

SUJET D’EVALUATION :
Pour le Poste DEVELOPPEUR SYSTEME ANDROID

21/10/2019

Rapport

Intruder-LockTrap

Piège la personne qui tente de déverrouiller le
Smartphone en prenant une photo



Stephane JEZIORNY
PREPARE POUR SUNERIS – GROUPE THALES
CONTACT : FONTAINE Boris

Rédaction : 13/10/19 – 21/10/19
Version 3.0 – 21/10/2019

VERSION :

Version	Rédacteur	Date	Mise à Jour
1.0	Stéphane JEZIORNY	13/10/2019	Début de rédaction : Présentation, Utilisation et Organigramme
1.1	Stéphane JEZIORNY	14/10/2019	Ajouts choix techniques et amélioration possibles
1.2	Stéphane JEZIORNY	14/10 – 15/10	Relecture
2.0	Stéphane JEZIORNY	16/10	MAJ Interface et état actuel
2.1	Stéphane JEZIORNY	17/10	MAJ Choix Techniques : - Tentatives RadioBouton - Galerie Améliorations possibles : - Feedback - Autres Améliorations
3.0	Stéphane JEZIORNY	21/10	MAJ Interface Choix techniques Autres Amélioration
->	Stéphane JEZIORNY	21/10	Création d'une Doc Utilisateur

Table des matières

1. PRESENTATION DU PROGRAMME	1-2
Présentation	1
Spécifications.....	1
Interface	1
Etat Actuel	2
2. UTILISATION DE L'APPLICATION	2-3
3. ORGANIGRAMME DE FONCTIONNEMENT.....	4
4. LE CODE	5-6
5. JUSTIFICATION DES CHOIX TECHNIQUES	7-9
Android Studio.....	7
Appli Administration Appareil	7
Camera API – SurfaceView	8
Stockage Interne / Stockage Externe	8
Nombre de tentatives - RadioBouton	9
Galerie	9
Fond de Tache	10
Notification.....	10
Redémarrage	11
6. AMELIORATIONS POSSIBLES ET LIMITES	11-14
Améliorations	11
Limites.....	12
Autres améliorations	13
Application en Fond de Tâche	14
Evolution.....	14
7. BIBLIOGRAPHIE.....	15

1. Présentation du programme

Présentation :

Intruder-LockTrap est une application développée pour une évaluation dans le cadre d'une embauche au poste développeur système Android au sein du groupe Thales.

Ce programme réalise un verrouillage de l'écran et permet de prendre une photo de toutes personnes qui se trompent dans le déverrouillage du smartphone. La prise de la photo est réalisée par la caméra frontale du téléphone au moment où l'utilisateur entre un code erroné.

L'objectif du programme est de permettre au propriétaire du smartphone de savoir si quelqu'un a tenté de déverrouiller son smartphone et d'obtenir la photo de cette personne.

Spécification :

Nom de l'application : Intruder-LockTrap :

Le programme a été codé en Java (Android) au travers d'Android Studio.

Version d'Android minimum : Android 5.0

Interface :

Aperçu de l'interface de l'application actuelle :

L'écran principal permet de :

- **Verrouiller le téléphone par mot de passe**
- **Activer Appli administration appareil nécessaire au verrouillage de l'application**
- **Désactiver Appli administration appareil**
- **Autoriser l'utilisation de la caméra**
- **Choisir le nombre de tentatives avant prise de la photo.**
- **Activer une notification par laquelle on peut verrouiller le téléphone**
- **Activer la possibilité d'utilisation en fond de tâche**
- **D'ouvrir la galerie.**
- **Visualiser la dernière photo prise.**
- **Récupérer la date et l'heure à laquelle la photo a été prise.**



Etat Actuel :

