Créer une application Calculatrice

1. Introduction

Après avoir réalisé l'application **Hello World**, il est temps de développer une application plus interactive permettant d'effectuer des opérations sur des données saisies par l'utilisateur. L'application **Calculatrice** constitue un excellent exercice pour apprendre à manipuler les composants graphiques, récupérer les valeurs saisies, gérer les événements et afficher un résultat à l'écran.

Cette application permettra à l'utilisateur de saisir deux nombres puis d'effectuer les quatre opérations arithmétiques fondamentales : **Addition**, **Soustraction**, **Multiplication** et **Division**.

> **Objectif pédagogique :** Cette application permet d'apprendre à utiliser les composants `EditText`, `Button` et `TextView`, tout en découvrant comment programmer les événements (*OnClick*) et manipuler des variables en Java.

2. Création du projet

Pour créer l'application Calculatrice :

1. Ouvrir **Android Studio**.
2. Cliquer sur **New Project**.
3. Sélectionner **Empty Activity**.
4. Nom du projet : **Calculatrice**.
5. Choisir le langage **Java**.
6. Cliquer sur **Finish**.

Android Studio génère automatiquement les fichiers principaux du projet :

- **MainActivity.java**: contient la logique de l'application.
- **activity_main.xml**: définit l'interface graphique.
- **AndroidManifest.xml**: décrit l'application Android.

3. Conception de l'interface graphique

L'interface est composée des éléments suivants :

- Deux zones de saisie (`EditText`) pour entrer les nombres.

apcpedagogie.com

- Quatre boutons correspondant aux opérations mathématiques.
- Un `TextView` pour afficher le résultat.

```
<?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"
    android:padding="20dp">

    <EditText
        android:id="@+id/editNombre1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Premier nombre"
        android:inputType="numberDecimal"/>

    <EditText
        android:id="@+id/editNombre2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Deuxième nombre"
        android:inputType="numberDecimal"/>

    <Button
        android:id="@+id/btnAddition"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Addition"/>

    <Button
        android:id="@+id/btnSoustraction"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Soustraction"/>

    <Button
        android:id="@+id/btnMultiplication"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Multiplication"/>
```

apcpedagogie.com

```
    <Button
        android:id="@+id/btnDivision"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Division"/>

    <TextView
        android:id="@+id/txtResultat"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Résultat :"
        android:textSize="24sp"
        android:textStyle="bold"
        android:paddingTop="20dp"/>

</LinearLayout>
```

4. Explication des composants

- **EditText :** permet à l'utilisateur de saisir des valeurs numériques.
- **Button :** déclenche une opération lorsque l'utilisateur clique dessus.
- **TextView :** affiche le résultat du calcul.
- **LinearLayout :** organise les composants verticalement.

> **Important :** L'attribut `inputType="numberDecimal"` permet d'afficher directement un clavier numérique adapté à la saisie des nombres décimaux.

5. Développement de la logique Java

Le code Java récupère les valeurs saisies, effectue les calculs puis affiche le résultat.

```
package com.example.calculatrice;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;
```

apcpedagogie.com

```
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {

    EditText nombre1, nombre2;
    Button addition, soustraction, multiplication, division;
    TextView resultat;

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

        nombre1 = findViewById(R.id.editNombre1);
        nombre2 = findViewById(R.id.editNombre2);

        addition = findViewById(R.id.btnAddition);
        soustraction = findViewById(R.id.btnSoustraction);
        multiplication = findViewById(R.id.btnMultiplication);
        division = findViewById(R.id.btnDivision);

        resultat = findViewById(R.id.txtResultat);

        addition.setOnClickListener(v -> calculer('+'));

        soustraction.setOnClickListener(v -> calculer('-'));

        multiplication.setOnClickListener(v -> calculer('*'));

        division.setOnClickListener(v -> calculer('/'));
    }

    private void calculer(char operation){

        double a = Double.parseDouble(nombre1.getText().toString());
        double b = Double.parseDouble(nombre2.getText().toString());

        double res = 0;

        switch(operation){

            case '+':
                res = a + b;
                break;
```

apcpedagogie.com

```
            case '-':
                res = a - b;
                break;

            case '*':
                res = a * b;
                break;

            case '/':
                if(b != 0)
                    res = a / b;
                else{
                    resultat.setText("Erreur : division par zéro");
                    return;
                }
                break;
        }

        resultat.setText("Résultat : " + res);
    }
}
```

6. Analyse du code

Le fonctionnement de l'application repose sur plusieurs étapes :

1. Les composants de l'interface sont reliés au code Java grâce à `findViewById()`.
2. Chaque bouton possède un écouteur d'événement (`setOnClickListener()`).
3. Lors d'un clic, les valeurs saisies sont récupérées sous forme de texte.
4. Les chaînes de caractères sont converties en nombres grâce à `Double.parseDouble()`.
5. L'opération choisie est exécutée à l'aide d'une structure `switch`.
6. Le résultat est affiché dans le composant `TextView`.

7. Résultat obtenu

L'application permet :

- Saisir deux nombres.
- Choisir une opération mathématique.
- Afficher immédiatement le résultat.
- Détecter une division par zéro et afficher un message d'erreur.

apcpedagogie.com

> **Bonnes pratiques :** Dans une application réelle, il est recommandé de vérifier que les deux champs ne sont pas vides avant d'effectuer les calculs et d'utiliser un bloc `try-catch` afin d'éviter les erreurs liées à une saisie invalide.

8. Compétences acquises

À travers cette application, l'étudiant apprend à :

- Créer une interface utilisateur complète sous Android Studio.
- Manipuler les composants graphiques Android.
- Programmer les événements liés aux boutons.
- Convertir des données texte en valeurs numériques.
- Effectuer des calculs arithmétiques en Java.
- Afficher dynamiquement un résultat à l'écran.
- Gérer une erreur simple comme la division par zéro.

Rédigé et structuré par : **Med Khalil Kribi**

Source : https://apcpedagogie.com/creer-application-calculatrice-android-studio · téléchargé le 07/11/2026 · © APC Pédagogie · apcpedagogie.com

apcpedagogie.com