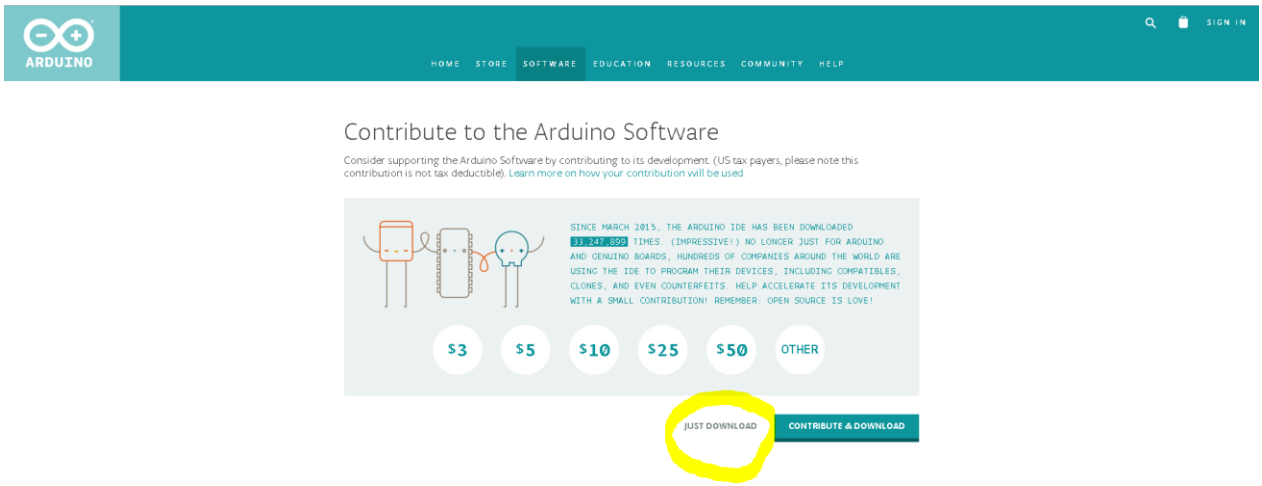
Programmer un microcontrôleur type Arduino™ en langage Arduino™

1) Installation de l'éditeur Arduino™ :

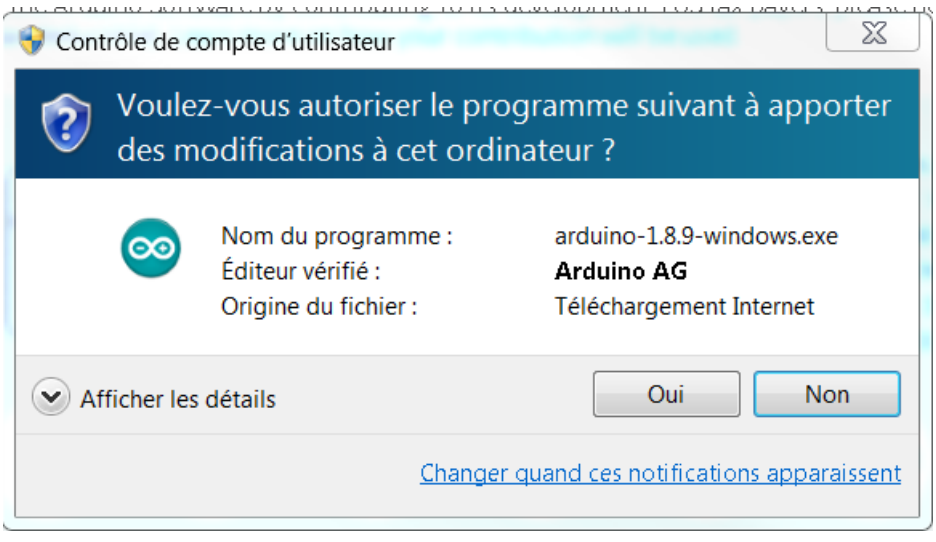
La programmation d'un microcontrôleur Arduino™ nécessite l'utilisation de l'éditeur Arduino™ IDE.
Ce logiciel est disponible en téléchargement gratuit :
http://arduino.org/

https://www.arduino.cc/en/Main/



Cliquer sur « JUST DOWNLOAD »

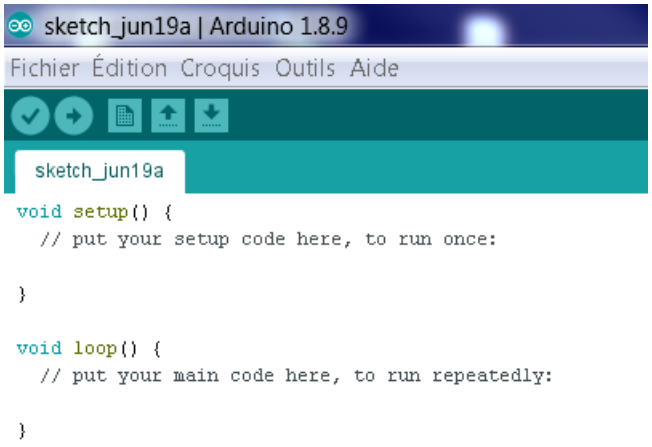
Lancer l'exécutable et accepter l'installation de tous les composants du logiciel Arduino™ :



Lancer ensuite le logiciel Arduino™ à partir de l'icône raccourci :


Arduino


ARDUINO
Genuino
AN OPEN PROJECT WRITTEN, DEBUGGED,
AND SUPPORTED BY ARDUINO.CC AND
THE ARDUINO COMMUNITY WORLDWIDE
LEARN MORE ABOUT THE CONTRIBUTORS
OF ARDUINO.CC on arduino.cc/credits
Démarrage


sketch_jun19a | Arduino 1.8.9
Fichier Édition Croquis Outils Aide
sketch_jun19a
void setup() {
// put your setup code here, to run once:
}
void loop() {
// put your main code here, to run repeatedly:
}

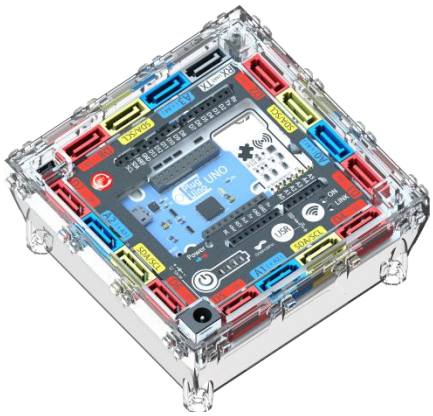
2) Etablir la liaison entre l'ordinateur et le microcontrôleur Plug'Uino® Uno :

Brancher l'interface Plug'Uino® sur un des ports USB de votre ordinateur, à l'aide du câble USB fourni avec Plug'Uino®Uno.

