# Annexe 9 – StateListDrawable, Selector, Shape

Dans le TP 2 ( Paint ) , on vous a demandé de passer du thème Material au thème AppCompat.Light afin d'avoir plus de liberté sur l'application de couleurs de fond sur des boutons par exemple.

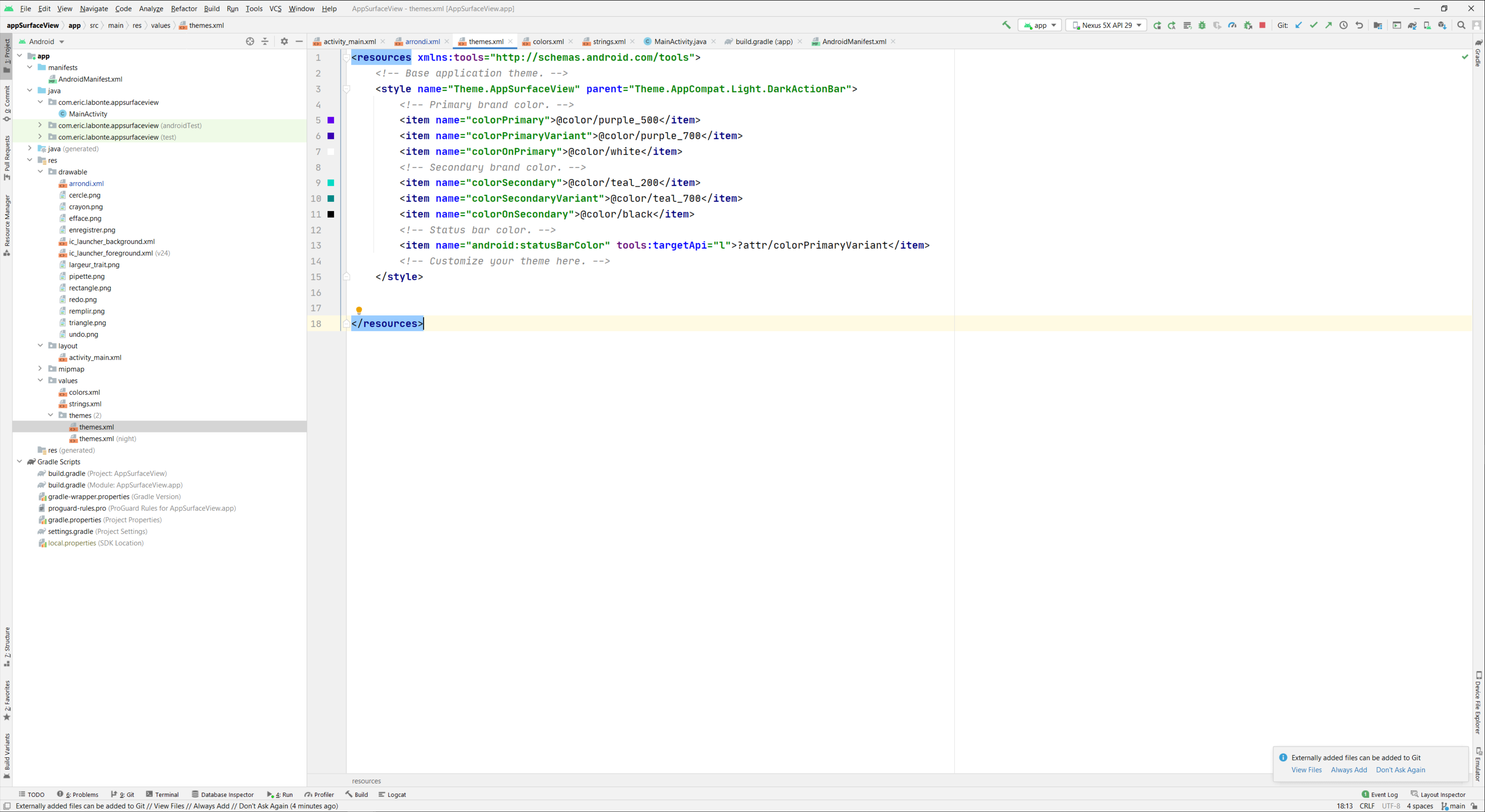
Cependant, certains d'entre vous se sont aperçus qu'on perdait, entre autres, les coins arrondis associés aux boutons. On peut corriger cet état de fait à l'aide de Drawables spéciaux, les StateListDrawables :