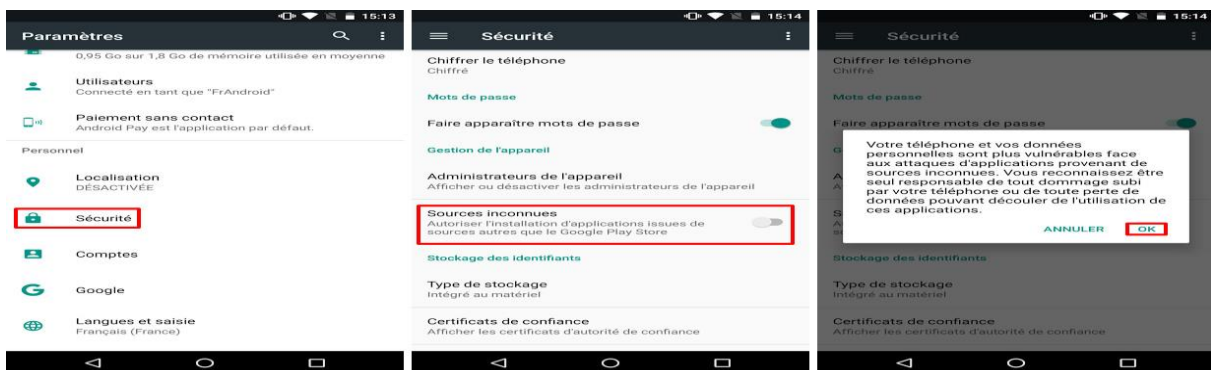
La fonctionnalité principale de l'application est opérationnelle.
L'application permet de verrouiller le smartphone et de prendre une photo (secrètement) en cas d'erreur sur le mot de passe.
La photo est enregistrée actuellement en interne (dans l'application) :
/data/user/0/com.example.stephane.locktrap/app_imageDir/test.jpg
L'utilisateur peut choisir le nombre d'erreurs avant la prise de la photo.
Elle permet également un aperçu de la photo en plus grand dans la galerie.
La notification ne marche que sur une version inférieure à Android 6.0. Il n'est pas possible de le faire fonctionner en fond de tâche « service illimité » (Activité principale détruite) par un appui sur le bouton power sans toucher à la surcoudre constructeur.

2. Utilisation de l'Application

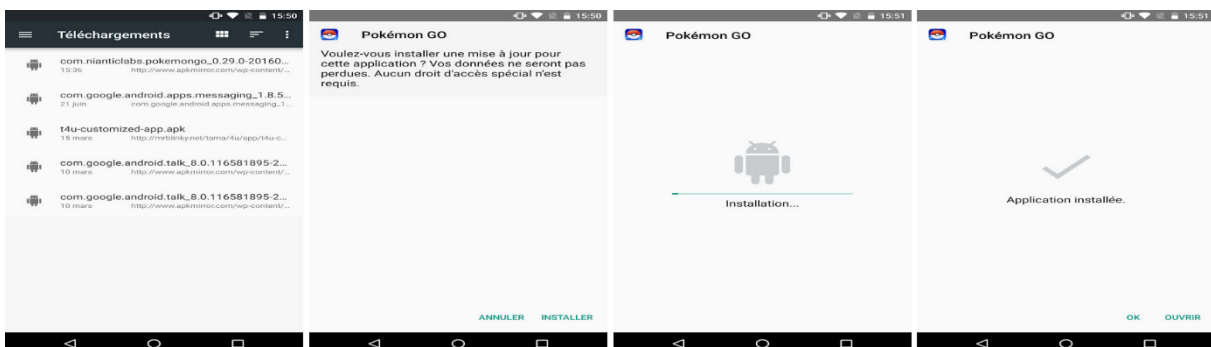
Installer un fichier APK :

Le fichier APK permet une installation du programme sans passer par la Play Store de Google. Cependant cela nécessite d'avoir confiance à l'application et au développeur, car elle n'est soumise à aucuns contrôles sur l'installation.

1) Autoriser les fichiers de source inconnues.



2) Télécharger l'APK sur Smartphone et l'exécuter.



Autorisation :

Pour le bon fonctionnement de l'application il faut autoriser l'application à utiliser la caméra et pouvoir enregistrer des fichiers. De même il faut faire attention à vérifier que les autorisations ne sont pas utilisées par le développeur pour soustraire des données ou autres informations.

