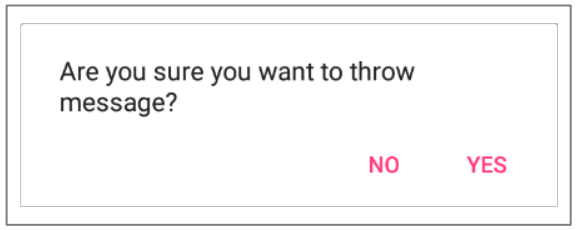
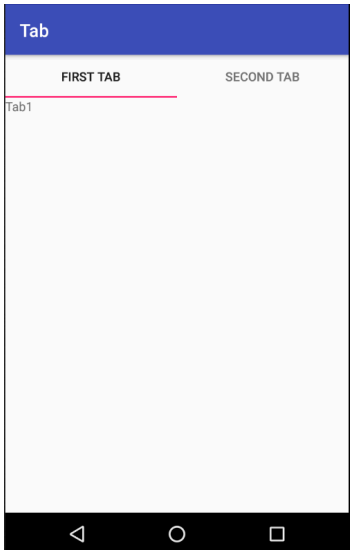
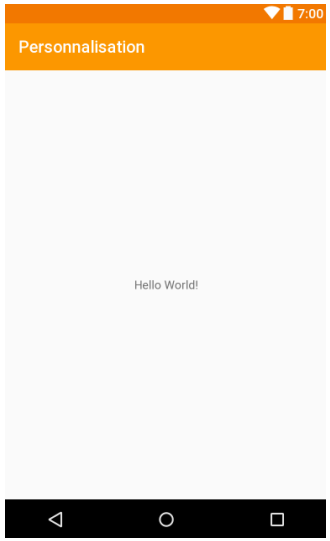
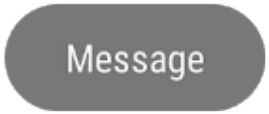
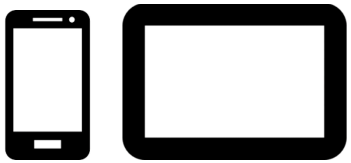
[TD3] Interfaces avancées & Données







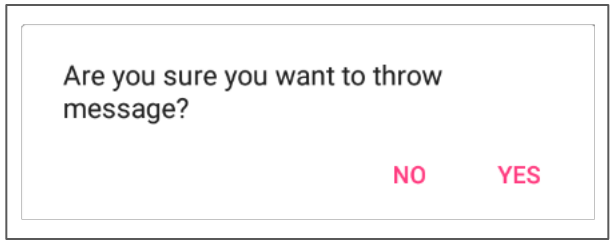




LES POP-UPS

AlertDialog

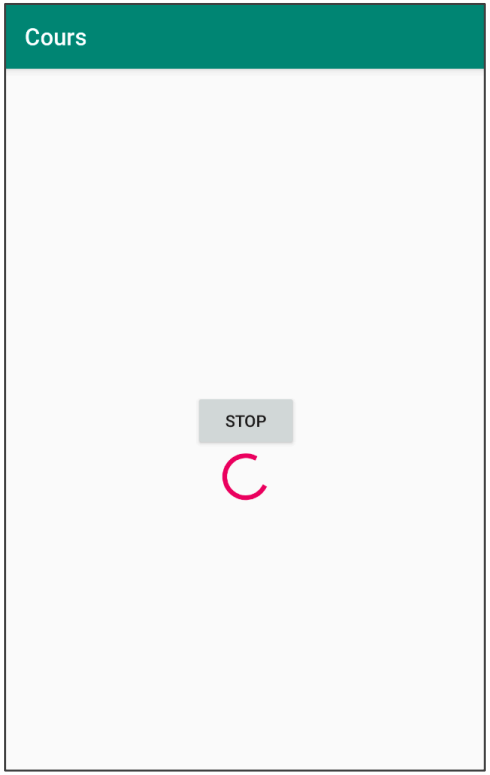
```
AlertDialog.Builder builder = new AlertDialog.Builder(MainActivity.this);
builder.setMessage("Are you sure you want to throw message?");
builder.setCancelable(false);
builder.setPositiveButton("Yes", new DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialog, int i) {
        Toast.makeText(MainActivity.this, "Hello",
Toast.LENGTH_SHORT).show();
    }
});
builder.setNegativeButton("No", new DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialog, int i) {
        dialog.cancel();
    }
});
builder.create().show();
```



PROGRESS BAR

Barre de progression non bornée

```xml
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <Button
        android:id="@+id/button_stop"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintBottom_toBottomOf="parent"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Stop" />

    <ProgressBar
        android:id="@+id/progressBar"
        style="?android:attr/progressBarStyle"
        app:layout_constraintTop_toBottomOf="@id/button_stop"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

</androidx.constraintlayout.widget.ConstraintLayout>
```



PROGRESS BAR

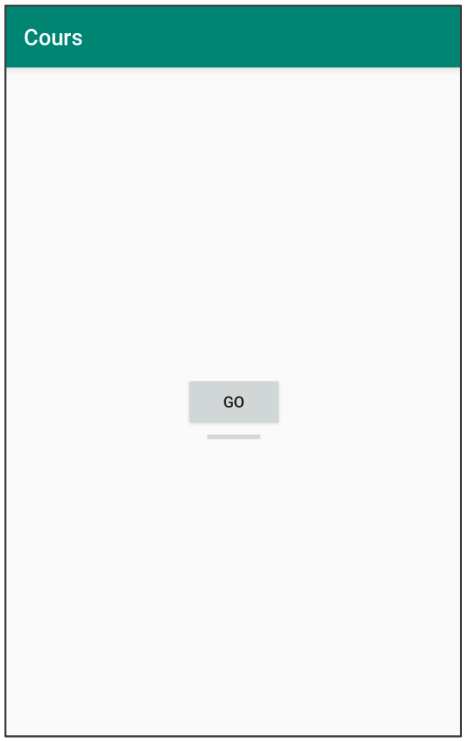
Barre de progression non bornée

```
public class MainActivity extends AppCompatActivity {

    private Button stop;
    private ProgressBar progressBar;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        stop = findViewById(R.id.button_stop);
        progressBar = findViewById(R.id.progressBar);
    }

    @Override
    protected void onStart() {
        super.onStart();
        stop.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                progressBar.setVisibility(view.INVISIBLE);
            }
        });
    }

}
```

PROGRESS BAR

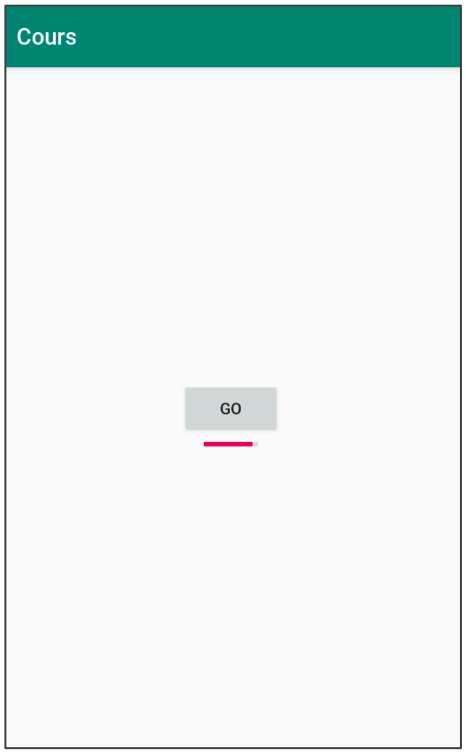
Barre de progression bornée

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <Button
        android:id="@+id/button_go"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintBottom_toBottomOf="parent"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="GO" />

