Actionneurs

- Sorties
- Son
- Affichage
- Moteur

Sorties

Le démarrage de chaque bloc

Le **démarrage de chaque bloc** permet d'exécuter un programme si une condition est remplie. Il fonctionne donc comme une distinction de cas, non seulement une fois, mais chaque fois que la condition est remplie, tout au long du programme. Le **démarrage de chaque bloc** :

Est une abréviation de la construction suivante :

On peut insérer dans le **démarrage chaque bloc** de la catégorie Sorties toutes les conditions de cette même catégorie.

Remarque : La section du programme à l'intérieur du démarrage de chaque bloc doit être courte et ne pas comporter d'appels bloqués ou de boucles de fin de session, afin que cette partie du programme puisse être traitée rapidement.

LED

Configurer

Les blocs **Configurer LED ...** et **Configurer luminosité LED ...** , il est possible d'allumer et d'éteindre la LED ou de régler sa luminosité sur une valeur définie (de 0 à 512).

Afficher

Le bloc **Récupérer la luminosité LED** permet de consulter la luminosité d'une LED et de la traiter comme une valeur.

Interroger

Les blocs **LED ...** et **Luminosité de la LED ...** permettent d'utiliser l'activité ou la luminosité d'une LED comme condition. Dans l'exemple, nous réglons la luminosité de la LED à 512, si elle n'a pas déjà cette luminosité.

Moteurs

Le symbole figurant sur les blocs moteurs représente tous les moteurs autres que les codeurs ou les servomoteurs.

Configurer

Le bloc **Définir la vitesse moteur à [] ...** permet de fixer la vitesse d'un moteur à une valeur donnée (de 0 à 512).

Afficher

Le bloc **Récupérer la vitesse du moteur** permet de consulter la vitesse d'un moteur et de la traiter comme une valeur.

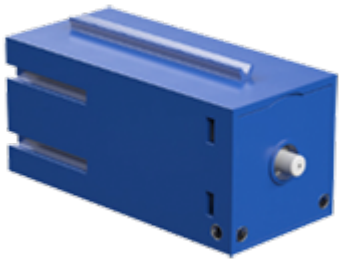
Interroger

Avec les blocs **Fonctionnement du moteur** et **Vitesse du moteur ...** il est possible d'utiliser l'activité ou la vitesse d'un moteur comme condition.

Arrêter

Le bloc **Arrêt moteur …** permet d'arrêter un moteur.

Compresseur



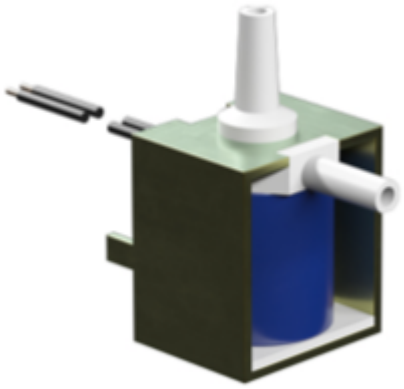
Configurer

Le bloc **Définir compresseur []** permet d'allumer ou d'éteindre le compresseur.

Interroger

Le bloc **Compresseur []** permet d'utiliser l'activité d'un compresseur comme condition.

Électrovanne



Configurer

Le bloc **Définir électrovanne []** permet d’allumer ou d’éteindre l'électrovanne. Ici, « marche » signifie que la valve est ouverte et « arrêt » signifie que la valve est fermée.

Interroger

Le bloc **Electrovanne []** permet d’utiliser l’activité d’une électrovanne comme condition.

Son

Skip to main content Logo ROBO Pro Coding Search Shelves Books Settings Admin Admin Son Formats Source code

Le TXT 4.0 Controller est équipée d’un haut-parleur intégré et permet de jouer des sons.

Le démarrage de chaque bloc

Le **démarrage de chaque bloc** permet d’exécuter un programme si une condition est remplie. Il fonctionne donc comme une distinction de cas, non seulement une fois, mais chaque fois que la condition est remplie, tout au long du programme. Le **démarrage de chaque bloc** :

Est une abréviation de la construction suivante :

On peut insérer dans le **démarrage chaque bloc** de la catégorie Son toutes les conditions de cette même catégorie.

