

# Développement d'Applications Natives avec Android

## **LES ACTIVITÉS ET LES INTENTS**



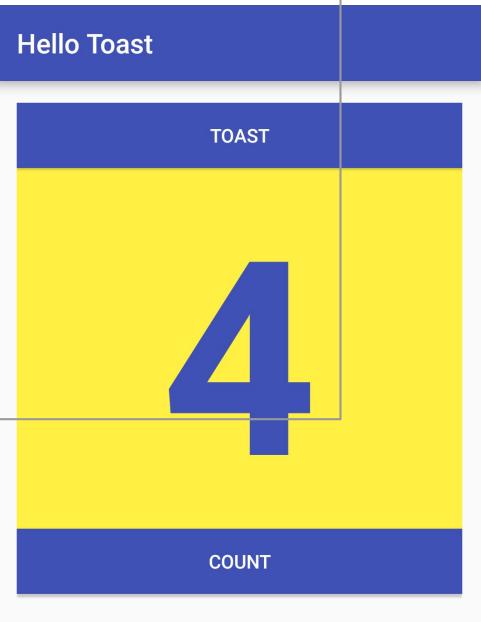
# Qu'est-ce qu'une activité?

- Une activité est un composant d'une application
- Représente une fenêtre, une hiérarchie de vues
- Remplit généralement l'écran, mais peut être intégré dans une autre activité ou apparaître comme une fenêtre flottante
- Classe Java, généralement une activité dans un fichier

# Que fait une activité?

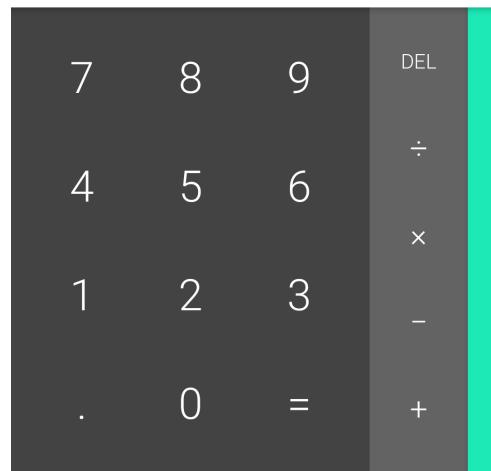
- Représente une activité, telle que la commande d'épicerie, l'envoi d'e-mails ou l'obtention d'un itinéraire
- Gère les interactions des utilisateurs, telles que les clics sur les boutons, la saisie de texte ou la vérification de la connexion
- Peut démarrer d'autres activités dans la même application ou dans d'autres
- A un cycle de vie: est créé, démarré, s'exécute, est mis en pause, repris, arrêté et détruit

## Exemples d'activités



$3.1415 \times 2.718$

8.538597



My Food List

*Cheese*

*Pepperoni*

*Black Olives*

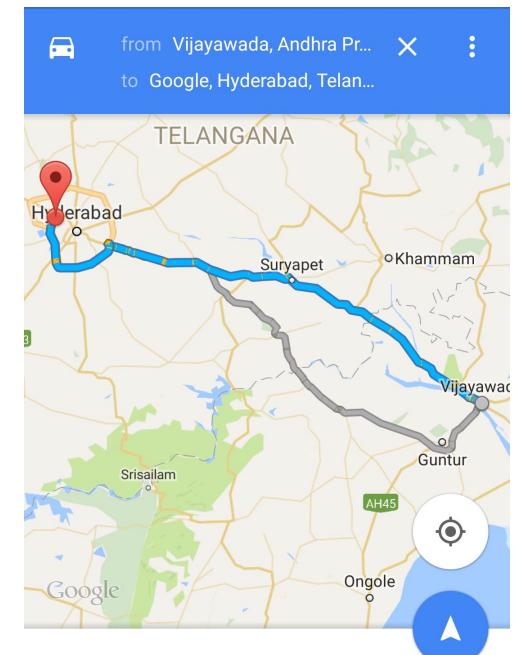
*Pineapple*

*Strawberries*

*Artichokes*

*Red peppers*

*Mushrooms*



4 hr 44 min (319 km) ⚡

Fastest route now, avoids road closure  
on NH65

# Applications et activités

- Les activités sont vaguement liées entre elles pour constituer une application
- La première activité que l'utilisateur voit est généralement appelée "activité principale"
- Les activités peuvent être organisées en relations parent-enfant dans le manifeste Android pour faciliter la navigation

# Layouts et Activités

- Une activité a généralement une disposition d'interface utilisateur(Layout)
- Le layout est généralement définie dans un ou plusieurs fichiers XML
- L'activité affiche le layout lors de sa création

# Mettre en œuvre de nouvelles activités

1. Définir la mise en page en XML
2. Définir la classe Java d'activité en étendant AppCompatActivity
3. Connectez l'activité avec la mise en page en définissant la vue du contenu dans onCreate ()
4. Déclarer une activité dans le manifeste Android

# 1. Définir la mise en page en XML

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Let's Shop for Food!" />
</RelativeLayout>
```

## 2. Définir la classe d'activité Java

```
public class MainActivity extends AppCompatActivity {  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
    }  
}
```

### 3. Connecter l'activité et le layout

```
public class MainActivity extends AppCompatActivity {  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
    }  
}
```

Resource is layout in this XML file

# 4. Déclarer une activité dans le manifeste Android

```
<activity android:name=".MainActivity">
```

# 4. Déclarer une activité dans le manifeste Android

MainActivity doit inclure un filtre d'intention pour démarrer à partir du lanceur

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

# Principe des intents

- Les Intents permettent de gérer l'envoi et la réception de messages afin de faire coopérer les applications. Le but des Intents est de déléguer une action à un autre composant, une autre application ou une autre activité de l'application courante. Un objet **Intent** contient les informations suivantes:
  - le nom du composant ciblé (facultatif)
  - l'action à réaliser, sous forme de chaîne de caractères
  - les données: contenu MIME et URI
  - des données supplémentaires sous forme de paires de clef/valeur
  - une catégorie pour cibler un type d'application
  - des drapeaux (informations supplémentaires)
- On peut envoyer des Intents informatifs pour faire passer des messages. Mais on peut aussi envoyer des Intents servant à lancer une nouvelle activité

# Que peuvent faire les intents?

- **Commencer une activité**
  - Un clic sur un bouton démarre une nouvelle activité pour la saisie de texte
  - Cliquer sur Partager ouvre une application qui vous permet de publier une photo
- **Démarrer un service**
  - Lancer le téléchargement d'un fichier en arrière-plan
- **Diffuser un Broadcast**
  - Le système informe tout le monde que le téléphone est en cours de chargement

# Intents explicites et implicites

- **Intent explicite:** démarre une activité spécifique
  - Demander du thé au lait livré par Mamadou
  - L'activité principale démarre l'activité ViewShoppingCart
- **Intent implicite:** demande au système de trouver une activité capable de gérer cette demande
  - Trouvez un magasin ouvert qui vend du thé vert
  - Cliquer sur Partager ouvre un sélecteur avec une liste d'applications

# Démarrer une activité avec un intent explicite

- Plusieurs façons de créer l'objet de type Intent qui permettra de lancer une nouvelle activité. Si l'on passe la main à une activité interne à l'application, on peut créer l'Intent et passer la classe de l'activité ciblée par l'Intent:

```
Intent login = new Intent(this, SeLoguer.class);  
startActivity(login);
```

# Démarrer une activité avec un Intent explicite

- Plusieurs façons de créer l'objet de type Intent qui permettra de lancer une nouvelle activité. Si l'on passe la main à une activité interne à l'application, on peut créer l'Intent et passer la classe de l'activité ciblée par l'Intent:

## 1. Créer un Intent

```
Intent intent = new Intent(this, ActivityName.class);
```

## 2. Utiliser l'Intent pour démarrer l'Activité

startActivity(intent);

```
Intent login = new Intent(this, SeLoguer.class);
startActivity(login);
```

# Démarrer une activité avec une intent implicite

- Pour demander à Android de trouver une activité pour traiter votre demande, utilisez un Intent implicite

## 1. Créer un Intent

- `Intent intent = new Intent(action, uri);`

## 2. Utiliser l'Intent pour démarrer l'Activité

- `startActivity(intent);`

# Démarrer une activité avec une intent implicite

S'il s'agit de passer la main à une autre application, on donne au constructeur de l'Intent les données et l'URI cible: l'OS est chargé de trouver une application pouvant répondre à l'Intent.

```
Button b = (Button) findViewById(R.id.Button01);
b.setOnClickListener(new OnClickListener() {
    @Override
    public void onClick(View v) {
        Uri telnumber = Uri.parse("tel:772051779");
        Intent call = new Intent(Intent.ACTION_CALL, telnumber);
        startActivity(call);
    }
});
```

Sans oublier la permission dans le Manifest

```
<uses-permission android:name="android.permission.CALL_PHONE" />
```

# Intent avec résultat attendu en retour

- Lorsque le bouton *retour* est pressé, l'activité courante prend fin et revient à l'activité précédente. Cela permet par exemple de terminer son appel téléphonique et de revenir à l'interface ayant initié l'appel.
- Au sein d'une application, une activité peut vouloir récupérer un code de retour de l'activité "enfant". On utilise pour cela la méthode **startActivityForResult** qui envoie un code de retour à l'activité enfant.

**public void startActivityForResult (Intent intent, int requestCode)**

- Lorsque l'activité parent reprend la main, il devient possible de filtrer le code de retour dans la méthode **onActivityResult** pour savoir si l'on revient ou pas de l'activité enfant.

**protected void onActivityResult(int requestCode, int resultCode, Intent data)**

# Retour d'une activité

- L'appel d'un *Intent* devient donc dans l'activité parent:

```
public void onCreate(Bundle savedInstanceState) {  
    Intent login = new Intent(getApplicationContext(), SaisirNumeroTelephone.class);  
    startActivityForResult(login,48);  
    ...  
}
```

- Le filtrage dans la classe parent permet de savoir qui avait appelé cette activité enfant:

```
protected void onActivityResult(int requestCode, int resultCode, Intent data) {  
    if (requestCode == 48)  
        Toast.makeText(this, "Code de requête récupéré (je sais d'où je viens)", Toast.LENGTH_LONG).show();  
    }  
}
```

# Résultat d'une activité

- Il est aussi possible de définir un résultat d'activité, avant d'appeler explicitement la fin d'une activité avec la méthode **finish()**. Dans ce cas, la méthode **setResult** permet d'enregistrer un code de retour qu'il sera aussi possible de filtrer dans l'activité parente.
- Dans l'activité enfant, on met donc:

```
Button terminer = (Button)findViewById(R.id.terminer);
terminer.setOnClickListener(new OnClickListener() {
    @Override
    public void onClick(View v) {
        setResult(RESULT_OK, getIntent());
        finish();
    }
});
```

# Résultat d'une activité

- Et la classe parente peut filtrer ainsi:

```
protected void onActivityResult(int requestCode, int resultCode, Intent data){  
    if (requestCode == 48)  
        Toast.makeText(this, "Code de requête récupéré (je sais d'où je viens)",  
                    Toast.LENGTH_LONG).show();  
    if (requestCode == 50)  
        Toast.makeText(this, "Code de retour ok (on m'a renvoyé le bon code)",  
                    Toast.LENGTH_LONG).show();  
}
```

# Ajout d'informations

- Les Intent permettent de transporter des informations à destination de l'activité cible. On appelle ces informations des Extra: les méthodes permettant de les manipuler sont **getExtra** et **putExtra**.
- Lorsqu'on prépare un Intent et que l'on souhaite ajouter une information de type
- "clef -> valeur" on procède ainsi:

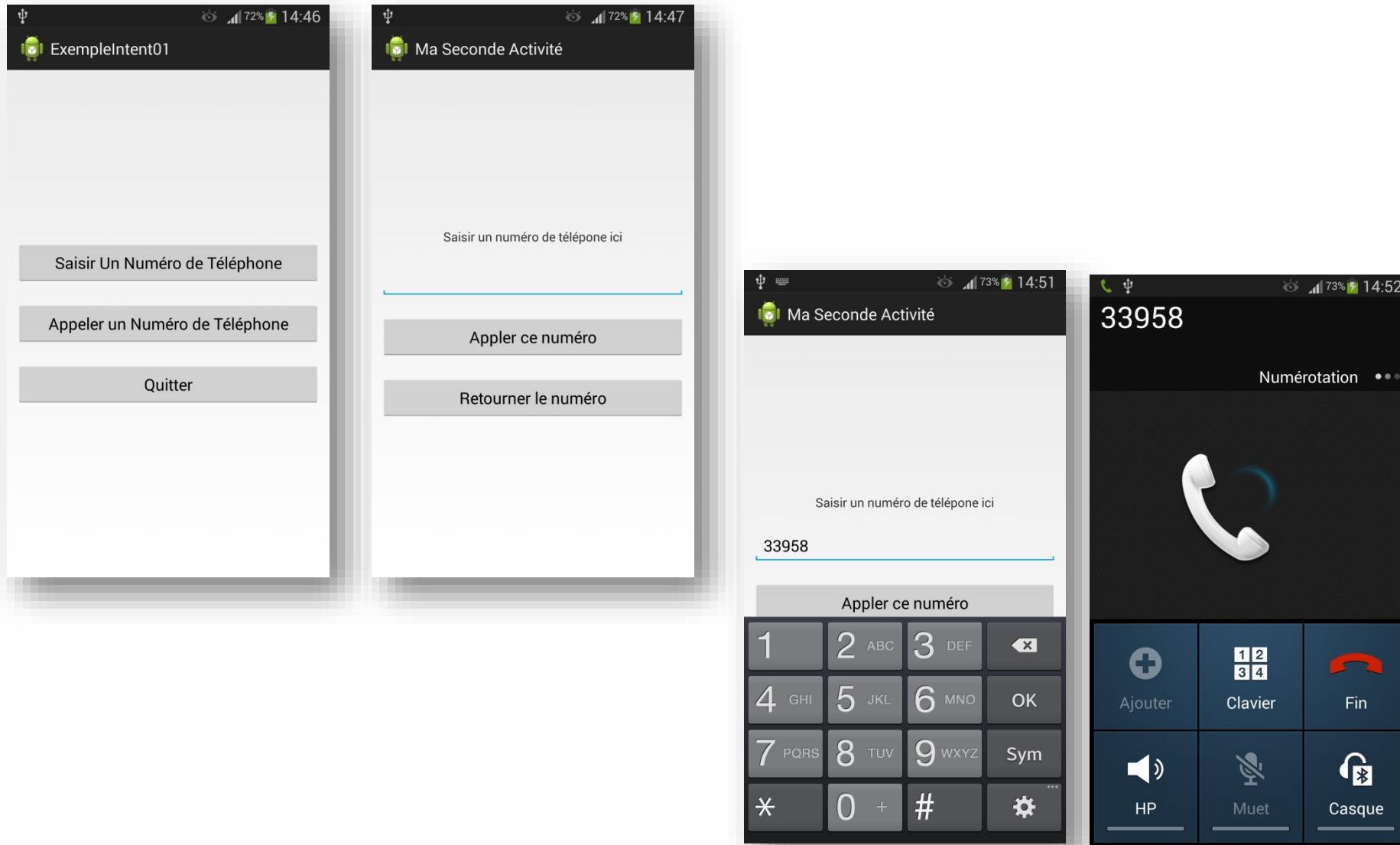
# Ajout d'informations