Si vous disposez de la version Plug'Uino® Uno Cube réf. 650001, photo ci-dessous :

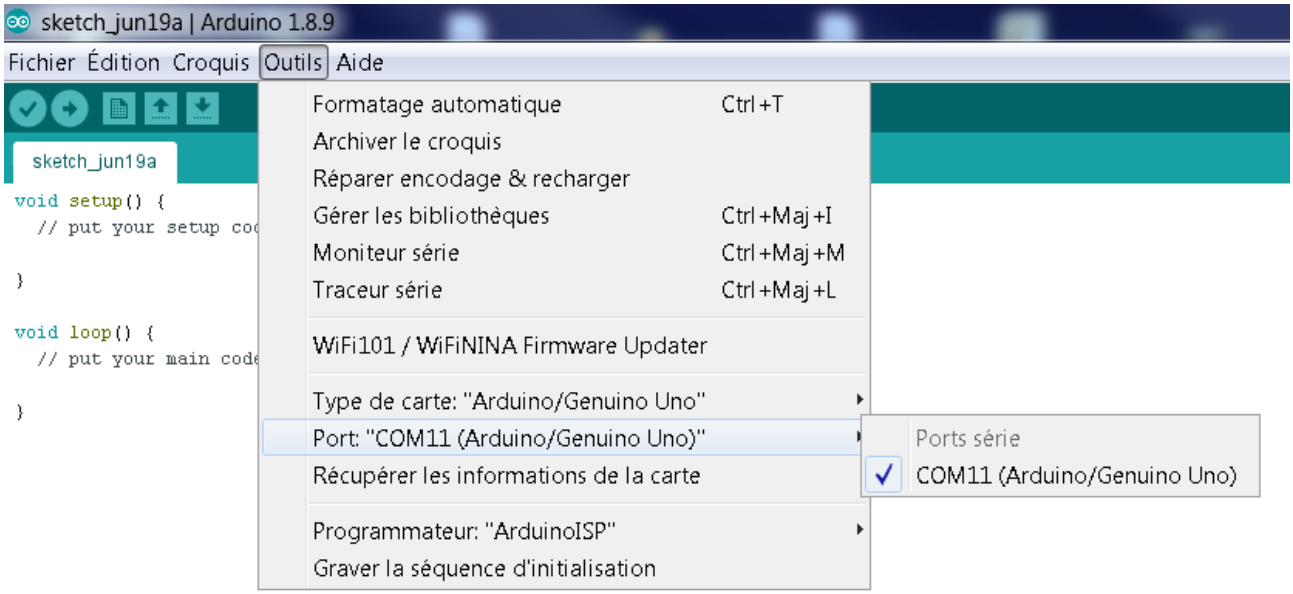
Appuyer sur le bouton ON/OFF près du symbole de la batterie, pour activer les entrées/sorties.

Cette opération est inutile sur la version Plug'Uino® Uno réf. 650 003



Etablir la liaison entre l'ordinateur et le microcontrôleur Plug'Uino® Uno :
Menu/Outils/Port

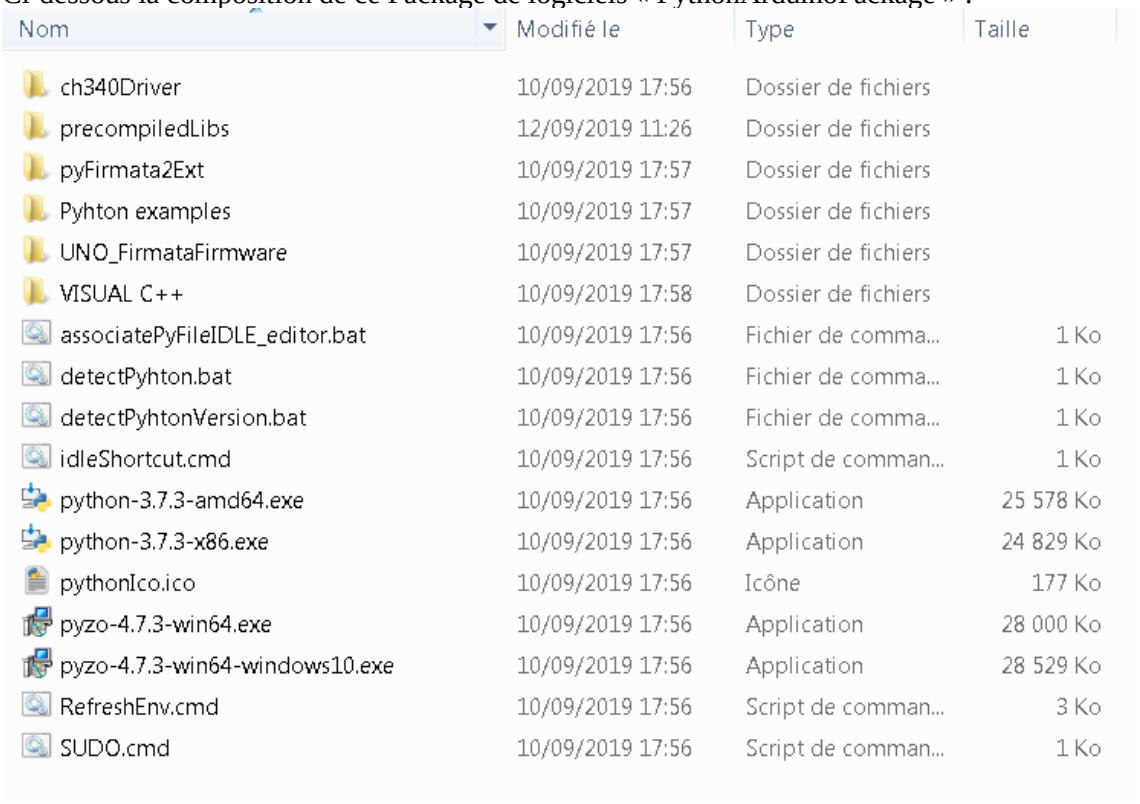
Sélectionner ✓ le port détecté suite au branchement du microcontrôleur sur votre ordinateur :



Le port détecté peut s'afficher « COMXX (Arduino/Génuino Uno) » ou simplement « COMXX ».
Il peut y avoir également plusieurs ports COM affichés, il convient alors de déterminer celui qui est utilisé pour la liaison série avec Arduino. Pour cela, débranché et branché le câble USB qui relie votre ordinateur à la carte Arduino et noter celui qui disparait de la liste. C'est celui qu'il faudra cocher une fois la liaison rétablie.

