LYCÉE POLYVALENT
Raoul Georges NICOLO

LAURETTA-PERONNE Aymerick

CLAVIER SCIENTIFIQUE BLUETOOTH

Logiciel de lecture des symboles scientifiques

Configuration d'un clavier spécifique par matière

Valeurs numériques des symboles scientifiques

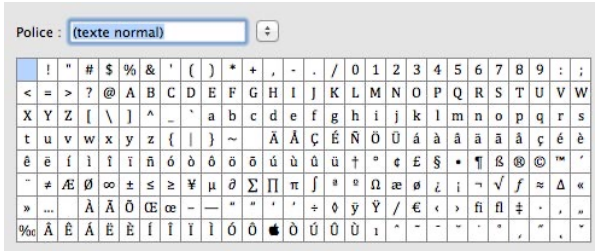


Diagramme FAST

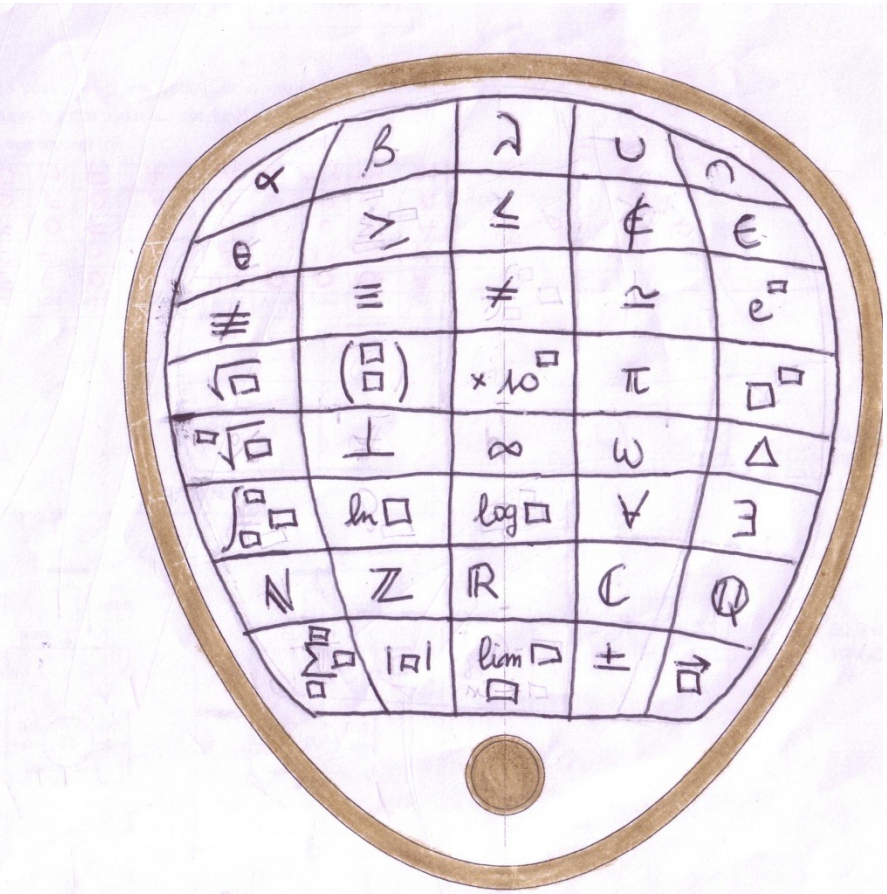
FS1:Permettre aux étudiants de noter des symboles mathématiques plus rapidement

- FT11:Acquérir les ordres
  - 111: Gérer les ordres de l'ordinateur pour afficher un clavier spécifique — Logiciel
  - 112: Acquérir les symboles à afficher à l'écran — Touches de clavier
- FT12:Gérer le fonctionnement — Microcontroleur
- FT13:Communiquer avec l'ordinateur portable
  - 131: Communiquer les symboles à l'ordinateur — Bluetooth
- FT14:Gérer l'énergie
  - 141: Prévenir la fin de batterie à l'utilisateur (seuil de 15%) — Signal lumineux LED
  - 142:Alimenter en énergie — Batterie
- FT15: Contenir le système
  - 151: Doit être anti-dérapant — Matériaux anti-dérapants
  - 152: Doit résister aux efforts — Matériaux résistants

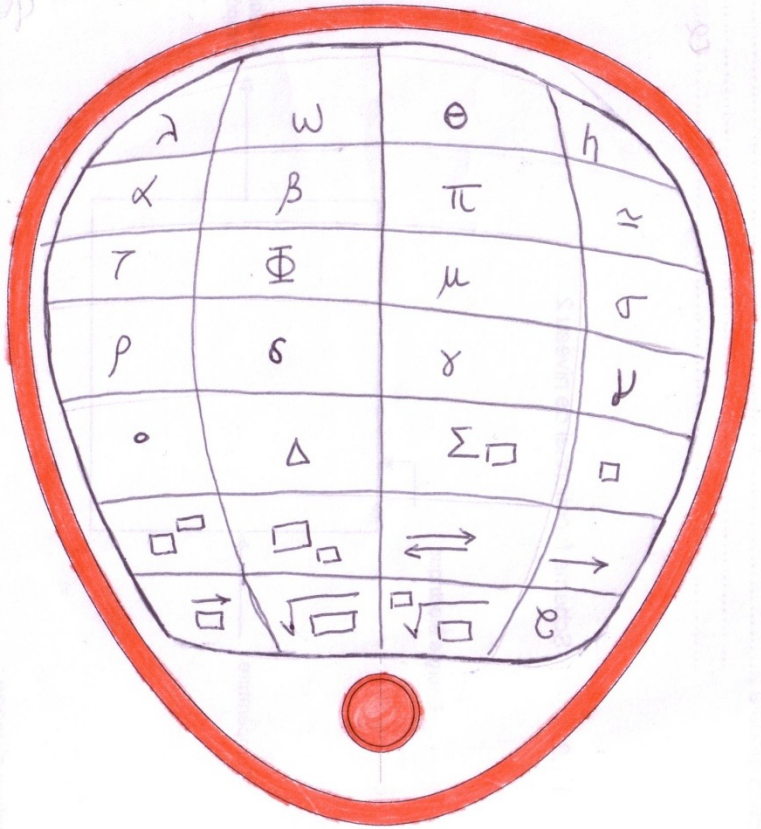
Caractéristiques techniques

Fonction de service	**Critère**	**Niveau**	**Limite**
FS1: permettre aux étudiants scientifiques de noter des symboles plus rapidement	Autonomie	48 h	max
	Dimensions	7x15 cm	min
		170 g	min
		5 ” pouces	max
	Distance de Communication	1 m	max

Configuration du clavier



Symboles Mathématiques

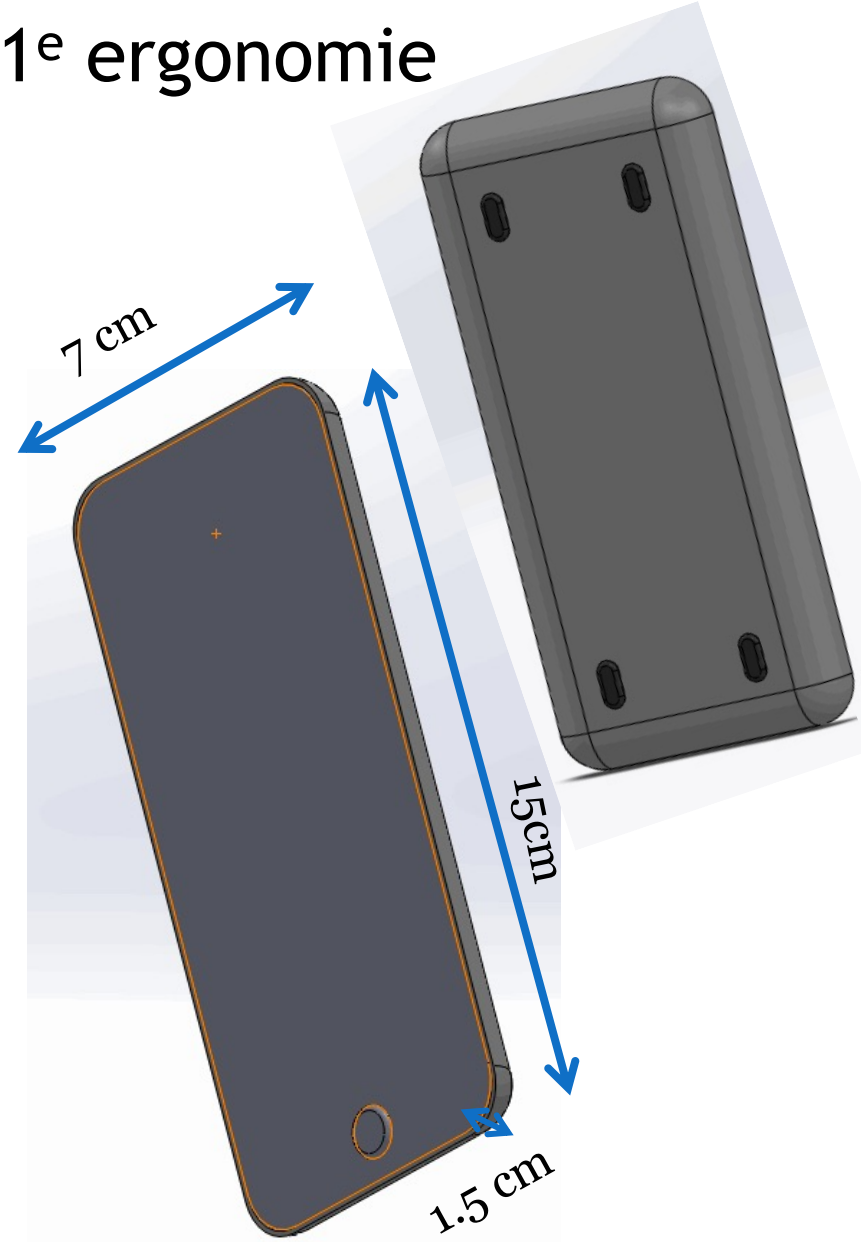


Symboles Physique-Chimie

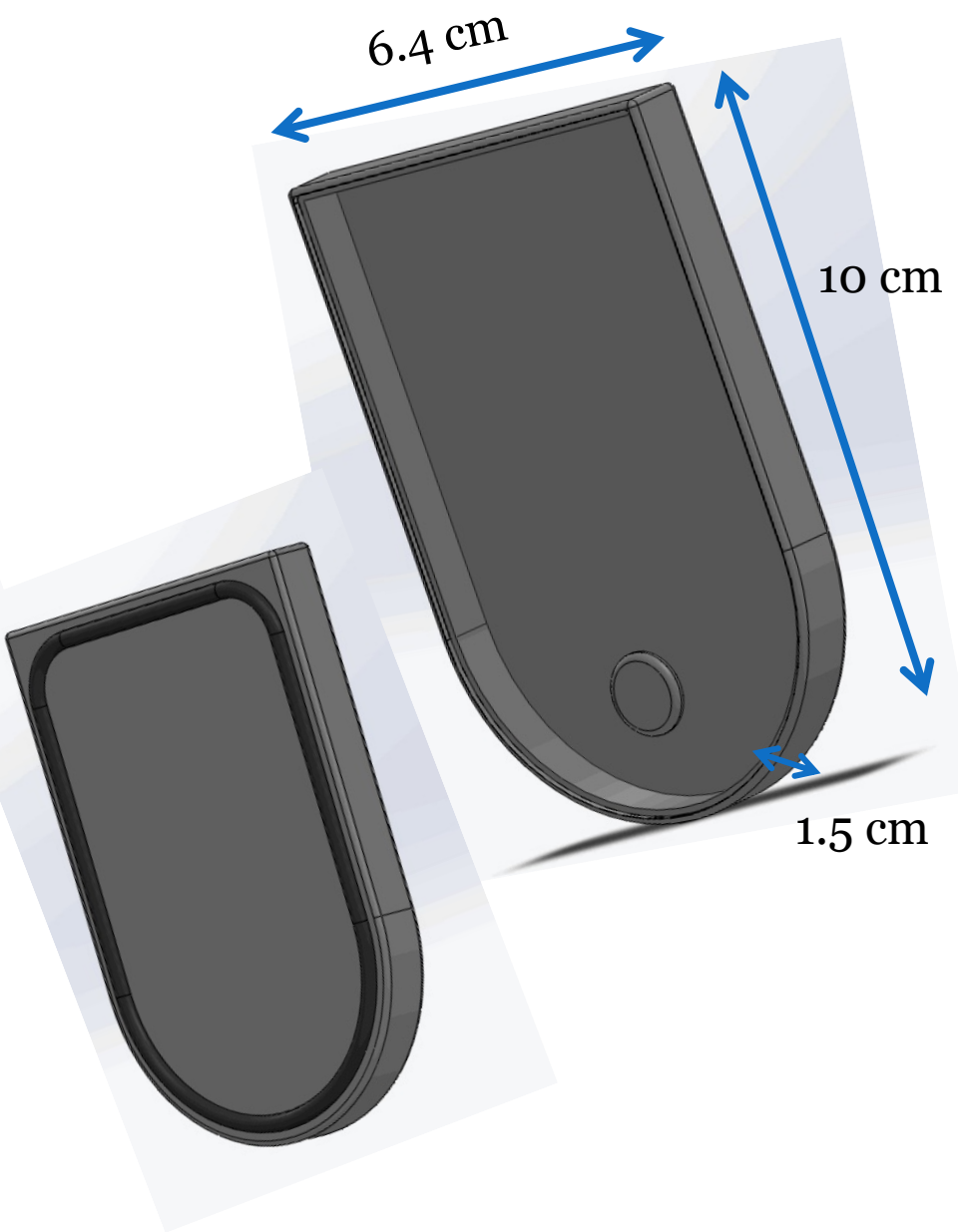
Matériaux envisageables

Matériau	Masse Volumique Kg/m^3	Rigidité GPa	Isolance thermique	Prix EUR/kg	Elément du boitier
Plastique (chlorure de polyvinyle)	1.4e3	3.14	Bon isolant	1.1	Coque Bouton
Fibre de carbone	1.5e3	109	Mauvais isolant	29.5	Coque Bouton
Verre	2.2e3	80	Mauvais isolant	6.2	Ecran
Aluminium	2.7e3	89	Mauvais isolant	1.6	Coque Bouton
Caoutchouc	925	0.0015	Bon isolant	3.3	Pièces antidérapantes