```
Intent callactivity2 = new Intent(getApplicationContext(), Activity2.class);
callactivity2.putExtra("login", "Lamane");
startActivity(callactivity2);
```

Du côté de l'activité recevant l'Intent, on récupère l'information de la manière suivante:

```
Bundle extras = getIntent().getExtras();
String s = new String(extras.getString("login"));
```

# Exemple



# Pour l'activité principale



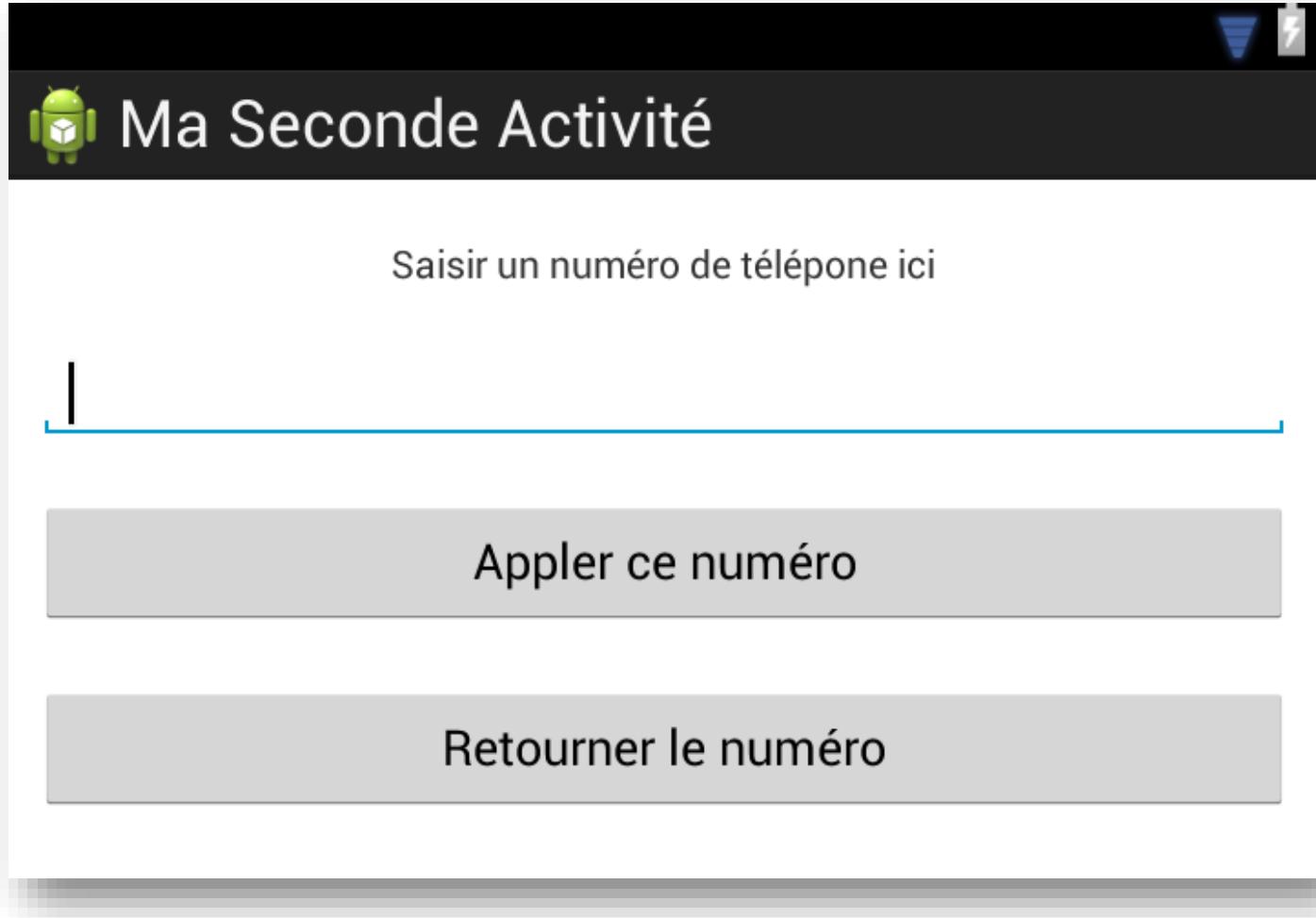
# Pour l'activité principale

```
applerButton = (Button) findViewById(R.id.buttonAppel);
applerButton.setOnClickListener(new OnClickListener() {
    public void onClick(View arg0) {
        Intent appelerIntent = new Intent(getApplicationContext(),
            AppelActivity.class);
        startActivity(appelerIntent);
    }
});
numeroButton = (Button) findViewById(R.id.buttonSaisie);
numeroButton.setOnClickListener(new OnClickListener() {
    public void onClick(View arg0) {
        Intent saisirIntent = new Intent(getApplicationContext(),
            AppelActivity.class);
        startActivityForResult(saisirIntent, 1);
    }
});
quitterButton = (Button) findViewById(R.id.buttonQuitter);
quitterButton.setOnClickListener(new OnClickListener() {
    public void onClick(View arg0) {
        finish();
    }
});
```

# Pour l'activité principale

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (requestCode == 1) {
        if (resultCode == RESULT_OK) {
            String filiere = data.getStringExtra("NumTel");
            Toast.makeText(getApplicationContext(),
                "Votre numéro est " + filiere, Toast.LENGTH_LONG)
                .show();
        } else if (resultCode == RESULT_CANCELED) {
            Toast.makeText(getApplicationContext(), "Opération annulée",
                Toast.LENGTH_LONG).show();
        }
    }
}
```

# Pour la seconde activité



# Pour la seconde activité

```
numTelEditText = (EditText) findViewById(R.id.editText1);
callButton = (Button) findViewById(R.id.buttonCallNumber);
callButton.setOnClickListener(new OnClickListener() {
    public void onClick(View arg0) {
        Uri numTel = Uri.parse("tel:" + numTelEditText.getText());
        Intent appelerIntent = new Intent(Intent.ACTION_CALL, numTel);
        startActivity(appelerIntent);
    }
});
retourButton = (Button) findViewById(R.id.buttonBack);
retourButton.setOnClickListener(new OnClickListener() {
    public void onClick(View arg0) {
        if (numTelEditText.getText().toString().trim().length() > 0) {
            getIntent().putExtra("NumTel", numTelEditText.getText().toString());
            setResult(RESULT_OK, getIntent());
        } else {
            setResult(RESULT_CANCELED, getIntent());
        }
        finish();
    }
});
```

# Intentions prédéfinies

- **ACTION\_MAIN**: action principale
- **ACTION\_VIEW**: visualiser une donnée
- **ACTION\_ATTACH\_DATA**: attachement de donnée
- **ACTION\_EDIT**: Edition de donnée
- **ACTION\_PICK**: Choisir un répertoire de donnée
- **ACTION\_CHOOSER**: menu de choix pour l'utilisateur – EXTRA\_INTENT contient l'Intent original, EXTRA\_TITLE le titre du menu
- **ACTION\_GET\_CONTENT**: obtenir un contenu suivant un type MIME
- **ACTION\_SEND**: envoyé un message (EXTRA\_TEXT|EXTRA\_STREAM) à un destinataire non spécifié

# Intentions prédéfinies

- **ACTION\_SEND\_TO**: on spécifie le destinataire dans l'URI
- **ACTION\_INSERT**: on ajoute un élément vierge dans le répertoire spécifié par l'URI
- **ACTION\_DELETE**: on supprime l'élément désigné par l'URI
- **ACTION\_PICK\_ACTIVITY**: menu de sélection selon l'EXTRA\_INTENT mais ne lance pas l'activité
- **ACTION\_SEARCH**: effectue une recherche etc...

# Intentions prédéfinies

The screenshot displays the Android Studio interface with the following components:

- Top Bar:** Shows project navigation (Nexus One), theme (AppTheme), activity (MainActivity), and build variants (Android 17).
- Left Panel (Preview):** Shows the UI layout for the "ExempleIntentGallerie" application. It features a title bar with the app icon and name, followed by three buttons labeled "Prendre une photo avec la Camera", "Prendre Une Image de la Galerie", and "Jouer un son MP3".
- Right Panel (Runtime Preview):** Shows the application running on a virtual device. The title bar reads "ExempleIntentGallerie". Below it are three buttons with the same labels as the preview. A large image of a woman with long dark hair is displayed at the bottom.
- Outline View:** Located in the top right, it lists the layout structure: LinearLayout1 containing button1, button2, button3, and imageView1.
- Properties View:** Located in the bottom right, it shows properties for the LinearLayout1 component, including Id (@+id/LayoutLinear1), Layout Type (vertical), and Gravity.

```
16 public class MainActivity extends Activity {  
17     Button galerieButton, cameraButton, mp3Button;  
18     ImageView myImageView;  
19  
20     @Override  
21     protected void onCreate(Bundle savedInstanceState) {  
22         super.onCreate(savedInstanceState);  
23         setContentView(R.layout.activity_main);  
24         myImageView = (ImageView) findViewById(R.id.imageView1);  
25         cameraButton = (Button) findViewById(R.id.button1);  
26         cameraButton.setOnClickListener(new OnClickListener() {  
27             @Override  
28             public void onClick(View arg0) {  
29                 Intent intent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);  
30                 startActivityForResult(intent, 1);  
31             }  
32         });  
33         galerieButton = (Button) findViewById(R.id.button2);  
34         galerieButton.setOnClickListener(new OnClickListener() {  
35             @Override  
36             public void onClick(View arg0) {  
37                 startActivityForResult(  
38                     new Intent(Intent.ACTION_PICK,  
39                         android.provider.MediaStore.Images.Media.INTERNAL_CONTENT_URI),2);  
40             }  
41         });  
42         mp3Button = (Button) findViewById(R.id.button3);  
43         mp3Button.setOnClickListener(new OnClickListener() {  
44             @Override  
45             public void onClick(View arg0) {  
46                 Intent intent = new Intent();  
47                 intent.setType("audio/mp3");  
48                 intent.setAction(Intent.ACTION_GET_CONTENT);  
49                 startActivityForResult(Intent.createChooser(intent, "Open Audio (mp3) file"),3);  
50             }  
51         });  
52     }  
}
```

```
Int 54    public void onActivityResult(int requestCode, int resultCode, Intent intent) {  
55        super.onActivityResult(requestCode, resultCode, intent);  
56        switch (requestCode) {  
57            case 1:  
58                if (resultCode == Activity.RESULT_OK) {  
59                    Bitmap bitmapImage = (Bitmap) intent.getExtras().get("data");  
60                    myImageView.setImageBitmap(bitmapImage);  
61                    myImageView.setVisibility(ImageView.VISIBLE);  
62                }  
63                break;  
64  
65            case 2:  
66                if (resultCode == Activity.RESULT_OK) {  
67                    Uri mImageCaptureUri = intent.getData();  
68                    myImageView.setImageURI(mImageCaptureUri);  
69                    myImageView.setVisibility(ImageView.VISIBLE);  
70                }  
71                break;  
72  
73            case 3:  
74                if (resultCode == Activity.RESULT_OK) {  
75                    Uri audioFileUri = intent.getData();  
76                    MediaPlayer myMediaPlayer = MediaPlayer.create(this, audioFileUri);  
77                    myMediaPlayer.start();  
78  
79                    // info.setText(audioFileUri.getPath());  
80                }  
81                break;  
82            }  
83        }  
Prof. Ousmane, Computer Programming, Page 36
```

