TD4 - Les Activités

[Introduction to Activities](https://developer.android.com/guide/components/activities/intro-activities)

Le but de ce TP est de créer une application composée de plusieurs activités. Chacune est réduite à un titre et un ou deux boutons servant de prétexte pour étudier le lancement et la terminaison d’activités. Les écrans contiennent un ou plusieurs boutons permettant d’aller sur les autres écrans ou d’autres activités, selon un schéma de navigation précis.

### Spécifications

Voici d’abord la présentation générale du projet, les paragraphes suivants permettront d’y arriver. Il comportera trois activités. Chacune devra afficher un titre à son nom et des boutons permettant d’aller dans les autres activités selon les spécifications qui suivent.

**LoginActivity** C’est l’activité qui est lancée au démarrage de l’application. Dans une application complète, son but serait de saisir un identifiant et un mot de passe d’utilisateur.

* Elle doit contenir un bouton « login » envoyant sur l’activité NewsActivity.
* Faites en sorte que le bouton back(le bouton physique de votre téléphone) ne permette pas de revenir dans cette activité (la détruire). C’est à dire que si l’utilisateur va de LoginActivity à NewsActivity et qu’il appuie sur back, il revient directement sur l’écran d’accueil Android.

**NewsActivity** Son rôle est d’afficher une liste d’informations.

* Le bouton back ne permet pas de revenir dans une précédente activité du logiciel. Si on appuie sur back, ça retourne sur l’écran d’accueil Android.
* L’activité contient un bouton « details » qui envoie sur l’activité DetailsActivity.
* Elle contient aussi un bouton « logout » qui fait revenir sur l’activité LoginActivity

**DetailsActivity** Elle sert à afficher les détails de l’une des informations de la liste.

* Si on appuie sur back, on revient dans l’activité NewsActivity.
* Cette activité contient un bouton libellé « ok » qui ramène sur l’activité NewsActivity.

### Création des activités

Commencez par créer un nouveau projet appelé NewsList. Nommez sa première activité LoginActivity et son layout login.xml ou activity\_login.xml.

Dans la classe LoginActivity.java, ajoutez setTitle(getLocalClassName()); juste après setContentView(...).

Faite de même pour les deux autres Activities selon les spécifications ci-dessus.

Attention n’oubliez pas, soit vous utilisez l’interface d’Android Studio pour générer des nouvelles activités ( Je vous préconise cette solution) ou vous copier coller les fichiers java et XML. Si vous décidez de retenir le second choix, n’oubliez pas de les déclarer dans le fichier AndroidManifest.xml sinon vous obtiendrez un crash de l’application.

### Lancement d’une activité

Cela se fait en créant un Intent. Voici un exemple à adapter.

| val intent = Intent(this, AutreActivity::class.java)  startActivity(intent) |
| --- |

Utilisez maintenant ce code pour l’ajouter aux clics listeners des différents bouton (Aidez vous du TD3 pour implémenter les clics listeners des bouton)

### Terminaison d’une activité

Pour détruire une activité, vous pouvez utiliser la méthode finish().

| class MainActivity : AppCompatActivity() {  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  setContentView(R.layout.*activity\_main*)  finish();  }  } |
| --- |

Dans cet exemple, si vous lancez votre App, l’activité se fermera aussitôt puisque une fois finie d’être créée (onCreate) on lui indique de se finir..

Pour gérer votre navigation et le bouton back, vous devrez implémenter la méthode onBackPressed et ensuite faire votre traitement comme voulu dans les spécifications (cf page 1) pour chaque activité.

| override fun onBackPressed() {  super.onBackPressed()  //*TODO*  } |
| --- |

### Lancement d’une activité pour afficher une page Web

Rajoutez un bouton libellé about dans le layout news.xml et faites en sorte que cela exécute le code suivant quand on clique dessus :

| val url = "https://news.google.com/"  val intent = Intent(Intent.*ACTION\_VIEW*, Uri.parse(url))  startActivity(intent) |
| --- |

### Transmission d’informations entre activités

Maintenant, on va faire en sorte que LoginActivity communique une valeur à NewsActivity : le nom de l’utilisateur. Dans un premier temps, on va utiliser les intents, et dans un second temps, on définira un contexte d’application

1. Extras d’intents

Dans le layout login.xml, rajoutez un EditText permettant de saisir un nom de compte. Ce qu’il faut maintenant, c’est que lorsque l’utilisateur clique sur le bouton login, l’écouteur lise d’abord le nom saisi dans l’EditText et l’envoie à NewsActivity. Pour cela, il faut utiliser un extra :

| val intent = Intent(this, NewsActivity::class.java)  intent.putExtra("login", 'récupérer le texte saisi dans l EditText')  startActivity(intent) |
| --- |

De l’autre côté, dans NewsActivity, il faut récupérer cet extra et le placer dans une (nouvelle) variable membre de la classe. Attention à bien utiliser la bonne méthode en fonctione du Type passé dans le putExtra (ici un string)

| if (*intent*.hasExtra("login")) {  val login = *intent*.getStringExtra("login")  } |
| --- |

On récupère donc à l’aide de la bonne méthode getStringExtra puisque notre variable préalablement passé en paramètre était de type String.

Vérifiez que ça marche, rajoutez un TextView dans le layout news.xml et faites en sorte qu’il affiche le login saisi dans l’activité de connexion. Normalement, ce login ne doit pas être oublié lors de la navigation vers DetailsActivity. Il n’est perdu que si on quitte l’activité NewsActivity. Le problème est que ce login n’est pas disponible dans la troisième activité. Il faudrait la transmettre à chaque lancement dans un extra. Ce n’est pas demandé car il y a mieux pour cela, le contexte d’application.

1. Contexte d’application

Le contexte d’application est une classe héritant de Application qui permet de stocker une information globale de l’application. Android crée automatiquement un singleton de cette classe, c’est à dire une instance unique pour toute la vie de l’application. Elle est disponible dans n’importe quelle activité à l’aide de la méthode getApplicationContext(). C’est un peu comme une session en PHP. Mais, par défaut celui créé par Android ne contient rien, donc nous allons le surcharger.

Définissez une classe appelée NewsListApplication héritant de android.app.Application. Voici son début :

| import android.app.Application  class NewsListApplication : Application() {} |
| --- |

Rajoutez-lui une variable membre privée login de type String. Définissez une surcharge de la méthode onCreate pour initialiser cette variable

| var login: String? = null  override fun onCreate() {  super.onCreate()  this.login = null  } |
| --- |

gModifiez le fichier AndroidManifest.xml afin que votre application repose sur cette nouvelle classe. Il suffit de rajouter l’attribut android:name à l’élément ; ne vous trompez pas entre activités et application :

| <manifest xmlns:android="http://schemas.android.com/apk/res/android" ...> ... <application android:name=".NewsListApplication" ... |
| --- |

Ensuite, cette classe et sa variable seront disponibles dans toutes les activités.

| class MainActivity : AppCompatActivity() {  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  setContentView(R.layout.*activity\_main*)  val login = (*applicationContext* as NewsListApplication).login  }  } |
| --- |

Alors maintenant, faites en sorte que l’activité LoginActivity mette à jour cette information et que les deux autres l’utilisent : rajoutez l’affichage du login dans l’activité DetailsActivity (laissez l’activité NewsActivity utiliser les extras).