penchant le robot. La LED allumée est toujours celle au point le plus bas.

- `leds.circle(led 1, led 2, led 3, led 4, led 5, led 6, led 7, led 8)` où `led 1` commande l'intensité de la LED avant du robot (numérotation dans le sens des aiguilles d'une montre).

Les LEDs RGB

Il y a deux LEDs RGB sur le robot pilotées ensemble, ce sont celles qui indiquent le comportement du robot. Deux autres LEDs RGB dessous sont pilotables séparément.

Activation par défaut : éteintes dans le mode Aseba.

- `leds.top(red, green, blue)` commande les valeurs de rouge, vert et bleu respectivement, pour les LEDs du dessus.
- `leds.bottom.left(red, green, blue)` commande les valeurs de rouge, vert et bleu respectivement, pour la LEDs du dessous à gauche.
- `leds.bottom.right(red, green, blue)` commande les valeurs de rouge, vert et bleu respectivement, pour la LEDs du dessous à droite.

Les LEDs des capteurs de proximité

Chacun des capteurs de proximité du robot est accompagné d'une LED rouge soudée juste à côté (deux pour la capteur du milieu avant).

Activation par défaut : s'allument en présence d'un objet devant le capteur associé, avec une intensité plus forte plus l'objet est proche.

- `leds.prox.h(led 1, led 2, led 3, led 4, led 5, led 6, led 7, led 8)` commande les LEDs des capteurs avant et arrière. `led 1` à `led 6` correspondent aux LEDs avant, de gauche à droite, et `led 7` et `led 8` aux LEDs des capteurs arrière, gauche et droite.
- `leds.prox.v(led 1, led 2)` commande les LEDs des capteurs dessous, gauche et droite.

Les LEDs des boutons

4 LEDs rouges sont placées entre les boutons sur Thymio.

Activation par défaut : Pour chacun des boutons flèches, une LED s'allume lorsqu'il est appuyé. Pour le bouton du centre, les 4 LEDs s'allument.

- `leds.buttons(led 1, led 2, led 3, led 4)` commande ces LED, `led 1` correspond à la LED avant, et les autres sont numérotées dans le sens des aiguilles d'une montre.

La LED du récepteur télécommande

Cette LED rouge est placée à côté du récepteur télécommande.

Activation par défaut : clignote lorsqu'un code RC5 est reçu.

- `leds.rc(led)` permet de la contrôler.

Les LEDs du capteur de température

Ces deux LEDs (une rouge, une bleue) se trouvent placée à côté du capteur de température.

Activation par défaut : rouge si la température et au dessus de 28°C, rouge et bleu entre 28° et 15°, bleu pour une température en dessous de 15°.

- `leds.temperature(red, blue)` permet de les contrôler.

La LED du micro

Cette LED bleue est placée à côté du microphone.

Activation par défaut : éteinte.

- `leds.sound(led)` permet de la contrôler.

Il y a aussi des LEDs qui ne peuvent pas etre contrôlées par l'utilisateur :

- 3 LEDs vertes dessus indiquent le niveau de batterie,
- une LED bleue et une rouge derrière indiquent l'état de la charge,
- une LED rouge à l'arrière est liée à la carte SD.

Moteurs

Vous pouvez changer la vitesse des roues du Thymio en écrivant dans ces variables :

- `motor.left.target` : vitesse demandée roue gauche
- `motor.right.target` : vitesse demandée roue droite

Vous pouvez relire la vitesse réelle des roues dans ces variables :

- `motor.left.speed` : vitesse réelle roue gauche
- `motor.right.speed` : vitesse réelle roue droite

Les valeurs peuvent aller de -500 à 500. Une valeur de 500 correspond approximativement à une vitesse linéaire de 20 cm/s. Vous pouvez lire la valeur de commande des moteurs dans les variables`motor.left.pwm` et `motor.right.pwm`.

Détection de l'intensité du son

Le Thymio peut détecter lorsque le bruit ambiant est au-dessus d'une intensité donnée et émet un événement. La variable `mic.intensity` montre l'intensité du microphone en cours, tandis que la variable `mic.threshold` contient la limite de l'intensité pour l'événement. Si `mic.intensity` est au-dessus de `mic.threshold`, l'événement `mic` est généré.

Lecture et enregistrement de sons

Vous pouvez jouer des sons de synthèse ou du système. De plus, si vous avez installé une carte micro-SD formatée FAT dans le robot, il est possible d'enregistrer et de jouer vos propres sons. Les fichiers sont stockés

sur la carte micro-SD au format wave, 8-bit non-signé, 8 kHz. Lorsque Thymio fini la lecture d'un son demandée par ASEBA, il déclenche l'événement `sound.finished`. Il ne se déclenche pas un événement si la lecture est interrompue ou si un nouveau son est joué.