# Enregistrement et restauration de l'état de l'activité

**Les modifications de configuration invalident la mise en page actuelle ou d'autres ressources de votre activité lorsque l'utilisateur:**

- Fait pivoter l'appareil
- Choisit une langue système différente, donc les paramètres régionaux changent
- Entre en mode multi-fenêtre (à partir d'Android 7)

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help Exemples [C:\Users\ousma\AndroidStudioProjects\Exemples] - ...\\app\\src\\main\\res\\values\\styles.xml [app] - Android Studio

Exemples app src main res values styles.xml

activity\_main.xml x styles.xml x MainActivity.java x

```
5     <item name="colorPrimary">#EE0800</item>
6     <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
7     <item name="colorAccent">@color/colorAccent</item>
8 </style>
9
10 <style name="MonStyle">
11     <item name="android:layout_width">match_parent</item>
12     <item name="android:layout_height">wrap_content</item>
13     <item name="android:layout_margin">10dp</item>
14     <item name="android:gravity">center</item>
15     <item name="android:textColor">#00FF00</item>
16     <item name="android:textSize">30sp</item>
17     <item name="android:typeface">monospace</item>
18 </style>
19
20 <style name="MonStyleFils" parent="MonStyle">
21     <item name="android:textColor">#FF0900</item>
22     <item name="android:textSize">150sp</item>
23     <item name="android:layout_marginBottom">20dp</item>
24 </style>
25 </resources>
```

Project Resource Manager Build Variants Favorites

Hierarchy Gradle Device File Explorer

resources style Prof. Ousmane SALL, Univ. Thiès, SN

Taper ici pour rechercher

Programmation Applications Mobiles

38 08:08 FRA 08/09/2020

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help Exemples [C:\Users\ousma\AndroidStudioProjects\Exemples] - ...app\src\main\res\layout\activity\_main.xml [app] - Android Studio

Exemples app src main res layout activity\_main.xml

activity\_main.xml styles.xml MainActivity.java

Code Split Design

Palette

Common Ab TextView

- Text
- Buttons
- Widgets
- Layouts
- Containers
- Google
- Legacy

Resource Manager

Component Tree

LinearLayout (vertical)

- Ab TextView "compteur"
- Ab textView "0"

Hierarchy

INFINIX MOBILITY LIMITED Infinix X650C

Attributes

Ab <unnamed>

id

Declared Attributes

style @style/MonStyle

text Compteur

Layout

layout\_width match\_parent

layout\_height wrap\_content

layout\_weight

visibility

visibility

Common Attributes

text Compteur

text

contentDescription

textAppearance @android:style/TextAppearance.

alpha

All Attributes

Device File Explorer



File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help Exemples [C:\Users\ousma\AndroidStudioProjects\Exemples] - ...app\src\main\res\layout\activity\_main.xml [app] - Android Studio

Exemples app src main res layout activity\_main.xml styles.xml MainActivity.java

INFINIX MOBILITY LIMITED Infinix X650C

Resource Manager Project I: Project

Palette Common Ab TextView Text Button ImageView Buttons RecyclerView Widgets <> <fragment> Layouts ScrollView Containers Switch Google Legacy

Component Tree LinearLayout (vertical) Ab TextView "Compteur" Ab textView "0"

Code Split Design

Attributes Ab textView id textView Declared Attributes layout\_width match\_parent layout\_height match\_parent id textView onClick compter style @style/MonStyleFils text 0 Layout layout\_width match\_parent layout\_height match\_parent layout\_weight visibility visibility Common Attributes text 0 text contentDescription @android:style/TextAppearance. alpha All Attributes

Prof. Ousmane SALL, Univ. Thiès, SN

Programmation Applications Mobiles

Event Log Layout Inspector 40

Install successfully finished in 2 s 692 ms. (6 minutes ago)

24:13 CRLF UTF-8 4 spaces

```
1 package com.example.exemples;  
2  
3 import ...  
4  
5  
6 public class MainActivity extends AppCompatActivity {  
7     int val=0;  
8     TextView t;  
9     @Override  
10    protected void onCreate(Bundle savedInstanceState) {  
11        super.onCreate(savedInstanceState);  
12        setContentView(R.layout.activity_main);  
13        t=findViewById(R.id.textView);  
14    }  
15  
16    public void compter(View v){  
17        t.setText(""+val++);  
18    }  
19  
20}  
21  
22  
23  
24  
25}
```

Project

Resource Manager

Build Variants

Favorites

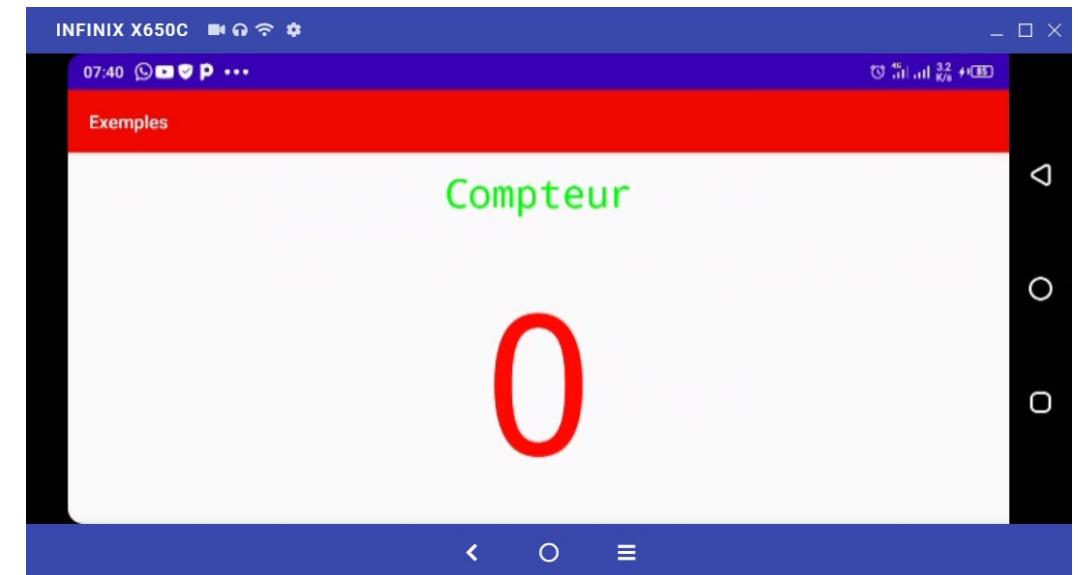
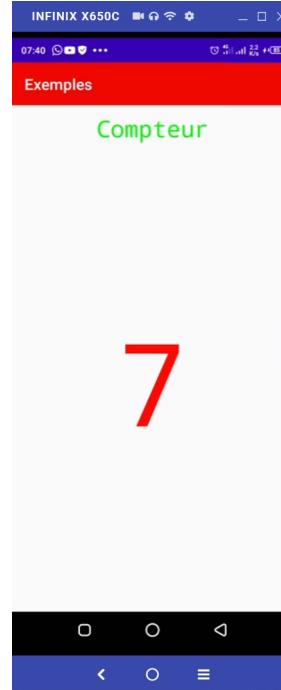
Hierarchy  
Gradle

Device File Explorer

# Enregistrement et restauration de l'état de l'activité

**Les modifications de configuration invalident la mise en page actuelle ou d'autres ressources de votre activité lorsque l'utilisateur:**

- Fait pivoter l'appareil
- Choisit une langue système différente, donc les paramètres régionaux changent
- Entre en mode multi-fenêtre (à partir d'Android 7)



# Enregistrement de l'état de l'instance avec onSaveInstanceState(Bundle outState)

Implémentez onSaveInstanceState () dans votre activité

- Appelé par l'environnement d'exécution Android lorsqu'il existe une possibilité que l'activité soit détruite
- Enregistre les données uniquement pour cette instance de l'activité pendant la session en cours

```
@Override
protected void onSaveInstanceState(@NonNull Bundle outState) {
    super.onSaveInstanceState(outState);
    outState.putString("compteur", String.valueOf(t.getText()));
}
```

# Restauration de l'état de l'instance avec onRestoreInstanceState(Bundle mySavedState)

```
@Override
protected void onRestoreInstanceState(@NonNull Bundle savedInstanceState) {
    super.onRestoreInstanceState(savedInstanceState);
    if (savedInstanceState != null) {
        String compteur = savedInstanceState.getString( key: "compteur" );
        if (compteur != null)
            t.setText(compteur);
    }
}
```

Exemples app src main java com example exemples MainActivity

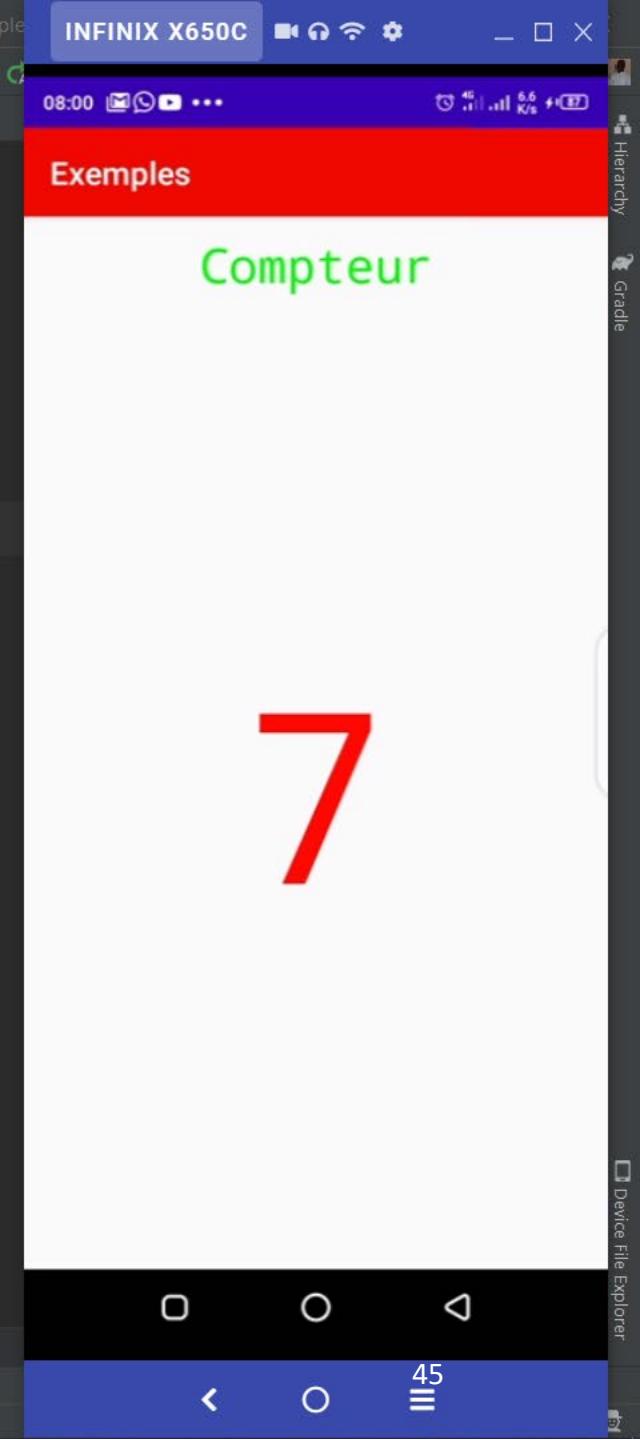
activity\_main.xml styles.xml MainActivity.java

```
public void compter(View v){  
    t.setText(""+val++);  
}  
  
@Override  
protected void onSaveInstanceState(@NotNull Bundle outState) {  
    super.onSaveInstanceState(outState);  
    outState.putString("compteur", String.valueOf(t.getText()));  
}  
  
@Override  
protected void onRestoreInstanceState(@NotNull Bundle savedInstanceState) {  
    super.onRestoreInstanceState(savedInstanceState);  
    if (savedInstanceState != null) {  
        String compteur = savedInstanceState.getString(key: "compteur");  
        if (compteur != null)  
            t.setText(compteur);  
    }  
}
```

MainActivity > onSaveInstanceState()  
Prof. Ousmane SALL, Univ. Thiès, SN

Programma

Install successfully finished in 2 s 234 ms. (5 minutes ago)



# État de l'instance et redémarrage de l'application

- Lorsque vous arrêtez et redémarrez une nouvelle session d'application, les états de l'instance d'activité sont perdus et vos activités retrouveront leur apparence par défaut
- Si vous devez enregistrer les données utilisateur entre les sessions d'application, utilisez les préférences partagées ou une base de données.

