Dev Android - notions fondamentales

Jérémie Vautard

2019 - 2020

Section 1

Fragments

Fragments

Généralités

Composants graphiques complexes et autonomes

“Mini-activity”

Fragments

Généralités

Composants graphiques complexes et autonomes

"Mini-activity"

- ont leur propre contentView

Fragments

Généralités

Composants graphiques complexes et autonomes

"Mini-activity"

- ont leur propre contentView
- gèrent leurs événements graphiques

Fragments

Généralités

Composants graphiques complexes et autonomes

"Mini-activity"

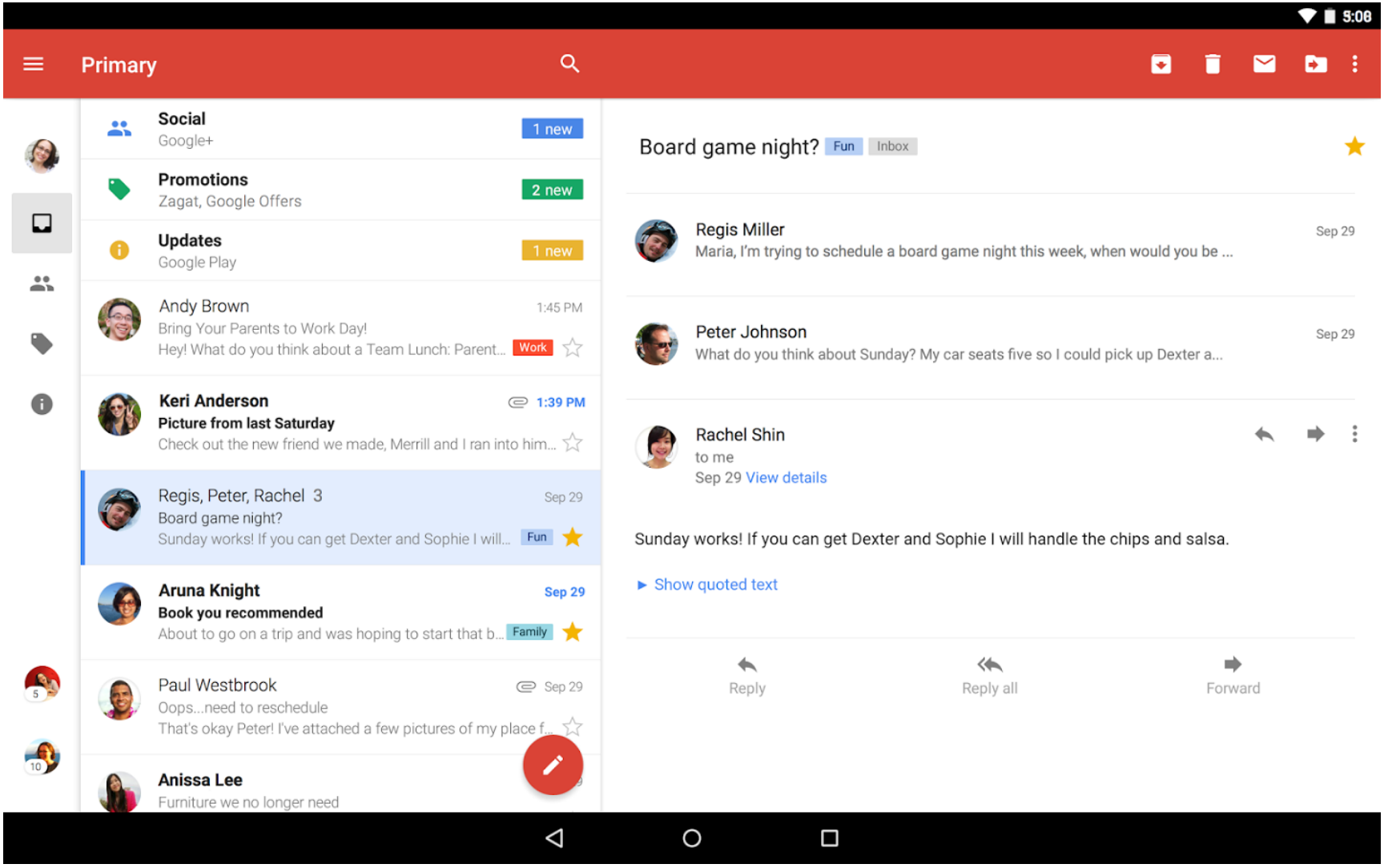
- ont leur propre contentView
- gèrent leurs événements graphiques
- ont besoin d'une activity pour s'afficher

Fragments

Cas d'utilisations

- Adaptation selon le device sans duplication du code
- Activités "tout en un"

Fragments (image ©Google)



Fragments

Utilisation

Fragments

Utilisation

- hériter de FragmentActivity

Fragments

Utilisation

- hériter de FragmentActivity
- statique : balise <fragment> dans le layout de l'activity

Fragments

Utilisation

- hériter de FragmentActivity
- statique : balise <fragment> dans le layout de l'activity
- dynamique : dans le programme
  - géré par FragmentTransaction

Fragments

Utilisation

- hériter de FragmentActivity
- statique : balise <fragment> dans le layout de l'activity
- dynamique : dans le programme
  - géré par FragmentTransaction
  - dans le XML : prévoir un FrameLayout

Fragments

Utilisation

- hériter de FragmentActivity
- statique : balise <fragment> dans le layout de l'activity
- dynamique : dans le programme
  - géré par FragmentTransaction
  - dans le XML : prévoir un FrameLayout
  - `getFragmentManager.beginTransaction()`

Fragments

Utilisation

- hériter de FragmentActivity
- statique : balise <fragment> dans le layout de l'activity
- dynamique : dans le programme
  - géré par FragmentTransaction
  - dans le XML : prévoir un FrameLayout
  - `getFragmentManager.beginTransaction()`
  - méthodes `add`, `remove`, `replace`

Fragments

Utilisation

- hériter de FragmentActivity
- statique : balise <fragment> dans le layout de l'activity
- dynamique : dans le programme
  - géré par FragmentTransaction
  - dans le XML : prévoir un FrameLayout
  - `getFragmentManager.beginTransaction()`
  - méthodes `add`, `remove`, `replace`
  - `attach` et `detach` pour conserver les instances de Fragment

Fragments

Utilisation

- hériter de FragmentActivity
- statique : balise <fragment> dans le layout de l'activity
- dynamique : dans le programme
  - géré par FragmentTransaction
  - dans le XML : prévoir un FrameLayout
  - `getFragmentManager.beginTransaction()`
  - méthodes `add`, `remove`, `replace`
  - `attach` et `detach` pour conserver les instances de Fragment
  - ne pas oublier `commit()` à la fin

