Projets Arduino PeiP2
Année scolaire 2019-2020

"Knock, Knock, Knock... Who's there ?"

Étudiants : RIGOMIER Lise
DJIAN-MARTIN Nathan

Encadrant : MASSON Pascal

Sommaire

INTRODUCTION

Étudiants en PeiP2 à Polytech Nice-Sophia, nous vous proposons notre projet d'électronique utilisant la carte Arduino Uno et les modules associés. Pendant plusieurs semaines, nous avons créé et tenté d'innover la manière dont nous déverrouillons nos portes au quotidien. Du 9 décembre au 9 mars 2020, nous avons réalisé Knocki. C'est une nouvelle façon d'ouvrir et de fermer une porte, idéale pour les escape games, ces salles où vous avez exactement 60 minutes pour résoudre des énigmes et vous évader. Les escape games étant très à la mode, cela fait une quantité importante de clients potentiels. Ouvrez la porte de Knocki et laissez-vous embarquer dans l'envers du décors.

Le compte à rebours est lancé !

CHAPITRE 1 : Knocki, les origines

1.1. Le projet en arduino

Le projet Arduino est obligatoire en PeiP2 à Polytech Nice-Sophia. Nous avions 8 séances, soit 24h (plus les heures chez nous), pour créer notre projet en utilisant les composants Arduino. De plus, nous avions l'obligation d'utiliser une connexion radiofréquence.
Le choix du projet était libre.

1.2. L'idée

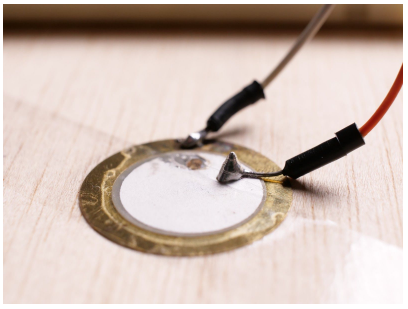
Nous sommes de grands adeptes d'escape games. Avec de nombreuses salles à notre compteur, il y a des types d'énigmes qui reviennent et les portes s'ouvrent quasiment tout le temps à l'aide de cadenas ou de clés. Nous trouvions cet aspect dommage et nous nous sommes donc demandés s'il n'existait pas d'autres moyens plus ludiques de déverrouiller une porte. En réfléchissant nous avons cherché un moyen d'ouvrir une porte en frappant dessus une combinaison secrète. En nous documentant sur Internet nous avons trouvé des projets déjà existant qui utilisaient ce principe. Nous avons regardé dans les grandes lignes les

projets trouvés sur Internet car nous voulions garder une certaine impartialité dans nos recherches de solutions.

CHAPITRE 2 : Les composants

2.1. Le capteur piézoélectrique

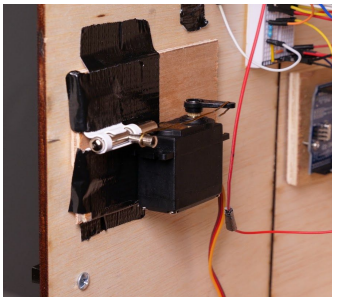
Le capteur piézoélectrique est le composant qui nous permet de détecter les coups de l'utilisateur.
En effet, le principe du capteur piézoélectrique est d'envoyer un signal électrique lorsqu'il subit une déformation. Ainsi, quand on va frapper sur notre porte, cela va déformer le capteur et nous envoyer le signal correspondant sur la carte Arduino.
Au début, nous pensions utiliser un micro pour détecter les coups de l'utilisateur mais, nous nous sommes rapidement rendu compte qu'il y aurait trop de sons parasites et que nous aurions du mal à détecter précisément ces coups.
Nous avons effectué plusieurs tests afin de définir la sensibilité du capteur pour qu'il capte les coups frappés mais ne prenne pas en compte les moindres petits effleurements.



