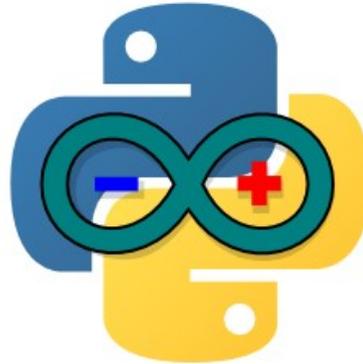


## Introduction à la librairie Pyduino :

**Programmer un mini-PC aussi facilement qu'une carte Arduino... et bien plus encore !**



# Ateliers Pyduino

par X. HINAULT

[www.mon-club-elec.fr](http://www.mon-club-elec.fr)

**www.mon-club-elec.fr**



Tous droits réservés – 2013.

### **Document gratuit.**

Ce support PDF d'atelier Arduino vous est offert.

Pour acheter d'autres supports d'ateliers Arduino, rendez-vous ici :

[http://www.mon-club-elec.fr/pmwiki\\_mon\\_club\\_elec/pmwiki.php?n=MAIN.ATELIERSPYDUINO](http://www.mon-club-elec.fr/pmwiki_mon_club_elec/pmwiki.php?n=MAIN.ATELIERSPYDUINO)

Vous avez constaté une erreur ? une coquille ? N'hésitez pas à nous le signaler à cette adresse : [support@mon-club-elec.fr](mailto:support@mon-club-elec.fr)

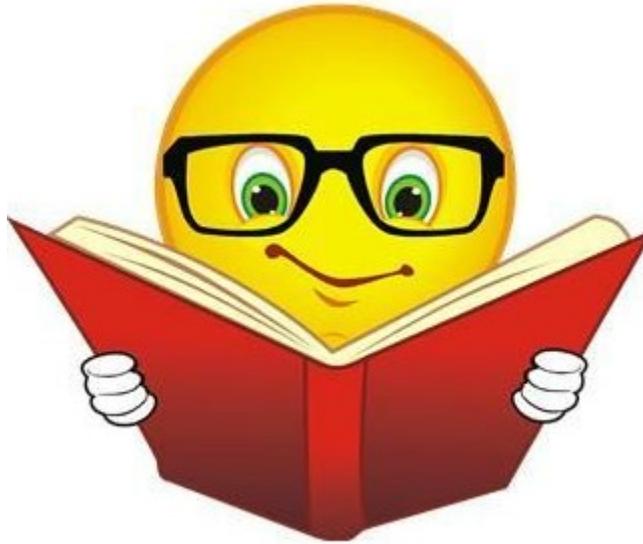
**Truc d'utilisation : visualiser ce document en mode diaporama dans le visionneur PDF. Navigation avec les flèches HAUT / BAS ou la souris.**

**En mode fenêtre, activer le panneau latéral vous facilitera la navigation dans le document. Bonne lecture !**

**Lancer également le logiciel Geany afin de pouvoir tester au fur et à mesure les codes d'exemples !**

## 1. *Intro*

L'objectif ici est de donner une rapide introduction à la librairie Pyduino afin de comprendre à quoi une telle librairie peut servir.



Prêt ? C'est parti !

## 2. Pyduino : c'est quoi ? Pourquoi faire ?

### Le point de départ

Arduino est apprécié pour plusieurs raisons importantes :

- un langage simple à prendre en main
- un accès simplifié aux ressources matérielles (une broche est désignée par un numéro)
- une connectique simple et standard
- un espace de développement clé en main
- des fonctions implémentées pratiques
- nombreuses possibilités par librairies
- etc...



### L'arrivée des mini-PC avec broches E/S

Depuis 1 an ½, arrivent de nouvelles plateformes, véritables mini-PC embarqués qui disposent d'un bornier E/S à la façon Arduino, voire même des broches analogiques, etc.. J'ai cité le Raspberry Pi, le pcduino, la cubieBoard, etc...

On se retrouve donc :

- avec un système qui dispose des fonctionnalités d'un PC (son, webcam, fichier, réseau) et même un peu plus (synthèse vocale, etc...)
- qui dispose également des éléments façon Arduino...



### Le problème

Chaque mini-PC a ses propres caractéristiques matérielles permettant d'accéder aux broches E/S...et les borniers sont plus ou moins pratiques à utiliser...

Il n'est pas simple d'envisager d'écrire un code intégrant simultanément l'utilisation des fonctions systèmes et des broches E/S. Et le tout en utilisant une solution « viable » sur une plateforme aux ressources limitées.

**En un mot, avec un mini-PC, on perd la simplicité du système Arduino tant appréciée... et on peut très vite se compliquer la vie !**

### Qu'est-ce que Pyduino ?

Pyduino va permettre de réunir ces 2 univers en un seul. Pyduino est :

- une librairie que j'ai écrite en langage Python, langage léger à installer, peu gourmand en ressources, puissant
- qui implémente l'essentiel des fonctions Arduino de base
- ainsi que des fonctions permettant l'utilisation des fonctions systèmes (fichier, réseau,..) et multimédia (webcam, audio, voix..).