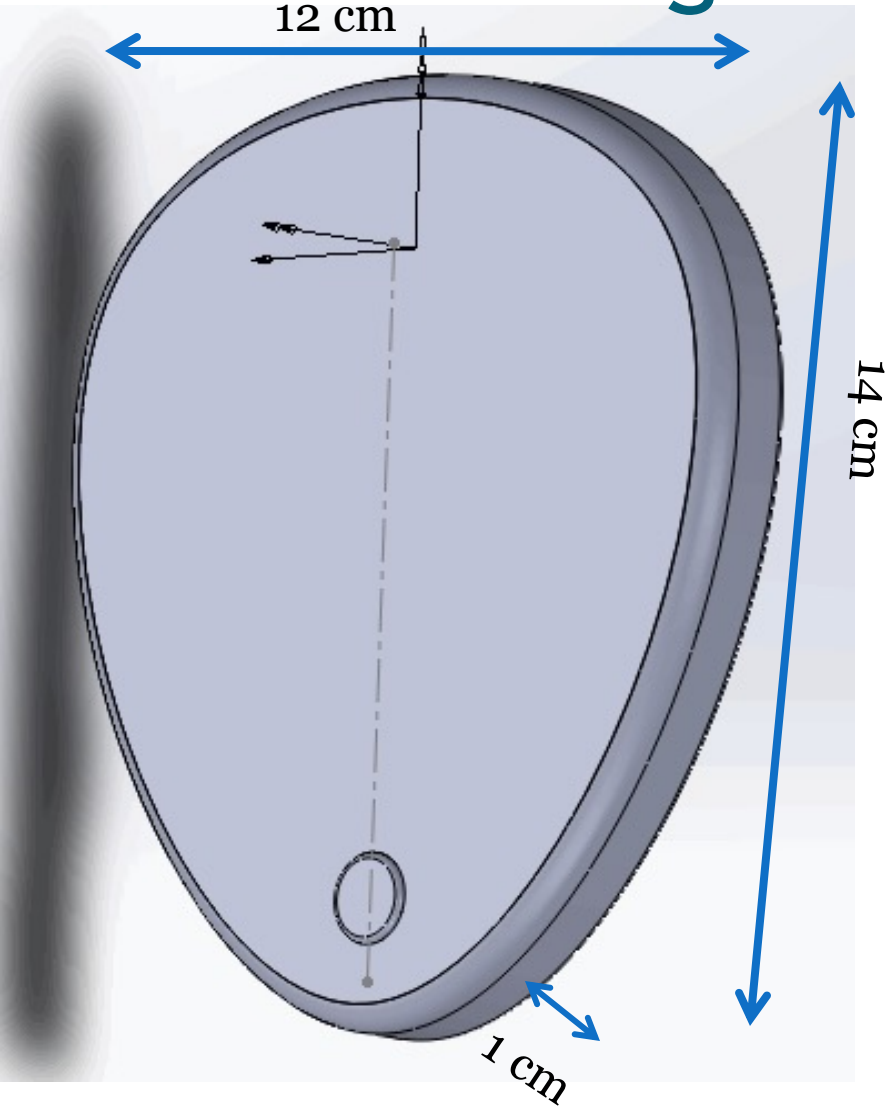
1e ergonomie

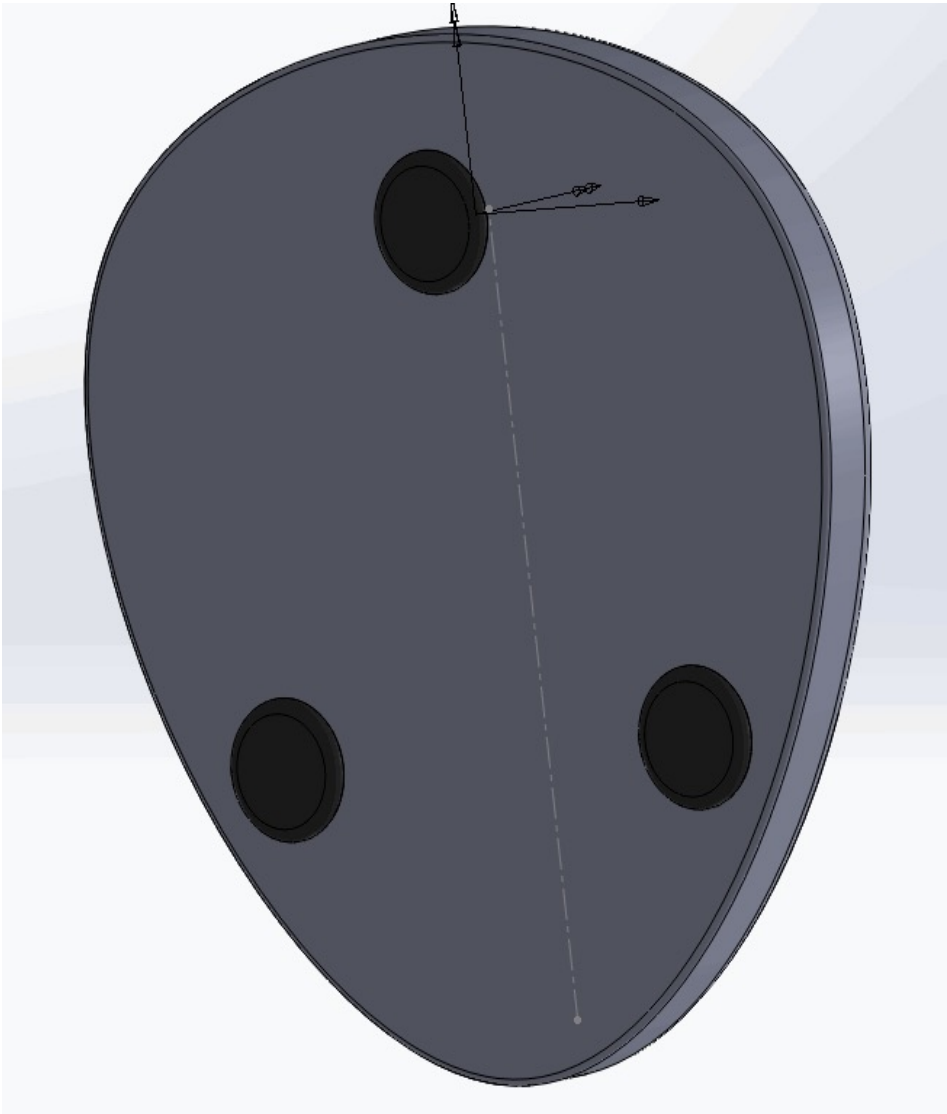


2e ergonomie

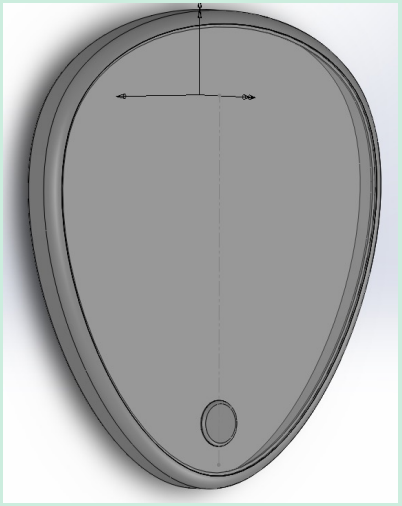
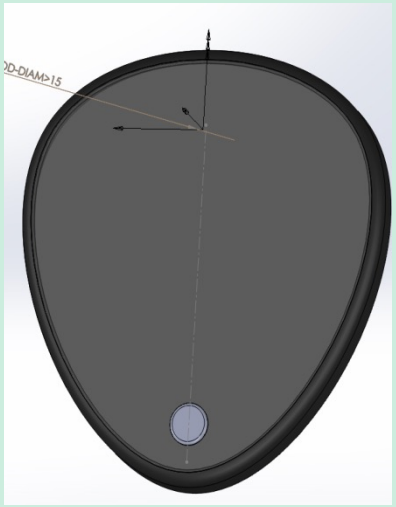
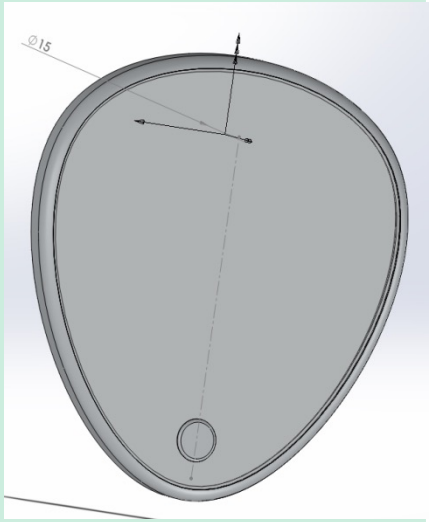


Ergonomie finale





Simulation matériaux

Matériaux:	Plastique	Fibre de Carbone	Aluminium
Masse du boitier :	90.14 g	118.5 g	147.8 g
Masse totale du boitier: (avec système interne)	180.14 g	208.5 g	237.8 g
Modélisation Solidworks:			

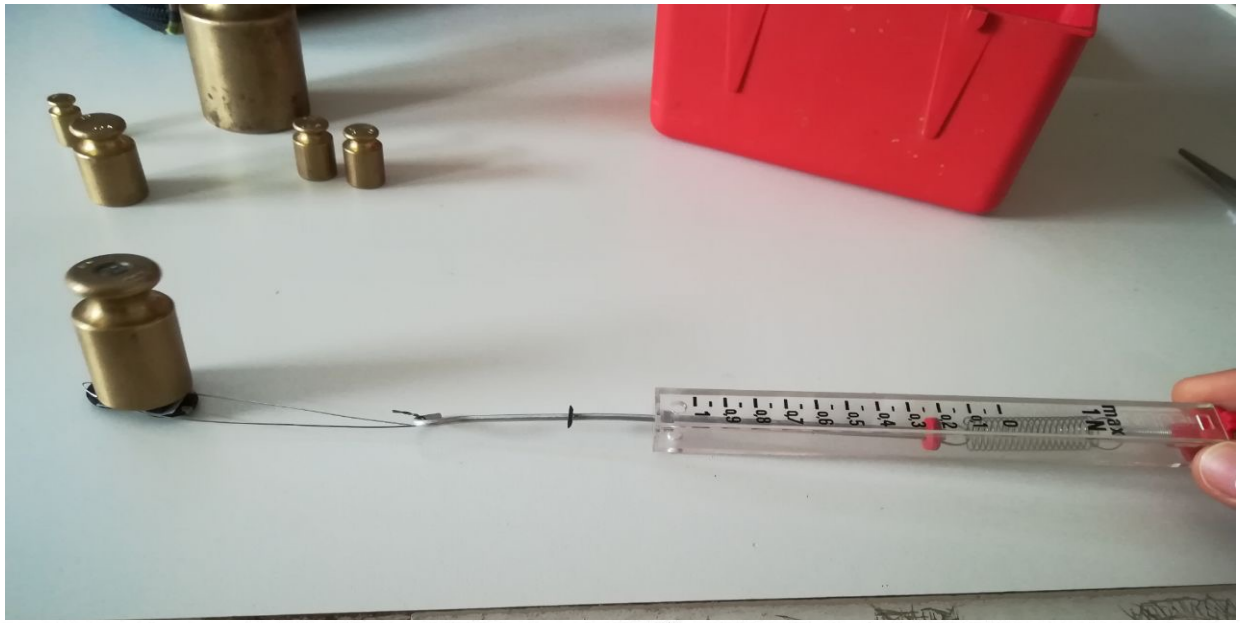
Détermination du prix total du clavier choisi (le plastique)

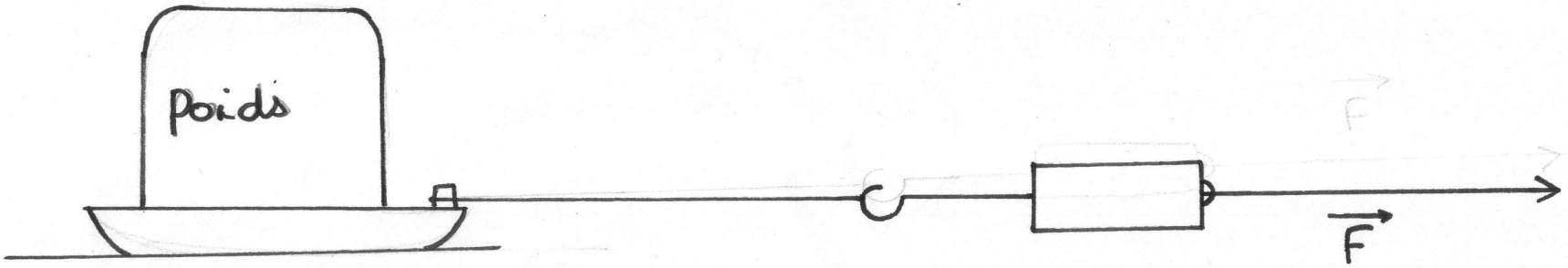
Système interne	Matériau Verre	Matériau Plastique	Matériau Caoutchouc	Prix total
90.17 €	0.19€	0.6€	0.26€	91.22€

Expérimentation (test anti dérapage)

