**Projet téléthon**

**Application web**

**Liste matériel pour développement web**

NetBeans : Pour coder en HTML/PHP/CSS

Bootstrap : pour adapter en mobile, pc, tablette

WampServer : Hébergement page web

Plugin PHP :

Apache : Serveur

MySQL : base de données

**Présentation de la page web**

**Formulaire de connexion :**

<http://www.codeurjava.com/2016/12/formulaire-de-login-avec-html-css-php-et-mysql.html>

**Permettre à l’administrateur et au superviseur de se connecter**

**En fonction du pseudo et mot de passe saisie, la personne sera redirigée vers une page dédiée**

* **Quatre étapes pour créer un formulaire de connexion :  
    
  1- Créer un fichier HTML contenant la structure du formulaire  
  2- Créer un fichier CSS pour définir le style  
  3- Créer un fichier de login PHP qui va vérifier les champs de formulaire s’ils correspondent aux valeurs de la table dans la base de données  
  4- Créer un fichier PHP de la page principale et de la déconnexion**

**Page de gestion des superviseurs**

Inscription, modification, suppression d’un superviseur :

Page de l’administrateur

Administrateur -> formulaire (ajout, modification et suppression de comptes superviseur)

**Page de gestion des créneaux**

Ajout, suppression des créneaux :

Page du superviseur

Superviseur -> formulaire (ajout, suppression de créneaux)

<https://www.grafikart.fr/tutoriels/gestion-membre-229>

<http://www.lephpfacile.com/cours/15-modifier-des-donnees-de-votre-base>

**Base de données**

Base de données pour administrateur :



Base de données pour administrateur :



Base de données pour créneaux :



**Enregistrements des créneaux**

**Application Arduino**

**Liste matériel pour développement sur Arduino**

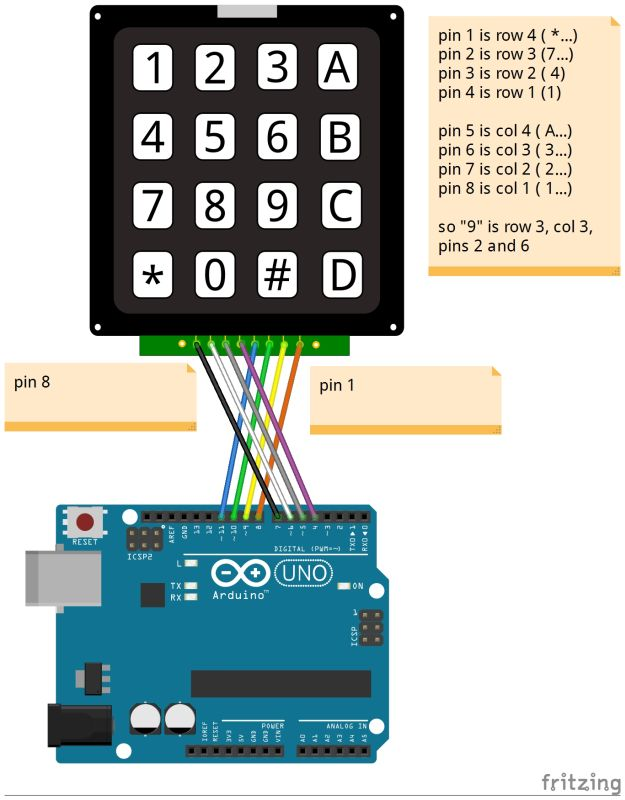
* Arduino
* Pavé numérique
* Buzzer
* Pavé numérique
* Écran LCD

**Installer logiciel Arduino sur pc Windows :**

<https://www.arduino.cc/en/Guide/Windows>

**Connecter pavé numérique sur Arduino :**

<https://knowledge.parcours-performance.com/clavier-projet-arduino/>

Câblage du pavé numérique :

Programme de connexion :

#include <Keypad.h>

const byte ROWS = 4; //four rows

const byte COLS = 4; //four columns

//define the cymbols on the buttons of the keypads

char keys[ROWS][COLS] = {

{'1', '2', '3', 'A'},

{'4', '5', '6', 'B'},

{'7', '8', '9', 'C'},

{'\*', '0', '#', 'D'}

};

byte rowPins[ROWS] = {11, 10, 9, 8}; //rows 1 to 4

byte colPins[COLS] = {7, 6, 5, 4}; //columns 4 to 1

//initialize an instance of class NewKeypad

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

void setup(){

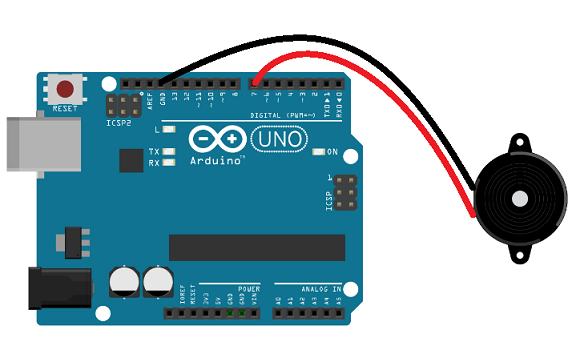
Serial.begin(9600);