Section 2

Stockage de données

Stockage en local

Généralités

3 types d'emplacement

Stockage en local

Généralités

3 types d'emplacement

- assets : ressources (non typées) de l'app lecture seule

Stockage en local

Généralités

3 types d'emplacement

- assets : ressources (non typées) de l'app lecture seule
- Internal storage : fichiers privés de l'app

Stockage en local

Généralités

3 types d'emplacement

- assets : ressources (non typées) de l'app lecture seule
- Internal storage : fichiers privés de l'app
- External storage : fichiers publics

Stockage en local

Internal Storage

Stockage en local

Internal Storage

- inaccessibles de l'extérieur de l'app (chmod 700)

Stockage en local

Internal Storage

- inaccessibles de l'extérieur de l'app (chmod 700)
- supprimés à la désinstallation

Stockage en local

Internal Storage

- inaccessibles de l'extérieur de l'app (chmod 700)
- supprimés à la désinstallation

External Storage

Stockage en local

Internal Storage

- inaccessibles de l'extérieur de l'app (chmod 700)
- supprimés à la désinstallation

External Storage

- accessibles à tous (lecture écriture effacement)

Stockage en local

Internal Storage

- inaccessibles de l'extérieur de l'app (chmod 700)
- supprimés à la désinstallation

External Storage

- accessibles à tous (lecture écriture effacement)
- peuvent être indexés (selon l'emplacement)

Stockage en local

Internal Storage

- inaccessibles de l'extérieur de l'app (chmod 700)
- supprimés à la désinstallation

External Storage

- accessibles à tous (lecture écriture effacement)
- peuvent être indexés (selon l'emplacement)
- persistants après désinstallation

Stockage en local

Au delà du fichier

Méthodes de stockage de haut niveau proposées par Android

Stockage en local

Au delà du fichier

Méthodes de stockage de haut niveau proposées par Android

- Shared Preferences : couples clé-valeur

Stockage en local

Au delà du fichier

Méthodes de stockage de haut niveau proposées par Android

- Shared Preferences : couples clé-valeur
- SQLite : Base de données SQL stockée dans un fichier

Stockage en local

Au delà du fichier

Méthodes de stockage de haut niveau proposées par Android

- Shared Preferences : couples clé-valeur
- SQLite : Base de données SQL stockée dans un fichier

Cas particuliers

- fichiers cache que le système peut effacer automatiquement pour gagner de la place

Stockage en local

Au delà du fichier

Méthodes de stockage de haut niveau proposées par Android

- Shared Preferences : couples clé-valeur
- SQLite : Base de données SQL stockée dans un fichier

Cas particuliers

- fichiers cache que le système peut effacer automatiquement pour gagner de la place
- external files dir : répertoire particulier en external storage
  - pas indexé par le système
  - supprimé à la désinstallation

Stockage local - Coté dev

Accès aux fichiers

Méthodes de la classe Context (Activity) renvoyant des FileInputStream et FileOutputStream

Stockage local - Coté dev

Accès aux fichiers

Méthodes de la classe Context (Activity) renvoyant des FileInputStream et FileOutputStream

- accès aux assets : getAssets().open(nom)

Stockage local - Coté dev

Accès aux fichiers

Méthodes de la classe Context (Activity) renvoyant des FileInputStream et FileOutputStream

- accès aux assets : getAssets().open(nom)
- Dans l'internal storage :
  - openFileOutput(nom,MODE_PRIVATE) (ou MODE_APPEND)
  - openFileInput(nom)

Stockage local - Coté dev

Accès aux fichiers publics

- Permissions requises : READ_EXTERNAL_STORAGE et WRITE_EXTERNAL_STORAGE
- Verifier la disponibilité de l'external storage

Stockage local - Coté dev

Accès aux fichiers publics

- Permissions requises : READ_EXTERNAL_STORAGE et WRITE_EXTERNAL_STORAGE
- Verifier la disponibilité de l'external storage

```
public boolean isExternalStorageWritable() {
    String state = Environment.getExternalStorageState();
    if (Environment.MEDIA_MOUNTED.equals(state)) {
        return true;
    }
    return false;
}
public boolean isExternalStorageReadable() {
    String state = Environment.getExternalStorageState();
    if (Environment.MEDIA_MOUNTED.equals(state) ||
        Environment.MEDIA_MOUNTED_READ_ONLY.equals(state)) {
        return true;
    }
    return false;
}
```

Stockage local - Coté dev

Accès aux fichiers publics

- Accès au répertoire général via Environment.getExternalStorageDirectory()

Stockage local - Coté dev

Accès aux fichiers publics

- Accès au répertoire général via Environment.getExternalStorageDirectory()
- OU au répertoire approprié via Environment.getExternalStoragePublicDirectory(...)
  - Environment.DIRECTORY_PICTURES
  - Environment.DIRECTORY_MOVIES
  - Environment.DIRECTORY_MUSIC (etc.)

Stockage local - Coté dev

Accès aux fichiers publics

- Accès au répertoire général via Environment.getExternalStorageDirectory()
- OU au répertoire approprié via Environment.getExternalStoragePublicDirectory(...)
  - Environment.DIRECTORY_PICTURES
  - Environment.DIRECTORY_MOVIES
  - Environment.DIRECTORY_MUSIC (etc.)
- Ouverture du fichier comme en Java standard (new File(directory,name))

Shared Preferences

Shared Preferences

Fichier de couples clé-valeur, stocké dans l'Internal Storage.

Shared Preferences

Shared Preferences

Fichier de couples clé-valeur, stocké dans l'Internal Storage.

- Création/accès via Context.getSharedPreferences(nom,MODE_PRIVATE)

Shared Preferences

Shared Preferences

Fichier de couples clé-valeur, stocké dans l'Internal Storage.

- Création/accès via Context.getSharedPreferences(nom,MODE_PRIVATE)
- récupération des valeurs via getType(cle,default)

Shared Preferences

Shared Preferences

Fichier de couples clé-valeur, stocké dans l'Internal Storage.

- Création/accès via Context.getSharedPreferences(nom,MODE_PRIVATE)
- récupération des valeurs via getType(cle,default)
- Mise à jour via une classe Editor :

Shared Preferences

Shared Preferences

Fichier de couples clé-valeur, stocké dans l'Internal Storage.

- Création/accès via Context.getSharedPreferences(nom,MODE_PRIVATE)
- récupération des valeurs via getType(cle,default)
- Mise à jour via une classe Editor :
  - à récupérer via SharedPreferences.edit()

Shared Preferences

Shared Preferences

Fichier de couples clé-valeur, stocké dans l'Internal Storage.