# Modifiez l'orientation de l'écran

```
public void setRequestedOrientation (int requestedOrientation)
```

- Si l'activité est actuellement au premier plan ou a un impact sur l'orientation de l'écran, l'écran sera immédiatement modifié (provoquant éventuellement le redémarrage de l'activité). Sinon, il sera utilisé la prochaine fois que l'activité sera visible.
- ActivityInfo.SCREEN\_ORIENTATION\_UNSPECIFIED,  
ActivityInfo.SCREEN\_ORIENTATION\_Landscape, ActivityInfo.SCREEN\_ORIENTATION\_PORTRAIT,  
ActivityInfo.SCREEN\_ORIENTATION\_USER, ActivityInfo.SCREEN\_ORIENTATION\_BEHIND,  
ActivityInfo.SCREEN\_ORIENTATION\_SENSOR, ActivityInfo.SCREEN\_ORIENTATION\_NOSENSOR,  
ActivityInfo.SCREEN\_ORIENTATION\_SENSOR\_Landscape,  
ActivityInfo.SCREEN\_ORIENTATION\_SENSOR\_Portrait,  
ActivityInfo.SCREEN\_ORIENTATION\_Reverse\_Landscape,  
ActivityInfo.SCREEN\_ORIENTATION\_Reverse\_Portrait,  
ActivityInfo.SCREEN\_ORIENTATION\_Full\_Sensor,  
ActivityInfo.SCREEN\_ORIENTATION\_USER\_Landscape,  
ActivityInfo.SCREEN\_ORIENTATION\_USER\_Portrait,  
ActivityInfo.SCREEN\_ORIENTATION\_Full\_User, or ActivityInfo.SCREEN\_ORIENTATION\_LOCKED
- `setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_Reverse_Portrait);`

# Développement d'Applications Natives avec Android

## ANDROID- ÉLÉMENTS **STYLE**: POLICES DE CARACTÈRES, **STYLES** ET THÈMES

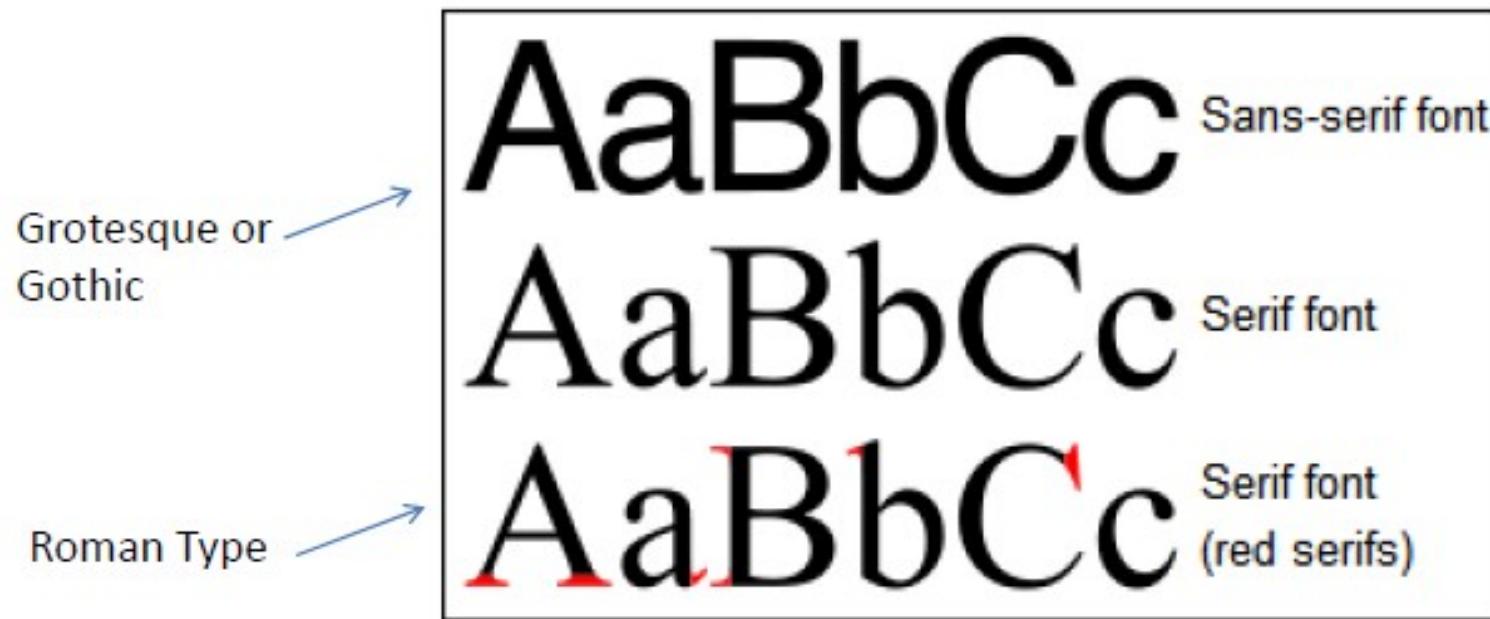


# LES FONTS

# Les Fonts

- Android propose naturellement trois familles de fonts:
  - **sans**: Une étudiante fantôme en LGI
  - **serif**: Une étudiante fantôme en LGI
  - **monospaced**: Une étudiante fantôme

# Fonts



Source: <http://en.wikipedia.org/wiki/Serif>

MyApplicationtest app src main res layout activity\_main.xml

Sync Now

Code Split Design

Attributes

button2

- fadingEdge
- fadingEdgeLength
- filterTouchesWhenObscured
- fitsSystemWindows
- focusable
- focusableInTouchMode
- fontFamily @font/allura
- foreground
- foregroundGravity
- freezesText
- gravity center\_vertical|center
- hapticFeedbackEnabled
- height
- hint
- id button2
- imeActionId
- imeActionLabel
- imeOptions
- importantForAccessibility
- includeFontPadding
- inputMethod
- inputType
- isScrollContainer
- keepScreenOn
- layerType

Pick a Resource

Module: app

Font Resource File More Fonts...

app (2)

Resources

Regular

Font

Font

Regular

allura

Font

Fonts

Downloadable

- A BeeZee
- Abel
- Abhaya Libre
- Abril Fatface
- Aclonica
- Acme
- Actor
- Adamina
- Advent Pro
- Aquafina Script
- Akronim
- Allura

Font Name:

Preview

Nothing to show

OK Cancel

These fonts are available under the [Apache License Version 2.0](#) or [Open Font License](#)

Terminal Build 6: Logcat Profiler 4: Run TODO

Add font resources to project

Event Log Layout Inspector

26:61 CRLF UTF-8 4 spaces

MyApplicationtest app src main res layout activity\_main.xml

Gradle files have changed since last project sync. A project sync may be necessary for the IDE to work properly.

Sync Now

Attributes

button2

fadingEdge

fadingEdgeLength

filterTouchesWhenObscured

fitsSystemWindows

focusable

focusableInTouchMode

fontFamily

@font/allura

foreground

serif

foregroundGravity

monospace

freezesText

serif-monospace

gravity

casual

hapticFeedback

cursive

height

sans-serif-smallcaps

hint

More Fonts...

id

button2

imeActionId

imeActionLabel

imeOptions

importantForAccessibility

includeFontPadding

inputMethod

inputType

isScrollContainer

keepScreenOn

layerType

OK Cancel

Palette

Common

Text

Buttons

Widgets

Layouts

Containers

Resources

Google

Legacy

Fonts

Source Google Fonts

Font Name: sans-serif-condensed-medium

Preview

- ✓ Condensed Regular
- Condensed Regular Italic
- Condensed Bold
- Condensed Bold Italic

Component Tree

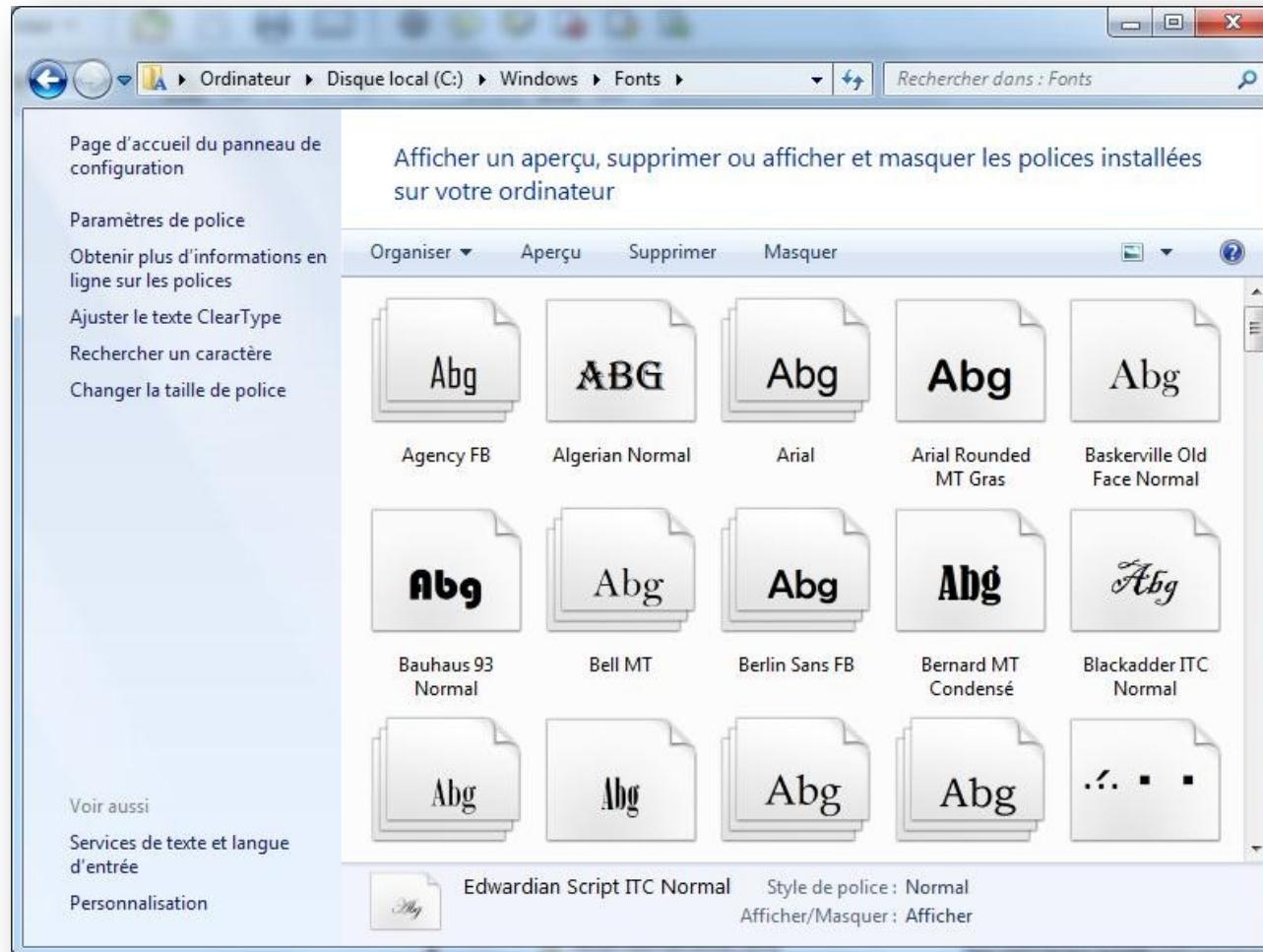
LinearLayout

button

button2

This screenshot shows the Android Studio interface for editing an XML layout file. The top navigation bar includes File, Edit, View, Navigate, Code, Analyze, Refactor, Build, Run, Tools, VCS, Window, Help, and a specific tab for 'My Application test'. Below the navigation bar, the project structure is visible with 'MyApplicationtest' as the root, followed by app, src, main, res, layout, and activity\_main.xml. The main workspace shows the XML code for 'activity\_main.xml' and other files like build.gradle, MainActivity.java, and AndroidManifest.xml. A message at the top of the workspace indicates that Gradle files have changed since the last sync, suggesting a project sync might be necessary. The central part of the screen displays the XML editor with a palette on the left containing various UI components like TextView, Button, ImageView, RecyclerView, etc. A font selection dialog is open over the XML code, showing a list of Google Fonts under the 'Google Fonts' source. The font 'sans-serif-condensed-medium' is selected both in the list and in the preview area. The right side of the screen shows the properties of a selected button, specifically 'button2', with attributes like 'fontFamily' set to '@font/allura', 'foreground' set to 'serif', and 'background' set to 'white'. The bottom of the screen features a toolbar with icons for Terminal, Build, Logcat, Profiler, Run, TODO, Event Log, and Layout Inspector, along with a status message about a successful install.

# Les autres Fonts(Windows)



The screenshot shows the Android Studio interface with the project 'ExempleaveclesListes' open. The 'activity\_main.xml' file is selected in the top navigation bar.

A context menu is open over the 'res' folder in the Project tool window. The 'New' option is highlighted, and the 'Android Resource Directory' item is selected from the submenu.

A 'New Resource Directory' dialog is displayed. The 'Directory name:' field contains 'font'. The 'Resource type:' dropdown is set to 'font'. The 'Source set:' dropdown is set to 'main'. The 'Available qualifiers:' section lists various qualifiers like 'Country Code', 'Network Code', 'Locale', etc. The 'Chosen qualifiers:' section is currently empty, showing 'Nothing to show'. At the bottom right of the dialog are 'OK', 'Cancel', and 'Help' buttons.