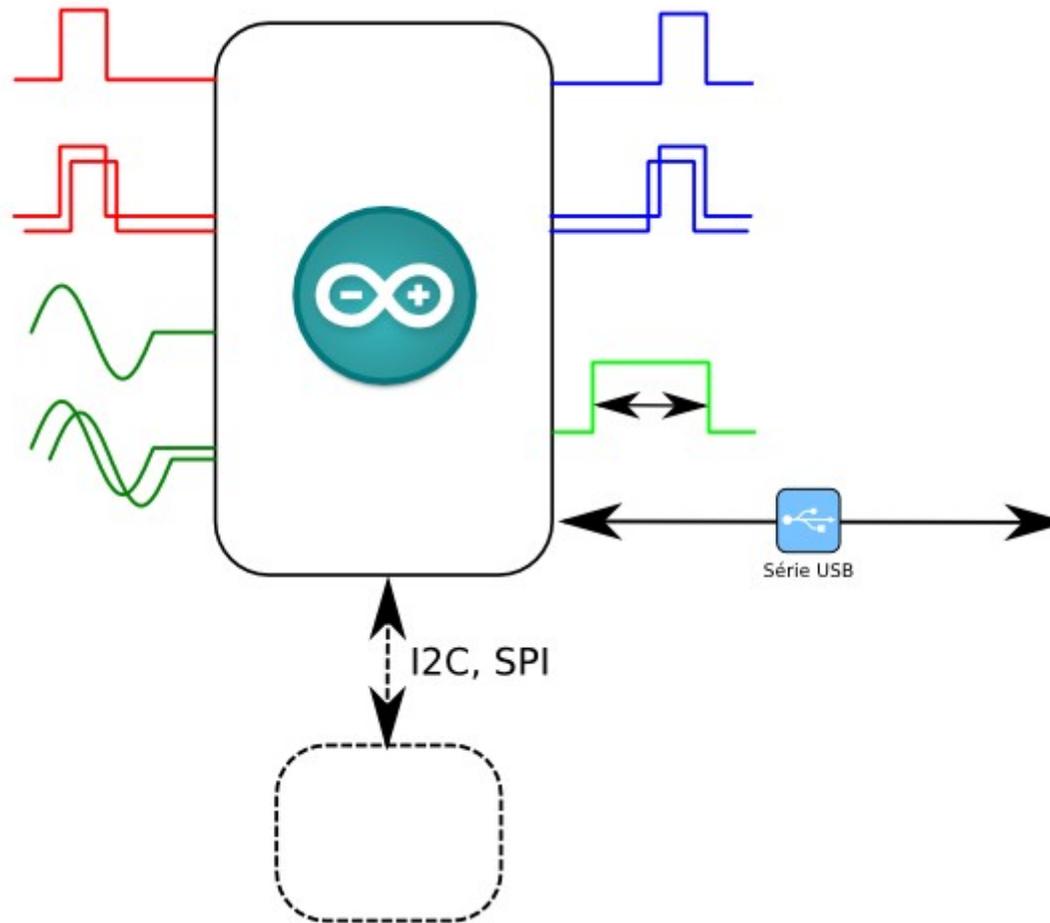


Du coup il devient possible :

