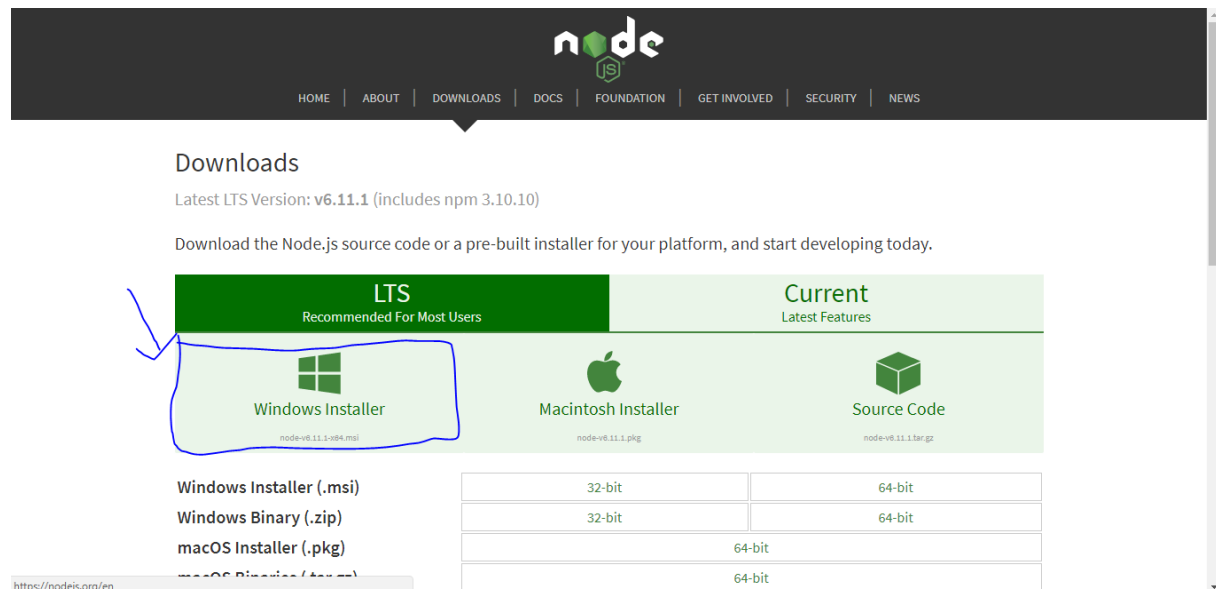


C'est vendredi soir je n'ai rien à faire, donc j'ai pensé à faire un petit tuto sur la communication série COM avec une carte Arduino Uno en node.js, l'objectif est de lire une valeur analogique puis la transférer sur mon PC avec la COM série puis afficher le résultat dans un navigateur internet, stop au bavardage ;)

Tout d'abord préparons l'ordinateur, on doit installer l'environnement Node.js et son gestionnaire de paquets npm, pour cela on se rend sur le site internet de node.js et nous allons télécharger la dernière version qui correspond à votre système d'exploitation, pour moi c'est Windows 64 bits ☺

<https://nodejs.org/en/download/>



Après avoir téléchargé node.js, on passe à l'installation, un simple clic sur le fichier téléchargé, puis next next ..., puis finish, pour vérifier que node.js et son gestionnaire de paquet sont bien installés, il suffit de lancer cmd, puis tapez les commandes suivantes,

Pour Node.js

```
C:\Windows\system32\cmd.exe
Microsoft Windows [version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. Tous droits réservés.

C:\Users\nassin>node --version
v6.11.1

C:\Users\nassin>
```

