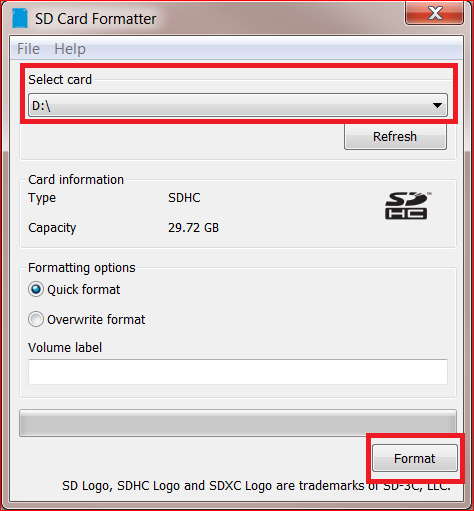
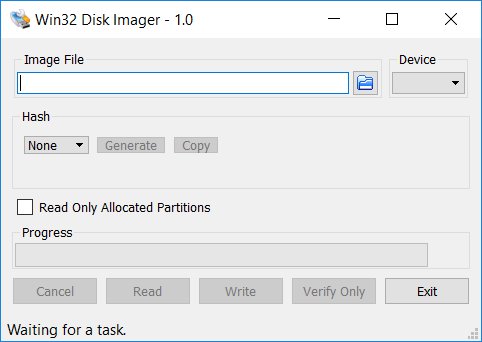
Installation of EasyCasting PiBox

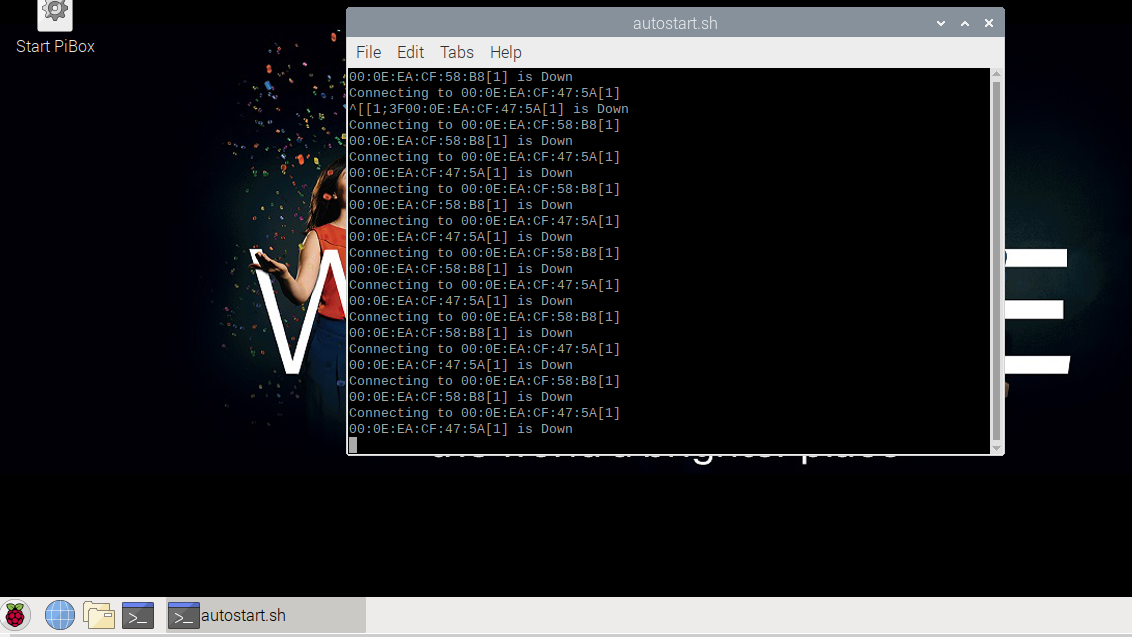
1. Introduction

* This project is a prototype
* Arduino detections by Raspberry Pi are manual
* Passwords are by defaults
* The code is not protected
* Communications are not protected
* The code is not ready for industry, it does not resist to component failures
* No GUI
* The installation is difficult

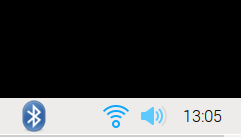
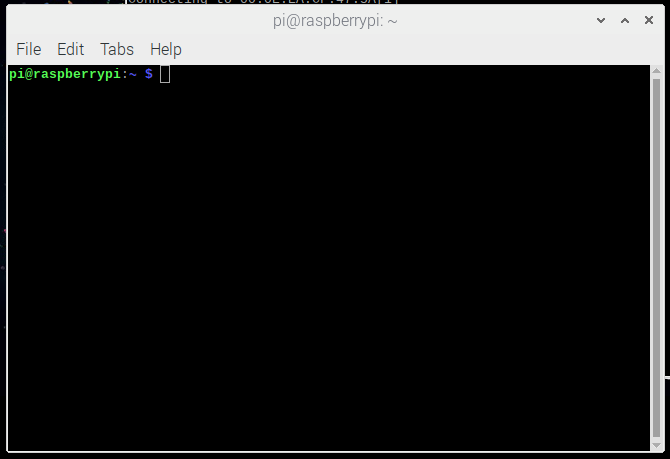
1. Create the SDCard

* Download and install SDCard Formatter : <https://www.sdcard.org/downloads/formatter/>
* Format a 32Go micro SDCard (with adapter) with SDCard Formatter with the option Overwrite Format
* 
* Download the Raspbian Pibox image here : www.cyrilvincent.com/download/raspbian-pibox.rar
* Dezip the file to obtain a .img
* Download and install win32diskimager: <https://sourceforge.net/projects/win32diskimager/>
* 
* Put the image file on the SDCard with win32diskimager
* Eject properly de SDCard

