TP
ANDROID - Prise en main

La première étape consiste à installer Android Studio sur vos machines. Suivez les instructions disponibles ici : `https://developer.android.com/sdk/index.html`

Exercice 1.1 (Conception d'interfaces - un premier exercice)

1. Créez un projet Android via `Start a new Android Studio Project`.
2. Sélectionnez une application Phone and Tablet avec l'API 15, avec une `Blank Activity` (pas de fragments pour l'instant).
3. Créez une IHM avec des composants graphiques permettant la saisie des éléments suivants, organisés verticalement :
   - nom, prenom
   - date de naissance (en utilisant les widgets fournis par Android)
   - ville de naissance
4. Ajoutez à cette interface un bouton avec le texte "Valider" qui prend toute la largeur de l'écran, et qui soit positionné en permanence en bas de l'écran (vérifiez que c'est bien le cas en modifiant la résolution et/ou l'orientation du terminal).
5. On désire maintenant executer une action particulière lorsque le bouton "Valider" est utilisé. Android vous propose deux modalités (cf `http://developer.android.com/reference/android/widget/Button.html`) :
   - soit vous spécifiez le nom de la méthode à appeler dans l'élément XML qui décrit votre boutton en utilisant l'attribut `android:onClick="votre_methode"`. Il suffit ensuite d'ajouter une méthode `public void votre_methode(View view)` à votre classe qui sera appelée dès que le bouton est sélectionné.
   - soit vous utilisez un listener `View.onClickListener` sur votre bouton. Pour cela, vous récupérez la `View` correspondant à votre bouton via la méthode `findViewById(R.id.votre_id)`. Il faut caster cette `View` en objet `Button`. Et ensuite, appeler la méthode `Button.setOnClickListener()` sur votre bouton. Dans le listener vous définissez la méthode `onClick()` qui sera appelée lorque le bouton est activé (cf Figure 1).
6. L'action sur le bouton valider doit consister à afficher à l'écran les données saisies par l'utilisateur à l'aide de la classe `Toast`. Celle-ci s'emploie de la façon suivante : `Toast.makeText(getApplicationContext(), "votre texte", Toast.LENGTH_SHORT).show();`. Pour récupérer les données saisies dans les champs, il faut accèder aux composants JAVA via la méthode `findViewById()` et `downcaster` les références obtenues dans les types réels des composants.
7. créez un menu en XML (dans le répertoire `res/menu` cf documentation Android) qui sera appelé par la touche menu du Smartphone et permettant :
   - l'appel d'une fonction de remise à zéro des données des champs de saisie l'IHM
   - l'appel d'une fonction d'ajout d'un composant graphique permettant la saisie d'un numéro de téléphone. Une façon simple d'ajouter un champ est de récupérer une référence sur un `Layout` (via son ID), et d'insérer directement un composant graphique que vous avez crée en JAVA (méthode `ViewGroup.addView(...)`).

   Pour afficher le menu, il est nécessaire d'implémenter la méthode `public boolean onCreateOptionsMenu(Menu menu){...}` de la classe `Activity`, et dans cette méthode, d'utiliser un MenuInflator pour convertir votre menu XML en objet Java `Menu`. Pour appeler les actions relatives aux items, il faut implémenter la méthode `public boolean onOptionsItemSelected(MenuItem item)` de la classe `Activity`

```java
public class MyActivity extends Activity {
    protected void onCreate(Bundle icicle) {
        super.onCreate(icicle);

        setContentView(R.layout.content_layout_id);

        final Button button = (Button) findViewById(R.id.button_id);
        button.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                // Perform action on click
            }
        });
    }
}
```

FIGURE 1 – Utilisation d'un listener pour executer une action sur un bouton.

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string-array name="planets_array">
        <item>Mercury</item>
        <item>Venus</item>
        <item>Earth</item>
        <item>Mars</item>
    </string-array>
</resources>
```

FIGURE 2 – Utilisation d'un string-array pour stocker les valeurs d'un composant Spinner.

Exercice 1.2 (Conception d'interfaces - Spinner & Intents)

1. Ajoutez à votre interface la possibilité de saisir le département de naissance. Utilisez un composant de type Spinner.
   - Pour pré-saisir en XML des valeurs dans le composant Spinner, vous pouvez définir l'attribut `android:entries` de votre Spinner : `android:entries="@array/departements"`.
   - La valeur de l'attribut spécifie que les valeurs du Spinner sont stockées dans un tableau nommé `departements` décrit par un fichier XML. Ce fichier XML doit être placé dans le répertoire `res/values` de votre projet et doit être formaté comme dans la figure 2.
2. Ajoutez une option au menu qui permet d'invoquer un browser web en utilisant le mécanisme des Intents implicites (communication entre les applications) pour afficher la page wikipedia correspondant à la ville de naissance de l'utilisateur. Pour utiliser un Intent implicite :
   - il faut spécifier l'action à réaliser et les données sur lesquelles les réaliser. Par exemple l'instruction `Intent intent = new Intent(Intent.ACTION_VIEW, Uri.parse("http://fr.wikipedia.org"));` spécifie une action de type "voir" sur l'adresse `"http://fr.wikipedia.org/"`.
   - ensuite, il faut lancer l'intent via la méthode `startActivity(intent);`

Exercice 1.3 (Communication entre activity par Intent)

L'objectif est ici de réaliser une deuxième activity Android qui affiche, dans un premier temps, les informations saisies dans la première activity. Pour appeler cette deuxième activity, il faudra utiliser le mécanisme des `Intent` explicites.

1. Commencez par réaliser une `Activity` qui permet l'affichage des données Nom, Prenom, Date de naissance et Ville de naissance (utilisez le projet existant, en veillant à bien déclarer cette nouvelle activité dans le fichier `AndroidManifest.xml`).

2. modifiez ensuite la première activity de la façon suivante : lors d'un appui sur le bouton "Valider", un Intent contenant les informations saisies, devra être transmis à la deuxième activity. Utilisez les champs `extra` de l'intent pour transmettre les données nécessaires.
3. modifiez la deuxième activity de façon à récupérer et afficher les données saisies.
4. lorsqu'il y a des objets complexes ou beaucoup de données à transmettre entre activités, il est généralement préférable d'utiliser des objets `Parcelable`. Pour cela, créez une classe qui contient les données à transmettre et qui implémente `Parcelable`. Implémentez les méthodes `describeContents()` et `writeToParcel(Parcel dest, int flags)`. Utilisez la méthode `Parcel.writeString(...)` pour sérialiser vos données dans le parcel `dest`.
5. pour récupérer les données dans un `Parcelable`, il est nécessaire de déclarer un objet Creator qui permet de reconstruire vos données à partir d'un `Parcel` (cf documentation `http://developer.android.com/reference/android/os/Parcelable.html`).
6. enfin, utilisez ensuite classiquement la méthode `Intent.putExtra(...)` pour ajouter votre objet `Parcelable` à l'intent.