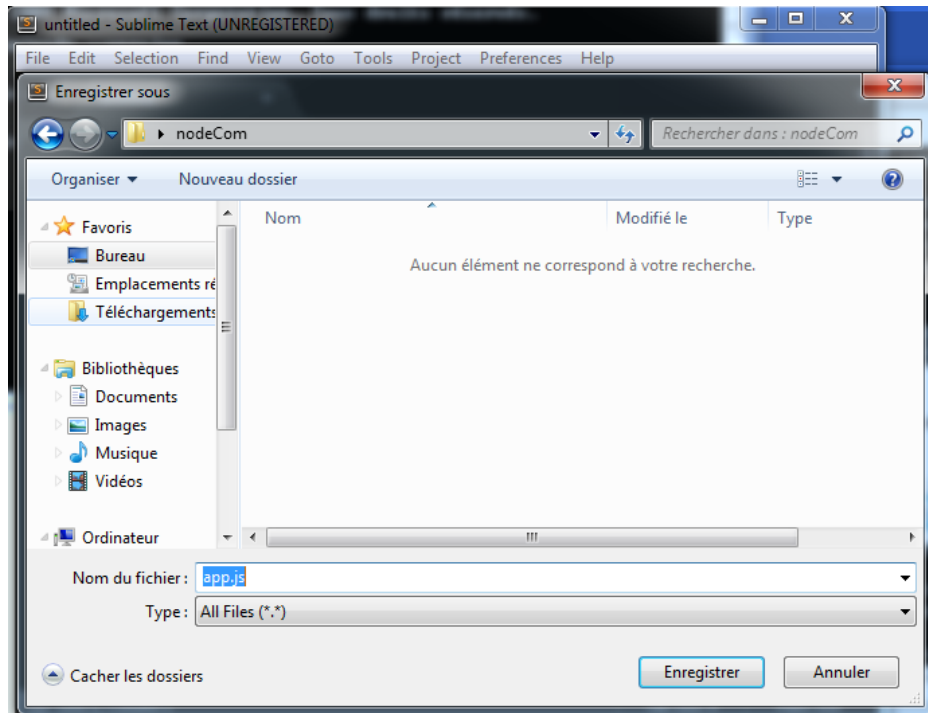
Pour gestionnaire de paquets npm

```
C:\Users\nassin>npm --version
3.10.10

C:\Users\nassin>
```

Maintenant que le node.js et npm sont bien installés nous passons à la programmation, vous avez besoin seulement d'un éditeur de texte, moi je vais utiliser sublime text

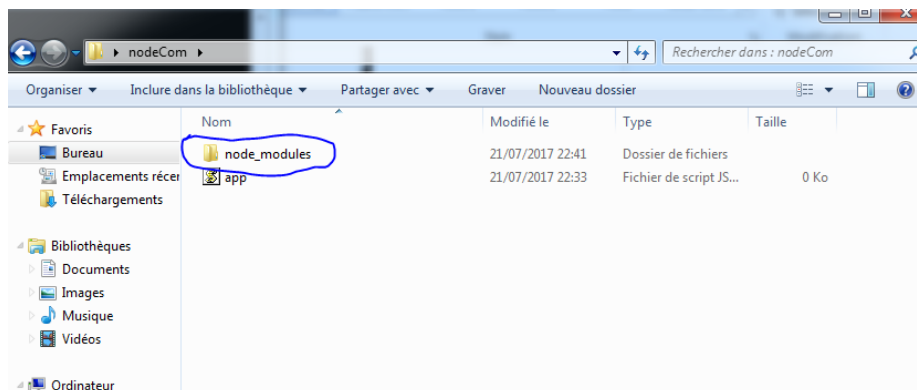
Je vais créer un répertoire où je stock tous mes fichiers de projet, dans le bureau 😊, je le nomme « nodeCom », dans ce répertoire j'enregistre un fichier sous le nom app.js, dans ce dernier nous allons écrire le code.



