Ecole Normale Supérieure de l'Enseignement Technique Mohammedia

Université Hassan II de Casablanca

ENSET

ا لمدرسة العليا لأساتذة التعليم التقني المحمدية

جامعة الحسن الثاني بالدار البيضاء

DÉPARTEMENT MATHÉMATIQUES ET INFORMATIQUE

Compte rendu de TP

Filière :

« Génie du Logiciel et des Systèmes Informatiques Distribués »

GLSID

TP 1 : Codez votre propre calculatrice

Réalisé par :

- Hajar ZARGUAN

Encadré par :

Pr. BOUIHI Bouchra

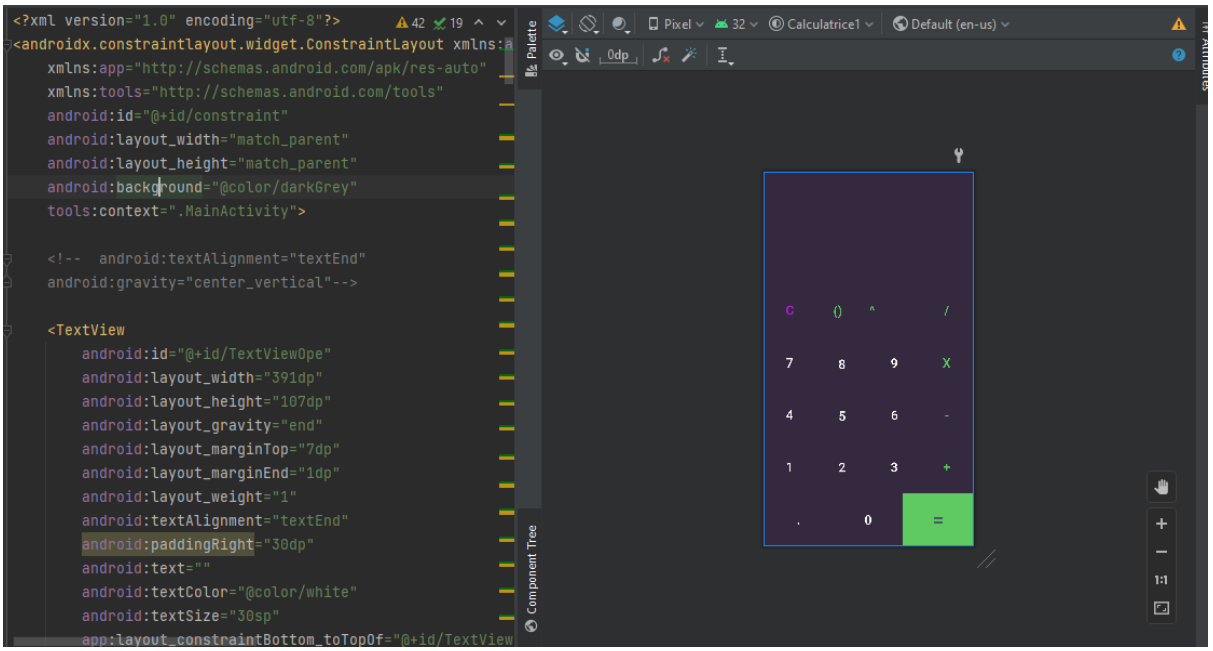
Année Universitaire : 2021-2022

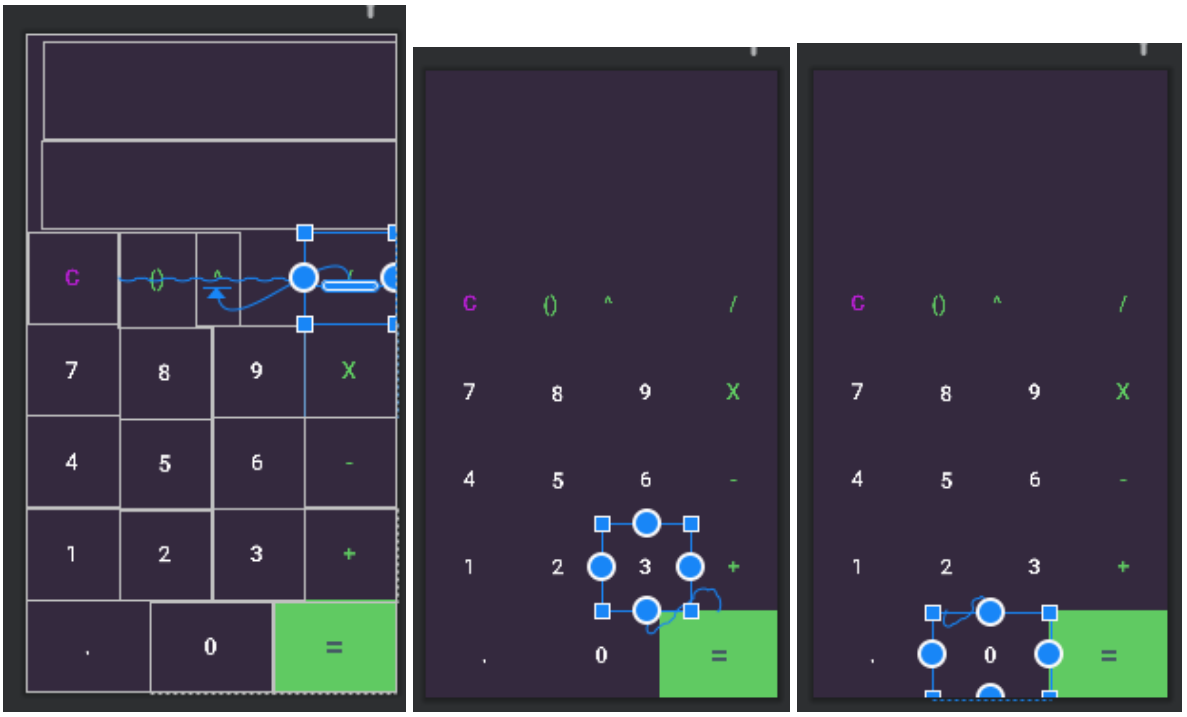
Introduction :

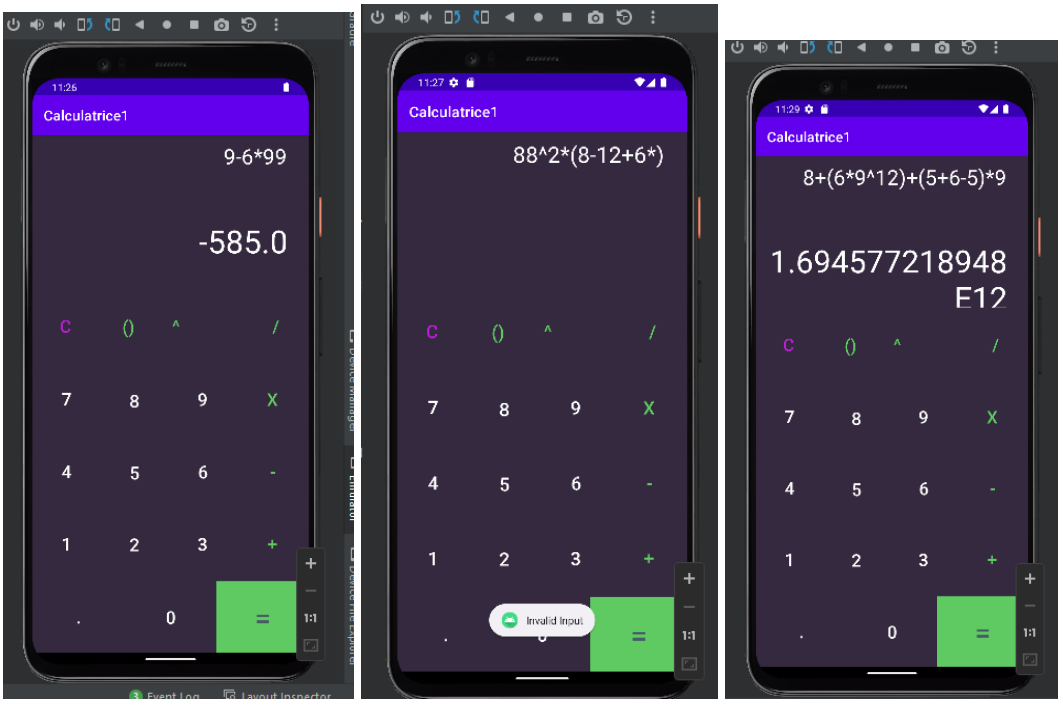
Dans ce TP, il est demandé de réaliser une application qui permet de mettre en pratique la gestion des événements avec une interface graphique un peu complexe. Il vous est proposé donc de réaliser une calculatrice, avec l'interface graphique suivante.

