



mBlock



MIT App Inventor

Sources : <http://webtoolsreview.blogspot.fr/2016/04/programming-mbot-with-app-inventor-2.html>
<http://learn.makeblock.com/makeblock-orion-protocol/>



I. PRESENTATION

Le **Robot Éducatif programmable mBot-Version Bluetooth** est un robot en kit économique et facile à utiliser, pour permettre aux jeunes élèves d'acquérir de l'expérience pratique en programmation graphique, en robotique, et en électronique. C'est une **solution tout-en-un pour l'apprentissage de la robotique en technologie !**

Ce robot est basé sur 3 caractéristiques principales :

- L'électronique est basée sur la plate-forme Arduino en sources libres.
- Deux outils de programmation : **mBlock** (un outil de programmation graphique par glisser-déposer basé sur Scratch) et **Arduino IDE**.
- Parvenez à réaliser différents projets tels que l'évitement de mur, le suivi de ligne, en utilisant des capteurs pour jouer avec des jeux en mBlock.



Le Mbot possède aussi un protocole de communication ORION qui permet de communiquer entre une interface et la carte du robot mbot.

C'est cette communication qui va nous intéresser. Certes, elle est un peu complexe mais elle va nous permettre de communiquer sans charger un programme dans le Mbot puisqu'il existe déjà. Pour communiquer, il faut juste appairer votre robot à une tablette ou smartphone.

Le protocole ORION est basé sur un codage de plusieurs octets qui vont envoyer des instructions au robot pour qu'il puisse soit activer des commandes, soit envoyer les valeurs des capteurs.



Pour communiquer il faut appairer le robot avec le téléphone ou la tablette

Envoyer des octets



ATTENTION

il faut impérativement installer les drivers, ou avoir à disposition les drivers d'installation de la carte avant de brancher sa première carte

II. PILOTAGE DES MOTEURS

En s'aidant du site makeblock.learn, on peut retrouver les instructions envoyés au robot en Bluetooth.

	begin_code_1	begin_code_2	length	index	action	device	port	speed_low	speed_high
Code hexa	ff	55	06	00	02	0a	09	ff	00
Code décimal	255	85	06	0	2	10	9	255	0

speed_low = speed & 0xff;
 speed_high = (speed >> 8) & 0xff;
 ("speed" is a number set by your own, ranging from -255 to 255.)

Exemple :

Faire fonctionner le moteur gauche avec une vitesse de 255

- ff 55 06 60 02 0a 09 ff 00

Faire fonctionner le moteur droit avec une vitesse de 255

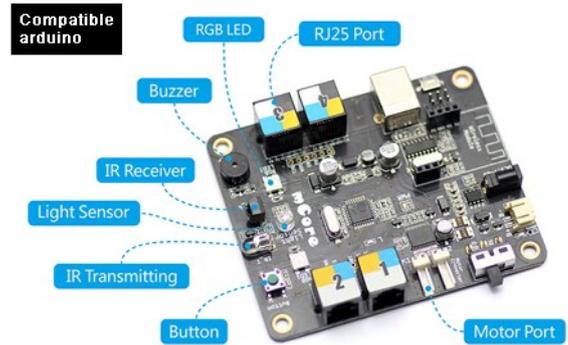
- ff 55 06 60 02 0a 0a ff 00

Avec le même principe, on peut faire fonctionner les deux Led RGB du robot mBot.

R pour RED=ROUGE (0=noir, 255=rouge)

G pour GREEN=VERT (0=noir, 255=vert)

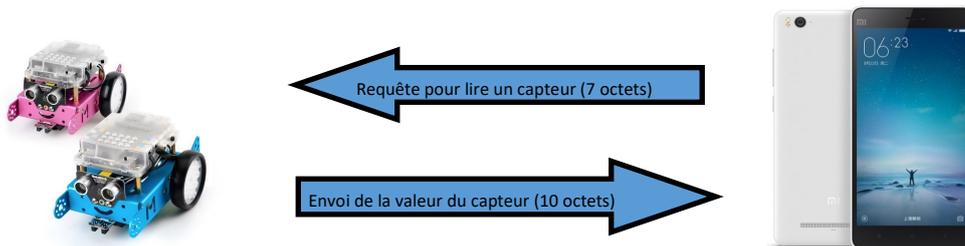
B pour BLUE=BLEU (0=noir, 255=bleu)



Si les trois valeurs sont à 255, la LED éclaire BLANC. Si les 3 valeurs sont à 0, la LED éclaire très faiblement.