Matériels utilisés

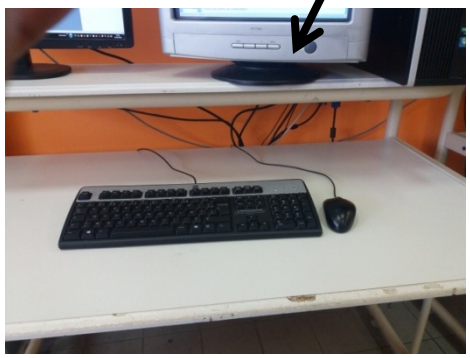
Protocole de mesure



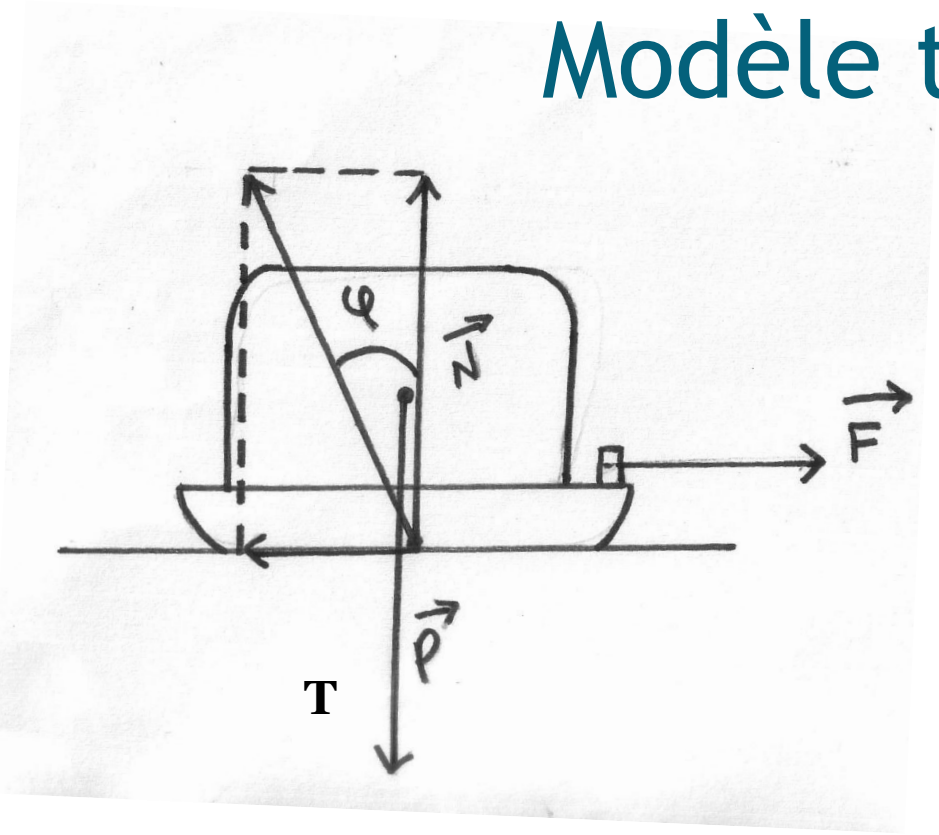


Résultats

Support	table informatique	table en bois vernis	table de cours
Force nécessaire pour 100 grammes	0.6 N	0.8 N	1.5 N
Force nécessaire pour 180 grammes	1.08 N	1.44 N	2.7 N



Modèle théorique



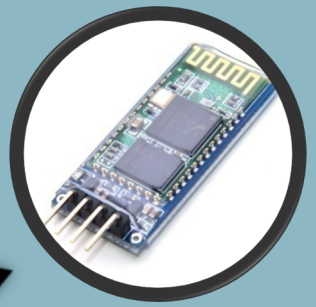
Loi de Coulomb:

Tangente φ =T/N

Coefficient d'adhérence

	Table informatique	Table en bois vernis	Table de cours
CA	1.67	1.25	0.67

Assurer la communication par Bluetooth



Module Bluetooth

Ordinateur

Communication par Bluetooth

Clavier scientifique

Le Bluetooth?

Le Bluetooth est un standard de communication permettant l'échange bidirectionnel de données à très courte distance en utilisant des ondes radio UHF sur une bande de fréquence de 2,4 GHz.
Son objectif est de simplifier les connexions entre les appareils électroniques en supprimant les liaisons filaires (câbles).

Caractéristiques

Le Bluetooth permet d'obtenir des débits de l'ordre de 1 Mbps, correspondant à 1600 échanges par seconde en full-duplex, avec une portée d'une dizaine de mètres environ avec un émetteur de classe II et d'un peu moins d'une centaine de mètres avec un émetteur de classe I.

Le standard Bluetooth définit en effet 3 classes d'émetteurs proposant des portées différentes en fonction de leur puissance d'émission :

Classe	Puissance (affaiblissement)	Portée
I	100 mW (20 dBm)	100 mètres
II	2,5 mW (4 dBm)	15-20 mètres
III	1 mW (0 dBm)	10 mètres

Protocole de suivi

(TRANSMISSION DES ONDES ELECTRO–MAGNETIQUES)

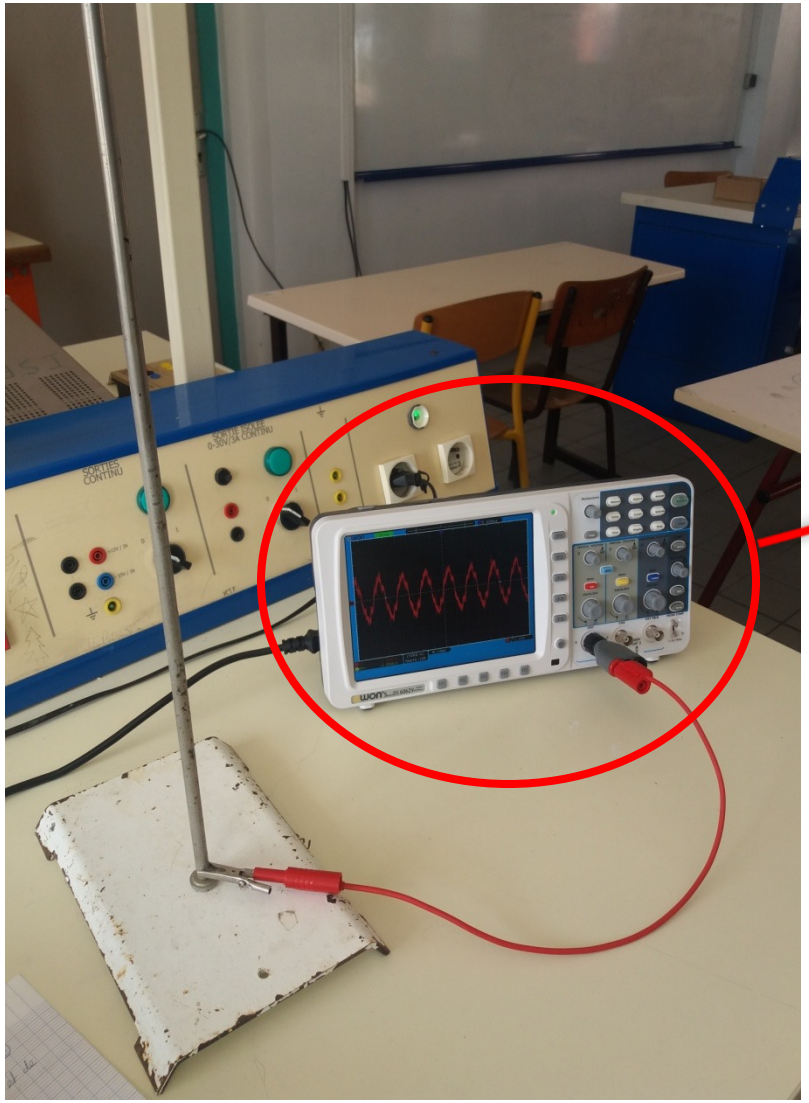
→ Couplage de champ magnétique et de champ électrique

MATERIAUX:
- 2 GBF électriques
- 2 tiges métalliques «Antennes»
- Câbles
- 1 oscilloscope

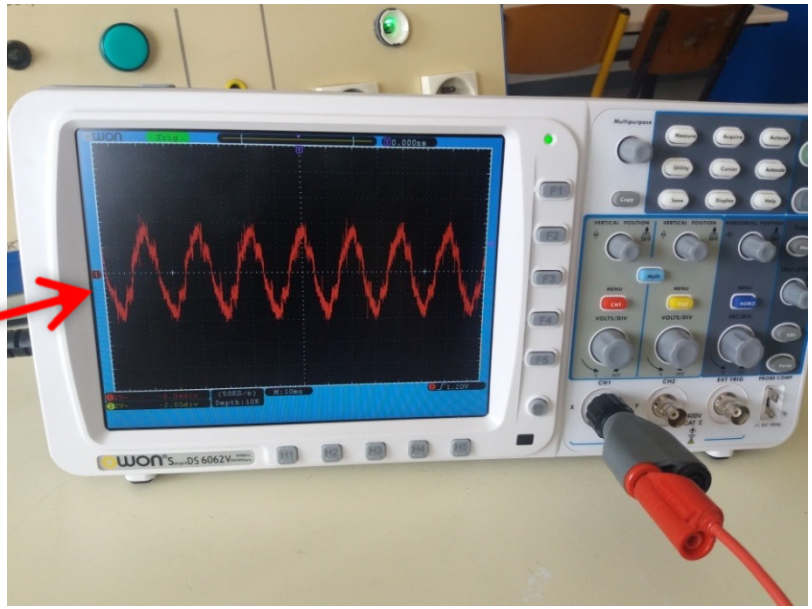
TESTS:
1) Montrer la présence d'ondes électromagnétiques autour de nous
2) Observer un signal électrique modulé en amplitude
3) Prouver le transfert d'onde par antennes

CONSTATS:
- Existence d'ondes électromagnétiques
- Possibilité d'émission-réception d'une portée ≤ à un 1m

Montrer la présence d'ondes électromagnétiques autour de nous



Sachant que la fréquence généré par le courant distribué par EDF vaut 50Hz



T= 2x10
= 20 ms

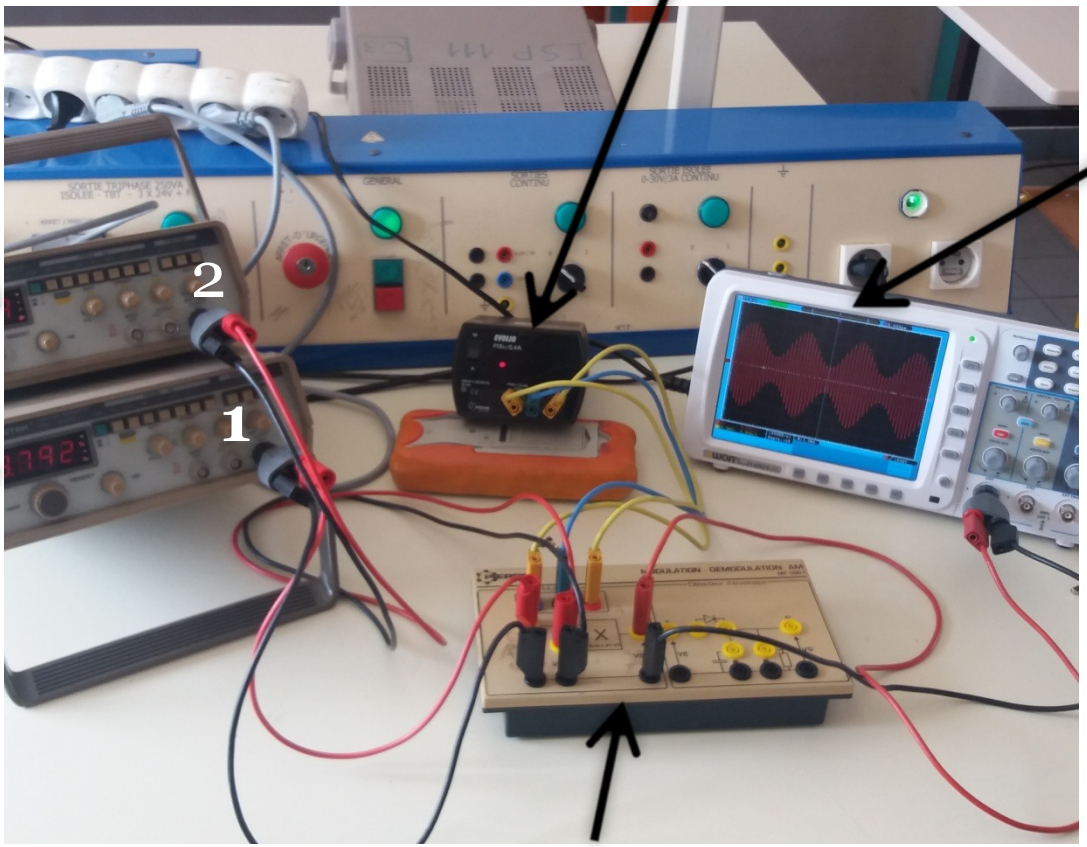
f=1/T = 1/0.002
= 50 Hz

Résultat: on retrouve bien une valeur de 50Hz

Représentation d'un signal modulé en amplitude

Générateur symétrique (+15 V/-15 V)

GBF à basse fréquence (Hz), le signal modulant



Oscilloscope

GBF à haute fréquence (KHz), la porteuse

Multiplieur (bobine+ condensateur intégrés)