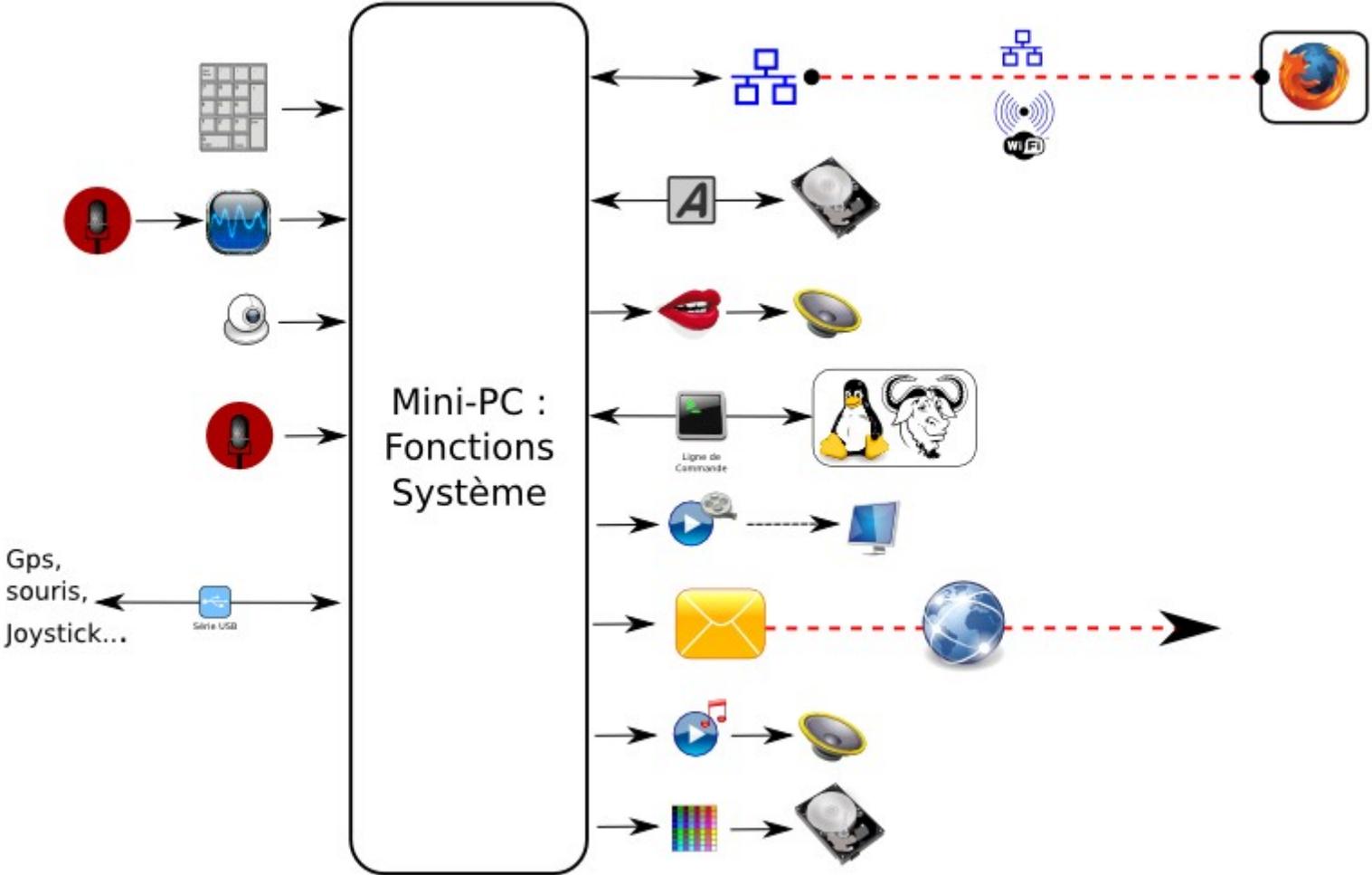
- d'écrire un code façon Arduino sur un miniPC
- tout en ayant **accès simultanément aux broches E/S et aux fonctions système ou multimédia** au sein d'un même code.
- Notamment, les fonctions d'utilisation du réseau, des fichiers, etc... sont directement accessibles.
- Pyduino intègre également des fonctions « Multimédia » : capture d'image, de son, synthèse vocale, reconnaissance vocale, etc..

```
22 # -- loop --
23 def loop():
24     digitalWrite(LED,HIGH)
25     Serial.println("La LED
26     )
27     delay(1000) # pause en
```