III. RECEVOIR LES INFORMATIONS DES CAPTEURS

- ◆ Afin de recevoir les informations d'un capteur sur le Mbot, on doit envoyer une requête au robot mBot pour qu'il puisse renvoyer les informations de la valeur du capteur.



- ◆ Comment lire la valeurs du capteur à ultra-son ?

Il faut envoyer une requête au robot pour lui demander qu'il renvoie la valeur Requête à Envoyer

Requête à envoyer

	begin_code_1	begin_code_2	length	index	action	device	port
Code hexa	ff	55	04	02	01	01	03
Code décimal	255	85	4	2	1	1	1



Recevoir les informations du capteur

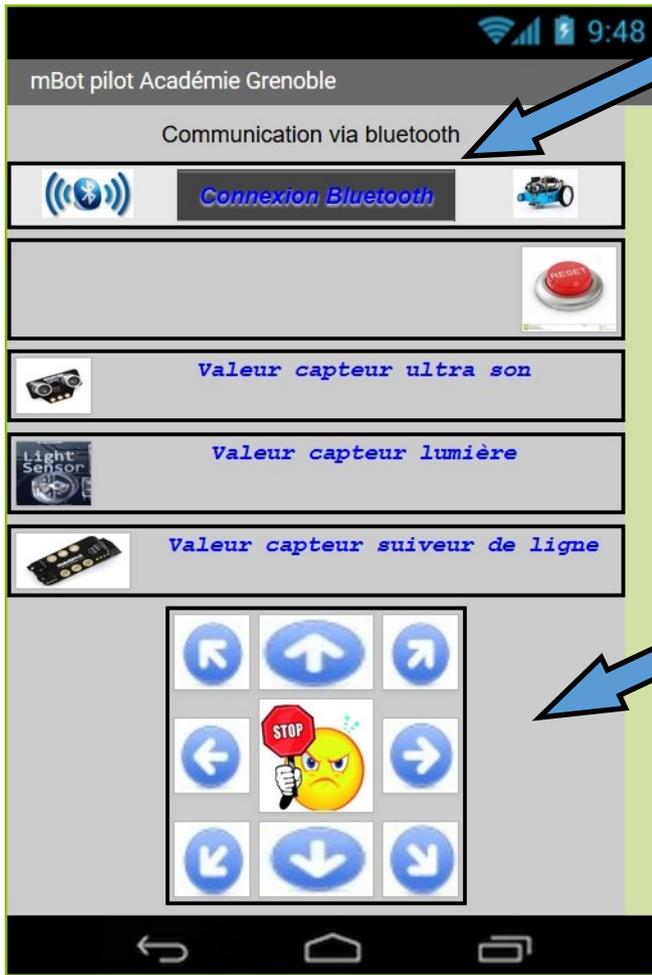
	Octet 1	Octet 2	Octet 3	Octet 4	Octet 5	Octet 6	Octet 7	Octet 8	Octet 9	Octet 10
Code hexa	ff	55	02	02	VAL1	VAL2	VAL3	VAL4	0a	0d
Code décimal	255	85	2	2	VAL1	VAL2	VAL3	VAL4	10	13

Le calcul est le suivant :

Résultat = (0xVAL1<<24) + (0xVAL4<<16)+(0xVAL2<<8)+0xVAL3

0x, c'est la valeur hexadécimal, et <<24 correspond à la place de l'octet car c'est une valeur sur 32 bits.