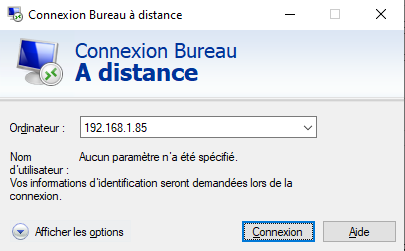
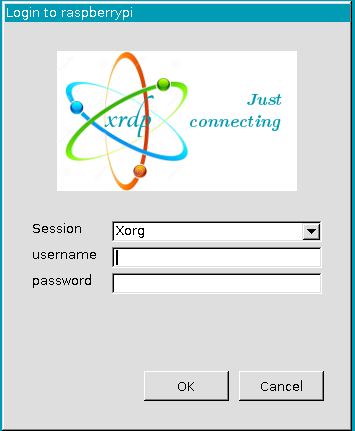
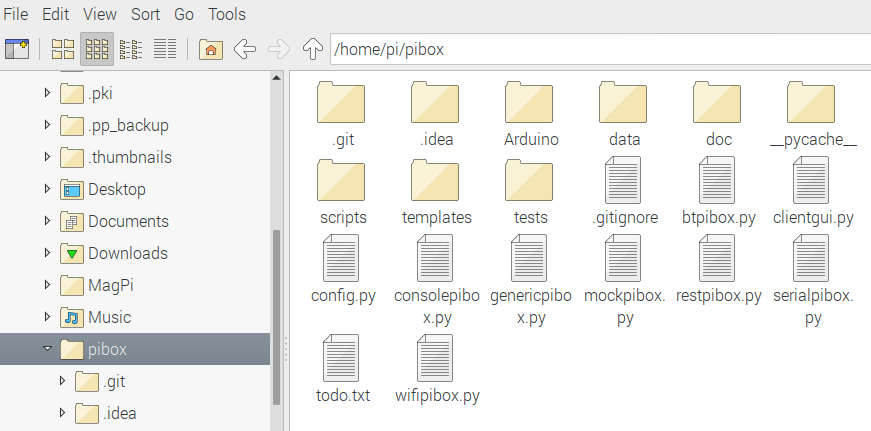
1. First boot of the Raspberry

* Put the SDCard into the Raspberry
* Plug a keyboard and a mouse on the USB
* Plug a screen on the micro-HDMI port
* Plug the Raspberry with a power supply USB –C 5V 2A
* 

1. Network

* Network is mandatory to access and debug the Raspberry from a PC
* Internet is mandatory to install drivers while the development process
* Connect to the WIFI with the Wifi icon at the bottom right of the screen
* 
* Test Internet with the Chromium browser (first icon after de raspberry icon)
* 
* If you have only an Ethernet connection plug it but it will be more difficult to share file
* Open a terminal with the terminal icon 
* 
* Type ifconfig
* Note your ip address in the section wlan0 / inet
* 
* In this example the ip address is 192.168.1.104

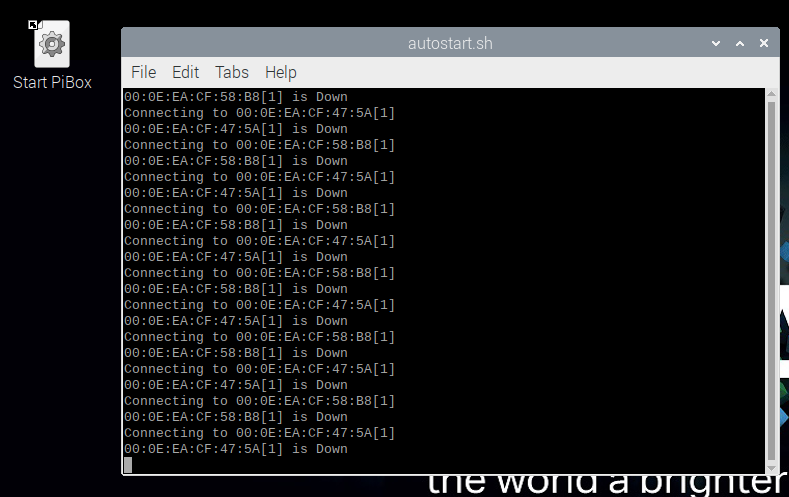
1. Remote access with Remote Desktop

* On Windows open a remote desktop
* 
* Enter raspberrypi, if it does not work enter the raspberry ip address
* The login is pi, the password is raspberry
* 
* You can now unplug the screen and the keyboard from the raspberry
* The code is into the pibox directory
* 
* This documentation is into pibox/doc directory
* Never change a file into the pibox directory, except config.py

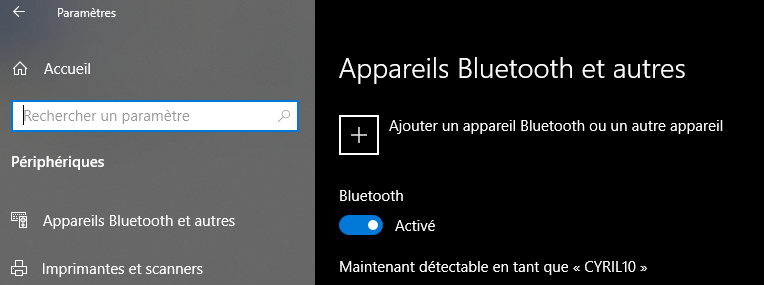
1. Remote access with Samba

* You can have a remote access to a shared directory with the Samba protocol
* On your PC open a file explorer
* Type [\\raspberrypi](file:///\\raspberrypi) or ip address
* You should see the directory of the pi user
* This documentation is into pibox/doc directory
* Never change a file into the pibox directory, except config.py
* Make a copy of the pibox directory on your PC