### 3. Ce que l'on peut faire avec Arduino

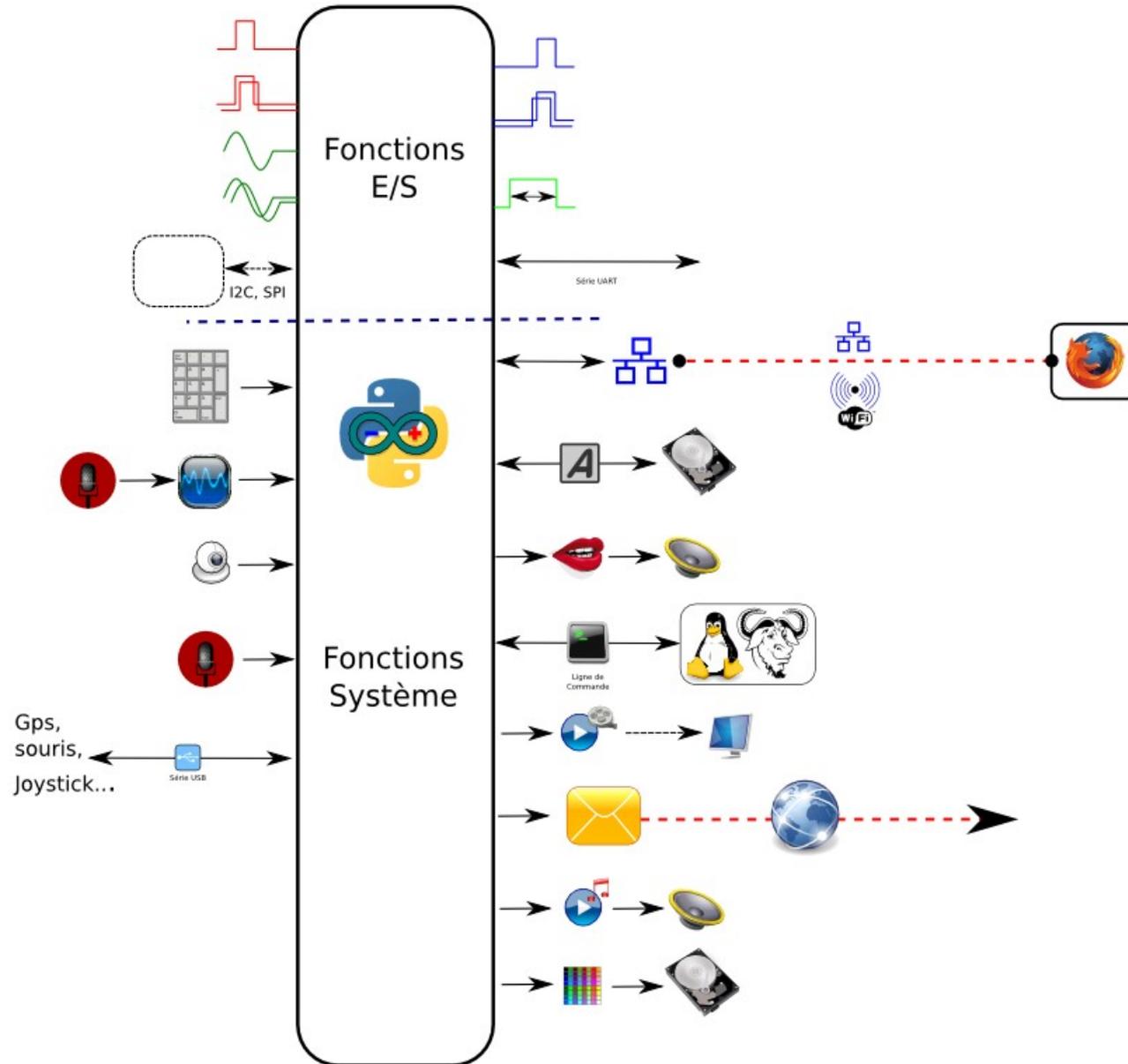


4. Ce qu'on peut faire avec un mini-PC



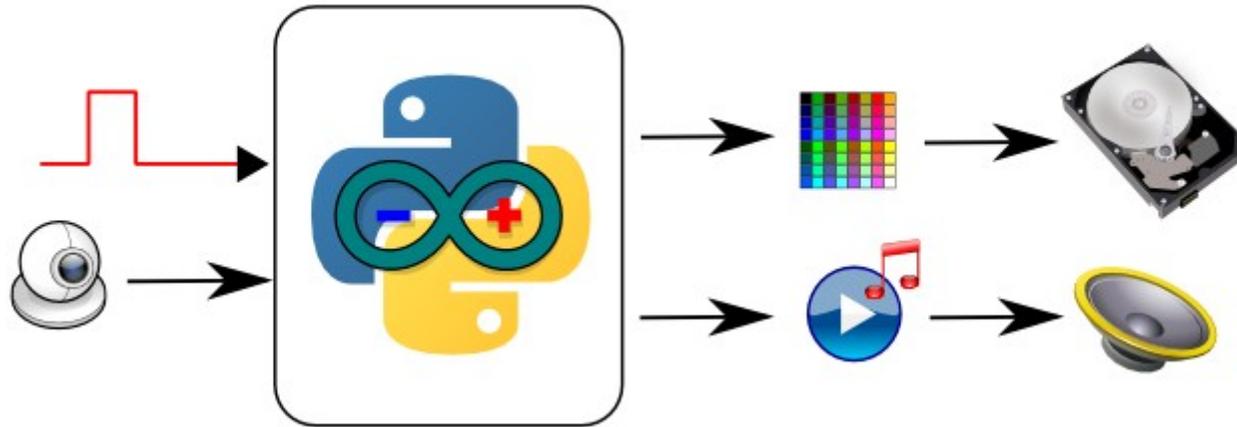
## 5. Ce qu'on peut faire avec un mini-PC et Pyduino

Pyduino permet de réunir au sein d'un même code, en style Arduino, les fonctions E/S et les fonctions système !



## 6. Exemple : l'appareil photo

Il devient ainsi possible par exemple de mixer à loisir les fonctions entre-elles, le tout dans un code compact. Par exemple, voici le schéma fonctionnel d'une application « appareil photo » :



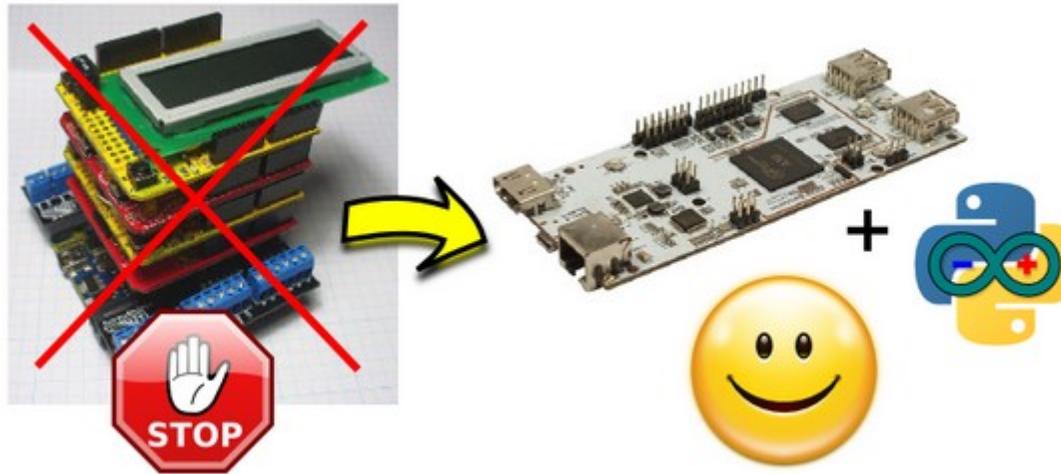
Ici :

