

MISE EN ROUTE

La carte Arduino est prête (*programmée avec FirmataPlus*), il ne vous reste plus qu'à exécuter le script « s2a_fr.bat ».

Écrire en majuscules COMx

...que l'on retrouve ici

```
C:\Windows\system32\cmd.exe

s2a : la communication Arduino+Scratch2 facile !

Vous devez juste indiquer le numero de port de la carte Arduino :
dans le gestionnaire de peripheriques, quel port COM lui est attribue ?
Si cette fenetre se ferme, c'est qu'il ne s'agit pas du bon port...

IMPORTANT
ce script doit etre execute depuis le dossier ou se trouve le fichier s2a_fm.py
Vous pouvez par contre en creer un raccourci pour etre execute depuis le bureau.

Le plus simple pour demarrer est d'ensuite demarrer Scratch2 quand il vous le
propose puis ouvrir le fichier "fichier_vide_FR.sb2" (attention extension sb2).

Indiquer le port COM sur lequel est connecte votre interface Arduino (COM1, COM2
, etc - voir le Gestionnaire de peripheriques) :COM3
s2a_fm version 1.5 Copyright(C) 2013-14 Alan Yorinks All Rights Reserved
PyMata version 1.58 Copyright(C) 2013-14 Alan Yorinks All rights reserved.

Opening Arduino Serial port COM3
Please wait while Arduino is being detected. This can take up to 30 seconds ...
Board initialized in 0 seconds
Total Number of Pins Detected = 20
Total Number of Analog Pins Detected = 6
Please wait for total Arduino Pin Discovery to complete. This can take up to 30
additional seconds.
Arduino Total Pin Discovery completed in 0 seconds
Starting HTTP Server!
Use <Ctrl-C> to exit the extension
Please start Scratch or Snap!
```

NE PAS FERMER CETTE FENETRE !

Si cette question répétitive vous gêne, il vous suffit d'éditer avec le bloc-notes le fichier « s2a_fr.bat » pour en changer les dernières lignes en transformant les commandes en commentaires (REM arqué) :

```
echo.
rem ancienne ligne de commande pour l'auto-det
rem for /f "tokens=4 delims=: " %%A in ('mode
set /p port=Indiquer le port COM sur lequel est connecte vo
.\Python\python s2a_fm.py %port%
```

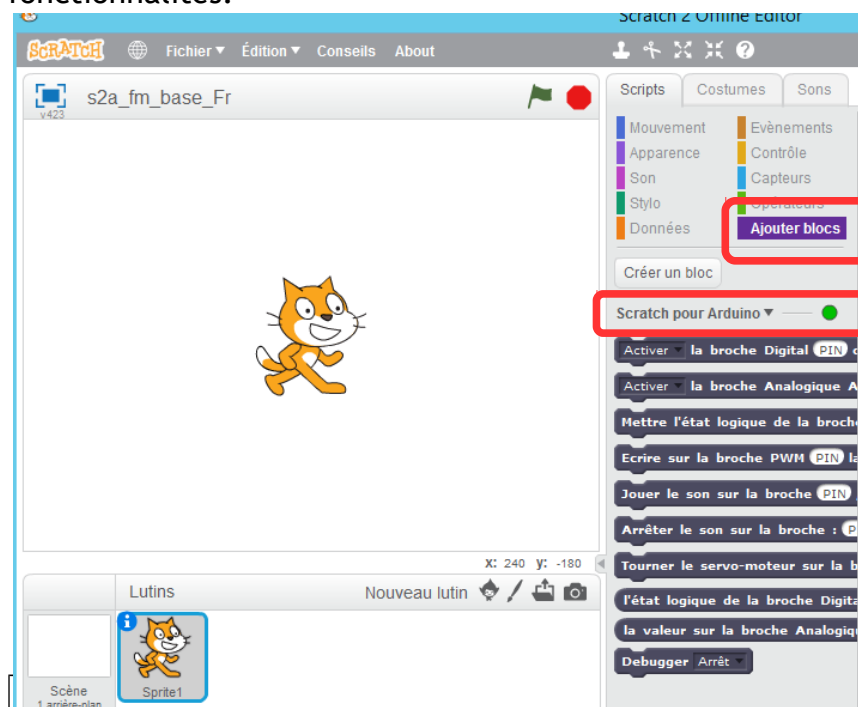
Si vous voulez utiliser une auto-détection de carte

Si vous avez toujours la même carte Arduino sur l'ordinateur (et donc sur le même port COM)

```
echo.
rem ancienne ligne de commande pour l'
for /f "tokens=4 delims=: " %%A in ('m
rem set /p port=Indiquer le port COM s
rem .\Python\python s2a_fm.py %port%
```

```
rem ancienne ligne de commande p
rem for /f "tokens=4 delims=: "
rem set /p port=Indiquer le port
.\Python\python s2a_fm.py COM3
```

Ça y est, Scratch2 va pouvoir envoyer des commandes qui seront transférées par le biais de s2a, il reste donc plus qu'à lancer Scratch2 (*hors-ligne ou en-ligne*) pour découvrir de nouvelles fonctionnalités.



