UCLouvain

Université Catholique de Louvain

Projet N°3
MagicLock

Auteurs :
Béranger Deketelaere
Romain Gobert
Bryan Devos
Julien Devos

Groupe :
N° 04
Référent :
Gaël Aglin

Résumé

Pour la réalisation de ce projet, nous avons du concevoir un cahier des charges ainsi qu'un planning pour organiser le déroulement de celui-ci. Ensuite, nous avons été introduit au concept de la cryptographie afin de sécuriser le message et le code. Nous avons également appris à utiliser le gestionnaire de version git sur un dépôt distant comme GitHub ce qui nous a permis de travailler en équipe sur le même code. Pour terminer, nous avons été amenés à présenter notre projet devant la classe à l'aide de différentes slides ainsi qu'a faire une démonstration de notre MagicLock en action, enfin il se clôture avec ce rapport.

18 décembre 2020

Table des matières

1 Introduction

Dans ce 3ème projet, notre mission principale a été de créer un MagicLock. Celui-ci permettra à un agent secret de transporter des informations confidentielles de manière simple et sécurisée. Pour ce faire, il faut que l'utilisateur puisse enregistrer un message dans le MagicLock et ensuite le sécuriser avec un code (suite de mouvements). Une fois le message et le code entrés ils doivent tous-deux être enregistré dans un fichier afin de pouvoir être récupérés même après avoir éteint le MagicLock. Par sécurité le message doit être chiffré à l'aide du code et le code, haché. Pour finir, l'utilisateur doit être capable de déchiffrer le message en entrant le code utilisé pour chiffrer celui-ci et ainsi, pouvoir visionner le message. Ce rapport va s'articuler en plusieurs points importants. Dans un premier temps, la partie "Présentation du programme". Nous y expliquerons ce que font les différentes fonctions que comporte notre programme et le déroulement de celui-ci. Nous nous pencherons ensuite sur le prochain point important de notre travail les "Aspects originaux". On y expliquera les différentes fonctionnalités que nous avons rajoutés ainsi que les multiples aspects sur lesquels nous avons voulu mettre l'accent.

2 Présentation du programme

2.1 Glossaire des fonctions

- *'main()'*
  La fonction principale qui permet d'appeler les autres fonctions au moment voulu.
- *'pin()'*
  Reconnaît quand la bonne suite de mouvement du Joystick est effectuée, par défaut le code pin est *"Haut", "Bas", "Gauche", "Droite"*.
- *'check_msg()'*
  Vérifie si un message est enregistré dans le fichier *"message.txt"*.
- *'show_menu(menu)'*
  Met à jour l'index du choix en fonction des mouvements du Joystick et appelle la fonction *display_ choice()* pour afficher le choix correspondant à l'index se trouvant dans la liste *menu* qui se compose de tuples sous cette forme *("description", liste de pixel)*.
- *'display_choice(menu)'*
  Affiche à l'écran le logo en position de l'index choisi dans *show_ menu()* en fonction du *menu*.
- *'choosed_option(menu)'*
  Fait une action précise en fonction de l'option choisie dans le menu. L'action est différenciée grâce la description présente dans la liste de tuples *menu*.
- *'save_message()'*
  Affiche le menu qui va permettre à l'utilisateur d'enregistrer un nouveau message dans le MagicLock.
- *'view(message)'*
  Affiche le message *message* sur l'écran du MagicLock.
- *'save_floppy_disk_icon()'*
  Affiche à l'utilisateur un logo de floppy disk pour indiquer que quelque chose s'enregistre. Voir Figure 1
- *'save_code()'*
  Enregistre le code (suite de mouvements) en fonction des différentes faces fournies par l'utilisateur dans une liste.
- *'code_list_to_str(code_list)'*
  Transforme ce que *save_ code()* retourne en un string correspondant à chaque élément. Voir 3.4
- *'encrypt_all(message_number_str, code_str)'*
  Chiffre le message *message_ number_ str* à l'aide du code *code_ str* enregistré. Pour ce faire le message est transformé en lettres à l'aide d'une liste de lettres dans un ordre différent pour plus de sécurité.
- *'save_encrypted_data(encrypted_data)'*
  Sauvegarde le message et le code dans leurs fichiers respectif *"message.txt", "code.txt"*.
- *'code_help()'*
  Affiche des messages d'aide pour enregistrer le code à l'écran.
- *'decode_all(code_str_tried)'*
  Si le code fourni est correct, décode le message enregistré.
- *'delete_all()'*
  Supprime le message et le code précédemment contenu dans leurs fichiers.