Remarque : La section du programme à l’intérieur du démarrage de chaque bloc doit être courte et ne pas comporter d’appels bloqués ou de boucles de fin de session, afin que cette partie du programme puisse être traitée rapidement.

Lire

Fichiers audio préinstallés

Le bloc suivant permet de lire l’un des 29 sons préinstallés. Le fichier audio souhaité peut être sélectionné dans le menu déroulant (petit triangle). Il est également possible de jouer le son en boucle. Pour cela, il faut cocher la case derrière le symbole de boucle.

Fichiers audios propres

Si vous voulez jouer votre propre son, vous pouvez utiliser le bloc

Pour intégrer son propre son dans le bloc, il faut :

1. Être connecté au contrôleur
2. Saisir l'adresse IP du contrôleur dans le navigateur (il faut sélectionner l'IP qui a également été utilisé pour se connecter au contrôleur)
3. Saisir USER : ft, PASSWORD : fischertechnik sur la page appelée
4. Ouvrir le dossier audio et y charger le fichier audio souhaité sur le contrôleur à l'aide du Plus (important : le fichier audio doit être au format wav)
5. Dans le bloc ROBO Pro Coding, saisir le chemin « ./dateiname.wav »

Ici aussi, il est possible de lire le son en boucle.

Interroger

Pour demander si un fichier audio doit être lu, il faut utiliser le bloc **Restituer le son**. Il peut être utilisé comme condition dans le programme.

Arrêter

Pour arrêter un son, il suffit d'utiliser le bloc **Arrêt Lecture du son** dans le programme.

OkCancel

Affichage

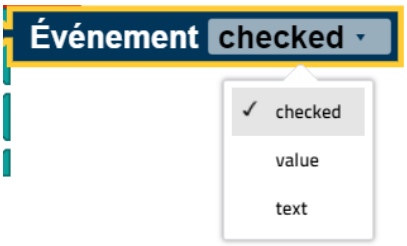
Les blocs de la catégorie Affichage permettent de configurer et d'utiliser l'écran du contrôleur TXT 4.0. Cela se fait en deux étapes :

1. Configurer, c'est-à-dire
   - • Ouvrir un nouveau fichier de la catégorie Affichage, via l'icône de page avec le Plus en haut à gauche
   - Faire glisser les éléments souhaités sur la zone encadrée (elle représente la partie configurable de l'écran)
   - Adapter les spécifications si nécessaire.
2. La programmation, c'est-à-dire
   - Programmer l'effet de l'interaction avec l'écran dans le programme principal avec les blocs de la catégorie Affichage.

Blocs

Demande d'évènement

Le bloc **Evénement []** appelle l'évaluation rétrospective d'un élément. Ce bloc ne peut être utilisé que dans les programmes d'événements. Dans ces programmes d'événements, le bloc fait automatiquement référence à l'événement dans le programme duquel il est utilisé. Le type approprié pour l'évaluation rétrospective peut être sélectionné par le menu déroulant (petit triangle) :



Champ d'étiquetage

L'élément Champ d'étiquetage permet de placer un texte sur l'écran. L'icône du configurateur d'affichage est l'étiquette. Si vous faites glisser cette icône dans la zone tramée, une fenêtre s'ouvre à droite. Ici, sous l'autorité de l'inspecteur, il est possible de définir

- la taille du champ d'étiquette en pixels
- la position en pixels du champ d'étiquette (au point indiqué se trouve le coin supérieur gauche de la zone de texte),
- le nom du champ d'étiquette et
- le contenu du champ d'étiquette (ce texte est représenté au démarrage de l'écran)

Le bloc **Définir le champ d'étiquette Texte ...** permet de modifier le texte affiché pendant le fonctionnement du programme.