    <ProgressBar
        android:id="@+id/progressBar"
        style="?android:attr/progressBarStyleHorizontal"
        app:layout_constraintTop_toBottomOf="@id/button_go"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:max="100"/>

</androidx.constraintlayout.widget.ConstraintLayout>
```

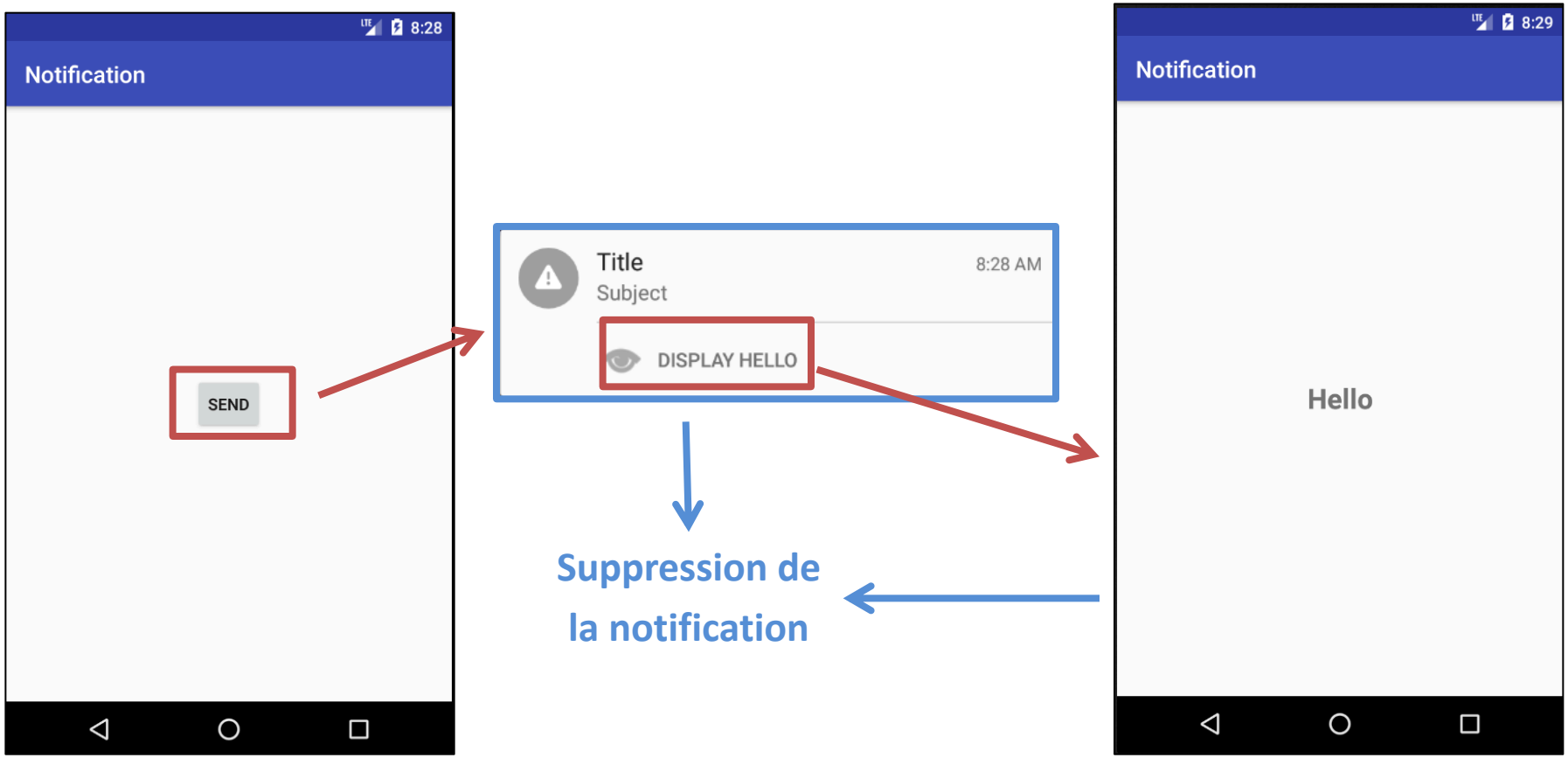


PROGRESS BAR

```
public class MainActivity extends AppCompatActivity {

    private Button go;
    private ProgressBar progressBar;
    private Handler handler;
    private Integer progress = 0;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        go = findViewById(R.id.button_go);
        progressBar = findViewById(R.id.progressBar);
        handler = new Handler();
    }

    @Override
    protected void onStart() {
        super.onStart();
        go.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                new Thread(new Runnable() {
                    @Override
                    public void run() {
                        while (progress < 10) {
                            try {
                                Thread.sleep(1000);
                            } catch (InterruptedException e) {
                                e.printStackTrace();
                            }
                            progress++;
                            handler.post(new Runnable() {
                                @Override
                                public void run() {
                                    progressBar.setProgress(progress*10);
                                }
                            });
                        }
                    }
                }).start();
            }
        });
    }

}
```



NOTIFICATIONS



NOTIFICATIONS

```
public int ID_NOTIFICATION = 1;
public int REQUEST_CODE = (int) 2;


NotificationManagerCompat manager = NotificationManagerCompat.from(this);
Intent intent = new Intent(this,MainActivity.class);
PendingIntent pendingIntent = PendingIntent.getActivity(this,
        REQUEST_CODE,intent,PendingIntent.FLAG_ONE_SHOT);
NotificationCompat.Builder notification = new NotificationCompat.Builder (this)
        .setContentTitle("Title")
        .setContentText("Subject")
        .setSmallIcon(android.R.drawable.ic_dialog_alert)
        .setWhen(System.currentTimeMillis())
        .setPriority(NotificationCompat.PRIORITY_MAX)
        .setAutoCancel(true);


Intent actionIntent = new Intent(this,NotifActivity.class);
actionIntent.putExtra("ID_NOTIFICATION",ID_NOTIFICATION);
PendingIntent actionPendingIntent = PendingIntent.getActivity(this,
        REQUEST_CODE,actionIntent,PendingIntent.FLAG_ONE_SHOT);
notification.addAction(android.R.drawable.ic_menu_view,
        "Display hello", actionPendingIntent);


notification.setContentIntent(pendingIntent);
manager.notify(ID_NOTIFICATION,notification.build());
```

NOTIFICATIONS

```
public int ID_NOTIFICATION = 1;
public int REQUEST_CODE = (int) 2;

NotificationManagerCompat manager = NotificationManagerCompat.from(this);
Intent intent = new Intent(this,MainActivity.class);
PendingIntent pendingIntent = PendingIntent.getActivity(this,
        REQUEST_CODE,intent,PendingIntent.FLAG_ONE_SHOT);
NotificationCompat.Builder notification = new NotificationCompat.Builder (this)
        .setContentTitle("Title")
        .setContentText("Subject")
        .setSmallIcon(android.R.drawable.ic_dialog_alert)
        .setWhen(System.currentTimeMillis())
        .setPriority(NotificationCompat.PRIORITY_MAX)
        .setAutoCancel(true);