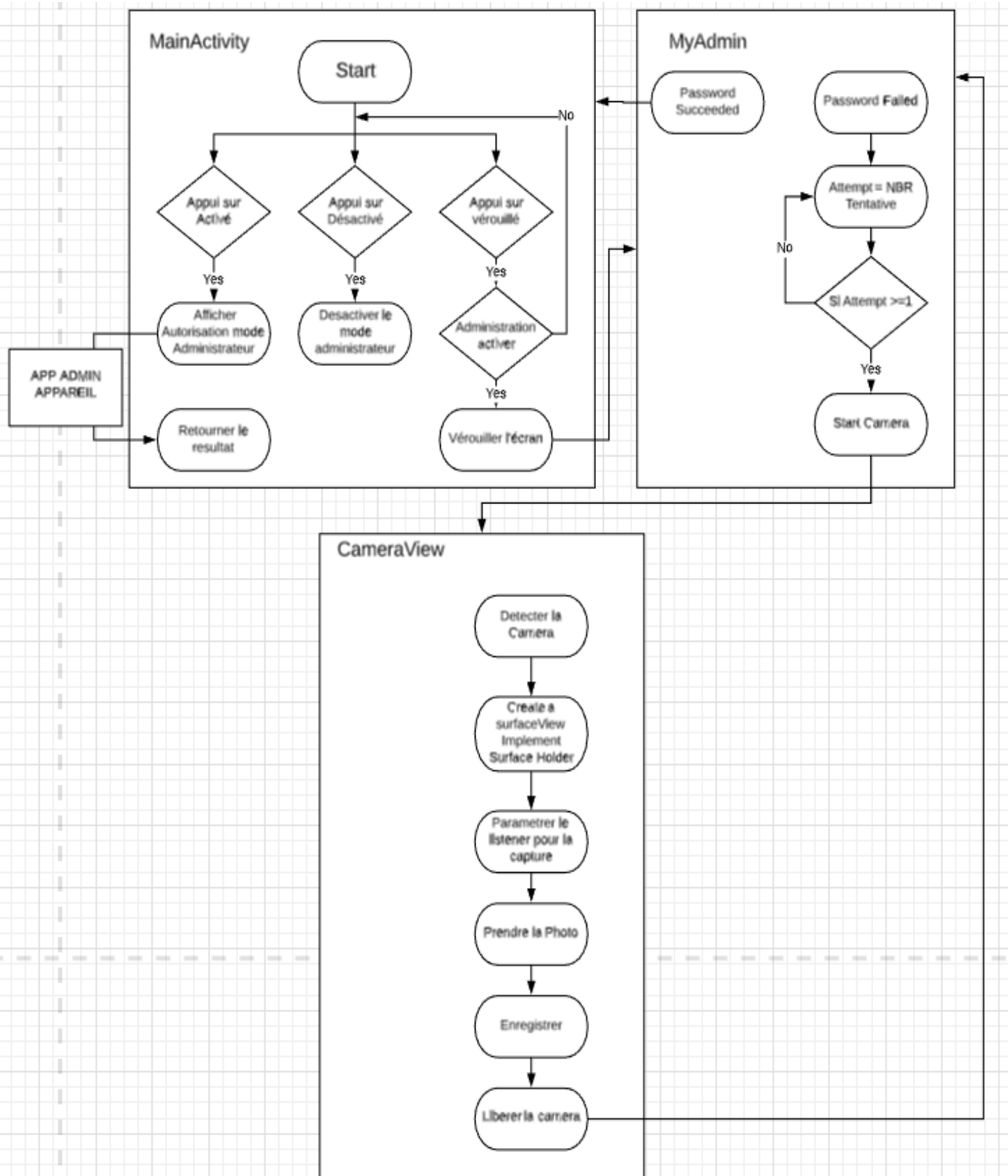


Appli Administration Appareil :

Pour pouvoir verrouiller le smartphone et détecter un mot de passe erroné l'application utilise Appli Administration Appareil. Pour l'activer il faut passer directement par l'application et cliquer sur activer. L'application administration appareil permet de lancer de nombreuses fonctions (remise état usine, cryptage des données...). A activer avec précaution et une totale confiance en l'application.