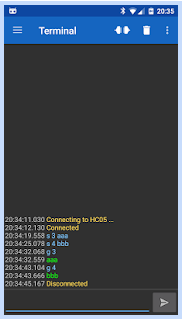
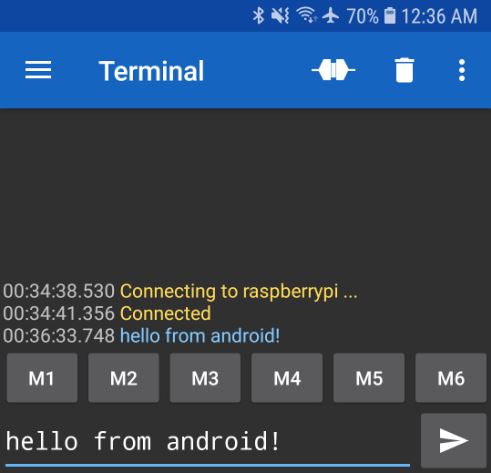
1. The PiBox application

* The Raspberry and Arduino applications and source code are into the pibox folder
* To stop the application just type Enter into the terminal and wait 12 seconds
* To start the application double click on Start PiBox on the desktop
* 
* You will see a lot of error message because there is no device
* 

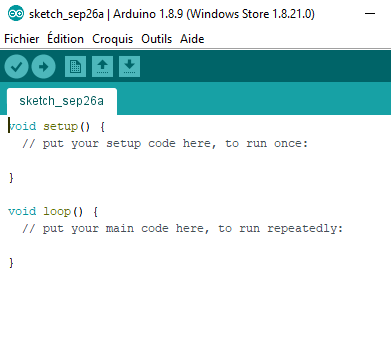
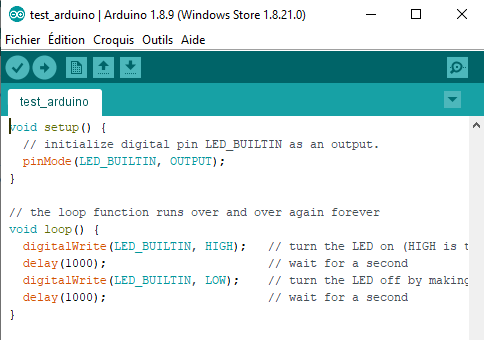
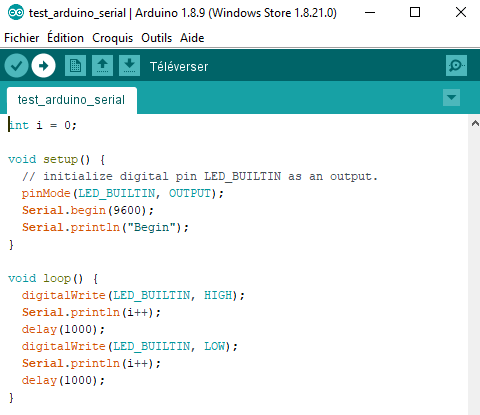
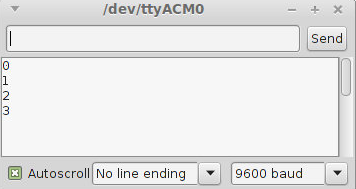
1. Pair the Raspberry with Windows

* On windows 10, type Bluetooth in Cortana and find the Bluetooth Parameter screen
* 
* On Raspbian click on the Bluetooth icon
* 
* If the Bluetooth icon does not appear, unplug the screen from the Raspberry and reboot
* Click on Make Discoverable
* Accept connections

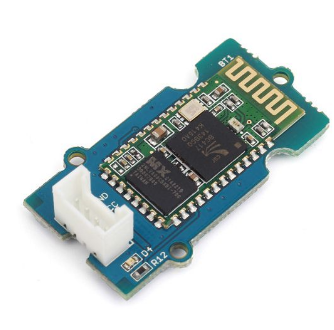
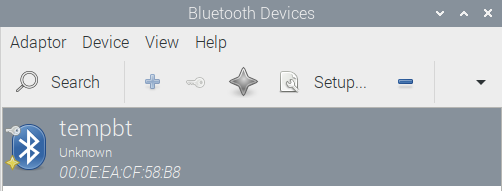
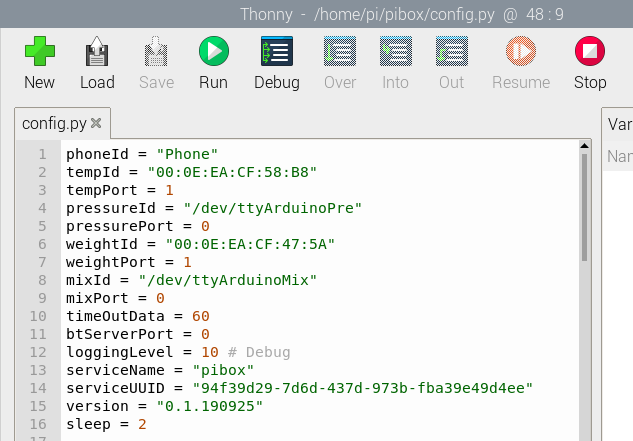
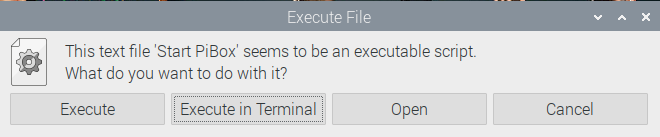
1. Pair the Raspberry with Android