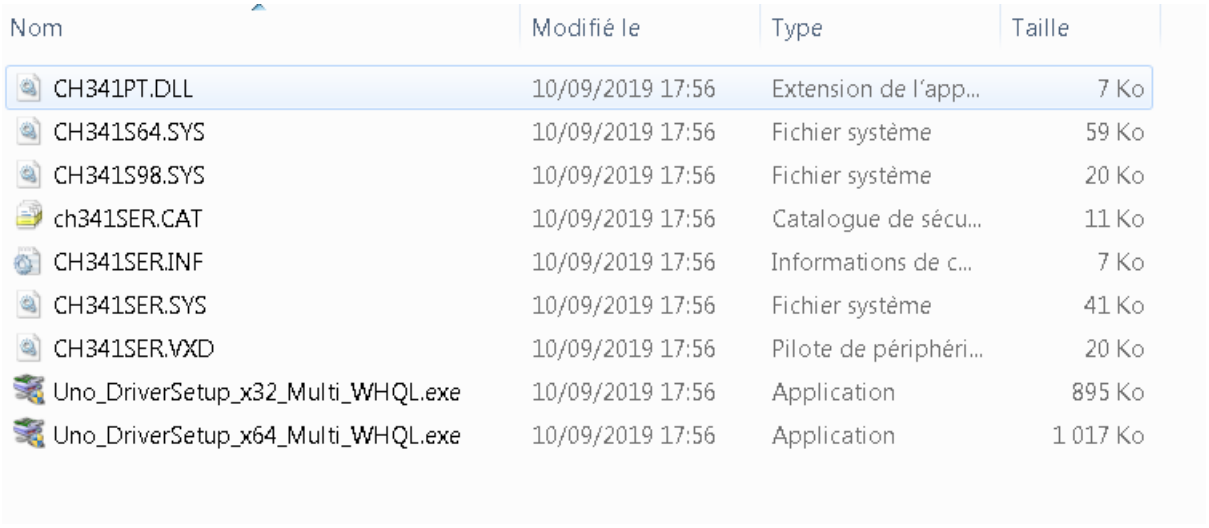
Si aucun port COM n'est pas détecté pour votre carte Arduino, c'est que le pilote de la carte n'est pas installé sur votre version de Windows. Dans ce cas, **il convient d'installer le pilote CH340**, dossier d'installation disponible dans le Package de logiciels «PythonArduinoPackage» que Sciencéthic fournit gratuitement sur demande des utilisateurs de ses microcontrôleurs et accessoires.

Ci-dessous la composition de ce Package de logiciels « PythonArduinoPackage » :



Nom	Modifié le	Type	Taille
ch340Driver	10/09/2019 17:56	Dossier de fichiers	
precompiledLibs	12/09/2019 11:26	Dossier de fichiers	
pyFirmata2Ext	10/09/2019 17:57	Dossier de fichiers	
Pyhton examples	10/09/2019 17:57	Dossier de fichiers	
UNO_FirmataFirmware	10/09/2019 17:57	Dossier de fichiers	
VISUAL C++	10/09/2019 17:58	Dossier de fichiers	
associatePyFileIDLE_editor.bat	10/09/2019 17:56	Fichier de comma...	1 Ko
detectPyhton.bat	10/09/2019 17:56	Fichier de comma...	1 Ko
detectPyhtonVersion.bat	10/09/2019 17:56	Fichier de comma...	1 Ko
idleShortcut.cmd	10/09/2019 17:56	Script de comman...	1 Ko
python-3.7.3-amd64.exe	10/09/2019 17:56	Application	25 578 Ko
python-3.7.3-x86.exe	10/09/2019 17:56	Application	24 829 Ko
pythonIco.ico	10/09/2019 17:56	Icône	177 Ko
pyzo-4.7.3-win64.exe	10/09/2019 17:56	Application	28 000 Ko
pyzo-4.7.3-win64-windows10.exe	10/09/2019 17:56	Application	28 529 Ko
RefreshEnv.cmd	10/09/2019 17:56	Script de comman...	3 Ko
SUDO.cmd	10/09/2019 17:56	Script de comman...	1 Ko

Détail du contenu du dossier ch340Driver :



Nom	Modifié le	Type	Taille
CH341PT.DLL	10/09/2019 17:56	Extension de l'app...	7 Ko
CH341S64.SYS	10/09/2019 17:56	Fichier système	59 Ko
CH341S98.SYS	10/09/2019 17:56	Fichier système	20 Ko
ch341SER.CAT	10/09/2019 17:56	Catalogue de sécu...	11 Ko
CH341SER.INF	10/09/2019 17:56	Informations de c...	7 Ko
CH341SER.SYS	10/09/2019 17:56	Fichier système	41 Ko
CH341SER.VXD	10/09/2019 17:56	Pilote de périphéri...	20 Ko
Uno_DriverSetup_x32_Multi_WHQL.exe	10/09/2019 17:56	Application	895 Ko
Uno_DriverSetup_x64_Multi_WHQL.exe	10/09/2019 17:56	Application	1 017 Ko

3) Exécuter un programme existant en langage Arduino™ (.ino):

Attention : pour « téléverser » un programme .ino, il est indispensable que le fichier « nomdefichier.ino» soit placé dans un dossier du même nom « nomdefichier », avec exactement la même syntaxe, voir exemple ci-dessous :

Dossier

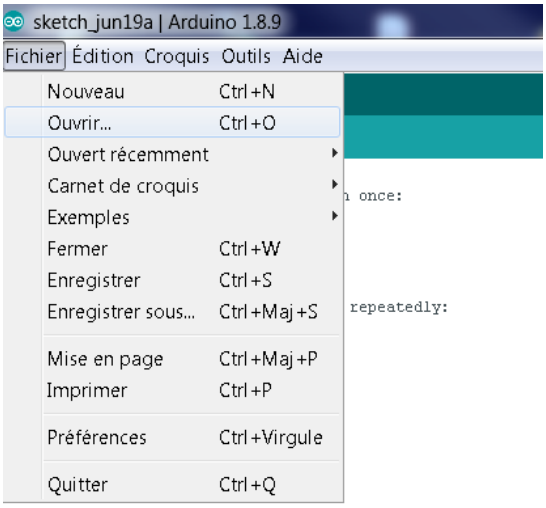
Contenu du dossier :

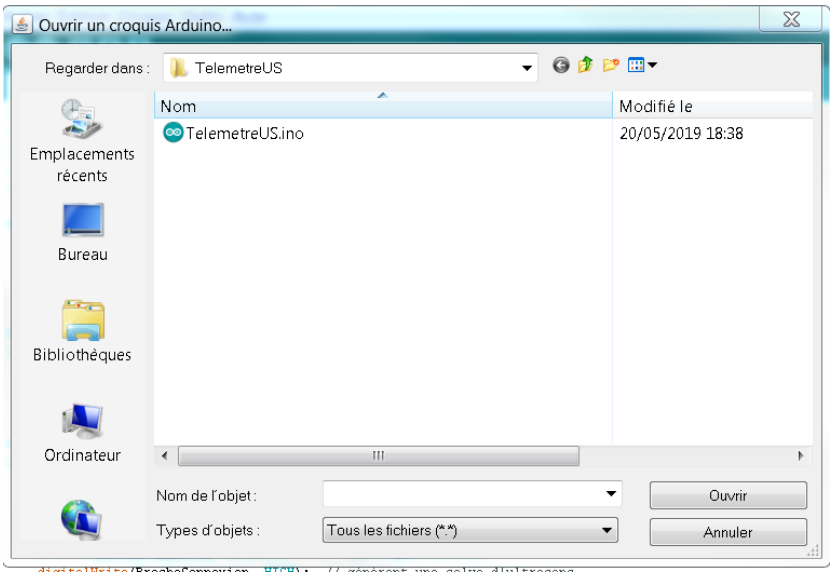
3-1 Exécuter le programme TelemetreUS.ino

Dans cet exemple, nous utilisons le programme permettant de mesurer la vitesse du son à l'aide du télémètre à US Plug'Uino® Uno branché **sur la voie D7** du Plug'Uino® Uno.

