Création des **a**pplications **i**nteractives

Licence 3
S06,
Dr. **Sabri Ghazi**

Pour gérer l’interaction?

- Une application **Android** est un ensemble d’**Activities.**
- Parmi ces activités il y a une activité **principale (Main)**.
- Chaque activité doit posséder une interface.
- Cette interface permettra l’interaction avec l’utilisateur.

Gestion de l’interaction avec l’utilisateur

- On peut catégoriser ces interactions en :
  1. Répondre aux cliques de l’utilisateur.
  2. Modifier l’état des éléments de l’interface pour afficher des informations.
  3. Afficher des messages de dialogue
  4. Naviguer entre les activités de l’application.

L’événement de clique

- Chaque composant de l’interface est « **Cliquable** »
  - Il possède un attribut **onClick**
  - Dans cet attribut on mis le nom de la fonction java qui s’exécutera lorsqu’on clique sur le composant.
  - Cette fonction doit posséder la signature

```
public void lancer(View V){
    //ici on mit le traitement qui se lancera lorsqu'on clique
}
```

L’événement de clique

- La fonction doit être :
  - **public**
  - Elle renvoi **void**
  - Et à comme argument **View v**
    - La variable **v** contient un pointeur sur l’élément qui a été cliqué.

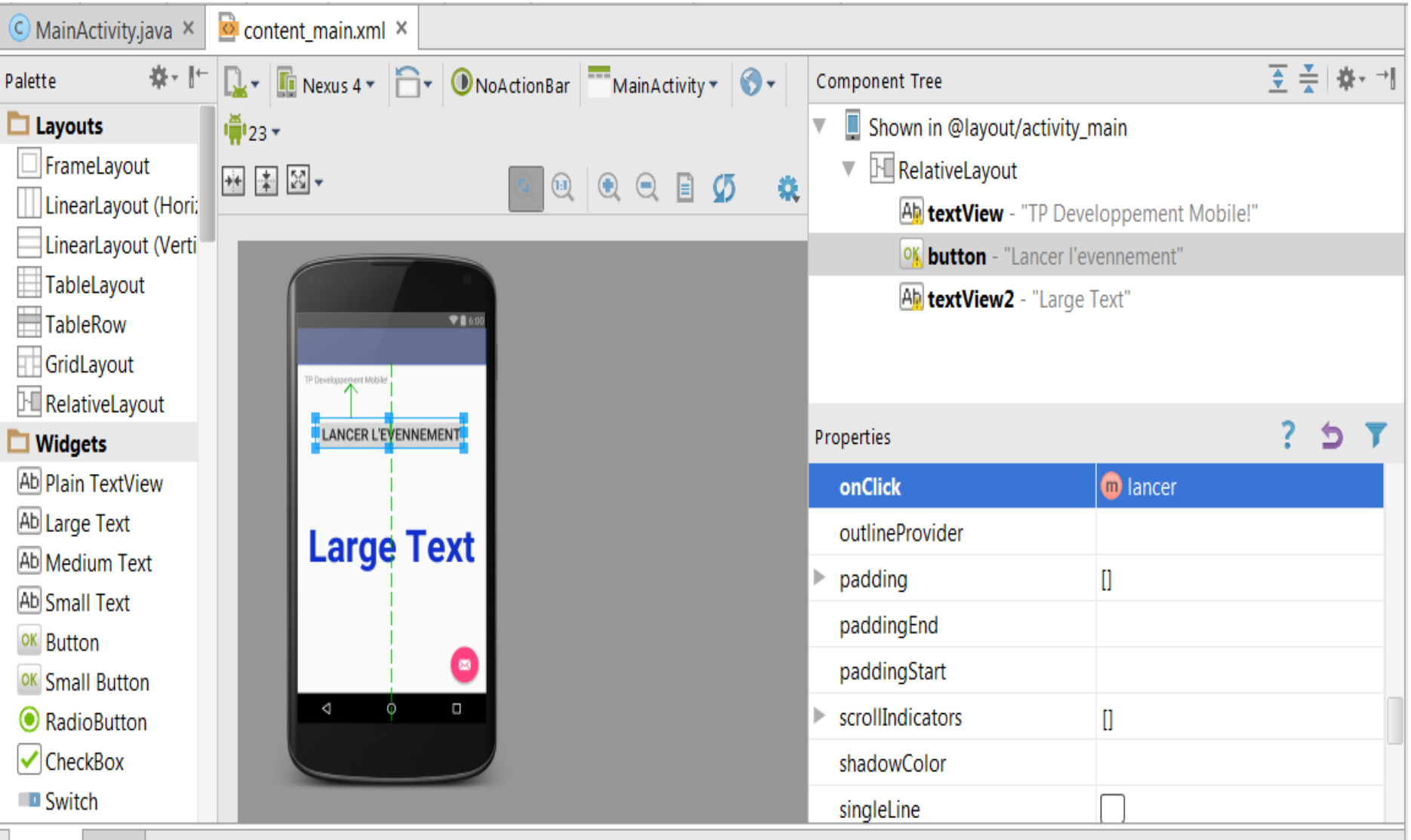
```
int cpt=0;
public void btn_click(View v){
    cpt=cpt+1;
    Button b=(Button)v;
    b.setText("Ce bouton a été cliqué : "+cpt);
}
```