* On Raspbian click on the Bluetooth icon
* Click on Make Discoverable
* 
* On Android, into the bluetooth parameters make the phone visible and search the Raspberrypi device
* Install the Serial Bluetooth Terminal : <https://play.google.com/store/apps/details?id=de.kai_morich.serial_bluetooth_terminal>
* Launch de Serial Bluetooth Terminal
* 
* Click on the 3 bars 
* Choose the raspberrypi
* 
* You should see something like this every 2s :
* {pho:0, temp:-4, pre:-4, wei:-4, mix:-4}
* -4 means device is down, try to reconnect every 10s
* -3 means the device is normally disconnected
* -2 means the device is disconnect, try to connect now
* -1 means the device is connected but no value was send
* 0 means the device is OK
* > 0 means the last value send by the device, with a timeout of 60s
* Type on your phone 4 + Send 
* Youd should receive {pho:4, temp:-4, pre:-4, wei:-4, mix:-4}
* Data send by Bluetooth respects the JSON standard
* Test with other values
* The PiBox software can work with 3 sensors, 1 mixer, 1 phone
* Each sensor can be accessed with Bluetooth, USB and Serial RS 232

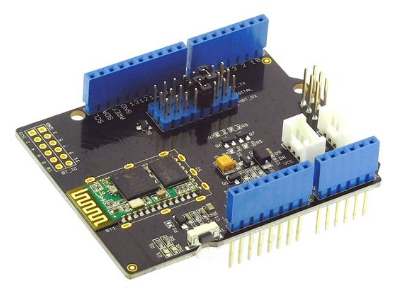
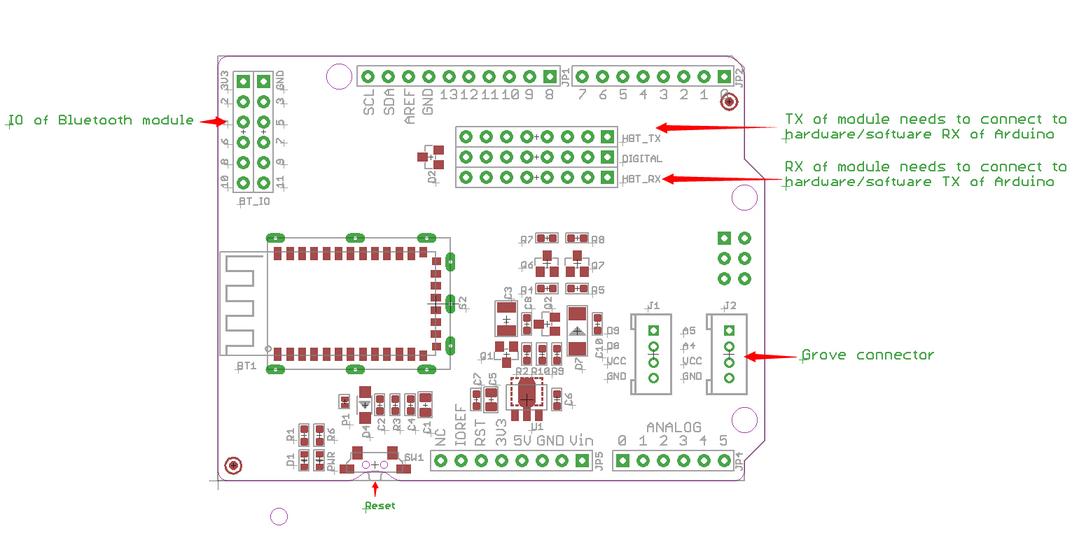
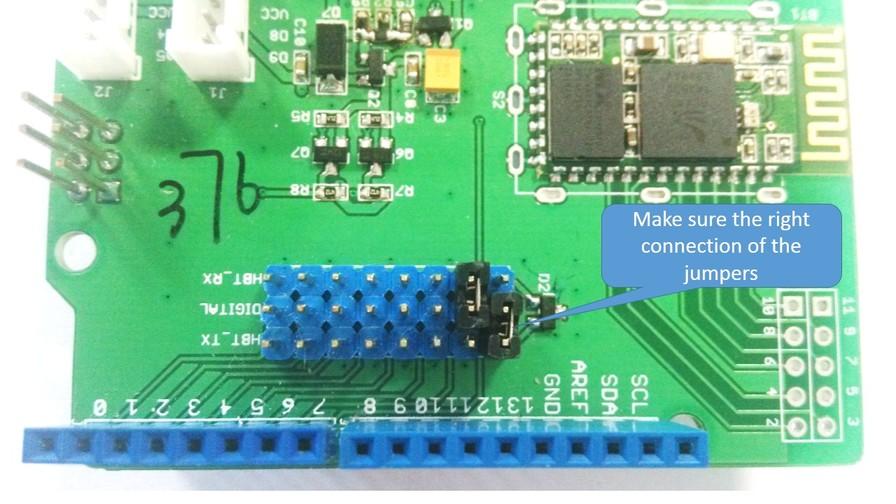
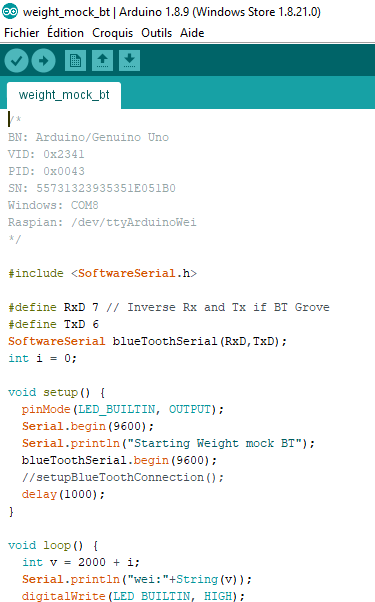
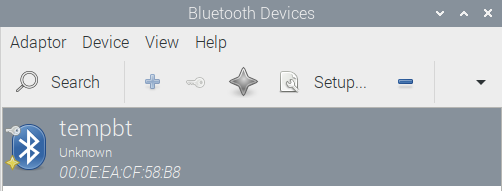
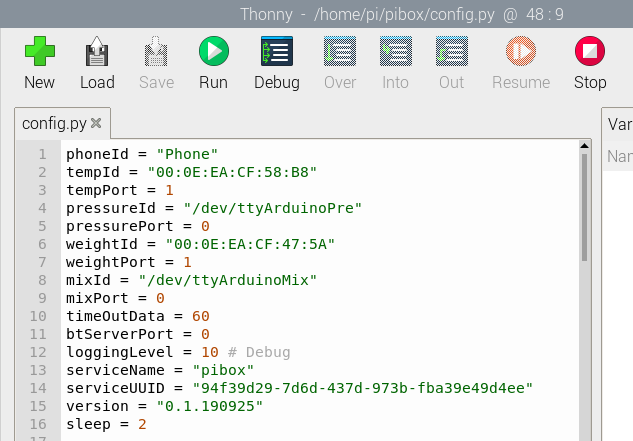
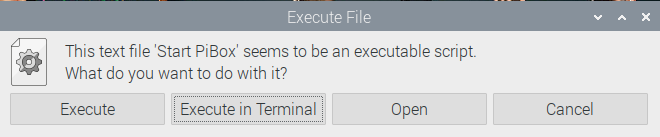
1. Install and push Arduino Software

