

*UniBot,  
de Arduino... à  
...Boum'Bot*

*(En passant par BrasRobot, Braccio et  
plein d'autres librairies)*

# Table des matières

---

<b>1. Contenu pédagogique pour 1 BoumBot</b>	<b>4</b>
1. Matériel nécessaire	4
2. Logiciels et fichiers nécessaires	4
3. Configuration minimale :	4
<b>2. Installation d'Arduino sous Windows</b>	<b>5</b>
1. Logiciel Arduino pour Windows	5
2. Pilote de la carte Arduino pour Windows	5
<b>3. Installation d'Arduino sous Linux</b>	<b>7</b>
1. Logiciel Arduino pour Linux	7
2. Configuration de l'utilisateur sous linux	7
<b>4. Installation d'UniBot, BoumBot et autres bibliothèques</b>	<b>8</b>
<b>5. Exécution</b>	<b>10</b>
1. Lancer Arduino	10
2. Configurer Arduino pour BoumBot	11
3. Lancer UniBot	13
4. Charger la bibliothèque BoumBot (ou autre)	14
<b>6. Créer vos bibliothèques</b>	<b>16</b>
1. Préambule	16
2. Format du code source	16
3. Hiérarchie de répertoires et fichiers	17
4. Limitations	17
<b>7. Boum'Bot vu de l'intérieur</b>	<b>18</b>
1. La carte Arduino	18
2. Le « shield »	18
3. Les capteurs d'échelle de gris	19
4. Le capteur de proximité	19
5. Les servomoteurs à rotation continue	20
6. Les potentiomètres	20
7. La capsule piézoélectrique (buzzer)	21
<b>8. Aide</b>	<b>22</b>
1. Téléversement sous Windows	22
1. Dans le « gestionnaire de périphériques »	22
2. Droits d'administration	22
3. Dans Arduino	22
1. Téléversement sous Linux	22

2. Compilation.....	23
1. Micro rappel de programmation .....	23
2. Les erreurs fréquentes .....	23

# 1. Contenu pédagogique pour 1 BoumBot

## 1. Matériel nécessaire

- 1 BoumBot
- 1 câble USB
- 1 bloc « 4 piles AA » et « 1 pile 9V » ou un chargeur de smartphone ( $\geq 1A$ )

## 2. Logiciels et fichiers nécessaires

- Arduino  $\geq 1.8.13$  (environ 160Mo)
- UniBot.jar (16Mo)
- Librairie BoumBot (3 Mo)

## 3. Configuration minimale :

- Pentium core i3 ou équivalent
- 4Go de RAM
- 160Mo d'espace disque
- Droits d'administrateur
- Windows 7 et + / Linux

**Total mémoire de stockage <200Mo**

**RAM nécessaire : 1Go disponible pour l'exécution**

## 2. Installation d'Arduino sous Windows

### 1. Logiciel Arduino pour Windows

Téléchargez la dernière version stable d'Arduino depuis le site officiel

<https://www.arduino.cc/en/software>

Et installez-le depuis votre session utilisateur sans changer les options (suivant, suivant, suivant, terminer)

Attention, vous avez besoin des droits d'administrateur pour installer le programme !

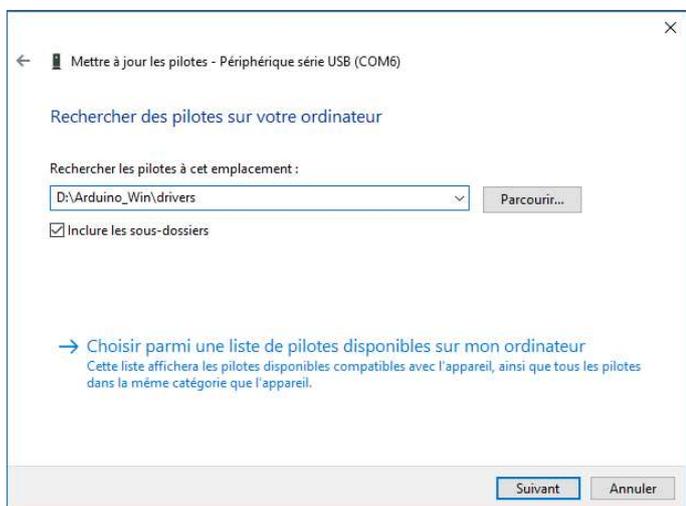
### 2. Pilote de la carte Arduino pour Windows

Lorsque vous branchez le câble USB entre l'ordinateur et le BoumBot pour la première fois, Windows va détecter un nouveau périphérique et vous proposer d'installer le pilote de la carte Arduino.

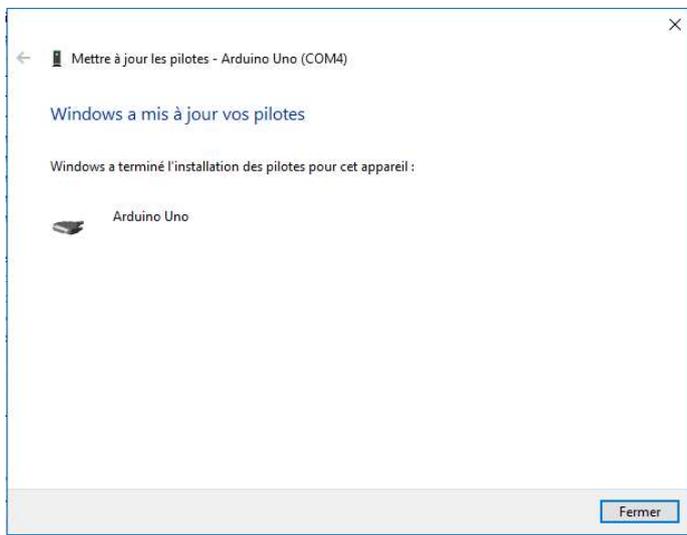
Lancez la recherche automatique :



S'il ne le détecte pas automatiquement, Cliquez sur « parcourir mon ordinateur... »



Puis sur **Parcourir...** et sélectionnez le répertoire « drivers » dans le répertoire « Arduino » situé dans le répertoire d'installation d'Arduino. Par défaut : c:\programFiles(x86) \Arduino



BoumBot est prêt à être programmé via Arduino/UniBot.

## 3. Installation d'Arduino sous Linux

### 1. Logiciel Arduino pour Linux

Téléchargez la dernière version stable d'Arduino depuis le site officiel

<https://www.arduino.cc/en/software>

Décompressez le contenu du fichier.tar.gz dans votre dossier personnel /home/pierrepaul

Ouvrez un terminal et lancer la commande :

```
./install.sh
```

Un raccourci apparaîtra sur le bureau et dans le menu « principal » de Linux sous la catégorie « programmation »

### 2. Configuration de l'utilisateur sous linux

Sous Linux, pas besoin de driver, il faut cependant autoriser l'utilisateur à accéder au port de communication entre le PC et la carte Arduino.