- l'entrée numérique (un bouton poussoir)
- déclenche la capture d'une image webcam
- et la lecture d'un son de déclic photo
- l'image obtenue étant enregistrée dans un fichier image.

Le code est ici : [http://www.mon-club-elec.fr/pmwiki\\_mon\\_club\\_elec/pmwiki.php?n=MAIN.PCDUINOAppESSysImageCaptureBp](http://www.mon-club-elec.fr/pmwiki_mon_club_elec/pmwiki.php?n=MAIN.PCDUINOAppESSysImageCaptureBp)

## 7. Quand utiliser Pyduino avec un mini-PC ?

Pour tous les projets où les shields commencent à s'empiler, l'option mini-PC + Pyduino deviendra préférable :

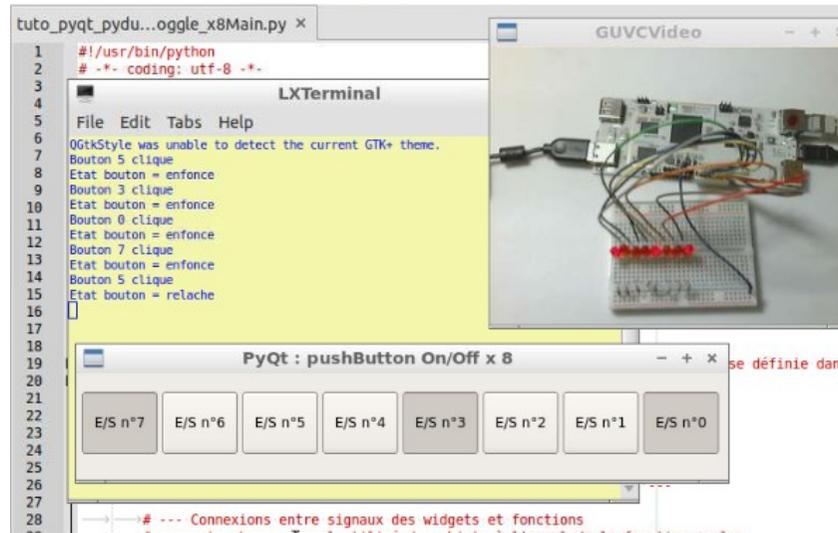


Ceci est particulièrement vrai si on souhaite utiliser des enregistrements de données, le réseau, etc..

## 8. Encore plus fort : Pyduino est utilisable directement au sein d'un code PyQt (interface graphique !)

Il devient dès lors très simple de contrôler un dispositif à partir d'une interface graphique puisque les instructions telles que `digitalWrite()` ou `analogRead()` peuvent être appelées depuis le code PyQt !

Le mini-PC étant accessible en accès VNC, il est alors facile de le contrôler à partir d'une tablette tactile par exemple.



Interface à 8 bouton graphique contrôlant 8 broches E/S .

Le code est ici : [http://www.mon-club-elec.fr/pmwiki\\_mon\\_club\\_elec/pmwiki.php?n=MAIN.PYQTLABMiniPCpDuinoSortieNumButtonAllumeLedx8](http://www.mon-club-elec.fr/pmwiki_mon_club_elec/pmwiki.php?n=MAIN.PYQTLABMiniPCpDuinoSortieNumButtonAllumeLedx8)

## 9. Pour passer à l'action... avec Pyduino : le matériel

### Pour utiliser Arduino :

On a besoin :

- d'une [carte Arduino](#)
- d'un [poste fixe](#) (netbook, portable, etc..)
- d'un [câble USB](#)

La programmation et la communication avec la carte Arduino :

- se fait à partir du logiciel Arduino, via le port USB
- la connexion au poste fixe n'est pas nécessaire en mode exécution