3. Organigramme de fonctionnement



4. Le code (A Venir : Description des différentes classes, méthodes utilisées)

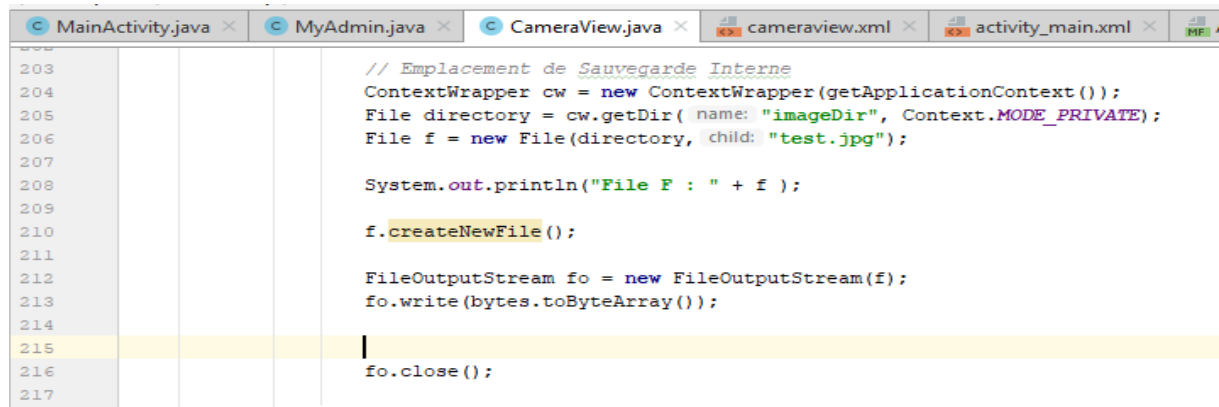
MainActivity : (Gestion Boutons)

```
MainActivity.java x MyAdmin.java x CameraView.java x cameraview.xml x activity_main.xml x AndroidManifest.xml x
62 @Override
63 public void onClick(View v) {
64
65     if(v == lock){
66         boolean active = deviceManger.isAdminActive(compName); // Verifier que l'administration est active
67         if (active) {
68             deviceManger.lockNow();
69         }
70     }
71
72     if(v == enable){
73         Intent intent = new Intent(DevicePolicyManager
74             .ACTION_ADD_DEVICE_ADMIN);
75         intent.putExtra(DevicePolicyManager.EXTRA_DEVICE_ADMIN,
76             compName);
77         intent.putExtra(DevicePolicyManager.EXTRA_ADD_EXPLANATION,
78             value: "L'autorisation permet de verrouiller l'application avec motde passe obligatoire");
79         startActivityForResult(intent, RESULT_ENABLE);
80     }
81
82     if(v == disable){
83         deviceManger.removeActiveAdmin(compName);
84         updateButtonStates();
85     }
86 }
```

MyAdmin : (Gestion Mauvais MDP)

```
MainActivity.java x MyAdmin.java x CameraView.java x cameraview.xml x activity_main.xml x AndroidManifest.xml x
47
48 @
49 public void onPasswordFailed(Context context, Intent intent) {
50     DevicePolicyManager policyManager = (DevicePolicyManager)context.getSystemService(Context.DEVICE_POLICY_
51     if(policyManager != null){
52         int attempts = policyManager.getCurrentFailedPasswordAttempts();
53         Log.v( tag: "Stephane", msg: "Attempts = " + attempts);
54         if (attempts >= 1) {
55
56             Intent i = new Intent(context, CameraView.class);
57             i.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
58             context.startActivity(i);
59         }
60     }
61 }
62
63
64 @Override
65 public void onPasswordSucceeded(Context context, Intent intent) {
66     showToast(context, msg: "Sample Device Admin: pw succeeded");
67 }
68
69 }
```


CameraView : (Sauvegarde Photo)