Pour ce projet nous allons utiliser une communication série, donc on va installer le module serialport, dans cmd mettez-vous dans le répertoire nodeCom puis taper la commande

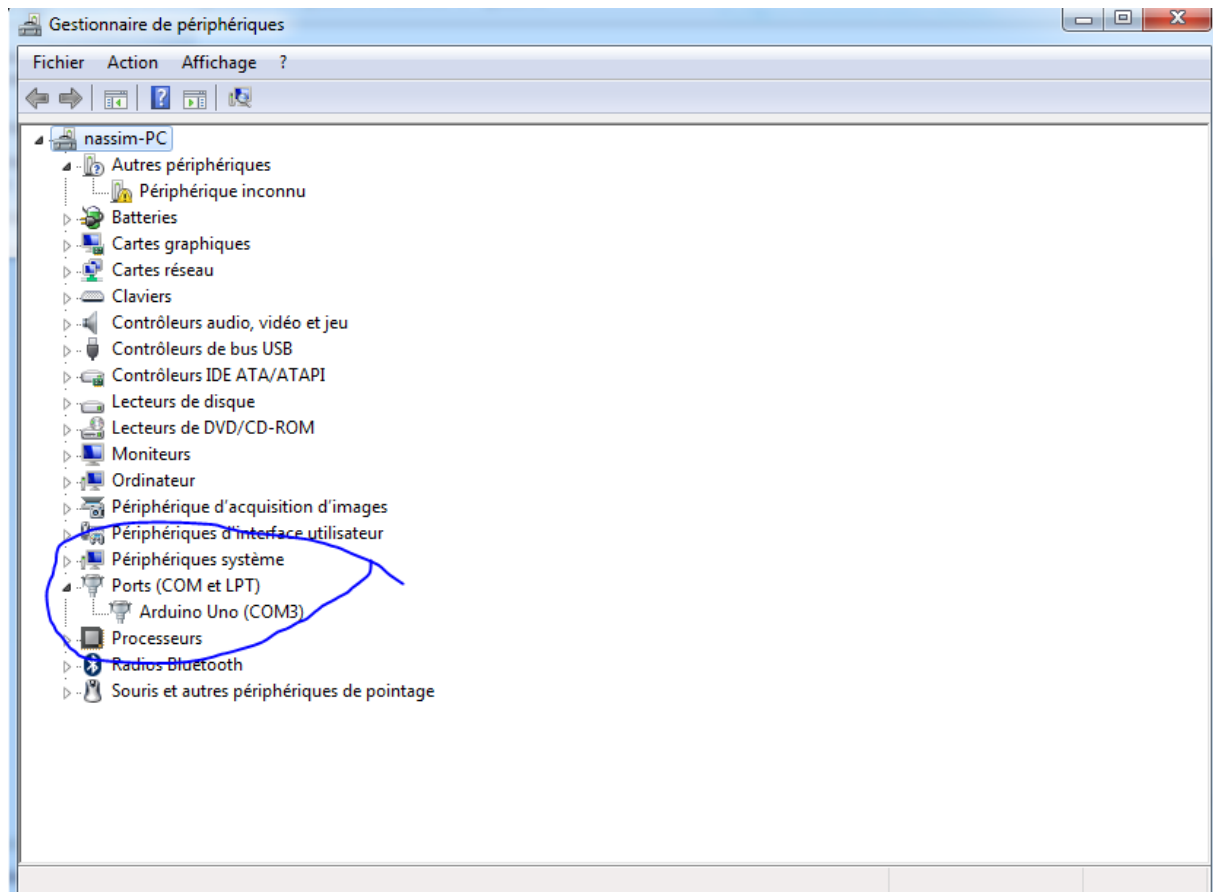
```
npm install serialport
```

À la fin de l'installation vous devez avoir un répertoire node\_module dans votre répertoire projet (nodeCom)



Maintenant le module est installé brancher votre carte arduino avec un câble USB, il faudra installer le driver de la carte, pour cela il suffit d'installer IDE arduino, pendant son installation une boîte de dialogue vous propose d'installer le driver.