Capteur piézoélectrique

2.2. Le servomoteur

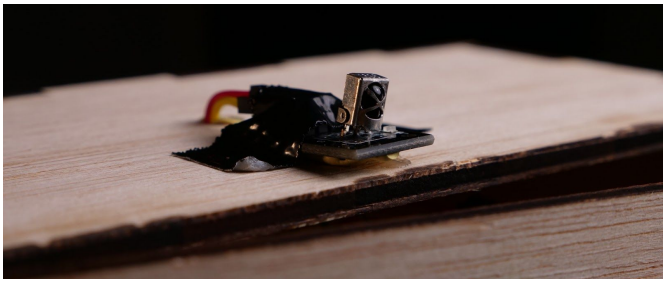
Pour ouvrir le verrou de manière automatique nous avons choisi un servomoteur. Le servomoteur doit absolument être branché sur une sortie PWM car c'est le rapport cyclique du signal envoyé qui va le placer dans une certaine position angulaire. Nous avons relié le bras du moteur au verrou à l'aide d'un trombone déformé. Bien que ce ne soit pas la manière la plus élégante, le trombone fonctionnait parfaitement. Pour trouver les positions "ouvert/fermé" nous avons testé plusieurs angles jusqu'à trouver le bon.



Servomoteur

2.3. La télécommande/récepteur IR

Nous avons décidé d'utiliser une télécommande avec un récepteur infrarouge parce que c'était le plus approprié. En effet, l'utilisation de la télécommande est secondaire dans notre projet, et de ce fait, on trouvait moins pratique de concevoir deux applications (iOS/Androïd) selon le téléphone de l'utilisateur et qu'il doive par la suite la télécharger. Ainsi, avec la télécommande, l'utilisateur à tout en mains pour se servir directement de Knocki.
Chaque bouton de la télécommande est associé à un code, qu'on peut afficher en hexadécimal sur le moniteur série. Nous avons ainsi codé une action à chaque bouton utilisé.

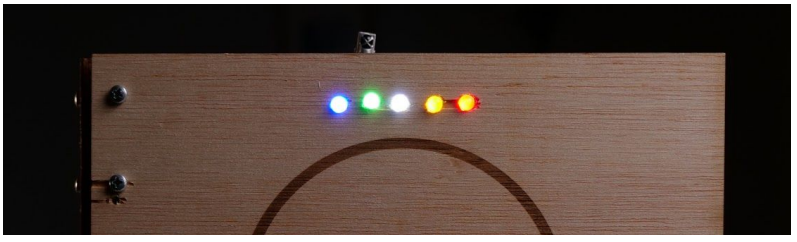


Récepteur infrarouge

2.4. LEDs

Il y a 5 LEDs sur la face avant de la porte, elles servent à communiquer avec l'utilisateur.

- La LED bleu indique que Knocki est en train d'écouter la combinaison.
- La LED verte indique que la combinaison est juste ou que la porte s'ouvre.
- La LED blanche clignote quand un coup est frappé pour indiquer qu'il a été pris en compte.
- La LED jaune indique que Knocki va sauvegarder la combinaison qu'il est en train d'écouter.
- La LED rouge indique que la combinaison est fausse ou que la porte se ferme.



Barre de LEDs sur la face de la porte

CHAPITRE 3 : Le programme

3.1. Architecture

Nous avons décidé de segmenter le code en plusieurs parties pour qu'il soit plus lisible, plus simple à maintenir et pour pouvoir ajouter simplement de nouvelles fonctionnalités.
Notre programme est composé de quatre fichiers :

- un fichier qui comporte tout le code permettant de gérer la combinaison
- un deuxième qui gère le comportement du servomoteur
- un troisième qui code toutes les actions des LEDs
- et le corps du programme qui relie tous les fichiers entre eux et qui contient l'algorithme principal.