The screenshot shows an IDE window with several tabs: MainActivity.java, MyAdmin.java, CameraView.java, cameraview.xml, and activity_main.xml. The CameraView.java tab is active, displaying the following Java code:

```
203 // Emplacement de Sauvegarde Interne
204 ContextWrapper cw = new ContextWrapper(getApplicationContext());
205 File directory = cw.getDir( name: "imageDir", Context.MODE_PRIVATE);
206 File f = new File(directory, child: "test.jpg");
207
208 System.out.println("File F : " + f );
209
210 f.createNewFile();
211
212 FileOutputStream fo = new FileOutputStream(f);
213 fo.write(bytes.toByteArray());
214
215
216 fo.close();
217
```

5. Justification des choix techniques

Android Studio (Java) :

Le premier choix technique à réaliser est l'environnement de développement et le langage de programmation utilisé :

J'ai choisi l'environnement en fonction de mes connaissances, contraintes imposées (Java Android) et des fonctionnalités de l'application.

J'ai des connaissances avec l'interface Kivy Designer qui permet de créer des applications multisupport (IOS, Android, Web...) mais il utilise le langage de programmation Python et ne permet pas de paramétrer le verrouillage d'un système Android

J'ai des connaissances avec l'interface LibGDX qui est plus tournée vers la création de jeux 2D utilisant la librairie OpenGL

Mon choix s'est porté sur l'environnement de développement Android Studio qui permet le plus facilement de développer les fonctionnalités demandées (verrouillage de l'écran et prise de photo).

Android Studio permet depuis 2017 de coder des applications en Kotlin et est devenu le langage officiel depuis Mai 2019. Ce langage permet une interopérabilité avec java (Utilisation de toutes les librairies Java, Outils permettant de traduire facilement un code Java en Kotlin).

Pour des questions de connaissances et de contrainte j'ai choisi de développer l'application en Java.

Appli Administration appareil :

Après plusieurs recherches la solution la plus simple dans le développement pour verrouiller le smartphone et vérifier que le mot de passe renseigné est correct est l'utilisation de Appli Administration Appareil.

Camera API – SurfaceView :

Pour prendre une photo avec Android il y a 2 solutions :

Un Intent vers l'app Camera

La solution la plus simple est d'utiliser un Intent permettant d'ouvrir l'application caméra et prendre la photo, mais cette technique est inutilisable dans l'application car elle nécessite la prise d'une photo sans ouvrir l'application caméra.

Le Framework Camera API

Ce Framework possède 2 classes (Camera et Camera2).

La documentation google conseil l'utilisation de Camera2 pour des raisons de fonctionnalités et de performance, néanmoins il n'est utilisable qu'à partir de la version Android 5.0

L'application ne nécessite pas d'avoir des performances élevées ou des fonctionnalités avancées. C'est la raison pour laquelle j'ai choisi d'utiliser Camera car elle est utilisable à partir d'Android 2.3 et permet de toucher une population plus large.

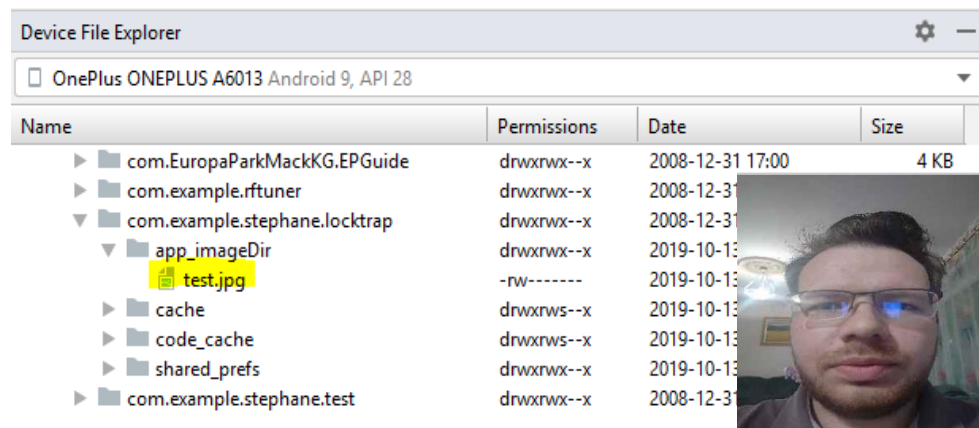
Cette méthode nécessite de créer sa propre View (SurfaceView) qui va afficher une preview de la caméra (de manière cachée) et de gérer le listener pour prendre la photo.