Son de synthèse

La fonction native `sound.freq` joue une fréquence, spécifiée en Hz, pour une certaine durée, spécifié dans 1/60 s. Une durée égale à 0 joue le son à l'infini et une durée de -1 arrête le son.

Modification de l'onde primaire

La génération de sons de synthèse fonctionne par ré-échantillonnage d'une onde primaire. Par défaut, l'onde est triangulaire, mais vous pouvez définir votre propre onde à l'aide de la fonction native `sound.wave`. Cette fonction prend en entrée un tableau de 142 échantillons, avec des valeurs de -128 à 127. Ce tableau doit représenter une onde tonique de la fréquence spécifiée dans `sound.freq`. Comme Thymio joue des sons à 7812,5 Hz, ce tableau est joué entièrement à la fréquence de 7812.5/142 = ~ 55 Hz. Jouer un son de fréquences plus élevées fait sauter des échantillons dans le tableau.

Enregistrement

Vous pouvez enregistrer des sons en utilisant la fonction native `sound.record` qui prend en paramètre un numéro d'enregistrement de 0 à 32767. Le fichier est stocké sur la carte micro-SD sous le nom `Rx.wav` où `x` est le paramètre passé à la fonction `sound.record`. Pour arrêter un enregistrement, appelez la fonction `sound.record` avec la valeur -1.

Relecture

Vous pouvez rejouer un son enregistré en utilisant la fonction native `sound.replay`. Cette fonction prend en paramètre un nombre de 0 à 32767 et rejouer le fichier `Rx.wav` de la carte SD où `x` est le paramètre passé à la fonction `sound.replay`. Pour arrêter une lecture, veuillez appeler la fonction `sound.replay` avec une valeur de -1.

Création de son depuis un ordinateur

Vous pouvez créer des sons pour le Thymio II depuis votre ordinateur. La façon plus efficiace de le faire est d'utiliser le logiciel audacity, version 1.3, qui existe pour les différents OS. Voici la démarche pour créer un son compatible avec Thymio II :

- Une fois démarré audacity, changez le *projcct ratc* situé en bas à gauche de 44100 Hz (défaut) à 8000 Hz.
- Enregistrez votre son avec la touche record rouge en haut à gauche. Vous devez voir le curseur avancer et la forme d'onde se visualiser. Stoppez une fois fini avec le bouton stop.
- Votre son doit être en mono (Piste->Piste stéréo vers mono)
- Allez sur le menu *Filc* sous *Export*...
- Donnez un nom de ficher, par exemple `P0.wav` pour le premier ficher à jouer en utilisant la fonction native `sound.play`.
- Choisissez comme *format*: *othcr uncomprcsscd filcs*
- Sous *options* choisissez un header *WAV (Microsoft)* et comme *Encoding* choisissez *Unsigncd 8 bit PCM*.
- assurez que le métadonnées de le ficher ne sont pas valorisé.
- Sauvez ou copiez le ficher sur une carte micro-SD.

Pour faire un test rapide vous pouvez sauver ou copier ce ficher sous le nom de S0.wav. Il sera joué au démarrage du Thymio II.

Ici vous pouvez trouver une vidéo montrant l'exécution de la démarche .

Lecture

Vous pouvez jouer des sons en utilisant la fonction native `sound.play` qui prend en paramètre un numéro d'enregistrement de 0 à 32767. Le fichier doit être stocké sur la carte micro-SD sous le nom `Px.wav` où `x` est le paramètre passé à la fonction `sound.play`. Pour arrêter de jouer un son, appelez la fonction `sound.play` avec la valeur -1.

Son système

Vous pouvez jouer un son système à l'aide de la fonction native `sound.system`, qui prend un nombre de 0 à 32767 en tant que paramètre. Certains sons sont disponibles dans le firmware (voir ci-dessous), mais vous pouvez remplacer ces sons et ajouter de nouveaux en utilisant la carte micro-SD. Dans ce cas, le fichier doit être nommé `Sx.wav` où `x` est le paramètre passé à la fonction `sound.system`. Pour arrêter la lecture d'un son, appelez la fonction `sound.system` avec la valeur `-1`.

Bibliothèque de son système

Les sons suivant sont disponibles :

paramètre	description

`-1`	arrête de jouer un son
`0`	son de démarrage
`1`	son d'arrêt (son non reconfiguration)
`2`	son des boutons fléchés
`3`	son du bouton central
`4`	son de chute libre (peureux)
`5`	son de choc
`6`	son de suivi d'objet
`7`	son de détection d'objet pour suivi

Pour ne plus entendre de son système, vous pouvez décompresser sur une carte micro-SD cette archive qui contient des sons muets.