L’événement de clique

MainActivity.java × content_main.xml ×

```
public boolean onOptionsItemSelected(MenuItem item) {
    // Handle action bar item clicks here. The action bar will
    // automatically handle clicks on the Home/Up button, so long
    // as you specify a parent activity in AndroidManifest.xml.
    int id = item.getItemId();

    //noinspection SimplifiableIfStatement
    if (id == R.id.action_settings) {
        return true;
    }

    return super.onOptionsItemSelected(item);
}

public void lancer(View V){
    //ici on mit le traitement qui se lancera lorsqu'on clique
}
}
```

L’événement de clique



L’événement de clique

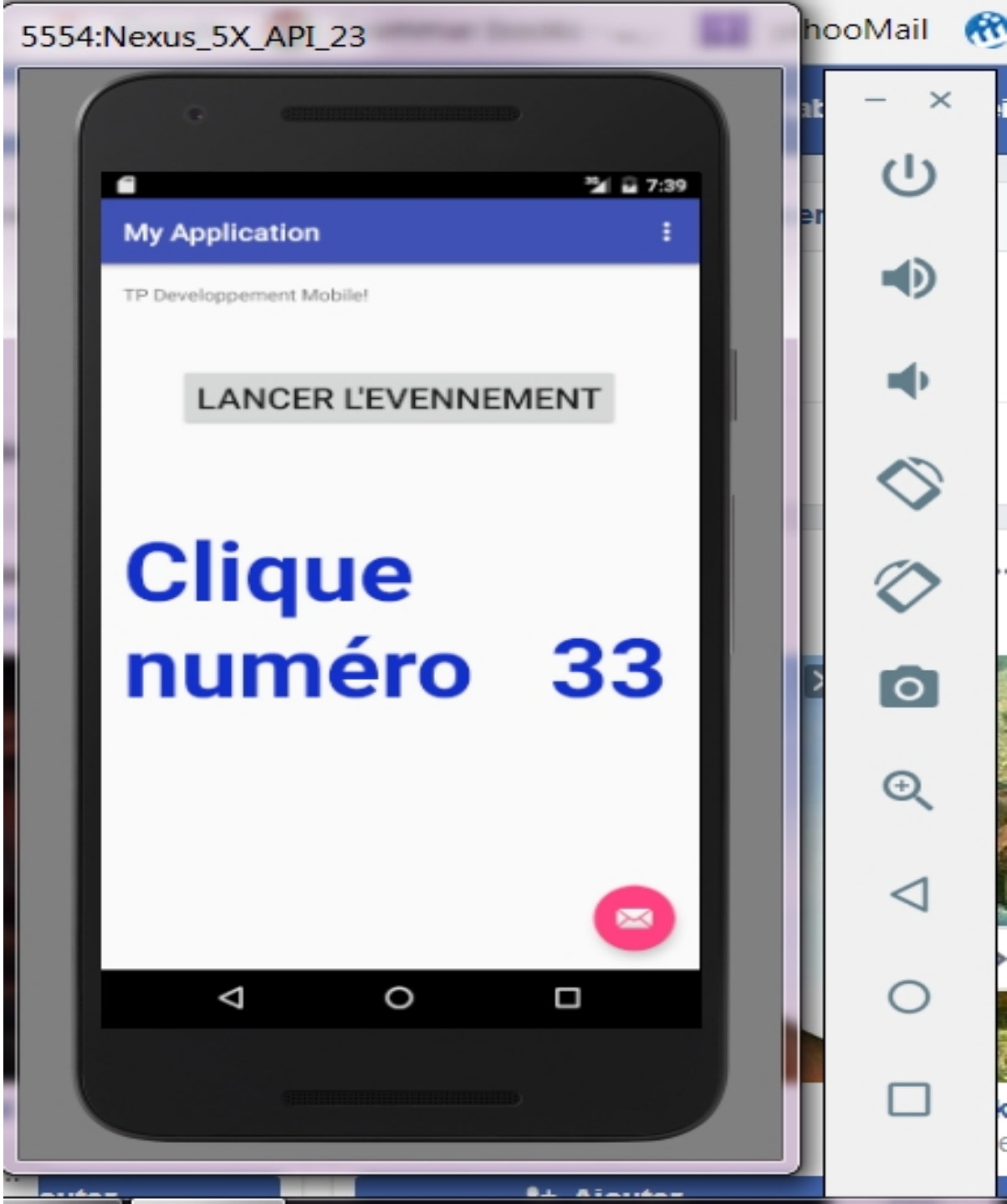
- À chaque clique sur ce composant d’interface, la méthode nommée dans l’exemple « lancer » sera exécutée.
- On va essayer de modifier cette méthode, on cherche a compter le nombre de clique sur ce bouton.