Stockage Interne – Stockage Externe :

Pour enregistrer les photos 2 types de stockage sont possible :

- Le stockage interne ou la photo ne sera disponible qu'à travers l'application
- Le stockage externe ou la photo pourra être visualisée par exemple dans la galerie ou autres espaces de stockage du téléphone.

J'ai choisi d'utiliser un stockage interne pour que la photo ne soit accessible et manipulable que depuis l'application et de pouvoir gérer les permissions du fichier selon les différents utilisateurs.



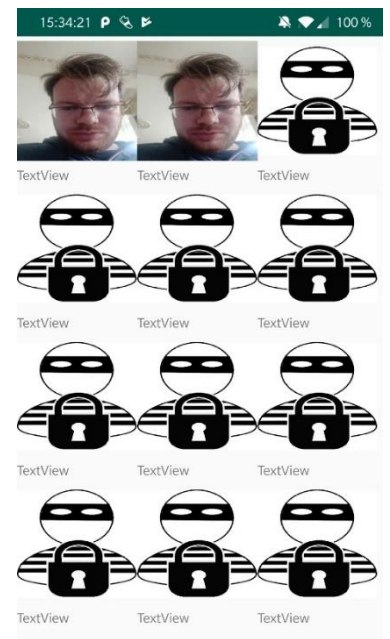
🚦 Nombre de tentatives → RadioButton

Mon choix c'est porté sur des radioButton pour obliger l'utilisateur à faire un choix entre 1-3 de manière plus ergonomique.

🚦 Galerie :

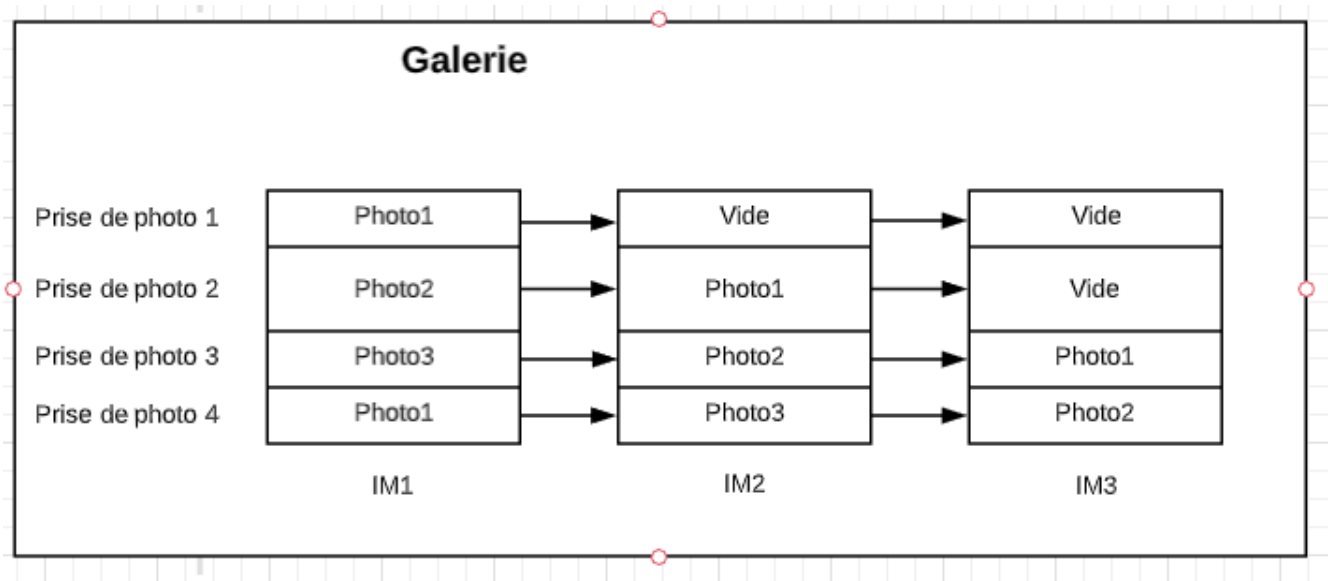
Limiter le nombre de photos a 12 puis les écraser pour rendre l'application moins volumineuse.