Le plus simple est d'ouvrir le fichier « fichier_vide_FR.sb2 » qui contient déjà les nouveaux blocs :

Tant que l'interpréteur s2a fonctionne correctement entre Scratch2 et Arduino, le point reste vert.

Suite à des problèmes de compatibilité dépendant des versions de Scratch2, plus aucun accent n'est utilisé dans les variables.

Le principe est d'activer les broches sur lesquelles vous avez connecté du matériel, et de les déclarer du bon type (Analogique ou Digital - *numérique en français*) :



Vous pouvez aussi les désactiver, voire ensuite en *changer le type*.

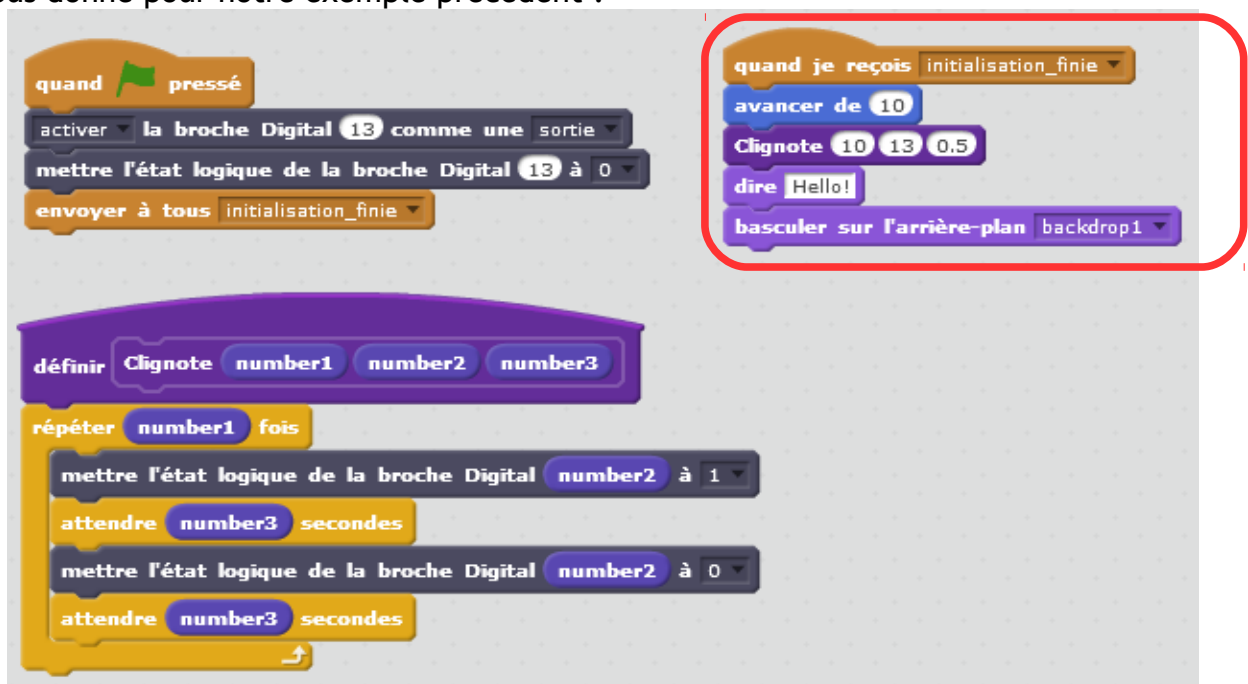
Vous pouvez alors mixer les interactions entre le virtuel de l'écran et le matériel connecté :



Suivant le niveau de l'élève, il est possible de définir des blocs, des fonctions déjà prêtes pour l'élève, à l'instar des macro-étapes :



Ce qui nous donne pour notre exemple précédent :





DÉTAIL DE TOUS LES
BLOCS DISPONIBLES -
SCRATCH2
SNAP !




Nom : Prénom :

2-utilisation.odt

POUR APPROFONDIR

Tout d'abord, il faut se souvenir qu'un programme (ou *script*) est lié au lutin (ou *sprite*) : on peut faire des programmes pour chaque lutin.