Après l'installation et le branchement de la carte vous devez détecter la carte pour vérifier ça, rendez-vous dans le gestionnaire de périphériques vous devez avoir un truc comme ça



Dans mon cas c'est COM3, pour vous peut-être différent.

Maintenant que la carte est détectée nous allons ouvrir une communication avec cette dernière avec le code suivant

```
var SerialPort = require('serialport');

var port = new SerialPort('COM3',{baudRate: 115200}, function (err) {
  if (err) {
    return console.log('Error: ', err.message);
  }
});
```

La 1<sup>ère</sup> instruction permet d'importer le module serialport, la 2<sup>ème</sup> d'ouvrir le port, la fonction en paramètre est une fonction callback elle sera appelée dans le cas on aura une erreur durant l'ouverture du port, quant au 2<sup>ème</sup> paramètre c'est la vitesse de communication.

Maintenant ajoutons un code qui nous permet de rester en écoute dès que des données sont dans le buffer, un événement se déclenche et une fonction callback est appelée

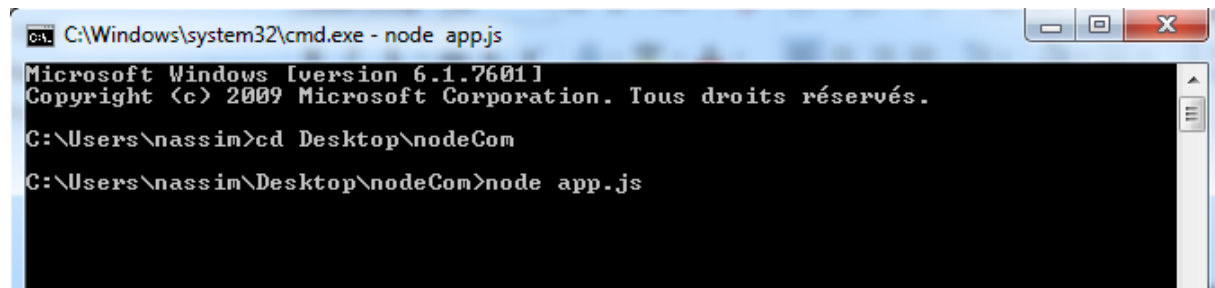
```

1  var SerialPort = require('serialport');
2  var Readline = SerialPort.parsers.Readline;
3  var port = new SerialPort('COM3',{baudRate: 115200}, function (err) {
4      if (err) {
5          return console.log('Error: ', err.message);
6      }
7  });
8
9  var parser = port.pipe(new Readline());
10
11 parser.on('data', console.log);

```

La fonction callback est appelée dès qu'une donnée est disponible dans le buffer, et se charge d'afficher ce qu'on a reçu dans la console. J'ai utilisé un objet Readline avec un pipe afin de convertir la valeur en string.

A ce point vous pouvez tester le code, pour cela on se rend dans cmd, on se déplace dans le dossier nodeCom puis entrez la commande suivante



```

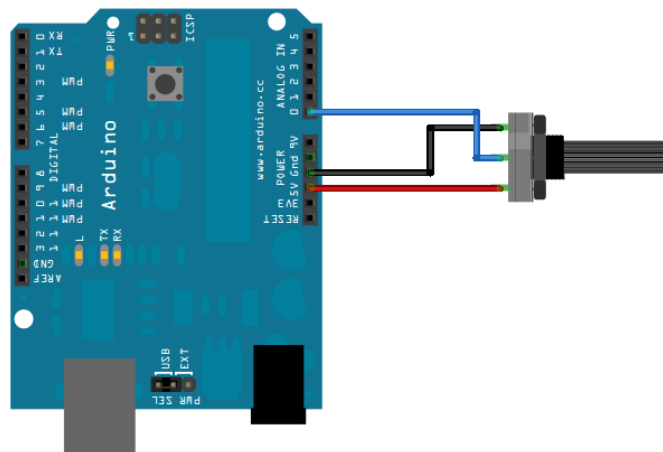
C:\Windows\system32\cmd.exe - node app.js
Microsoft Windows [version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. Tous droits réservés.

C:\Users\nassin>cd Desktop\nodeCom
C:\Users\nassin\Desktop\nodeCom>node app.js

```

Rien ne s'affiche, c'est tout à fait normal car nous avons rien reçu, pour cela faudra programmer l'arduino pour qu'elle nous envoie des données ;)

Pour le montage nous aurons besoin d'un potentiomètre afin de faire varier la valeur.