- Création/accès via Context.getSharedPreferences(nom,MODE_PRIVATE)
- récupération des valeurs via getType(cle,default)
- Mise à jour via une classe Editor :
  - à récupérer via SharedPreferences.edit()
  - modifier : Editor.putType(cle,valeur)

Shared Preferences

Shared Preferences

Fichier de couples clé-valeur, stocké dans l'Internal Storage.

- Création/accès via Context.getSharedPreferences(nom,MODE_PRIVATE)
- récupération des valeurs via getType(cle,default)
- Mise à jour via une classe Editor :
  - à récupérer via SharedPreferences.edit()
  - modifier : Editor.putType(cle,valeur)
  - commit des modifs dans le fichier : Editor.commit()

Shared Preferences

Accès

```
// Restore preferences
SharedPreferences settings = getSharedPreferences(PREFS_NAME, 0);
boolean silent = settings.getBoolean("silentMode", false);
```

Shared Preferences

Accès

```
// Restore preferences
SharedPreferences settings = getSharedPreferences(PREFS_NAME, 0);
boolean silent = settings.getBoolean("silentMode", false);
```

Modification

```
SharedPreferences settings = getSharedPreferences(PREFS_NAME, 0);
SharedPreferences.Editor editor = settings.edit();
editor.putBoolean("silentMode", mSilentMode);

// Commit the edits!
editor.commit();
```

Sqlite - Ancienne méthode

BDD Sqlite

Base de données relationnelle stockée dans un fichier

Sqlite - Ancienne méthode

BDD Sqlite

Base de données relationnelle stockée dans un fichier

- Support des mises à jour de la BDD

Sqlite - Ancienne méthode

BDD Sqlite

Base de données relationnelle stockée dans un fichier

- Support des mises à jour de la BDD
- Stockage dans l'Internal Storage

Sqlite - Ancienne méthode

BDD Sqlite

Base de données relationnelle stockée dans un fichier

- Support des mises à jour de la BDD
- Stockage dans l'Internal Storage

classe Helper (à étendre)

- Mise en place des tables à la création (onCreate())

Sqlite - Ancienne méthode

BDD Sqlite

Base de données relationnelle stockée dans un fichier

- Support des mises à jour de la BDD
- Stockage dans l'Internal Storage

classe Helper (à étendre)

- Mise en place des tables à la création (onCreate())
- Support de versions de la BDD
  - Mise à jour de la BDD (onUpgrade(db,oldV,newV))
  - Retour à une version précédente (onDowngrade(db,oldV,newV))

Sqlite - Ancienne méthode

Exemple

```
public class DictionaryOpenHelper extends SQLiteOpenHelper {

    private static final int DATABASE_VERSION = 2;
    private static final String DICTIONARY_TABLE_NAME = "dictionary";
    private static final String DICTIONARY_TABLE_CREATE =
                "CREATE TABLE " + DICTIONARY_TABLE_NAME + " (" +
                KEY_WORD + " TEXT, " +
                KEY_DEFINITION + " TEXT);";

    DictionaryOpenHelper(Context context) {
        super(context, DATABASE_NAME, null, DATABASE_VERSION);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        db.execSQL(DICTIONARY_TABLE_CREATE);
    }
}
```

Sqlite - Ancienne méthode

Utilisation

Méthodes getReadableDatabase() et getWritableDatabase()

- créent ou mettent à jour la BDD
- retournent une instance de SQLiteDatabase

Sqlite - Ancienne méthode

Utilisation

Méthodes getReadableDatabase() et getWritableDatabase()

- créent ou mettent à jour la BDD
- retournent une instance de SQLiteDatabase

SQLiteDatabase

- Requête bas niveau : rawQuery()
- méthodes de requête/d'insertion plus commodes

Sqlite - Ancienne méthode

Select

```
public Cursor query (
        String table,
        String[] columns,
        String whereClause,
        String[] selectionArgs,
        String groupBy,
        String having,
        String orderBy )
```

Sqlite - Ancienne méthode

Cursor

- Permet de naviguer dans les lignes du résultat de la requête :
  - Nombre d'éléments : getCount()
  - Se positionner au début : moveToFirst()
  - Aller au suivant : moveToNext()

Sqlite - Ancienne méthode

Cursor

- Permet de naviguer dans les lignes du résultat de la requête :
  - Nombre d'éléments : getCount()
  - Se positionner au début : moveToFirst()
  - Aller au suivant : moveToNext()
- Accès aux éléments d'une ligne :
  - Nombre de colonnes : getColumnCount()
  - Nom d'une colonne : getColumnName(int)
  - Index d'une colonne : getColumnIndex(String)

Sqlite - Ancienne méthode

Cursor

- Permet de naviguer dans les lignes du résultat de la requête :
  - Nombre d'éléments : getCount()
  - Se positionner au début : moveToFirst()
  - Aller au suivant : moveToNext()
- Accès aux éléments d'une ligne :
  - Nombre de colonnes : getColumnCount()
  - Nom d'une colonne : getColumnName(int)
  - Index d'une colonne : getColumnIndex(String)
  - récupération d'un élément getXXX(int)

Sqlite - Ancienne méthode

Cursor

- Permet de naviguer dans les lignes du résultat de la requête :
  - Nombre d'éléments : getCount()
  - Se positionner au début : moveToFirst()
  - Aller au suivant : moveToNext()
- Accès aux éléments d'une ligne :
  - Nombre de colonnes : getColumnCount()
  - Nom d'une colonne : getColumnName(int)
  - Index d'une colonne : getColumnIndex(String)
  - récupération d'un élément getXXX(int)
- fermer quand on a fini :close()

Sqlite - Ancienne méthode

Insert

- Classe ContentValues : map des noms de colonnes vers les valeurs de la ligne à ajouter
  - ajout d'une valeur : put(Colonne,valeur)

Sqlite - Ancienne méthode

Insert

- Classe ContentValues : map des noms de colonnes vers les valeurs de la ligne à ajouter
  - ajout d'une valeur : put(Colonne,valeur)
- public long insert (
  String table,
  String nullColumnHack,
  ContentValues values)

Sqlite - Room

Room

- Bibliothèque de plus haut niveau

Sqlite - Room

Room

- Bibliothèque de plus haut niveau
  - vision BDD objet de la base

Sqlite - Room

Room

- Bibliothèque de plus haut niveau
  - vision BDD objet de la base
  - fort couplage table - objets

Sqlite - Room

Room

- Bibliothèque de plus haut niveau
  - vision BDD objet de la base
  - fort couplage table - objets
  - beaucoup de code SQL autogénéré

Sqlite - Room

Room

- Bibliothèque de plus haut niveau
  - vision BDD objet de la base
  - fort couplage table - objets
  - beaucoup de code SQL autogénéré
- déclarations via les @annotations java/Kotlin