(Ce programme .ino est disponible dans votre « Espace Enseignant », en téléchargement gratuit sur www/sciencethic.com)

Menu/Fichier/Ouvrir…





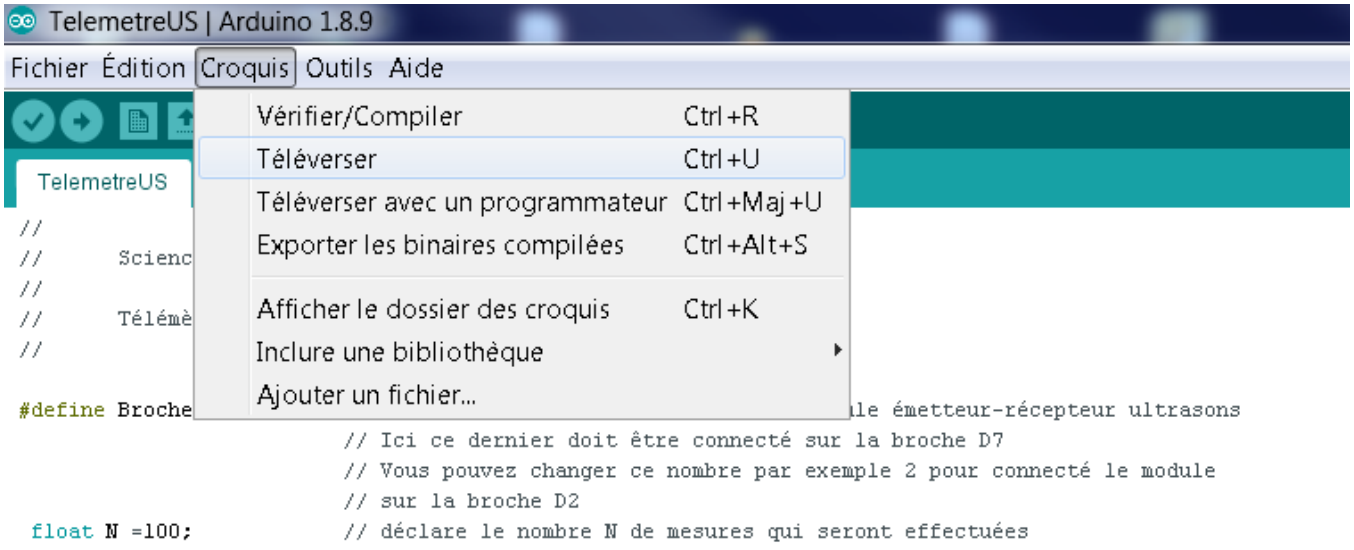
TelemetreUS | Arduino 1.8.9

Fichier Édition Croquis Outils Aide

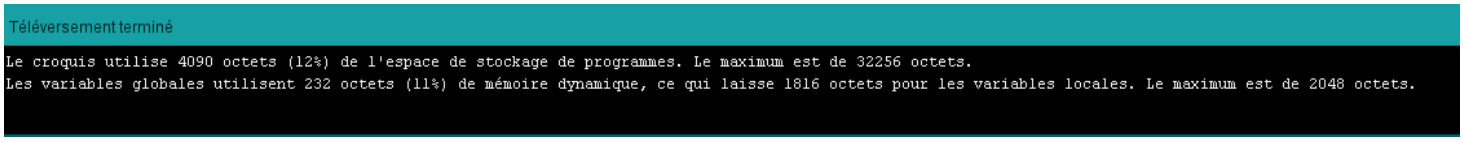
TelemetreUS

```
//
//      Sciencéthic
//
//      Télémètre à Ultrasons
//

#define BrocheConnexion 7 // Définit la broche de connexion du module émetteur-récepteur ultrasons
                          // Ici ce dernier doit être connecté sur la broche D7
                          // Vous pouvez changer ce nombre par exemple 2 pour connecté le module
                          // sur la broche D2
 float N =100;            // déclare le nombre N de mesures qui seront effectuées

void setup()              // Initialisation du programme
{
    Serial.begin(9600);   // Initialise la vitesse de communication entre la carte Plug'Uino et le PC
                          // à 9600 bps

    }
void loop()               // Boucle principale
{
  float distance = 0.3;   // Valeur de la distance entre l'émetteur-récepteur et l'obstacle en mètre
  float duree = 0;  // initialise la variable duree à 0
  for(int i = 1; i<=N; i++) // démarre une boucle de N mesures
  {
    pinMode(BrocheConnexion, OUTPUT);     // Les lignes 25 à 28
    digitalWrite(BrocheConnexion, HIGH);  // génèrent une salve d'ultrasons
    delayMicroseconds(5);                 //
    digitalWrite(BrocheConnexion,LOW);    //
    pinMode(BrocheConnexion,INPUT);       //
    duree = duree + pulseIn(BrocheConnexion,HIGH); // mesure la durée d'aller-retour de la salve et
                                                   // l'ajoute aux mesures précédentes
    delay(200);                            // Attend 200 ms avant de renvoyer une salve (évite les échos parasites)
  }
  float dureemoyenne = duree*1e-6/N ;         // calcule la durée moyenne d'aller-retour, convertie en s
  float vitesse = distance*2/dureemoyenne ;   // Calcule la vitesse qui correspond au rapport de la
                                              // distance parcourue par l'onde (soit le double de la distance
                                              // entre le module émetteur-récepteur et l'obstacle)
                                              // par la durée moyenne d'aller-retour
  Serial.print("La vitesse moyenne est de "); // Affiche la phrase entre guillements
  Serial.print(vitesse);                      // Affiche la vitesse
  Serial.println(" m/s");                     // Affiche l'unité, et passe à la ligne
  delay(250);                                 // Attend 250 ms avant de recommencer
}
```

Il est ensuite nécessaire d'introduire (« Téléverser ») ce programme dans le microcontrôleur pour qu'il puisse l'exécuter :



