Environnements Logiciels pour l'Informatique Mobile

Android : Persistance des données

Présentation

- Polytech'Nice-Sophia 2012 (IAM)
- 5 ans chez Sopra-Steria
  - Développement
  - Architecture
  - Projets innovants
  - Formation
- gregory.marro@soprasteria.com

Sommaire

- Introduction
- Sauvegarder des préférences
- Écrire un fichier
- Gérer une base de données

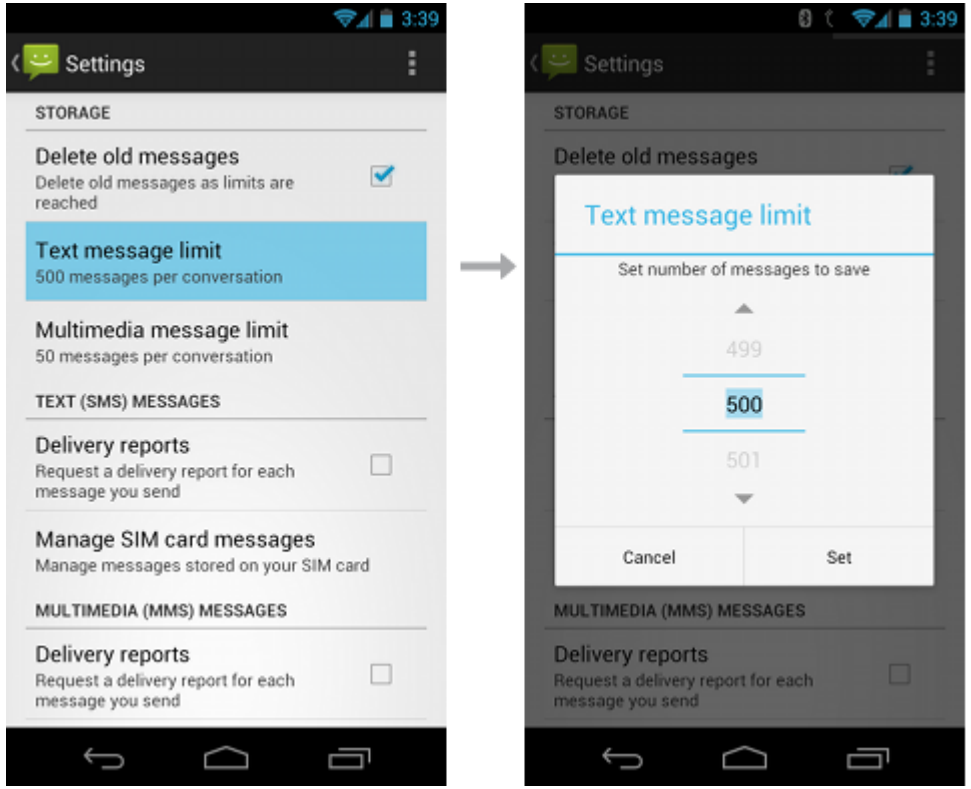
Introduction

- De nombreuses applications nécessitent de la persistance de données
  - Paramètres utilisateurs
  - Données de synchronisation
  - Volume de données important à transférer
  - Données persistantes pour l'application

- Plusieurs solutions Android

Sauvegarder des préférences

- Permet de rendre pérenne un petit volume de données
- Généralement utilisé pour les préférences utilisateurs
- Souvent associé aux Settings
- Sous forme clé/valeur



Sauvegarder des préférences

- Utilisation de la classe SharedPreferences

```
SharedPreferences sharedPref = getActivity().getPreferences(Context.MODE_PRIVATE);
```

- Lecture directement de SharedPreferences

```
long highScore = sharedPref.getInt(getString(R.string.saved_high_score), defaultValue);
```

- Modification à l'aide du SharedPreferences.Editor

```
SharedPreferences.Editor editor = sharedPref.edit();
editor.putInt(getString(R.string.saved_high_score), newHighScore);
```

- Ne pas oublier le commit !

```
editor.commit();
```

Écriture dans les fichiers

- Dans le cas de données volumineuses (par exemple : images, vidéos …)

- 2 solutions :

Interne :
- Toujours disponible
- Uniquement accessible par l'application
- Désinstallation « propre »

Externe :
- Disponible si elle existe et si elle est montée
- Partagé sur tout le téléphone
- Restriction pour la désinstallation
- Permissions requises

Écriture d’un fichier en interne

- 2 types de fichier :
  - Classique : getFilesDir
  - Cache : getCacheDir

```
File file = new File(context.getFilesDir(), filename);
```

- Utilisation de FileOutputStream

- Ne pas oublier le close()

```
outputStream = openFileOutput(filename, Context.MODE_PRIVATE);
outputStream.write(string.getBytes());
outputStream.close();
```

Écriture dans un fichier externe

- Ajout d'une permission au Manifest
- Il faut vérifier que l'on a accès au stockage externe
  - Grace à Environment.getExternalStorageState();
- Puis identique au fichier interne

Gestion de la base de données

- Utilisation de SQLite (écriture dans un fichier)
- Une base de donnée n’est pas partagée
- Utilisation conseillée de SQLiteOpenHelper
- Dans un Thread séparé

Gestion de la base de données

- Création d'une classe qui hérite de SQLiteOpenHelper

```
public class FeedReaderDbHelper extends SQLiteOpenHelper {
```

- Surcharge des méthodes :

```
public void onCreate(SQLiteDatabase db) {
    db.execSQL(SQL_CREATE_ENTRIES);
}
public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
    // This database is only a cache for online data, so its upgrade policy is
    // to simply to discard the data and start over
    db.execSQL(SQL_DELETE_ENTRIES);
    onCreate(db);
}
public void onDowngrade(SQLiteDatabase db, int oldVersion, int newVersion) {
    onUpgrade(db, oldVersion, newVersion);
}
```

- Utilisation de la base en lecture/écriture

```
SQLiteDatabase db = mDbHelper.getWritableDatabase();

SQLiteDatabase db = mDbHelper.getReadableDatabase();
```

TP : Application météo

- Créer une application qui affiche les prévisions météo dans une liste à partir du point actuel ou d'une ville, si elle a été ajoutée dans les préférences :
  - API OpenWeatherMap : http://openweathermap.org/forecast5#geo5
  - Clé API : c5be641c0caea8bab91e5eaf884bfccc

- Au runtime, récupérer et stocker dans des fichiers des icones pour le temps et les afficher dans la liste en fonction du temps prévu

- Stocker en base de données l'historique des choix de villes (geocodé ou entré par l'utilisateur) pour l'utiliser dans une AutoCompleteTextView