Intent actionIntent = new Intent(this,NotifActivity.class);
actionIntent.putExtra("ID_NOTIFICATION",ID_NOTIFICATION);
PendingIntent actionPendingIntent = PendingIntent.getActivity(this,
        REQUEST_CODE,actionIntent,PendingIntent.FLAG_ONE_SHOT);
notification.addAction(android.R.drawable.ic_menu_view,
        "Display hello", actionPendingIntent);


notification.setContentIntent(pendingIntent);
manager.notify(ID_NOTIFICATION,notification.build());
```

Récupérer une instance du gestionnaire de notification.

NOTIFICATIONS

```
public int ID_NOTIFICATION = 1;
public int REQUEST_CODE = (int) 2;

NotificationManagerCompat manager = NotificationManagerCompat.from(this);
Intent intent = new Intent(this,MainActivity.class);
PendingIntent pendingIntent = PendingIntent.getActivity(this,
        REQUEST_CODE,intent,PendingIntent.FLAG_ONE_SHOT);
NotificationCompat.Builder notification = new NotificationCompat.Builder (this)
        .setContentTitle("Title")
        .setContentText("Subject")
        .setSmallIcon(android.R.drawable.ic_dialog_alert)
        .setWhen(System.currentTimeMillis())
        .setPriority(NotificationCompat.PRIORITY_MAX)
        .setAutoCancel(true);


Intent actionIntent = new Intent(this,NotifActivity.class);
actionIntent.putExtra("ID_NOTIFICATION",ID_NOTIFICATION);
PendingIntent actionPendingIntent = PendingIntent.getActivity(this,
        REQUEST_CODE,actionIntent,PendingIntent.FLAG_ONE_SHOT);
notification.addAction(android.R.drawable.ic_menu_view,
        "Display hello", actionPendingIntent);


notification.setContentIntent(pendingIntent);
manager.notify(ID_NOTIFICATION,notification.build());
```

Créer un **PendingIntent** qui permet à un mécanisme externe de pouvoir déclencher un intent dans l'application.

La méthode **getActivity** permet de créer un PendengIntent, les arguments sont :

- le contexte
- code de la requête
- intent (activité à lancer)
- une option :
  - **FLAG_CANCEL_CURRENT**
  - **FLAG_NO_CANCEL**
  - **FLAG_ONE_SHOT**
  - **FLAG_UPDATE_CURRENT**

NOTIFICATIONS

```
public int ID_NOTIFICATION = 1;
public int REQUEST_CODE = (int) 2;

NotificationManagerCompat manager = NotificationManagerCompat.from(this);
Intent intent = new Intent(this,MainActivity.class);
PendingIntent pendingIntent = PendingIntent.getActivity(this,
        REQUEST_CODE,intent,PendingIntent.FLAG_ONE_SHOT);
NotificationCompat.Builder notification = new NotificationCompat.Builder (this)
        .setContentTitle("Title")
        .setContentText("Subject")
        .setSmallIcon(android.R.drawable.ic_dialog_alert)
        .setWhen(System.currentTimeMillis())
        .setPriority(NotificationCompat.PRIORITY_MAX)
        .setAutoCancel(true);

Intent actionIntent = new Intent(this,NotifActivity.class);
actionIntent.putExtra("ID_NOTIFICATION",ID_NOTIFICATION);
PendingIntent actionPendingIntent = PendingIntent.getActivity(this,
        REQUEST_CODE,actionIntent,PendingIntent.FLAG_ONE_SHOT);
notification.addAction(android.R.drawable.ic_menu_view,
        "Display hello", actionPendingIntent);

notification.setContentIntent(pendingIntent);
manager.notify(ID_NOTIFICATION,notification.build());
```

Créer un **builder**. Vous pourrez, entre autres, ajouter :

un titre

une description

une icône

préciser le temps

préciser l'importance

faire que la notification disparaisse lorsque l'on clique dessus.

NOTIFICATIONS

```
public int ID_NOTIFICATION = 1;
public int REQUEST_CODE = (int) 2;

NotificationManagerCompat manager = NotificationManagerCompat.from(this);
Intent intent = new Intent(this,MainActivity.class);
PendingIntent pendingIntent = PendingIntent.getActivity(this,
        REQUEST_CODE,intent,PendingIntent.FLAG_ONE_SHOT);
NotificationCompat.Builder notification = new NotificationCompat.Builder (this)
        .setContentTitle("Title")
        .setContentText("Subject")
        .setSmallIcon(android.R.drawable.ic_dialog_alert)
        .setWhen(System.currentTimeMillis())
        .setPriority(NotificationCompat.PRIORITY_MAX)
        .setAutoCancel(true);

Intent actionIntent = new Intent(this,NotifActivity.class);
actionIntent.putExtra("ID_NOTIFICATION",ID_NOTIFICATION);
PendingIntent actionPendingIntent = PendingIntent.getActivity(this,
        REQUEST_CODE,actionIntent,PendingIntent.FLAG_ONE_SHOT);
notification.addAction(android.R.drawable.ic_menu_view,
        "Display hello", actionPendingIntent);

notification.setContentIntent(pendingIntent);
manager.notify(ID_NOTIFICATION,notification.build());
```

Associer le PendingIntent et la notification.

Enfin, déclencher la notification grâce à la méthode **notify**.

NOTIFICATIONS

```
public int ID_NOTIFICATION = 1;
public int REQUEST_CODE = (int) 2;

NotificationManagerCompat manager = NotificationManagerCompat.from(this);
Intent intent = new Intent(this,MainActivity.class);
PendingIntent pendingIntent = PendingIntent.getActivity(this,
        REQUEST_CODE,intent,PendingIntent.FLAG_ONE_SHOT);
NotificationCompat.Builder notification = new NotificationCompat.Builder (this)
        .setContentTitle("Title")
        .setContentText("Subject")
        .setSmallIcon(android.R.drawable.ic_dialog_alert)
        .setWhen(System.currentTimeMillis())
        .setPriority(NotificationCompat.PRIORITY_MAX)
        .setAutoCancel(true);

Intent actionIntent = new Intent(this,NotifActivity.class);
actionIntent.putExtra("ID_NOTIFICATION",ID_NOTIFICATION);
PendingIntent actionPendingIntent = PendingIntent.getActivity(this,
        REQUEST_CODE,actionIntent,PendingIntent.FLAG_ONE_SHOT);
notification.addAction(android.R.drawable.ic_menu_view,
        "Display hello", actionPendingIntent);

notification.setContentIntent(pendingIntent);
manager.notify(ID_NOTIFICATION,notification.build());
```

L'action associée à la notification sera ajoutée avec la méthode **addAction** qui prend comme arguments :

une icône

une description

un **PendingIntent** pour déclencher l'action lors du clic

NOTIFICATIONS