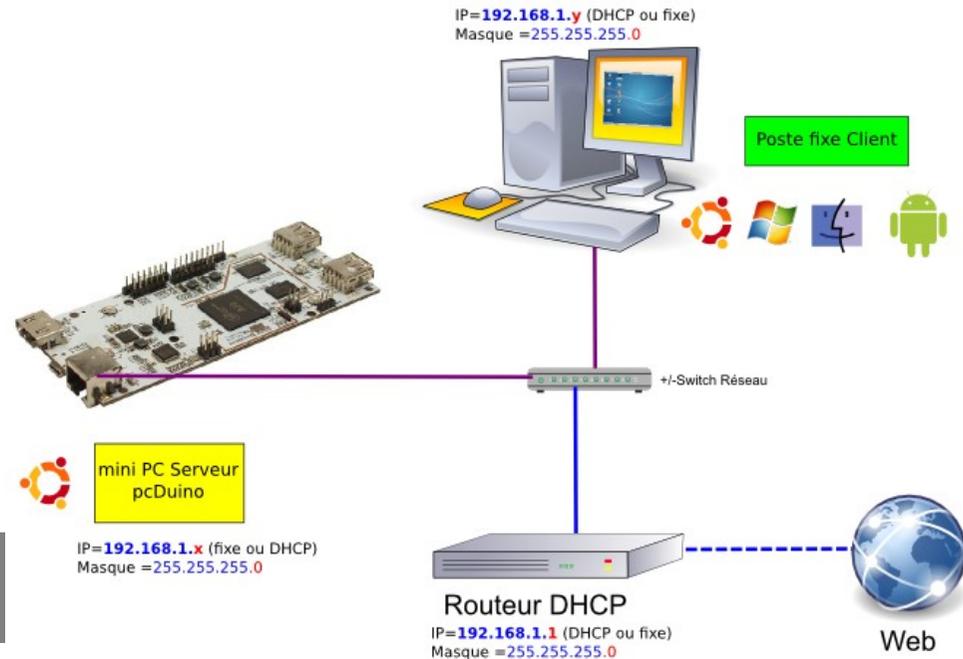
### Pour utiliser un mini-PC avec Pyduino,

On a besoin :

- d'une [carte mini-PC](#) avec un système Gnu/Linux installé opérationnel (typiquement interface graphique LXDE)
- d'un [poste fixe](#) (netbook, portable, etc..)
- d'un [routeur réseau](#) ethernet / wifi +/- switch (pas indispensable)
- de 2 [câbles ethernet](#) et/ou clé wifi

La programmation et la communication avec le miniPC :

- se fait par « accès au bureau distant » depuis le poste fixe : on programme directement dans l'éditeur Geany sur le miniPC à partir du poste fixe.
- la connexion au poste fixe n'est pas nécessaire en mode exécution



**La vraie différence consiste à utiliser un routeur en plus.**

Noter qu'un tel routeur serait aussi nécessaire avec Arduino dès lors que l'on voudrait utiliser un shield Ethernet.