Télécommande

Thymio possède un détecteur de télécommande infrarouge compatible RC5. Lorsque le Thymio reçoit un code RC5, il génère l'événement `rc5`. Dans ce cas, les variables `rc5.address` et `rc5.command` sont mises à jour.

Timer (minuterie)

Thymio fournit deux timers défini par l'utilisateur. Un tableau de 2 valeurs, `timer.period` , permet de spécifier la période des timers :

- `Timer.period [0]` : période du timer 0 en milliseconde
- `Timer.period [1]` : période du timer 1 en milliseconde

Lorsque le délai expire, le timer génère un événement `timer0` respectivement `timer1` .

Lecture et écriture de données depuis la carte SD

Si une carte SD est présente, Thymio peut lire et écrire des données dans des fichiers. Un seul fichier peut être ouvert à la fois. L'unité de lecture/écriture est une valeur signée de 16 bits. Les fonctions disponibles sont :

- `sd.open(x,status)` : ouvre le `Ux.DAT`. La valeur `x` doit être comprise entre [0:32767], en utilisant -1, on ferme le fichier courant ouvert. Une valeur de 0 est écrite dans la variable `status` si l'opération est réussie, -1 si l'opération a échoué.
- `sd.write(data,written)` : écrit le tableau de données `data` dans le fichier ouvert courant. Le nombre de valeurs écrites est retourné dans le paramètre `written`. Il devrait être égal à la longueur de `data`, sauf si la carte est pleine, le fichier fait plus de 4 Go, ou qu'aucun fichier n'est ouvert.
- `sd.read(data,read)` : lit des données dans le fichier ouvert courant et remplit le tableau `data` avec. Le nombre de valeurs lues est retourné dans le paramètre `read`. Il devrait être égal à la taille du tableau `data`, sauf lorsque la fin du fichier est atteinte ou qu'aucun fichier n'est ouvert.
- `sd.seek(position,status)`: déplace le curseur de lecture et d'écriture dans le fichier ouvert. Le curseur est déplacé à la `position` absolue dans le fichier ouvert. La plage valide est [0:65535]. Il n'est actuellement pas possible de chercher à se positionner après 65535. Une valeur de 0 est écrite dans la variable `status` si l'opération est réussie, -1 si l'opération a échoué.

Vous pouvez décoder les données écrites dans un fichier DAT en utilisant la feuille Excel muni d'une macro. Le format consiste en une simple concaténation de valeurs binaires 16 bits signées.

Remarque : ne retirez pas la carte SD pendant que le robot est allumé. Toujours éteindre le robot avant de retirer la carte SD.

Chargement d'un programme à partir de la carte SD

Thymio peut charger un programme à partir de la carte SD. Quand il démarre, Thymio charge le fichier `vmcode.abo` de la carte SD si il est présent. Le fichier doit se conformer aux AS 001 specification. Plus de détail sur cette page du forum : enregistrer le programme sur carte SD

Tableau des événements locaux

événement	description	fréquence (Hz)	résultat
`button.backward`	la flèche arrière a été appuyée ou	suivant l'action	`button.backward`

	relâchée		
`button.left`	la flèche gauche a été appuyée ou relâchée	suivant l'action	`button.left`
`button.center`	le bouton central a été appuyé ou relâché	suivant l'action	`button.center`
`button.forward`	la flèche avant a été appuyée ou relâchée	suivant l'action	`button.forward`
`button.right`	la flèche droite a été appuyée ou relâchée	suivant l'action	`button.right`
`button`	les boutons ont été testés	20	`buttons.backward`, `buttons.left`, `buttons.center`, `buttons.forward`, `buttons.right`
`prox`	une mesure sur les capteurs de proximité été effectuée	10	`prox.horizontal[0-7]`, `prox.ground.ambiant[0-1]`, `prox.ground.reflected[0-1]` et `prox.ground.delta[0-1]`
`prox.comm`	valeur reçue par infrarouge	à la réception d'une valeur	`prox.comm.rx`
`tap`	un choc a été detecté	suivant les chocs	`acc[0-2]`
`acc`	une mesure sur l'accéléromètre a été effectuée	16	`acc[0-2]`
`mic`	l'intensité sonore ambiant était au dessus du seuil	quand la condition est vraie	`mic.intensity`
`sound.finished`	la lecture d'un son lus par ASEBA fini de par elle-même	lorsque le son se fini	
`temperature`	une mesure de température a été effectuée	1	`temperature`
`rc5`	le capteur de télécommande infra-rouge a reçu un signal	à la reception d'un signal	`rc5.address` et `rc5.command`
`motor`	Le PID moteur est exécuté	100	`motor.left/right.speed`, `motor.left/right.pwm`
`timer0`	lorsque la période du timer0 expire	défini par l'utilisateur	
`timer1`	lorsque la période du timer1 expire	défini par l'utilisateur	