```
public class NotifActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_notif);
        NotificationManagerCompat manager = NotificationManagerCompat.from(this);
        manager.cancel(getIntent().getIntExtra("ID_NOTIFICATION",-1));
    }
}
```

Dans l'activité lancée, nous allons récupérer une instance du gestionnaire de notification et le code de la notification.

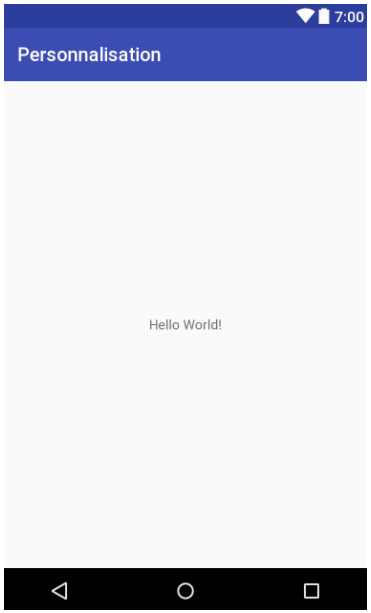
Nous pourrons ainsi supprimer la notification.

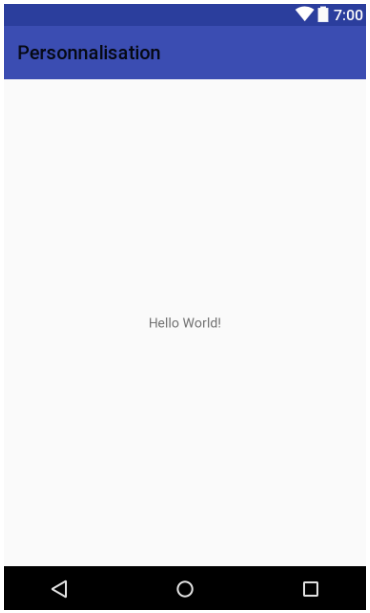
PERSONNALISER L'INTERFACE

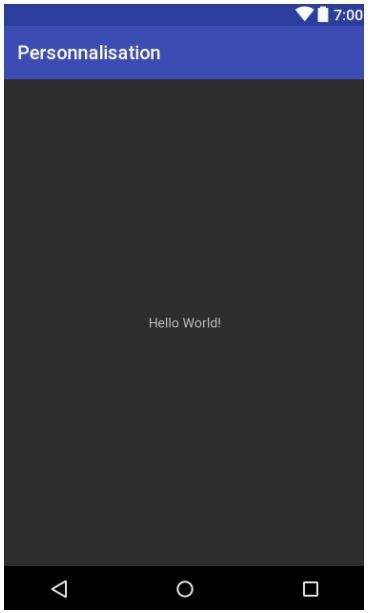
Thèmes

res/values/styles.xml

```xml
<resources>

    <style name="AppTheme" parent="Theme.AppCompat.Light.DarkActionBar">
        <item name="colorPrimary">@color/colorPrimary</item>
        <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
        <item name="colorAccent">@color/colorAccent</item>
    </style>

</resources>
```







PERSONNALISER L'INTERFACE

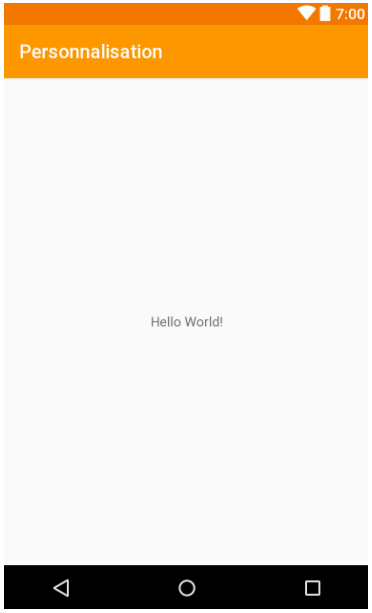
Thèmes

res/values/styles.xml

```
<resources>

    <style name="AppTheme" parent="Theme.AppCompat.Light">
        <item name="colorPrimary">@color/colorPrimary</item>
        <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
        <item name="android:textColorPrimary">@color/textColorPrimary</item>
    </style>

</resources>
```

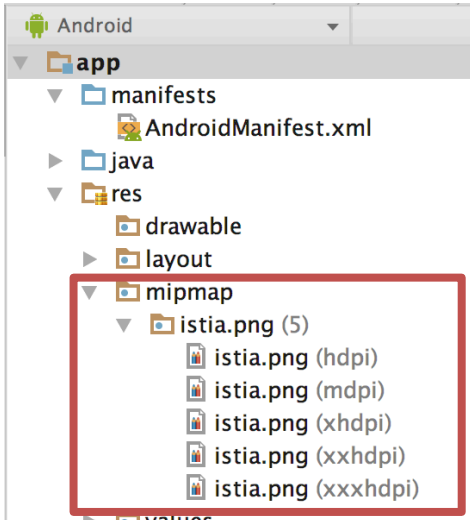
res/values/colors.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="colorPrimary">#FF9800</color>
    <color name="colorPrimaryDark">#F57C00</color>
    <color name="textColorPrimary">#FFFFFF</color>
</resources>
```



PERSONNALISER L'INTERFACE

Icônes



AndroidManifest.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3      package="com.cfc.personnalisation" >
4
5      <application
6          android:allowBackup="true"
7          android:icon="@mipmap/istia"
8          android:label="@string/app_name"
9          android:supportsRtl="true"
10         android:theme="@style/AppTheme" >
11         <activity android:name=".MainActivity" >
12             <intent-filter>
13                 <action android:name="android.intent.action.MAIN" />
14
15                 <category
16 android:name="android.intent.category.LAUNCHER" />
17             </intent-filter>
18         </activity>
19     </application>
20
21 </manifest>
```

GESTION D'ÉVÉNEMENTS

Appuyer sur une touche

```
@Override
public boolean onKeyDown(int keyCode, KeyEvent event) {
    super.onKeyDown(keyCode, event);
    if(keyCode == KeyEvent.KEYCODE_VOLUME_UP) {
        Toast.makeText(MainActivity.this, "Volume up!",Toast.LENGTH_LONG).show();
        return true;
    }
    return false;
}
```

Deux méthodes permettent de gérer les événements d'appui :

onKeyDown : quand une touche est appuyée

onKeyUp : quand une touche est relâchée

GESTION D'ÉVÉNEMENTS

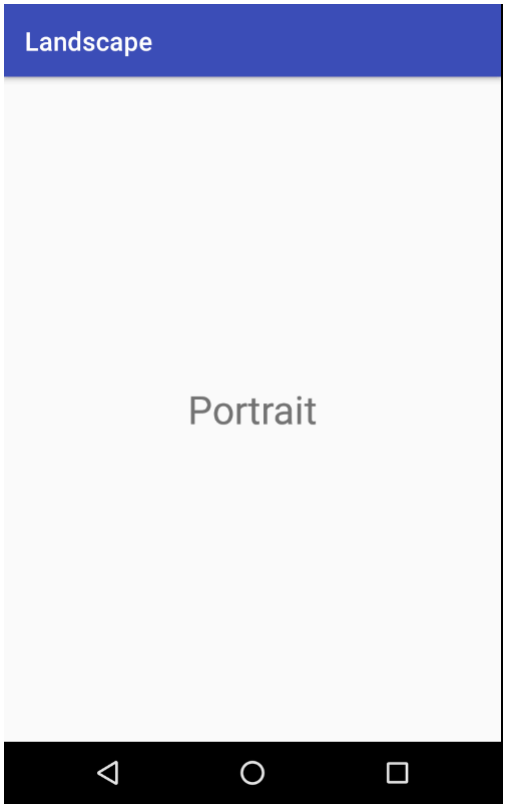
Toucher l'écran

```
public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    @Override
    public boolean onTouchEvent(MotionEvent event) {
        if(event.getAction() == MotionEvent.ACTION_UP) {
            String p = "("+Float.toString(event.getX())+" ; "
                    +Float.toString(event.getY())+")";
            Toast.makeText(MainActivity.this, p, Toast.LENGTH_LONG).show();
        }
        return true;
    }
}
```

GESTION D'ÉVÉNEMENTS

Mode paysage

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="30sp"
        android:text="Portrait"/>

</LinearLayout>
```

