HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

Laboratoire 1 : Introduction à Android

Systèmes mobiles

Auteur : Spinelli Isaia et Simonet Yoann
Prof : Dutoit Fabien
Assist. : Christophe Greppin
Date : 20.09.2019
Classe : SYM-A

Table des matières

Introduction

Ce laboratoire est constitué de plusieurs manipulations destinées à implémenter une application élémentaire sur un émulateur et/ou sur un smartphone Android afin de nous familiariser avec l'environnement de développement Android.

Réponse aux questions

1.Comment organiser les textes pour obtenir une application multi-langues (français, allemand, italien, langue par défaut: anglais)?Que se passe-t-il si une traduction est manquante dans la langue par défaut ou dans une langue supplémentaire?

Il ne faut pas écrire les strings en dure dans le code. Il est préférable d'utiliser des références sur un fichier xml.

Values est le dossier par défaut, mais il existe des conventions pour faire un fichier par langue. Par exemple, values-fr pour toutes les valeurs françaises.

Si la basile XML existe mais qu'elle est vide, ça n'affiche rien. Si elle n'existe pas, le programme lève une exception. Il en va de soit s'il le dossier de la langue de traduction manque.

2.Dans l'exemple fourni, sur le dialogue pop-up, nous affichons l'icône android.R.drawable.ic_dialog_alert, disponible dans le SDK Android mais qui n'est pas très bien adapté visuellement à notre utilisation. Nous souhaitons la remplacer avec notre propre icône, veuillez indiquer comment procéder. Dans quel(s) dossier(s) devons-nous ajouter cette image ? Décrivez brièvement la logique derrière la gestion des ressources de type «image» sur Android.Info: Google met à disposition des icônes open source dans le style «Material Design» utilisé actuellement sur Android:https://material.io/resources/icons/

On doit aller ajouter l'image qui se trouve dans res/mipmap-xxxx/ (xxxx en fonction de la taille de l'icone) .

Les image que l'on met dans le dossier res sont directement charger dans l'exécutable de l'application, pour pouvoir y accéder il suffit de mettre leur chemin da res exemple : *R.mipmap.error_log*

3.Lorsque le login est réussi, vous êtes censé chaîner une autre Activity en utilisant un Intent. Si je presse le bouton "Back" de l'interface Android, que puis-je constater ? Comment faire pour que l'application se comporte de manière plus logique ? Veuillez discuter de la logique derrière les activités Android.

Le mot de passe ainsi que l'email sont toujours dans les champs de saisie.

Il faudrait aller clear les champs dans la fonction onRestart().

On remarque que comme l'activité n'est pas détruite, elle conserve son état. (voir question 8).

4.On pourrait imaginer une situation où cette seconde Activity fournit un résultat (par exemple l'IMEI ou une autre chaîne de caractères) que nous voudrions récupérer dans l'Activity de départ. Comment procéder ?

Dans la seconde Activity, il faut aller set le résultat avec la fonction setResult.
ex :Activity2.this.setResult();

Finalement, il faut mettre en place un callback dans la première activité ex :

```
1  public class MainActivity extends Activity {
2      @Override
3      protected void onActivityResult(int requestCode, int resultCode, Int
4
5        if( resultCode==1 ) {
6          String s = data.getStringExtra(EXTRA_MESSAGE);
7          Toast.makeText(this, s, Toast.LENGTH_SHORT).show();
8        }
9
10       super.onActivityResult(requestCode, resultCode, data);
11     }
12     //...
13 }
```

5.Vous noterez que la méthode getDeviceId() du TelephonyManager, permettant d'obtenir l'IMEI du téléphone, est dépréciée depuis la version 26 de l'API. Veuillez discuter de ce que cela implique lors du développement et de présenter une façon d'en tenir compte avec un exemple de code.

Il ne faudra en tenir compte pour la rétrocompatibilité sur les anciennes versions !

Afin d'en tenir compte, il est possible de gérer le problème comme ceci :

```
@SuppressWarnings("deprecation")
private String getIMEINumber() {
    String IMEINumber = "";
    if (ActivityCompat.checkSelfPermission(this, android.Manifest.permission.READ_PHONE_STATE) =
        TelephonyManager telephonyMgr = (TelephonyManager) getSystemService(TELEPHONY_SERVICE);
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
            IMEINumber = telephonyMgr.getImei();
        } else {
            IMEINumber = telephonyMgr.getDeviceId();
        }
    }
    return IMEINumber;
}
```

6.Dans l'activité de login, en plaçant le téléphone (ou l'émulateur) en mode paysage(landscape), nous constatons que les 2 champs de saisie ainsi que le bouton s'étendent sur toute la largeur de l'écran. Veuillez réaliser un layout spécifique au mode paysage qui permet un affichage mieux adapté et indiquer comment faire pour qu'il soit utilisé automatiquement à l'exécution.

Premièrement, il faut créer un nouveaux layout avec le même nom que celui du portrais. Ensuite, on le place dans le dossier layout-land des ressources.

Finalement, le layout se change automatiquement lorsqu'on passe en mode paysage.