Pour connaître le nom de l'utilisateur, dans le terminal taper :

```
whoami
```

Disons que cette commande vous renvoie pierrepaul

Exécutez alors les deux commandes :

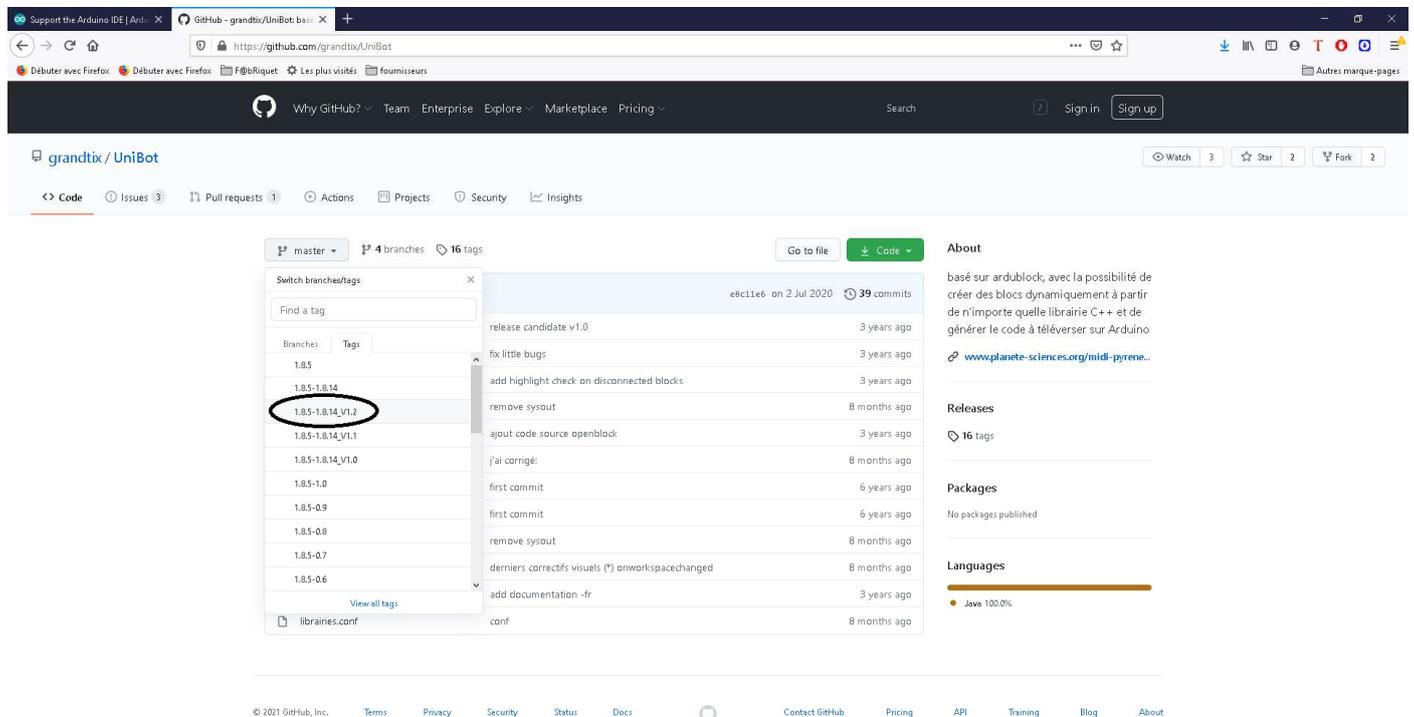
```
sudo usermod -a -G dialout pierrepaul
```

```
sudo usermod -a -G tty pierrepaul
```

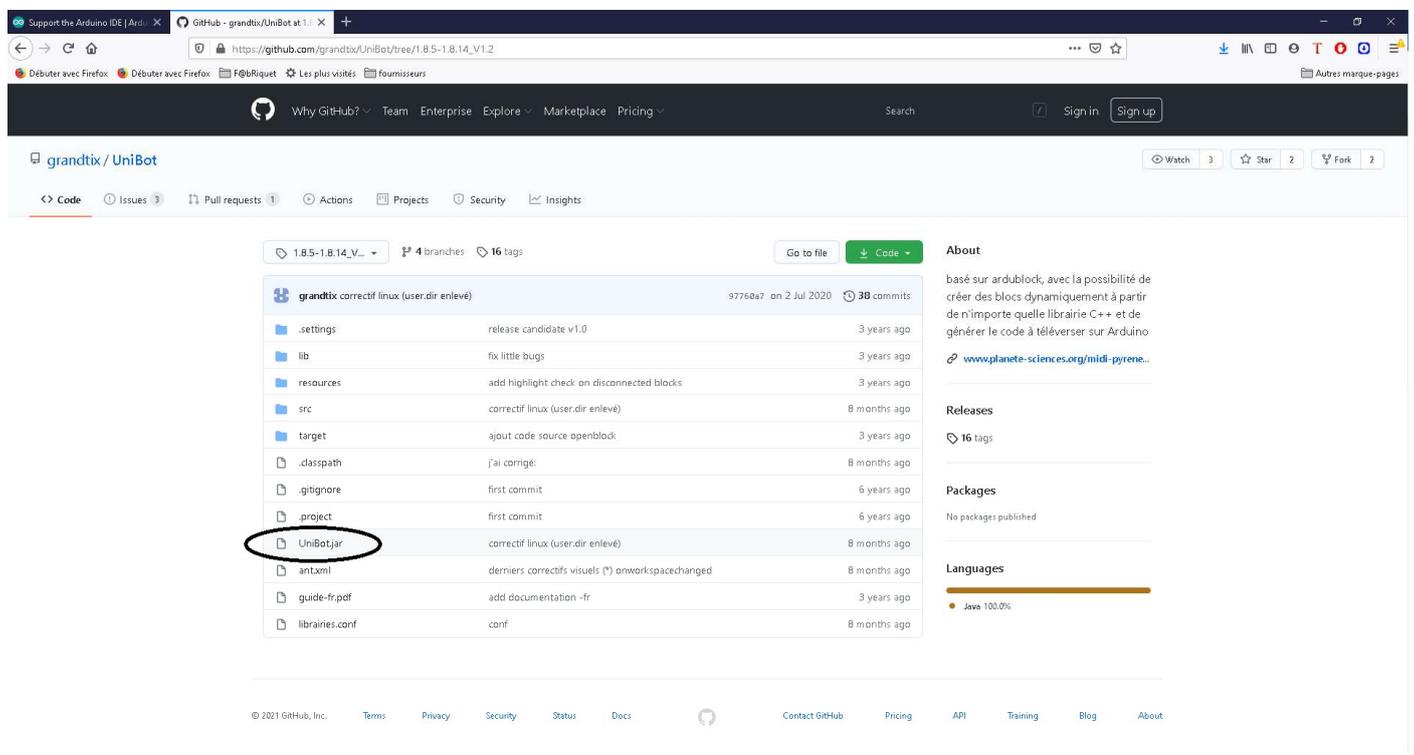
## 4. Installation d'UniBot, Boumbot et autres librairies

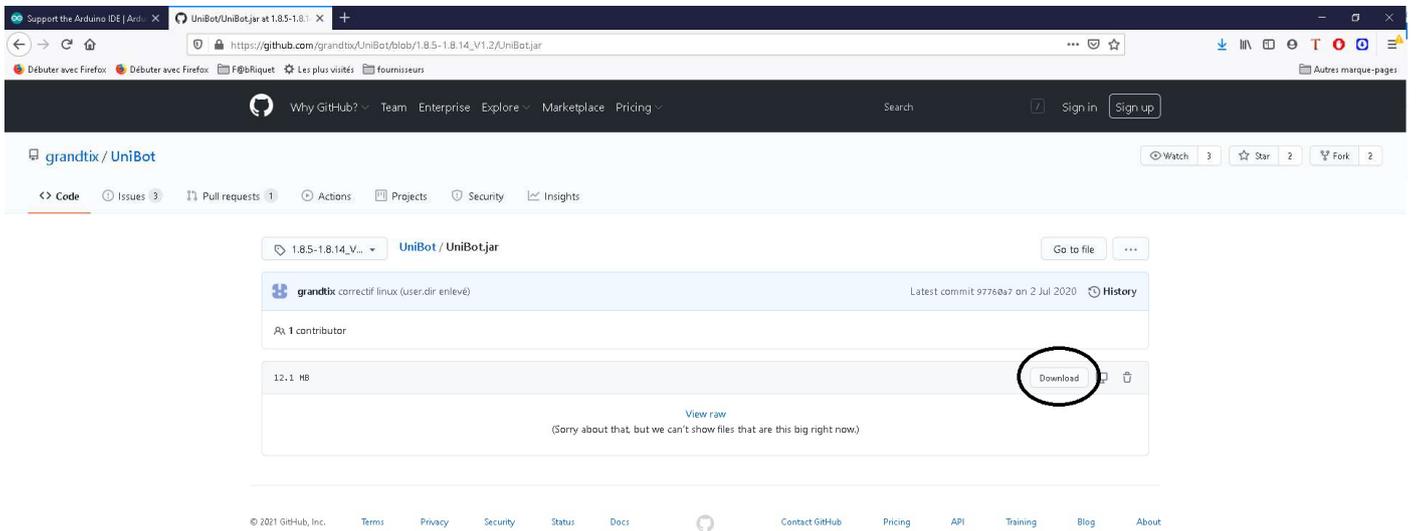
Téléchargez la dernière Unibot.jar à l'adresse suivante :