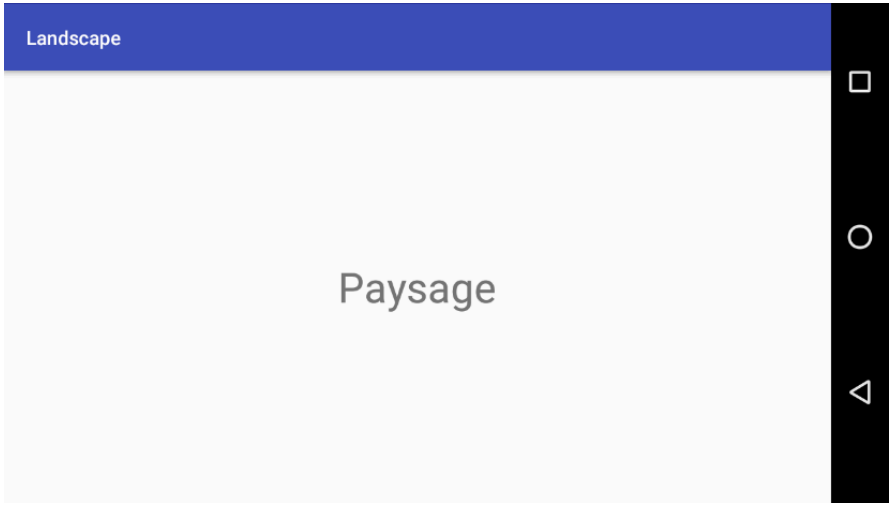


GESTION D'ÉVÉNEMENTS

Mode paysage

res/layout-land/

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="30sp"
        android:text="Paysage"/>

</LinearLayout>
```



GESTION D'ÉVÉNEMENTS

Mode paysage

Il est aussi possible de bloquer des vues dans le mode portrait ou paysage.

Manifeste :

```
<activity android:name=".MainActivity"
    android:screenOrientation="portrait">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

GESTION D'ÉVÉNEMENTS

Mode paysage

Il est possible de redéfinir le comportement lors de la rotation.

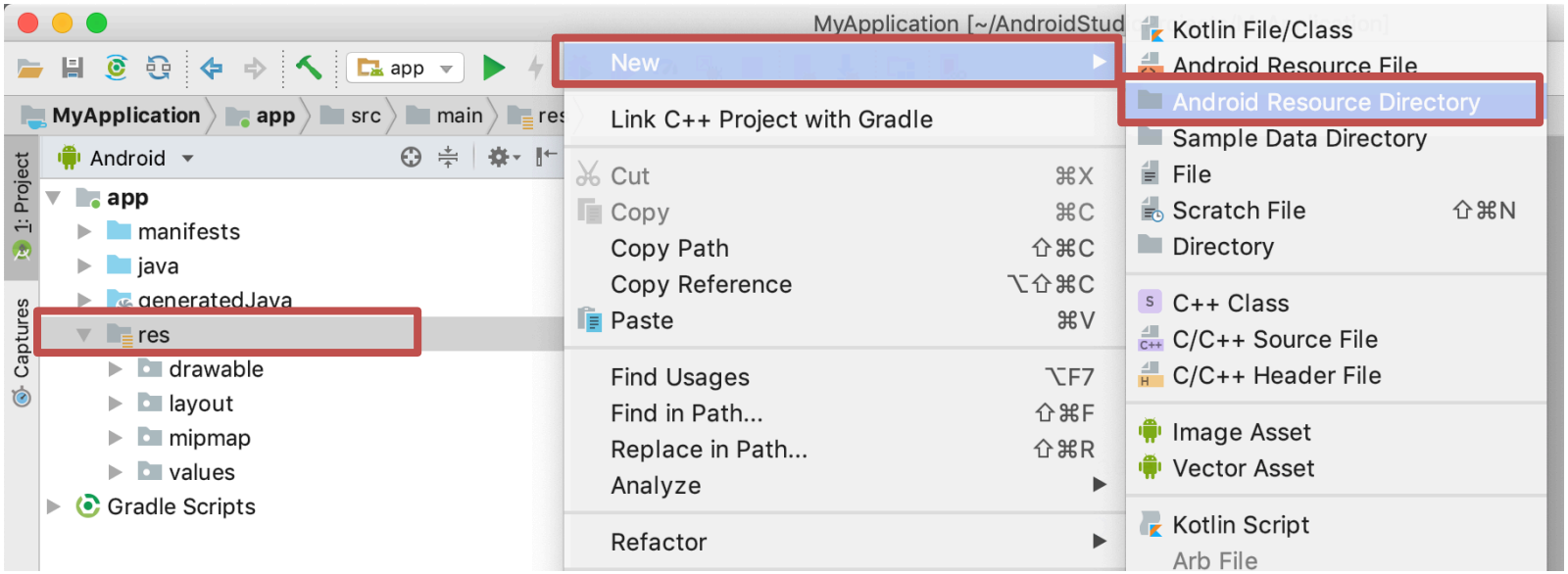
Manifeste : `android:configChanges="orientation|screenSize">` ← *Attribut de la balise activity*

```
public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    @Override
    public void onConfigurationChanged(Configuration configuration) {
        super.onConfigurationChanged(configuration);
        String message = "Portrait";
        if(configuration.orientation==Configuration.ORIENTATION_LANDSCAPE) {
            message = "Landscape";
        }
        Toast.makeText(MainActivity.this,message,Toast.LENGTH_LONG).show();
    }
}
```