- *'wrong_code_display()'*
  Affiche à l'écran des logos indiquant que le code et les messages seront supprimés.

2.2 Déroulement du programme

Notre programme démarre quand l'utilisateur met en mouvement le joystick, deux possibilités sont alors possibles : soit l'utilisateur effectue la bonne combinaison et dans ce cas, le programme passe à la prochaine étape et l'utilisateur en est averti grâce à un indicateur à l'écran ; ou soit la combinaison est mauvaise et dans ce cas il est forcé à réessayer. Le programme vérifie alors si un message est déjà contenu dans le MagicLock.

Si aucun message n'est présent, l'utilisateur est alors face à un nouveau menu où il a le choix de soit revenir en arrière et de tout annuler ou bien d'enregistrer un (nouveau) message. Dans le cas où il souhaite en enregistrer un, il est alors face à un nouveau menu composé de logos : un pour chaque chiffre, un pour supprimer le dernier caractère entré, un pour sauvegarder le message ainsi qu'un pour tout annuler. Pour enregistrer son message il doit alors sélectionner les logos des chiffres composants son message et valider en appuyant sur le joystick. Lorsqu'il a validé son message, il doit alors enregistrer un code pour sécuriser son message. Pour ce faire, il devra choisir une série de positions et entre chacune d'entre elles, valider la position par une pression du joystick. Lorsque l'utilisateur est satisfait du code, il doit l'enregistrer par un mouvement du joystick vers la gauche. A noter qu'il peut afficher un texte d'aide avec un mouvement du joystick vers la droite et qu'il peut tout quitter avec un mouvement vers le haut. Lorsque le message et le code sont validés, ils sont tout deux chiffrés et le MagicLock se verrouille.

Si un message est bien enregistré, l'utilisateur est maintenant face à 2 options : soit déchiffrer le message en reproduisant le bon code ou simplement annuler. S'il décide donc de déchiffrer le message et qu'il reproduit le bon code, il pourra alors choisir entre voir le message, le supprimer ou annuler pour quitter le programme. Par contre, s'il se trompe pour reproduire le code, il pourra seulement réessayer 2 fois après quoi, si le code est toujours incorrect, le message et le code seront supprimés et impossible à récupérer.

3 Aspects originaux

3.1 Les logos

Nous avons réfléchi à comment nous pouvions afficher les différentes informations à l'utilisateur. La solution la plus pratique s'est avérée être l'utilisation des logos ou pictogrammes suivants.

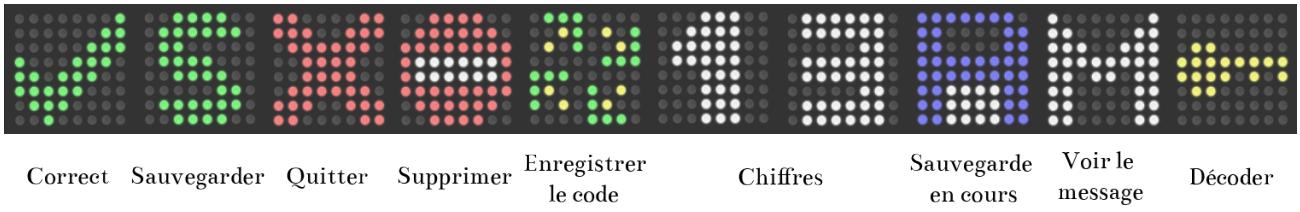

FIGURE 1 – Légende des différents logos ou pictogrammes

3.2 Le code pin

Nous avons voulu rajouter un code pin, se composant de mouvement du Joystick, dans le souci de sécuriser encore plus notre MagicLock. Ainsi, l'écran de base de celui-ci n'affiche rien et le MagicLock paraît éteint. Pour "l'allumer" l'utilisateur devra donc entrer le bon code pin. Notons également que l'utilisateur possède un nombre de tentatives illimité pour réaliser celui-ci. Le code pin est par défaut *"Haut", "Bas", "Gauche", "Droite".*

3.3 Les différents menus

Pour faciliter le choix des différentes options, nous avons décidé d'intégrer une sorte de menu qui reprend les différentes options que peut choisir l'utilisateur. En fonction d'où il se trouve dans le déroulement du programme et en fonction bien sûr de si le MagicLock contient un message ou non. Et pour rendre la navigation dans les menus plus facile, nous avons opté pour un système de navigation en boucle, l'utilisateur peut donc naviguer dans le menu en n'allant que par la gauche ou que par la droite. Arrivé au bout du menu, le prochain mouvement vers la gauche ou la droite va faire revenir l'utilisateur au début de celui-ci.