Sqlite - Room

Room

- Bibliothèque de plus haut niveau
  - vision BDD objet de la base
  - fort couplage table - objets
  - beaucoup de code SQL autogénéré
- déclarations via les @annotations java/Kotlin

Par rapport à l'ancienne méthode

Sqlite - Room

Room

- Bibliothèque de plus haut niveau
  - vision BDD objet de la base
  - fort couplage table - objets
  - beaucoup de code SQL autogénéré
- déclarations via les @annotations java/Kotlin

Par rapport à l'ancienne méthode

- vérification des requêtes SQL à la compilation

Sqlite - Room

Room

- Bibliothèque de plus haut niveau
  - vision BDD objet de la base
  - fort couplage table - objets
  - beaucoup de code SQL autogénéré
- déclarations via les @annotations java/Kotlin

Par rapport à l'ancienne méthode

- vérification des requêtes SQL à la compilation
- force à créer du beau code

Sqlite - Room

Room

- Bibliothèque de plus haut niveau
  - vision BDD objet de la base
  - fort couplage table - objets
  - beaucoup de code SQL autogénéré
- déclarations via les @annotations java/Kotlin

Par rapport à l'ancienne méthode

- vérification des requêtes SQL à la compilation
- force à créer du beau code
- coté "magique" des annotations

Room - Composants principaux

Entity

- représente une (ligne de) table

Room - Composants principaux

Entity

- représente une (ligne de) table
- déclaration via annotations

Room - Composants principaux

Entity

- représente une (ligne de) table
- déclaration via annotations
  - @entity : sur la classe elle-même

Room - Composants principaux

Entity

- représente une (ligne de) table
- déclaration via annotations
  - @entity : sur la classe elle-même
  - @columninfo : lier un attribut à une colonne

Room - Composants principaux

Entity

- représente une (ligne de) table
- déclaration via annotations
  - @entity : sur la classe elle-même
  - @columninfo : lier un attribut à une colonne
  - @primarykey : indique la clé primaire (attribut également)

Room - Composants principaux

DAO

- objet d'accès aux données

Room - Composants principaux

DAO

- objet d'accès aux données
- Interface annotée (avec les requêtes SQL)

Room - Composants principaux

DAO

- objet d'accès aux données
- Interface annotée (avec les requêtes SQL)
  - @Dao : sur l'interface

Room - Composants principaux

DAO

- objet d'accès aux données
- Interface annotée (avec les requêtes SQL)
  - @Dao : sur l'interface
  - @query : méthode correspondant à une requête SQL custom

Room - Composants principaux

DAO

- objet d'accès aux données
- Interface annotée (avec les requêtes SQL)
  - @Dao : sur l'interface
  - @query : méthode correspondant à une requête SQL custom
  - @insert et @delete : ajout-suppression
    (liste d')Entity en paramètre

Room - Composants principaux

Database

- La BDD en elle-même

Room - Composants principaux

Database

- La BDD en elle-même
- classe anstraite héritant de `RoomDatabase`

Room - Composants principaux

Database

- La BDD en elle-même
- classe anstraite héritant de `RoomDatabase`
- instancié par un Room.databaseBuilder

Room - Composants principaux

Database

- La BDD en elle-même
- classe anstraite héritant de `RoomDatabase`
- instancié par un Room.databaseBuilder
- méthode abstraite pour obtenir une instance du DAO

Room - Exemple

Entity

```
@Entity
public class User {
    @PrimaryKey
    public int uid;

    @ColumnInfo(name = "first_name")
    public String firstName;

    @ColumnInfo(name = "last_name")
    public String lastName;
}
```

Room - Exemple

DAO

```
@Dao
public interface UserDao {
    @Query("SELECT * FROM user")
    List<User> getAll();

    @Query("SELECT * FROM user WHERE uid IN (:userIds)")
    List<User> loadAllByIds(int[] userIds);

    @Query("SELECT * FROM user WHERE first_name LIKE :first AND " +
           "last_name LIKE :last LIMIT 1")
    User findByName(String first, String last);

    @Insert
    void insertAll(User... users);

    @Delete
    void delete(User user);
}
```

Room - Exemple

Database

```
@Database(entities = {User.class}, version = 1)
public abstract class AppDatabase extends RoomDatabase {
    public abstract UserDao userDao();
}
```

```
AppDatabase db = Room.databaseBuilder(getApplicationContext(),
        AppDatabase.class, "database-name").build();
```

Section 3

Multithreading

Threads

Vue d'ensemble

Exécution parallèle de différentes portions du code de l'app

Threads

Vue d'ensemble

Exécution parallèle de différentes portions du code de l'app

- chaque thread a sa propre pile d'appel

Threads

Vue d'ensemble

Exécution parallèle de différentes portions du code de l'app

- chaque thread a sa propre pile d'appel
- reste de la mémoire partagée

Threads

Vue d'ensemble

Exécution parallèle de différentes portions du code de l'app

- chaque thread a sa propre pile d'appel
- reste de la mémoire partagée

En java pur

- Classe Thread : gestion (start, stop, join...)

Threads

Vue d'ensemble

Exécution parallèle de différentes portions du code de l'app

- chaque thread a sa propre pile d'appel
- reste de la mémoire partagée

En java pur

- Classe Thread : gestion (start, stop, join...)
- Objet Runnable : code à exécuter en parallèle (méthode run())

Threads Android

Principe

Gestion des threads plus élaborée : HandlerThread

Threads Android

Principe

Gestion des threads plus élaborée : HandlerThread

- exécutent une boucle (Looper.loop)

Threads Android

Principe

Gestion des threads plus élaborée : HandlerThread

- exécutent une boucle (Looper.loop)
- reçoivent des messages (et des Runnable) via la classe Handler

Threads Android

Principe

Gestion des threads plus élaborée : HandlerThread

- exécutent une boucle (Looper.loop)
- reçoivent des messages (et des Runnable) via la classe Handler

Threads prédéfinis

Threads Android

Principe

Gestion des threads plus élaborée : HandlerThread

- exécutent une boucle (Looper.loop)
- reçoivent des messages (et des Runnable) via la classe Handler

Threads prédéfinis

- 1 thread principal (“Graphique”)
  - gestion de l'UI
  - doit tourner rapidement

Threads Android

Principe

Gestion des threads plus élaborée : HandlerThread

- exécutent une boucle (Looper.loop)
- reçoivent des messages (et des Runnable) via la classe Handler

Threads prédéfinis

- 1 thread principal (“Graphique”)
  - gestion de l'UI
  - doit tourner rapidement

Threads Android

Principe

Gestion des threads plus élaborée : HandlerThread

- exécutent une boucle (Looper.loop)
- reçoivent des messages (et des Runnable) via la classe Handler