Serial.println("Setup done");

}

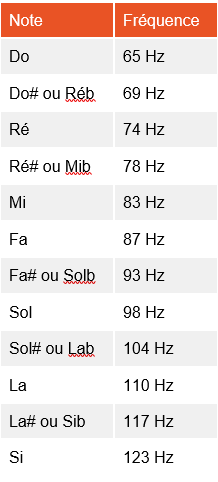
**Connecter buzzer sur Arduino :**

Câblage du buzzer :



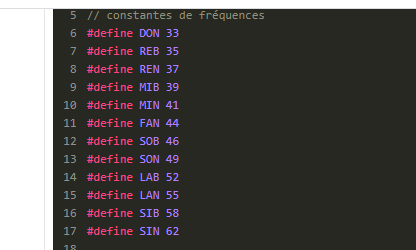
Fréquence :

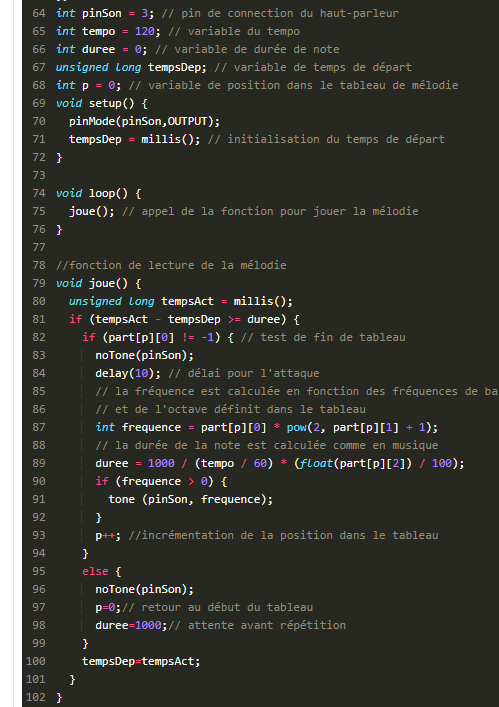
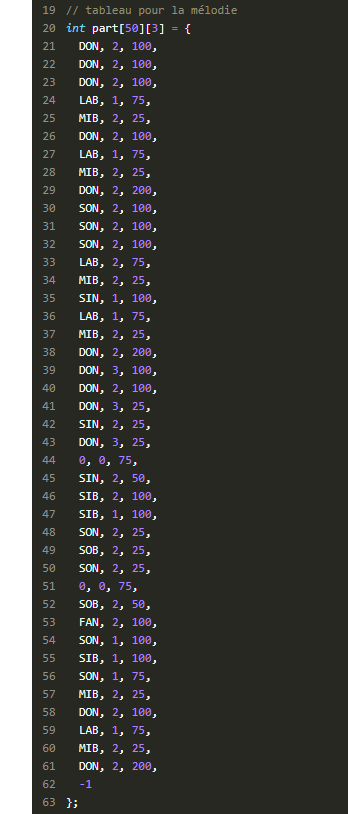
Note de musique au clair de la lune :



Sol sol sol la si la sol si la la sol  
Sol sol sol la si la sol si la la sol  
La la la la mi mi  
La sol fa mi ré  
Sol sol sol la si la sol si la la sol

Code pour faire jouer la musique :



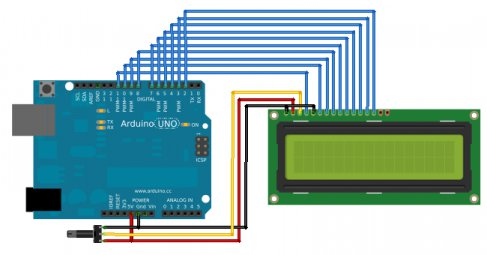


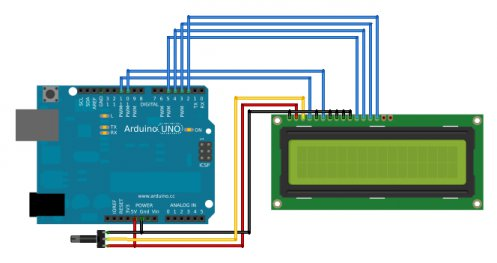
<https://openclassrooms.com/fr/courses/3290206-perfectionnez-vous-dans-la-programmation-arduino/3295331-generez-des-sons>

<https://dyrk.org/2016/11/04/tuto-jouer-la-reine-des-neiges-avec-un-arduino/>

**Afficher énigme sur écran LCD**

Câblage de l’écran LCD à l’Arduino :