7.Le layout de l'interface utilisateur de l'activité de login qui vous a été fourni a été réalisé avec un LinearLayout à la racine. Nous vous demandons de réaliser un layout équivalent utilisant cette fois-ci un RelativeLayout.

Nous avons refait le layout en mode relativeLayout comme demandé. (voir authent.xml). On peut constater que le relativeLayout permet de disposer librement les composants dans le layout tous en donner des distances relatives au bord du layout ou à d'autre composant ce qui est bien pratique.

8.Implémentezdans votre code les méthodes onCreate(), onStart(), onResume(), onPause(), onStop(), etc... qui marquent le cycle de vie d'une application Android, et tracez leur exécutiondans le logcat. Décrivez brièvement à quelles occasions ces méthodes sont invoquées. Vous expliquerez aussi l'enchainement de ces appels lorsque l'on passe d'une activité à l'autre. Comment pouvez-vous factoriser votre code pour éviter de devoir réimplémenter ces méthodes dans chacune de vos activités ?

Pour commencer, les méthodes onCreate() -> onStart() -> onResume() sont appelées.

```
2019-10-04 08:43:17.027 15006-15006/? W/vd.sym.templat: Accessing hidden method Landroid/view/ViewGroup;->makeOptionalFitsSystemWindows
2019-10-04 08:43:17.094 15006-15006/? W/MainActivity: START !!
2019-10-04 08:43:17.097 15006-15006/? W/MainActivity: RESUME !!
2019-10-04 08:43:17.177 15006-15036/? W/OpenGLRenderer: Failed to choose config with EGL_SWAP_BEHAVIOR_PRESERVED, retrying without...
2019-10-04 08:43:17.369 15006-15036/? D/eglCodecCommon: setVertexArrayObject: set vao to 0 (0) 0 0
```

Ensuite, si on passe à une autre activité on passe en onPause() - > onStop().

```
2019-10-04 10:45:41.283 1528-1528/ch.heigvd.sym.template W/MainActivity: PAUSE !!
```

```
2019-10-04 10:45:42.644 1528-1528/ch.heigvd.sym.template W/MainActivity: STOP !!
```

Si on revient dans mainActivity avec la flèche on retombe sur onDestroy->onCreat() ->onStart() ->onResume()

```
2019-10-04 10:48:31.040 1528-1528/ch.heigvd.sym.template W/MainActivity: DESTROY !!
```

```
2019-10-04 10:48:31.102 1528-1528/ch.heigvd.sym.template W/MainActivity: START !!
2019-10-04 10:48:31.105 1528-1528/ch.heigvd.sym.template W/MainActivity: RESUME !!
```

Du coup les états de l'activité sont perdus.

Par contre, si on revient avec le bouton android on passe par onRestart() -> onStart() ->onResume()

```
2019-10-04 10:51:12.639 1528-1528/ch.heigvd.sym.template W/MainActivity: RESTART !!
2019-10-04 10:51:12.640 1528-1528/ch.heigvd.sym.template W/MainActivity: START !!
2019-10-04 10:51:12.641 1528-1528/ch.heigvd.sym.template W/MainActivity: RESUME !!
```

De ce fait, les états de l'activité sont conservés.