Threads prédéfinis

- 1 thread principal (“Graphique”)
  - gestion de l'UI
  - doit tourner rapidement
- 1 ou plusieurs threads secondaires
  - Traitements lourds
  - Traitement non immédiats (e.g. réseau)

Threads

Dans le code

Classe AsyncTask : Tâche à exécuter dans le thread secondaire

Runnable amélioré, fonctionnant en 4 étapes

Threads

Dans le code

Classe AsyncTask : Tâche à exécuter dans le thread secondaire
Runnable amélioré, fonctionnant en 4 étapes

- Thread graphique - pré-exécution

Threads

Dans le code

Classe AsyncTask : Tâche à exécuter dans le thread secondaire

Runnable amélioré, fonctionnant en 4 étapes

- Thread graphique - pré-exécution
- Thread secondaire - exécution de la tâche

Threads

Dans le code

Classe AsyncTask : Tâche à exécuter dans le thread secondaire

Runnable amélioré, fonctionnant en 4 étapes

- Thread graphique - pré-exécution
- Thread secondaire - exécution de la tâche
- Annonce de la progression (plusieurs fois)
  - Thread secondaire - Annonce
  - Thread graphique - publication

Threads

Dans le code

Classe AsyncTask : Tâche à exécuter dans le thread secondaire

Runnable amélioré, fonctionnant en 4 étapes

- Thread graphique - pré-exécution
- Thread secondaire - exécution de la tâche
- Annonce de la progression (plusieurs fois)
  - Thread secondaire - Annonce
  - Thread graphique - publication
- Thread graphique - post-exécution

AsyncTask

Implémentation

```
public class AsyncTaskExample extends AsyncTask<String,Integer,Double> {

    @Override
    protected void onPreExecute() { super.onPreExecute(); }
    @Override
    protected Double doInBackground(String... strings) {
        // strings a été passé par execute(...)
        publishProgress(Integer.valueOf(42));
        return null;
    }
    @Override
    protected void onPostExecute(Double aDouble) { super.onPostExecute(aDouble); }
    @Override
    protected void onProgressUpdate(Integer... values) { super.onProgressUpdate(values); }
}
```

AsyncTask

Utilisation

Méthode execute() de AsyncTask

AsyncTask

Utilisation

Méthode execute() de AsyncTask

- À appeler dans le thread graphique !

AsyncTask

Utilisation

Méthode execute() de AsyncTask

- À appeler dans le thread graphique !
- Pour s'en assurer : méthode runOnUIThread(Runnable r) de Activity

AsyncTask

Utilisation

Méthode execute() de AsyncTask

- À appeler dans le thread graphique !
- Pour s'en assurer : méthode runOnUIThread(Runnable r) de Activity

```
this.runOnUiThread(new Runnable() {
    @Override
    public void run() {
        maTache.execute();
    }
});
```

Contraintes d'exécution

Seulement dans le thread graphique

- méthodes réimplémentées de Activity : onCreate ... onDestroy (automatique)
- toute méthode de tout élément graphique
- tout traitement de toute action utilisateur (automatique)

Contraintes d'exécution

Seulement dans le thread graphique

- méthodes réimplémentées de Activity : onCreate ... onDestroy (automatique)
- toute méthode de tout élément graphique
- tout traitement de toute action utilisateur (automatique)

Seulement dans un thread secondaire

- Tout appel réseau
- toute méthode bloquante en général

Contraintes d'exécution

Seulement dans le thread graphique

- méthodes réimplémentées de Activity : onCreate ... onDestroy (automatique)
- toute méthode de tout élément graphique
- tout traitement de toute action utilisateur (automatique)

Seulement dans un thread secondaire

- Tout appel réseau
- toute méthode bloquante en général

Attention

IllegalStateException si non respecté !

Section 4

Accès réseau

Connexion réseau

Connexion HTTP

Comme en Java :

Connexion réseau

Connexion HTTP

Comme en Java :

- Définir une instance de la classe URL
  url = new URL("http ://www.exemple.com"

Connexion réseau

Connexion HTTP

Comme en Java :

- Définir une instance de la classe URL
  url = new URL("http ://www.exemple.com"
- Récupérer un HttpURLConnection
  cnx = url.openConnection()

Connexion réseau

Connexion HTTP

Comme en Java :

- Définir une instance de la classe URL
  url = new URL("http ://www.exemple.com"
- Récupérer un HttpURLConnection
  cnx = url.openConnection()
- récupérer les streams
  cnx.getInputStream() ; cnx.getOutputStream()

Parsing

Parsing JSON

via la classe JSONObject

Parsing

Parsing JSON

via la classe JSONObject

- Construction via JSONObject(String json)

Parsing

Parsing JSON

via la classe JSONObject

- Construction via JSONObject(String json)
- Récupération d'items
  - getXXX(nom) : Exception si la valeur n'existe pas
  - opeXXX(nom,defaut) : peut retrouner une valeur par défaut

Parsing

Parsing JSON

via la classe JSONObject

- Construction via JSONObject(String json)
- Récupération d'items
  - getXXX(nom) : Exception si la valeur n'existe pas
  - opeXXX(nom,defaut) : peut retrouner une valeur par défaut
- Sous-structure JSON : getJSONObject(nom)

Parsing

Parsing XML

Via les classes DocumentBuilderFactory, DocumentBuilder et Document :

Parsing

Parsing XML

Via les classes DocumentBuilderFactory, DocumentBuilder et Document :

- récupérer une instance de DocumentBuilderFactory : dbf = DocumentBuilderFactory.newInstance() (statique)

Parsing

Parsing XML

Via les classes DocumentBuilderFactory, DocumentBuilder et Document :

- récupérer une instance de DocumentBuilderFactory : dbf = DocumentBuilderFactory.newInstance() (statique)
- l'utiliser pour construire un DocumentBuilder : db = dbf.newDocumentBuilder()

Parsing

Parsing XML

Via les classes DocumentBuilderFactory, DocumentBuilder et Document :

- récupérer une instance de DocumentBuilderFactory : dbf = DocumentBuilderFactory.newInstance() (statique)
- l'utiliser pour construire un DocumentBuilder : db = dbf.newDocumentBuilder()
- utiliser le builder pour parser le XML : doc = db.parse(String | InputStream | File)

Parsing

Navigation dans le XML

Via les interfaces Element, Node et NodeList

Parsing

Navigation dans le XML

Via les interfaces Element, Node et NodeList

- un Document est vu comme le Node racine

Parsing

Navigation dans le XML

Via les interfaces Element, Node et NodeList

- un Document est vu comme le Node racine
- Node.getChildNodes() : liste des fils du noeud