The screenshot shows the Android Studio interface with the project "ExempleaveclesListes" open. The left sidebar displays the project structure, showing the "font" folder selected in the "res" directory. The main area shows the XML code for "activity\_main.xml". A floating "Fonts" palette is open, listing various installed fonts. The font "Jokerman" is currently selected.

**Project**

- app
  - manifests
  - java
    - com.example.exempleavecleslistes (an)
    - com.example.exempleavecleslistes (te)
  - generatedJava
  - res
    - drawable
    - font**
    - layout
    - mipmap
    - values
      - arrays.xml
      - colors.xml
      - strings.xml
      - styles.xml
- 2: Favorites
- Build Variants
- Layout Captures

**activity\_main.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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"
    android:layout_margin="16dp"
    android:orientation="vertical"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Choix ?" />

    <GridView
        android:id="@+id/listeEtudian"
        android:layout_width="match_j"
        android:layout_height="wrap_c
        android:addStatesFromChildren="true"
        android:choiceMode="singleCh
        android:numColumns="2" />

    <Button
        android:id="@+id/button2"
        android:layout_width="match_j
        android:layout_height="wrap_c
        android:onClick="validerChois
        android:text="Valider votr" />

</LinearLayout>
```

**Fonts**

Page d'accueil du panneau de configuration

Paramètres de police

Obtenir plus d'informations en ligne sur les polices

Ajuster le texte ClearType

Rechercher un caractère

Télécharger les polices de toutes les langues

Organiser Aperçu Supprimer Masquer

Shadow Normal Normal

Ink Free Normal Javanese Text Normal Jokerman Normal

Juice ITC Normal Kristen ITC Normal Kunstler Script Normal

Jokerman Normal

Voir aussi Services de texte et langue d'entrée

Device File Explorer

1: Project

2: Structure

3: Favorites

4: Build Variants

5: Layout Captures

6: Logcat

7: TODO

8: Terminal

9: Build

10: Profiler

11: Run

12: Event Log

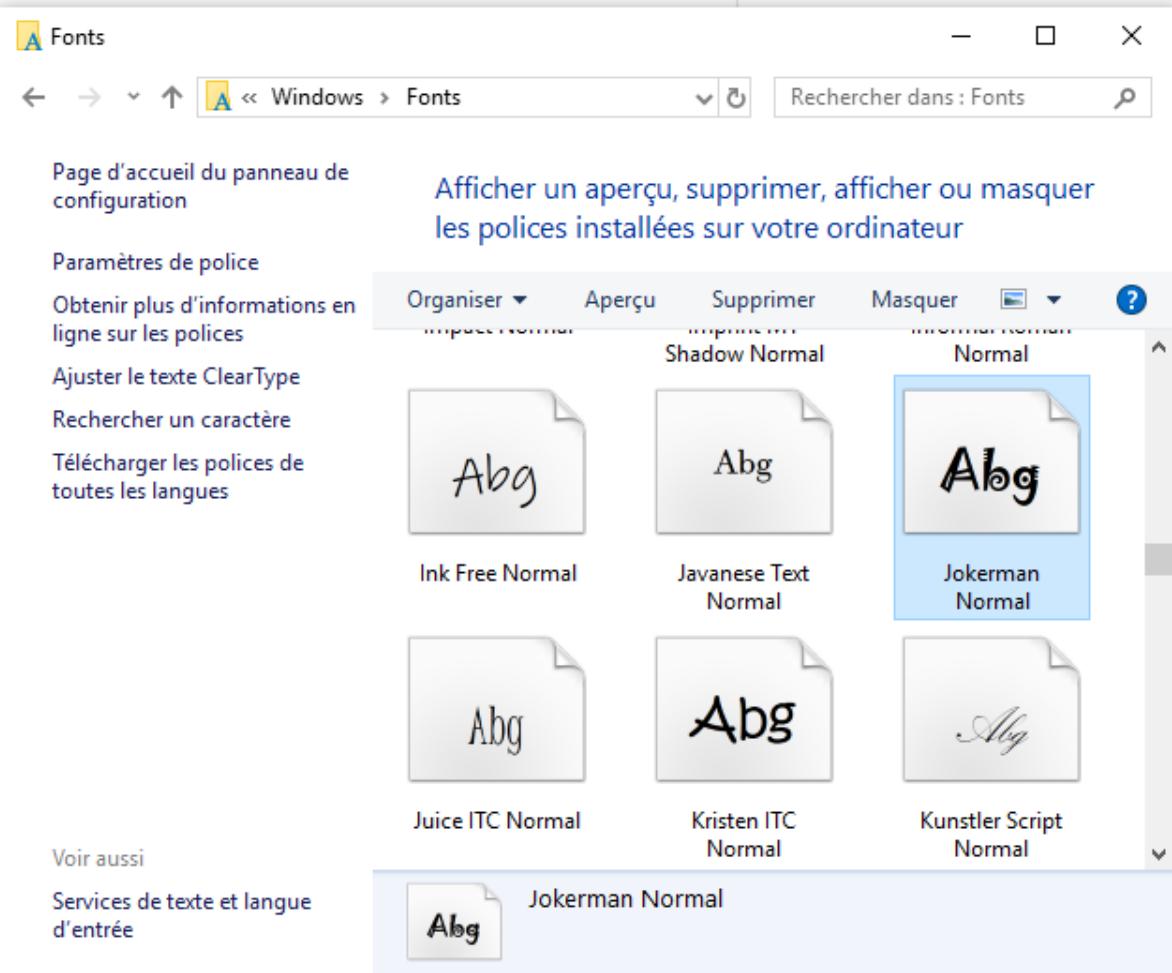
13: app

14: activity\_main.xml

15: arrays.xml

16: MainActivity.java

```
1 package com.example.exempleavecleslistes;
2
3 import ...
4
5 public class MainActivity extends AppCompatActivity {
6     GridView liste;
7     String[] etudiants;
8
9     @Override
10    protected void onCreate(Bundle savedInstanceState) {
11        super.onCreate(savedInstanceState);
12        setContentView(R.layout.activity_main);
13        liste = (GridView) findViewById(R.id.gridView1);
14
15        Button b=(Button)findViewById(R.id.button1);
16
17        etudiants = getResources().getStringArray(R.array.array);
18        ArrayAdapter<String> adapteur = new ArrayAdapter<String>(this, android.R.layout.simple_list_item_1, etudiants);
19        for (int i = 0; i < etudiants.length; i++) {
20            adapteur.add(etudiants[i]);
21        }
22        liste.setAdapter(adapteur);
23
24    }
25
26    public void validerChoix(View v) {
27        String choix="Vous avez choisi "+etudiants[0];
28        Toast.makeText(context, choix, Toast.LENGTH_SHORT).show();
29    }
30
31 }
32
33
34 }
```



1: Project

2: Favorites

3: Build Variants

4: Layout Captures

5: Device File Explorer

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

activity\_main.xml arrays.xml MainActivity.java jokerman.ttf

Android

app

- manifests
- java
  - com.example.exempleavecleslistes
    - MainActivity
- generatedJava
- res
  - drawable
  - font
    - jokerman.ttf
  - layout
  - mipmap
  - values
    - arrays.xml
    - colors.xml
    - strings.xml
    - styles.xml

- Gradle Scripts

Jokerman

**LOREM IPSUM DOLOR SIT AMET, CONSECTETUR ADIPISCING ELIT. Ad aut cum,  
est exercitationem explicabo facilis illum itaque iure, iusto possimus  
quae qui quod quos sed, velit? Exercitationem harum inventore ipsa  
magnam magni quae quos. Alias cum dolor eveniet fuga molestiae  
molestias odit rerum voluptatum. Adipisci placeat quis saepe ullam  
voluptates.**

6: Logcat 7: TODO 8: Terminal 9: Build 10: Profiler 11: Run 12: Event Log

1: Project

2: Favorites

3: Build Variants

4: Layout Captures

5: Device File Explorer

6: Logcat

7: TODO

8: Terminal

9: Build

10: Profiler

11: Run

12: Event Log

13: Instant Run applied code changes and restarted the app. R Class Change. // (Don't show again) (2 minutes ago)

14: Context: <no context>

15: Smiley

16: Sad

17: Neutral

18: Project

19: activity\_main.xml

20: arrays.xml

21: MainActivity.java

22: jokerman.ttf

23: Pixel

24: 28

25: AppTheme

26: 100%

27: Attributes

28: editable

29: editorExtras

30: ellipsis

31: ems

32: enabled

33: fadeScrollbars

34: fadingEdge

35: fadingEdgeLength

36: filterTouchesWhenC

37: firstBaselineToTopH

38: fitsSystemWindows

39: focusable

40: focusableInTouchM

41: fontFamily @font/jokerman

42: foreground

43: foregroundGravity

44: freezesText

45: gravity

46: hapticFeedbackEnab

47: height

48: hint

49: imeActionId

50: imeActionLabel

51: imeOptions

52: includeFontPadding

53: inputMethod

54: inputType

55: isScrollContainer

56: Component Tree

57: LinearLayout(vertical)

58: textView- "Choix ?"

59: listeEtudiants

60: button2- "Valider vot..."

61: Design

62: Text

63: Choix ?

64: Item 1

65: Sub Item 1

66: Item 3

67: Sub Item 3

68: Palettes

69: ConstraintL...

70: Guideline (...)

71: Guideline (...)

72: LinearLayo...

73: LinearLayo...

74: FrameLayout

75: TableLayout

76: TableRow

77: Space

78: Layouts

79: Containers

80: Google

81: Legacy



1: Project

```

    package com.example.exempleavecleslistes;
    import ...
    public class MainActivity extends AppCompatActivity {
        GridView liste;
        String[] etudiants;
        @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main);
            liste = (GridView) findViewById(R.id.listeEtudiants);

            Button b=(Button)findViewById(R.id.button2);

            Typeface typeface = ResourcesCompat.getFont(context: this, R.font.jokerman);
            b.setTypeface(typeface);

            etudiants = getResources().getStringArray(R.array.etudiants);
            ArrayAdapter<String> adapteur = new ArrayAdapter<String>( context: this,
                android.R.layout.simple_list_item_single_choice);
            for (int i = 0; i < etudiants.length; i++)
                adapteur.add(etudiants[i]);
            liste.setAdapter(adapteur);
        }
        public void validerChoix(View v) {
            String choix="Vous avez choisi "+etudiants[liste.getSelectedItemPosition()];
            Toast.makeText( context: this, choix, Toast.LENGTH_LONG).show();
        }
    }

```

MainActivity > onCreate()



# STYLES

# Qu'est-ce qu'un style?

- Collection d'attributs qui définissent l'apparence visuelle d'une vue
- Réduire la duplication de code relatif au style
- Rendre le code plus compact
- Gérer l'apparence visuelle de nombreux composants avec un seul style

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help Exemples [C:\Users\ousma\AndroidStudioProjects\Exemples] - ...app\src\main\res\layout\activity\_main.xml [app] - Android Studio

Exemples app src main res layout activity\_main.xml

activity\_main.xml MainActivity.java

Resource Manager

1: Project

2: Favorites

Build Variants

Z: Structure

Code Split Design activity\_main.xml Custom

Layout Validation

INFINIX MOBILITY LIMITED Infinix X650C

Default (Current File)

Hello World!

3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20

```
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:orientation="vertical"
tools:context=".MainActivity">>

<TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="10dp"
    android:gravity="center"
    android:text="Hello World!"
    android:textColor="#00FF00"
    android:textSize="30sp"
    android:typeface="monospace" />

</LinearLayout>
```

LinearLayout > TextView

Prof. Ousmane SALL, Univ. Thiès, SN

Terminal Build Log Profiler Run TODO

Programmation Applications Mobiles

Event Log Layout Inspector

63

Install successfully finished in 6 s 143 ms. (4 minutes ago)

261 chars, 6 line breaks 17:32 CRLF UTF-8 4 spaces

Exemples app src main res values styles.xml

activity\_main.xml styles.xml MainActivity.java

Andr. Gradle

Resource Manager 1: Project

Gradle Scripts

1: Favorites

Build Variants

Z: Structure

Device File Explorer

```
<resources>
    <!-- Base application theme. -->
    <style name="AppTheme" parent="Theme.AppCompat.Light.DarkActionBar">
        <!-- Customize your theme here. -->
        <item name="colorPrimary">@color/colorPrimary</item>
        <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
        <item name="colorAccent">@color/colorAccent</item>
    </style>

    <style name="MonStyle">
        <item name="android:layout_width">match_parent</item>
        <item name="android:layout_height">wrap_content</item>
        <item name="android:layout_margin">10dp</item>
        <item name="android:gravity">center</item>
        <item name="android:textColor">#00FF00</item>
        <item name="android:textSize">30sp</item>
        <item name="android:typeface">monospace</item>
    </style>
</resources>
```

Définir des styles dans **res/values/styles.xml**

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help Exemples [C:\Users\ousma\AndroidStudioProjects\Exemples] - ...app\src\main\res\layout\activity\_main.xml [app] - Android Studio

Exemples app src main res layout activity\_main.xml app INFINIX MOBILITY LIMITED Infinix X650C

Andr. activity\_main.xml styles.xml MainActivity.java Layout Validation

Resource Manager

1: Project

1: Favorites

Build Variants

Structure

Layout Validation

Code Split Design activity\_main.xml Custom

Default (Current File)

Hello World!

TextView

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

    <TextView
        style="@style/MonStyle"
        android:text="Hello World!"/>

    <TextView
        android:id="@+id/textView"
        style="@style/MonStyle"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="TextView" />

</LinearLayout>
```

Device File Explorer

Terminal Build Log Profiler Run TODO Event Log Layout Inspector

Prof. Ousmane SALL, Univ. Thiès, SN

Programmation Applications Mobiles

8 chars 11:31 CRLF UTF-8 4 spaces

Install successfully finished in 3 s 114 ms. (7 minutes ago)

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help Exemples [C:\Users\ousma\AndroidStudioProjects\Exemples] - ...app\src\main\res\layout\activity\_main.xml [app] - Android Studio

Exemples app src main res layout activity\_main.xml app INFINIX MOBILITY LIMITED Infinix X650C

Andr. Activity styles.xml MainActivity.java

Resource Manager

Project

1: Project

Gradle Scripts

Component Tree

LinearLayout (vertical)

Ab TextView "Hello World!"