<https://github.com/grandtix/UniBot>



[https://github.com/grandtix/UniBot/tree/1.8.5-1.8.14\\_V1.2](https://github.com/grandtix/UniBot/tree/1.8.5-1.8.14_V1.2)





(À ce jour, c'est la version 1.8.5-1.8.14 v1.2, comptable avec les versions d'Arduino comprises entre 1.8.5 et 1.8.14 incluse)

Partant du principe qu'Arduino place le répertoire utilisateur dans les Documents (vérifiable dans Arduino, sous fichier → préférences)

Le fichier UniBot.jar est à placer dans le répertoire suivant :

**Documents/Arduino/tools/UniBot/tool/uniBot.jar**

Les librairies sont à installer dans le répertoire :

**Documents/Arduino/libraries**

Exemple :

**Document/Arduino/libraries/BoumBot/**

**BoumBot.h**

**BoumBot.cpp**

**img/**

**avance.png**

**recule.png**

**...**

## 5. Exécution

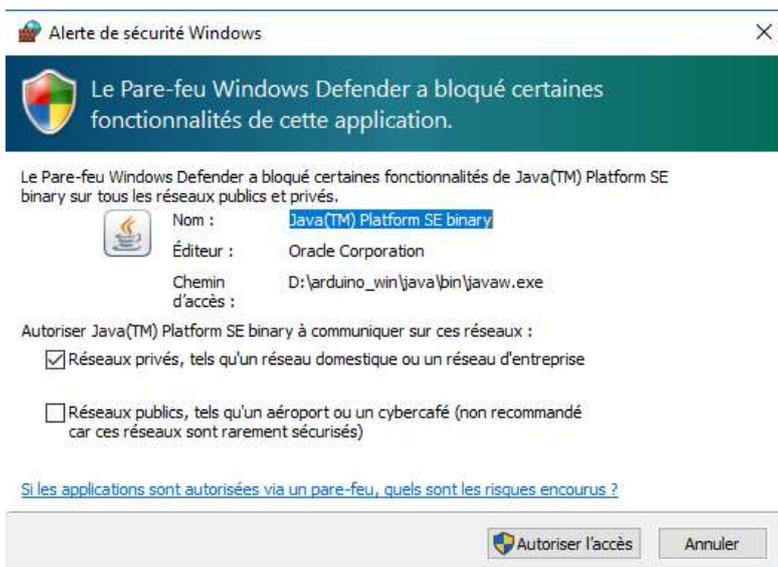
### 1. Lancer Arduino

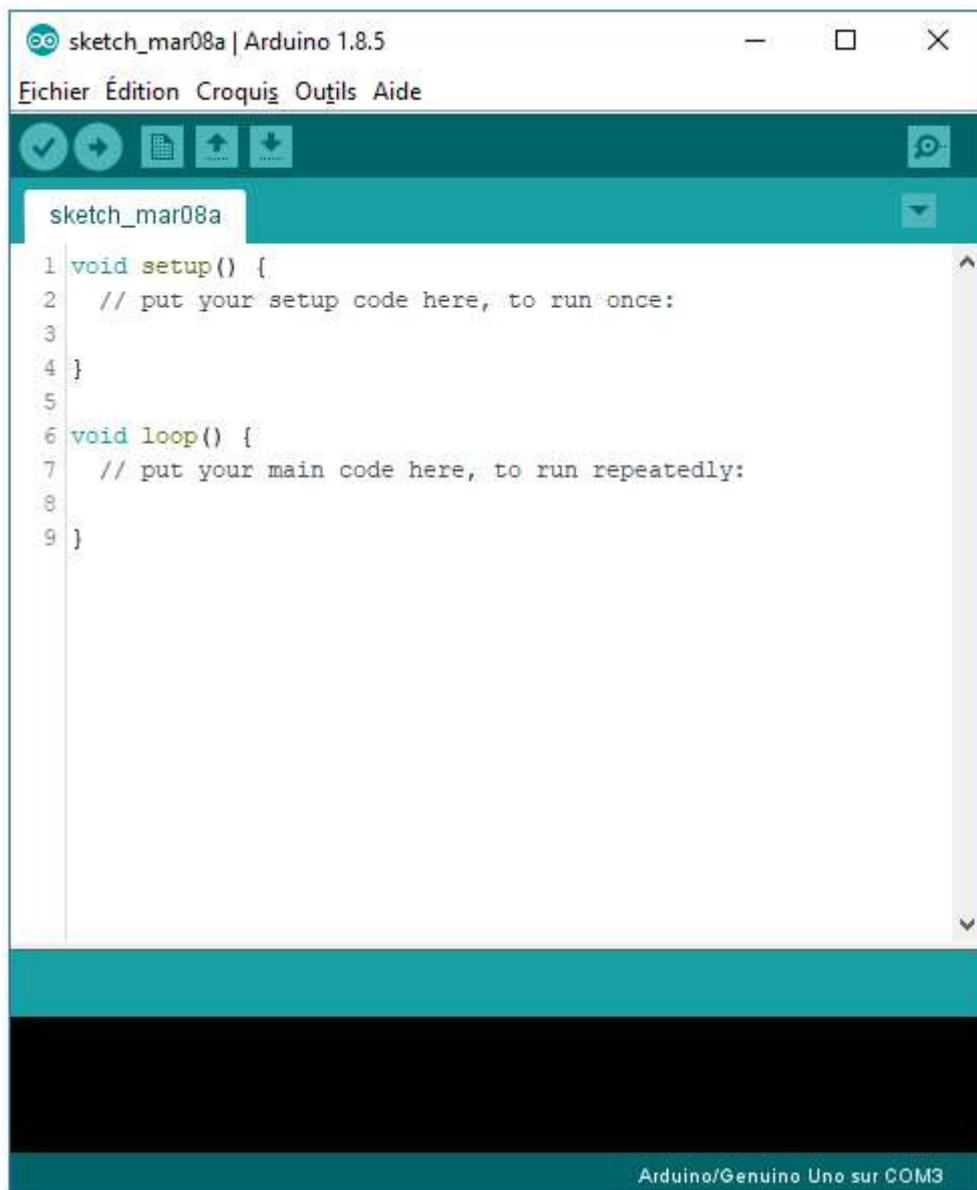
Lancez Arduino depuis le raccourci sur le bureau.