Partie I : L'interface graphique de l'application

L'organisation générale demandée peut se décomposer en un texte en haut de l'écran et un tableau de boutons en bas. En utilisant le LinearLayout avec une orientation verticale, on peut donc placer un TextView en haut de l'écran sur l'intégralité de la largeur avec un texte aligné à droite ainsi qu'un TableLayout en bas. Il faut mentionner que les cases du tableau sont redimensionnables pour occuper un certain nombre de cases si nécessaire (stretchColumn et shrinkColumns).




C () ^ /
7 8 9 X
4 5 6 -
1 2 3 +
. 0 =


11:26
Calculatrice1
9-6*99
-585.0
11:27
Calculatrice1
88^2*(8-12+6*)
Invalid Input
11:29
Calculatrice1
8+(6*9^12)+(5+6-5)*9
1.694577218948
E12

Partie II : Code JAVA

Au niveau du comportement, on se rend compte que pour faire des opérations binaires (avec deux opérandes), il faudra mémoriser deux opérandes et l'opération effectuée. L'action du bouton égal (=) sera celle qui fera le calcul. Il faut aussi mémoriser si on est en train de saisir le premier ou le second opérande. Ainsi, lancer le calcul ne correspondra qu'à faire l'opération demandée entre les 2 opérandes en mémoire, stocker le résultat en premier opérande et mettre à jour l'affichage.

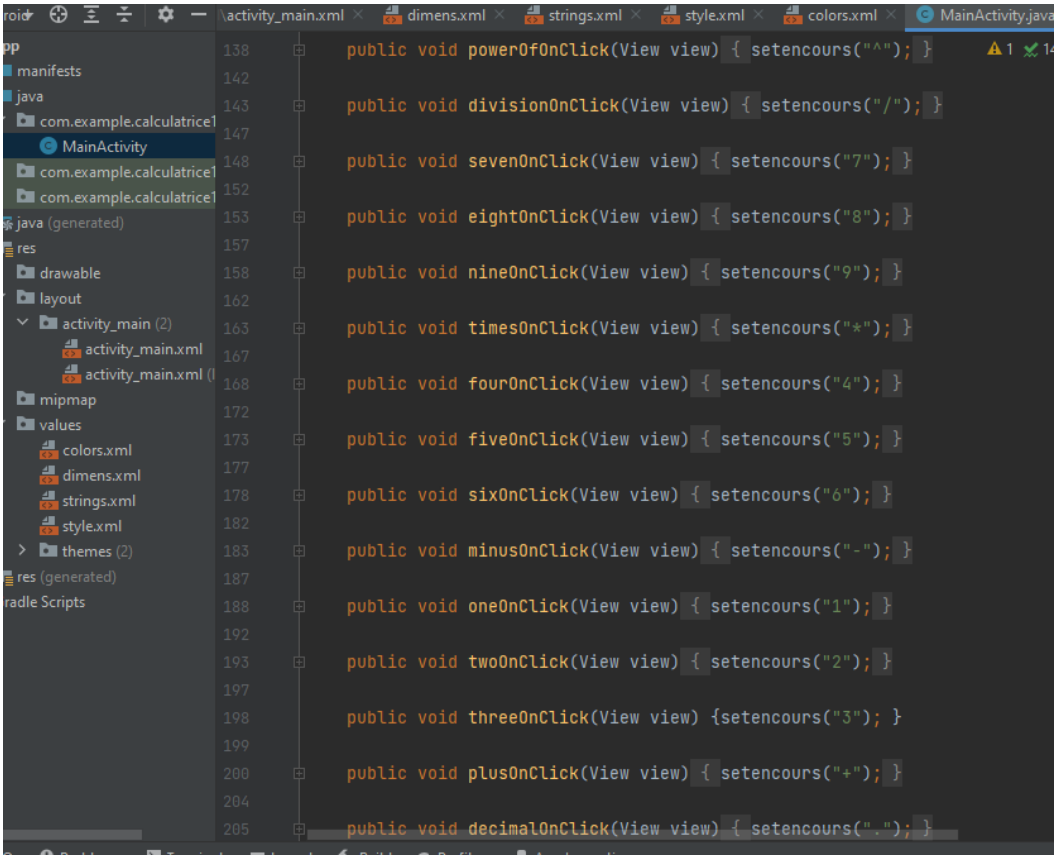
1. Déclarer les attributs nécessaires au traitement dans la classe MainActivity :