IV. CREATION DE L'APPLICATION sur AppInventor2



Créer un label pour écrire «communication via Bluetooth»

```

quand Screen1 .Initialise
faire
mettre connect bluetooth . Éléments à BluetoothClient1 . Adresses et noms
mettre Pilotage Moteurs . Visible à faux
mettre ultra son . Visible à faux
mettre luminosite . Visible à faux
mettre suivre ligne . Visible à faux
mettre Horloge1 . ChronomètreActivé à faux
    
```

Quand l'application s'ouvre, tous les éléments sont cachés. Cela évite de demander des informations à la carte du mBot sans être connecté au Bluetooth.

Créer une interface avec des boutons pour gérer les déplacements du robot.

```

initialise global octet_dispo à 0
initialise global vitesse_haute à 0
initialise global vitesse_basse à 0
initialise global moteur à 0

à command_moteur
faire
appeler BluetoothClient1 .Envoyer1Octet nombre 255
appeler BluetoothClient1 .Envoyer1Octet nombre 85
appeler BluetoothClient1 .Envoyer1Octet nombre 6
appeler BluetoothClient1 .Envoyer1Octet nombre 0
appeler BluetoothClient1 .Envoyer1Octet nombre 2
appeler BluetoothClient1 .Envoyer1Octet nombre 10
appeler BluetoothClient1 .Envoyer1Octet nombre obtenir global moteur
appeler BluetoothClient1 .Envoyer1Octet nombre obtenir global vitesse_basse
appeler BluetoothClient1 .Envoyer1Octet nombre obtenir global vitesse_haute
    
```

Créer une procédure ou sous-programme pour envoyer les 9 octets qui permettent de commander les moteurs du robot.

Pour cela créer 3 variables :
 - Moteur qui indique quel moteur piloter
 9 c'est le gauche, 10 c'est le droit

- ◆ Vitesse basse pour indiquer la vitesse basse
 - ◆ Vitesse haute pour la vitesse haute
- Ces 2 dernières variables vont permettre de définir le sens de rotation

Pour éviter d'écrire les mêmes programmes pour chaque commande, créer des procédures sur lesquelles on va jouer avec les variables.

```

quand droite .Clic
faire
  appeler STOP
  mettre global moteur à 10
  appeler command_moteur
  appeler Avant_mot_gauche
  appeler command_moteur

quand gauche .Clic
faire
  appeler STOP
  mettre global moteur à 9
  appeler command_moteur
  appeler Avant_mot_droit
  appeler command_moteur

à Avant_mot_gauche
faire
  mettre global moteur à 9
  mettre global vitesse_basse à 0
  mettre global vitesse_haute à 255

à Avant_mot_droit
faire
  mettre global moteur à 10
  mettre global vitesse_basse à 255
  mettre global vitesse_haute à 0

quand stop .Clic
faire
  appeler STOP
  mettre global moteur à 9
  appeler command_moteur
  mettre global moteur à 10
  appeler command_moteur

quand avancer .Clic
faire
  appeler Avant_mot_gauche
  appeler command_moteur
  appeler Avant_mot_droit
  appeler command_moteur

à STOP
faire
  mettre global vitesse_basse à 0
  mettre global vitesse_haute à 0

quand reculer_gauche .Clic
faire
  appeler Recul_mot_droit
  appeler command_moteur
  appeler Recul_mot_gauche
  mettre global vitesse_basse à 125
  appeler command_moteur

quand reculer_droite .Clic
faire
  appeler Recul_mot_droit
  mettre global vitesse_basse à 125
  appeler command_moteur
  appeler Recul_mot_gauche
  appeler command_moteur

à Recul_mot_droit
faire
  mettre global moteur à 10
  mettre global vitesse_basse à 0
  mettre global vitesse_haute à 255
  
```

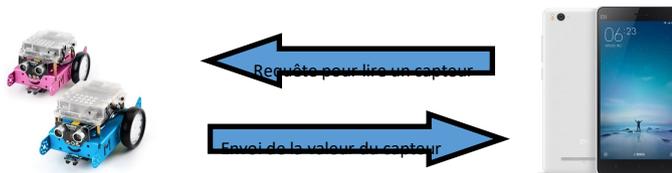
On utilise la fonction chronomètre. On observe toutes les 500ms si on reçoit une donnée sur le port Bluetooth. Si une donnée est reçue on la stocke dans une variable.

```

initialise global octet_recu à ""

quand Horloge1 .Chronomètre
faire
  si
    appeler BluetoothClient1 .Octets disponibles pour le réception ≥ 0
  alors
    mettre global octet_recu à appeler BluetoothClient1 Recevoir texte
    nombre d'octets
    appeler BluetoothClient1 .Octets disponibles pour le réception
  
```

Pour recevoir les informations d'un capteur du Mbot, on doit envoyer une requête au Mbot pour qu'il puisse renvoyer les informations de la valeur du capteur.