Arduino se lance :



Cette fenêtre peut apparaître (sous Windows seulement), cliquez sur « autoriser l'accès »

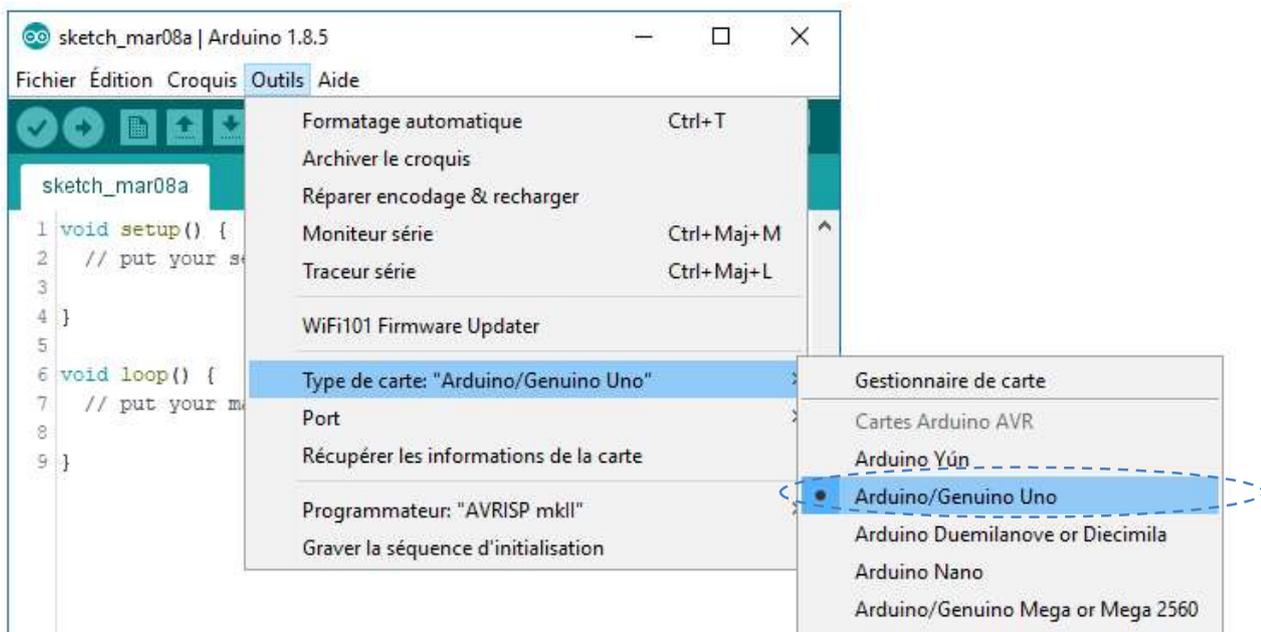




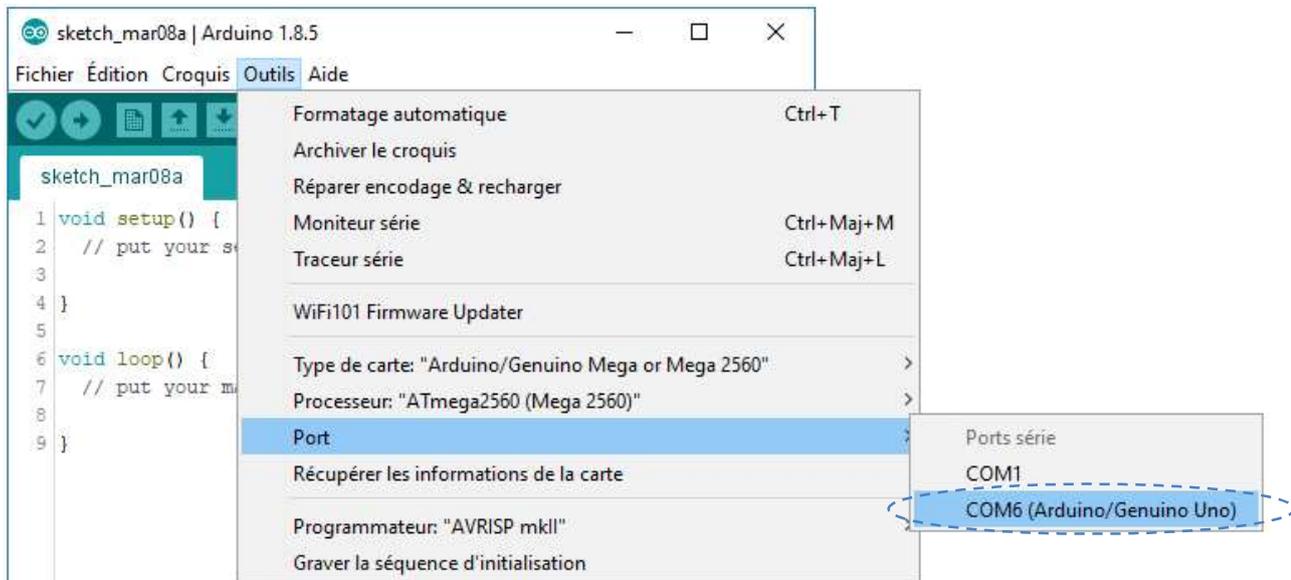
## 2. Configurer Arduino pour BoumBot

Le BoumBot doit être branché au PC.

Sélectionner « Arduino/Genuino Uno » dans «Outils» → «Type de carte»



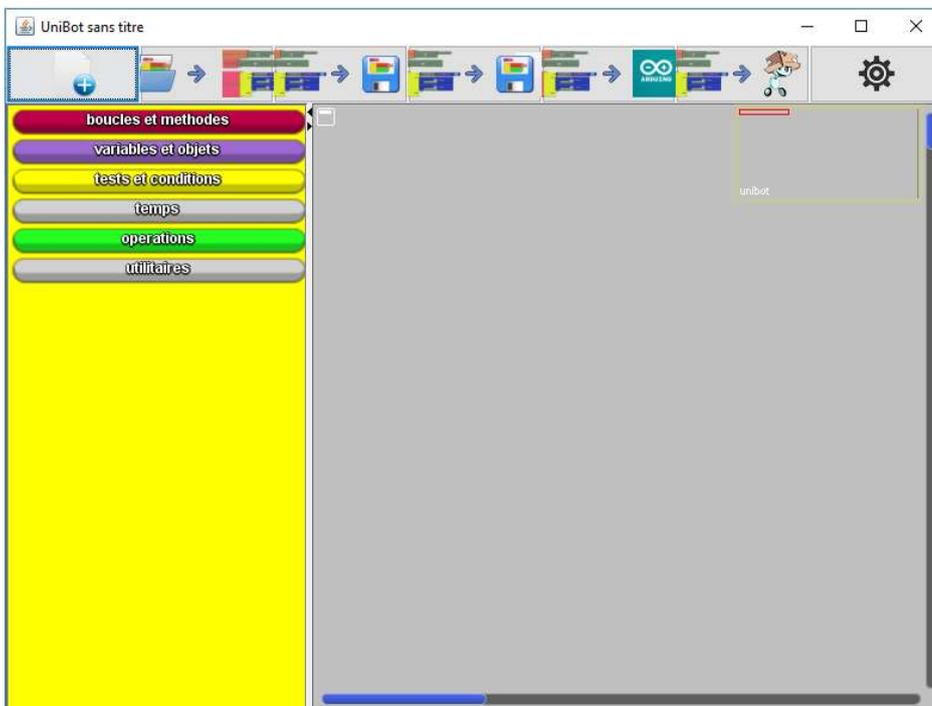
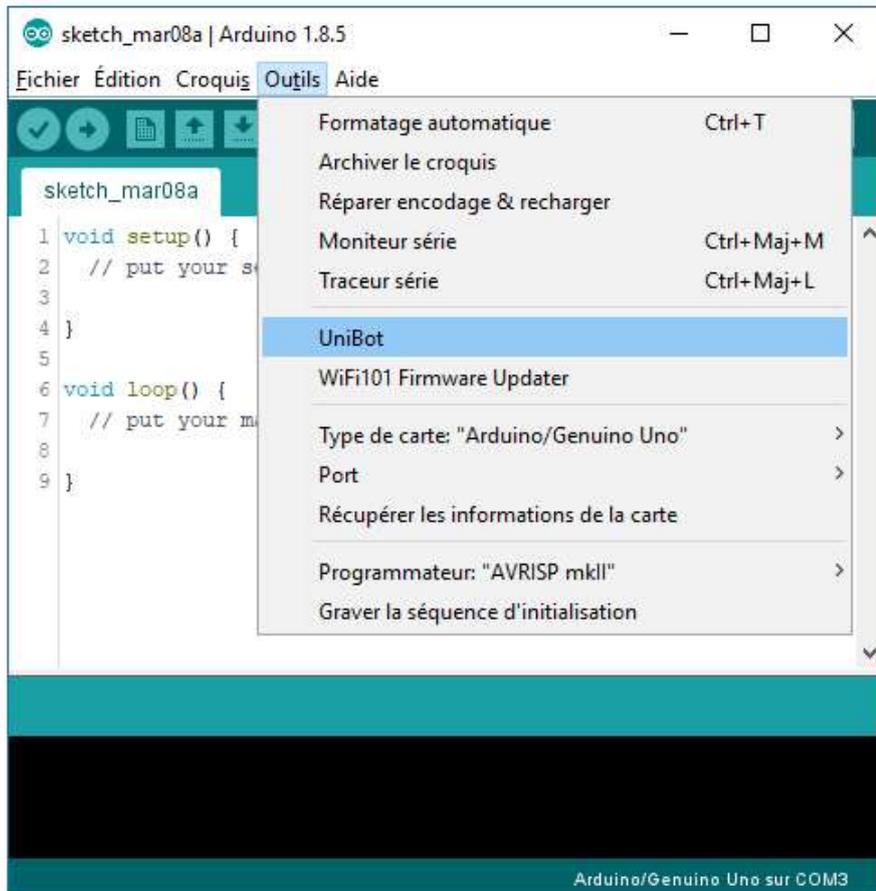
Puis sélectionnez le bon port COM, dans «Outils» → «Port»