Saisies

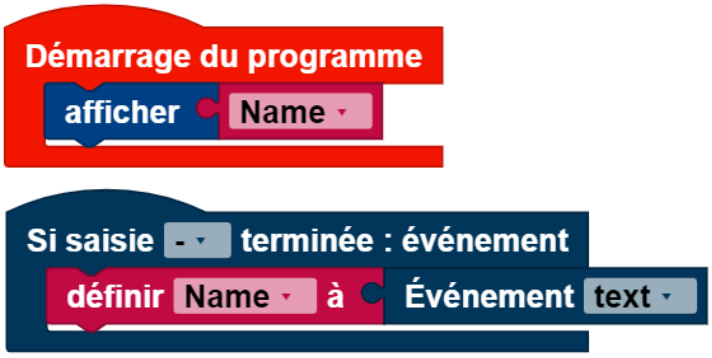
L’élément **Saisie** permet aux utilisateurs/utilisatrices de saisir du texte via le contrôleur. Le symbole associé dans le configurateur d’affichage est le caractère « T ». Si vous faites glisser cette icône dans la zone tramée, une fenêtre s’ouvre à droite. Ici, sous l’autorité de l’inspecteur, il est possible de définir

- la taille du champ d’étiquette en pixels,
- la position en pixels du champ de saisie (au point indiqué se trouve le coin supérieur gauche du champ de saisie),
- le nom du champ de saisie et
- le contenu du champ de saisie (ce texte est représenté au démarrage de l’écran)

Le bloc **Définir le champ de saisie Texte ...** permet de modifier le texte affiché pendant le fonctionnement du programme.

Programme de saisie

Le programme de saisie s'exécute lorsqu’une saisie est terminée. Il est rédigé séparément du programme principal. Les variables fonctionnent globalement sur les deux programmes. Le programme de saisie s'exécute dans le bloc **lorsqu’une saisie est terminée**. Le bloc **Evénement []** est défini sur « texte » dans le programme de saisie. Dans cet exemple, le **nom** de la variable est placé sur le texte saisi, puis utilisé dans le programme principal pour éditer le texte saisi :



Instrument de mesure

La fonction de l’instrument de mesure peut représenter des valeurs (pas de valeurs inférieures à 1). Le symbole associé dans le configurateur d’écran est l’échelle. Si vous faites glisser cette icône dans la zone tramée, une fenêtre s’ouvre à droite. Ici, sous l’autorité de l’inspecteur, il est possible de définir

- la taille de l’instrument de mesure en pixels,
- la position en pixels de l’instrument de mesure (au point indiqué se trouve le coin supérieur gauche de l’instrument de mesure),
- le nom de l’instrument de mesure,
- l’orientation de l’instrument de mesure
- la plage de valeurs que représente l’instrument de mesure, et
- la valeur de l’instrument de mesure affiché au démarrage de l’écran

Le bloc **Définir l’instrument de mesure sur la valeur ...** , il est possible de définir l’instrument de mesure sur la valeur saisie. Cette valeur doit se situer dans la plage de valeurs prédéfinie. Si la valeur est en dehors de la plage de valeurs, l’une des limites de la plage de valeurs est affichée, selon que la valeur est trop grande ou trop petite.

Affichage de l'état

L’indicateur d’état indique l’activité de quelque chose. Selon le statut, il s'allume (« actif ») ou ne s’allume pas (« inactif »). Le symbole du configurateur d’affichage est une diode lumineuse. Si vous faites glisser cette icône dans la zone tramée, une fenêtre s’ouvre à droite. Ici, sous l’autorité de l’inspecteur, il est possible de définir

- la taille de l'affichage de l’état en pixels,
- la position en pixels de l’affichage de l’état (au point indiqué se trouve le coin supérieur gauche de l’affichage de l'état),
- le nom de l’affichage de l’état,
- la couleur de l'affichage de l’état et
- si l’affichage de l’état doit être actif ou inactif au démarrage

Le bloc **Activer l’affichage de l’état []** permet d’activer ou de désactiver l’affichage de l’état. Dans le menu déroulant (petit triangle), il est possible de choisir si l’affichage de l’état doit être activé ou désactivé.