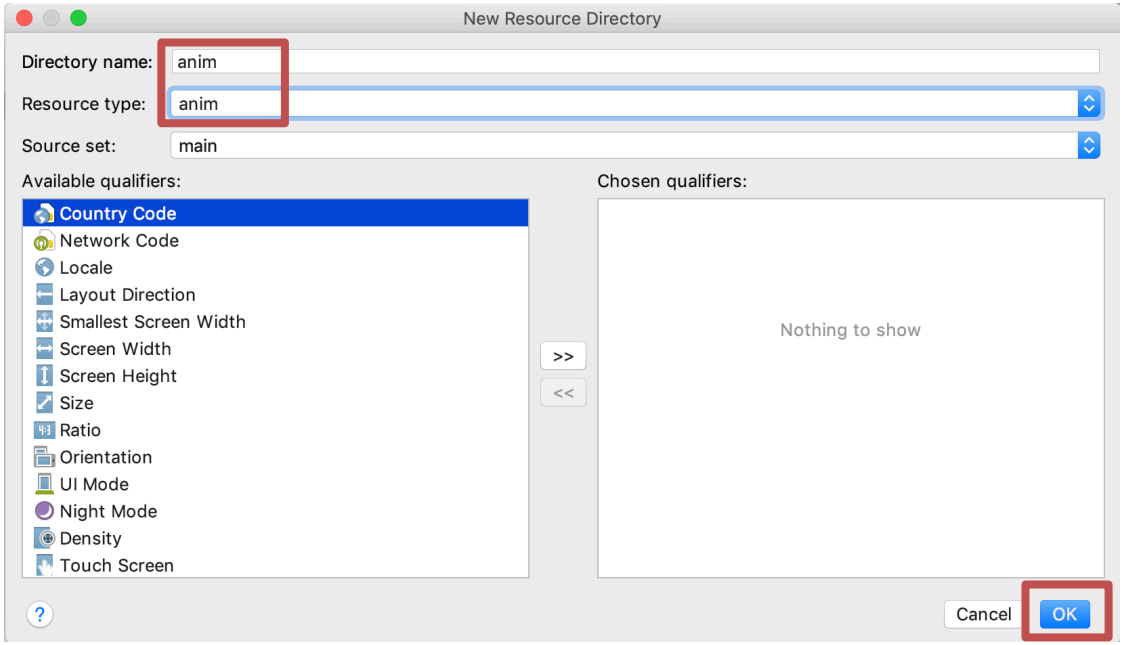
ANIMATION

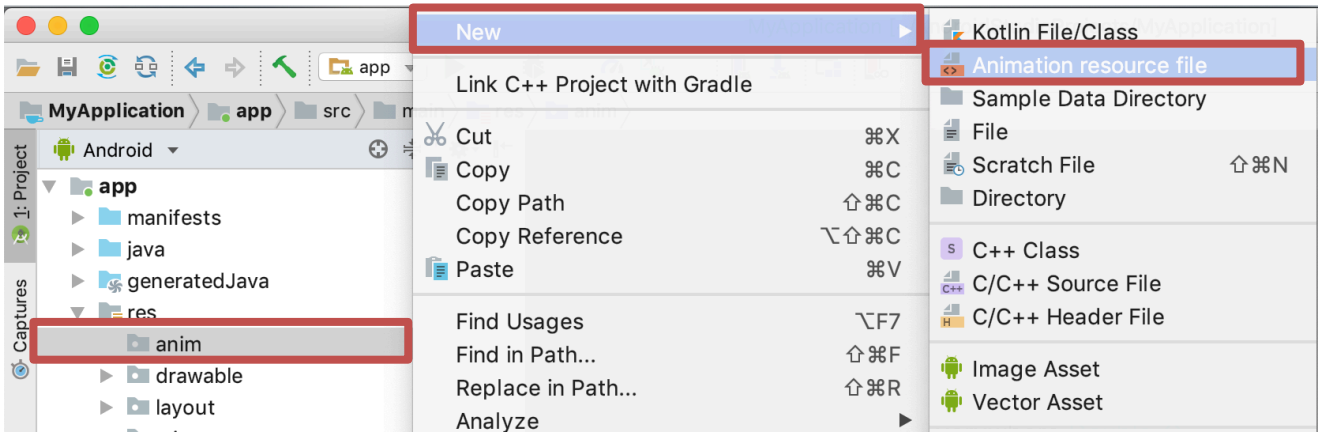
Il est possible de créer des animations (rotation, translation, changement de dimension, etc.) en créant une ressource dans le répertoire res/anim

Il faut d'abord créer ce répertoire :

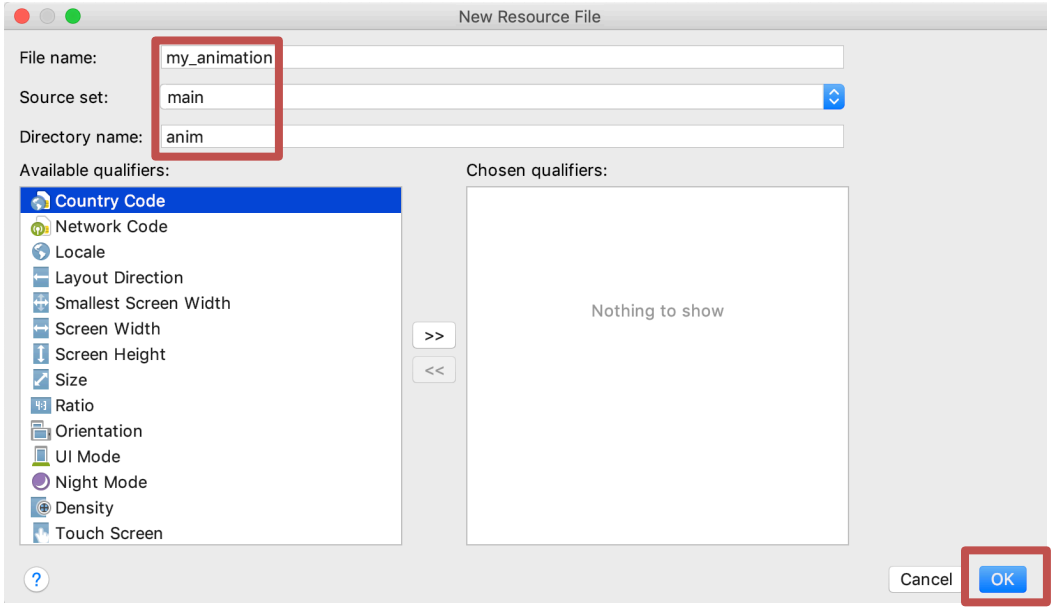


ANIMATION


New Resource Directory
Directory name: anim
Resource type: anim
Source set: main
Available qualifiers:
Country Code
Network Code
Locale
Layout Direction
Smallest Screen Width
Screen Width
Screen Height
Size
Ratio
Orientation
UI Mode
Night Mode
Density
Touch Screen
>>
<<
Chosen qualifiers:
Nothing to show
Cancel
OK


MyApplication
app
src
Android
1: Project
Captures
app
manifests
java
generatedJava
res
anim
drawable
layout
New
Link C++ Project with Gradle
Cut ⌘X
Copy ⌘C
Copy Path ⇧⌘C
Copy Reference ⌥⇧⌘C
Paste ⌘V
Find Usages ⌥F7
Find in Path... ⇧⌘F
Replace in Path... ⇧⌘R
Analyze
Kotlin File/Class
Animation resource file
Sample Data Directory
File
Scratch File ⇧⌘N
Directory
C++ Class
C/C++ Source File
C/C++ Header File
Image Asset
Vector Asset

ANIMATION



Il faut maintenant définir le comportement de l'animation dans le xml.

ANIMATION

<alpha> : faire apparaître ou disparaître une vue.

<rotate> : faire tourner une vue autour d'un axe.

<scale> : agrandir ou de réduire une vue.

<translate> : translater à une vue.

Exemple d'une animation d'une image

```xml
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
    <alpha
        android:fromAlpha = "1.0"
        android:toAlpha = "0.5"
        android:duration = "300">
    </alpha>
    <scale
        android:fromXScale = "1"
        android:toXScale = "0.9"
        android:fromYScale = "1"
        android:toYScale = "0.9"
        android:pivotX="50%"
        android:pivotY="50%"
        android:duration = "50">
    </scale>
</set>
```

```java
public class MainActivity extends AppCompatActivity {

    private ImageView img;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        img = findViewById(R.id.with_anim);
        img.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                img.startAnimation(AnimationUtils.loadAnimation(getApplication(),
R.anim.img_anim));
            }
        });
    }
}
```

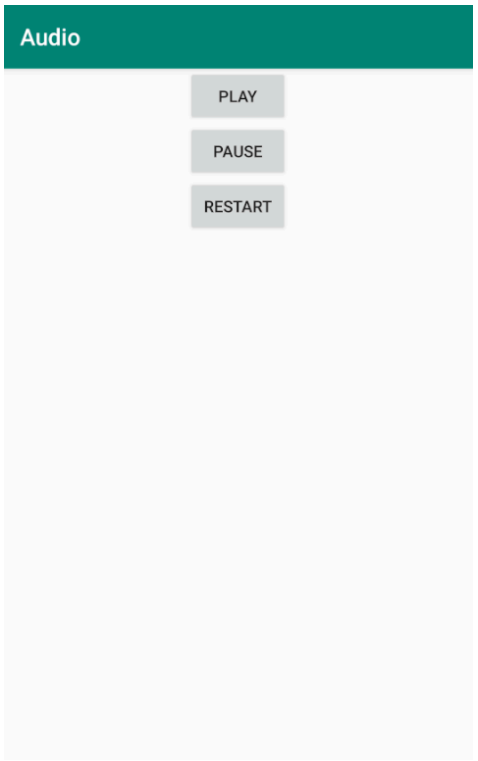
MUSIQUE

La classe MediaPlayer vous permettra de gérer des fichiers audio / vidéo.