Sous Linux, le port se nomme ttyACM[0-9]

### 3. Lancer UniBot

Cliquez sur « UniBot » dans « outils »



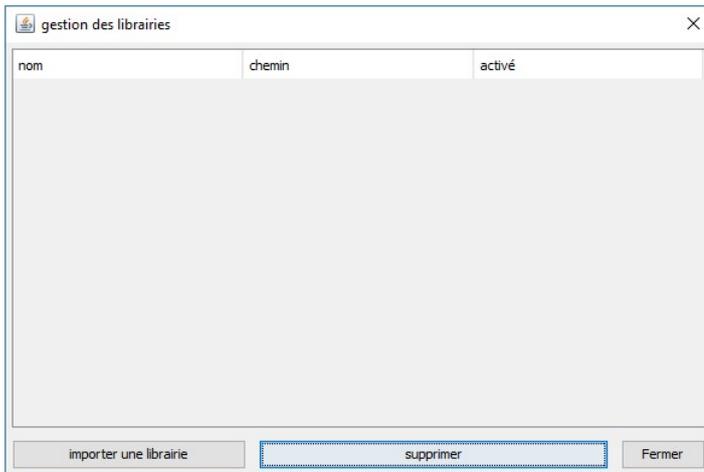
UniBot est un fichier exécutable java. « UniBot .jar » se trouve dans ce répertoire :

→ Documents/Arduino/tools/UniBot/tool/UniBot.jar

Le fichier librairies.conf situé à côté d'UniBot.jar contient les informations des librairies chargées et/ou mémorisées.

## 4. Charger la librairie BoumBot (ou autre)

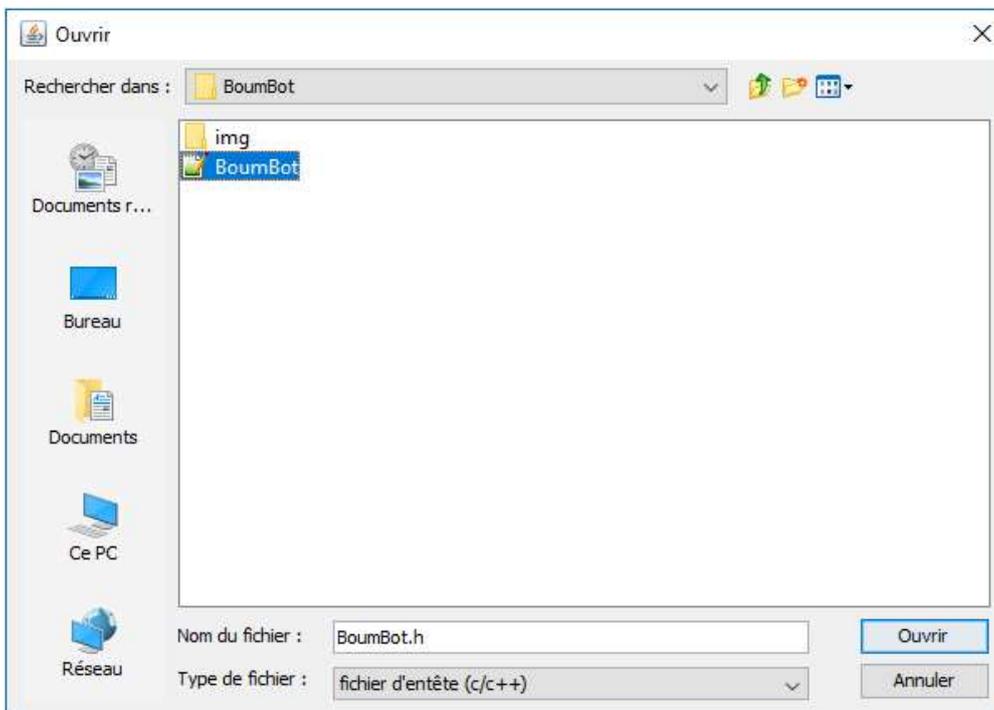
Cliquez sur le bouton , la fenêtre suivante s'ouvre :



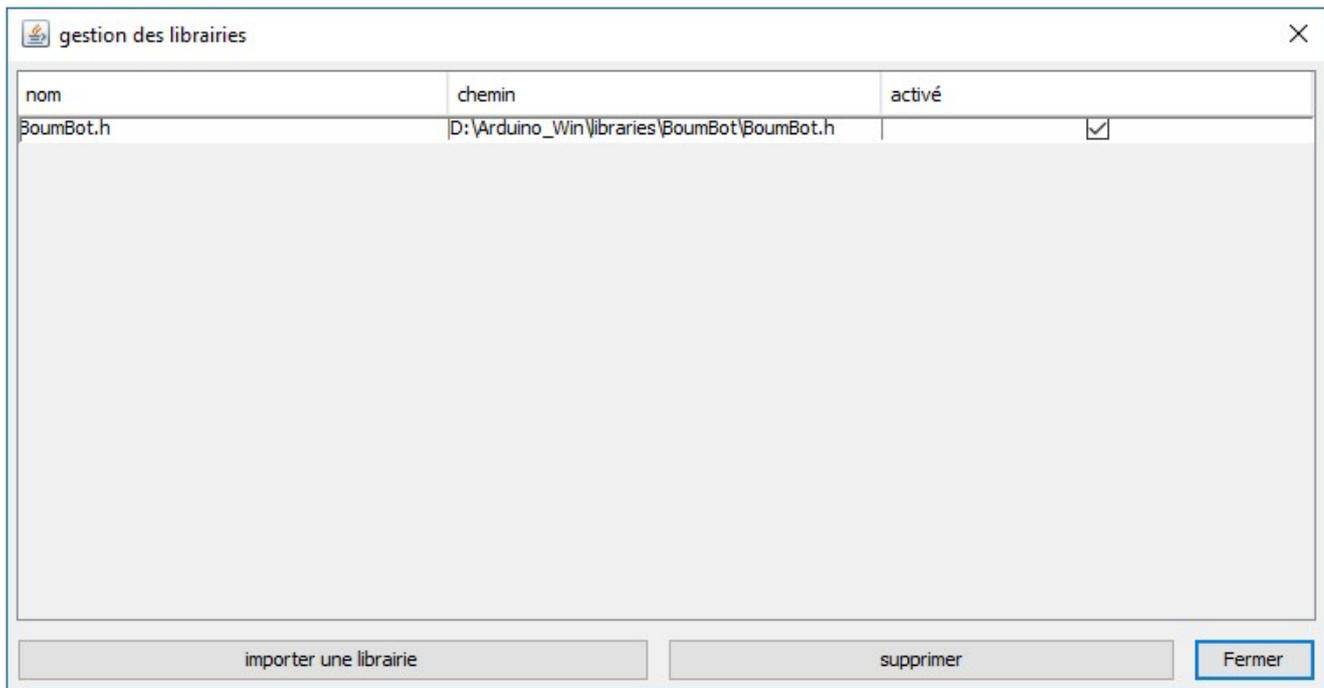
Cliquez sur « importer une librairie », une boîte de dialogue s'ouvre :