Curseur

Le curseur renvoie les valeurs en fonction de sa position. La position peut être modifiée par l’utilisateur/l’utilisatrice à l’aide de l’écran tactile. La valeur peut être extraite du bloc **Evénement []** dès que le curseur est en pause. La valeur extraite est un nombre décimal. Pour avoir la valeur du curseur en nombre entier, il faut utiliser le bloc **rond**. Le symbole associé au curseur est le trait avec le cercle. Si vous faites glisser cette icône dans la zone tramée, une fenêtre s’ouvre à droite. Ici, sous l’autorité de l’inspecteur, il est possible de définir

- la taille du curseur en pixels,
- la position en pixels du curseur (au point indiqué se trouve le coin supérieur gauche du curseur)
- le nom du curseur,
- l'activité du curseur,
- l’orientation du curseur,
- la plage de valeurs couverte par le curseur et
- la valeur sur laquelle se trouve le régulateur au démarrage de l’écran

Le bloc **Définir la valeur du curseur ...** le curseur peut être déplacé sur une autre valeur.

La fonction **Activer le curseur []** kann permet de modifier l'activité via le menu déroulant (petit triangle).

Programme du curseur

Le programme du curseur s'exécute une fois le curseur déplacé. Il est rédigé séparément du programme principal. Les variables fonctionnent globalement sur les deux programmes. Le programme du curseur s'exécute dans le bloc **lorsqu’une saisie est terminée**. Le bloc **Evénement []** est défini sur « valeur » dans le programme du curseur. Dans cet exemple, la vitesse du moteur est contrôlée via le curseur. La valeur du curseur doit être arrondie car le moteur n’accepte que des nombres entiers comme régime :

Bouton

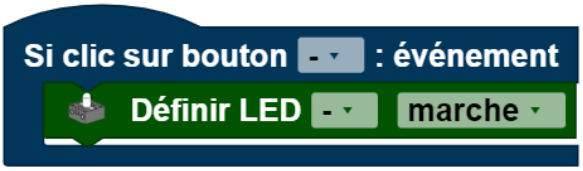
Le bouton est un champ marqué qui peut être enfoncé. Si vous appuyez sur le bouton, le programme de bouton s'arrête dès qu'il est relâché. L'icône associée au bouton est le carré marqué « OK ». Si vous faites glisser cette icône dans la zone tramée, une fenêtre s'ouvre à droite. Ici, sous l'autorité de l'inspecteur, il est possible de définir

- la taille du bouton en pixels,
- la position en pixels du bouton (au point indiqué se trouve le coin supérieur gauche du bouton),
- le nom du bouton,
- le texte affiché sur le bouton et
- l'activité du bouton

Le bloc **Activer le bouton []** permet de modifier l'activité via le menu déroulant (petit triangle).

Programme de boutons

Le programme de boutons s'exécute dès que le bouton n'est plus enfoncé. Il est rédigé séparément du programme principal. Les variables fonctionnent globalement sur les deux programmes. Le programme de bouton s'exécute dans le bloc **lorsque le bouton est enfoncé**. Le bloc **Evénement []** ne peut pas être utilisé dans le programme de boutons car le bouton n'a pas de valeur de retour. Dans cet exemple, la LED est activée une fois le bouton enfoncé.



Interrupteur

L'interrupteur peut adopter deux positions et se trouve toujours exactement dans l'une de ces deux positions. En fonction de la position, il retourne un signal **correct** ou **incorrect** . Le symbole associé à l'interrupteur est l'ovale avec le point. Si vous faites glisser cette icône dans la zone tramée, une fenêtre s'ouvre à droite. Ici, sous l'autorité de l'inspecteur, il est possible de modifier