## Avantages des StateListDrawables ( Selectors ) :

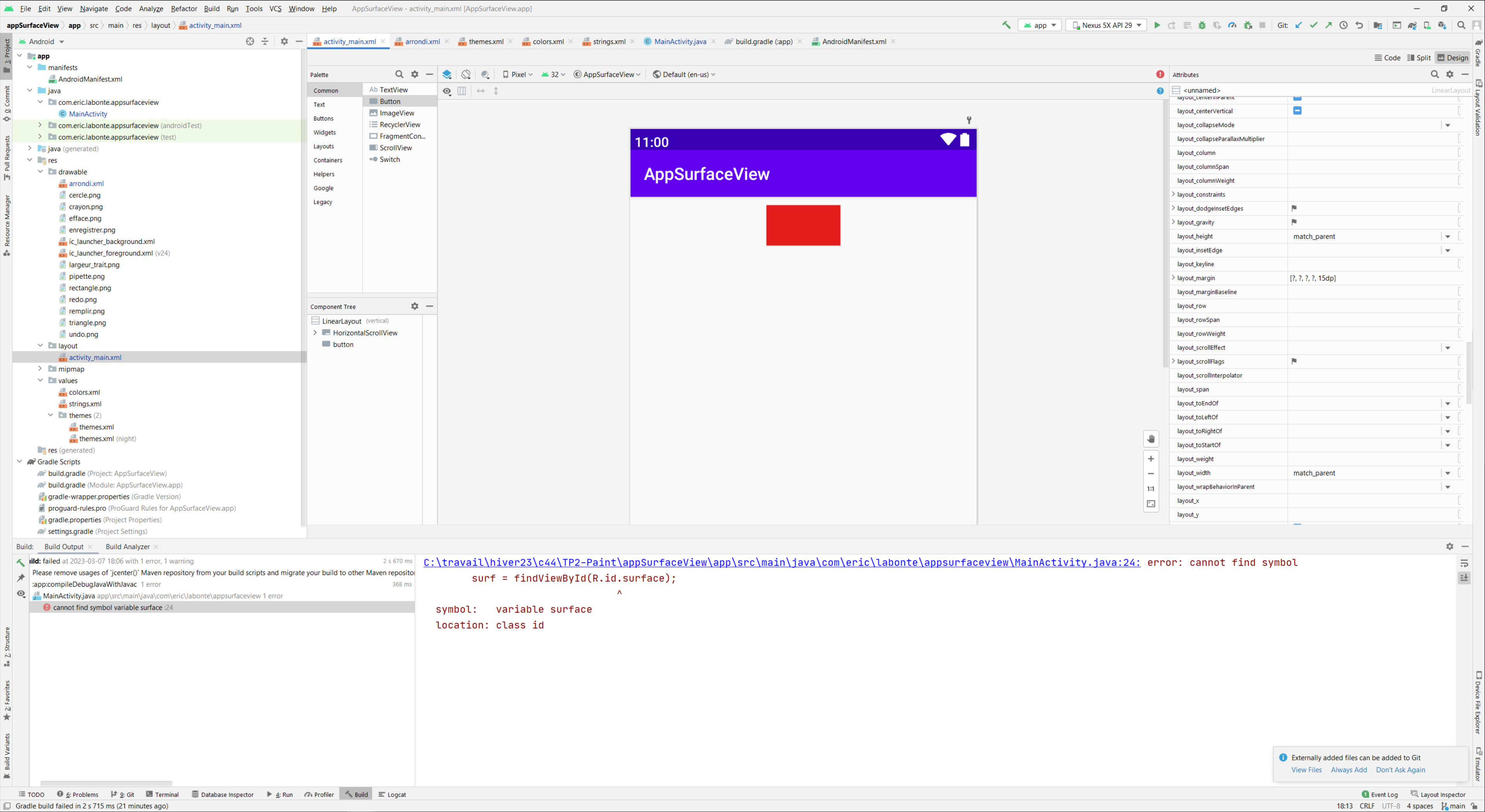
* Le modele/logique du programme n’a pas a s’occuper du changement visuel d’une composante . C’est fait dans le StateListDrawable
* On peut modifier l’apparence d’une composante sans avoir a redéfinir /a faire une sous-classee de la class de la composante

Exercice :