L’événement de clique

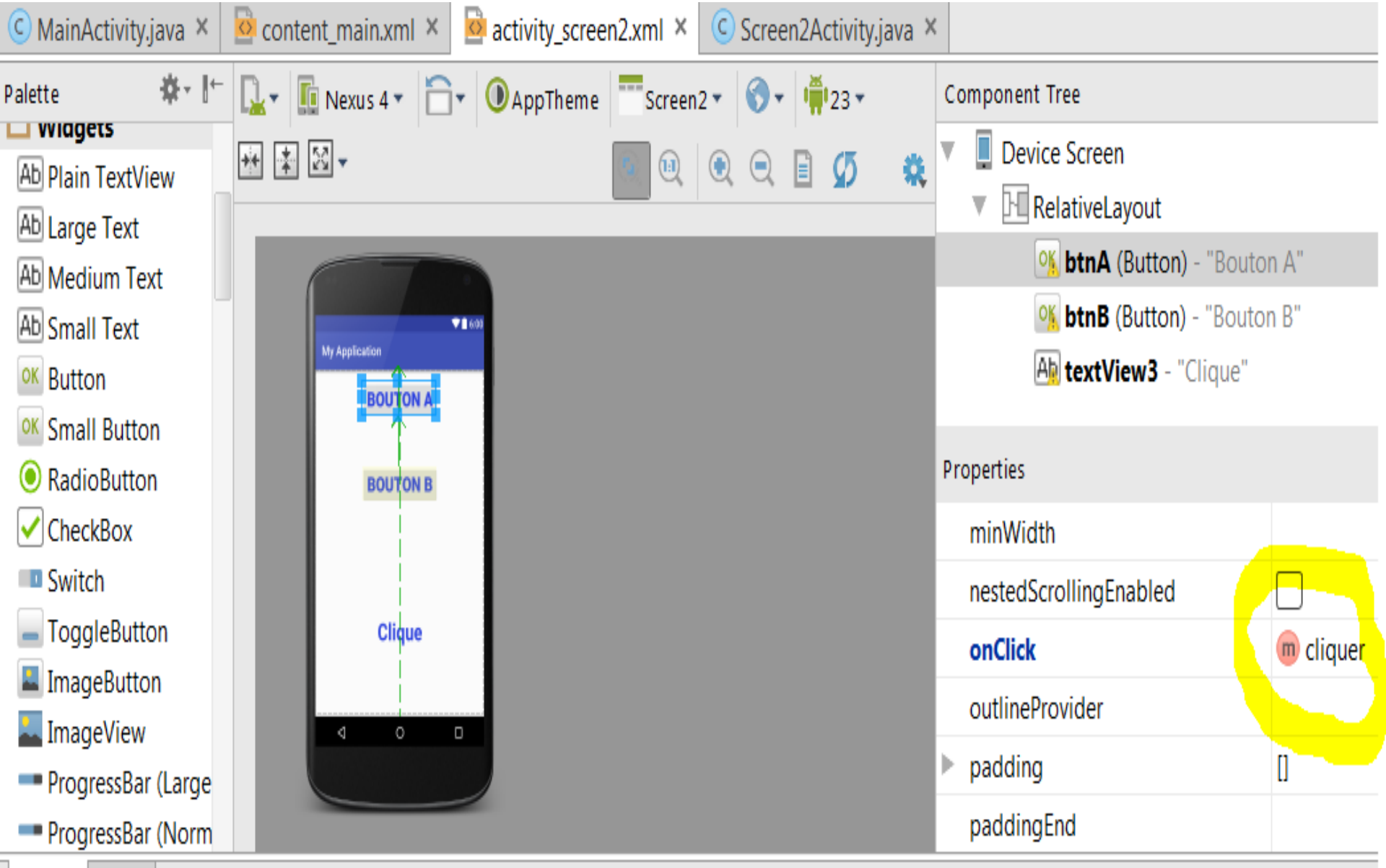
MainActivity.java × content_main.xml ×

```
        }

        return super.onOptionsItemSelected(item);
    }

    int count=0;
    public void lancer(View V){

        TextView txtV=(TextView)findViewById(R.id.textView2);
        txtV.setText("Clique numéro :"+count);
        count++;
        //ici on mit le traitement qui se lancera lorsqu'on clique
    }
}
```