Int valeur1, Int valeur2, String operation et Boolean isOp1 (qui teste si on est sur le premier ou le deuxième opérande).

```
        TextViewOpe = (TextView)findViewById(R.id.TextViewOpe);
        TextViewResult = (TextView)findViewById(R.id.TextViewResult);
    }
```

2. Ecrire une méthode void afficher() qui affiche l'opérande en cours de saisie. Utiliser la (si nécessaire) dans les méthodes qui suivent.

1. Ecrire une méthode void setOperator(View view) pour gérer les 4 boutons d'opérateurs (+, -, * et /). Cette méthode devra prendre une View en paramètre et retrouver le bouton d'opération qui a été pressé pour agir en conséquence. Pour ce faire, vous pouvez utiliser la méthode getId() présente dans toutes les vues et qui permet de récupérer l'identifiant de la vue qui a généré l'événement.



J'ai utilisé la bibliothèque rhino

```
dependencies {

    implementation 'androidx.appcompat:appcompat:1.4.1'
    implementation 'com.google.android.material:material:1.5.0'
    implementation 'androidx.constraintlayout:constraintlayout:2.1.3'
    implementation 'io.apisense:rhino-android:1.0'
    testImplementation 'junit:junit:4.13.2'
    androidTestImplementation 'androidx.test.ext:junit:1.1.3'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.4.0'
}
```

4. Écrire une méthode void clean() qui initialise l'affichage.

```
public void clearOnClick(View view)
{
    TextViewOpe.setText("");
    encours = "";
    TextViewResult.setText("");
    GueillemetGauche = true;
}
```

5. Associez ces méthodes aux boutons concernés dans le fichier XML.

```
<androidx.appcompat.widget.AppCompatButton
    android:id="@+id/ButtonPlus"
    android:layout_width="103dp"
    android:layout_height="103dp"
    android:layout_marginBottom="1dp"
    android:background="@null"
    android:onClick="plusOnClick"
    android:text="+"
    android:textColor="@color/orange"
    android:textSize="25sp"
    app:layout_constraintBottom_toTopOf="@+id/ButtonEgal"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toEndOf="@+id/Button3"
    app:layout_constraintTop_toBottomOf="@+id/ButtonMoins" />

<androidx.appcompat.widget.AppCompatButton
    android:id="@+id/ButtonVirgule"
    android:layout_width="0dp"
    android:layout_height="0dp"
    android:background="@null"
    android:onClick="decimalOnClick"
    android:text="."
    android:textColor="@color/white"
    android:textSize="25sp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toStartOf="@+id/Button0"
```