Commencer par créer un répertoire res/raw où vous déposerez vos fichiers (méthode similaire au répertoire des animations).

Exemple de lecture d'un fichier audio

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <Button
        android:id="@+id/play"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:text="Play"/>
    <Button
        android:id="@+id/pause"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:text="Pause"/>

    <Button
        android:id="@+id/restart"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:text="Restart"/>

</LinearLayout>
```



```
public class MainActivity extends AppCompatActivity {

    private Button play, pause, restart;
    private MediaPlayer mediaPlayer;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        play = findViewById(R.id.play);
        pause = findViewById(R.id.pause);
        restart = findViewById(R.id.restart);
        mediaPlayer = MediaPlayer.create(this, R.raw.youarewrong);
    }

    @Override
    protected void onResume() {
        super.onResume();
        play.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                mediaPlayer.start();
            }
        });
        pause.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                mediaPlayer.pause();
            }
        });
        restart.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                mediaPlayer.seekTo(0);
                mediaPlayer.start();
            }
        });
    }
}
```

Déclaration du MediaPlayer

Initialisation avec le fichier audio

Démarrage de la lecture

Mise en pause

Repositionnement de la lecture à 0 milliseconde et démarrage de la lecture

FRAGMENTS

Un fragment est une interface utilisateur indépendante et réutilisable. Il possède sa propre vue ainsi que son propre cycle de vie.

Un fragment ne peut pas exister sans être attaché à une activité.

FRAGMENTS

onCreate : appelée à la création du fragment

initialiser les composants essentiels du Fragment

onCreateView : appelée lorsque le fragment dessine son contenu

pour dessiner une interface utilisateur pour votre fragment

onActivityCreated : appelée quand le fragment a fini de se créer et de se dessiner

onStart : passage du fragment au premier plan

onResume : le fragment interagit avec l'utilisateur

onPause : le fragment devient inactif

pour « sauvegarder » qui doivent être les éléments persistants

onStop : le fragment n'est plus au premier plan

Destruction du fragment

onDestroyView : destruction de la vue

onDestroy : destruction du fragment

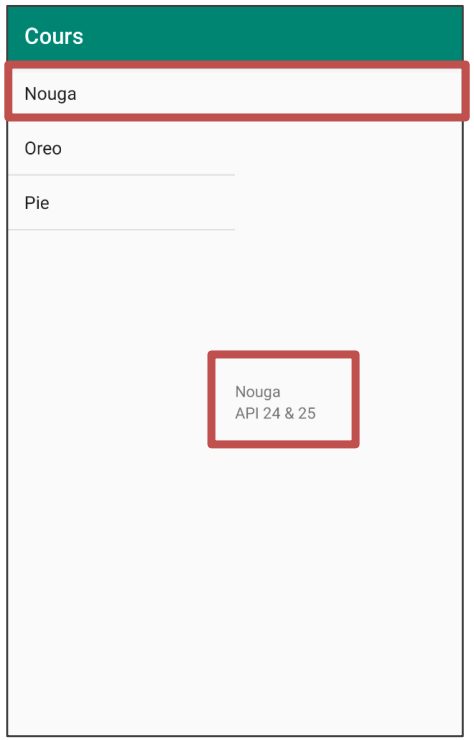
onDetach : le fragment est séparé de l'activité parente

FRAGMENTS

Fragments statiques

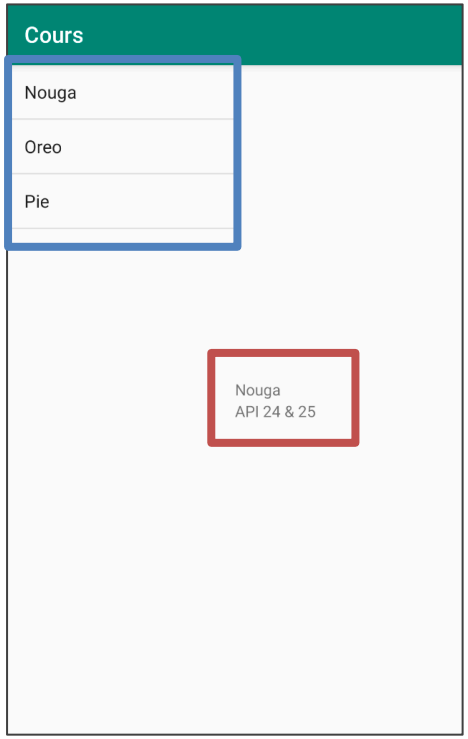
Ils sont déclarés dans le layout de l'activité à l'aide de la balise **fragment**.

Ils ne peuvent pas être ajoutés, remplacés ou supprimés dynamiquement.



FRAGMENTS

Fragments statiques



Pour cela deux fragments vont être créés :

DetailFragment : représentant la vue détaillée

ListFragment : représentant une liste des versions

FRAGMENTS

Fragments statiques

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal">

    <fragment
        android:id="@+id/listFragment"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="1"
        class="com.cfc.cours.ListFragment"/>

    <fragment
        android:id="@+id/detailFragment"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="1"
        class="com.cfc.cours.DetailFragment"/>

</LinearLayout>
```

```java
public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

FRAGMENTS

Fragments statiques

```
public class DetailFragment extends Fragment {

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        return inflater.inflate(R.layout.fragment_detail, container, false);
    }

    public void setText(String item) {
        TextView version = getView().findViewById(R.id.version);
        version.setText(item);
        TextView API = getView().findViewById(R.id.API);
        switch (item) {
            case "Nouga" :
                API.setText("API 24 & 25");
                break;
            case "Oreo" :
                API.setText("API 26 & 27");
                break;
            case "Pie" :
                API.setText("API 28");
                break;
        }
    }
}
```

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center"
    android:orientation="vertical">

    <TextView
        android:id="@+id/version"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"/>

    <TextView
        android:id="@+id/API"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"/>

</LinearLayout>
```

FRAGMENTS

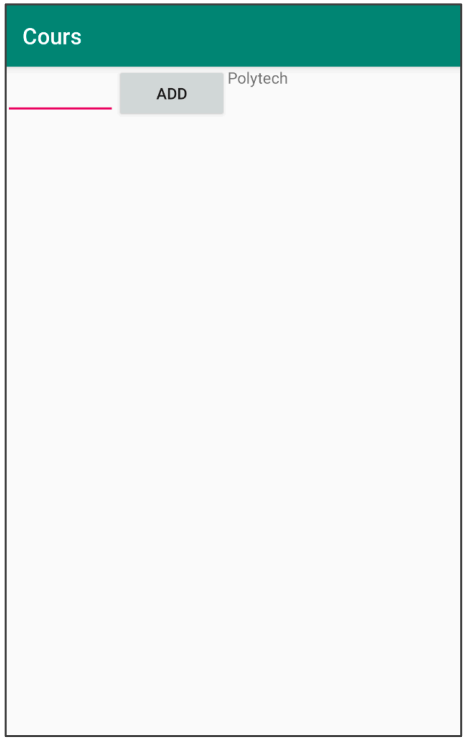
Fragments statiques

```
public class ListFragment extends androidx.fragment.app.ListFragment {

    private final List<String> values = Arrays.asList("Nouga","Oreo","Pie");

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        ArrayAdapter<String> adapter = new ArrayAdapter<>(getActivity(),
                android.R.layout.simple_list_item_1,values);
        setListAdapter(adapter);
    }

    @Override
    public void onListItemClick(ListView l, View v, int pos, long id) {
        String item = (String) getListAdapter().getItem(pos);
        DetailFragment fragment = (DetailFragment)
                getFragmentManager().findFragmentById(R.id.detailFragment);
        fragment.setText(item);
    }
}
```

FRAGMENTS

Fragments dynamiques

Les fragments dynamiques peuvent être ajoutés, supprimés ou remplacés pendant l'exécution d'une activité.