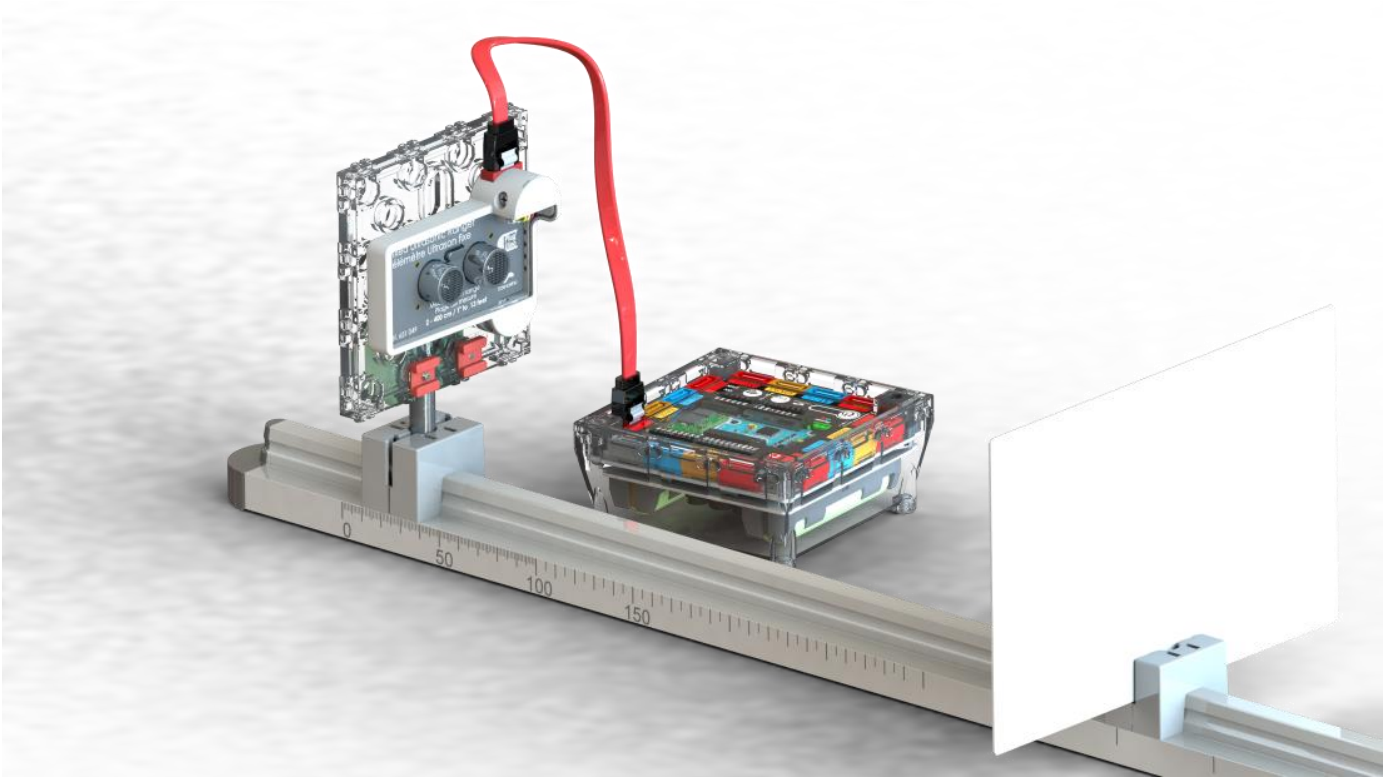
Ce « téléversement » prend quelques secondes…
Une fois réalisé, le message ci-dessous s'affiche en bas de la fenêtre :



Placer le capteur télémètre à US réf. 651 049, branché sur la voie D7 de Plug'Uino® Uno, à une distance de 30 cm d'un écran, comme indiqué dans le programme ligne 21,

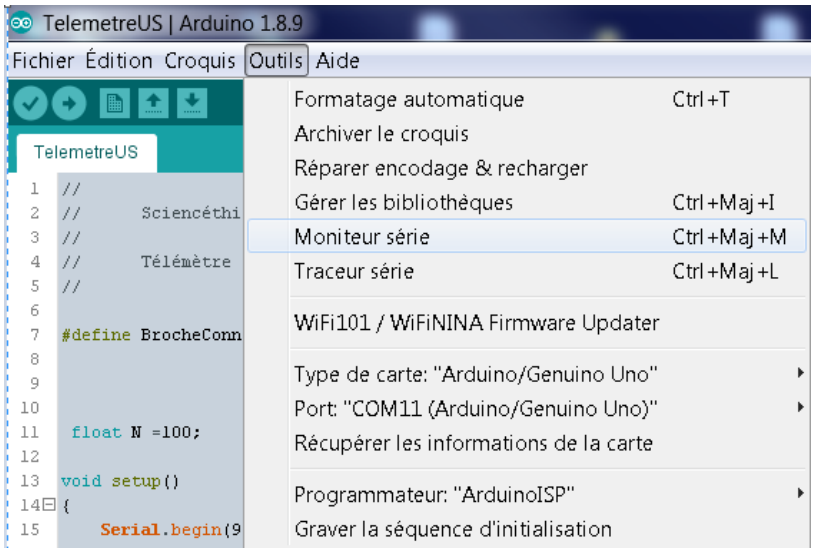
```
float distance = 0.3;   // Valeur de la distance entre l'émetteur-récepteur et l'obstacle en mètre
```

Exemple ci-dessous, en utilisant votre banc d'optique et le support de télémètre sur tige Ø10 réf. 656 015.

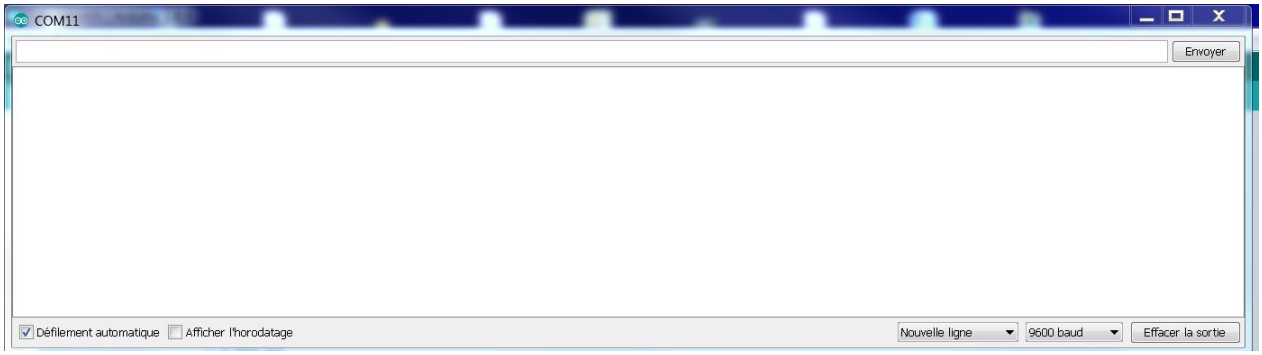


Ouvrir ensuite le moniteur série pour afficher les données demandées par le programme téléversé :

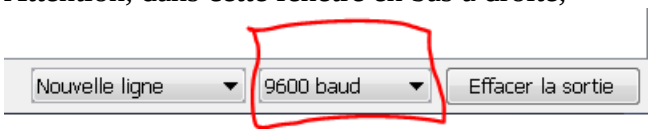
Menu/Outils/Moniteur série



La fenêtre vide ci-dessous s'affiche :