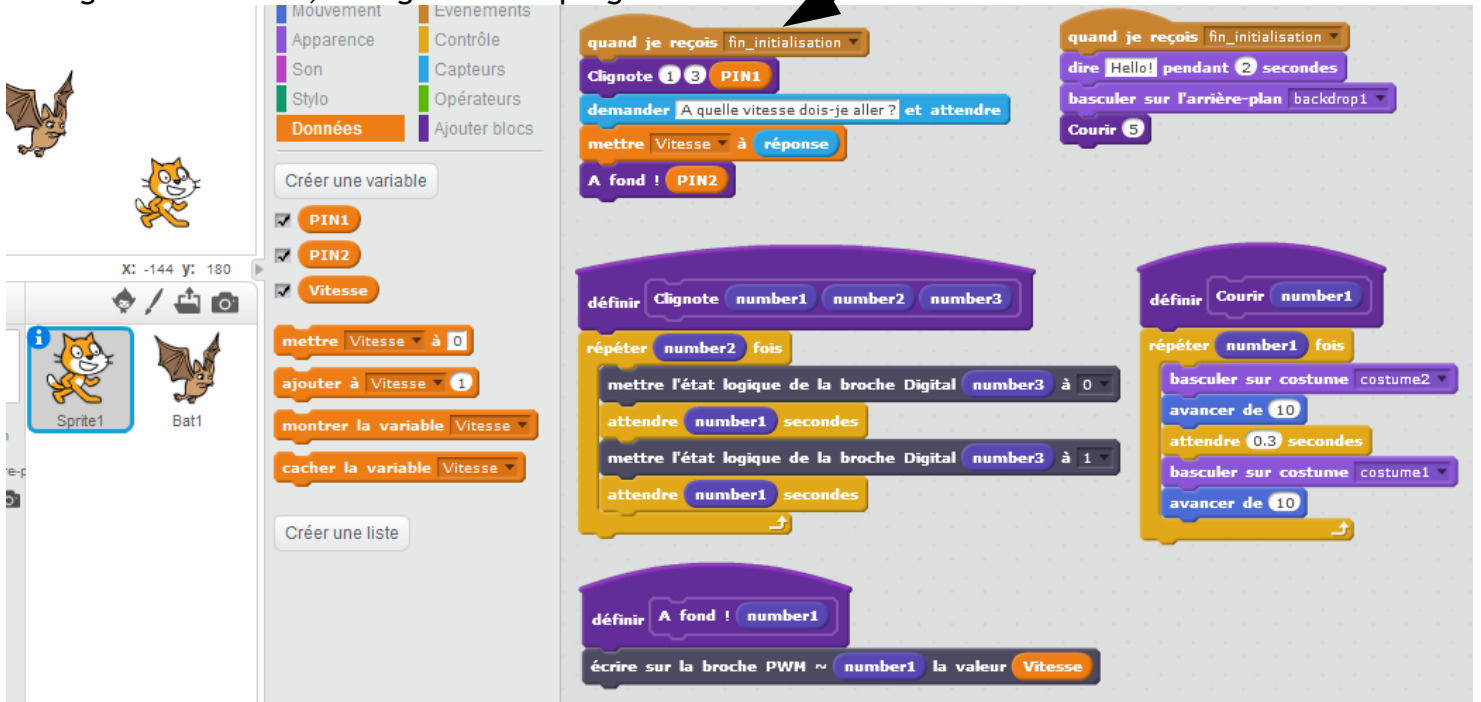


Il n'y a qu'une 'réponse', il faut donc stocker à chaque question.

Prendre un peu de temps entre l'initialisation et l'utilisation.

Ce message sera propagé à tous les sprites/lutins.

Changement de lutin, changement de programme :



A l'instar d'un vrai programma en C ou Java ou autre, j'initialise mon programme et je fais appel à des **variables globales** qui vont contenir des données ré-exploitées d'un programme à l'autre (PIN1, PIN2 & Vitesse) :

- je pose une question et je stocke (→ *PIN1*) la réponse pour que le résultat (→ *réponse*) ne soit pas changé par la prochaine question, puis je lance ma macro 'Clignote (1s)(3 fois)(le PIN1)' ;
- la chauve-souris demande sur quel pin est branché le moteur (→ *PIN2*) mais c'est le chat qui demande la vitesse (→ *vitesse*), puis je lance la macro 'A fond !' qui exploite les 2 questions posées par les lutins.

En même temps (en **parallèle**) que la diode 'Clignote', *macro qui dure donc 6s*, le chat va dire Hello pendant 2s puis courir pendant 1,5s. Donc il sera arrêté quand il posera sa question.