Allez chercher, au choix :

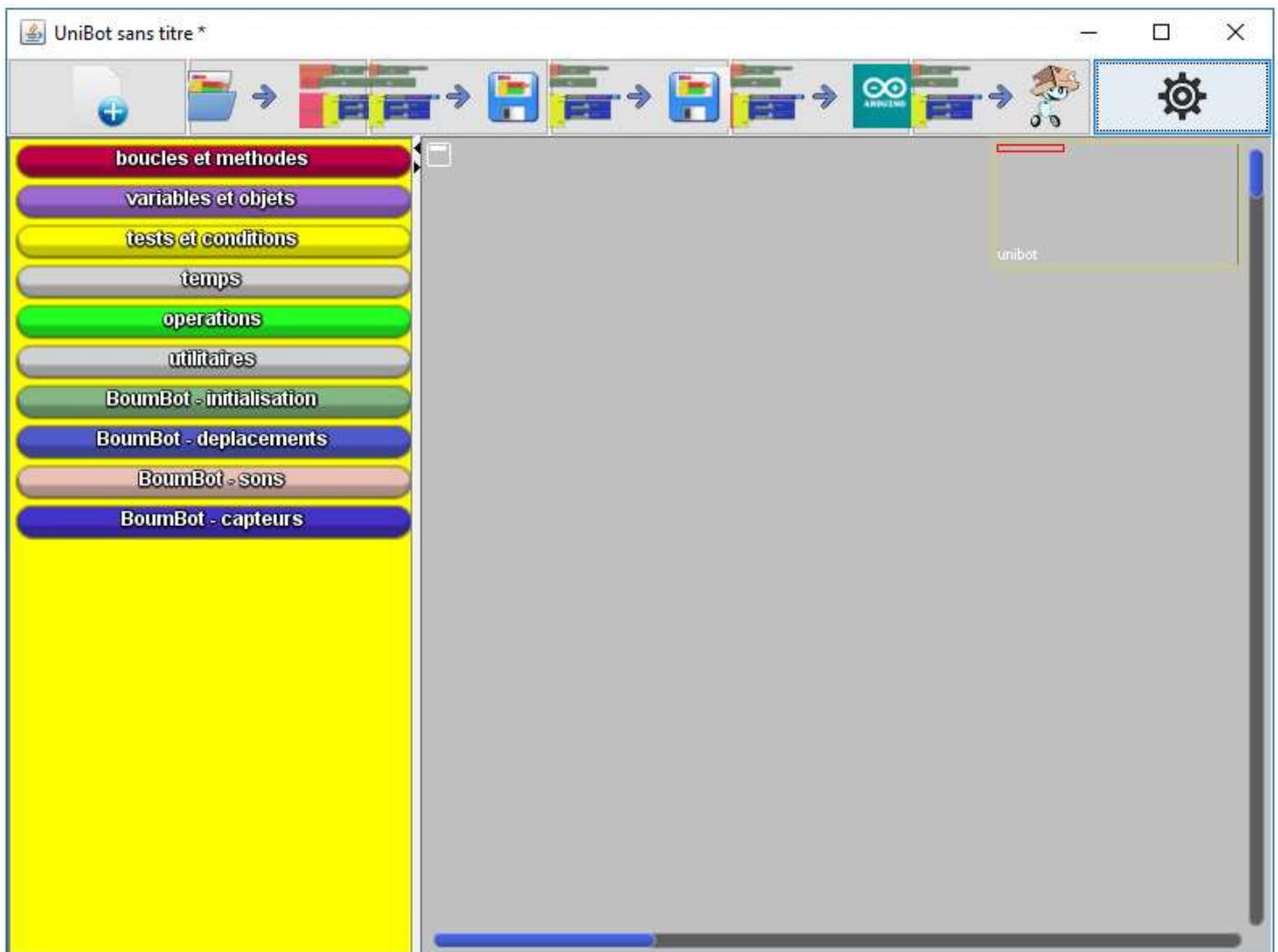
- *BoumBot.h* dans le répertoire *Arduino\librairies\BoumBot*
- *boumBotGrille.h* dans le répertoire *Arduino\librairies\BoumBotGrille*
- vos propres librairies (classes)...



La colonne de droite permet d'afficher/occulter les blocs de la librairie dans UniBot, tout en les gardant dans la bibliothèque.



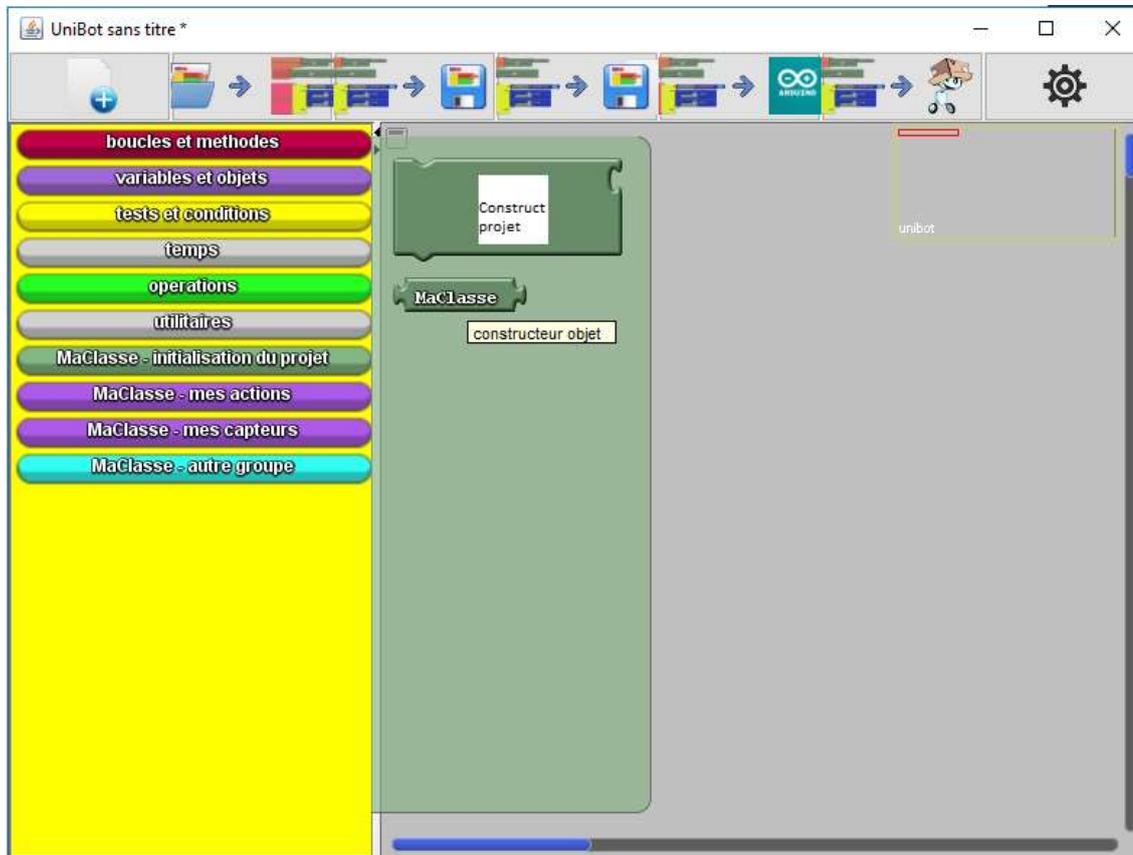
Fermer la fenêtre, de nouveaux blocs sont dans la liste de droite d'UniBot.



