1 Mot de passe

On souhaite protéger l'accès à notre app de notes par un mot de passe, afin qu'un utilisateur tiers ayant accès à l'appareil (déverrouillé) ne puisse les consulter. On stockera ce mot de passe dans un simple fichier situé dans l'Internal Storage.

1. Créer une activité comportant une TextView, une EditText et un bouton labellisé "OK". On voudra modifier de façon dynamique le texte contenu dans la TextView. Il faut donc renseigner son attribut *android :id*.
2. Dans le cas où le fichier nommé "password" existe dans l'internal storage, la TextView devra afficher "entrer le mot de passe". Dans le cas contraire, elle affichera "Définir un mot de passe".
3. Faire en sorte que l'utilisateur accède à cette activité avant que la liste de ses notes ne soit affichée. S'il entre le bon mot de passe, l'appui sur le bouton OK l'enverra vers la liste de ses notes. Sinon, il déclenchera l'affichage d'un Toast "Mauvais mot de passe". Si aucun mot de passe n'était défini, l'appui sur le bouton OK stockera ce mot de passe dans le fichier *password* avant de rediriger l'utilisateur vers sa liste de notes.

2 Shared Preferences

1. Lors de la fermeture de l'activity d'édition de note autre que via le bouton enregistrer, stocker en SharedPreferences les contenu des champs d'édition de la note (titre et contenu).
2. Lors de l'ouverture de l'activity d'édition de note, pré-remplir les Edittext Titre et Contenu avec les éventuelles valeurs sauvegardées dans la question précédente.

3 Sauvegarde des notes (Sqlite)

À l'heure actuelle, l'app ne permet toujours pas de sauvegarder les notes : celles-ci sont toujours perdues dès que le processus de l'app est tué. On se propose de stocker les notes dans une BDD Sqlite afin de les récupérer.

La base de données sera accédée depuis la classe `ListeNotes` déjà existante. Afin de ne pas multiplier les accès disque au moindre appel des méthodes *get* et *size*, on continuera à conserver en RAM la liste des notes créées : la BDD sera lue et chargée en mémoire à la création de l'instance de ListeNotes, les méthodes *get* et *size* iront lire les notes en mémoire, et les méthodes de modification *ajouteNote* et *deleteNote* mettront à jour et la RAM et la base.

1. Créer une classe dérivée de *SqliteOpenHelper* permettant de créer une BDD contenant une unique table *note* sont les champs sont respectivement *id* (entier), *titre* (texte) et *contenu* (texte).
2. Dans le constructeur de *ListeNotes* récupérer la base de données en utilisant l'helper créé dans la question précédente (on ne fera que lite la base dans cette partie. On peut donc n'utiliser que *getReadableDatabase*), et créer une note à partir de chaque ligne de la table *note*.
3. Dans les méthodes *deleteNote* et *ajouteNote*, ouvrir la base de données en écriture, et y répercuter les changements correspondants.