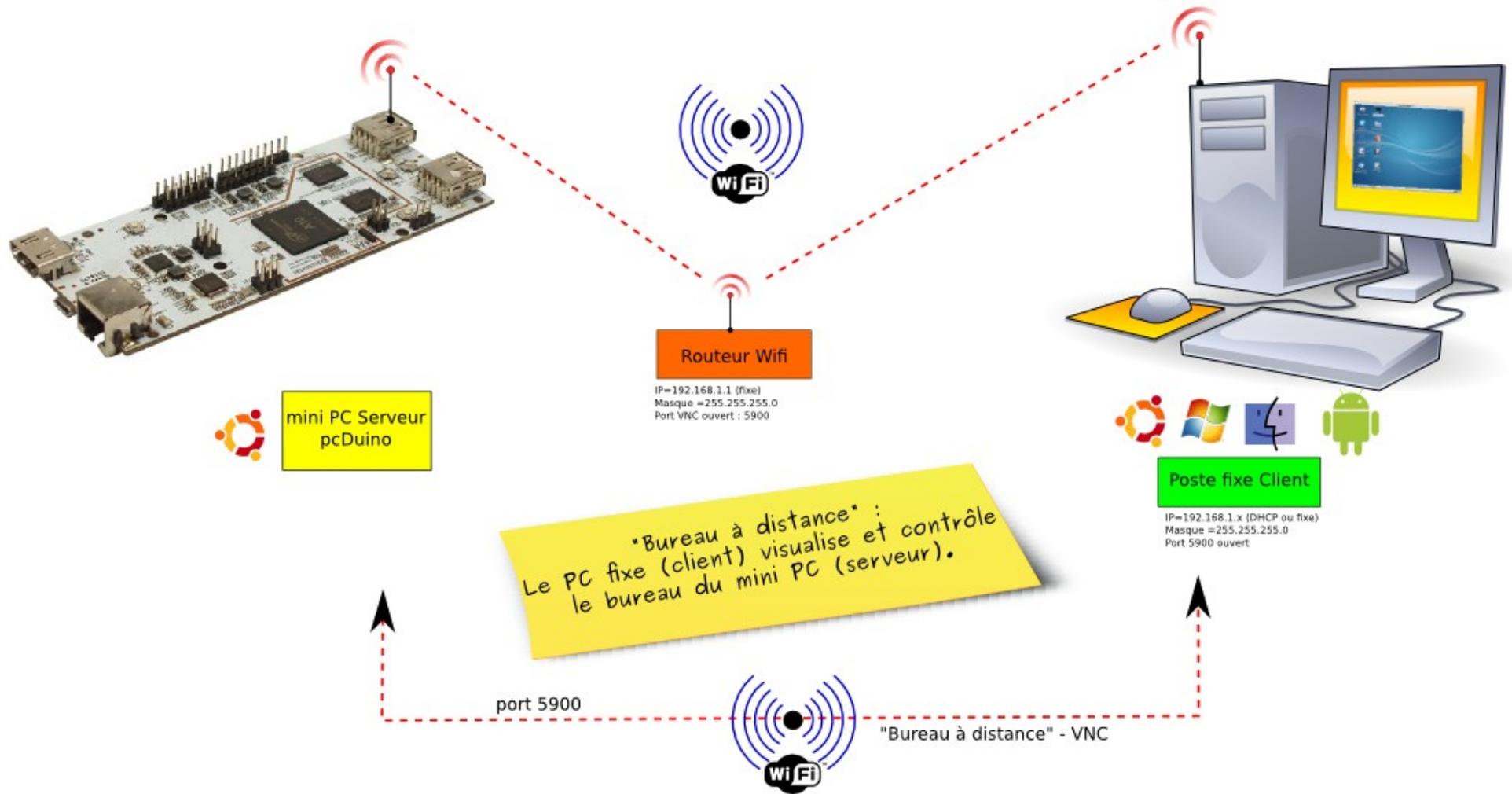
**En résumé :**

Un principe de fonctionnement relativement similaire, à la différence près que l'on communique avec le poste fixe par réseau au lieu de communiquer par USB.

## 10. Les différentes possibilités d'utiliser Pyduino

### Le réseau type : mini-PC + poste fixe sur un même réseau

- soit en ethernet
- soit en wifi
- soit un mix des 2 (le poste fixe en filaire et le mini-PC en wifi : c'est la situation potentiellement la plus courante)



## Autre possibilité d'utilisation (expérimental) : un poste fixe + arduino

- Afin de faciliter le test de la librairie Pyduino, j'ai également prévu la possibilité de d'installer Pyduino sur un poste fixe Gnu/Linux et de communiquer avec Arduino préprogrammée :
- Le principe est le suivant :
  - la carte Arduino est connectée au port USB du poste fixe Gnu/Linux
  - la carte Arduino doit être préprogrammée au préalable avec un code fournit qui « interprète » des chaînes reçues sur le port série et renvoie la résultat demandé.
  - au niveau du code Pyduino, on appelle de façon transparente les fonctions telle que analogRead(), comme si on exécutait le code sur un mini-PC



Cette solution est de type « expérimental » et présente plusieurs intérêt :  
se familiariser avec Pyduino  
utiliser les instructions Arduino directement depuis une console Python, sans utiliser l'IDE Arduino.

## 11. Installation de l'IDE Pyduino et de la librairie Pyduino à partir d'un paquet \*.deb

Tout est expliqué ici en détail : [http://www.mon-club-elec.fr/pmwiki\\_reference\\_pyduino/pmwiki.php?n=Main.Telecharger](http://www.mon-club-elec.fr/pmwiki_reference_pyduino/pmwiki.php?n=Main.Telecharger)

Si vous utilisez le pcduino 2, voir la procédure complète ici : [pcduino v2 + Pyduino : Soyez opérationnels en 15 minutes chrono !](#)

### Référence Pyduino

- Accueil
- Pour Débuter
- Telecharger**
- Fonctions Pyduino
- Par thème
- Cartes mini-PC
- FAQ
- Apprendre
- Docs et Trucs
- Licence
- So

Main



**Pyduino:**



**Pyduino Multimedia:**

---

[Télécharger Pyduino et Pyduino Multimédia](#)

---

**Vous venez de recevoir votre pcduino v2 ? Soyez opérationnels en 15 minutes !**

 Vous venez de recevoir le pcduino v2 dans sa boîte en carton... Si vous voulez aller droit au but et pouvoir rapidement coder votre pcduino avec ma librairie Pyduino, je vous propose une procédure en 4 étapes simples qui vous rendra opérationnel en 15 minutes chrono ! Suivez le guide :

[pcduino v2 + Pyduino : Soyez opérationnels en 15 minutes chrono !](#)

## 12. Documentation de la librairie

Toutes les instructions de la librairie sont expliquées et documentées à la façon « Arduino ».

La documentation complète de la librairie Pyduino est disponible ici :

[http://www.mon-club-elec.fr/pmwiki\\_reference\\_pyduino/pmwiki.php?n=Main.ReferenceEtendue](http://www.mon-club-elec.fr/pmwiki_reference_pyduino/pmwiki.php?n=Main.ReferenceEtendue)

### Structure

#### FONCTIONS DE BASE

- def : déclaration d'une fonction

Ces deux fonctions sont obligatoires dans tout programme utilisant les fonctions Pyduino :

- setup()
- loop()

#### SYNTAXE DE BASE

- fin de ligne
- : (deux-points)
- \# (commentaire sur une ligne)
- --- --- (commentaire sur plusieurs lignes)
- ( )
- import
- #define

#### STRUCTURES DE CONTRÔLE

- if
- if...elif...else
- for
- for.. in..
- while
- break
- continue
- return
- pass
- exit
- do...while
- switch-case
- goto

#### OPÉRATEURS ARITHMÉTIQUES

- = (égalité)
- + (addition)
- - (soustraction)
- \* (multiplication)

### Variables, constantes, objets

Les variables sont des expressions que vous pouvez utiliser dans les programmes pour stocker des valeurs, telles que la tension de sortie d'un capteur présente sur une broche analogique.