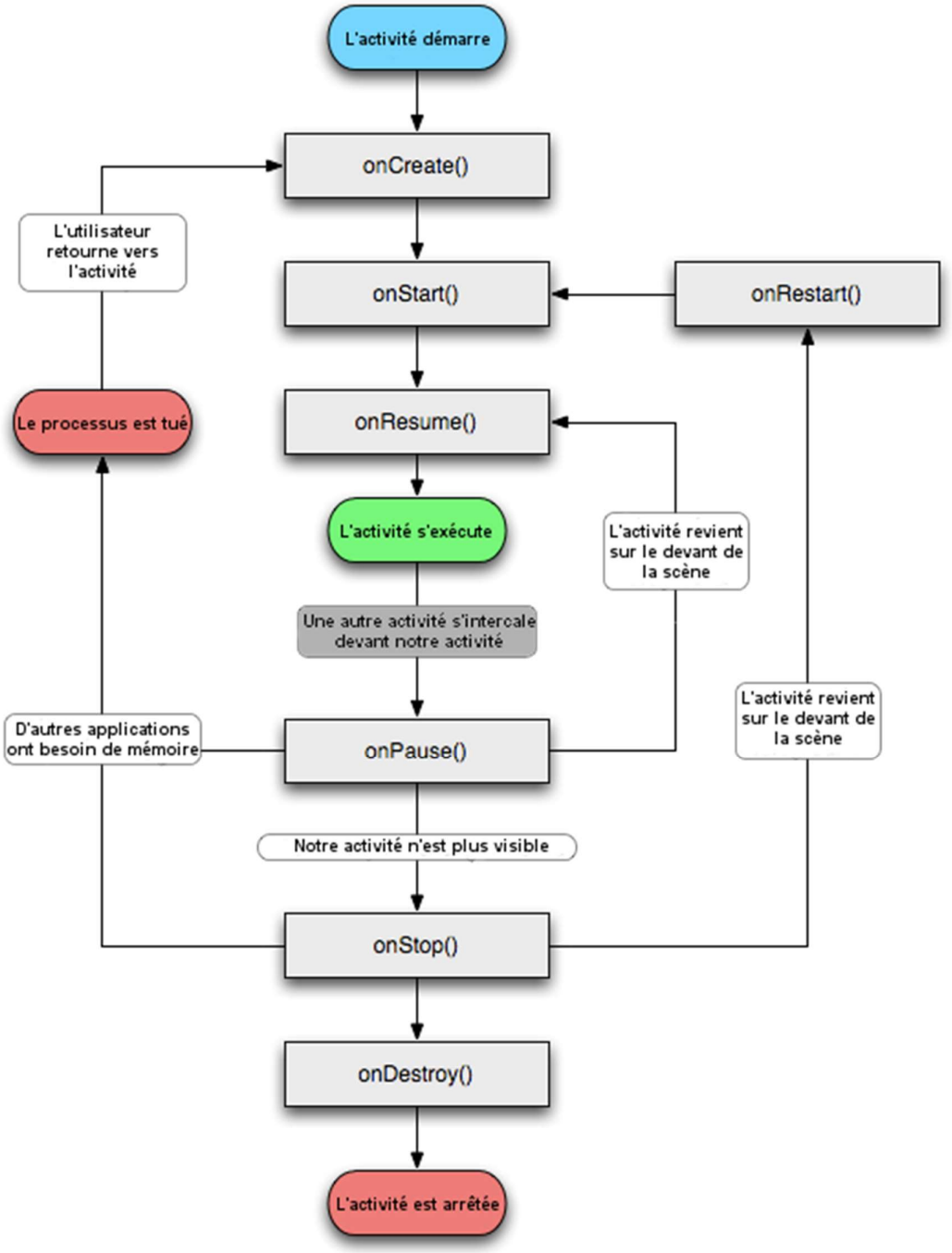
Si on verrouille le téléphone, l'activité se met en pause puis, restarte quand on le déverrouille.

```
2019-10-04 10:40:22.042 1528-1528/ch.heigvd.sym.template W/MainActivity: PAUSE !!
2019-10-04 10:40:22.077 1528-1528/ch.heigvd.sym.template W/MainActivity: STOP !!
2019-10-04 10:40:29.145 1528-1528/ch.heigvd.sym.template V/InputMethodManager: Starting
2019-10-04 10:40:29.155 1528-1528/ch.heigvd.sym.template I/InputMethodManager: [IMM] sta
2019-10-04 10:40:29.157 1528-1528/ch.heigvd.sym.template D/InputTransport: Input channel
2019-10-04 10:40:29.157 1528-1528/ch.heigvd.sym.template D/InputTransport: Input channel
2019-10-04 10:40:29.158 1528-1528/ch.heigvd.sym.template W/MainActivity: RESTART !!
2019-10-04 10:40:29.198 1528-1528/ch.heigvd.sym.template W/MainActivity: START !!
2019-10-04 10:40:29.200 1528-1528/ch.heigvd.sym.template W/IInputConnectionWrapper: fini
2019-10-04 10:40:29.201 1528-1528/ch.heigvd.sym.template W/MainActivity: RESUME !!
```

Finalement, quand on quitte l'application on passe par onPause() -> onStop() -> onDestroy()

```
2019-10-04 17:50:35.373 26369-26369/ch.heigvd.sym.template W/MainActivity: PAUSE !!
2019-10-04 17:50:35.446 26369-26369/ch.heigvd.sym.template W/MainActivity: STOP !!
2019-10-04 17:50:38.919 26369-26369/ch.heigvd.sym.template W/MainActivity: DESTROY !!
```

Si on veut avoir une vision d'ensemble, on peut se référer à ce schéma :



Pour éviter de devoir réimplémenter ses méthodes dans chaque activité on pourrait créer une super classe Trace qui hériterai de AppCompatActivity. Ensuite, on ferait que toute nos activités héritent de Trace.

9.Question Bonus facultative - S'il vous reste du temps, nous vous conseillons de le consacrer à mettre en place la résolution des permissions au runtime.

Afin d'avoir les permissions au runtime, nous avons décidé de demander au démarrage (OnStart) de l'application les permissions. S'il refuse, l'application se ferme automatiquement.

Nous avons décidé que s'il accepte les permissions, nous ne les redemanderons jamais. Cependant, s'il refuse, nous demanderons à chaque ouverture de l'application la permission qu'il a refusée.

Conclusion

Difficultés rencontrées

Lire la carte SD sur notre téléphone car la fonction ne renvoie pas le bon chemin.

```
Environment.getExternalStoragePublicDirectory(Environment.DIRECTORY_DOWNLOADS);
```

La résolution les permissions au runtime, il a fallu utiliser un callback.

Compétences acquises

Mise en place d'une application simple sur Android et familiarisation avec l'environnement et la documentation.

Résultats obtenus

Nous pensons avoir réussi complétement le laboratoire avec la question bonus.

Date : 05.10.19

Nom de l'étudiant : Spinelli Isaia et Simonet Yoann