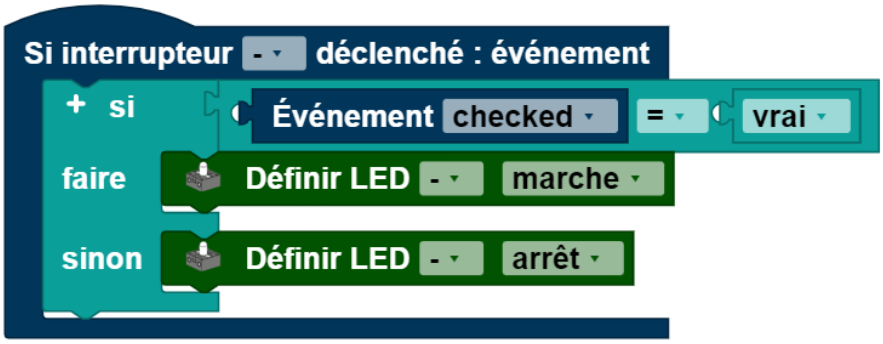
- la taille de l'interrupteur en pixels,
- la position en pixels de l'interrupteur (au point indiqué se trouve le coin supérieur gauche de l'interrupteur),
- le nom de l'interrupteur,
- le texte à côté duquel se trouve l'interrupteur,
- l'activité de l'interrupteur et
- l'état dans lequel l'interrupteur doit se trouver au démarrage du programme

Le bloc

a deux fonctions. Vous pouvez définir l'activité (sélectionner enabled dans le menu déroulant) ou l'état (sélectionner checked dans le menu déroulant) sur **vrai** ou **faux**.

Programme d'interrupteur

Le programme d'interrupteur s'exécute à chaque fois que l'interrupteur est déplacé. Il est rédigé séparément du programme principal. Les variables fonctionnent globalement sur les deux programmes. Le programme de l'interrupteur s'exécute dans le bloc **lorsque l'interrupteur est commuté**. Le bloc **EvénemENT []**est défini sur « checked » dans le programme d'interrupteur, il retourne un signal **vrai** ou **faux**. Cet exemple de programme allume la LED lorsque l'interrupteur est commuté, sinon la LED est éteinte :

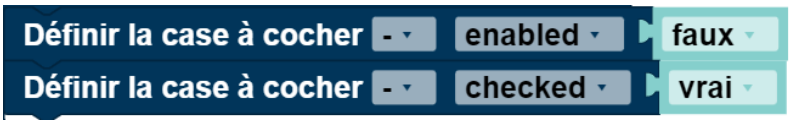


Case à cocher

La case à cocher peut adopter deux positions et se trouve toujours exactement dans l'une de ces deux positions. En fonction de la position, elle retourne un signal **vrai** ou **faux** . Le symbole de la case à cocher est le carré avec la coche. Si vous faites glisser cette icône dans la zone tramée, une fenêtre s'ouvre à droite. Ici, sous l'autorité de l'inspecteur, il est possible de définir

- la taille de la case à cocher en pixels,
- la position en pixels de la case à cocher (au point indiqué se trouve le coin supérieur gauche de la case à cocher),
- le nom de la case à cocher,
- le texte à côté duquel se trouve la case à cocher,
- l'activité de la case à cocher et
- l'état dans lequel la case à cocher doit se trouver au démarrage du programme

Le bloc suivant a deux fonctions. Le menu déroulant (petit triangle) permet d'indiquer celui qui est utilisé. Vous pouvez définir l'activité (sélectionner enabled dans le menu déroulant) ou l'état (sélectionner checked dans le menu déroulant) sur **vrai** ou **faux**.



Programme de case à cocher

Le programme de case à cocher s'exécute à chaque fois que la case à cocher est actionnée. Il est rédigé séparément du programme principal. Les variables fonctionnent globalement sur les deux programmes. Le programme de la case à cocher s'exécute dans le bloc **lorsque la case à cocher est commutée**. Le bloc **E vénemENT []**est défini sur « checked » dans le programme d'interrupteur, il retourne un signal **vrai** ou **faux**. Cet exemple de programme allume la LED lorsque la case est cochée, sinon la LED est éteinte.