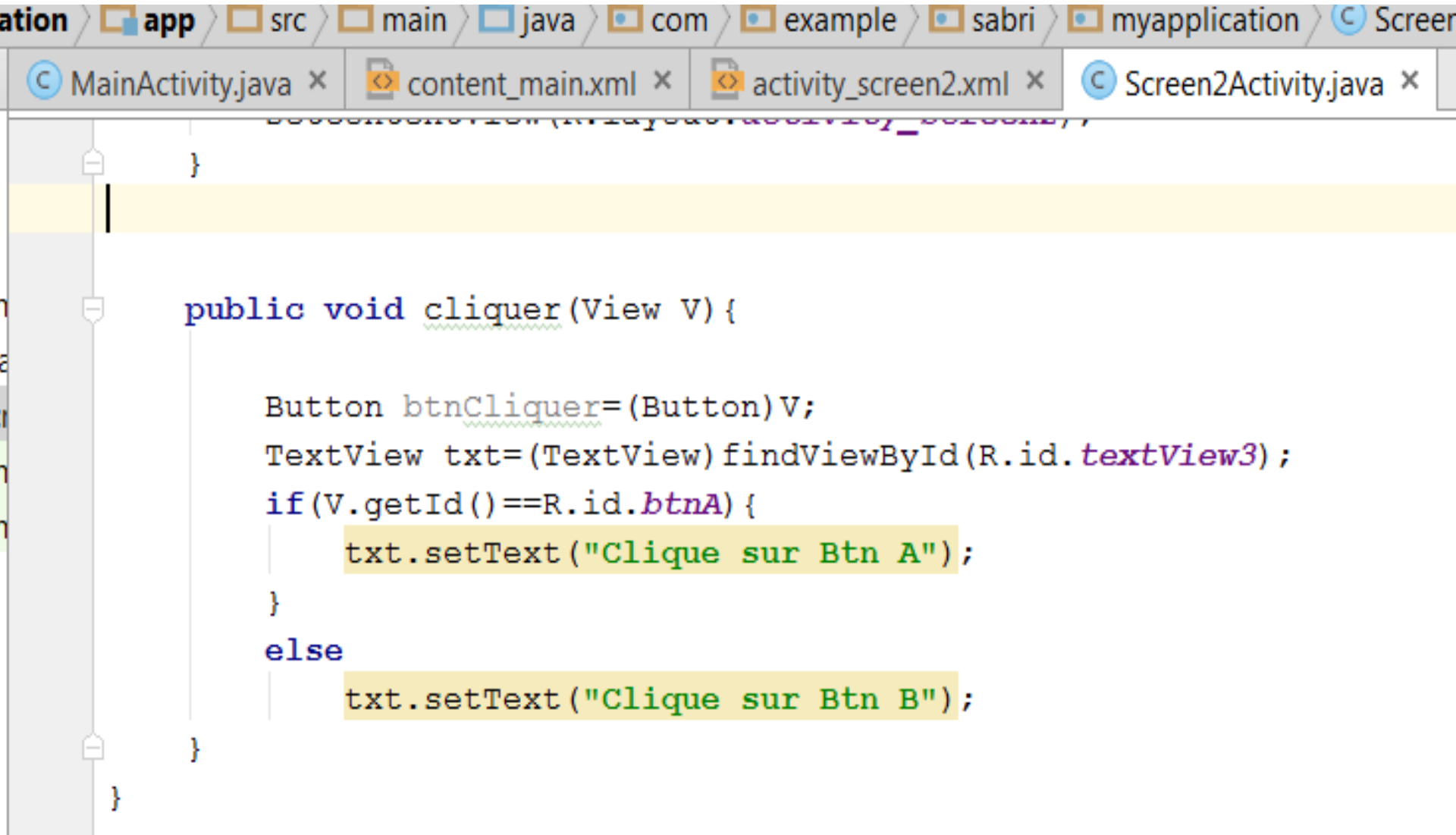
Exemple


5554:Nexus_5X_API_23
hooMail
7:39
My Application
TP Developpement Mobile!
LANCER L'EVENNEMENT
Clique
numéro 33