3.4 Le fonctionnement du code (suite de mouvements)

Nous avions tout d'abord pensé à utiliser le gyroscope du Sense Hat mais au vu de la difficulté à obtenir les mêmes valeurs en refaisant le code, même en ayant laissé une marge d'erreur importante. Nous avons donc opté pour l'utilisation de l'accéléromètre. Celui-ci nous permet d'avoir une composition de x y et z différente pour chaque face du Sense Hat. La fonction *code_list_to_str(code_list)* transforme donc un liste de type : `[['x',valeur],['y',valeur],..]` en string de type : `"FACE1FACE2..."`

```
def code_list_to_str(code_list):
    code_str = ""
    for pos in code_list:
        x, y, z = pos[0][1], pos[1][1], pos[2][1]

        if x == 1 and y == 0 and z == 0:
            code_str += "FACE1"
        elif x == 0 and y == 1 and z == 0:
            code_str += "FACE2"
        elif x== 0 and y== 0 and z== 1 or str(x)== 'NaN' or str(y)== 'NaN' or str(z)== 'NaN':
            code_str += "FACE3"
        elif x == -1 and y == 0 and z == 0:
            code_str += "FACE4"
        elif x == 0 and y == -1 and z == 0:
            code_str += "FACE5"
        elif x == 0 and y == 0 and z == -1:
            code_str += "FACE6"

    return code_str
```

Listing 1 – La fonction `code_list_to_str(code_list)`

3.5 La sécurité en plus du code

Au moment de faire la suite de mouvement pour déchiffrer le message, s'il y en a un, nous avons pensé à une sécurité supplémentaire qui pourrait s'avérer utile en cas de perte ou de vol du MagicLock. Tandis que le code Pin a un nombre illimité d'essai, nous avons décider de ne mettre que trois tentatives lorsqu'on entre la suite de mouvement pour déchiffrer le message. Si les trois essais viennent à être épuisés, le message ainsi que son code seront aussitôt effacés.

4 Analyse dynamique du groupe

Pour ce projet 3 nous avions la possibilité de choisir chacun des membres qui allaient composer notre groupe. Comme vous vous en doutez sûrement, nous sommes partis sur base d'un trio d'amis qui était déjà existant au cours de nos années de secondaires. Suite à ça, Romain est venu nous demander s'il pouvait se joindre à nous. L'entente entre nous trois était déjà bien ancrée et on savait déjà comment on voulait fonctionner. Romain s'est très facilement intégré et on s'est vite aperçu qu'il résonnait comme nous, autant dire que c'était déjà un excellent point ! Grâce à ça nous n'avons jamais eu de désaccord quel qu'il soit. Les séances de travail étaient donc constructives et rapides sans pour autant paraître trop longues ou trop ennuyantes. Nous nous mettions d'accord sur les différents problèmes à résoudre, nous veillions bien à ce que tous aient bien compris la matière qui venait d'être travaillée, etc. Nous finirons par dire que c'était un groupe organisé et qui savait rester dans une bonne ambiance !

5 Conclusion

Pour conclure, l'objectif principal de notre groupe était dans un premier temps de savoir répondre à chacune des spécificités qui nous étaient demandées pour la création de ce MagicLock. Une fois cela fait, nous avions même réfléchis à différents critères qui pourraient encore l'améliorer comme par exemple l'ajout d'un code Pin pour démarrer l'appareil, le choix d'une interface sobre, claire et colorée ou encore la sécurité supplémentaire qui vient à supprimer le message ainsi que son code si l'utilisateur a épuisé ses trois tentatives ! Le code a également été pensé pour permettre une certaine facilité en cas de modification lors d'une éventuelle mise à jour en commentant l'intégralité des fonctions disponibles dans le programme. Nous pensons donc avoir correctement implémenté les demandes mais également innové en trouvant de nouvelles fonctionnalités.