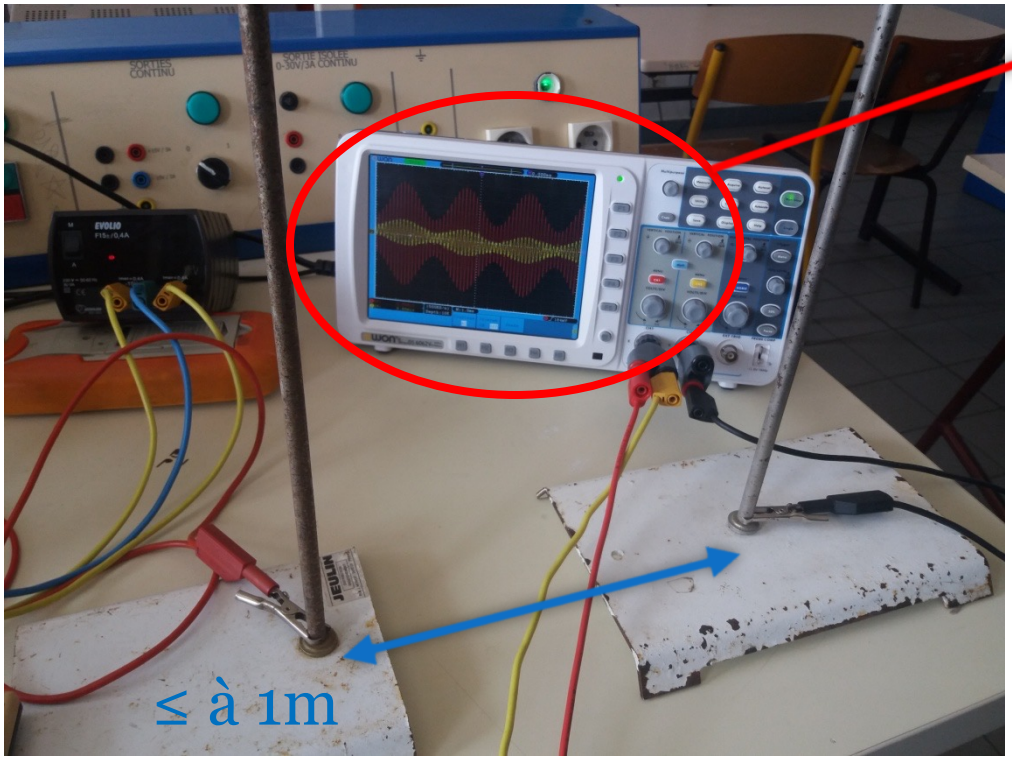
Résultats de mesure

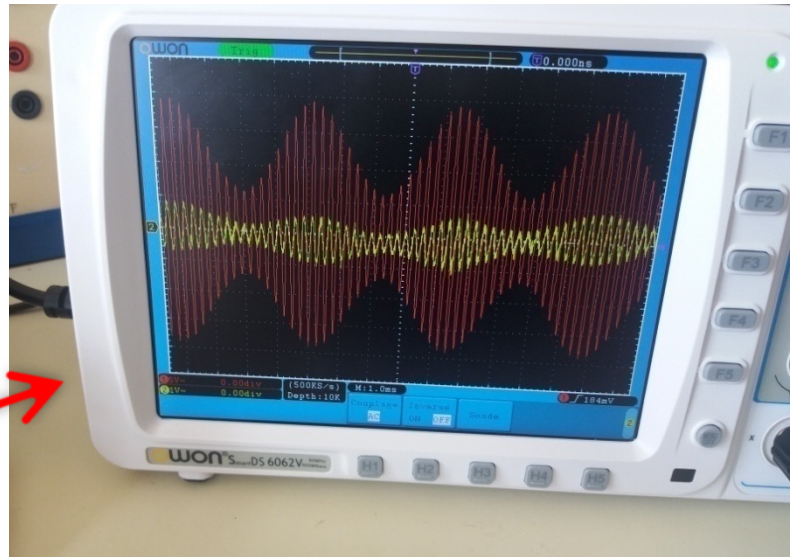
	1) GBF à haute fréquence (la porteuse)	2) GBF à basse fréquence
Valeur obtenu sur le GBF	3, 770 KHz	269 Hz
Période T en (s)	2,6 x 100E-06 = 2,6E-04	0,8 x 5.10-3 = 0,004 s
Fréquence f en (Hz)	1 / 2,6E-04 = 3846 Hz	1/0,004 = 250 Hz
Ecart relatif en %	3846-3770/ 3846 = 0,019 ≈ 0,02 ≈ 2 %	269-250/269 = 0,07 = 7 %

Constat: Variation de l'amplitude de la porteuse= signal modulé en amplitude

Prouver le transfert d'onde par antennes

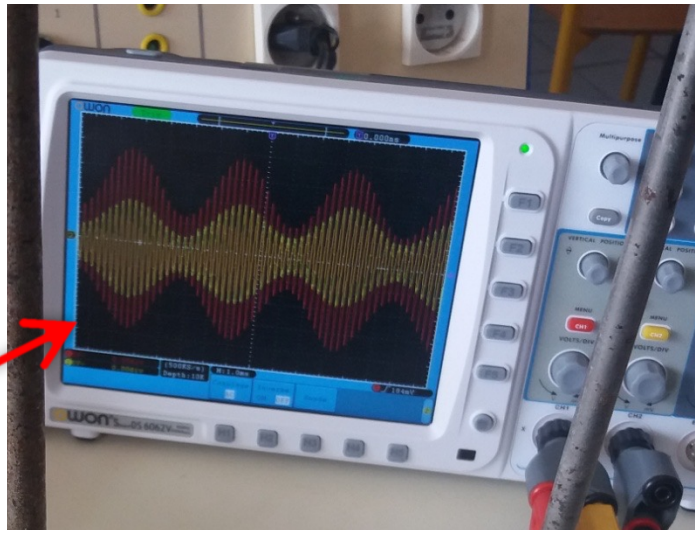
1er Cas: Lorsque les antennes sont éloignées

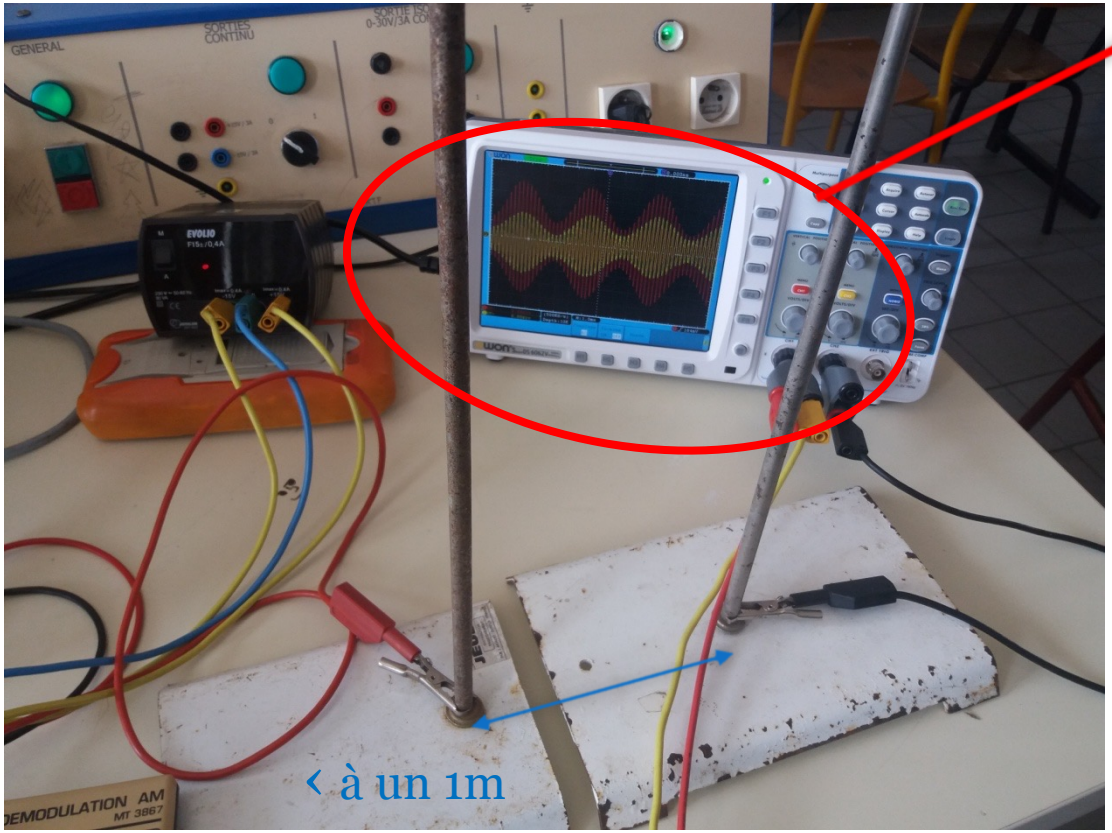




Antennes éloignées = Amplitude de l'antenne réceptrice faible

2e Cas: Lorsque les antennes sont rapprochées





Antennes rapprochées = Augmentation de l'amplitude de l'antenne réceptrice

Protocol expérimental

Traiter les informations numériques et les communiquer au Pc via Bluetooth

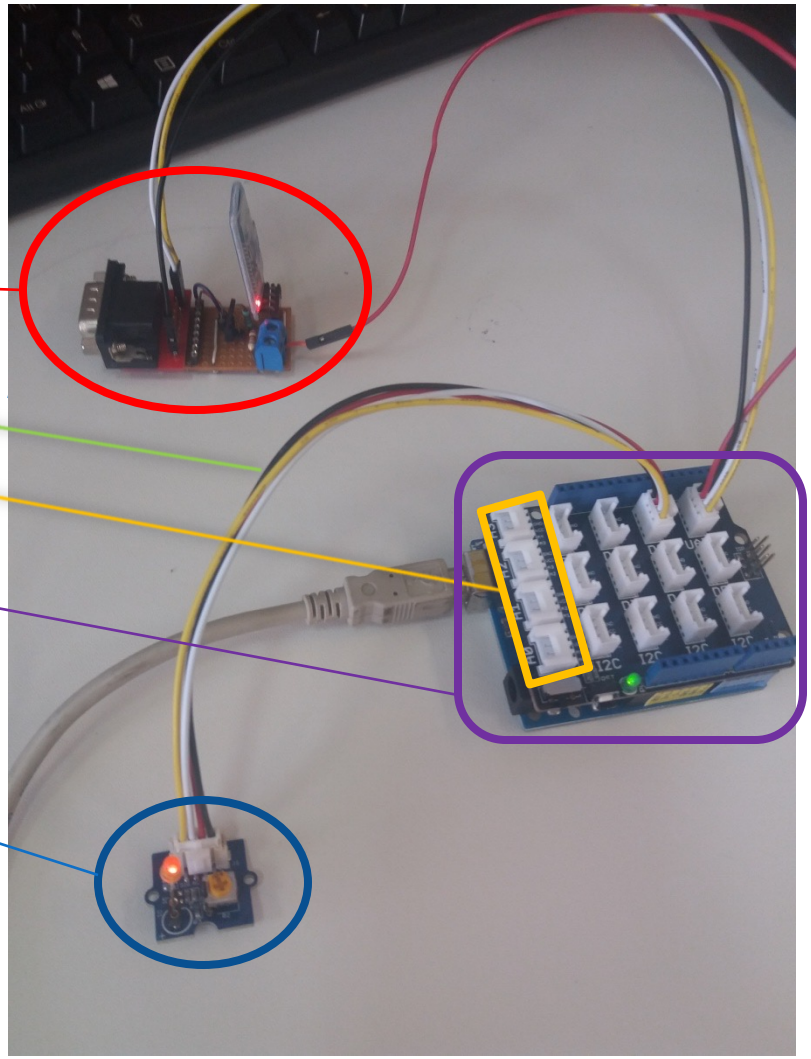
MATERIAUX:
- Module Bluetooth
- 4 cordons «grove»
- Une base shield «grove»
- 1 carte arduino uno
- Une Led

TESTS:
1) Avertir à l'utilisateur que le module Bluetooth est activé via une Led
2) Emettre et recevoir des données par Bluetooth avec l'application serial Bluetooth terminal

CONSTATS:
- Emission/ Réception des données via Bluetooth
- Affichage des valeurs numériques sur le clavier et l'ordinateur

Matériels utilisés

- Module Bluetooth
- 4 cordons «grove»
- Une base shield «grove»
- 1 carte arduino uno
- Une Led