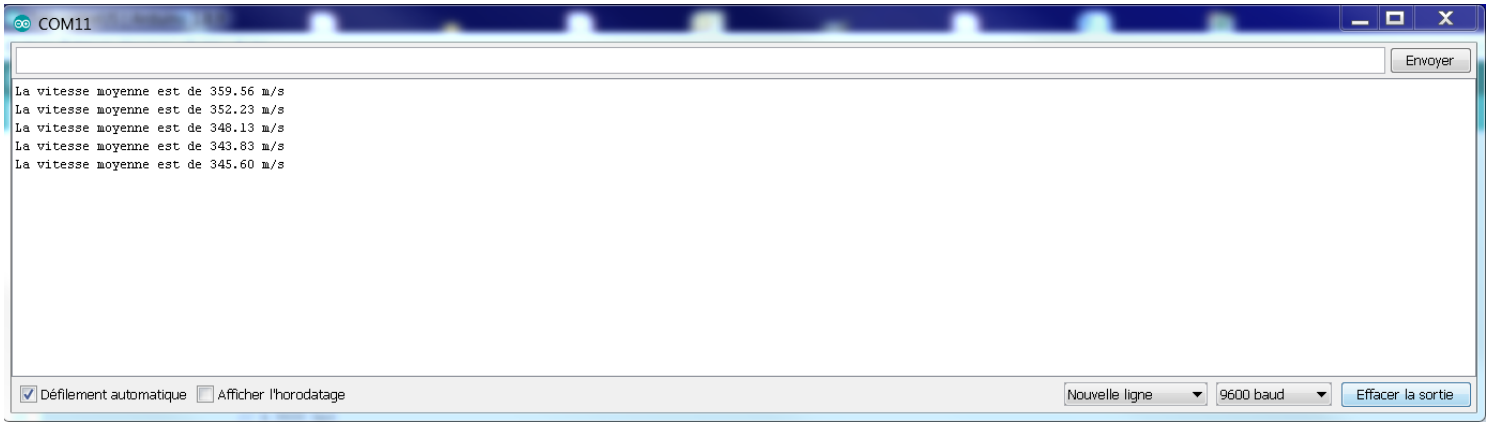
Attention, dans cette fenêtre en bas à droite,



il est important de vérifier que la vitesse de communication en baud (bits/seconde ou bps) soit bien paramétrée à 9600, comme dans le code du programme lignes 15 et 16 :

```
15  Serial.begin(9600);    // Initialise la vitesse de communication entre la carte Plug'Uino et le PC
16                         // à 9600 bps
```

Au bout de quelques secondes le résultat de l'exécution du programme « téléversé » dans le microcontrôleur, s'affiche comme ci-dessous :



Il est également possible de copier-coller ces informations dans un autre logiciel de traitement (tableur, Word etc…)

3-2 Exécuter le programme GenerateurSonSimple.ino

(Ce programme .ino est disponible dans votre « Espace Enseignant », en téléchargement gratuit sur www/sciencethic.com)

GenerateurSonSimple | Arduino 1.8.9

Fichier Édition Croquis Outils Aide

GenerateurSonSimple

```
1  //
2  //      Sciencéthic
3  //      Générateur de son
4  //      Version simple
5  //
6
7  #define SortieSon 7        // Définit la broche de connexion du module jack femelle
8                             // Ici ce dernier doit être connecté sur la broche D7
9                             // Vous pouvez changer ce nombre par exemple 2 pour connecté le module
10                            // sur la broche D2
11 float frequence=440;        // La variable "frequence" correspond à la valeur de la fréquence
12                            // qu'on veut émettre, en Hz.
13 float periode=1/frequence;// "periode" contient la valeur de la période associée, en s.
14
15
16 void setup()               // Initialisation du programme
17 {
18     pinMode(SortieSon, OUTPUT); // Déclare la broche de connexion SortieSon comme étant une sortie
19
20     }
21
22 void loop()                // Boucle principale
23 {
24   {
25   digitalWrite(SortieSon,HIGH);               // Met la SortieSon à l'état haut
26   delayMicroseconds(periode/2.0000*1000000);  // Attend une demi-période à l'état haut, en µs
27   digitalWrite(SortieSon,LOW);                // Met la SortieSon à l'état haut
28   delayMicroseconds(periode/2.0000*1000000);  // Attend une demi-période à l'état haut, en µs
29   }
30 }
```

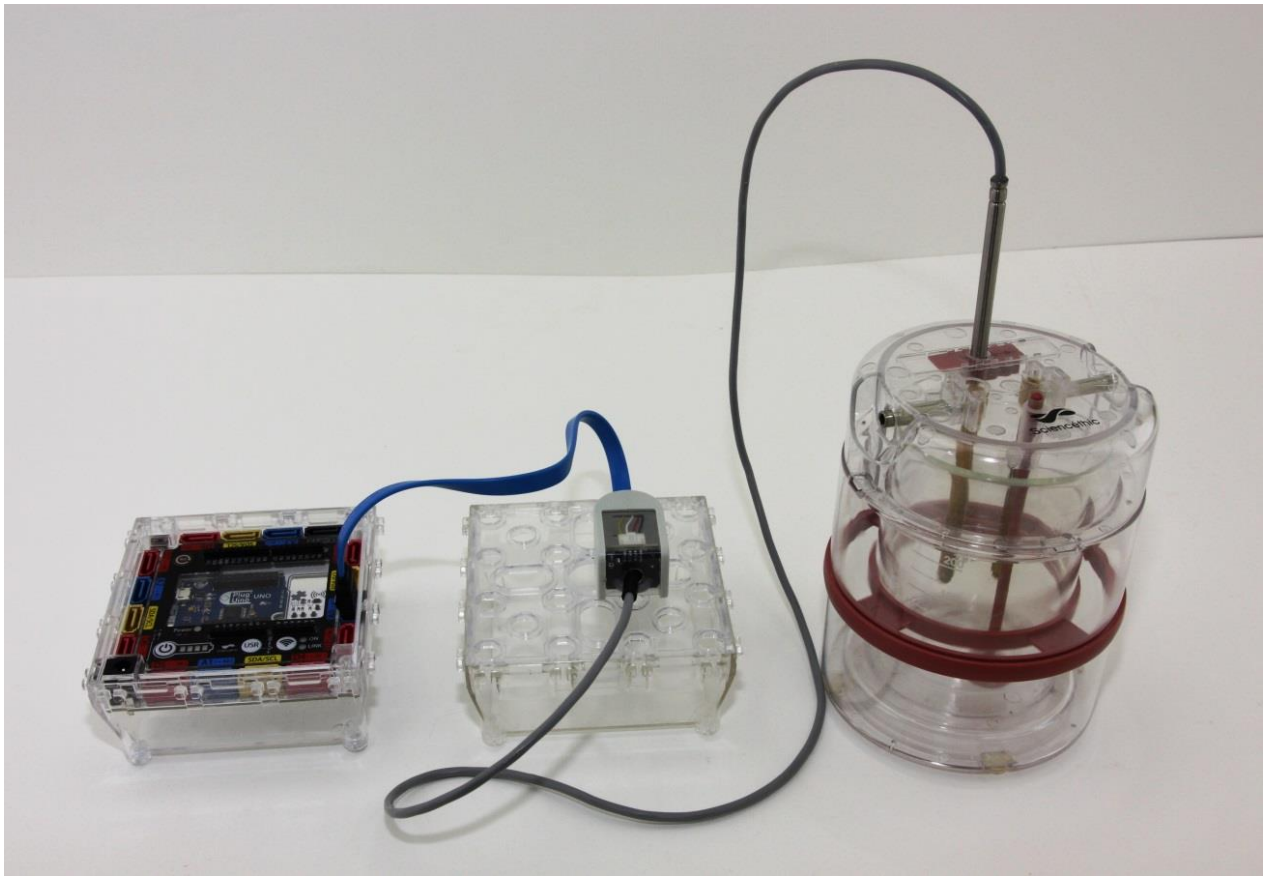
Ce programme permet de générer un signal périodique à une fréquence définie à 440 Hz (modifiable dans le code du programme ligne 11) sur la broche D7 (définie à la ligne 7 du programme) du microcontrôleur Plug'Uino® Uno.
Sur cette broche D7, définie en sortie dans le programme (ligne18), on branche le connecteur Plug'Uino® SATA- JACK FEMELLE Ø3.5 mm.
Pour « écouter » le signal périodique généré, il convient de brancher des écouteurs équipés d'une prise jack mâle 3.5 mm et d'un réglage de volume.
Par mesure d'hygiène, il est préférable de demander aux élèves d'utiliser leurs écouteurs de smart phone personnels.