Un bouton RESET permet de remettre les variables à 0.



```

quand init .Clic
faire
  mettre global octet à 0
  mettre Val_UltraSon . Texte à 0
  mettre Val_lumiere . Texte à 0
  mettre Val_SuiveurLigne . Texte à 0
  
```

Voici le bouton qui demande au mbot d'envoyer la valeur du capteur ultrason



C'est le protocole à envoyer au mbot pour lui demander la valeur en décimal du capteur ultrason
255
85
4
2
1
1
1

```

faire
  appeler BluetoothClient1 .Envoyer1Octet nombre 255
  appeler BluetoothClient1 .Envoyer1Octet nombre 85
  appeler BluetoothClient1 .Envoyer1Octet nombre 4
  appeler BluetoothClient1 .Envoyer1Octet nombre 2
  appeler BluetoothClient1 .Envoyer1Octet nombre 1
  appeler BluetoothClient1 .Envoyer1Octet nombre 1
  appeler BluetoothClient1 .Envoyer1Octet nombre 3
  
```

J'envoie 7 octets pour demander au mBot qu'il renvoie la valeur du capteur ultrason

```

initialise global octet à 0
initialise global bt_us à ""
initialise global US_1 à 0
initialise global US_2 à 0
initialise global US_3 à 0
initialise global US_4 à 0
initialise global US à 0

à convertir_val_US
faire
  mettre global bt_us à joint obtenir global US_1
  obtenir global US_4
  obtenir global US_2
  obtenir global US_3
  mettre Val_UltraSon .Texte à obtenir global bt_us

quand Ultrason .Clic
faire
  appeler Appel_val_ultrason
  si BluetoothClient1 .Disponible
  alors
    mettre global octet à appeler BluetoothClient1 .Recevoir octet non signé nombre d'octets 4
    mettre global US_1 à appeler BluetoothClient1 .Recevoir octet non signé nombre d'octets 1
    mettre global US_2 à appeler BluetoothClient1 .Recevoir octet non signé nombre d'octets 1
    mettre global US_3 à appeler BluetoothClient1 .Recevoir octet non signé nombre d'octets 1
    mettre global US_4 à appeler BluetoothClient1 .Recevoir octet non signé nombre d'octets 1
    mettre global octet à appeler BluetoothClient1 .Recevoir octet non signé nombre d'octets 1
    appeler convertir_val_US
  
```

Quand j'appuie sur le bouton alors j'appelle la procédure appel_val_ultrason

Si une valeur est disponible sur le port Bluetooth alors

- Mettre les 4 premiers octets dans une variable
- mettre octet (5) dans US1
- mettre octet (6) dans US2
- mettre octet (7) dans US3
- mettre octet (8) dans US4
- Mettre les deux octets dans une variable
- Appeler convertir_val_US

On va afficher les valeurs à l'écran.

Ensuite le calcul est le suivant : $Résultat = (0xVAL1 \ll 24) + (0xVAL4 \ll 16) + (0xVAL2 \ll 8) + 0xVAL3$



C'est le même principe pour le capteur de luminosité mais avec une autre valeur pour le protocole.

grâce à l'envoi de ces 7 octets (protocole)

```

à Appel_val_lumiere
faire
  appeler BluetoothClient1 .Envoyer1Octet nombre 255
  appeler BluetoothClient1 .Envoyer1Octet nombre 85
  appeler BluetoothClient1 .Envoyer1Octet nombre 4
  appeler BluetoothClient1 .Envoyer1Octet nombre 5
  appeler BluetoothClient1 .Envoyer1Octet nombre 1
  appeler BluetoothClient1 .Envoyer1Octet nombre 3
  appeler BluetoothClient1 .Envoyer1Octet nombre 3
  