Parsing

Navigation dans le XML

Via les interfaces Element, Node et NodeList

- un Document est vu comme le Node racine
- Node.getChildNodes() : liste des fils du noeud
- Node.getElementsByTagName(tag) : liste des éléments nommés “nom” décendants du noeud

Parsing

Navigation dans le XML

Via les interfaces Element, Node et NodeList

- un Document est vu comme le Node racine
- Node.getChildNodes() : liste des fils du noeud
- Node.getElementsByTagName(tag) : liste des éléments nommés “nom” décendants du noeud
- Node.getElementById(id) : récupère l’élément d’attribut id correspondant

Parsing

Navigation dans le XML

Via les interfaces Element, Node et NodeList

- un Document est vu comme le Node racine
- Node.getChildNodes() : liste des fils du noeud
- Node.getElementsByTagName(tag) : liste des éléments nommés “nom” décendants du noeud
- Node.getElementById(id) : récupère l’élément d’attribut id correspondant
- Element.getAttribute(nom) : récupère un attribut du noeud (sous forme de String)

Parsing

Navigation dans le XML

Via les interfaces Element, Node et NodeList

- un Document est vu comme le Node racine
- Node.getChildNodes() : liste des fils du noeud
- Node.getElementsByTagName(tag) : liste des éléments nommés “nom” décendants du noeud
- Node.getElementById(id) : récupère l'élément d'attribut id correspondant
- Element.getAttribute(nom) : récupère un attribut du noeud (sous forme de String)
- Node.getTextContent() : récupère le texte à l'intérieur du noeud.

Section 5

Messages broadcast - Broadcast Receivers

Broadcast receiver

Généralités

Système de passage de messages entre composants applicatifs.

Broadcast receiver

Généralités

Système de passage de messages entre composants applicatifs.

- Fonctionnement de type publication / abonnements

Broadcast receiver

Généralités

Système de passage de messages entre composants applicatifs.

- Fonctionnement de type publication / abonnements
- Les messages passés sont des Intent

Broadcast receiver

Généralités

Système de passage de messages entre composants applicatifs.

- Fonctionnement de type publication / abonnements
- Les messages passés sont des Intent
- Utilisation : notification d'événements divers

Broadcast receiver

Généralités

Système de passage de messages entre composants applicatifs.

- Fonctionnement de type publication / abonnements
- Les messages passés sont des Intent
- Utilisation : notification d'événements divers
- Broadcasts implicites ou explicites

Broascasts - Réception

Statique

Broascasts - Réception

Statique

- déclarer le receiver dans `AndroidManifest.xml` (+ intent filters)

Broascasts - Réception

Statique

- déclarer le receiver dans `AndroidManifest.xml` (+ intent filters)
- hériter de `BroadcastReceiver` et implémenter `onReceive()`

Broascasts - Réception

Statique

- déclarer le receiver dans `AndroidManifest.xml` (+ intent filters)
- hériter de `BroadcastReceiver` et implémenter `onReceive()`

Dynamique (via Context)

Receiver implicite uniquement

Broascasts - Réception

Statique

- déclarer le receiver dans `AndroidManifest.xml` (+ intent filters)
- hériter de `BroadcastReceiver` et implémenter `onReceive()`

Dynamique (via Context)

Receiver implicite uniquement

- hériter de `BroadcastReceiver` et implémenter `onReceive()`

Broascasts - Réception

Statique

- déclarer le receiver dans `AndroidManifest.xml` (+ intent filters)
- hériter de `BroadcastReceiver` et implémenter `onReceive()`

Dynamique (via Context)

Receiver implicite uniquement

- hériter de `BroadcastReceiver` et implémenter `onReceive()`
- créer une instance du receiver

Broascasts - Réception

Statique

- déclarer le receiver dans `AndroidManifest.xml` (+ intent filters)
- hériter de `BroadcastReceiver` et implémenter `onReceive()`

Dynamique (via Context)

Receiver implicite uniquement

- hériter de `BroadcastReceiver` et implémenter `onReceive()`
- créer une instance du receiver
- enregistrer le receiver via `registerReceiver(BroadcastReceiver, IntentFilter)`

Broascasts - Réception

Statique

- déclarer le receiver dans `AndroidManifest.xml` (+ intent filters)
- hériter de `BroadcastReceiver` et implémenter `onReceive()`

Dynamique (via Context)

Receiver implicite uniquement

- hériter de `BroadcastReceiver` et implémenter `onReceive()`
- créer une instance du receiver
- enregistrer le receiver via `registerReceiver(BroadcastReceiver, IntentFilter)`
- éventuellement, le supprimer via `unregisterReceiver(BroadcastReceiver)`

Broascasts - Envoi

Envoi

Broascasts - Envoi

Envoi

- créer un Intent à destination du BroadcastReceiver

Broascasts - Envoi

Envoi

- créer un Intent à destination du BroadcastReceiver
  - explicitement : `Intent(this,MyBroadcast.class)`

Broascasts - Envoi

Envoi

- créer un Intent à destination du BroadcastReceiver
  - explicitement : `Intent(this,MyBroadcast.class)`
  - implicitement : `intent.setAction(...)`
- envoyer cet intent via :
  - `sendBroadcast()`

Broascasts - Envoi

Envoi

- créer un Intent à destination du BroadcastReceiver
  - explicitement : `Intent(this,MyBroadcast.class)`
  - implicitement : `intent.setAction(...)`
- envoyer cet intent via :
  - `sendBroadcast()`
  - `sendOrderedBroadcast()`

Broascasts - Envoi

Envoi

- créer un Intent à destination du BroadcastReceiver
  - explicitement : `Intent(this,MyBroadcast.class)`
  - implicitement : `intent.setAction(...)`
- envoyer cet intent via :
  - `sendBroadcast()`
  - `sendOrderedBroadcast()`
  - ~~`sendStickyBroadcast()`~~ Deprecated

Broadcasts

OrderedBroadcast

Broadcasts

OrderedBroadcast

- les receivers sont exécutés par ordre de priorité

Broadcasts

OrderedBroadcast

- les receivers sont exécutés par ordre de priorité
- un receiver peut communiquer le résultat de son exécution au suivant
  - `getResultCode()`, `getResultData()`, `getResultExtras()`

Broadcasts

OrderedBroadcast

- les receivers sont exécutés par ordre de priorité
- un receiver peut communiquer le résultat de son exécution au suivant
  - `getResultCode()`, `getResultData()`, `getResultExtras()`
  - `setResult(code,data,extras)`
- l'expéditeur peut enregistrer un receiver spécial pour obtenir le dernier résultat