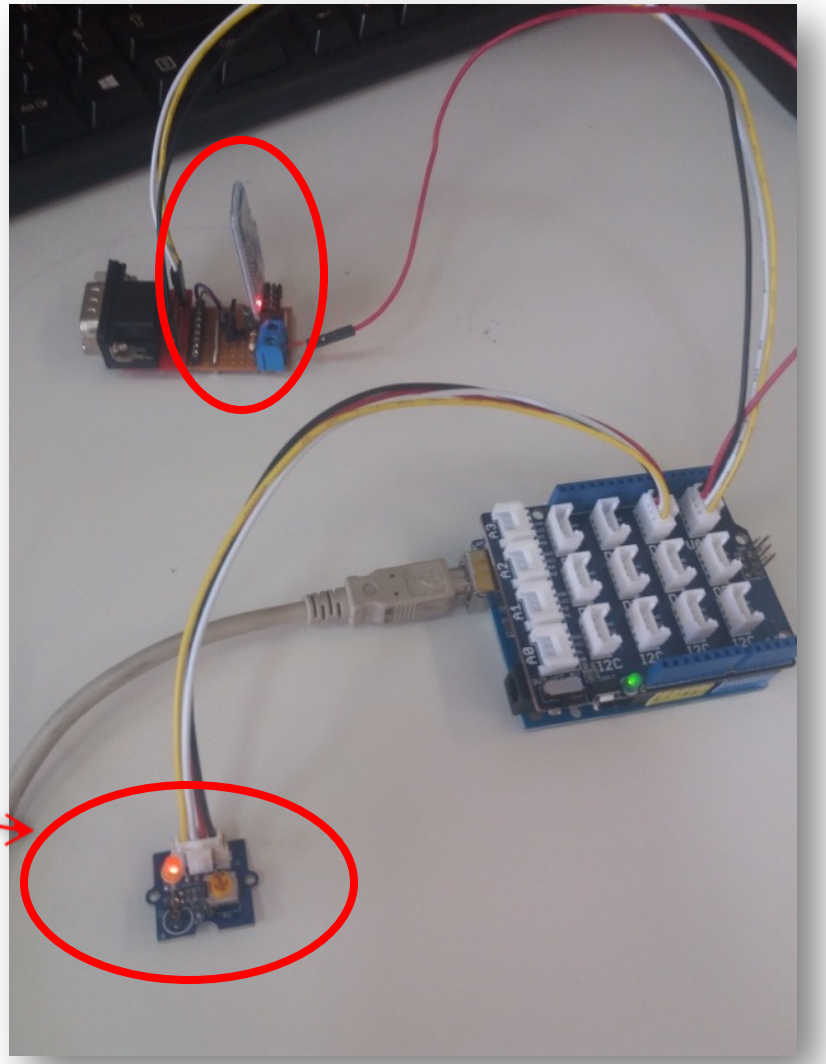


Simulation avec la LED

Blink §

```
  Most Arduinos have an on-board LED you can control. On the UNO, MEGA and ZERO
  it is attached to digital pin 13, on MKR1000 on pin 6. LED_BUILTIN takes care
  of use the correct LED pin whatever is the board used.
  If you want to know what pin the on-board LED is connected to on your Arduino model, check
  the Technical Specs of your board  at https://www.arduino.cc/en/Main/Products

  This example code is in the public domain.

  modified 8 May 2014
  by Scott Fitzgerald

  modified 2 Sep 2016
  by Arturo Guadalupi
*/

// Pin 2 has a LED connected to it
int LED = 2;


// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

Téléversement terminé



Ordinateur → Programme

SoftwareSerialHC06 | Arduino 1.6.12

Fichier Édition Croquis Outils Aide

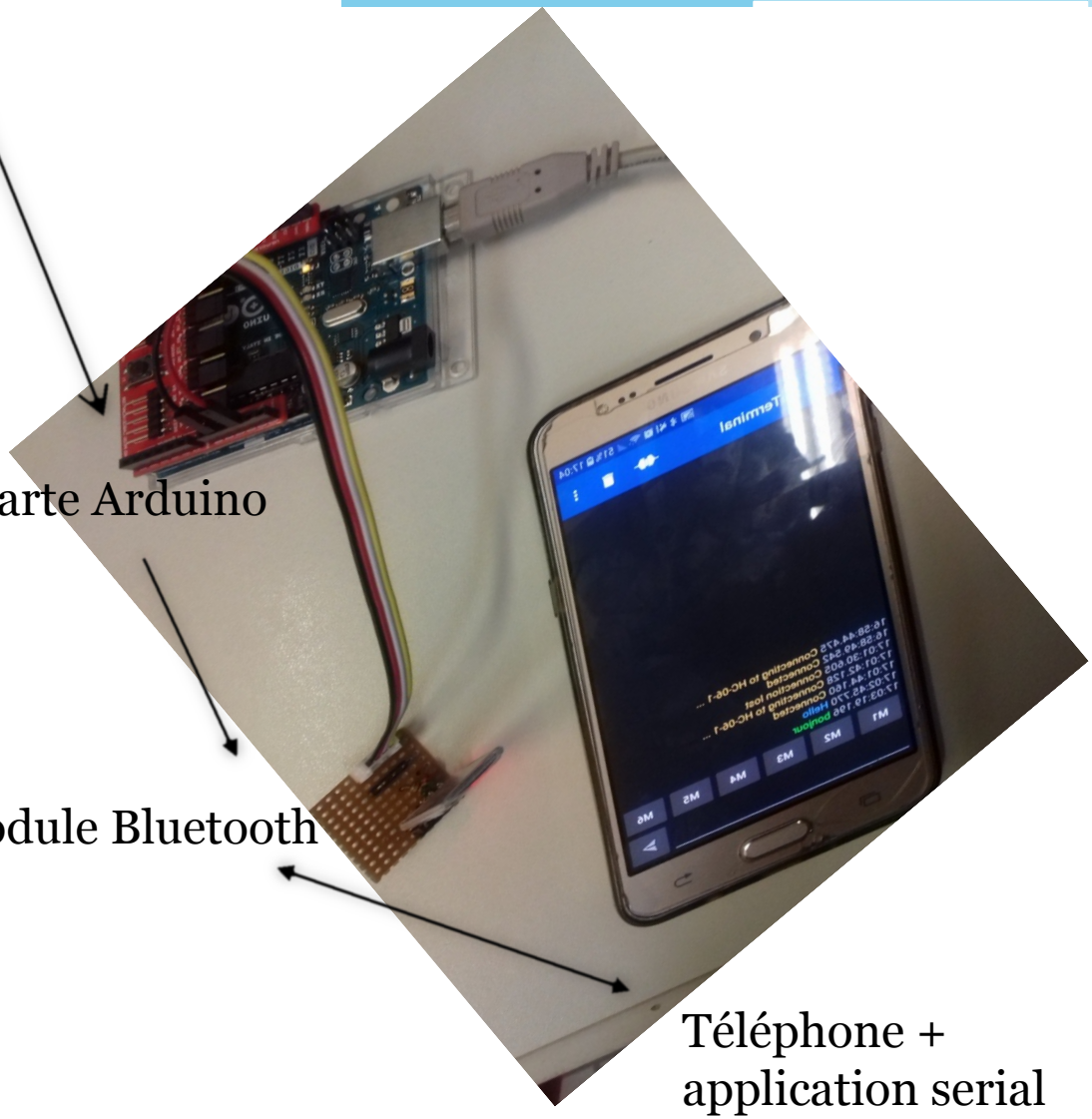
SoftwareSerialHC06

```
 by Tom Igoe
 based on Mikal Hart's example

 This example code is in the public domain.

 */
#include <SoftwareSerial.h>
SoftwareSerial HC06(10, 11); // RX, TX


void setup() {
  // Open serial communications and wait for port to open:
 HC06.begin(9600);
 Serial.begin(9600);
 Serial.println("Goodnight the moon!");
}

void loop() {
  if (HC06.available()) {
    Serial.write(HC06.read());
    }

  if (Serial.available()) {
    HC06.write(Serial.read());
  }
}
```

Téléversement terminé

```
Le croquis utilise 3 174 octets (9%) de l'espace de stockage de
Les variables globales utilisent 321 octets (15%) de mémoire dyn
```

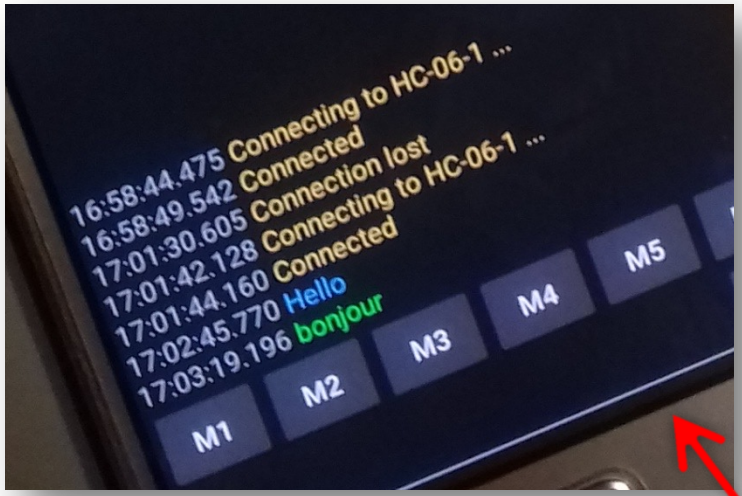
35

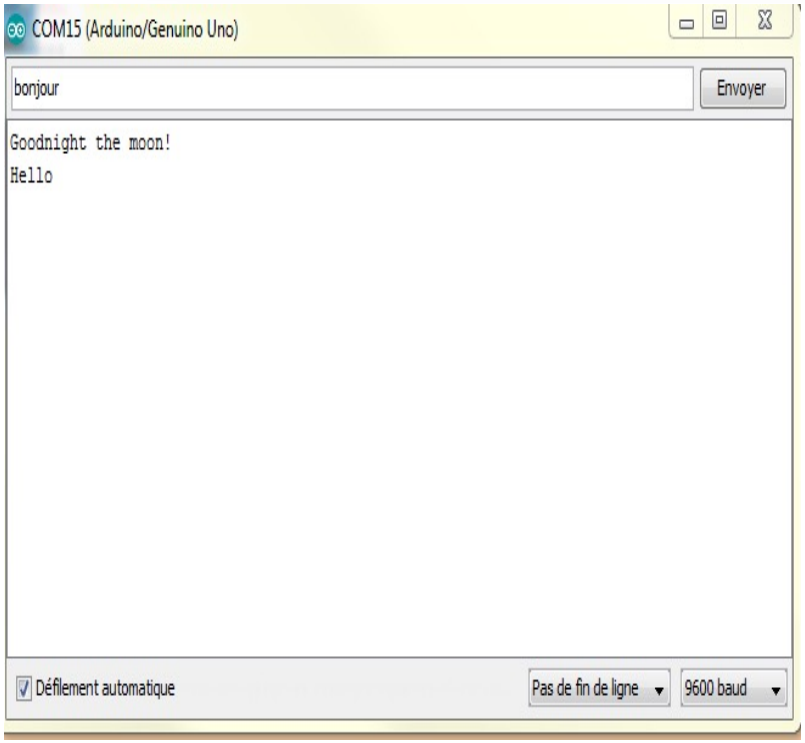


Carte Arduino

Module Bluetooth

Téléphone + application serial Bluetooth terminal


16:58:44.475 Connecting to HC-06-1 ...
16:58:49.542 Connected
17:01:30.605 Connection lost
17:01:42.128 Connecting to HC-06-1 ...
17:01:44.160 Connected
17:02:45.770 Hello
17:03:19.196 bonjour
M1
M2
M3
M4
M5