* Download and Install the Arduino IDE : <https://www.arduino.cc/en/main/software>
* Plug an Arduino Uno Board with an USB B Cable on your PC or an external power supply (6V-12V 0.2A- 1A)
* 
* Start the Arduino IDE
* 
* In Tools menu select Card Type : Arduino / Genuino Uno
* In Port select the COM port which display Arduino
* Choose Récupérer les informations de la carte, you should see the serial number of the card
* 
* From the raspberry pi copy the pibox/Arduino folder on your PC
* From the Arduino IDE open the file pibox/Arduino/tests/test\_arduino/test\_arduino.ino
* 
* Click on the push button 
* The led of Arduino should blink every second
* Open the file test\_arduino\_serial.ino
* 
* Push the code 
* Open Tools / Serial monitor
* Choose 9600 bauds
* You should see 

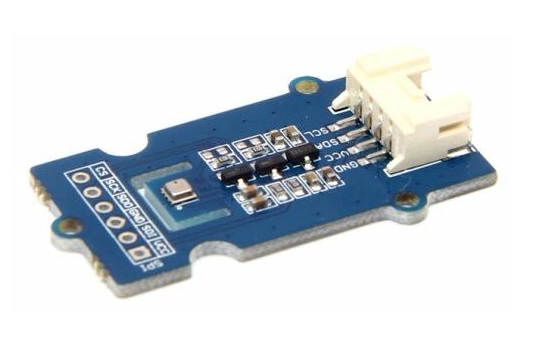
1. Build and configure the Arduino temperature sensor

* Take an Arduino board
* 
* Take a Grove Base shield
* 
* Put the little switch button (5) on the 5v position
* 
* Plug the base on the Arduino
* Take a temperature sensor
* 
* Connect the sensor on the A0 connector of the base shield
* Take a bluetooth device
* 
* Connect the device on the D6 connector of the base shield
* In Arduino IDE open the file pibox/Arduino/temp\_sensor\_12\_bt/temp\_sensor\_12\_bt.ino
* 
* This device work both bluetooth and serial
* Choose the correct serial port in Tools / Port
* Choose Tools / Récupérer les informations de la carte, you should see the serial number of the card
* 
* Notice the Serial Number in the comment
* Uncomment the line //setupBlueToothConnection(); (remove the //) and save
* Push the code
* Open the serial monitor and you should see the tem:22.00 where 22.00 is the temperature in degrees
* Comment the line setupBlueToothConnection() (put // before the function) and save
* Push
* On Android add the bluetooth device tempbt with the PIN code 1234
* Open the Serial Bluetooth Terminal, choose the tempbt device, you should see the temperature
* On raspbian pair the tempbt device
* On the bluetooth menu choose devices
* 
* Notice the MAC address of tempbt (eg. 00: 0E:EA:CF:58:B8)
* Open the file pibox/config.py (double click)
* 
* Modifiy the tempId value with the real MAC address
* Stop the pibox software if running simply by pressing Enter, wait 12s
* Start de Pibox with  and click on Execute in terminal
* 
* Wait
* On Android open the Serial Bluetooth Terminal and choose the raspberry device
* Wait
* You should receive {pho:-4, temp:22.00, pre:-4, wei:-4, mix:-4}