## 6. Créer vos bibliothèques

### 1. Préambule

UniBot peut prendre en charge une multitude de classes en C/C++ fonctionnant pour Arduino.



### 2. Format du code source

Pour adapter une classe à UniBot, il suffit de faire précéder le variable ou la fonction par une ligne de commentaire avec ce format :

---

```
//@initialisation du projet  
//@bloc texte= constructeur objet png=constructeur.png  
MaClasse();
```

---

Le bloc MaClasse sera visible dans le sous-bloc « initialisation du projet ». Il y aura le bloc pour créer une instance de la classe et un bloc pour la variable instance de cette même classe.

Il en va de même pour les fonctions :

---

```
//@mes actions  
//@bloc texte=action 1 png=action1.png  
int action1();  
//@bloc texte=action 2 png=action2.png  
void action2();  
//@bloc texte=action 3 png=action3.png  
char* action3();
```

---

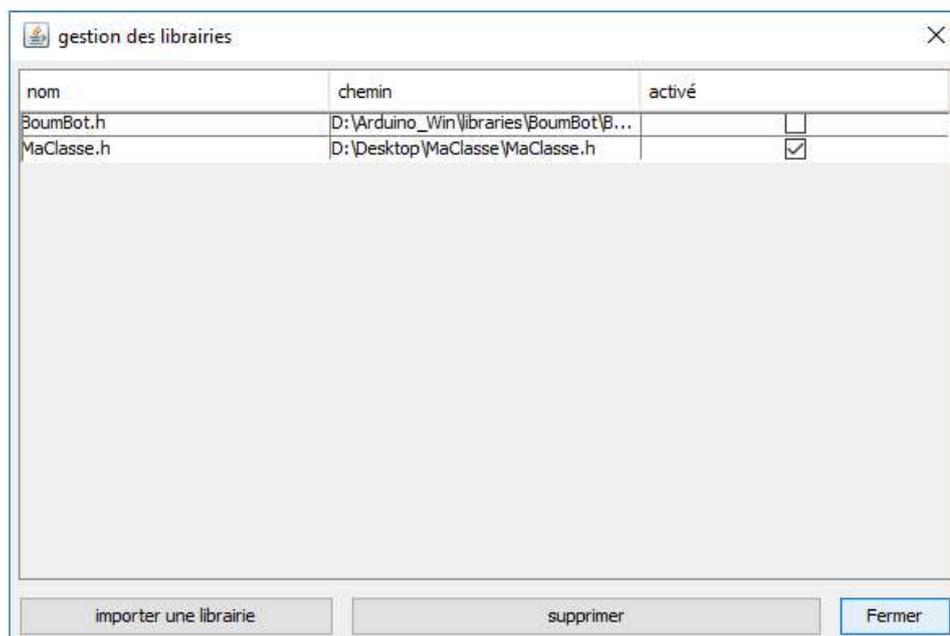
### 3. Hiérarchie de répertoires et fichiers

Voici la structure de fichiers pour votre classe :

```
MaClasse/  
  Maclasse.cpp  
  MaClasse.h  
  img/  
    consructeur.png  
    action1.png  
    action2.png  
    action3.png
```

Le répertoire img contient les images qui sont affichées dans les blocs.

Il ne vous reste qu'à inclure votre Classe dans la bibliothèque d'UniBot.



### 4. Limitations

Les limitations sont les suivantes :

- ❖ Les types : int, bool, String, float sont pris en charge
- ❖ Les fonctions ne prennent au maximum qu'un paramètre, au delà, le bloc n'est plus lisible
- ❖ UniBot ne charge que les fonctions et variables publiques

## 7. Boum'Bot vu de l'intérieur

### 1. La carte Arduino

C'est le « cerveau » du BoumBot, c'est dans cette carte qu'est stocké le programme que vous avez réalisé avec Arduino/UniBot.

Le modèle utilisé est la « Arduino Uno ». Elle propose en plus d'un microcontrôleur (grosse calculatrice avec un peu de mémoire) des connecteurs (ou « pins ») analogiques et numériques afin de pouvoir « dialoguer » avec des moteurs, capteurs et autres périphériques.

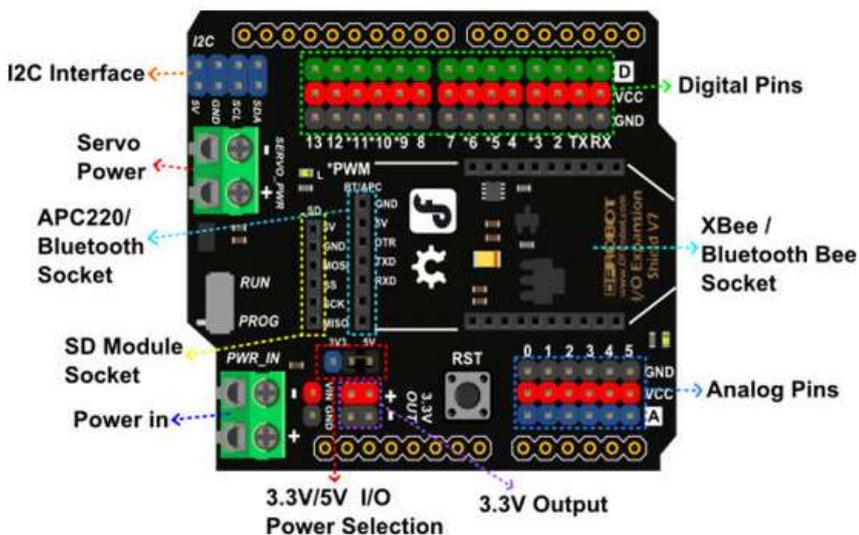
Voici ses principales caractéristiques :

- 6 entrées analogiques
- 14 entrées/sorties numériques



### 2. Le « shield »

C'est la carte située sur la carte Arduino. Elle réagence les pins en les plaçant par groupe de 3. 1 pin pour le signal (analogique/numérique, 2 pins pour l'alimentation du capteur ou actionneur.

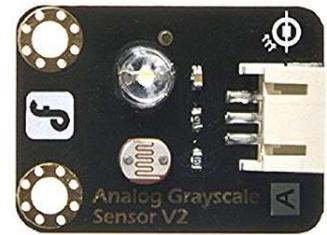


### 3. Les capteurs d'échelle de gris

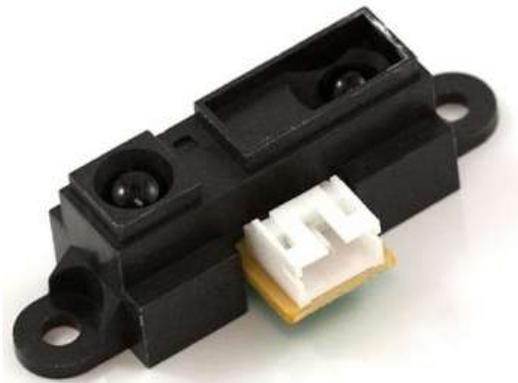
Ce sont des capteurs analogiques, cela signifie que leur réponse va de 0 jusque 5v, qui se traduit par une valeur allant de 0 jusque 1024. Plus le sol est sombre et plus la valeur renvoyée par le capteur tend vers 0 et plus il est clair, vers 1024.

