P82 Mobile Application Development - Android

Data Persistence

Storing Data 1

- Shared Preferences
  - Use the SharedPreferences class, retrieve key-value pairs of primitive type (Boolean, Integer, String, Float, Long)
  - Access Level
    - Private to the application
    - Shared with other read-only apps
    - Shared with other read / write app
  - There is a framework to generate a selection of preferences (depreciate)

Storing Data 2

- Using files
  - In internal memory: use FileInput / OutputStream with 3 levels of security
    - Private
    - Shared read only
    - Shared read / write
  - In external memory
    - Depends on the device
    - Attention no guarantee of presence of the data (can remove the card SD)

Storing Data 3

Databases

- Databases for Android are provided using SQLite.
- Compact DBMS

SharedPreferences

- Retain options and user preferences
- 3 Ways to recover them:
  - SharedPreferences preferences = PreferenceManager.getDefaultSharedPreferences (Context); Data private to the activity
  - getPreferences (int mode). Or mode and level of access (Context.MODE_PRIVATE, MODE_WORLD_READABLE, MODE_WORLD_WRITEABLE)
  - getSharedPreferences (String name, int mode) where name is the name of the file => several preference files

SharedPreference.Editor

- To add or modify preferences we need an Editor

SharedPreferences preferences = PreferenceManager.getDefaultSharedPreferences(ctx);

SharedPreferences.Editor editor = preferences.edit();

- Addition or modification:
  - Use method corresponding to the type
  - int maValeur = 22;
    editor.putInt("clef", myValeur);

  editor.commit() (do not forget)

- Getting data
  - int returnValue = preferences.getInt("key",0);
- Removing Data
  - editor.remove("key");
  - editor.clear();

SharedPreferences frameWork

- Depreciate
- http://openclassrooms.com/courses/creez-de
- Advocated Method
- http://developer.android.com/reference/andr

Using files: internally

- FileInputStream to read a file
  - openFileInput (String name).
    - inPut= openFileInput («myFile.txt»);

```
inPut.read();
inPut.close();
```

- FileOutputStream to write to a file
  - openFileOutput (String name, int mode) To get the file
    - Name
    - Access level mode : MODE_PRIVATE, MODE_WORLD_READABLE, MODE_WORLD_WRITABLE, MODE_APPEND (écrire à la fin d un fichier)
    - output = openFileOutput(PRENOM, MODE_PRIVATE);

```
output.write(userName.getBytes());
if(output != null)
  output.close();
```

Using files: internally

- Useful Methods
- getFilesDir() Get the path of the backup file.
- getDir() Create or Open a directory in the internal space
- deleteFile() Delete files
- fileList() Returns an Array[File] of your application

Using files: externally SD

- First you must declare it in the manifesto

```
<uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
```

- Be careful when doing tests... disconnect cable
- No certainty that the data is present
- Used to back up public data, accessed by other app or from the computer
- Exist preset directory
  - music we will put the files in / Music /
  - For downloads we will use / Download /
  - For the ringtones of telephone one will use / Ringtones /.

Useing files: externally

- CreateNewFile () to create a file if it does not exist
- mFile = new File (Environment.getExternalStorageDirectory (). getPath () + "/ Android / data /" + getPackageName () + "/ files /" + "myFile.txt");