Dans un projet nommé Annexe9, ouvrez le fichier res / values / themes.xml. Changez le parent Theme.MaterialComponents pour Theme.AppCompat.Light.DarkActionBar



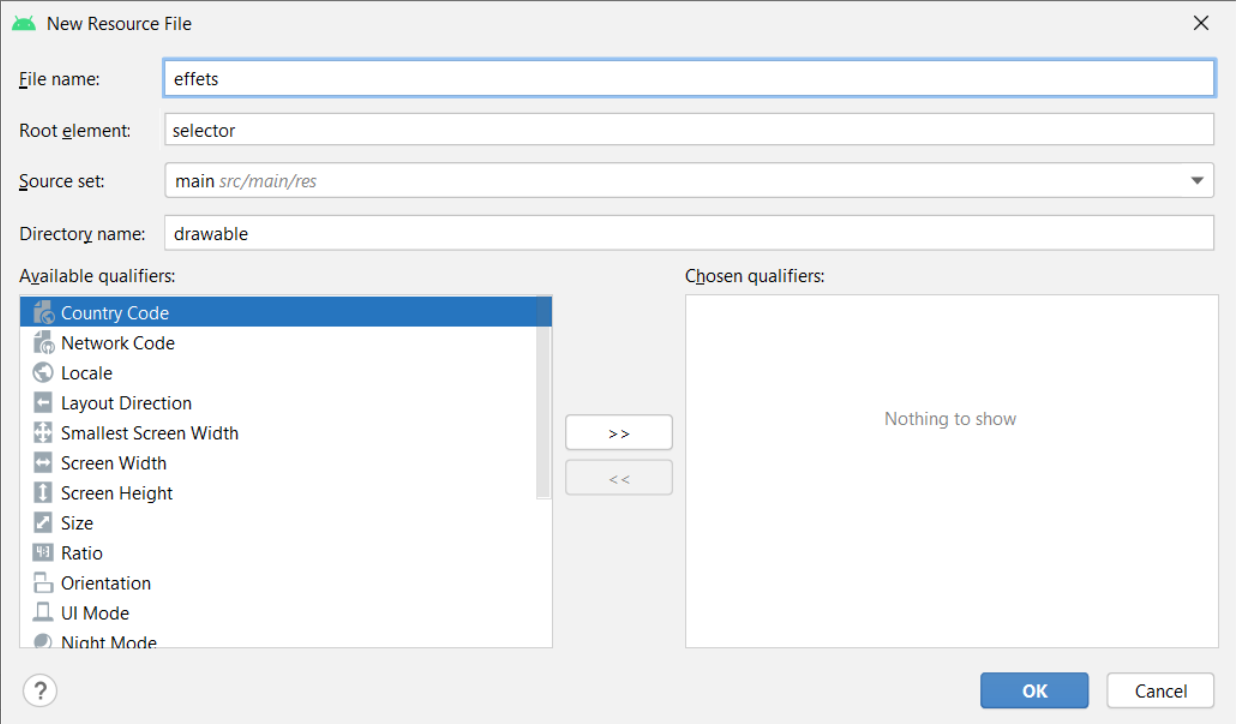
Ajoutez un bouton sur le visuel de l'activité et donnez-lui une couleur de fond :



*Un bouton rouge*

Il a perdu ses coins arrondis et ses effets quand on clique dessus. Tentons de remédier à cela en ajoutant un StateListDrawable à ce bouton

En vous plaçant sur res / drawable , créer un nouveau Drawable Resource File :



Assurez-vous que selector est l'élément-racine, donnez un nom à votre fichier ( effets )

Un fichier effets.xml est produit. Ajoutez le système de balises suivant comme point de départ à l'intérieur de la balise <selector>:

<**item android:state\_pressed="true"** >  
 <**shape android:shape="rectangle"** >  
 <**solid android:color="#ff0000"**/>  
 <**corners android:radius="12dip"** />  
 <**stroke android:width="12px" android:color="@color/teal\_200"** />  
 </**shape**>  
</**item**>  
  
<**item** >  
 <**shape android:shape="rectangle"** >  
 <**corners android:radius="12dp"** />  
 <**stroke android:width="12dp" android:color="#ff0000"** />  
 <**solid android:color="#ff0000"**/>  
 </**shape**>  
</**item**>

De cette façon, le bouton aura une bordure de couleur différente si on appuie dessus.