FRAGMENTS

Fragments dynamiques

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    android:layout_gravity="center_horizontal">

    <EditText
        android:id="@+id/text"
        android:layout_width="100dp"
        android:layout_height="wrap_content"/>

    <Button
        android:id="@+id/button"
        android:layout_width="100dp"
        android:layout_height="wrap_content"
        android:text="ADD"/>

    <LinearLayout
        android:id="@+id/fragment_container"
        android:layout_width="wrap_content "
        android:layout_height="wrap_content"
        android:orientation="vertical">
    </LinearLayout>

</LinearLayout>
```

```
public class MainActivity extends AppCompatActivity {

    private Button add;
    private TextView message;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        add = findViewById(R.id.button);
        message = findViewById(R.id.text);
    }

    @Override
    protected void onStart() {
        super.onStart();
        add.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Bundle bundle = new Bundle();
                bundle.putString("message",message.getText().toString());
                MyFragment fragment = new MyFragment();
                fragment.setArguments(bundle);
                getSupportFragmentManager().beginTransaction()
                        .add(R.id.fragment_container, fragment)
                        .commit();
            }
        });
    }
}
```

FRAGMENTS

```java
public class MyFragment extends Fragment {

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        String message = this.getArguments().getString("message");
        View view = inflater.inflate(R.layout.fragment_my, container, false);
        TextView text = view.findViewById(R.id.text);
        text.setText(message);
        return view;
    }
}
```

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <TextView
        android:id="@+id/text"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_gravity="center_horizontal"/>

</LinearLayout>
```

FRAGMENTS

Fragments dynamiques

À noter, vous pourrez aussi cacher ou rendre visible un fragment :

```
// Cacher un fragment nommé myFrag
FragmentTransaction ft = getSupportFragmentManager().beginTransaction();
ft.hide(myFrag);
ft.commit();

// Rendre visible un fragment nommé myFrag
FragmentTransaction ft = getSupportFragmentManager().beginTransaction();
ft.show(myFrag);
ft.commit();
```

PRÉSENTATION

Afin de stocker des données vous pourrez utiliser trois types de stockages :

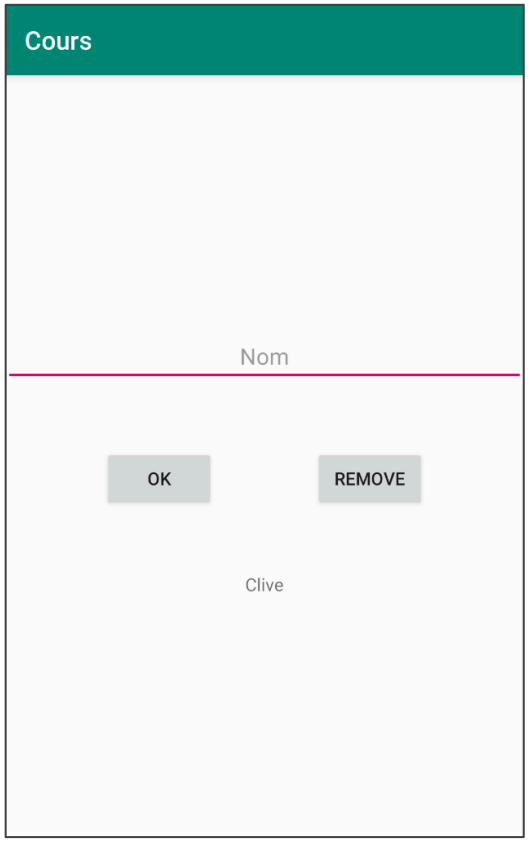
- **SharedPreferences** : données sous forme de paires clef / valeur
- **Fichiers** : données stockées dans des fichiers
- **Base de données** : données stockées dans une base de données SQLite

SHAREDPREFERENCES

Les données stockées avec cette méthode seront persistantes jusqu'à la désinstallation de l'application.

Les données seront stockées sous forme de paire clef / valeur où la valeur sera un type primitif (float, int, …).

```xml
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <EditText
        android:id="@+id/edit"
        app:layout_constraintTop_toTopOf="parent"
        android:layout_marginTop="200dp"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:hint="Nom"/>

    <Button
        android:id="@+id/button_ok"
        app:layout_constraintTop_toBottomOf="@id/edit"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toLeftOf="@id/button_remove"
        android:layout_marginTop="50dp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="OK"/>

    <Button
        android:id="@+id/button_remove"
        app:layout_constraintTop_toBottomOf="@id/edit"
        app:layout_constraintLeft_toRightOf="@id/button_ok"
        app:layout_constraintRight_toRightOf="parent"
        android:layout_marginTop="50dp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Remove"/>

    <TextView
        android:id="@+id/text"
        app:layout_constraintTop_toBottomOf="@id/button_ok"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        android:layout_marginTop="50dp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>

</androidx.constraintlayout.widget.ConstraintLayout>
```



```
public class MainActivity extends AppCompatActivity {

    private EditText edit;
    private Button ok;
    private Button remove;
    private TextView text;
    private SharedPreferences prefs;
    private SharedPreferences.Editor editor;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        prefs = getSharedPreferences("MY_PREFS_NAME",MODE_PRIVATE);
        editor = prefs.edit();
        edit = findViewById(R.id.edit);
        ok = findViewById(R.id.button_ok);
        remove = findViewById(R.id.button_remove);
        text = findViewById(R.id.text);
    }

    @Override
    protected void onStart() {
        super.onStart();
        update();
        ok.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                editor.putString("NAME", edit.getText().toString());
                editor.apply();
                update();
            }
        });
        remove.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                editor.remove("NAME");
                editor.apply();
                update();
            }
        });
    }

    private void update() {
        text.setText(prefs.getString("NAME",""));
    }
}
```

SHAREDPREFERENCES

Gson

Pour des données plus complexes (List, etc.) l'utilisation de la librairie Gson peut s'avérer pratique :

Exemple d'enregistrement

```
SharedPreferences prefs = getSharedPreferences("MY_PREFS_NAME",MODE_PRIVATE);
SharedPreferences.Editor editor = prefs.edit();
Gson gson = new Gson();
String json = gson.toJson(list); // list is an ArrayList
editor.putString("LIST",json);
editor.apply();
```

Exemple de lecture

```
Gson gson = new Gson();
String json = prefs.getString("LIST","");
List<Integer> list = gson.fromJson(json,
   new TypeToken <ArrayList<Integer>>(){}.getType());
```

Il faut ajouter la dépense Gson dans le gradle :

```
implementation 'com.google.code.gson:gson:2.8.6'
```

PM_EI5_TD3_EXO1.pdf