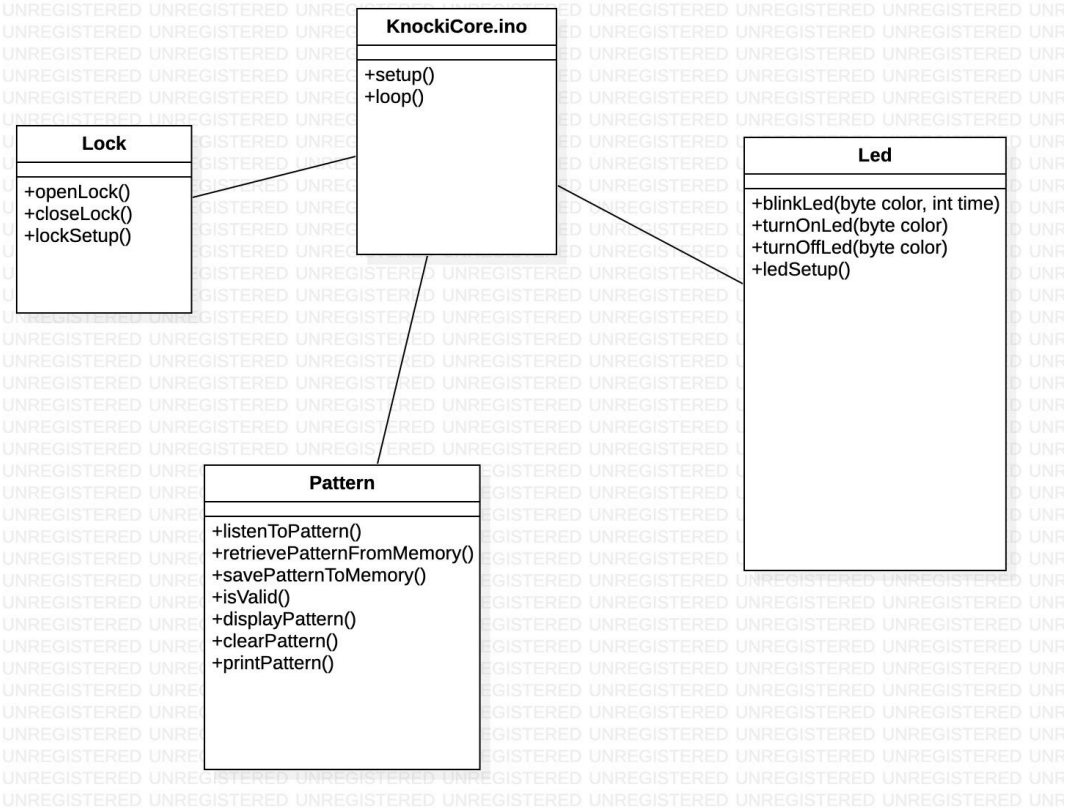


Diagramme UML du projet

3.2. Algorithme principal

Les deux parties les plus importantes de notre algorithme principal sont la détection du bouton de la télécommande utilisé et la détection d'une combinaison.
Lorsqu'on appuie sur un bouton de la télécommande, cette dernière envoie un code hexadécimal au récepteur qui le transmet ensuite à l'Arduino. Nous programmons simplement des cas pour chaque bouton que nous prenons en compte à l'aide de la clause "**switch (...) case (...):**" du langage Arduino. Le bouton 1 ouvre le verrou, le bouton 2 ferme le verrou, le bouton 3 lance le mode "**recording**" (un simple booléen) qui indique à Knocki que l'on souhaite enregistrer une nouvelle combinaison et finalement le bouton "play/pause" fait clignoter la LED blanche au même rythme que la combinaison de référence. Cette vérification sur le code reçu par la télécommande IR a lieu en temps réel et en boucle dans la fonction "**loop()**". Juste après cette vérification se trouve la fonction "**listenToPattern()**" dont le rôle est d'écouter une combinaison. Elle se déclenche dès qu'un coup est détecté. Son fonctionnement est détaillé dans la section suivante.

