Programmation de composant mobiles - groupe 2 : 10h45-12h45

9 janvier 2019

Durée : 2 heures

Documents et matériel autorisés :

- tous les documents papier sauf livres,
- documentation android `https://developer.android.com/index.html`,
- documentation java `https://docs.oracle.com/javase/7/docs/api/index.html`
- tous les documents sur moodle du cours,
- vos propres programmes de TP et votre projet,
- vos appareils android pour tester vos programmes.

Il est strictement **interdit** d'utiliser d'autres adresse WWW que mentionnées dessus, en particulier :

- il est interdit d'utiliser google ou n'importe quel autre moteur de recherche (vous devez aller directement sur la page de la documentation d'android et faire les recherches dans la documentation depuis le site d'android , sans passer par google),
- d'utiliser les forums de développeurs,
- utiliser l'ordinateur, tablettes, smartphones à d'autres fins que le développement et le test de vos programmes.

Le sujet comporte 3 pages.

Consignes

Pour chaque problème vous devez compresser le répertoire **VotreAppli/app/src/main** qui contient les sources de votre application :

```
tar cvf VotreNom_VotrePrenom_Exoi.tar  VotreAppli/app/src/main
gzip VotreNom_VotrePrenomExoi.tar
```

où *i* sera remplacé par le numéro du problème et `VotreNom` par votre nom.
Déposez le fichier tar.gz qui contient la solution de l'exercice sur moodle (cours Android) dans le dépôt dans la section Examen.
J'accepte aussi les archives zip. **Par contre pas d'archives rar svp.**
Avant de passer à l'exercice suivant il est prudent de sauvegarder au moins la dernière version qui marche de problème sinon vous risquez de vous retrouver avec un programme où rien ne marche plus.

Avant de commencer les exercices récupérez sur moodle (section Examen) les fichiers layout pour toutes les activités et le fichier *.java.
Vous pouvez modifier à votre guise tous les fichiers layout qu'on vous donne (ou même fabriquer vos propres layout).

Problème 1

Exercice 1 :

Implémenter une application avec une seule activité qui affiche deux `RadioButtons`, un bouton et un `TextView`. Initialement le `TextView` affiche `Hallo World"`.
Quand on l'utilisateur clique sur le le bouton `Afficher` le texte affiché dans `TextView` doit être remplacé par le texte du `RadioButton` qui est actuellement sélectionné.

Éviter de recopier en dur dans le programme le texte qui est dans le RadioButton, vous devez récupérer ce texte en utilisant la référence vers le RadioButton et non pas copier « en dur » la chaîne de caractères depuis le fichier xml vers le programme java.

Exercice 2 :

Quand on tourne l'appareil alors le texte « Hello World » revient dans le TextView. Faites en sorte que ce qui est affiché dans TextView ne change pas quand on tourne l'appareil.

Problème 2 - Quiz d'histoirique

Le but de l'application à construire dans cette partie est d'implémenter un quiz d'histoire (non pas que cela soit très intéressant mais plutôt pour entraîner sa mémoire). La bases de données contient une seule table avec 4 colonnes : une colonne texte qui décrit un événement historique (comme par exemple « chute de Rome ») et trois colonnes pour donner la date de l'événement (la date découpée en trois entiers : l'année, le mois et le jours, c'est pour simplifier le traitement et éviter les complications liées aux dates en java).
La classe `BD.java` dérivée de `SQLiteOpenHelper` est fournie. En étudiant cette classe vous trouverez les détails de la définition de la table. La classe `BD.java` n'aura pas besoin d'être modifié.
Je vous fournis aussi un `ContentProvider` approprié : `MyContentProvider`, tout prêt à être utilisé avec la BD. A priori il n'y aura rien à modifier dans `MyContentProvider` sauf sans doute le changement de `AUTHORITY` qui doit correspondre à votre authority de ContentProvider et changement de la version de la bases de données (vous pouvez commencer avec la version 1). Pour de raison de commodité aussi bien AUTHORITY que VERSION sont données comme des constantes `static final` au début de `MyContentProvider`. Donc les seules modification à apporter dans cette classe concernent ces deux constantes.
A part de la BD et de ContentProvider l'application est composée de trois activités :

(1) `SelectionActivite` - c'est l'activité de démarrage qui sert à sélectionner une des deux autres activités et faire initialiser la base de données,
(2) l'activité `AjouterEvenement` qui permet d'ajouter un nouveau événèment dans la bases de de données.
(3) l'activité `ChosirEvenement` qui implémente le quiz.

Les fichiers layout pour toutes les trois activités sont fournis.

Très important concernant la BD :

Dans tous les exercices qui nécessitent l'accès à la base de données vous devez le faire par `ContentResover`, donc grâce au `MyContentProvider` fourni. Si le `ContentResolver/ContentProvider` ne sont pas utilisés cela signifie que vous ne savez pas vous en servir et vous ne savez pas fabriquer un Uri correct (dont le contenu vous devez être capable de trouver en analysant `MyContentProvider`). En conséquence l'accès à la base de donnée qui utilise directement la référence vers `SQLOpenHelper` est prohibé (ou plus exactement baisse **substantiellement** la note).
Par contre **on ne vous demande pas** d'utiliser un `CursorLoader`, vous pouvez (et devez) utiliser directement `query` de `ContentResolver` pour lancer les requêtes.

Exercice 3 :

Implémenter l'activité `SelectionActivite` qui à l'aide de deux boutons permet de lancer soit `AjouterEvenement` ou `ChoisirEvenement`.

Exercice 4 :

Implementer la classe `AjouterEvenement` qui permet d'ajouter un nouveau événement dans la base de données.

Maintenant c'est le tour de l'activité `ChoisirEvenement`.

Exercice 5 :

L'activité `ChoisirEvenement` est composée d'une `ListView`, de trois `TextView` et de bouton `Soumettre`. Le `ListView` doit être initialisé avec tous les événements de la base de donnée (juste les valeurs de la colonne EVENT).
Comme le fichier layout pour une ligne de `ListView` vous utiliserez

`simple_list_item_checked`

Ce layout fait partie de l'android donc son id est

`android.R.layout.simple_list_item_checked`

Vous le trouverez sur la page

`https://android.googlesource.com/platform/frameworks/base/+/master/core/res/res/layout`

En regardant la définition de ce layout on peut voir qu'il est composé d'un `CheckedTextView` dont identifiant est `android.R.id.text1`, c'est là où vous placerez les descriptions d'événements.

Exercice 6 :

On complète l'activité `ChoisirEvenement`. L'utilisateur choisit un événement sur la liste, remplit les `TextViews` avec la date et clique sur le bouton `Soumettre`. Un dialogue s'affiche pour informer si l'utilisateur a donnée une date correcte où non.
(Utilisez un `Toast` affichant « Bravo » si l'utilisateur a donné la date correcte de l'événement ou « Oops » sinon si vous ne savez pas faire un dialogue.)

Exercice 7 :

Utilisez `SharedPreferences` pour mémoriser la dernière question (identifiant de l'événement) pour laquelle l'utilisateur a donnée une réponse incorrecte. Chaque fois quand on retourne à l'activité `ChoisirEvenement` la `ListView` est initialisée avec la liste de tous les événement mais en plus l'événement mémorisé dans `SharedPreferences` est sélectionné par défaut. (Intuitivement, on propose à l'utilisateur de tester ses connaissances de dates sur l'événement pour lequel sa mémoire s'est avérée défaillante).