```

Quand j'appuie sur le bouton lum alors - J'appelle la procédure

```

quand Lumiere .Clic
faire
  appeler Appel_val_lumiere
  si BluetoothClient1 .Disponible
  alors
    mettre global octet à appeler BluetoothClient1 .Recevoir octet non signé nombre d'octets 4
    mettre Val_lumiere .Texte à appeler BluetoothClient1 .Recevoir octet non signé nombre d'octets 4
    mettre global octet à appeler BluetoothClient1 .Recevoir octet non signé nombre d'octets 2
  
```

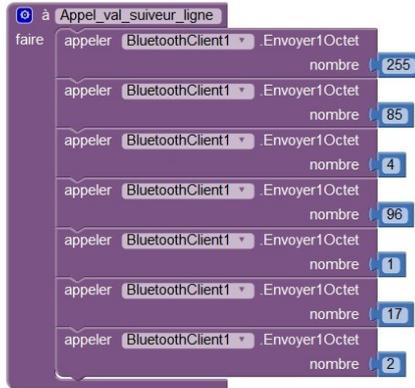
Si une valeur est disponible sur le port Bluetooth alors

- Mettre les 4 premiers octets dans une variable
- Afficher dans la case Val_lum, les 4 octets reçus
- Mettre les deux octets dans une variable

Ensuite le calcul est le suivant : $Résultat = (0xVAL1 \ll 24) + (0xVAL4 \ll 16) + (0xVAL2 \ll 8) + 0xVAL3$



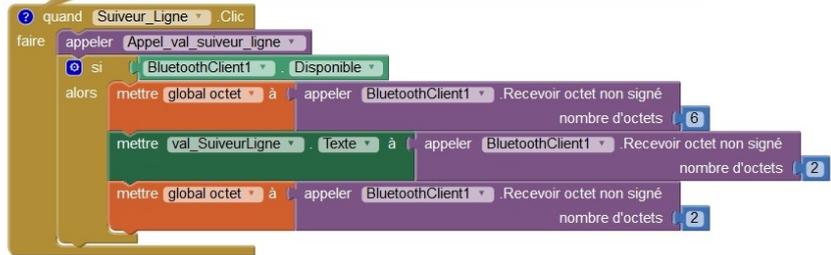
Afficher les valeurs du capteur de suiveur de ligne



Envoyer 7 octets pour lui demander que le robot renvoie la valeur du capteur suiveur de ligne

```
Recevoir les informations du capteur
Le capteur de ligne renvoie 4 informations différentes.
Soit les 2 cellules renvoient une information pour dire que la ligne est blanche
- (64,64)
Soit la cellule droite renvoie une ligne noire et celle de gauche une ligne
blanche (0,64)
Soit la cellule droite renvoie une ligne blanche et celle de gauche une ligne
noire (128,63)
Soit les 2 cellules renvoient une information pour dire que la ligne est noire.
(0,0)

Quand j'appuie sur le bouton suiveur_ligne alors
- J'appelle la procédure appel_val_suiveur_ligne
Si une valeur est disponible sur le port Bluetooth
Alors
- Mettre les 6 premiers octets dans une variable
```



Quand j'appuie sur le bouton suiveur_ligne alors appeler la procédure appel_val_suiveur_ligne

Si une valeur est disponible sur le port Bluetooth alors

- Mettre les 6 premiers octets dans une variable
- Afficher dans la case Val_SL, les 2 octets reçus
- Mettre les deux octets dans une variable

Recevoir les informations du capteur

Le capteur de ligne renvoie 4 informations différentes :

- ◆ Soit les 2 cellules renvoient une information pour dire que la ligne est blanche (64,64)
- ◆ Soit la cellule droite renvoie une ligne noire et celle de gauche une ligne blanche (0,64)
- ◆ Soit la cellule droite renvoie une ligne blanche et celle de gauche une ligne noire (128,63)
- ◆ Soit les 2 cellules renvoient une information pour dire que la ligne est noire. (0,0)