Broadcasts - Permissions

Sur le Receiver

Broadcasts - Permissions

Sur le Receiver

- `registerReceiver(receiver,filter,String)`

Broadcasts - Permissions

Sur le Receiver

- `registerReceiver(receiver,filter,String)`
- dans le manifest : attribut `android:permission`

Broadcasts - Permissions

Sur le Receiver

- `registerReceiver(receiver,filter,String)`
- dans le manifest : attribut `android:permission`
- L'émetteur doit avoir obtenu cette permission

Broadcasts - Permissions

Sur le Receiver

- `registerReceiver(receiver,filter,String)`
- dans le manifest : attribut `android:permission`
- L'émetteur doit avoir obtenu cette permission

Sur le message

- `sendBroadcast(Intent,String)`

Broadcasts - Permissions

Sur le Receiver

- `registerReceiver(receiver,filter,String)`
- dans le manifest : attribut `android:permission`
- L'émetteur doit avoir obtenu cette permission

Sur le message

- `sendBroadcast(Intent,String)`
- Le receiver doit avoir obtenu cette permission (sinon, le message ne lui parvient pas)

Broadcasts

Restrictions

Broadcasts

Restrictions

- Andorid 8 : Receivers statiques implicites interdits (sauf pour une liste blanches d'intent filters)

Broadcasts

Restrictions

- Andorid 8 : Receivers statiques implicites interdits (sauf pour une liste blanches d'intent filters)
- Selon les versions d'Android : infos plus disponibles (e.g. changement de réseau Wifi)

Broadcasts

Restrictions

- Andorid 8 : Receivers statiques implicites interdits (sauf pour une liste blanches d'intent filters)
- Selon les versions d'Android : infos plus disponibles (e.g. changement de réseau Wifi)
- Timeout de 10 secondes sur l'exécution de `onReceive()`

Section 6

Services

Services

Généralités

Services

Généralités

- Composant applicatif d'arrière-plan

Services

Généralités

- Composant applicatif d'arrière-plan
- Durée de vie arbitraire

Services

Généralités

- Composant applicatif d'arrière-plan
- Durée de vie arbitraire
- Local ou inter-app

Services

Généralités

- Composant applicatif d'arrière-plan
- Durée de vie arbitraire
- Local ou inter-app

Cas d'utilisation

- Téléchargement long
- Musique de fond
- etc.

Services - modes

Unbounded

Services - modes

Unbounded

- démarré via `startService()`

Services - modes

Unbounded

- démarré via `startService()`
- stoppé via `stopService()` (ou par lui-même)

Services - modes

Unbounded

- démarré via `startService()`
- stoppé via `stopService()` (ou par lui-même)
- pas de communication avec l'appelant

Services - modes

Unbounded

- démarré via `startService()`
- stoppé via `stopService()` (ou par lui-même)
- pas de communication avec l'appelant

Bounded

- établir la connexion via `bindService()`

Services - modes

Unbounded

- démarré via `startService()`
- stoppé via `stopService()` (ou par lui-même)
- pas de communication avec l'appelant

Bounded

- établir la connexion via `bindService()`
  - plusieurs composants peuvent s'y connecter

Services - modes

Unbounded

- démarré via `startService()`
- stoppé via `stopService()` (ou par lui-même)
- pas de communication avec l'appelant

Bounded

- établir la connexion via `bindService()`
  - plusieurs composants peuvent s'y connecter
  - si le service était arrêté, il est démarré

Services - modes

Unbounded

- démarré via `startService()`
- stoppé via `stopService()` (ou par lui-même)
- pas de communication avec l'appelant

Bounded

- établir la connexion via `bindService()`
  - plusieurs composants peuvent s'y connecter
  - si le service était arrêté, il est démarré
- Fermer la connexion via `unbindService()`

Services - modes

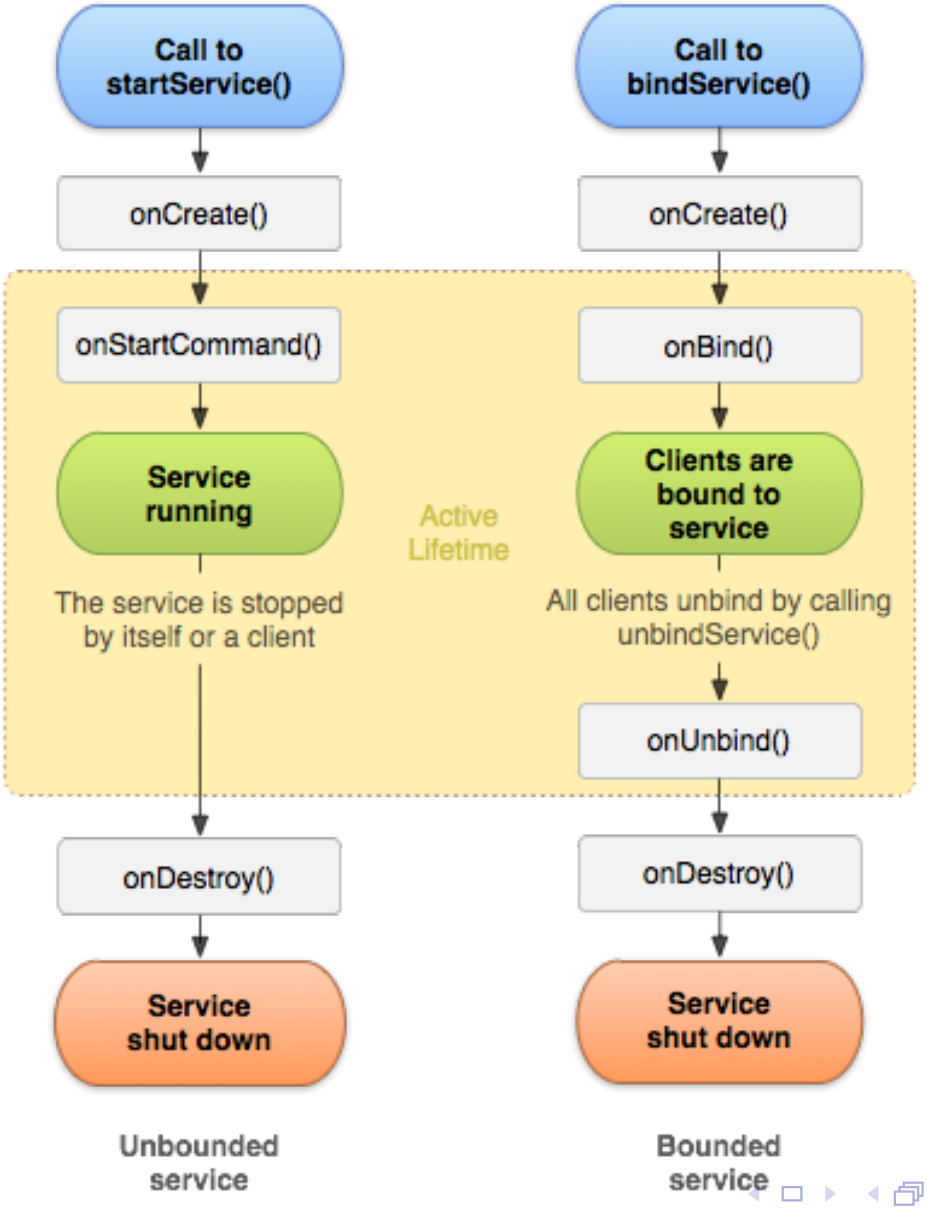
Unbounded

- démarré via `startService()`
- stoppé via `stopService()` (ou par lui-même)
- pas de communication avec l'appelant