Ils fonctionnent de la manière suivante :

1 LED éclaire la surface, et la photorésistance reçoit la lumière réfléchiée par cette surface.

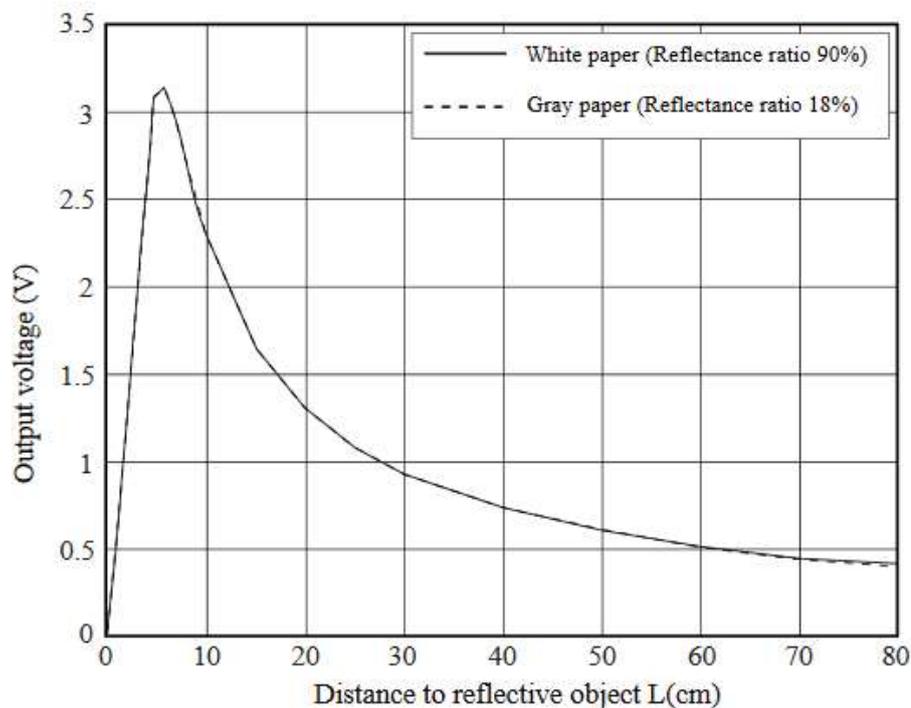


### 4. Le capteur de proximité



Il est efficace entre 10 et 80cm.

Voici sa courbe de réponse :



## 5. Les servomoteurs à rotation continue



Ces moteurs peuvent tourner en continu, dans les 2 sens. Ils sont pilotables en vitesse et direction.

On utilise la même librairie que pour un servomoteur classique, mais ici, la valeur 90 arrête le moteur (cette valeur est approximative), plus la valeur s'approche de 180 et plus le servomoteur tourne vite dans un sens, idem vers le 0.

Pour le piloter depuis Arduino :

```
#include <Servo.h>
```

```
Servo roue;
```

```
roue.attach(pinNumber);
```

```
roue.write(entre 0 et 180);
```

## 6. Les potentiomètres

Ce sont des capteurs analogiques, ils renvoient une valeur entre 0 et 1024 par la commande Arduino suivante :

```
analoRead(PinNumber) ;
```

Ils sont utilisés pour « fixer » la valeur d'arrêt des servomoteurs. 90 est la valeur moyenne, mais une légère variation due aux caractéristiques de chaque servomoteur existe. Avant de passer la main devant le capteur de proximité, tournez les potentiomètres jusqu'à arrêt complet des roues.



## 7. La capsule piézoélectrique (buzzer)



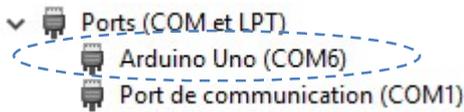
## 8. Aide

### 1. Téléversement sous Windows

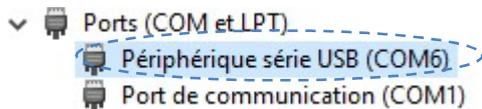
#### 1. Dans le « gestionnaire de périphériques »

**Astuce :** raccourci clavier «win+pause», puis cliquez sur gestionnaire de périphérique en haut à gauche de la fenêtre

Vous devez y voir ceci :



Si vous voyez ceci :



Clic-droit sur « périphérique série USB », et « mettre à jour le pilote », puis suivez les instructions d'installation du driver plus haut.

Si Arduino Uno n'apparaît pas, vérifiez/changez le câble.

Essayez un autre Boum'Bot et recommencez le diagnostic.

Quand vous voyez la ligne Arduino Uno, notez la valeur COMXXX écrit juste à droite. Elle est utilisée dans Arduino.

#### 2. Droits d'administration

L'installation du driver, nécessite les droits administrateurs (window 7 et +).

Assurez-vous que l'utilisateur Windows que vous utilisez soit administrateur. Si ce n'est pas le cas, connectez-vous avec un compte administrateur ou prenez contact avec l'administrateur de la machine.

#### 3. Dans Arduino

Vérifiez le type de carte et le port COM

### 1. Téléversement sous Linux

Vérifiez que votre utilisateur fasse bien partie du groupe dialout

Ouvrez un terminal, et lancez la commande :

*ld*

## 2. Compilation

### 1. Micro rappel de programmation

On rappelle que BoumBot est une classe, et lorsque l'on ajoute le bloc nouveau BoumBot, on crée une instance de cette classe qui portera par défaut le nom « boumbot » (note : tout en minuscule). Cette instance est aussi appelée variable, on dit alors que c'est la variable boumbot de type BoumBot. On pourrait lui donner un nom différent...pierre, paul, jaques...terminabot

Les variables sont utilisées dans tous les programmes, partout.

Dans Arduino, elle peut être soit globale, soit locale :

- En global, elle est vue par tout le programme.
- En local, elle est créée dans une fonction et n'est vue qu'à l'intérieur de cette dernière. On peut les faire « passer » d'une fonction à une autre en utilisant des paramètres.

Les fonctions, comme leur nom l'indique, réalisent un traitement (calcul, action, édition de texte...).

Elles peuvent accepter des arguments (des variables) et renvoyer un résultat (une variable).

### 2. Les erreurs fréquentes

- Les blocs « boucle principale, « initialisation des variables », et « déclaration des variables » ne doivent apparaître qu'une seule fois dans UniBot
- De même, si le programme Arduino comporte plusieurs onglets, les fonctions setup() et loop() ne doivent apparaître qu'une seule fois.
- Le nom d'une variable n'est pas le même dans tous les blocs, exemple : majuscules et minuscules
  - ✓ Vérifiez l'orthographe des noms de variables
- Un bloc n'est pas « bien » connecté à son voisin
  - ✓ Déplacez le bloc pour le recoller à son voisin. Si il ne colle pas, vérifiez que les connecteurs soient les mêmes
- Une variable est déclarée dans une fonction et utilisée dans une autre.
  - ✓ Pour qu'une variable soit vue, elle doit soit être :
    - ✓ Utilisée dans la même fonction que celle où elle est créée
    - ✓ Globale, ie : créée dans le bloc « déclaration des variables »
    - ✓ Passée en paramètre d'une fonction : variable accolé coté droit du bloc « fonction »
- Les roues ne tournent pas ou les capteurs ne fonctionnent pas
  - ✓ Les câbles ont été rebranchés aux mauvais endroits.
    - ✓ Ouvrez BoumBot.h, vous y verrez les attributions de pins pour les moteurs et capteurs
  - ✓ Le capteur est HS
    - ✓ Essayez un autre BoumBot
- Les capteurs de ligne fonctionnent mal
  - ✓ Etalonnez en lançant le programme Arduino d'étalonnage sur la clé USB

.....et pensez aux piles !