Pour appliquer le StateListDrawable au bouton, on change le background du bouton pour

@drawable/effets

## Les états possibles ( dépendant de la composante ) :

    android:state\_pressed=["true" | "false"]  
        android:state\_focused=["true" | "false"]  
        android:state\_hovered=["true" | "false"]  
        android:state\_selected=["true" | "false"]  
        android:state\_checkable=["true" | "false"]  
        android:state\_checked=["true" | "false"]  
        android:state\_enabled=["true" | "false"]  
        android:state\_activated=["true" | "false"]  
        android:state\_window\_focused=["true" | "false"]

\*\*\* important de terminer par la version au repos ( sans événement )

## Les types de "shape" possibles :

|  |  |
| --- | --- |
| "rectangle" | Rectangle qui remplit l'affichage parent. Il s'agit de la forme par défaut. |
| "oval" | Forme ovale qui correspond aux dimensions de l'affichage parent. |
| "line" | Ligne horizontale qui occupe toute la largeur de l'affichage parent.  Cette forme nécessite que l'élément <stroke> définisse la largeur de la ligne. |
| "ring" | Forme circulaire. Demande d'autres attributs tels android:innerRadius ou  android:thickness |

## Les propriétés des formes :

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

<shape

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

android:shape=["rectangle" | "oval" | "line" | "ring"] >

<corners

android:radius="integer"

android:topLeftRadius="integer"

android:topRightRadius="integer"

android:bottomLeftRadius="integer"

android:bottomRightRadius="integer" />

<gradient

android:angle="integer"

android:centerX="float"

android:centerY="float"

android:centerColor="integer"

android:endColor="color"

android:gradientRadius="integer"

android:startColor="color"

android:type=["linear" | "radial" | "sweep"]

android:useLevel=["true" | "false"] />

<padding

android:left="integer"

android:top="integer"

android:right="integer"

android:bottom="integer" />

<size

android:width="integer"

android:height="integer" />

<solid

android:color="color" />

<stroke

android:width="integer"

android:color="color"

android:dashWidth="integer"

