Travaux pratiques
Développement mobile

Patrick Albers

January 4, 2021

1 Découverte de l'environnement *Android Studio*

1. Lancez *Android Studio*
2. Créez un nouveau projet $MonPremierProjet$ pour une application *smartphone* et une activité *Basic Activity* (langage *Java*, version de l'API 21).
3. Compilez et exécutez l'application sur un terminal virtuel (AVD), puis sur un terminal Android (ADB).

2 Activité et intention

1. Créez un nouveau module $Projet2$ pour une application *smartphone* sans activité.
2. Ajoutez une activité vide (clic droit sur $projet2{>}New{>}Activity{>}Empty\ activity$).
3. Lancez l'application sur un terminal. Que se passe-t-il ? Que faire pour que cela fonctionne ?
4. Chaque activité possède une vue associée (à quelques exceptions près). Pour modifier cette vue, on peut modifier directement le fichier *Java*, mais il est plus commode de passer par l'assistant Android.
   (a) Depuis la vue *Android*, cliquez le fichier $res/activity\_main.xml$.
   (b) Ajoutez un texte ($TextView$) en cliquant droit sur *Add to design*.
   (c) Ajoutez un message de bienvenue sur votre écran principal.
   (d) Compilez et exécutez votre application.
5. On veut maintenant ajouter une seconde activité.
   (a) Ajoutez dans le fichier manifeste, une ligne pour l'activité : *SecondActivity*.
   (b) Créez une nouvelle classe *SecondActivity* dans votre application.
   (c) Créez le nouveau fichier *second_activity.xml* dans le répertoire *layout* (click droit : $New/Layout\ Resource\ File$)
   (d) Ajoutez un champ texte dans ce deuxième layout
   (e) Dans le fichier *Java*, connectez l'activité à la vue en ajoutant la méthode $onCreate()$
   (f) Ajoutez un bouton dans l'activité principale au milieu de la page (fichier *second_activity.xml*)
   (g) Ajoutez une variable *Button* et connectez la à la vue (méthode $findViewById()$), et ajouter un événement sur ce bouton à l'aide du code suivant :

```
Button monBouton = findViewById(R.id.monBouton);
monBouton.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v){
       startActivity(new Intent(MainActivity.this, SecondActivity.class));
    }
});
```

(h) Testez ce code.

6. On veut maintenant ajouter une intention implicite pour aller sur le site *https://www.esaip.org*.

   (a) Ajoutez un second bouton dans *second_activity.xml*

   (b) Ajoutez une variable *Button* comme précédemment et associez le clic à l'appel implicite d'un navigateur pour le site *https://www.esaip.org*.

   On utilisera l'intention vu en cours :

   ```
   new Intent(Intent.ACTION_VIEW, Uri.parse("https://www.esaip.org/"));
   ```

   (c) Testez le code

7. On voudrait tester le retour d'une activité[1] en utilisant la méthode :

   ```
   void startActivityForResult(Intent intent, int requestCode)
   ```

   (a) Créez une troisième activité avec les différents fichiers nécessaires.

   (b) Ajoutez un troisième bouton dans l'activité principale, qui appelera la troisième activité.

   (c) Dans la troisième activité, ajoutez un champ texte et ajoutez un bouton qui appelera la méthode :

   ```
   void setResult(int resultCode, Intent data)
   ```

   Utilisez la méthode *putExtra*() pour passer la valeur du champ texte.

   (d) Dans l'activité principale, surchargez la méthode :

   ```
   void onActivityResult(int requestCode, int resultCode, Intent data)
   ```

   Cette méthode sera appelée lors du retour dans cette activité[2].

   (e) Récupérez la valeur avec *getStringExtra*() et affichez la dans un message *Toast*.

   (f) Testez le code.

[1] https://developer.android.com/training/basics/intents/result

[2] La valeur du paramètre *resultCode* doit être le même dans les méthodes *onActivityResult*() et *setResult*(). C'est cette valeur qui permettra de reconnaître de quelle activité vient le retour, dans le cas où plusieurs activités seraient appelées ici dans l'activité principale. Si les deux valeurs *requestCode* des méthodes *onActivityResult*() et *startActivityForResult*() sont les mêmes, c'est que l'appel s'est bien passé.