Il y a plusieurs librairies permettant de créer une galerie (Picasso), mais comme l'appli ne contient que 12 images (Au maximum), j'ai décidé de créer 12 ImageView avec une variable qui s'incrémente à la prise de chaque photos permettant de faire défiler les photos.



Stratégie de développement :

Exemple avec 3 ImageView



On a une variable qui s'incrémente avec le nombre de photos. A la quatrième photo on réinitialise la variable à 1 qui permet d'écraser la première photo prise.

Cette Stratégie permet de toujours avoir la photo la plus récente au même emplacement.

+ Fond de tache (sans l'interface principale) :

J'ai choisi l'utilisation de service, celui-ci permet d'être exécuté en arrière-plan alors que l'activité principale a été détruite.

➔ Le problème est que dans un service on ne peut pas réaliser une détection de la touche power pour verrouiller le téléphone.

+ Notification pour verrouiller :

J'ai choisi de travailler sur une notification avec un bouton qui permet également de verrouiller l'application alors que l'activité principale a été détruite.

➔ Fonctionnelle sur l'émulateur d'un google pixel, mais mon one plus 6T ne reçoit pas la notification.

Lancer le service après redémarrage :

Utilisation d'un BroadcastReceiver pour détecter le redémarrage du téléphone et démarrer directement le service.

➔ Bonne détection du reboot, mais le service n'est lancé qu'au clic sur l'application.

6. Améliorations possibles et limites

Améliorations Possibles :

De nombreuses améliorations peuvent être apportées à Intruder-LockTrap, permettant d'optimiser la fonctionnalité principale de l'application.

Dans un premier temps une amélioration qui porte sur la sécurité et la confiance que les utilisateurs apportent à l'application :

- Retravailler le code pour que l'appli Administration Appareil n'accorde que l'autorisation de verrouiller l'écran.
➔ OK, Demande de gérer les tentatives de déverrouillage et le verrouillage de l'écran.
L'autorisation ne concerne que l'utilisation de la caméra car la photo est enregistrée en interne (pas besoin d'avoir accès à des zones externes de l'application).
- Travailler sur un stockage interne à l'application des photos afin qu'elles ne puissent être échangées avec d'autres applications qu'à partir de Intruder-LockTrap.
➔ OK – les photos sont uniquement stockées en interne. (Actuellement 1 seule photo qui se réécrit lors de la prise de la photo suivante (économie de place).

Pour continuer à augmenter le nombre de systèmes permettant d'utiliser l'application.

- Retravailler le code pour permettre de lancer l'application sur une version d'Android 2.3 (ANDROID 5.0 – pour l'instant)
 ➔ Je suis passé d'une version mini 6.0 à 5.0 pour l'instant.
 ➔ Travail sur le fonctionnement de l'appli en tâche de fond ➔ Android 5.0

Pour finir optimiser l'ergonomie, la simplicité d'utilisation, l'organisation et l'agencement de l'application.

- Développement d'un bouton Switch dans l'application permettant aux choix de l'utilisateur de verrouiller le smartphone par le bouton physique Power (sans passer par l'application)
 ➔ **Abandonné** : (Nécessite un tel root et de toucher à la surcouche constructeur : kernel)

- Nommer les photos avec la date du jour et l'heure.
 ➔ OK, Donne la date du jour et l'heure au format DD/MM/YYYY_HH.MM.SS

- Créer une galerie contenant les photos directement dans l'application
 ➔ **En Cours**, Actuellement le layout de la galerie est créé. Elle permettra de visualiser les 12 dernières photos.

Problème : Je rencontre actuellement un problème avec l'utilisation de sharedPreferences pour sauvegarder l'incrémentation d'une variable. Sur la première erreur du mot de passe l'application prend une photo, mais pas sur les suivantes avec l'utilisation du sharedPreferences.

Etape actuelle – trouver la raison du problème.