COM15 (Arduino/Genuino Uno)
bonjour
Envoyer
Goodnight the moon!
Hello
Défilement automatique
Pas de fin de ligne
9600 baud

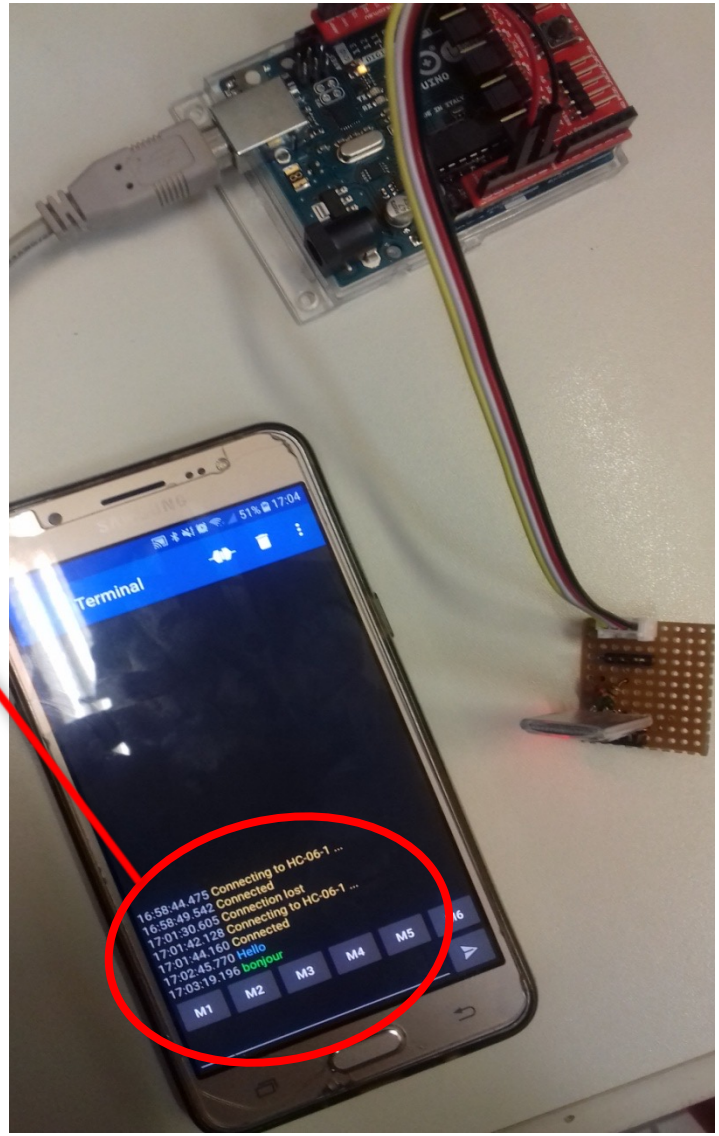
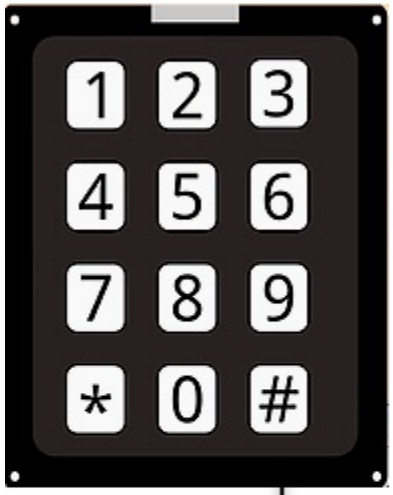

Terminal
16:58:44.475 Connecting to HC-06-1 ...
16:58:49.542 Connected
17:01:30.605 Connection lost
17:01:42.128 Connecting to HC-06-1 ...
17:01:44.160 Connected
17:02:45.770 Hello
17:03:19.196 bonjour
M1
M2
M3
M4
M5

Schéma de principe



Clavier à touches
Appuyer sur une touche,
Envoyer les informations sur la carte Arduino

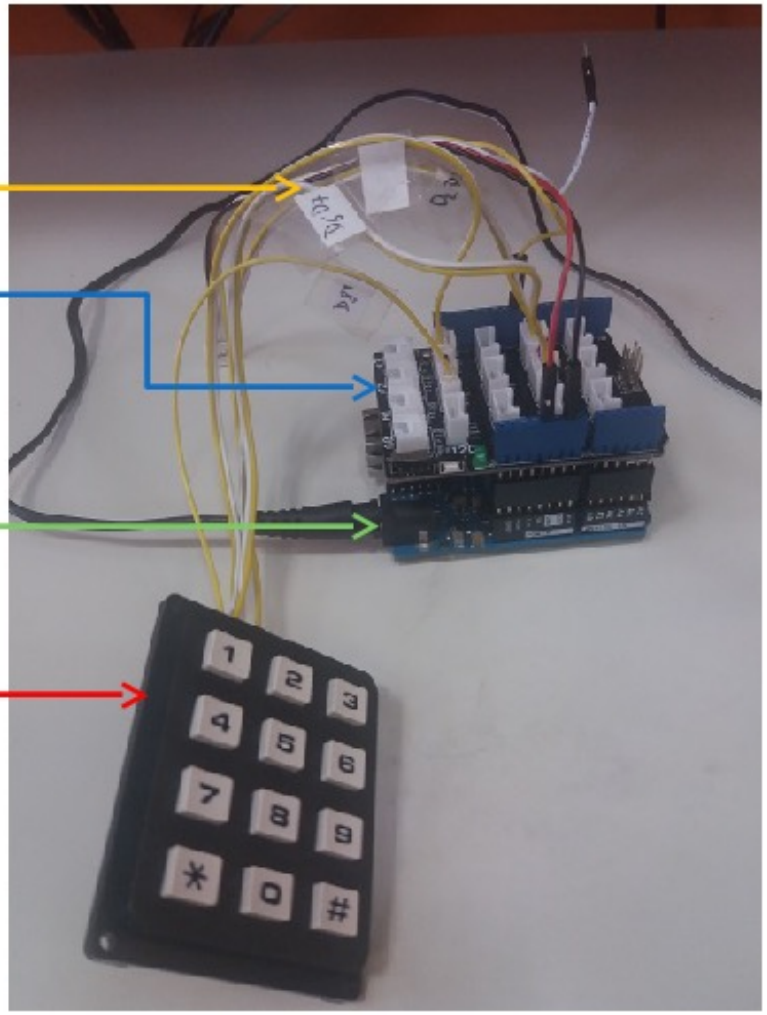
Ordinateur
Recevoir les informations des touches appuyées sur le clavier,
Afficher le symbole des touches appuyées sur le clavier

ARDUINO

Carte Arduino
Récupérer les informations des touches appuyées et les envoyer à l'ordinateur

Matériels utilisés

- 4 cordons « grove »
- Une base shield « grove »
- 1 arduino uno
- Module Clavier



Protocole suivi

- EXPERIMENTATIONS:

- A l'aide d'un cable USB, brancher la carte Arduino à l'ordinateur pour créer un programme.

- Ouvrir le logiciel de programmation Arduino et connecter la carte.

- Relier le module clavier avec la carte Arduino

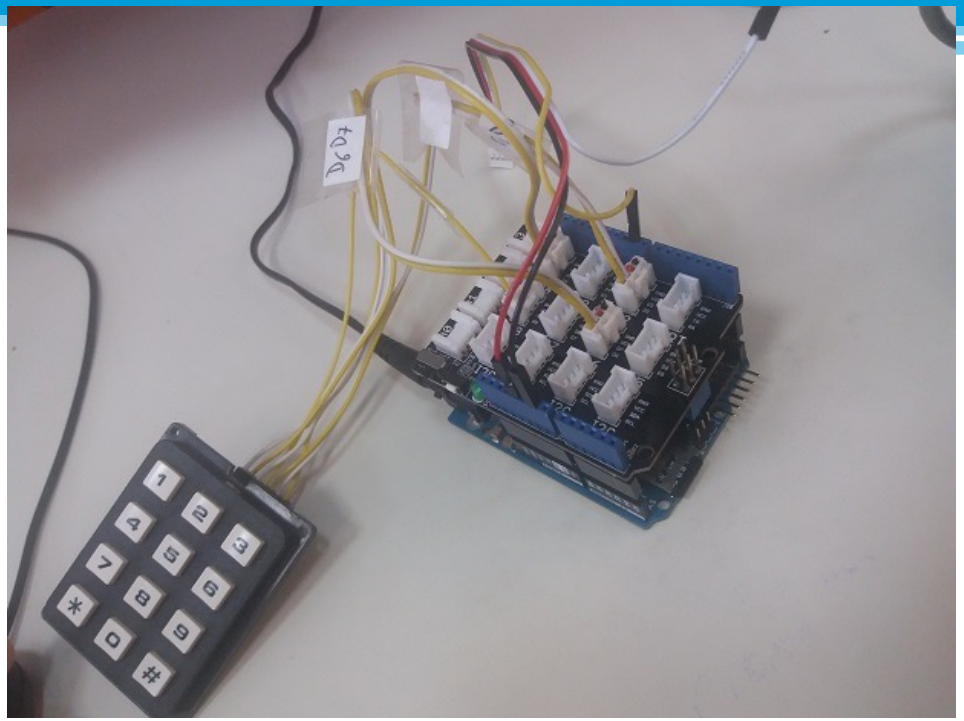
- Créer un programme qui permet à l'ordinateur de recevoir correctement les touches enfoncées du clavier

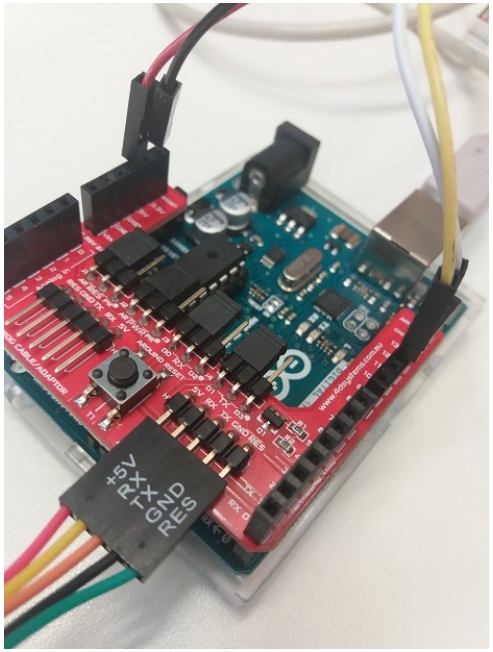
- Observer si elles sont envoyées dans le bon ordre ou si se sont les bons chiffres envoyés.

CONSTAT: Grâce à un programme adapté, la réception de chiffres d'un clavier est possible.

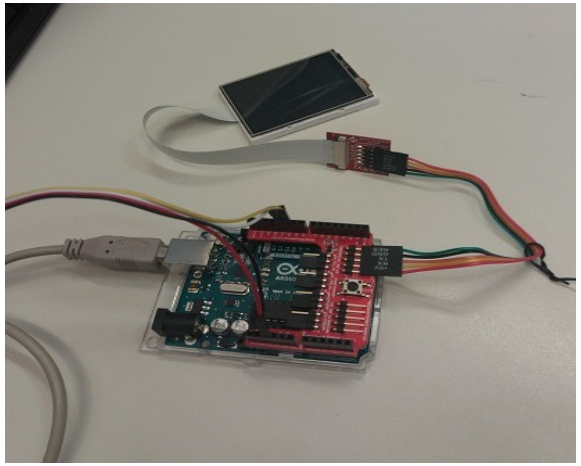
```
#include <Keypad.h>
const byte ROWS = 4; //four rows
const byte COLS = 3; //three columns
char keys[ROWS][COLS] = {
{'1','2','3'},
{'4','5','6'},
{'7','8','9'},
{'*','0','#'}
};
byte rowPins[ROWS] = {5, 6, 7, 8}; //connect to the row pinouts of the keypad
byte colPins[COLS] = {2, 3, 4}; //connect to the column pinouts of the keypad