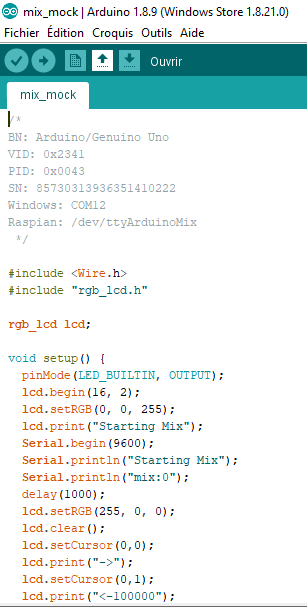
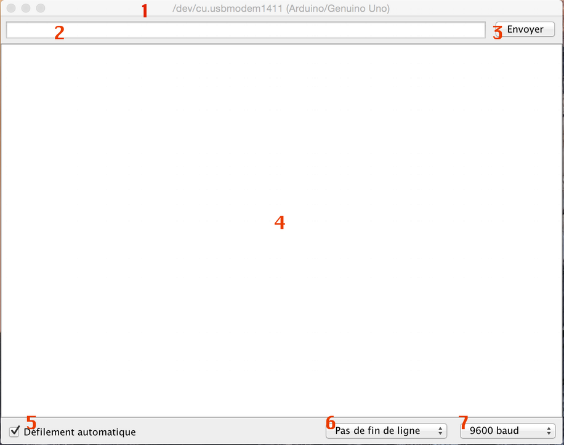
1. Build and configure the Arduino weight mock

* Take an Arduino board
* 
* Take a Bluetooth Shield
* 
* Locate the Rx and Tx jumpers on the board
* 
* Put Rx jumper at the 6th position and Tx jumper at the 7th
* 
* Plug the shield on the Arduino
* In Arduino IDE open the file weight\_mock\_bt.ino
* 
* This device work both bluetooth and serial
* Choose the correct serial port in Tools / Port
* Choose Tools / Récupérer les informations de la carte, you should see the serial number of the card
* 
* Notice the Serial Number
* Uncomment the line //setupBlueToothConnection(); (remove the //) and save
* Push the code
* Open the serial monitor and you should see the wei:2500 where 2500 is the mock weight in gram (it's an iteration from 2000 to 3000 every 2s)
* Comment the line setupBlueToothConnection() (put //) and save
* Push
* On Android add the bluetooth device weightbt with the PIN code 1234
* Open the Serial Bluetooth Terminal, choose the weightbt device, you should see the temperature
* On raspbian pair the weightbt device
* On the bluetooth menu choose devices
* 
* Notice the MAC address of weightbt (eg. 00: 0E:EA:CF:58:B8)
* Open the file pibox/config.py (double click)
* 
* Modifiy the weightId value with the real MAC address
* Stop the pibox software if running simply by pressing Enter, wait 12s
* Start de Pibox with  and click on Execute in terminal
* 
* Wait
* On Android open the Serial Bluetooth Terminal and choose the raspberry device
* Wait
* You should receive {pho:-4, temp:22.00, pre:-4, wei:2500, mix:-4}

1. Build and configure the Arduino pressure sensor

* Take an Arduino board
* 
* Take a Grove Base shield
* 
* Put the little switch button (5) on the 5v position
* 
* Plug the base on the Arduino
* Take a pressure sensor
* 
* Connect the sensor on a I2C connector
* In Arduino IDE open the file pressure\_bm280\_sensor.ino
* 
* Choose the correct serial port in Tools / Port
* Choose Tools / Récupérer les informations de la carte, you should see the serial number of the card
* 
* Notice the Serial Number
* Push the code
* Open the serial monitor and you should see the pre:99500 where 99500 is the atmospheric pressure in Pa
* Plug the device on any USB port of the raspberry
* On the raspberry open a terminal
* Type cd /etc/udev/rules.d
* Type sudo nano 99-com.rules
* 
* With arrows keys scroll down to the end of the file
* Find this line (before last line) : SUBSYSTEM=="tty", ATTRS{idVendor}=="2341", ATTRS{idProduct}=="0043", ATTRS{serial}=="857303139363515181D2", SYMLINK+="ttyArduinoPre"
* Replace the value in ATTRS{serial}=="857303139363515181D2" by the serial number of the Arduino board
* Verify that idVendor and idProduct are the same than the Arduino board
* Control+S to save
* Control+X to exit
* Reboot and wait
* On Android open the Serial Bluetooth Terminal and choose the raspberry device
* Wait
* You should receive {pho:-4, temp:22.00, pre:99.5, wei:2500, mix:-4}
* Pressure is now in HPa