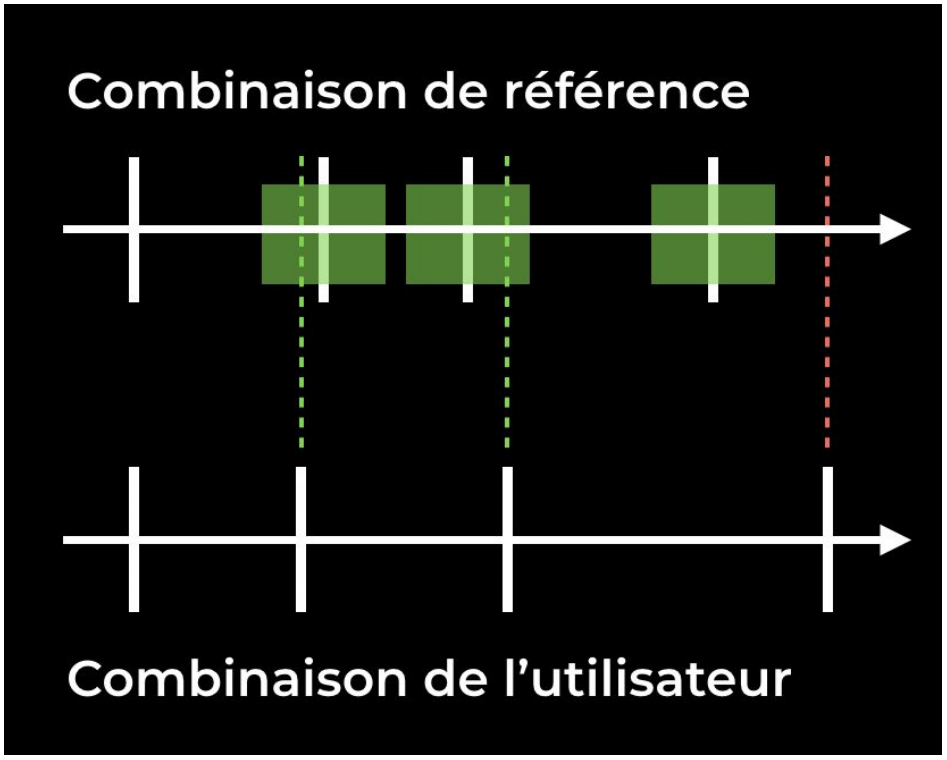
3.3. Algorithme de détection

Quand un coup est frappé, donc une déformation est détecté, alors on commence à enregistrer la combinaison. En effet, on commence à enregistrer la combinaison dès qu'on capte le premier coup en stockant le temps du coup dans une variable temporaire. A chaque coup on met à jour la variable contenant le temps du coup précédent. On va ainsi stocker dans un tableau, l'intervalle de temps entre chaque coup ce qui forme la combinaison. Ce code est imbriqué dans une boucle "**do... while...**" ce qui assure qu'il est exécuté au moins un fois. La condition du "**while**" permet de limiter le temps d'écoute. Elle compare l'écart de temps entre le dernier coup et le temps actuel obtenu avec "**millis()**", si cet écart dépasse 2 secondes alors on sort de la boucle et on arrête d'écouter. Maintenant que la combinaison est stockée, il va falloir la traiter.

3.4. Traitement de la combinaison

Si l'utilisateur a appuyé sur le bouton 3 et qu'il a activé le mode "**recording**", alors Knocki va sauvegarder le tableau contenant la combinaison dans la mémoire EEPROM de l'Arduino pour la faire persister après un éventuel redémarrage de la carte. La combinaison étant un tableau d'entiers, on stocke chaque entier sur 2 "cases" de la mémoire car les entiers sont codés sur 2 octets. Le principe est le même pour récupérer la combinaison de la mémoire lors du démarrage de la carte, on lit la mémoire toutes les 2 "cases" et on récupère l'entier associé aux cases.
Si nous ne sommes pas dans le mode "**recording**" alors on regarde si la combinaison est valide. Etant donné que l'utilisateur ne pourra surement pas etre capable de retaper à la milliseconde près sa combinaison, il a fallu créer une fonction "**patternsValid()**" qui valide ou non la combinaison qui vient d'être frappée en prenant en compte une marge d'erreur. On va alors la comparer à la combinaison de référence en vérifiant que chaque valeur de la combinaison qui vient d'être frappée correspond bien à celle de la combinaison de référence, plus ou moins la marge d'erreur qu'on a définie. Ainsi, cette fonction va renvoyer true ou false selon la validité de la combinaison frappée.

On regarde chaque écart de temps de la combinaison frappée. Soit E = {x1, … , xn} le tableau de la combinaison de référence. Pour qu'un écart de la combinaison frappée soit valide il doit être compris dans l'intervalle [xk - t ; xk + t] où t est la marge d'erreur. Si la combinaison est juste alors on commande l'ouverture de la porte sinon on regarde le nombre de coups frappés. Si 2 coups ont été frappés et que l'écart entre ces 2 coups est de 300 millisecondes, alors on sait que l'utilisateur souhaite refermer la porte donc on la referme.
Finalement si aucune de ces conditions est validée, cela signifie que la combinaison entrée par l'utilisateur est fausse, donc on fait clignoter la LED rouge pour le lui faire comprendre.



CHAPITRE 4 : Déroulement du projet

4.1. Travail du binôme

Nous nous sommes répartis les tâches de manière à pouvoir travailler en même temps sur deux tâches distinctes. Cependant, certaines tâches comme le programme, étaient bien trop compliquées pour être traité par une seule personne. De ce fait, nous avons travaillé en binôme sur certains aspect du projet, ce qui nous a permis d'avancer plus vite étant donné que nous étions complémentaires.