char code[4] = {'1', '2', '3', '4'}; //connect to the row pinouts of the keypad
int chiffre = 0;
boolean bon = false;

Keypad keypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS );

void setup(){
  Serial.begin(9600);
  pinMode(9, OUTPUT);
}

void loop()
  char key = keypad.getKey();
  if (key !=NO_KEY){
    Serial.println(key);
  }
}
```

Module clavier
à touches





Module clavier tactile



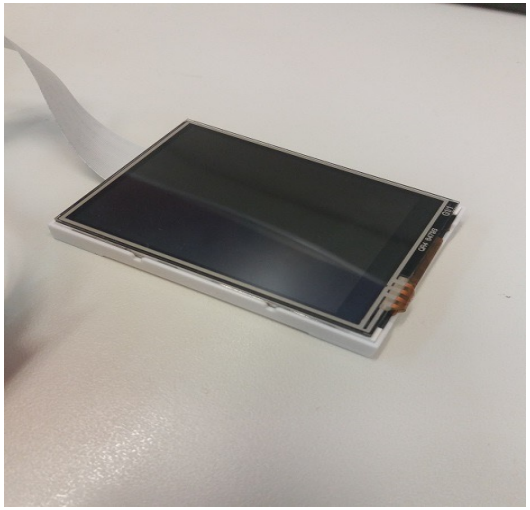
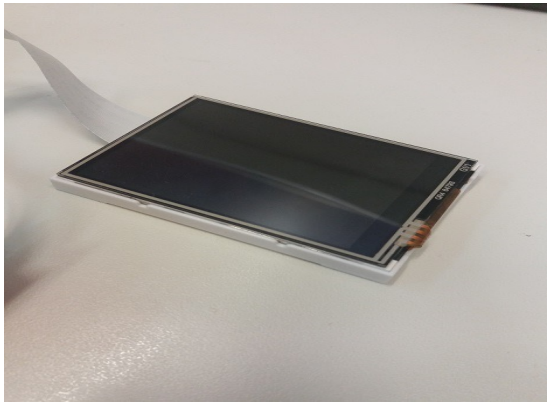
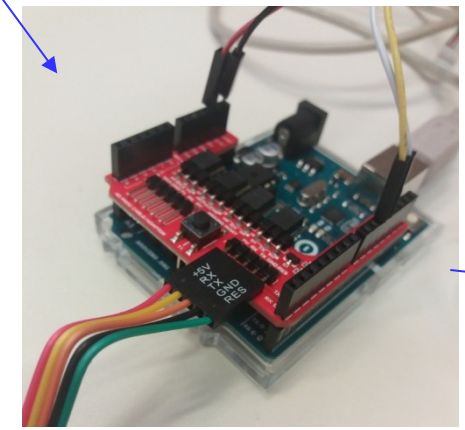


Schéma de principe



Connection Bluetooth

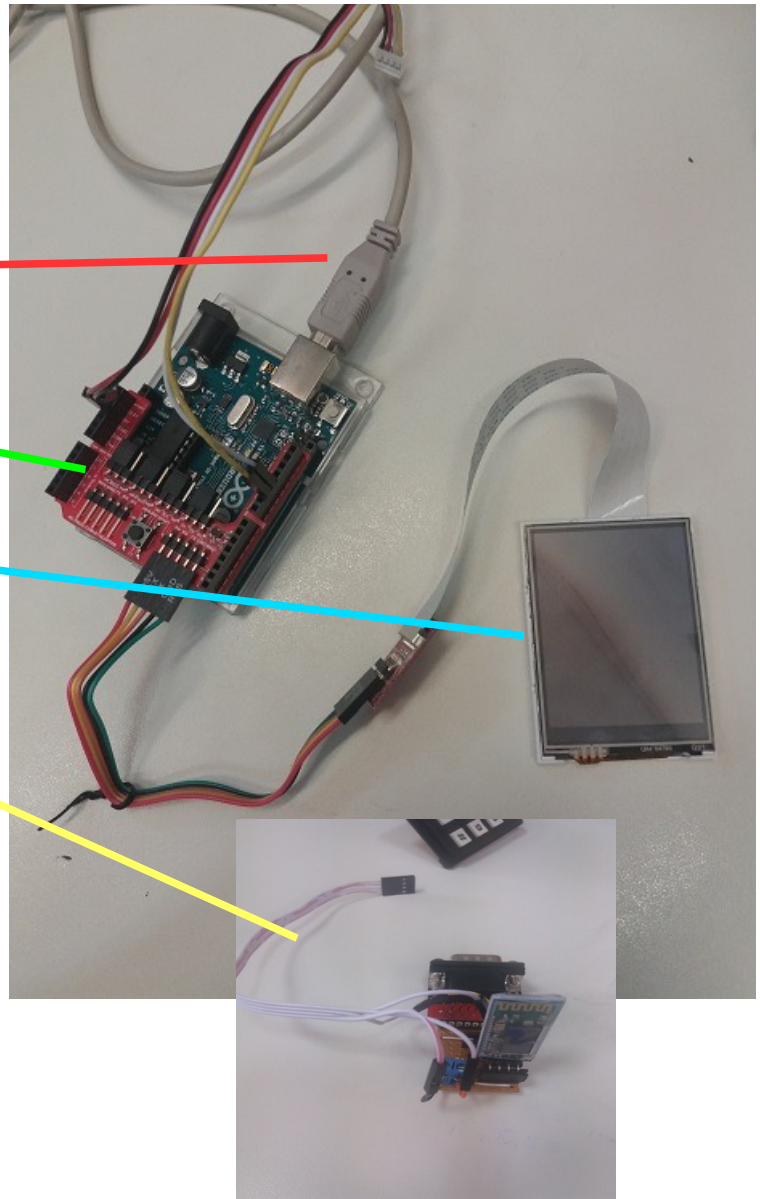
Connection filaire par bus série numérique (Rx,Tx)



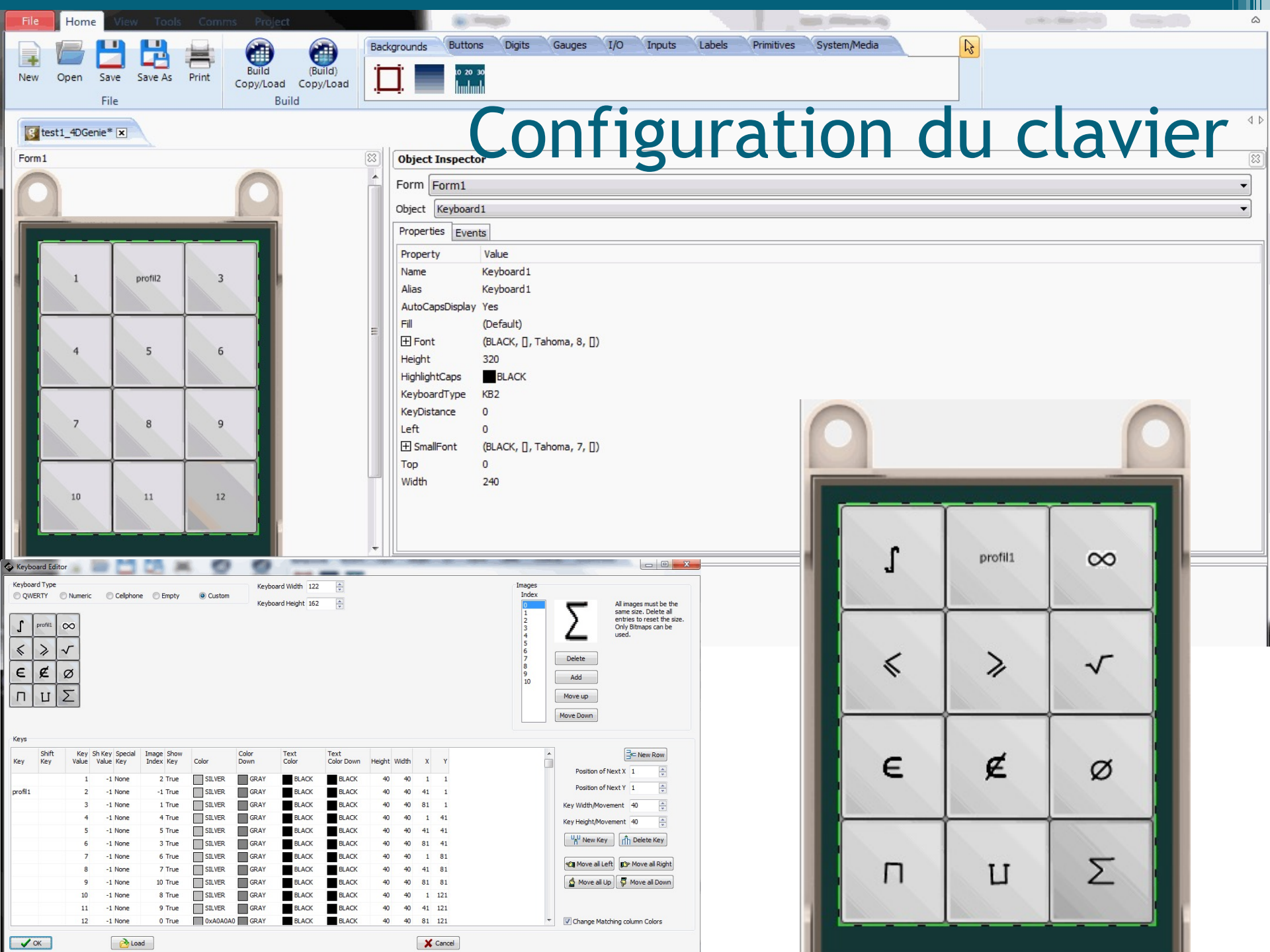
Connection filaire

Matériels utilisés

- Un cable USB
- Une carte Arduino Uno
- Un module clavier tactile
- Module Bluetooth



Configuration du clavier



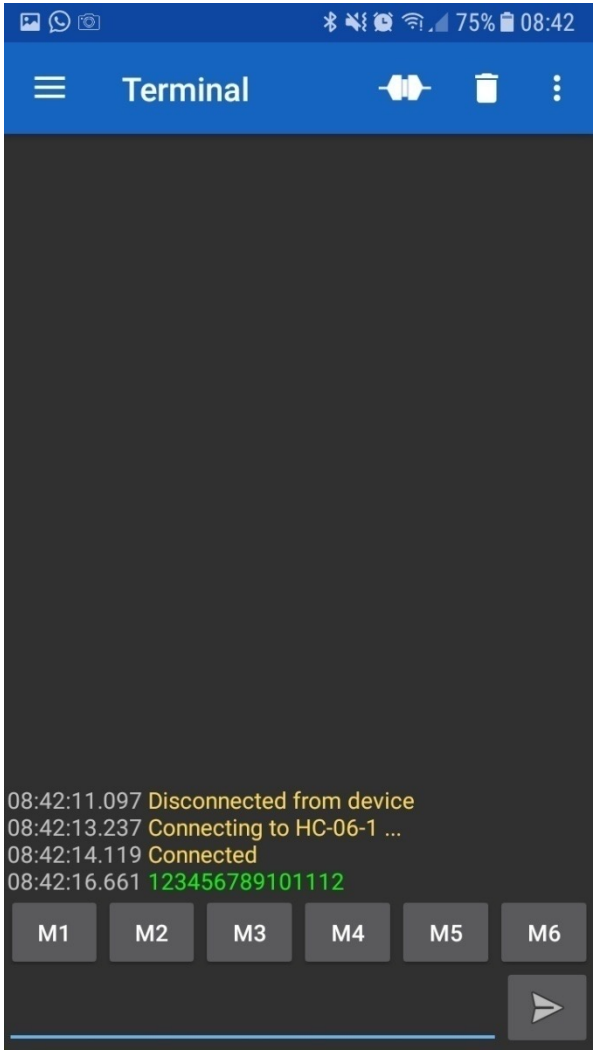
programme_calvier_tactile

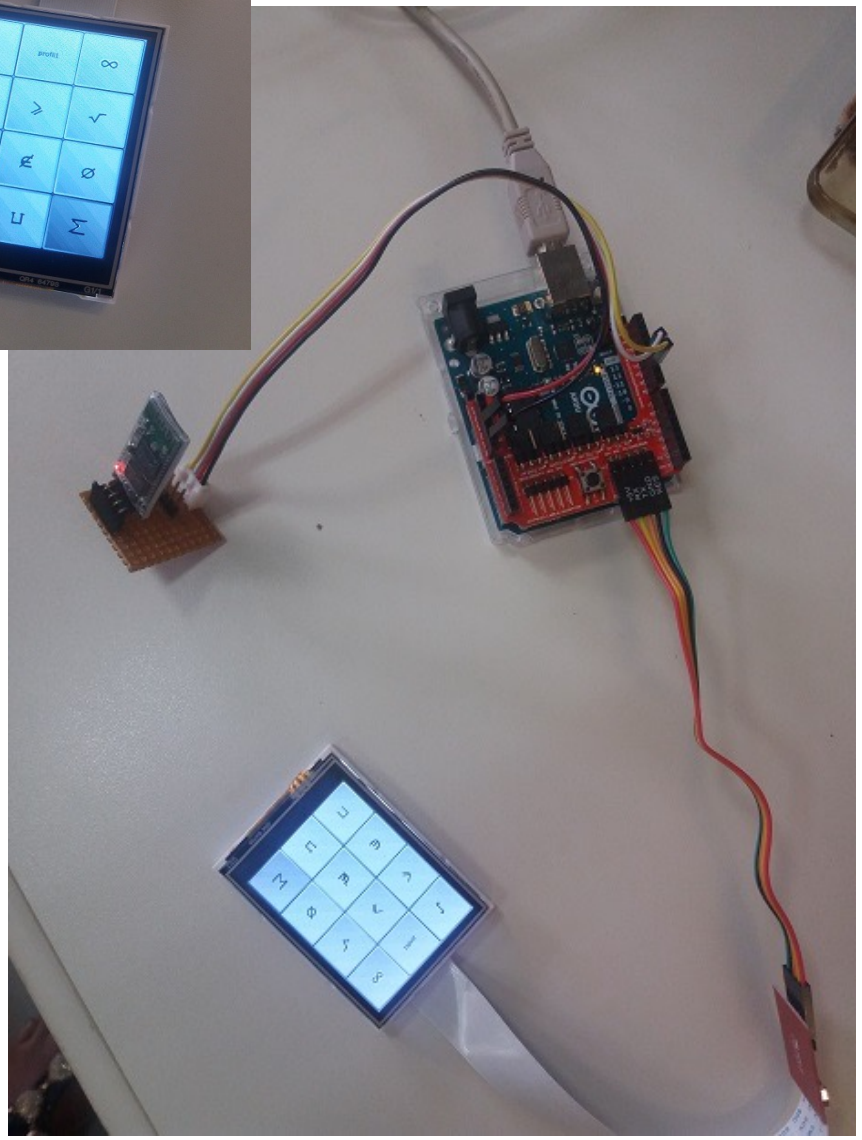
```
/*
Communication avec le clavier GEN4-ULCD-24DT configure avec visi-genie
*/

#include <genieArduino.h>
#include <SoftwareSerial.h>

#define RESETLINE 4  // Change this if you are not using an Arduino Adaptor Shield Version 2 (see code below)
#define profil1 0
#define profil2 1
#define profil3 2
#define profil4 3

#define LED 7   // la LED est connectee a la broche 7

char profil = '1';
Genie genie;  // genie est un objet de type clavier GEN4
SoftwareSerial hc06Serial(10, 11); // hc06Serial est une liaison série logicielle sur les broches RX=10, TX=11

void setup(){
  pinMode(LED, OUTPUT);      // configuration broche LED en sortie

// configuration communication série avec le bluetooth
  hc06Serial.begin(9600);   // set the data rate for the SoftwareSerial with module HC06

// configuration communication série avec le clavier GEN4-ULCD-24DT
  Serial.begin(9600);  // Serial0 9600 Baud
  genie.Begin(Serial);   // Use Serial0 for talking to the Genie Library, and to the 4D Systems display

  genie.AttachEventHandler(myGenieEventHandler); // Attach the user function Event Handler for processing events
  pinMode(RESETLINE, OUTPUT);  // Set D4 on Arduino to Output (4D Arduino Adaptor V2 - Display Reset)
  digitalWrite(RESETLINE, 1);  // Reset the Display via D4
  delay(100);
  digitalWrite(RESETLINE, 0);  // unReset the Display via D4
  delay (3500); //let the display start up after the reset (This is important)

  // Set the brightness/Contrast of the Display - (Not needed but illustrates how)
  // Most Displays, 1 = Display ON, 0 = Display OFF. See below for exceptions and for DIABLO16 displays.
  // use 0-15 for Brightness Control, where 0 = Display OFF, though to 15 = Max Brightness ON.
  genie.WriteContrast(14);
  genie.WriteObject(GENIE_OBJ_FORM, profil1, 0);  // on fixe le profil 1 par defaut
  }
```

programme_calvier_tactile

```
  // Most Displays, 1 = Display ON, 0 = Display OFF. See below for exceptions and for DIABLO16 displays.
  // use 0-15 for Brightness Control, where 0 = Display OFF, though to 15 = Max Brightness ON.
  genie.WriteContrast(14);
  genie.WriteObject(GENIE_OBJ_FORM, profil1, 0);  // on fixe le profil 1 par defaut
  }

void loop(){
  genie.DoEvents(); // This calls the library each loop to process the queued responses from the display

  if (hc06Serial.available()){
    profil=hc06Serial.read();
    if (profil=='1'){
      genie.WriteObject(GENIE_OBJ_FORM, profil1, 0);     // Write to keyboard the value of the profil GENIE_OBJ_FORM=10 , index of form , sans importance
    }
    else if (profil=='2'){
      genie.WriteObject(GENIE_OBJ_FORM, profil2, 1);
    }
    else if (profil=='3'){
      genie.WriteObject(GENIE_OBJ_FORM, profil3, 2);
    }
    else if (profil=='4'){
      genie.WriteObject(GENIE_OBJ_FORM, profil4, 3);
    }
    }
  }


void myGenieEventHandler(void){
  /*************************************************************************************************
  Event.reportObject.cmd is used to determine the command of that event, such as an reported event
  Event.reportObject.object is used to determine the object type, such as a Slider
  Event.reportObject.index is used to determine the index of the object, such as Slider0
  genie.GetEventData(&Event) us used to save the data from the Event, into a variable.
  *************************************************************************************************/
  genieFrame Event;
  genie.DequeueEvent(&Event); // Remove the next queued event from the buffer, and process it below

  int keyboard_val = 0;