1. Build and configure the Arduino mix mock

* Take an Arduino board
* 
* Take a Grove Base shield
* 
* Put the little switch button (5) on the 5v position
* 
* Plug the base on the Arduino
* Take a LCD Display
* 
* Connect the sensor on a I2C connector
* In Arduino IDE open the file mix\_mock.ino
* 
* Choose the correct serial port in Tools / Port
* Choose Tools / Récupérer les informations de la carte, you should see the serial number of the card
* 
* Notice the Serial Number
* Push the code
* Open the serial monitor
* 
* Enter a number between 1 and 31 into the serial monitor (part 2 of the screen) and press the buttons end (3)
* The Arduino print mix:101010
* The sending number represent the input of the mix board, the response is a binary implementation of the device outputs, the left bit represent the state of the board (1 for ok). The 5 other bits represents the 5 outputs of the board
* Plug the device on any USB port of the raspberry
* On the raspberry open a terminal
* Type cd /etc/udev/rules.d
* Type sudo nano 99-com.rules
* 
* With arrows keys scroll down to the end of the file
* Find this line (last line) : SUBSYSTEM=="tty", ATTRS{idVendor}=="2341", ATTRS{idProduct}=="0043", ATTRS{serial}=="857303139363515181D2", SYMLINK+="ttyArduinoMix"
* Replace the value in ATTRS{serial}=="857303139363515181D2" by the serial number of the Arduino board
* Verify that idVendor and idProduct are the same
* Control+S to save
* Control+X to exit
* Reboot the system
* Wait
* On Android open the Serial Bluetooth Terminal and choose the raspberry device
* Wait
* You should receive {pho:-4, temp:22.00, pre:99.5, wei:2500, mix:100000}
* The LCD should display -> and <-100000
* 100000 means the device is ready and the 5 ouputs are set to 0
* On the phone send a number between 1 and 31
* You should receive {pho:1, temp:22.00, pre:99.5, wei:2500, mix:100001}
* The LCD should display ->number and <-100001 the response of the device (1 bit for the device status, 5 bits for the 5 outputs)

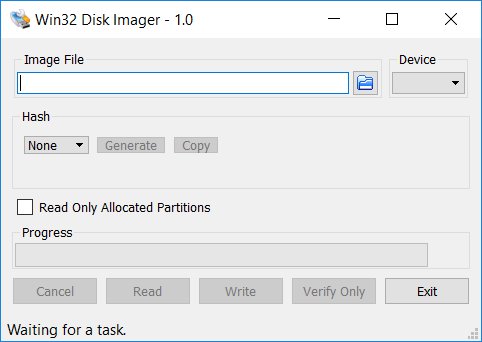
1. Plug and configure the screen

* Open and read the documentation in pibox/doc/pj2-rb-tft3-2-3-5-fr-1573.pdf

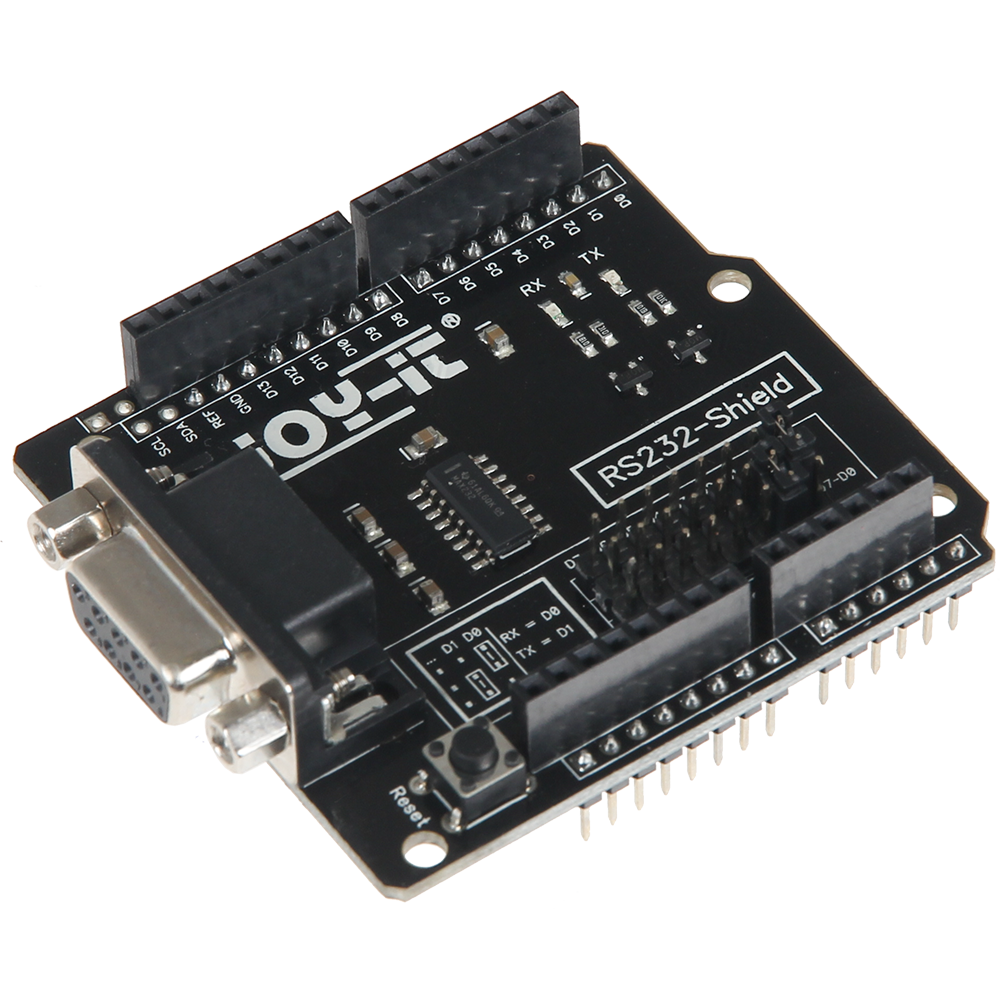
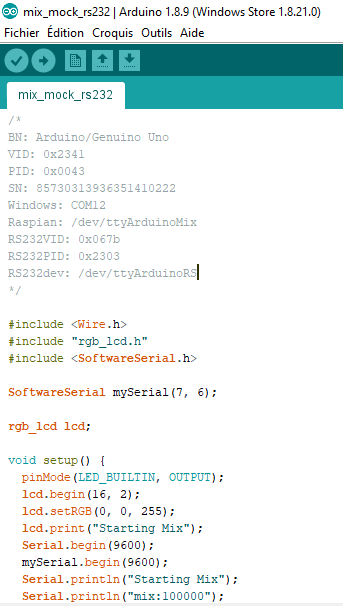
1. Start automatically without screen