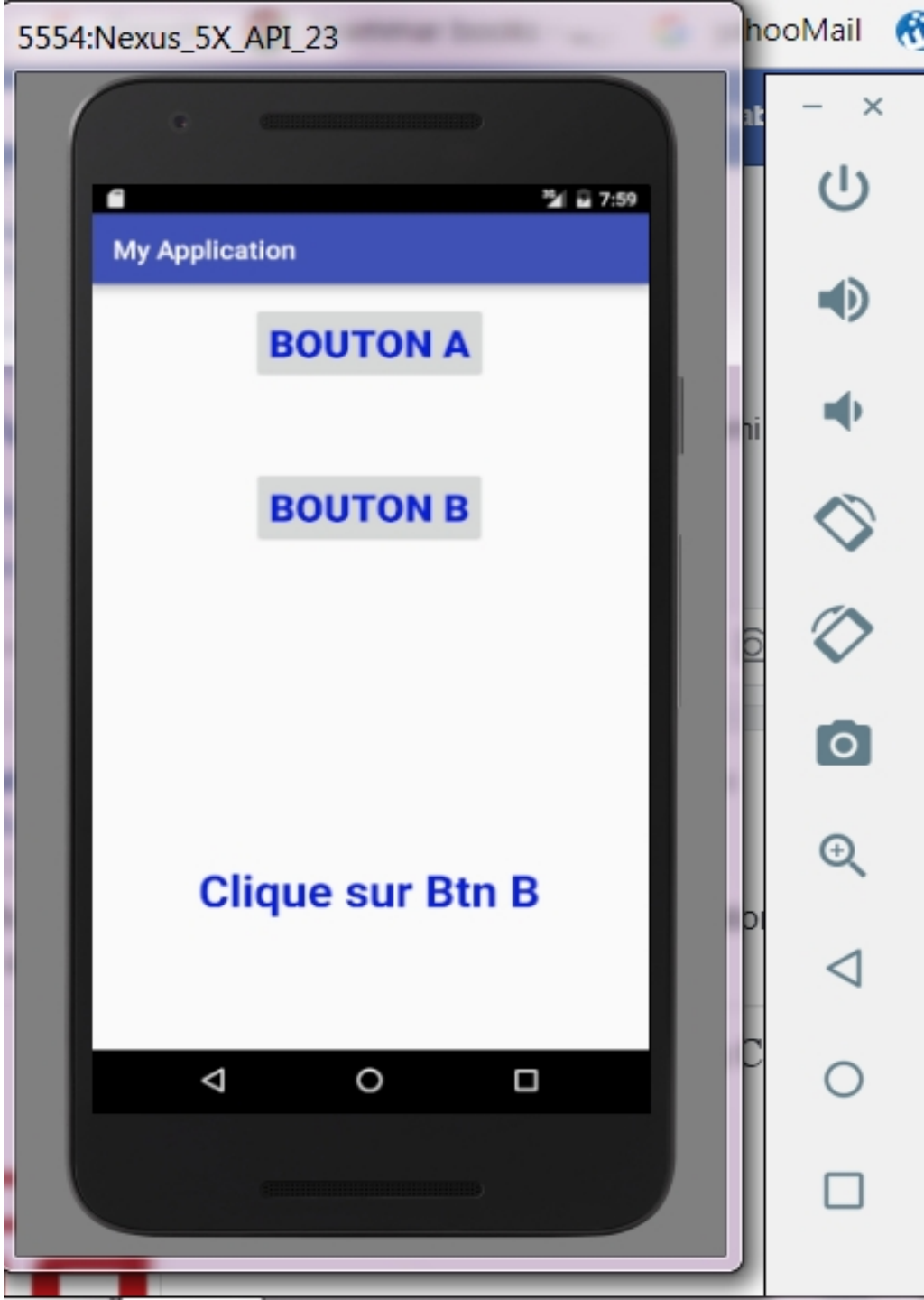
Exemple 2


MainActivity.java
content_main.xml
activity_screen2.xml
Screen2Activity.java
Palette
Widgets
Plain TextView
Large Text
Medium Text
Small Text
Button
Small Button
RadioButton
CheckBox
Switch
ToggleButton
ImageButton
ImageView
ProgressBar (Large
ProgressBar (Norm
Nexus 4
AppTheme
Screen2
23
My Application
BOUTON A
BOUTON B
Clique
Component Tree
Device Screen
RelativeLayout
btnA (Button) - "Bouton A"
btnB (Button) - "Bouton B"
textView3 - "Clique"
Properties
minWidth
nestedScrollingEnabled
onClick
cliquer
outlineProvider
padding
[]
paddingEnd

Exemple 2


ation app src main java com example sabri myapplication Screen
MainActivity.java
content_main.xml
activity_screen2.xml
Screen2Activity.java
}
public void cliquer(View V){
Button btnCliquer=(Button)V;
TextView txt=(TextView)findViewById(R.id.textView3);
if(V.getId()==R.id.btnA){
txt.setText("Clique sur Btn A");
}
else
txt.setText("Clique sur Btn B");
}
}

Exemple 2


5554:Nexus_5X_API_23
7:59
My Application
BOUTON A
BOUTON B
Clique sur Btn B

Changer l’état des éléments de UI

- Chaque éléments de la UI possède un identifiant.
- En utilisant cet identifiant , dans le code java on peut récupérer une référence , sur cette élément, et de ce fait on peut modifier tous ses attributs.

Exemple

- Supposant qu'on a un TextView dans l'interface nommé **txtNom,**
- Pour modifier le texte du textview on fait :
  - La fonction ***findViewById*** reçoit le ***id*** du composant et nous donne la référence .

```
public void modifierText(View v){

    TextView txtNom=findViewById(R.id.txtNom);

    txtNom.setText("Text Modifier");
    txtNom.setTextSize(60.00f);
}
```

Changer l’état des éléments de UI

- Il faut noter que la méthode findViewById doit toujour être appelé après le chargement de l’interface , donc après l’exécution de la fonction ***onCreate*** de l’activité et précisément après l’appel de :
  - ***setContentView***

Afficher des messages et des dialogues

- En plus de son interface, l’application a parfois besoin de communiquer et de demander des informations a travers les dialogues et les messages.
- Toast Message apportant des informations , mais ne necesitant aucune interaction de la part de l’utilisateur.
- **AlertDialog** des messages qui peuvent necessiter une action de la part de l’utilisateur.