Ab textView "TextView"

Code Split Design

Palette

Common Ab TextView

Text Button

Buttons ImageView

Widgets RecyclerView

Layouts ScrollView

Containers Switch

Google

Legacy

Attributes

Ab textView

scrollHorizontally

scrollX

scrollY

scrollbarAlwaysD...

scrollbarAlwaysD...

scrollbarDefaultD... 400

scrollbarFadeDur... 250

scrollbarSize 4dp

scrollbarStyle

scrollbarThumbH... @android:drawable/s

scrollbarThumbV... @android:drawable/s

scrollbarTrackHo... @null

scrollbarTrackVer... @android:drawable/c

scrollbars

selectAllOnFocus

shadowColor

shadowDx

shadowDy

shadowRadius

singleLine

soundEffectsEna...

style @style/MonStyle

tag

targetApi

text TextView

textAllCaps

Device File Explorer

Terminal Build Log Profiler Run TODO

Prof. Ousmane SALL, Univ. Thiès, SN

Programmation Applications Mobiles

8 chars 11:31 CRLF UTF-8 4 spaces

66 Event Log Layout Inspector

Install successfully finished in 3 s 114 ms. (8 minutes ago)

Exemples > app > src > main > res > values > styles.xml

activity\_main.xml > styles.xml > MainActivity.java

INFINIX MOBILITY LIMITED Infinix X650C

```
<item name="colorPrimaryDark">@color/colorPrimaryDark</item>
<item name="colorAccent">@color/colorAccent</item>
</style>

<style name="MonStyle">
    <item name="android:layout_width">match_parent</item>
    <item name="android:layout_height">wrap_content</item>
    <item name="android:layout_margin">10dp</item>
    <item name="android:gravity">center</item>
    <item name="android:textColor">#00FF00</item>
    <item name="android:textSize">30sp</item>
    <item name="android:typeface">monospace</item>
</style>

<style name="MonStyleFils" parent="MonStyle">
    <item name="android:textColor">#FF0900</item>
</style>
</resources>
```

Exemples > app > src > main > res > layout > activity\_main.xml

activity\_main.xml styles.xml MainActivity.java

Andr. Resource Manager Code Split Design Grade

Palette Pixel 30 AppTheme Default (en-us) Attributes

Common Ab TextView

Text Button

Buttons ImageView

Widgets RecyclerView

Layouts ScrollView

Containers Switch

Google

Legacy

Component Tree

LinearLayout (vertical)

Ab TextView "Hello World!"

Ab textView "TextView"

Pick a Resource

Module: app

Style

app (3)

AppTheme

Style | 1 version

MonStyle

Style | 1 version

MonStyleFils

Style | 1 version

core-1.3.1.aar (5)

Preview

Name: MonStyleFils

Reference: @style/MonStyleFils

Configuration: default

Value: No value

OK Cancel

Device File Explorer

116 chars, 2 line breaks 20:1 CRLF UTF-8 4 spaces

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help Exemples [C:\Users\ousma\AndroidStudioProjects\Exemples] - ...app\src\main\res\layout\activity\_main.xml [app] - Android Studio

Exemples app src main res layout activity\_main.xml app INFINIX MOBILITY LIMITED Infinix X650C

Andr. activity\_main.xml styles.xml MainActivity.java

Resource Manager

Project

1: Project

Gradle Scripts

Component Tree

LinearLayout (vertical)

Ab TextView "Hello World!"

Ab textView "TextView"

Palette

Common Ab TextView

Text Button

Buttons ImageView

Widgets RecyclerView

Layouts ScrollView

Containers Switch

Google

Legacy

Attributes

Default (en-us)

Code Split Design

Layout Validation

Hello World!

TextView

scrollHorizontally

scrollX

scrollY

scrollbarAlwaysDra...

scrollbarAlwaysDra...

scrollbarDefaultDel...

scrollbarFadeDurati...

scrollbarSize

scrollbarStyle

scrollbarThumbHori...

scrollbarThumbVert...

scrollbarTrackHoriz...

scrollbarTrackVertical

scrollbars

selectAllOnFocus

shadowColor

shadowDx

shadowDy

shadowRadius

singleLine

text

textAllCaps

style

style

tag

targetApi

text

textAllCaps

Device File Explorer

Terminal Build Log Profiler Run TODO

Prof. Ousmane SALL, Univ. Thiès, SN

Programmation Applications Mobiles

69 Event Log Layout Inspector

Install successfully finished in 3 s 114 ms. (13 minutes ago)

16:1 CRLF UTF-8 4 spaces

# THEMES

Exemples > app > src > main > AndroidManifest.xml

activity\_main.xml styles.xml AndroidManifest.xml MainActivity.java

Android Manager -

Resource Manager

I: Project

Gradle Scripts

Build Variants

Build Structure

Z: Favorites

1: Project

Android app manifests AndroidManifest.xml

java java (generated)

res drawable layout mipmap values

colors.xml strings.xml styles.xml

Gradle Scripts

1 <?xml version="1.0" encoding="utf-8"?>

2 <manifest xmlns:android="http://schemas.android.com/apk/res/android"

3 package="com.example.exemples">

4

5 <application>

6 android:allowBackup="true"

7 android:icon="@mipmap/ic\_launcher"

8 android:label="Exemples"

9 android:roundIcon="@mipmap/ic\_launcher\_round"

10 android:supportsRtl="true"

11 android:theme="@style/AppTheme">

12 <activity android:name=".MainActivity">

13 <intent-filter>

14 <action android:name="android.intent.action.MAIN" />

15

16 <category android:name="android.intent.category.LAUNCHER" />

17 </intent-filter>

18 </activity>

19 </application>

20

21 </manifest>

manifest > application > activity > intent-filter > category

Text Merged Manifest

Terminal Build 6: Logcat Profiler 4: Run TODO

Event Log Layout Inspector

Install successfully finished in 1 s 822 ms. (3 minutes ago)

16:40 CRLF UTF-8 4 spaces 😊 😐 😕

Un thème est un style appliqué à une activité entière ou même à l'application entière  
Les thèmes sont appliqués dans AndroidManifest.xml

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help Exemples [C:\Users\ousma\AndroidStudioProjects\Exemples] - ...\\app\\src\\main\\res\\values\\styles.xml [app] - Ar

Exemples app src main res values styles.xml

Android Manager

Resource Manager

Project

Gradle Scripts

Favorites

Build Variants

Structure

Terminal Build Logcat Profiler Run TODO

Install successfully finished in 1 s 822 ms. (4 minutes ago)

activity\_main.xml styles.xml AndroidManifest.xml MainActivity.java

```
1 <resources>
2     
3     <style name="AppTheme" parent="Theme.AppCompat.Light.DarkActionBar">
4         
5         <item name="colorPrimary">#EE0800</item>
6         <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
7         <item name="colorAccent">@color/colorAccent</item>
8     </style>
9
10    <style name="MonStyle">
11        <item name="android:layout_width">match_parent</item>
12        <item name="android:layout_height">wrap_content</item>
13        <item name="android:layout_margin">10dp</item>
14        <item name="android:gravity">center</item>
15        <item name="android:textColor">#00FF00</item>
16        <item name="android:textSize">30sp</item>
17        <item name="android:typeface">monospace</item>
18    </style>
19
20    <style name="MonStyleFils" parent="MonStyle">
21        <item name="android:textColor">#FF0900</item>
22    </style>

```

resources > style > item



# Développement d'Applications Natives avec Android

**ANDROID- ÉLÉMENTS  
D'INTERACTION:  
BOITES DE DIALOGUE,  
NOTIFICATIONS, MENUS,  
SÉLECTION DATES ET HEURES**



# Boite de dialogue: La classe Toast

- Texte qui apparaît en premier plan puis disparaît au bout d'un temps donné
- Crédit d'un Toast
  - `Toast.makeText(Context, String, int)` renvoie l'objet de classe Toast créé.
  - Le premier paramètre est l'activité
  - Le deuxième paramètre est le message à afficher
  - Le dernier paramètre indique la durée d'affichage les seules valeurs possibles sont : `Toast.LENGTH_SHORT` (2 secondes) ou `Toast.LENGTH_LONG` (5 secondes).
- Positionnement d'un Toast
  - `setGravity(int, int, int)` appelée avant l'affichage par `show` pour indiquer où s'affichera le message.
  - Le premier paramètre sert à placer le message par rapport à l'écran. Il peut prendre l'une des valeurs définies dans la classe Gravity soit : `Gravity`. (`TOP`, `BOTTOM`, `LEFT`, `RIGHT`, `CENTER_VERTICAL`, `FILL_VERTICAL`, `CENTER_HORIZONTAL`, `FILL_HORIZONTAL`, `CENTER`, `FILL`).
  - Les deux paramètres suivants indiquent le décalage (en pixels).
- Affichage d'un Toast
  - `show()` affiche le message pour la durée définie lors de sa création.

Répertoire illisible

# Boite de dialogue: Exemple de Toast

- Un Toast est un message qui apparaît et disparaît
- Il est impossible de savoir si l'utilisateur l'a vu ou pas
- Configuration:
  - Un String contenant le message
  - Une durée
    - Toast.LENGTH\_LONG ou Toast.LENGTH\_SHORT
- Il est aussi possible de donner une View de votre choix en paramètre



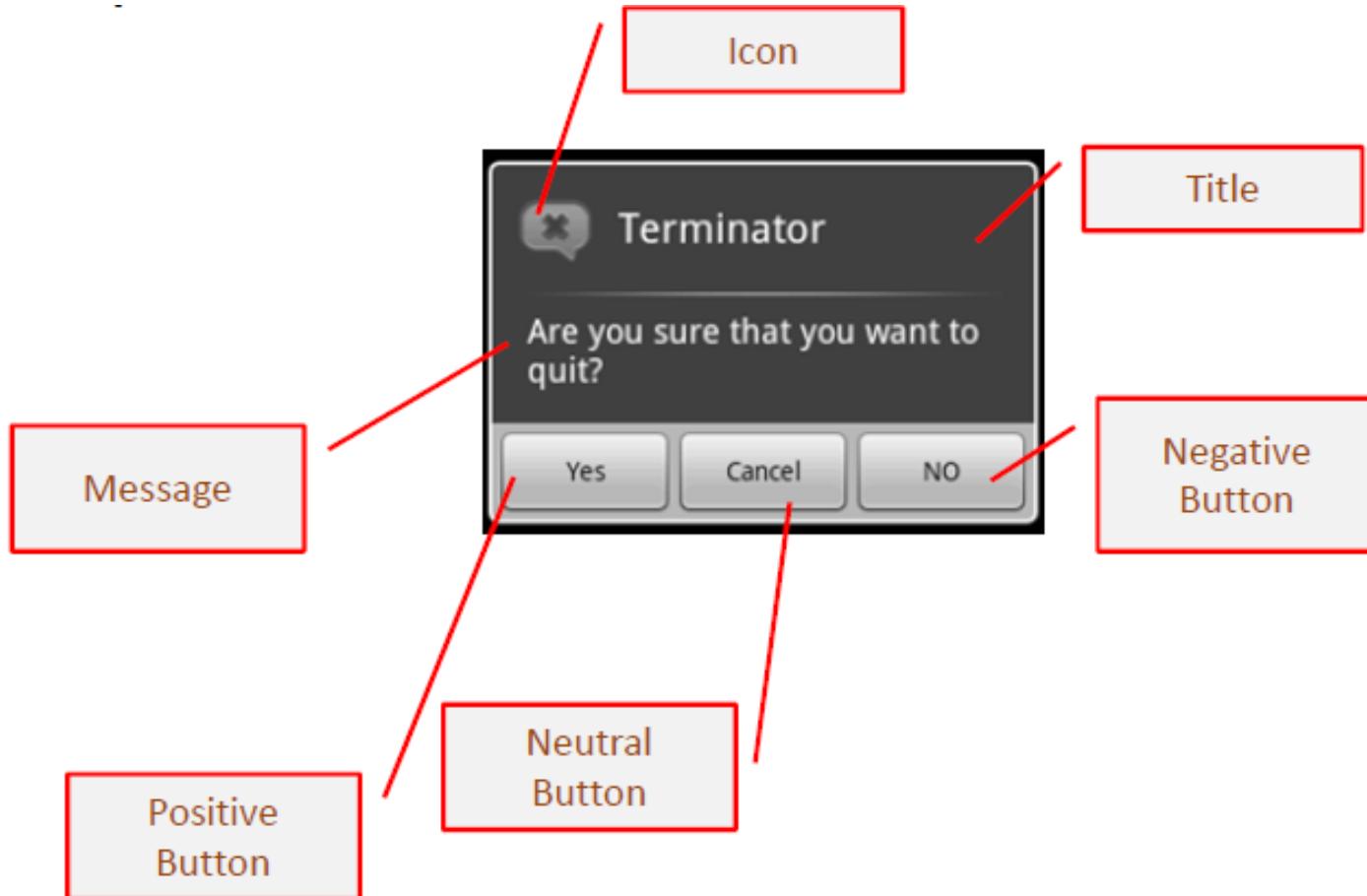
# Toast : exemple

```
Toast.makeText(MainActivity.this,  
        "C'est vous qui voyez !!!",  
        Toast.LENGTH_SHORT).show();
```