	Rigomier Lise	Djian-Martin Nathan
Système servomoteur verrou	●	●
Télécommande/Récepteur IR	●	●
Capteur piézoélectrique	●	●
LEDs	●	●

Design maquette	●	•
Assemblage maquette	•	●
Mise en place des composants	●	●
Programme principal	●	●

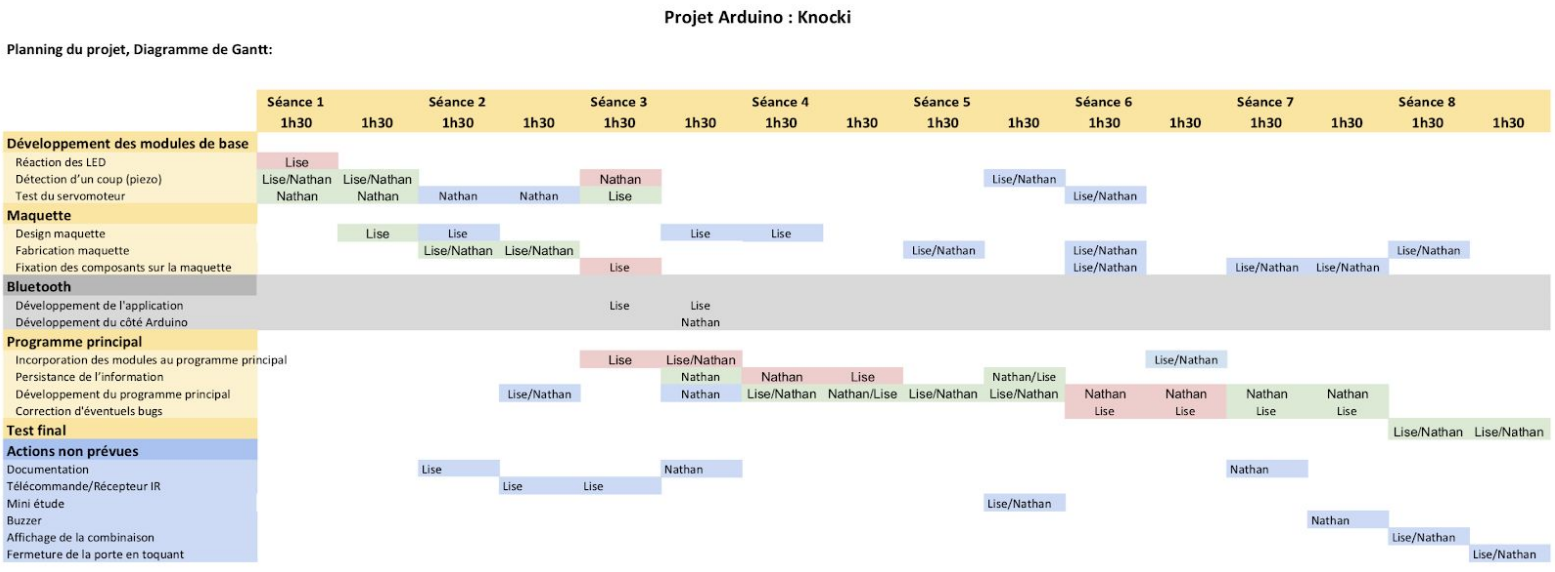
Légende

● : fait

• : a apporté de l'aide

A l'avenir, nous pourrons encore plus nous répartir les tâches, mais comme c'était notre premier projet il y a certaines tâches que nous voulions tout de même faire ensembles pour attiser notre curiosité.

4.2. Planning prévisionnel et réel



Lors de la réalisation du projet nous nous sommes rendus compte que notre diagramme de Gantt manquait de réalisme et de détails. Les cases rouges représentent les actions qui étaient prévues mais qui n'ont pas été réalisées à ce moment là. Les cases vertes représentent les actions qui ont été réalisées au moment initialement prévu. Finalement les cases bleus représentent les actions

que nous n'avions pas prévues lors de l'état de l'art ou auxquelles nous n'avions pas dédié assez de séances. Ces cases bleues modélisent aussi des fonctionnalités que nous avons jugées intéressantes et que nous avons donc rajoutées en cours de route. La grande partie grise correspond au Bluetooth que nous avons rapidement décidé d'abandonner pour passer sur de l'infrarouge.
Nous avons sûr-estimé et sous-estimé certaines tâche ce qui a créé des disparités dans notre emploi du temps. A l'avenir nous pensons être capables de réaliser un diagramme de Gantt tout de même plus réaliste et plus détaillé.