C'est un code simple qui se charge de lire une entrée analogique (A0) chaque 200 ms, et envoie la valeur au PC

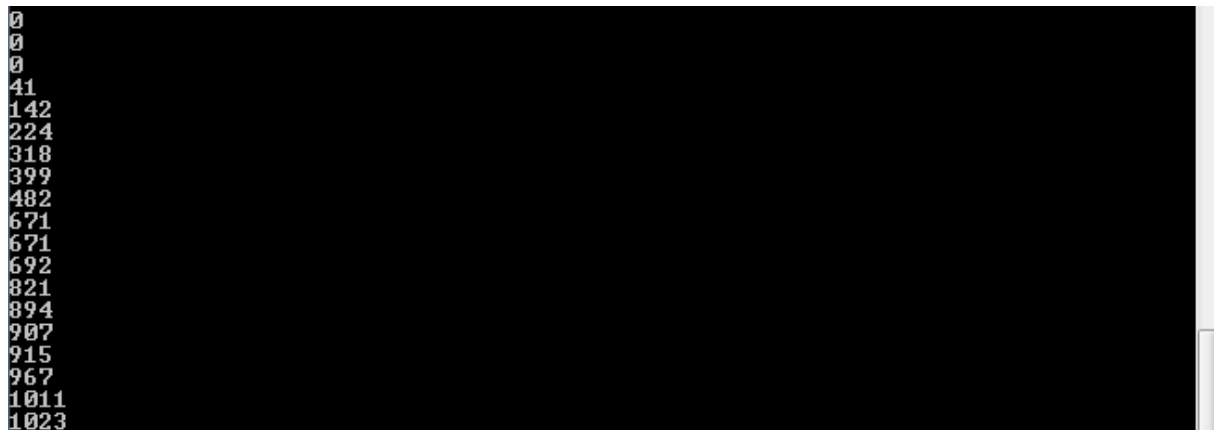
```

int analogPin = 0;    // potentiometer wiper (middle terminal) connected to analog pin 3
int val = 0;          // variable to store the value read
String thisString;
void setup() {
  Serial.begin(115200);    // setup serial
}

void loop() {
  val = analogRead(analogPin);    // read the input pin
  thisString = String(val);
  Serial.println(val);            // send value to PC
  delay(200);
}

```

Résultat après avoir lancé app.js et en variant la tension de l'entrée ANO.



Nous avons réussi à transmettre une valeur de l'Arduino au PC, Maintenant nous allons utiliser un navigateur internet pour afficher la valeur en temps réel c.à.d la valeur elle se mettra à jour dès qu'une nouvelle valeur est disponible.

Tout d'abord nous allons créer un serveur http, vous allez voir c'est très simple avec node.js, pour cela nous allons utiliser le module http, pour la communication en temps réel on utilisera le module socket.io.

Installation des modules

```
npm install express
```

```
npm install socket.io
```

```
npm install fs
```

Le module fs nous permet de lire un fichier html et l'envoyer au client.

L'objectif de socket est de garder une communication ouverte entre le serveur et le client.