➔ **Abandonné** : Je me suis concentré sur l'utilisation de l'application en tâche de fond (service illimité).

La galerie actuellement affiche la dernière image avec la date/heure de prise.

Limites

Dans le développement de l'application on retrouve 2 types de limites :

Les limites techniques : (choix des différentes solutions utilisées)

- L'application ne peut être lancée sur une version Android inférieure à 2.3 car elle utilise Caméra API pour prendre une photo quand l'écran est verrouillé.
- Obligation de passer par Appli Administration appareil pour accorder des droits supplémentaires à l'application est pouvoir gérer la fonctionnalité de verrouillage de l'écran.

Les limites imposées par le développeur (priorisations des fonctionnalités) : Je privilégie une application qui est optimisée et qui fonctionne, avant de continuer à développer des fonctionnalités supplémentaires.

- Paramètre supplémentaire : Utilisateur choisit le nombre de tentative avant la prise de la photo
➔ OK, Choix d'utilisation de RadioBouton pour définir le nombre de tentative avant prise de la photo.
- Prendre la photo uniquement si détection de visage.
- Envois d'une alertes intrusion avec Image de la personne et Géolocalisation du smartphone par mail.
- ...
➔ Pas encore regardé

Autres Améliorations :

Au cours du test de l'application j'ai également noté plusieurs améliorations possibles et problèmes que j'ai corrigé.

Améliorations :

- ➔ Un lien vers les autorisations pour que l'utilisateur peut autoriser l'utilisation de la caméra plus facilement.
- ➔ Un affichage de la dernière photo sur l'écran principal.
- ➔ Interface de l'application en général

Problèmes corrigés :

- ➔ Fermer l'écran principal après verrouillage et le rouvrir après la prise de la photo. Mise à jour de l'interface avec la dernière photo.
- ➔ Bloqué l'affichage en mode portrait.
- ➔ Bouton retour ne recharge pas une ancienne activité de l'interface principal.
- ➔ Sauvegarde de l'état des RadiosBoutons et le choix du nombre de tentatives à la fermeture de l'application. - SharedPreference
- ➔ Travail sur différents BUG.

Application en tâche de Fond (interface principale détruite)

- ➔ Tâche de fond « service illimité » (Activité principale détruite) par un appui sur le bouton power sans toucher à la surcouche constructeur.

Abandonné : Evolution travailler sur la surcouche constructeur.
travail > 1 semaine

- ➔ Notification : Opérationnel sur un google pixel (émulateur), mon one plus 6T n'affiche pas la notification.

Evolution : Rechercher d'où vient le problème.

- ➔ Détecter un redémarrage et lancer le service.

Evolution : Le service ne démarre que quand l'utilisateur lance l'application, voir si possibilité de le démarrer avant saisie du code pin.

Evolution possible :

- ➔ Travailler sur des fragments pour adapter l'affichage à plusieurs tailles d'écrans.
- ➔ Améliorer l'interface graphique : menu type drawer.
- ➔ Améliorer la vitesse de prise de la photo.
- ➔ Reprendre la galerie.
- ➔ Travailler avec des String pour traduire l'application
- ➔

7. Bibliographie

AUTRES

Image libre et gratuit

<https://pixabay.com>

LudiChart – Organigramme de fonctionnement

<https://www.lucidchart.com>

DEVELOPPEMENT ANDROID

Appli Administration Appareil

<https://developer.android.com/guide/topics/admin/device-admin.html#top>

Caméra :

- SurfaceHolder

<https://developer.android.com/reference/android/view/SurfaceHolder>

- SurfaceView

<https://developer.android.com/reference/android/view/SurfaceView>

- Camera API

<https://developer.android.com/guide/topics/media/camera>

- Picture Storage

<https://developer.android.com/guide/topics/data/data-storage.html>

- Barre de Notification

<https://developer.android.com/training/notify-user/navigation>

Information Générale :

- Stackoverflow

<https://stackoverflow.com>

- Android Developer

<https://developer.android.com>

-Service + KeyEvent Power

<https://stackoverflow.com/questions/2986337/is-it-possible-to-create-an-android-service-that-listens-for-hardware-key-presse>