  //If the cmd received is from a Reported Event (Events triggered from the Events tab of Workshop4 objects)
  if (Event.reportObject.cmd == GENIE_REPORT_EVENT){
    if (Event.reportObject.object == GENIE_OBJ_KEYBOARD){            // If the Reported Message was from the keyboard
      keyboard_val = genie.GetEventData(&Event);                     // Receive the event data from the keyboard
      keyboard_val = keyboard_val + 48;                               // transforme la valeur numérique en valeur ASCII pb! unicode!
      hc06Serial.write("∫");
    }
```

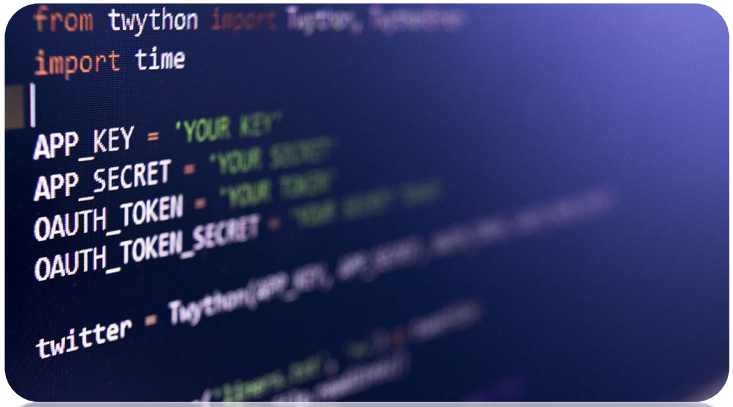
RENDU


75% 08:42
Terminal
08:42:11.097 Disconnected from device
08:42:13.237 Connecting to HC-06-1 ...
08:42:14.119 Connected
08:42:16.661 123456789101112
M1
M2
M3
M4
M5
M6



Bluetooth®

Application part



python

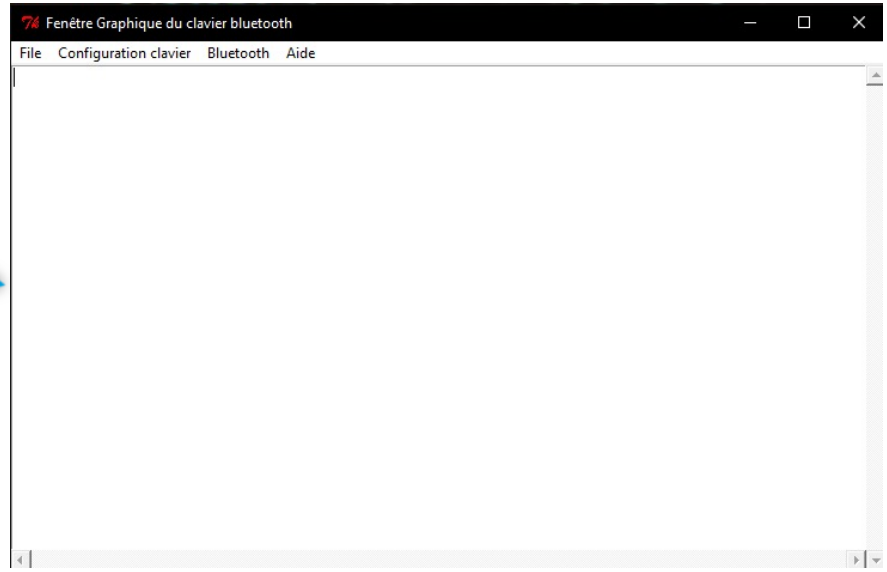
Analyse du besoin

- Application de communication avec un clavier Bluetooth permettant la saisie de caractères mathématiques (langage Python, Modules Tkinter, Pyserial)

- A partir d'un clavier connecté en Bluetooth avec l'ordinateur, récupérer le code des touches tapées et intégrer les symboles mathématiques dans la zone texte de l'application.

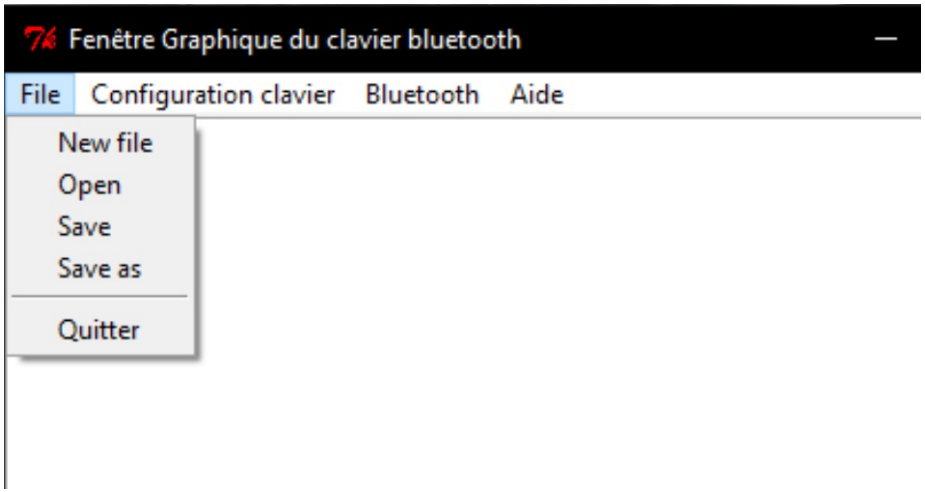
```python
# Le programme principal
fen1 = Tk() # Création de la fenêtre principale
fen1.title("Fenêtre Graphique du clavier bluetooth") # Titre ecrit dans la fenetre
fen1.config(bg='#ffffff') # Couleur de fond
fen1.geometry("400x200+400+100") # Dimension fenêtre (largeur x hauteur et postion X + Y)
fen1.resizable(width=True, height=True)  # Fenêtre non redimensionnable
#fenetre.maxsize(width=500, height=200)
```

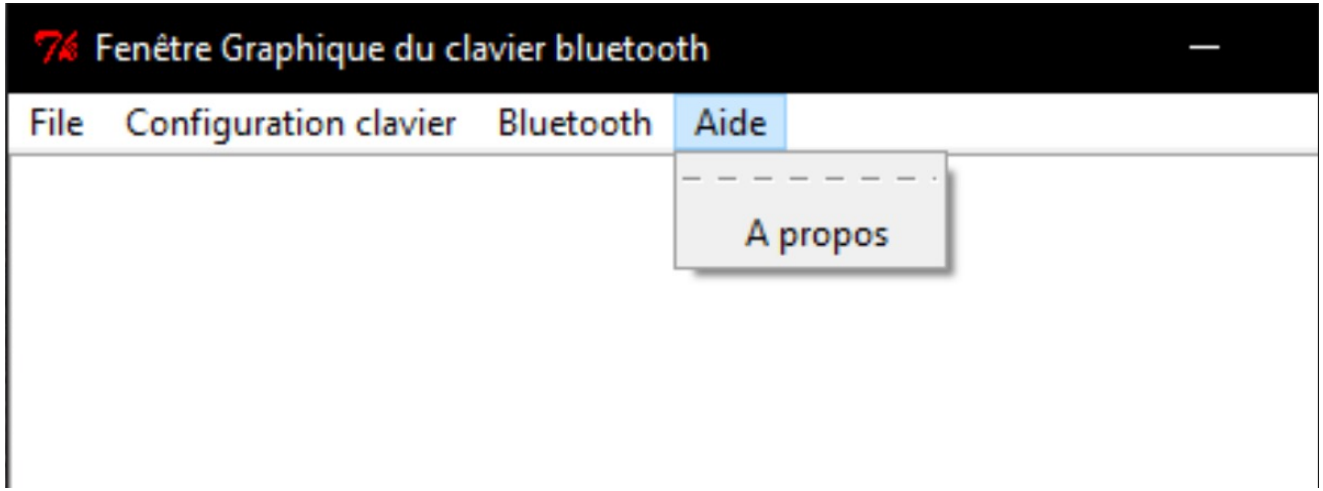
Programme principal qui crée la fenêtre




```python
#Ajout des menus fils a la barre de menus
mainMenu.add_cascade(label = "File", menu=menuExample)
mainMenu.add_cascade(label = "Configuration clavier", menu=menuConfiguration)
mainMenu.add_cascade(label = "Bluetooth", menu=menuBluetooth)
mainMenu.add_cascade(label = "Aide", menu=menuHelp)
```

Création des différents menu


Fenêtre Graphique du clavier bluetooth
File
Configuration clavier
Bluetooth
Aide
New file
Open
Save
Save as
Quitter


Fenêtre Graphique du clavier bluetooth
File
Configuration clavier
Bluetooth
Aide
A propos

On appelle notre fonction « configuration »

Qui nous permet d'envoyer une valeur

```
def configuration(valString):
    """
    valString : (str) : valeur sous forme de chaine à envoyer
    """
    global comEtablie
    global ser1

    if comEtablie==True:
        ser1.write(valString.encode())
        time.sleep(0.3)
```

```
menuConfiguration = Menu(mainMenu) #Menu fils
menuConfiguration.add_command(label="Profil 1", command=lambda :configuration("1")) #
menuConfiguration.add_command(label="Profil 2", command=lambda :configuration("2"))
menuConfiguration.add_command(label="Profil 3", command=lambda :configuration("3"))
menuConfiguration.add_command(label="Profil 4", command=lambda :configuration("4"))
```

De ce fait si on choisie le profil 3 dans notre menu déroulant on envoie la valeur 3

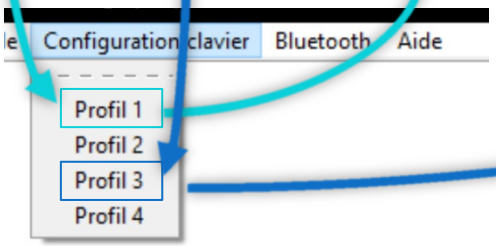
3

Sélection du profil 1

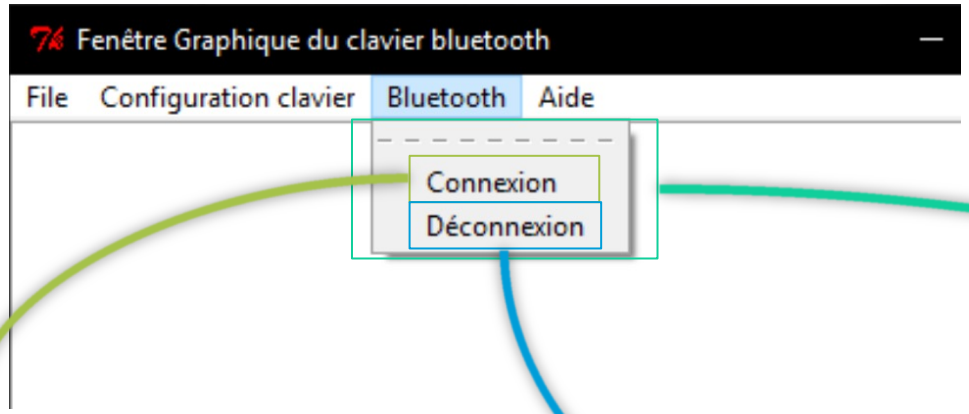
```
menuConfiguration = Menu(mainMenu) #Menu fils
menuConfiguration.add_command(label="Profil 1", command=lambda :configuration("1"))
menuConfiguration.add_command(label="Profil 2", command=lambda :configuration("2"))
menuConfiguration.add_command(label="Profil 3", command=lambda :configuration("3"))
menuConfiguration.add_command(label="Profil 4", command=lambda :configuration("4"))
```

Sélection du profil 3

Renvoie la valeur 1 qui correspond à une configuration clavier



Renvoie la valeur 3 qui correspond à une autre configuration clavier



Affichage du menu déroulant Connexion et Déconnexion

```
menuBluetooth = Menu(mainMenu)
menuBluetooth.add_command(label="Connexion", command=connecter)
menuBluetooth.add_command(label="Déconnexion", command=deconnecter)
```

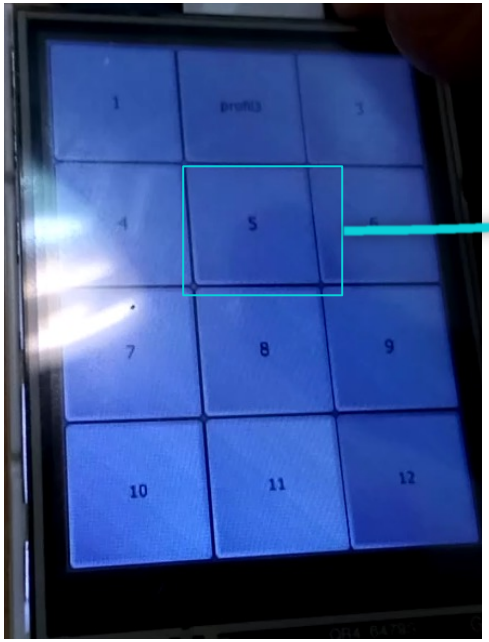
Utilisation du module Serial

```
def connecter():
    # Connection Ã  la cible
    global comEtablie
    global ser1

    try:
        ser1 = serial.Serial("COM8",timeout=1) #Configurer le com manuellement
        print "connexion etablie"

        comEtablie=True

    except serial.SerialException:
        print "erreur de connexion"
```

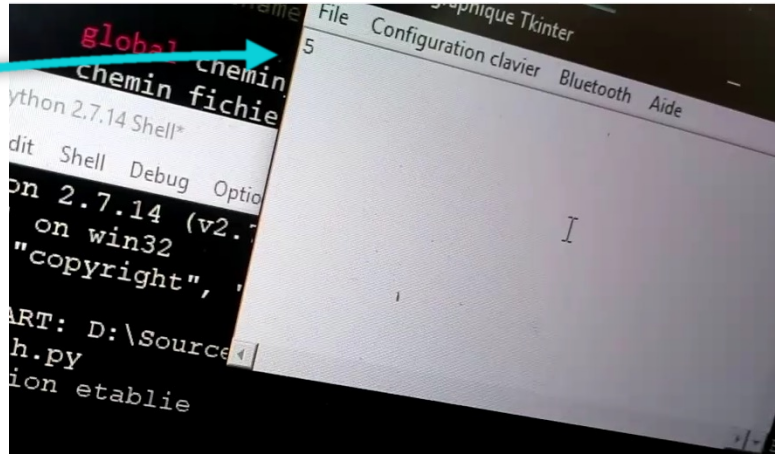
Connexion au port com

```
def deconnecter():
    # Deconnection de la cible
    global comEtablie
    global ser1

    ser1.close()

    comEtablie=False
```

Déconnexion en fermant la connexion com



Si on presse la touche 5
Du clavier la touche 5
S'affiche dans la zone
de texte



Si la connexion sur le port com est « True » et qu'elle n'est pas en attente

On viens lire la valeur reçue dans « valrecue » et on insert dans notre zone de texte

On fait une actualisation tout les 500miliseconde

```
def recevoir():
    # Recoit les données sur RXD
    global ser1
    global comEtablie

    if comEtablie == True:
        if ser1.inWaiting() != 0 :
            """
            valrecue = ser1.readline()    # Lecture de 10 octets max dans le buffer de reception
            # Valrecue = valrecue.decode('ascii')
            E1.delete(0, END)
            E1.insert(END,valrecue)
            """
            valrecue = ser1.readline()    # Lecture de 10 octets max dans le buffer de reception
            Valrecue = valrecue.decode('ascii').encode('utf-8')
            textW.insert(END,valrecue)
            """print valrecue"""
        else:
            pass
    else:
        pass
    fen1.after(500,recevoir) # Mise à jour toutes les 500 ms
```

CONCLUSION

REMERCIEMENTS ET SOURCES

Mme PALAYSI
M.BROUSSE
M.PROMENEUR
M.JOLIE

http://pecquery.wixsite.com/arduino-passion/le-clavier

http://pecquery.wixsite.com/arduino-passion/copie-de-le-module-joystick-1

http://pecquery.wixsite.com/arduino-passion/copie-de-app-inventor
http://pecquery.wixsite.com/arduino-passion/copie-de-le-detecteur-a-ultrasons-h-1

http://pecquery.wixsite.com/arduino-passion/la-liaison-serie