Donc il y a deux tâches à faire, créer le programme qui tourne sur le serveur, il est en écoute sur le port 80 (requêtes http) et y répond, la 2<sup>ème</sup> tâche le programme client c'est un simple fichier html avec un script (javascript) pour communiquer en utilisant les socket.io, on peut résumer ça comme suit

- 1- Le client demande la page html en tapant l'adresse du serveur dans notre cas c'est localhost ou 127.0.0.1.
- 2- Le serveur reçoit la requête du client et répond avec la page html et garde une communication ouverte avec ce client grâce au socket.io.
- 3- Dès qu'une nouvelle valeur est disponible dans le buffer, elle est envoyée aux clients.
- 4- Les clients mettent à jour leurs affichages.

Intéressons-nous au serveur, Tout d'abord on créera un serveur http qui est en écoute sur le port 80, puis un socket créé afin de garder la communication avec les clients.

```
1
2
3
4 var app = require('http').createServer(handler)
5 var io = require('socket.io')(app);
6 var fs = require('fs');
7
8 app.listen(80);
9
10 function handler (req, res) {
11   fs.readFile(__dirname + '/index.html',
12     function (err, data) {
13       if (err) {
14         res.writeHead(500);
15         return res.end('Error loading index.html');
16       }
17
18       res.writeHead(200);
19       res.end(data);
20     });
21 }
22
23
24 var SerialPort = require('serialport');
25 var Readline = SerialPort.parsers.Readline;
26
27 var port = new SerialPort('COM3',{baudRate: 115200}, function (err) {
28   if (err) {
29     return console.log('Error: ', err.message);
30   }
31 });
32
33 var parser = port.pipe(new Readline());
34 parser.on('data', console.log);
35
36
```

Pour ce qui concerne le client Dans le dossier « nodeCom » nous allons créer un fichier et nommer le « index.html » qui contiendra le code suivant

```

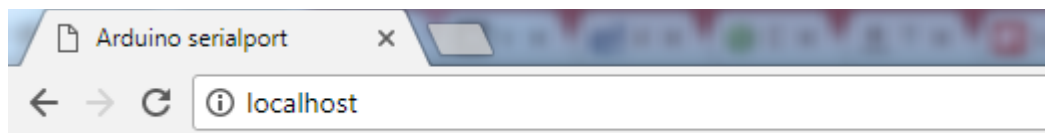
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Arduino serialport</title>
  </head>
  <body>
    <div>
      <p>valeur analogique <span id="val">0</span></p>
    </div>
  </body>

  <script src="/socket.io/socket.io.js"></script>
  <script>
    var socket = io('http://localhost');
    socket.on('getValue', function (data) {
      console.log(data);
      document.getElementById('val').innerHTML=data;
    });
  </script>
</html>

```

C'est une page html tout à fait ordinaire, la partie la plus intéressante c'est le script, un socket est instancié en indiquant en paramètre l'adresse du serveur, puis on demande de rester en écoute pour ce serveur dès qu'on a reçu une donnée du serveur on récupère un élément html dont son id=val et on change son texte avec la valeur qu'on a reçu.

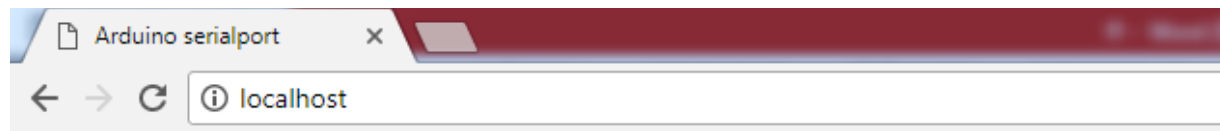
Maintenant lancez votre navigateur préféré en tapant dans la barre d'adresse localhost ou 127.0.0.1 , faites varier la valeur de tension vous allez voir que la valeur change dans votre navigateur.



valeur analogique 1023



valeur analogique 507



valeur analogique 0

J'ai pris des captures d'écran pour 3 valeurs 😊 mais ça marche très bien.

Voilà, j'espère que ce tuto vous a plu n'hésitez pas à me contacter si besoin 😊

[n.ouaret91@gmail.com](mailto:n.ouaret91@gmail.com)