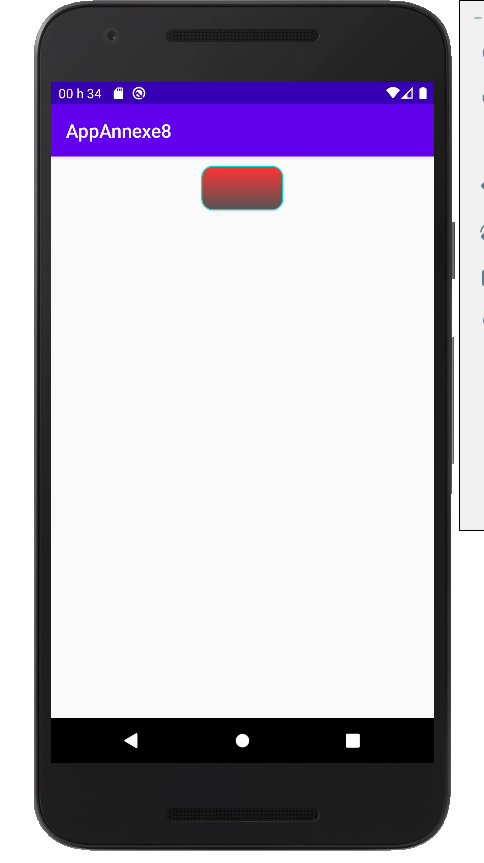
android:dashGap="integer" />

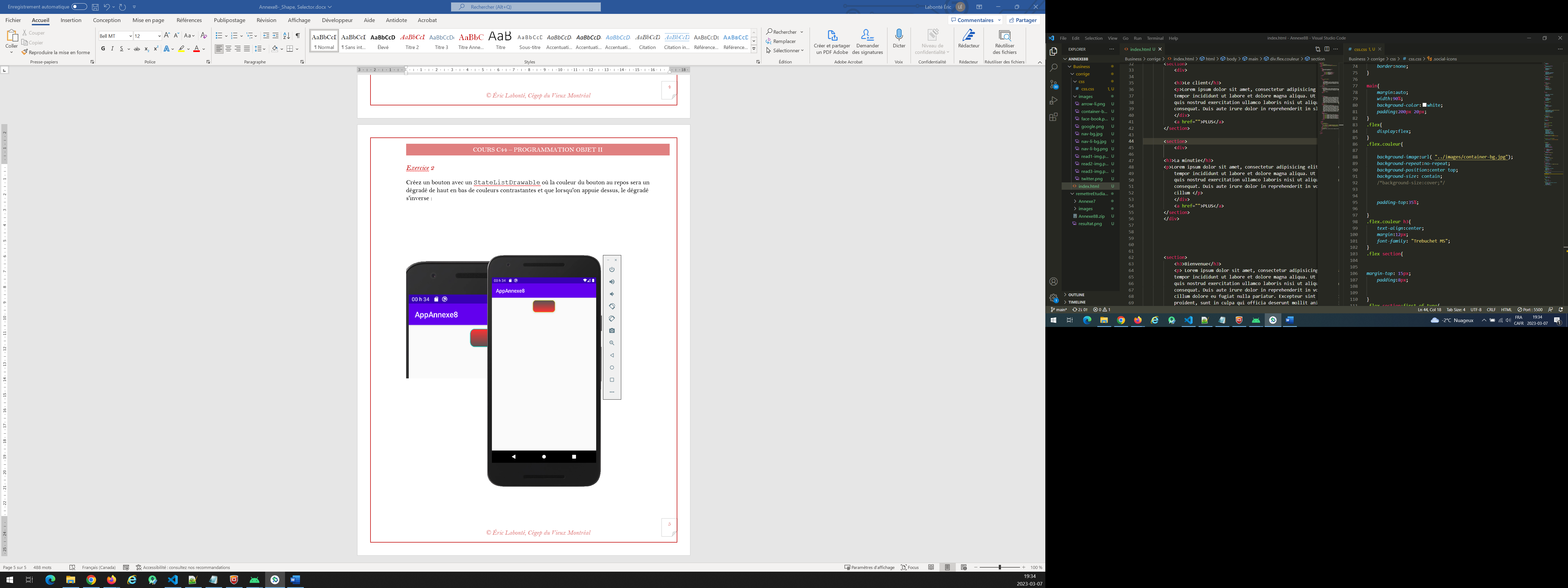
</shape>

<https://developer.android.com/guide/topics/resources/drawable-resource?hl=fr#Shape>

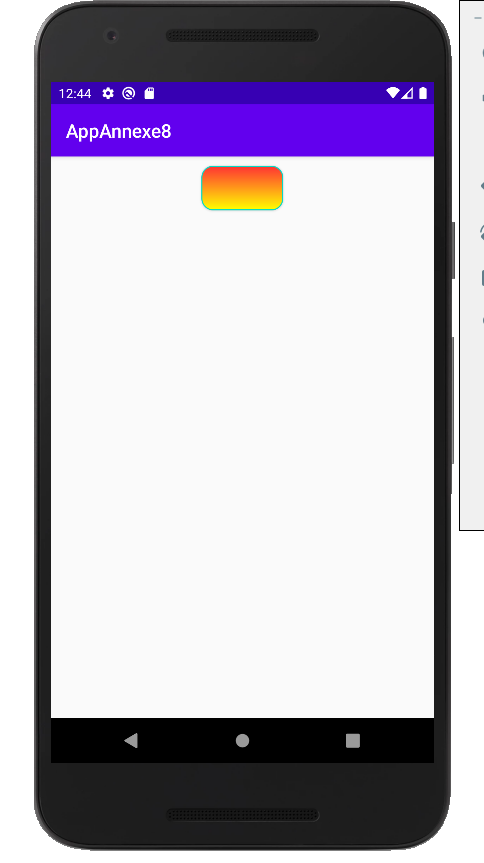
## Exercice 2

Créez un bouton avec un StateListDrawable où la couleur du bouton au repos sera un dégradé de haut en bas de couleurs contrastantes et que lorsqu'on appuie dessus, le dégradé s'inverse :





Faire un autre fichier selector utilisant la locale "espagnol" pour que le bouton ait les couleurs du drapeau de l'Espagne quand la langue de l'émulateur est l'espagnol :



*Le téléphone utilise le fichier correspondant à la langue du téléphone*

## Pour récupérer la couleur de fond en code Java : \*\*\* attention, depuis api29 seulement

getBackground() retournera un objet StateListDrawable

getStateDrawable( n ) sur l'objet StateListDrawable retourne un objet GradientDrawable correspondant à l'item n du fichier selector

…