université abdelmalek essaadi

ENSA TANGER

Département d'Informatique

Rapport de TP2

Application Android "MyNotes"

Gestion de données Multimédia

ListViews, Intents, Caméra et Stockage

Réalisé par :
Elarbi Allam

Encadré par :
Pr. Abdelilah AZYAT

Année Universitaire 2024/2025

Table des matières

1 Introduction

1.1 Contexte du projet

Après avoir acquis les bases de l'interface utilisateur lors du TP1, ce second travail pratique se concentre sur la gestion des données structurées et l'utilisation des fonctionnalités matérielles du téléphone. L'application **MyNotes** est un gestionnaire de notes personnelles permettant non seulement de stocker du texte, mais aussi d'intégrer la prise de photos.

1.2 Objectifs pédagogiques

Ce TP vise à maîtriser des concepts avancés du développement Android :

— **ListView personnalisée :** Utilisation d'un `Custom Adapter` pour afficher des données complexes avec une mise en forme conditionnelle.

— **Navigation et Transmission de données :** Utilisation des `Intents` explicites et de l'interface `Serializable` pour passer des objets entre activités.

— **Multimédia et Hardware :** Intégration de la Caméra via des Intents implicites.

— **Gestion des fichiers :** Utilisation de `FileProvider` et du stockage interne pour sécuriser et sauvegarder des photos.

2 Analyse et Conception

2.1 Fonctionnalités de l'application

L'application se décompose en plusieurs modules fonctionnels :

Fonctionnalités Clés

1. **Liste des notes :** Affichage des titres et dates avec un code couleur selon la priorité.
2. **Création de note :** Formulaire complet (Titre, Description, Date, Priorité).
3. **Détails :** Consultation de la note complète.
4. **Module Caméra :** Prise de photo en haute qualité.
5. **Galerie Privée :** Visualisation des photos stockées dans l'espace privé de l'application.

2.2 Modélisation des données (Classe Note)

Les données sont structurées autour de la classe `Note` qui implémente `Serializable` pour faciliter le transfert entre activités.

— **Attributs :** Nom (String), Description (String), Date (String), Priorité (String).
— **Niveaux de priorité :** "Haute", "Moyenne", "Basse".

3 Interface Utilisateur

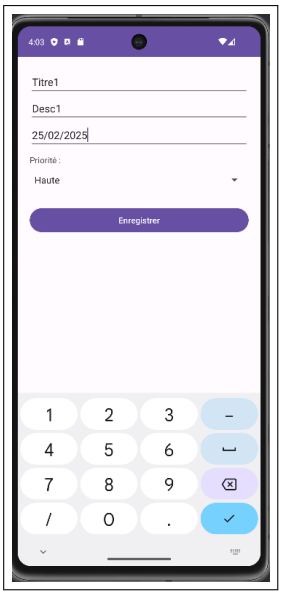
3.1 Navigation et Ergonomie

L'application démarre sur `NoteListActivity`. Un bouton flottant (FAB) permet d'ajouter une note, tandis qu'un bouton dédié ouvre le module caméra. Le clic sur un élément de la liste ouvre les détails.

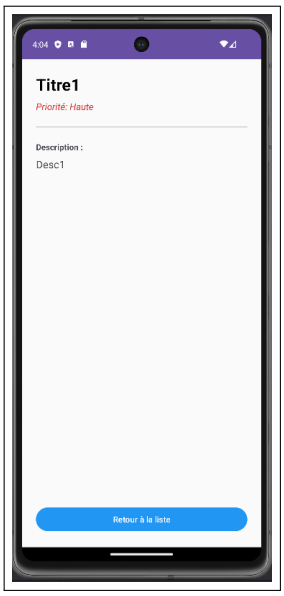
3.2 Aperçu visuel de l'application

Les captures d'écran suivantes illustrent le flux principal de l'application **MyNotes**, mettant en évidence la gestion des priorités par couleur et l'intégration multimédia.