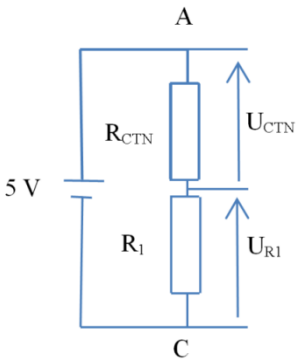
Montage à réaliser, le connecteur SATA-JACK Femelle étant branché sur la broche D7 du microcontrôleur.

3-3 Exécuter le programme ScEthic_ThermometreNumerique_CTN_R_V3.ino

(Ce programme .ino est disponible dans votre « Espace Enseignant », en téléchargement gratuit sur www/sciencethic.com)

A l'aide d'un câble SATA, brancher la sonde CTN Plug'Uino® sur l'entrée analogique A0 du microcontrôleur.





Le montage réalisé dans la sonde CTN Plug'Uino® est un pont diviseur :

U_{AC} est la tension de référence du microcontrôleur Arduino™ Uno, c'est-à-dire 5 V. La tension en sortie de la sonde CTN Plug'Uino®, qui sera mesurée par le microcontrôleur, est U_{R1}, la tension aux bornes de la résistance R_1

On a $U_{AC} = 5 = (R_1 + R_{CTN}) * I$

et $U_{R1} = R_1 * I$

donc $U_{R1} = \frac{5R_1}{R_{CTN}+R_1}$

On peut donc obtenir la valeur de la résistance R_{CTN} de la CTN à partir de la mesure de U_{R_1}, $R_{CTN} = \frac{R_1(5-U_{R1})}{U_{R1}}$

et en déduire la valeur de la température à partir de la courbe de réponse $T = f(R_{CTN})$ déjà définie par la formule T = -26,04 ln(R_{CTN}) + 265,26 .

Vous retrouverez donc ces fonctions dans le script Arduino pour effectuer tous les calculs à partir de la mesure de la tension aux bornes de la résistance R_1 , U_{R1}.