Si case à cocher - actionnée : événement
si
Événement checked = vrai
faire
Définir LED - marche
sinon
Définir LED - arrêt

Moteur

Le démarrage de chaque bloc

Le **démarrage de chaque bloc** permet d'exécuter un programme si une condition est remplie. Il fonctionne donc comme une distinction de cas, non seulement une fois, mais chaque fois que la condition est remplie, tout au long du programme. Le **démarrage de chaque bloc** :

Est une abréviation de la construction suivante :

On peut insérer dans le **démarrage chaque bloc** de la catégorie Moteur toutes les conditions de cette même catégorie.

Remarque : La section du programme à l'intérieur du démarrage de chaque bloc doit être courte et ne pas comporter d'appels bloqués ou de boucles de fin de session, afin que cette partie du programme puisse être traitée rapidement.

Moteur

Le symbole figurant sur les blocs moteurs représente tous les moteurs autres que les codeurs ou les servomoteurs.

Configurer

Le bloc **Définir la vitesse moteur à [] ...** permet de fixer la vitesse d'un moteur à une valeur donnée (de 0 à 512). Le menu déroulant (petit triangle) permet de choisir le sens de rotation.

Afficher

Le bloc **Récupérer la vitesse du moteur** permet de consulter la vitesse d'un moteur et de la traiter comme une valeur.

Interroger

Avec les blocs **Fonctionnement du moteur** et **Vitesse du moteur ...** il est possible d'utiliser l'activité ou la vitesse d'un moteur comme condition.

Arrêter

Le bloc **Arrêt moteur …** permet d’arrêter un moteur. Le bloc **Arrêt moteur []** permet d’arrêter un moteur directement ou à échéance. L’option souhaitée peut être sélectionnée dans le menu déroulant (petit triangle).

Servomoteur

Configurer

Le bloc **Définir la position sur ...** permet de définir la position d'un servomoteur sur une valeur spécifique (de 0 à 512). 0 et 512 sont les valeurs de déviation maximale à droite et à gauche. Pour la valeur 256, le servomoteur est placé au centre.

Afficher

Le bloc **Récupérer la position**permet de consulter la position d'un servomoteur et de la traiter comme une valeur.

Moteur encodeur

Le moteur encodeur a les mêmes fonctions qu’un moteur normal, mais il permet en outre de compter les rotations et de faire fonctionner plusieurs moteurs de façon synchronisée. Un tour est divisé en ~64 pas.

Configurer

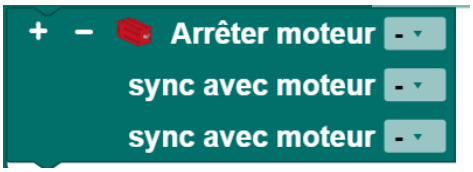
Avec le bloc

il est possible de définir la vitesse d'un moteur sur une valeur spécifique (de 0 à 512). Le menu déroulant (petit triangle) permet de choisir le sens de rotation. On peut également saisir le nombre de pas que le moteur doit parcourir. Dans cet exemple, le moteur tourne à 100 pas, soit un et un tiers de tour. Comme on peut le voir par exemple, ce bloc a un signe plus permettant de faire fonctionner plusieurs moteurs de façon synchronisée. Il est possible de synchroniser les moteurs sur le maître ou sur une extension, une synchronisation croisée par exemple entre les moteurs du maître et une extension est impossible.

Remarque : Des appels de synchronisation se succédant rapidement, tels qu'ils sont possibles par une boucle (voir exemple), peuvent nuire à la synchronisation, voire l'empêcher complètement.

Arrêter

Le bloc **Arrêt moteur …** permet d'arrêter un moteur. Si vous voulez arrêter plusieurs moteurs en même temps, vous pouvez ajouter jusqu'à trois moteurs en plus à gauche du bloc.



Interroger

Le bloc **a atteint sa position** et est utilisé comme une condition pour atteindre sa position. Par position, on entend ici la position finale d'un moteur encodeur à pas complet.