* The application start automatically only if a desktop (if a screen is plug) or via a remote desktop
* To start automatically the PiBox without screen and remote type sudo nano /boot/config.txt
* Uncomment the line #hdmi\_force\_hotplug=1
* Save & Reboot
* You can also access remotely to raspbian with X11VNC, type x11vnc
* From your PC download and install VNC Viewer
* In VNC Viewer type raspberrypi and you can connect to raspberry
* If you want to automatically start a VNC server type sudo raspi-config and serach for enable VNC server
* You can also access remotely to raspbian with SSH, simply put a empty file ssh at the root of the sd card, reboot and access with a SSH client : ssh pi@raspberrypi

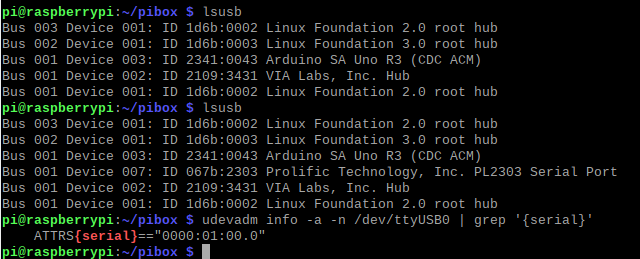
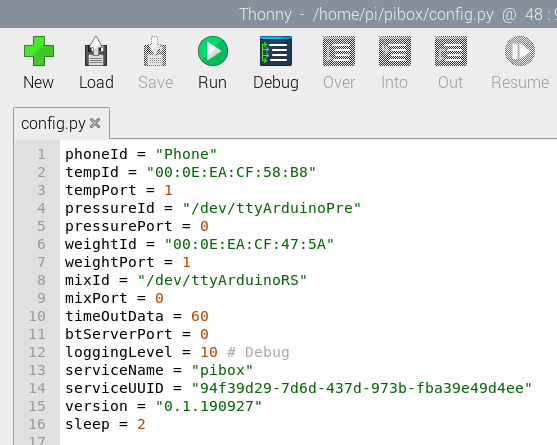
1. Create a Raspbian image

* Shut down the Raspberry
* Eject the SD Card
* Put it into your PC
* Start Win32 Disk Imager
* 
* Select the device
* Create an image file
* Click Read

1. RS232 on the Arduino Mix Mock

* Take an RS232 Arduino Shield
* 
* Put jumpers at this position : Rx: 7, Tx: 6
* Plug the shield on the Grove Base of the Mix Mock board
* In Arduino IDE open the file mix\_mock\_rs232.ino
* 
* Choose the correct serial port in Tools / Port
* Push the code
* Test with Android

1. RS232 on the Raspberry

* I did not get to work the blueooth shield and the RS232 at the same time
* As you can see in this documentation of the RS232 shield for Raspberry, this device work directly on the UART with GPIO 14 and 15 : <https://joy-it.net/en/products/RB-RS232>
* But the UART is setting by default to manage Bluetooth, Raspberry have only one UART so he can't manage Bluetooth and Hardware serial UART RS232 at the same time : <https://www.framboise314.fr/le-port-serie-du-raspberry-pi-3-pas-simple/>
* There is a software UART named miniUART at the GPIO 8 and 10, but the shield cannot address this GPIO
* I solved the problem by buying and USB – RS232 converter on Amazon
* <https://www.amazon.fr/UGREEN-PL2303-Windows-Chrome-Connecteur/dp/B00QUZY4UG/>
* 
* Be sure that the USB cable is not plugged to the raspberry
* On the raspberry type lsusb
* Notice all the USB devices
* Plug the USB – RS232 adapter between Arduino and Raspberry
* Type lsusb and notice the difference
* 
* Here the difference is the device Prolific Technology, Inc. PL2303 Serial Port
* Notice the ID, before ":" it's the vendorid, after it's the productid
* Here vid = 067b, pid = 2303
* Type cd /etc/udev/rules.d
* Type sudo nano 99-com.rules
* 
* With arrows keys scroll down to the end of the file
* Find the following line (at the end) SUBSYSTEM=="tty", ATTRS{idVendor}=="067b", ATTRS{idProduct}=="2303", SYMLINK+="ttyArduinoRS"
* If this line does not exist create it at the end
* Put the correct idVendor and idProduct
* Open the file pibox/config.py
* Replace the line mix = "/dev/ttyArduinoMix" by mix = "/dev/ttyArduinoRS"
* 
* Reboot and Test
* If it does not work return to the USB configuration

1. The code

* All the code is under the pibox directory
* The main program is btpibox.py
* All raspbian scripts are into the scripts folder
* Communications between phone and raspberry can be done with Bluetooth 5, Wifi and Rest API + Websocket
* All communications between raspberry and devices can be done with Bluetooth 3 or USB
* The code is updated automatically at every start via Git
* Never modify a file under the pibox directory except the config.py
* Raspberry development : Python 3.7
* Raspberry scripts : bash
* Arduino development : C++ 11 + GCC 11 Arduino 1.8
* Raspberry OS : Raspbian Buster 10 based on Debian Buster 10 ARM32
* IDE Python : Pycharm 2019 CE
* IDE Arduino : Arduino IDE 1.8
* Tested on Raspian 10 and Windows 10
* Tested on Seeeduino board