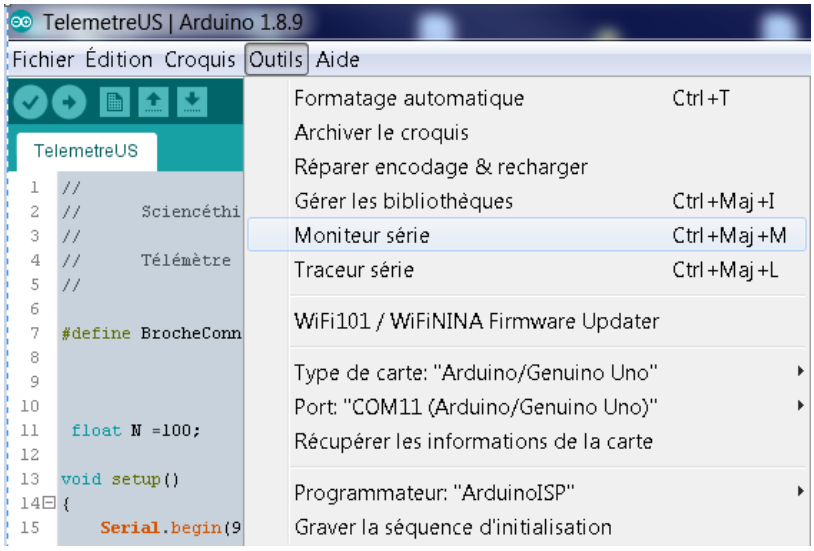
Fichier Édition Croquis Outils Aide

ScEthic_ThermometreNumerique_CTN_R_V3

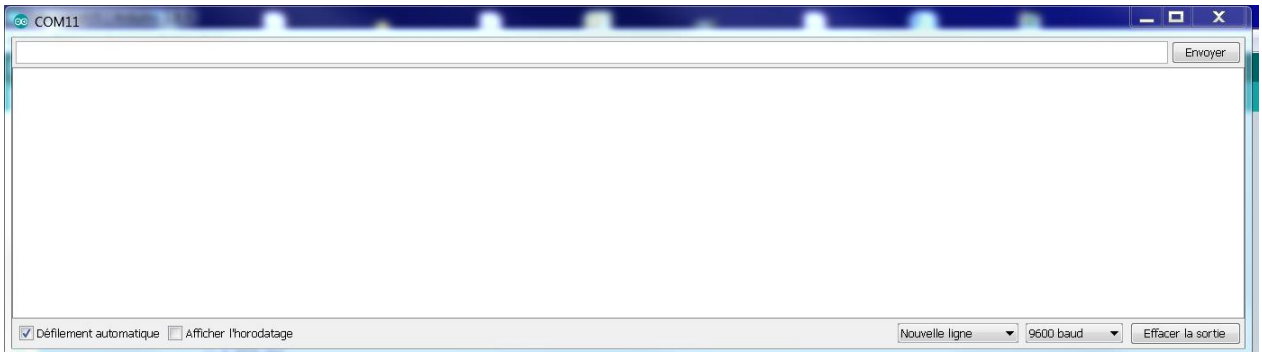
```
//
//      Sciencéthic
//
//      Thermomètre numérique par étalonnage de résistance CTN
//

float N;                    // Définit un nombre flottant N
float T;                    // Définit un nombre flottant T
float R1 = 10000.0000;      // Définit la valeur de R1 la résistance de 10 kOhms

#define BrocheConnexion A0  // Définit la broche sur laquelle sera connectée la sonde de température CTN
                            // Ici, elle sera reliée à l'entrée A0

void setup() {              // Initialisation du programme
  Serial.begin(9600);       // Initialise la vitesse de communication entre la carte Plug'Uino et le PC
                            // à 9600 bps
}

void loop() {               // Boucle principale
 N = analogRead(BrocheConnexion);  // Met dans N la mesure faite sur l'entrée BrocheConnexion
float Ur1 = 5.000000*N/1023.000000; // Comvertit le nombre N en tension Ur mesurée sur BrocheConnexion
float R = R1*(5-Ur1)/Ur1;       // Calcule la résistance de la CTN à partir de la valeur de Ur1
 T =-26.04*log(R)+265.26;
                            //  La ligne 23 calcule la valeur de la température qui correspond à R
                            // à partir de la courbe d'étalonnage
                            // Remarque : dans Arduino, la fonction logarithme néperien se note log
                            // et le logarithme décimal log10
 Serial.print("Température : "); // Affiche "Température :"
 Serial.print(T);           // Affiche la valeur de la température calculée
 Serial.println(" °C");     // Affiche l'unité et passe à la ligne
 delay(1000);               // Attend une seconde avant de boucler
}
```

Téléverser le programme dans me microcontrôleur puis ouvrir ensuite le moniteur série pour afficher les données demandées par le programme téléversé :

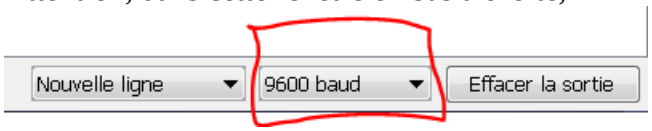
Menu/Outils/Moniteur série



La fenêtre vide ci-dessous s'affiche :

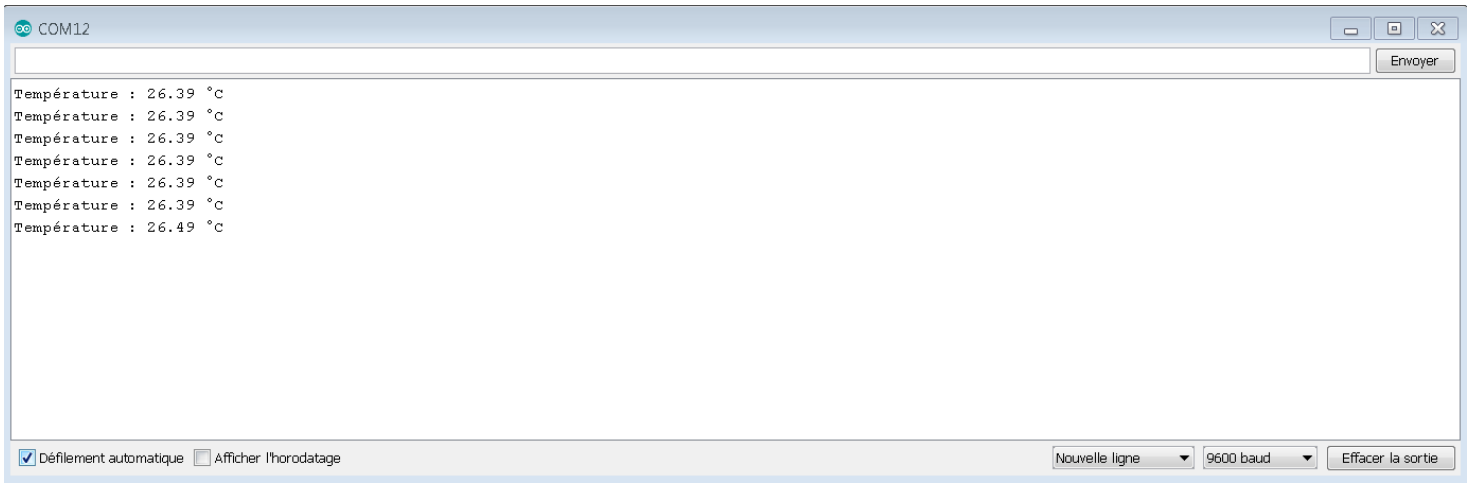


Attention, dans cette fenêtre en bas à droite,



il est important de vérifier que la vitesse de communication en baud (bits/seconde ou bps) soit bien paramétrée à 9600, comme dans le code du programme lignes 15 et 16 :

La température s'affiche alors toutes les secondes :



3-3 Exécuter le programme Loi_de_Mariotte.ino

(Ce programme .ino est disponible dans votre « Espace Enseignant », en téléchargement gratuit sur www/sciencethic.com)

Le capteur de pression Plug'Uino® réf. 651055 délivre une tension analogique 0 à 5 V proportionnelle **à la différence de pression** entre la pression atmosphérique et la pression mesurée sur la tétine, sur une gamme de pression de -1000 à +2000 hPa par rapport à la pression atmosphérique.

L'entrée analogique A0 du microcontrôleur sur laquelle est branché le capteur de pression, mesure des tensions analogiques comprises entre 0 et 5 volts et les numérise sur 10 bits (soit 1024 valeurs possibles). La conversion de cette valeur en une pression peut se faire à partir de la loi affine modélisant le capteur (pression=a*mesure+b) ; elle peut aussi être réalisée par étalonnage à partir de mesures données par un pressiomètre.

Le programme Loi_de_Mariotte.ino ci-dessous, retourne la valeur moyenne sur 20 mesures, de la tension numérisée sur l'entrée A0.


```
//
//      Sciencéthic
//      Loi de Mariotte
//      Mesure de pression



void setup()              // Initialisation du programme
{
  Serial.begin(9600);     // Initialise la vitesse de communication entre la carte Plug'Uino et le PC
                          // à 9600 bps

}
void loop()               // Boucle principale repétée à l'infini
{
  int mesure = 0;         // initialisation de la variable mesure
  int total_mesures = 0;  // initialisation de la variable somme des mesures
  for (int compteur = 0; compteur < 20; compteur = compteur + 1) //boucle effectuant 20 mesures
  {                                                              //séparées de 100 ms
    mesure = analogRead(A0);                        // enregistrement de la mesure
    total_mesures = mesure + total_mesures;         // cumul des mesures
    delay(500);                                     //délai entre deux mesures;
  }
  Serial.print("moyenne pour 20 mesures : ");
  Serial.println(total_mesures / 20);               // calcul de la valeur moyenne et affichage
}
```

A l'aide d'un câble SATA, brancher le capteur de pression Plug'Uino® sur l'entrée analogique A0 du microcontrôleur et connecter une seringue.

.