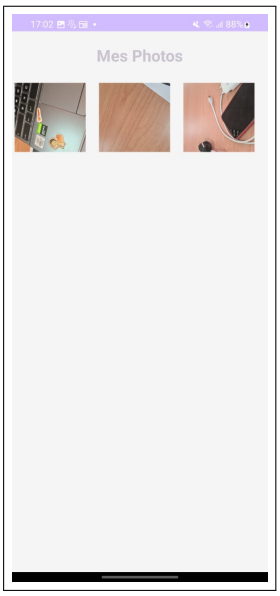
(a) Liste principale avec code couleur selon la priorité



(b) Formulaire de création d'une nouvelle note



(c) Consultation des détails d'une note spécifique



(d) Galerie privée stockant les photos prises in-app

FIGURE 1 – Vue d'ensemble des fonctionnalités clés de l'application MyNotes

4 Implémentation Technique

4.1 L'Adapter Personnalisé (NoteAdapter)

C'est le cœur de l'affichage. J'ai redéfini la méthode `getView` pour modifier dynamiquement la couleur de fond de chaque item selon la priorité de la note.

```
Override
publicView getView(int position, View convertView, ViewGroup parent) {
    // ... Inflation de la vue ...

    // Logique de couleur selon la priorit
    if (note.getPriorite().equals("Haute")) {
        layoutRoot.setBackgroundColor(Color.parseColor("#FFCDD2")); //
            Rouge
    } else if (note.getPriorite().equals("Moyenne")) {
        layoutRoot.setBackgroundColor(Color.parseColor("#FFF9C4")); //
            Jaune
    } else {
        layoutRoot.setBackgroundColor(Color.parseColor("#C8E6C9")); //
            Vert
    }
    return convertView;
}
```

Listing 1 – Logique de coloration dans NoteAdapter.java

4.2 Gestion de la Caméra et FileProvider

Pour gérer la sécurité d'Android (Scoped Storage), l'accès direct aux fichiers est restreint. J'ai dû implémenter un `FileProvider`.

4.2.1 Configuration Manifeste

```
<queries>
    <intent>
        <action android:name="android.media.action.IMAGE_CAPTURE" />
    </intent>
</queries>

<provider
    android:name="androidx.core.content.FileProvider"
    android:authorities="com.elarbiallam.notes.fileprovider"
    android:exported="false"
    android:grantUriPermissions="true">
    <meta-data
        android:name="android.support.FILE_PROVIDER_PATHS"
        android:resource="@xml/file_paths" />
</provider>
```

Listing 2 – Déclaration du Provider et Queries

4.2.2 Logique de Stockage (CameraActivity)

Les photos sont stockées dans le dossier privé de l'application (`getExternalFilesDir`) pour garantir qu'elles restent associées à l'application.

```
private File createImageFile() throws IOException {
```

```
2       String timeStamp = new SimpleDateFormat("yyyyMMdd_HHmmss").format(
            new Date());
3       String imageFileName = "NOTE_" + timeStamp + "_";
4       // Stockage interne priv
5       File storageDir = getExternalFilesDir(Environment.DIRECTORY_PICTURES
            );
6       return File.createTempFile(imageFileName, ".jpg", storageDir);
7   }
```

Listing 3 – Création sécurisée du fichier photo

4.3 Galerie Privée (GalleryActivity)

Au lieu d'utiliser la galerie publique, j'ai créé une activité qui scanne le dossier interne de l'application et affiche les photos prises dans une `GridView`. Cela permet de garder les notes "privées".

5 Difficultés et Solutions

5.1 Problème de lancement (Manifest)

Erreur

L'application démarrait toujours sur "Hello World" (MainActivity) au lieu de ma liste de notes, ou crashait au démarrage.

Solution : J'ai dû modifier le fichier `AndroidManifest.xml` pour déplacer le filtre `<intent-filter>` contenant `LAUNCHER` vers `NoteListActivity` et supprimer l'activité par défaut générée par Android Studio.

5.2 Crash de la Caméra sur Android 11+

Erreur

La caméra ne s'ouvrait pas ou l'application plantait avec une erreur de "Resource Linking" liée aux `<queries>`.

Solution :

1. Ajout du bloc `<queries>` dans le Manifeste pour la visibilité du package.
2. Création correcte du dossier `res/xml` et du fichier `file_paths.xml` référencé par le FileProvider.
3. Utilisation de `ContextCompat.checkSelfPermission` pour gérer les permissions dynamiques.

6 Conclusion

Ce TP2 a été une étape cruciale dans l'apprentissage d'Android. Il m'a permis de comprendre que le développement mobile va bien au-delà de l'interface graphique : il faut gérer le cycle de vie, la persistance des données (même temporaire) et les permissions matérielles.

J'ai particulièrement apprécié la mise en place de l'Adapter personnalisé qui donne un aspect professionnel à l'application, ainsi que la création de la galerie privée qui m'a fait comprendre le fonctionnement du système de fichiers Android.

Livrables

Code Source complet : Disponible sur le dépôt GitHub associé.
Le projet respecte l'architecture standard Android et les conventions de nommage.