CHAPITRE 5 : Synthèse du projet

5.1. Les améliorations possibles

En ayant plus de temps plusieurs options peuvent être ajoutées pour améliorer notre projet.
Tout d'abord, nous avons pensé à ajouter un système multi-utilisateurs, ce qui impliquerait d'avoir plusieurs combinaisons possibles pour déverrouiller la porte.
Nous pensions ajouter un écran LCD pour communiquer avec l'utilisateur.
Nous avions également pensé au fait que ce soit l'utilisateur qui configure lui-même la sensibilité du capteur piézoélectrique et la précision du rythme en modifiant la marge d'erreur. L'écran LCD aurait alors été nécessaire pour que l'utilisateur modifie visuellement les différentes options.
De plus, si nous voulons par la suite, commercialiser notre projet, il serait bien de faire rentrer tout notre projet dans une petit boite qui se fixerait directement à une porte. Ainsi, le client n'aurait plus qu'à installer cette petite boîte d'un côté de sa porte et le verrou de l'autre.

5.2. Finalité du projet

Bien que l'application aux escape games soit le but principal de notre projet nous avions tout de même réfléchi au fait de l'intégrer dans des maisons. Le problème majeur de l'utilisation à domicile est la sécurité. N'importe qui peut vous entendre taper votre combinaison secrète lorsque vous rentrez chez vous.
Nous sommes entrés en contact avec le directeur de Planet Expériences, un centre d'escape games basé à Antibes. Il nous a dit que notre projet pourrait avoir sa place dans une salle. La technologie est cohérente avec ce qui se fait dans le milieu, ils utilisent en effet eux aussi des cartes Arduino. Le directeur nous a même conseillé d'utiliser un électro-aimant solenoid à la place du verrou et du servomoteur car c'est le mécanisme qu'ils utilisent dans leurs salles pour verrouiller les portes.

CONCLUSION :

Pour conclure, lors de ce projet nous avons tout d'abord gagné de la confiance en nous. En effet au début le projet nous paraissait irréalisable. Le fait de parvenir à obtenir un dispositif fonctionnel pour notre soutenance finale, nous a montré que nous étions capable de mener à bien ce projet.
Ce projet nous a aussi permis de développer notre capacité à travailler en équipe. En effet, nous avons dû apprendre à déléguer certaines tâches à notre partenaire, parfois se reposer sur lui, ce qui impliquait de lui faire confiance.
Nous avons aussi acquis de l'expérience. Nous avons prit part à notre première expérience dans un projet et nous l'avons réalisé de A à Z.
Mais, nous avons également appris à travailler davantage en autonomie car nous devions nous documenter et régler nos problèmes au maximum seul pour ne pas empêcher notre partenaire de travailler sur sa partie.
De plus, nous nous sommes rendu compte de la complexité et de l'importance de l'organisation d'un projet. Nous n'étions pourtant que deux mais, ce projet nous a fait prendre conscience de la nécessité d'être organisé et de communiquer avec ses partenaires dans des projets à plus grande envergure.

REMERCIEMENTS :

Nous tenons à remercier Monsieur Masson de nous avoir permi de réaliser ce projet en Arduino et de nous avoir accordé son temps et son aide durant plusieurs mois. Nous voulons également remercier Monsieur Abderrahmane d'avoir pris le temps de nous débloquer sur certains points. Enfin, nous remercions monsieur Forner (gérant du Fablab) qui nous a aidé pour la mise en forme de notre maquette.

Lien démonstration de Knocki : https://www.youtube.com/watch?v=I42KXvuVLV4

Resources :

- Utilisation capteur piezo : https://www.arduino.cc/en/Tutorial/Knock
- Exemple d'un projet similaire : https://www.instructables.com/id/Secret-Knock-Detecting-Door-Lock/
- Exemple d'un projet similaire : https://circuitdigest.com/microcontroller-projects/secret-knock-pattern-detecting-door-lock-arduino