#### CONSTANTES PRÉDÉFINIES

Les constantes prédéfinies du langage Arduino sont des valeurs particulières ayant une signification spécifique.

- HIGH | LOW
- INPUT | OUTPUT | PULLUP
- True | False
- Identifiants de broches prédéfinis
- Constantes mathématiques prédéfinies

A ajouter : constantes décimales prédéfinies

#### EXPRESSIONS NUMÉRIQUES

- Expressions numériques entières
- Expressions numériques à virgule

#### TYPES DES DONNÉES

Les variables peuvent être de type variés qui sont décrits ci-dessous.

Synthèse des types de données Pyduino

#### Variables numériques

- None
- bool
- int (32 bits)
- long (64 bits)
- float (nombres à virgules)
- complex (nombres complexes)
- char
- byte
- unsigned int

### Fonctions Broches E/S

#### ENTRÉES/SORTIES NUMÉRIQUES

- pinMode(broche, mode)
- digitalWrite(broche, valeur)
- digitalRead(broche) --> int
- toggle(broche)
  
- digitalWriteMulti(listBroches, listValeurs)
- pinModeMulti(listBroches, listMode)
- digitalReadMulti(listBroches) --> listState

#### ENTRÉES ANALOGIQUES

- analogRead(brocheAnalog) --> int
- analogReadmV(brocheAnalog) --> float
- analogReference(type)

#### SORTIES "ANALOGIQUES" (GÉNÉRATION D'IMPULSION)

- analogWrite(brochePWM, valeur) - PWM 0-255
- analogWritePercent(brochePWM, valeur) - PWM 0-100%
- analogWriteHardware(brochePWM, valeur) - PWM
- setFrequencyPWM(broche, frequence)

#### ENTRÉES/SORTIES AVANCÉES

##### Sons

- tone()
- noTone()
- voir également playSound()

##### Emission sérielle

- shiftOut(broche, BrocheHorloge, OrdreBit, valeur)

### **13. La suite ?**

Une fois Pyduino installée :

- écrivez le programme minimum et les exemples de bases ici : [http://www.mon-club-elec.fr/pmwiki\\_reference\\_pyduino/pmwiki.php?n=Main.ApprendreExemples](http://www.mon-club-elec.fr/pmwiki_reference_pyduino/pmwiki.php?n=Main.ApprendreExemples)
- explorez les nombreux exemples disponibles ici : [http://www.mon-club-elec.fr/pmwiki\\_mon\\_club\\_elec/pmwiki.php?n=MAIN.PCDUINO](http://www.mon-club-elec.fr/pmwiki_mon_club_elec/pmwiki.php?n=MAIN.PCDUINO)
- apprenez pas à pas Pyduino à l'aide des tutos Pyduino : [http://www.mon-club-elec.fr/pmwiki\\_mon\\_club\\_elec/pmwiki.php?n=MAIN.ATELIERSPYDUINO](http://www.mon-club-elec.fr/pmwiki_mon_club_elec/pmwiki.php?n=MAIN.ATELIERSPYDUINO)

## Table des matières

Introduction à la librairie Pyduino :

Programmer un mini-PC aussi facilement qu'une carte Arduino... et bien plus encore !

Intro |

Pyduino : c'est quoi ? Pourquoi faire ? |

Ce que l'on peut faire avec Arduino |

Ce qu'on peut faire avec un mini-PC |

Ce qu'on peut faire avec un mini-PC et Pyduino |

Exemple : l'appareil photo |

Quand utiliser Pyduino avec un mini-PC ? |

Encore plus fort : Pyduino est utilisable directement au sein d'un code PyQt (interface graphique !) |

Pour passer à l'action... avec Pyduino : le matériel |

Les différentes possibilités d'utiliser Pyduino |

Installation de l'IDE Pyduino et de la librairie Pyduino à partir d'un paquet \*.deb |

Documentation de la librairie |

La suite ? |

**Bravo !**  
vous avez terminé cet atelier Pyduino !



Prêt pour la suite ? Retrouvez de nombreux autres thèmes d'ateliers Pyduino ici :  
[http://www.mon-club-elec.fr/pmwiki\\_mon\\_club\\_elec/pmwiki.php?n=MAIN.ATELIERSPYDUINO](http://www.mon-club-elec.fr/pmwiki_mon_club_elec/pmwiki.php?n=MAIN.ATELIERSPYDUINO)