8 fils :

4 fils :

Code de connexion

#include "LiquidCrystal.h" *// ajout de la librairie*

*// Vérifiez les broches !*

LiquidCrystal lcd(11,10,9,8,7,6,5,4,3,2); *// liaison 8 bits de données*

LiquidCrystal lcd(11,10,5,4,3,2); *// liaison 4 bits de données*

**void** setup()

{

lcd.begin(16,2); *// utilisation d'un écran 16 colonnes et 2 lignes*

lcd.write(**"Salut ca zeste ?"**); *// petit test pour vérifier que tout marche*

}

**void** loop() {}

**Code permettant d’activer le buzzer et d’afficher l’énigme sur l’écran**

* Librairie :

#include <Password.h>

#include <Keypad.h>

* Définir le mot de passe
* Configuration du clavier
* Fonction permettant de vérifier le code et d’activer le buzzer et d’afficher l’énigme

#include <Password.h> // http://www.arduino.cc/playground/uploads/Code/Password.zip  
#include <Keypad.h>   // http://www.arduino.cc/playground/uploads/Code/keypad.zip  
  
  
Password pwd = Password("1234");   // définition du mot de passe  
  
boolean alarm = false;             // variable stockant l'état de l'alarme  
                                   // false = alarme OFF | true = alarme ON  
boolean intruder = false;          // variable stockant l'état de l'intrusion  
                                   // false = pas d'intrus détecté | true = intrus détecté  
  
#define ROWS 4                                                       //  
#define COLS 3                                                       //  
char keys[ROWS][COLS] =                                              //  
{                                                                    //  
  {'1','2','3'},                                                     //  
  {'4','5','6'},                                                     // configuration  
  {'7','8','9'},                                                     // du clavier  
  {'\*','0','#'}                                                      //  
};                                                                   //  
byte rowPins[ROWS] = {8, 7, 6, 5};                                   //  
byte colPins[COLS] = {4, 3, 2};                                      //  
                                                                     //  
Keypad kpd = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS); //  
  
void setup()  
{  
  pinMode(red, OUTPUT);  
  pinMode(green, OUTPUT);  
   
  kpd.addEventListener(kpdEvent);  
}  
  
void loop()  
{  
  char Key = kpd.getKey();  
  // en cas d'intrus détecté :  
  if(intruder == true && alarm == true)  
  {  
    unsigned long currentMillis = millis();  
    // ... on regarde s'il est temps de faire clignoter la LED, c'est-à-dire  
    // s'il s'est passé suffisamment de temps depuis la dernière fois qu'on a vérifié  
    if(currentMillis - previousMillis > interval)  
    {  
      previousMillis = currentMillis;  
      digitalWrite(red, !digitalRead(red));  
    }  
  }  
  else  
  {  
    digitalWrite(red, LOW);  
  }  
}  
  
void kpdEvent (KeypadEvent Key)  
{  
  switch (kpd.getState())  
  {  
    case PRESSED :  
      switch (Key)  
      {  
        // appui sur '\*' -> vérification de la saisie en cours  
        case '\*' : checkPassword(); break;  
        // appui sur '#' -> réinitialisation de la saisie en cours  
        case '#' : pwd.reset(); break;  
        // sinon on ajoute le chiffre à la combinaison  
        default  : pwd.append(Key); break;  
      }  
    default : break;  
  }  
}  
  
void checkPassword(void)  
{  
  // on remet à zéro l'état du mot de passe  
  intruder = false;  
  // si le mot de passe est juste...  
  if (pwd.evaluate())  
  {  
    // ...on met à jour l'état de l'alarme : ON>OFF / OFF>ON  
    alarm = !alarm;  
     
    switch (alarm)  
    {  
      case true :  
      // la LED rouge signale la mise en service de l'alarme  
        digitalWrite(red, HIGH);  
        delay(2000);  
        digitalWrite(red, LOW);  
        pwd.reset();  // on remet à zéro la saisie  
        break;  
      case false :  
      // la LED verte signale l'extinction de l'alarme  
        digitalWrite(green, HIGH);  
        delay(2000);  
        digitalWrite(green, LOW);  
        pwd.reset();  // on remet à zéro la saisie  
        break;  
      default :  
        break;  
    }  
  }  
  // si le mot de passe est faux...  
  else  
 {  
   // ...on signale l'intrusion à la loop() qui déclenche l'alarme  
   // jusqu'à ce que le bon mot de passe soit rentré  
   intruder = true;  
 }  
 // on remet à zéro systématiquement après avoir vérifié pour ne pas avoir d'erreur  
 pwd.reset();  
}

[http://forum.snootlab.com/viewtopic.php?f=43&t=671#](http://forum.snootlab.com/viewtopic.php?f=43&t=671)