# Boite de dialogue: La classe AlertDialog

- Style plus classique des boîtes de dialogue. Un AlertDialog s'ouvre, prend le focus et reste affiché tant que l'utilisateur ne le ferme pas
- Pour créer un AlertDialog:
  - Utiliser la classe Builder offrant un ensemble de méthodes permettant de configurer un AlertDialog. Méthodes renvoie le Builder afin de faciliter le chaînage des appels.
  - À la fin, il suffit d'appeler la méthode show() de l'objet Builder pour afficher la boîte de dialogue.
- Méthodes de configuration de Builder :
  - setMessage() permet de définir le "corps" de la boîte de dialogue
  - setTitle() et setIcon() permettent de configurer le texte et/ou l'icône
  - setPositiveButton(), setNeutralButton() et setNegativeButton() permettent d'indiquer les boutons qui apparaîtront en bas de la boîte de dialogue, leur emplacement latéral (respectivement, à gauche, au centre ou à droite), leur texte et le code qui sera appelé lorsqu'on clique sur un bouton (en plus de refermer la boîte de dialogue).

# Boîte de dialogue: La classe AlertDialog



1: Project

MainActivity.java activity\_main.xml exit.png macky\_sall.jpg

Palette

Common Text Buttons Widgets Layouts Containers Google Legacy

2: Favorites

Component Tree

LinearLayout(vertical)

- Ab textView- "Vous avez cliqué..."
- Ab editText(Plain Text)
- button- "Valider la saisie"
- checkbox2- "Mon Exemple ..."
- resto- "Aller au Restaurant"
- button3- "Afficher la liste"
- button2- "Quitter"
- + floatingActionButton

Design Text

6: Logcat TODO Terminal Build Profiler 4: Run

Attributes

- nextFocusDown
- nextFocusForward
- nextFocusLeft
- nextFocusRight
- nextFocusUp
- ▶ numeric
- onClick
- overScrollMode
- password
- phoneNumber
- privateimeOptions
- ▶ requiresFadingEdge
- rotation
- rotationX
- rotationY
- saveEnabled
- scaleX
- scaleY
- scrollHorizontally
- scrollX
- scrollY
- scrollbarAlwaysDrawHorizontalTrack
- scrollbarAlwaysDrawVerticalTrack
- scrollbarDefaultDelayBeforeFade
- scrollbarFadeDuration
- scrollbarSize
- scrollbarStyle
- scrollbarThumb
- scrollbarThumb

IDE and Plugin Updates  
Android Studio is ready to update.

Context: <no context>

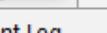
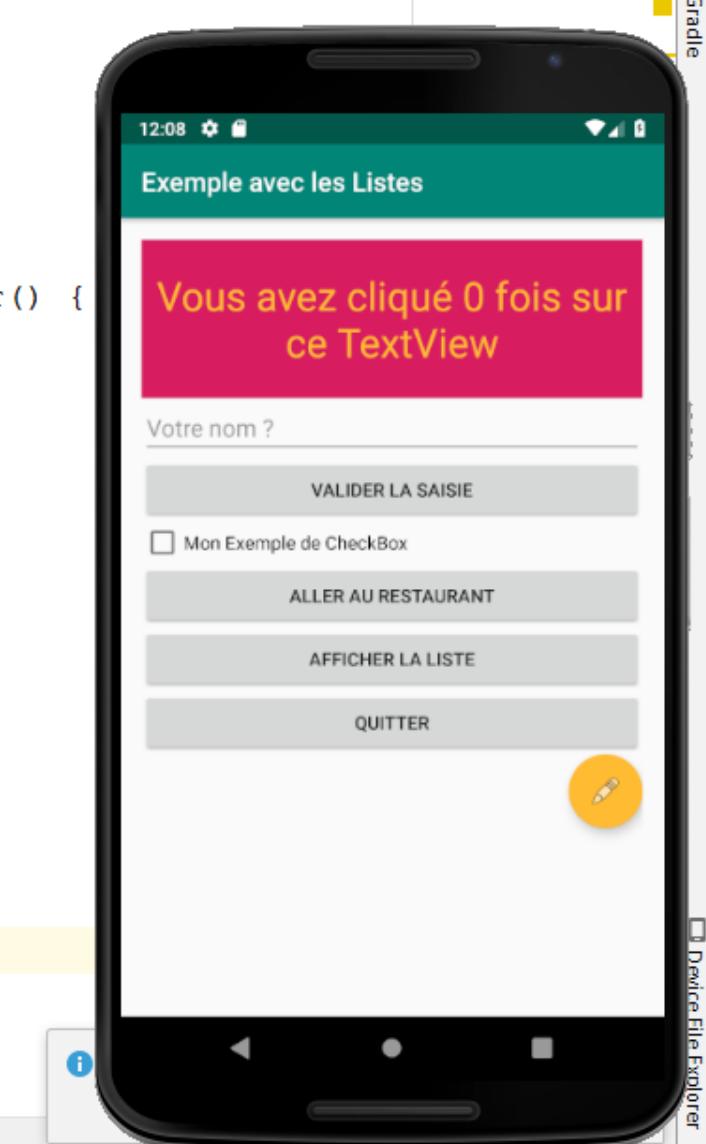
Gradle Device File Explorer Event Log

The screenshot shows the Android Studio interface with the project 'ExempleaveclesListes' open. The main window displays the 'activity\_main.xml' layout. The layout consists of a red header with the text 'Vous avez cliqué 0 fois sur ce TextView', followed by a white input field with the placeholder 'Votre nom ?'. Below the input field are three buttons: 'VALIDER LA SAISIE', 'Mon Exemple de CheckBox' (with a checked checkbox), 'ALLER AU RESTAURANT', 'AFFICHER LA LISTE', and 'QUITTER'. A floating action button is located in the bottom right corner. The 'Component Tree' panel on the left shows the structure of the layout, with the 'button2' node (labeled 'Quitter') currently selected. The 'Attributes' panel on the right shows the properties for the selected button, specifically setting the 'onClick' event to 'quitter'. A notification at the bottom right indicates that 'Android Studio is ready to update'.



MainActivity.java

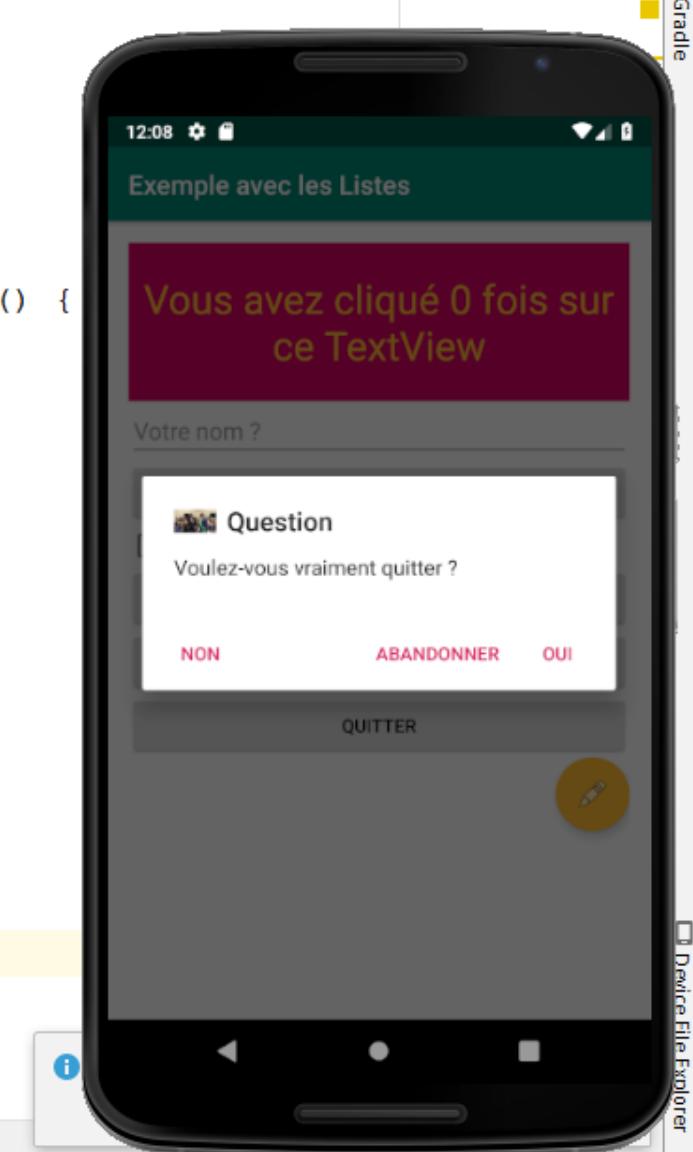
```
64     public void quitter(View v) {
65         // Créer l'alert builder
66         AlertDialog.Builder builder = new AlertDialog.Builder( context: this );
67         builder.setTitle("Question");
68         builder.setMessage("Voulez-vous vraiment quitter ?");
69         // Ajouter les bouttons
70         builder.setPositiveButton( text: "Oui", new DialogInterface.OnClickListener() {
71             @Override
72             public void onClick(DialogInterface dialog, int which) {
73                 finish();
74             }
75         });
76         builder.setNeutralButton( text: "Non", listener: null );
77         builder.setNegativeButton( text: "Abandonner", listener: null );
78         // créer et montrer la boîte alerte
79         AlertDialog dialog = builder.create();
80         dialog.setIcon(R.drawable.macky_sall);
81         dialog.show();
82     }
83 }
84
```





MainActivity.java

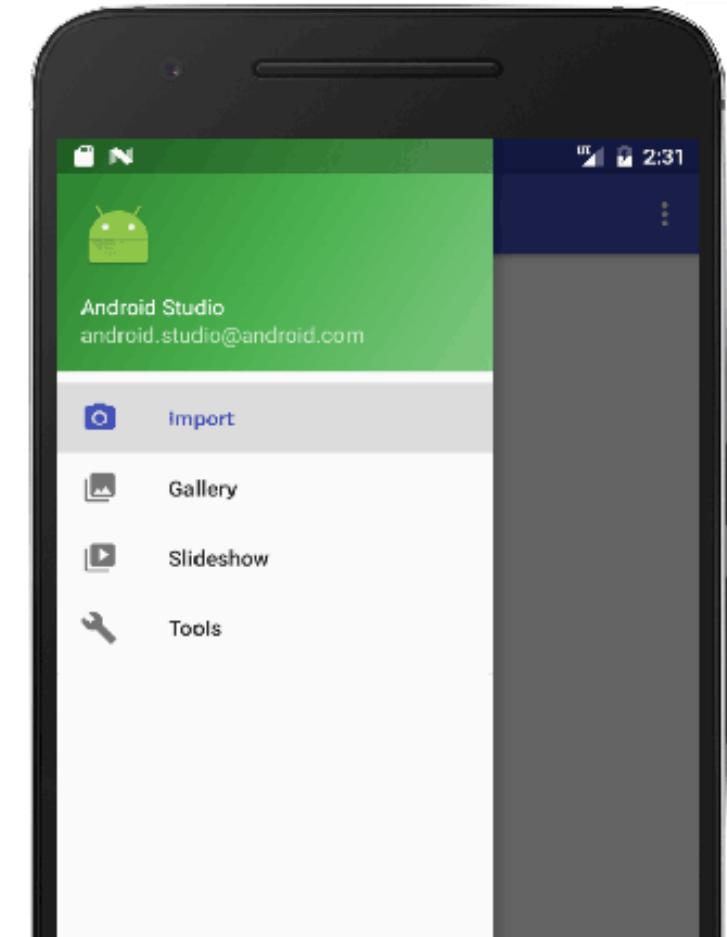
```
64     public void quitter(View v) {
65         // Créer l'alert builder
66         AlertDialog.Builder builder = new AlertDialog.Builder( context: this );
67         builder.setTitle("Question");
68         builder.setMessage("Voulez-vous vraiment quitter ?");
69         // Ajouter les bouttons
70         builder.setPositiveButton( text: "Oui", new DialogInterface.OnClickListener() {
71             @Override
72             public void onClick(DialogInterface dialog, int which) {
73                 finish();
74             }
75         });
76         builder.setNeutralButton( text: "Non", listener: null );
77         builder.setNegativeButton( text: "Abandonner", listener: null );
78         // créer et montrer la boîte alerte
79         AlertDialog dialog = builder.create();
80         dialog.setIcon(R.drawable.macky_sall);
81         dialog.show();
82     }
83 }
84
```