Bounded

- établir la connexion via `bindService()`
  - plusieurs composants peuvent s'y connecter
  - si le service était arrêté, il est démarré
- Fermer la connexion via `unbindService()`
- service arrêté à la fermeture de la dernière connexion

Services - cycle de vie



Services - restrictions

Depuis Android 9

Pas de service en arrière-plan si l'app n'est pas en avant-plan

Services - restrictions

Depuis Android 9

Pas de service en arrière-plan si l'app n'est pas en avant-plan

- Démarrer un service provoque une exception

Services - restrictions

Depuis Android 9

Pas de service en arrière-plan si l'app n'est pas en avant-plan

- Démarrer un service provoque une exception
- tous les services d'arrière-plan sont arrêtés quand l'app passe en arrière-plan

Services - restrictions

Depuis Android 9

Pas de service en arrière-plan si l'app n'est pas en avant-plan

- Démarrer un service provoque une exception
- tous les services d'arrière-plan sont arrêtés quand l'app passe en arrière-plan

Foreground Service

Service visible par l'utilisateur (via une notification)

Services - restrictions

Depuis Android 9

Pas de service en arrière-plan si l'app n'est pas en avant-plan

- Démarrer un service provoque une exception
- tous les services d'arrière-plan sont arrêtés quand l'app passe en arrière-plan

Foreground Service

Service visible par l'utilisateur (via une notification)

- Démarrer le service via `startForegroundService()`

Services - restrictions

Depuis Android 9

Pas de service en arrière-plan si l'app n'est pas en avant-plan

- Démarrer un service provoque une exception
- tous les services d'arrière-plan sont arrêtés quand l'app passe en arrière-plan

Foreground Service

Service visible par l'utilisateur (via une notification)

- Démarrer le service via `startForegroundService()`
- appeler `startForeground(int,Notification)`

Services - restrictions

Depuis Android 9

Pas de service en arrière-plan si l'app n'est pas en avant-plan

- Démarrer un service provoque une exception
- tous les services d'arrière-plan sont arrêtés quand l'app passe en arrière-plan

Foreground Service

Service visible par l'utilisateur (via une notification)

- Démarrer le service via `startForegroundService()`
- appeler `startForeground(int,Notification)`
- (sinon, l'app est signalée ANR)

Section 7

Content Providers

Content Provider

Généralités

Interface d'accès à des données structurées à d'autres applications.

Content Provider

Généralités

Interface d'accès à des données structurées à d'autres applications.

- Accès "à la SQL" (CRUD)

Content Provider

Généralités

Interface d'accès à des données structurées à d'autres applications.

- Accès "à la SQL" (CRUD)
- Curseur sur une liste d'entrées

Content Provider

Généralités

Interface d'accès à des données structurées à d'autres applications.

- Accès "à la SQL" (CRUD)
- Curseur sur une liste d'entrées
- Chaque entrée est un ensemble de couples clés-valeurs

Content Provider

Généralités

Interface d'accès à des données structurées à d'autres applications.

- Accès "à la SQL" (CRUD)
- Curseur sur une liste d'entrées
- Chaque entrée est un ensemble de couples clés-valeurs
- clés identiques pour chaque enregistrement

Content Provider - Utilisation

Identification

Identification du provider (et des données) via des URI

Content Provider - Utilisation

Identification

Identification du provider (et des données) via des URI

Généralement :

- `content://nom.du.package/leProvider` pour le provider
- `content://nom.du.package/leProvider/42` pour une donnée

Content Provider - Utilisation

Requête

Content Provider - Utilisation

Requête

Via un `ContentResolver`

Content Provider - Utilisation

Requête

Via un `ContentResolver`

- récupérer le resolver via `Context.getContentResolver()`

Content Provider - Utilisation

Requête

Via un `ContentResolver`

- récupérer le resolver via `Context.getContentResolver()`
- méthode `query(`

Content Provider - Utilisation

Requête

Via un `ContentResolver`

- récupérer le resolver via `Context.getContentResolver()`
- méthode `query(`
  - `uri,`

Content Provider - Utilisation

Requête

Via un `ContentResolver`

- récupérer le resolver via `Context.getContentResolver()`
- méthode `query(`
  - `uri,`
  - `projection,`

Content Provider - Utilisation

Requête

Via un `ContentResolver`

- récupérer le resolver via `Context.getContentResolver()`
- méthode `query(`
  - `uri,`
  - `projection,`
  - `selection,`

Content Provider - Utilisation

Requête

Via un `ContentResolver`

- récupérer le resolver via `Context.getContentResolver()`
- méthode `query(`
  - `uri,`
  - `projection,`
  - `selection,`
  - `selectionArgs,`

Content Provider - Utilisation

Requête

Via un `ContentResolver`

- récupérer le resolver via `Context.getContentResolver()`
- méthode `query(`
  - `uri,`
  - `projection,`
  - `selection,`
  - `selectionArgs,`
  - `sortOrder)`

Content Provider - Utilisation

Requête

Via un `ContentResolver`

- récupérer le resolver via `Context.getContentResolver()`
- méthode `query(`
  - `uri,`
  - `projection,`
  - `selection,`
  - `selectionArgs,`
  - `sortOrder)`

Lecture des données

Via `Cursor` retourné par `query(...))`

Content Provider - Utilisation

Insertion

Content Provider - Utilisation

Insertion

- créer un `ContentValues`
- via le ContentResolver : Méthode `insert(`

Content Provider - Utilisation

Insertion

- créer un `ContentValues`
- via le ContentResolver : Méthode `insert(`
  - `uri,`
  - `values)`

Content Provider - Utilisation

Mise à jour

- créer un `ContentValues`
- via le ContentResolver : Méthode `update(`

Content Provider - Utilisation

Mise à jour

- créer un `ContentValues`
- via le ContentResolver : Méthode `update(`
  - `uri,`
  - `values,`
  - `selection,`
  - `selectionArgs)`