# NAVIGATION DRAWER

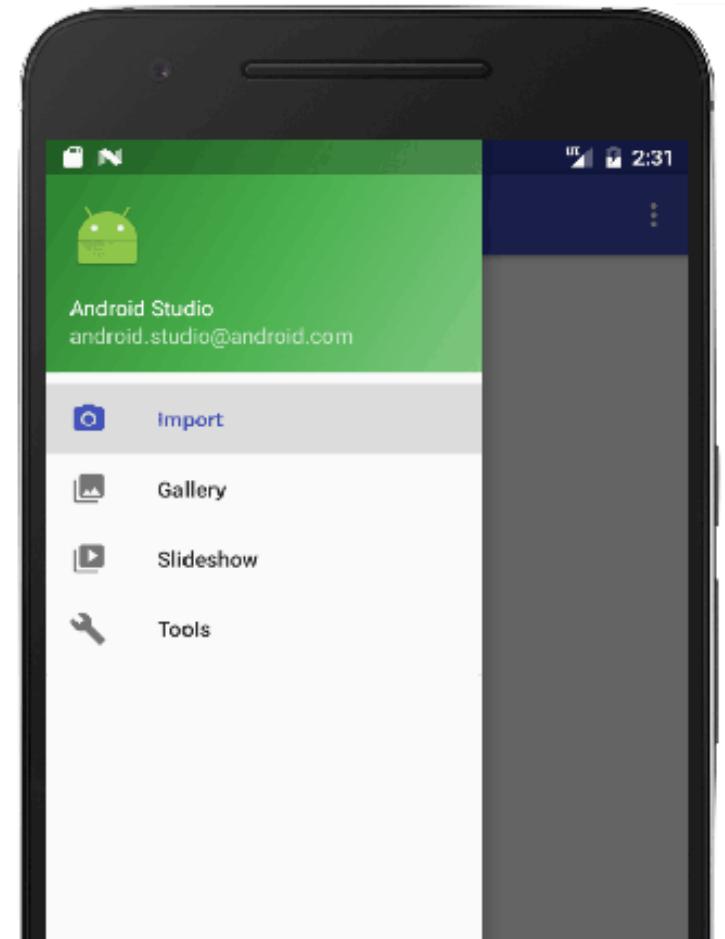
# Navigation Drawer

- The navigation drawer est un panneau d'interface utilisateur qui affiche le menu de navigation principal de votre application.
- Il est masqué lorsqu'il n'est pas utilisé, mais apparaît lorsque l'utilisateur fait glisser un doigt du bord gauche de l'écran.



# Drawer Layout

- Ajouter à Android v 22.1
  - Donc, vous devez utiliser la bibliothèque de support Android
- Déclarez votre mise en page racine avec DrawerLayout.
- Dans DrawerLayout, ajoutez une disposition pour le contenu principal de l'interface utilisateur et une autre vue contenant le contenu du navigation drawer.



# Drawer Layout

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Use DrawerLayout as root container for activity -->
<android.support.v4.widget.DrawerLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:id="@+id/drawer_layout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true">

</android.support.v4.widget.DrawerLayout>
```

# Drawer Layout

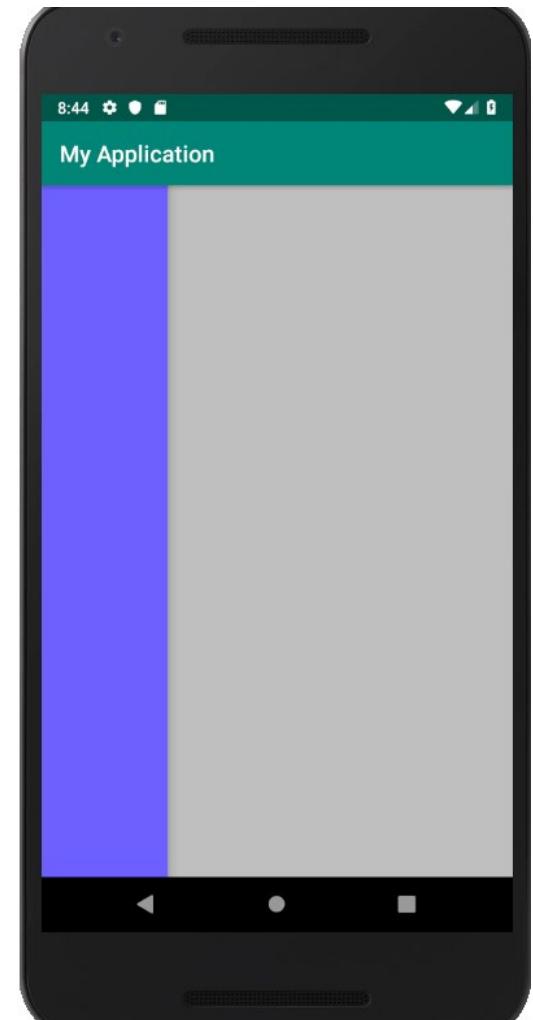
```
<?xml version="1.0" encoding="utf-8"?>
<!-- Use DrawerLayout as root container for activity -->
<android.support.v4.widget.DrawerLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:id="@+id/drawer_layout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true">

    <!-- Layout to contain contents of main body -->
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="horizontal"></LinearLayout>

</android.support.v4.widget.DrawerLayout>
```

# Drawer Layout

```
<android.support.v4.widget.DrawerLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:id="@+id/drawer_layout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true">
    <!-- Layout to contain contents of main body -->
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="horizontal"></LinearLayout>
    <!-- Container for contents of drawer -->
    <android.support.design.widget.NavigationView
        android:layout_width="wrap_content"
        android:layout_height="match_parent"
        android:layout_gravity="start"/>
</android.support.v4.widget.DrawerLayout>
```



# Items de menu pour nav drawer

Créer une ressource de menu avec le nom de fichier correspondant

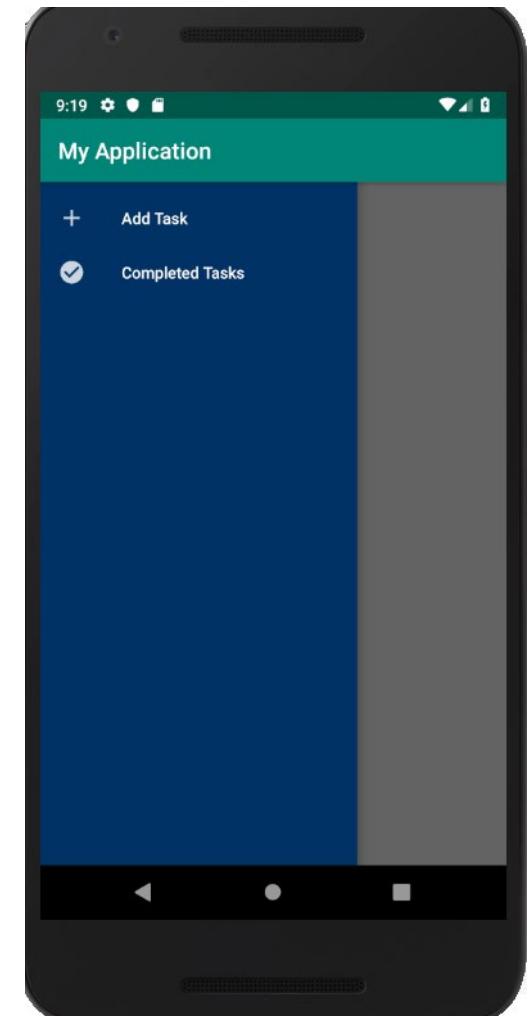
Ajouter des éléments de menu au fichier de ressources

```
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <group>
        <item
            android:id="@+id/nav_camera"
            android:icon="@drawable/ic_addw"
            android:title="Add Task" />
        <item
            android:id="@+id/nav_gallery"
            android:icon="@drawable/ic_list"
            android:title="Completed Tasks" />
    </group>
</menu>
```

# Menu items for nav drawer

- Configurez les éléments de menu répertoriés dans le nav drawer
- Spécifiez une ressource de menu avec l'attribut app: menu

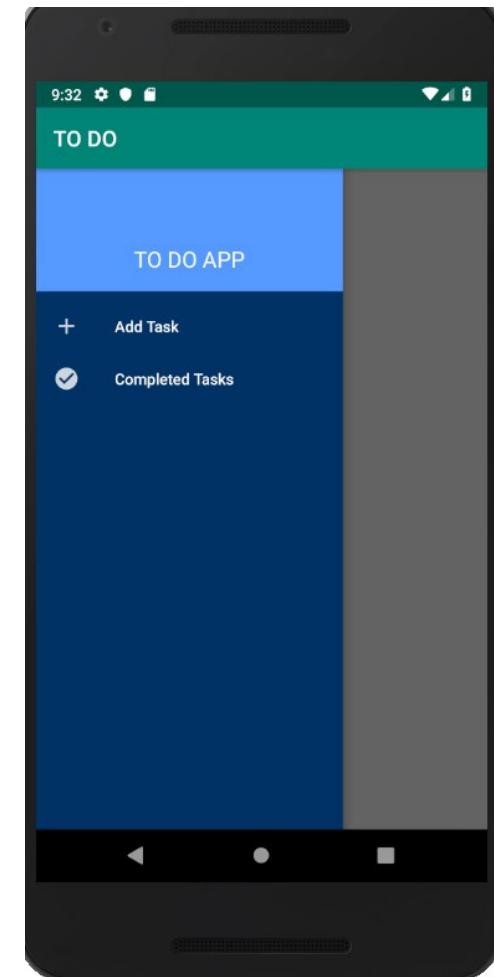
```
<android.support.design.widget.NavigationView  
    android:layout_width="wrap_content"  
    android:layout_height="match_parent"  
    android:layout_gravity="start"  
    android:background="#003366"  
    app:itemIconTint="@android:color/white"  
    app:itemTextColor="@android:color/white"  
    app:menu="@menu/navigation_menu" />
```



# Ajouter un en-tête à la nav drawer

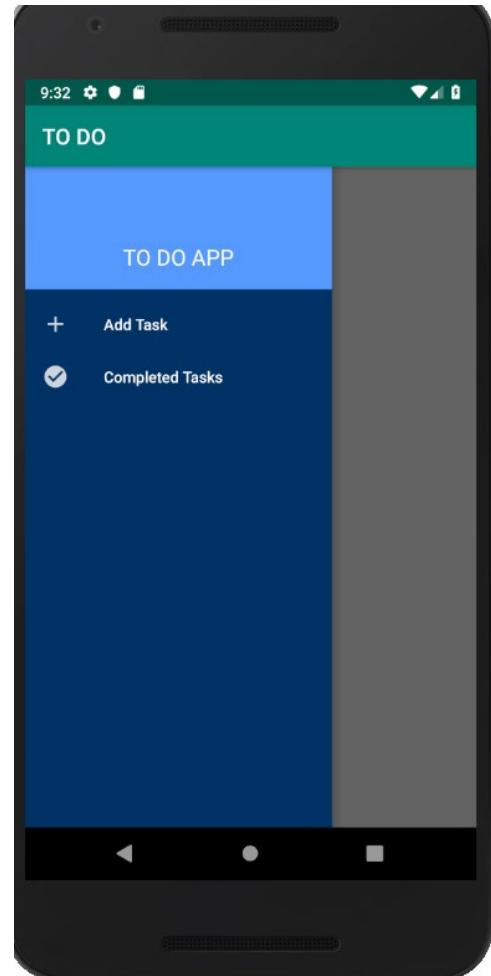
- Vous pouvez ajouter un en-tête au nv drawer en spécifiant une présentation avec l'attribut **app: headerLayout**.

```
<android.support.design.widget.NavigationView  
    android:layout_width="wrap_content"  
    android:layout_height="match_parent"  
    android:layout_gravity="start"  
    android:background="#003366"  
    app:itemIconTint="@android:color/white"  
    app:itemTextColor="@android:color/white"  
    app:menu="@menu/navigation_menu"  
    app:headerLayout="@layout/nav_header"/>
```



# Ajouter l'entête

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="192dp"
    android:background="#5599ff"
    android:gravity="bottom"
    android:orientation="vertical"
    android:padding="16dp"
    android:theme="@style/ThemeOverlay.AppCompat.Dark">
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="TO DO APP"
        android:textAppearance="@style/TextAppearance.AppCompat.Body1"
        android:textSize="20sp"
        android:textStyle="bold" />
</LinearLayout>
```



# Gestion de la navigation au click

Pour recevoir des rappels, implémentez l'interface OnNavigationItemSelectedListener et ajoutez-la à votre NavigationView en appelant setNavigationItemSelectedListener () .

```
NavigationView navigationView = findViewById(R.id.nav_view);
navigationView.setNavigationItemSelectedListener(
    new NavigationView.OnNavigationItemSelectedListener() {
        @Override
        public boolean onNavigationItemSelected(MenuItem menuItem){
            if (menuItem.getTitle().toString().startsWith("Add")){
                getJob();
            }
            mDrawerLayout.closeDrawers();
            return true;
        }
    });
});
```