Ejercicios prácticos

Asignatura: Programación Multimedia y dispositivos móviles
Unidades: 5 y 6

Requisitos Principales:

1. Interfaz de Usuario Atractiva: Diseña una interfaz de usuario intuitiva y atractiva que permita a los usuarios realizar cálculos de manera eficiente. Debe incluir botones para dígitos (0-9), operadores aritméticos (+, -, *, /), y botones para borrar y calcular.
2. Operaciones Básicas: La calculadora debe ser capaz de realizar operaciones de suma, resta, multiplicación y división con números enteros y decimales.
3. Historial de Cálculos: Implementa una funcionalidad que permita a los usuarios ver el historial de cálculos realizados en la sesión actual. (opcional)
4. Tema Oscuro y Claro: Proporciona la opción de cambiar entre un tema oscuro y claro en la aplicación para satisfacer las preferencias de los usuarios.
5. Manejo de errores: La aplicación debe manejar errores, como la división por cero, de manera elegante y mostrar mensajes de error claros al usuario.

Entregables:

1. Código fuente de la aplicación de Android (exportar proyecto).
2. Un informe breve que describa las decisiones de diseño tomadas y los desafíos enfrentados durante el desarrollo. Con fotos

Nota Importante: Asegúrate de seguir las mejores prácticas de desarrollo de aplicaciones Android y de probar exhaustivamente tu aplicación para garantizar su funcionamiento correcto. La originalidad, la calidad de la interfaz de usuario y la funcionalidad de la aplicación serán criterios clave en la evaluación.

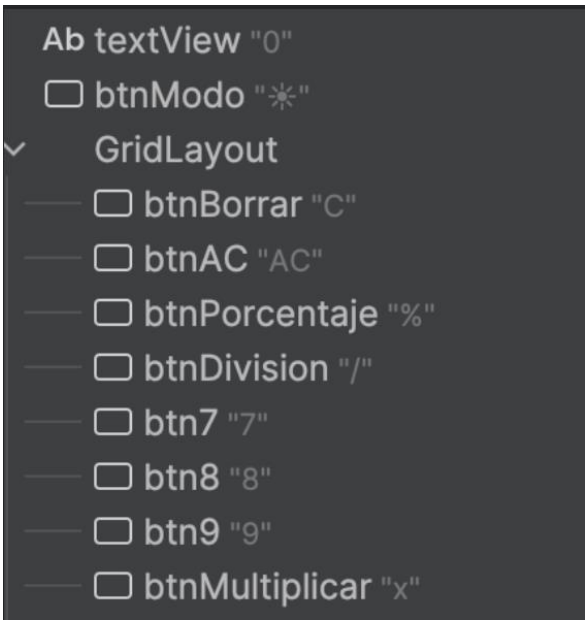
Código en GitHub:

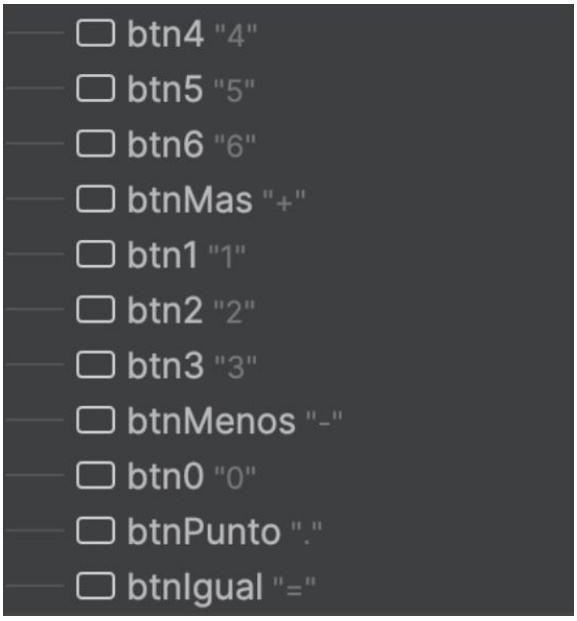
https://github.com/robertodfj/Ejercicios-2-DAM/tree/main/programacion_android/Ej2Calculadora

Solución:

1. **Interfaz de usuario atractiva:**

Utilizamos un grid layout para realizar la organización de los botones y un textview para mostrar la operación y resultado.





Los botones utilizados son android Material Button, estos se usan para tener el diseño redondo en estos, esta es el codigo usado para cada uno de ellos simplemente cambiando los botones de color:

```
<com.google.android.material.button.MaterialButton
   android:id="@+id/btn3"
   android:layout_width="90dp"
   android:layout_height="90dp"
   android:backgroundTint="#8B8B8B"
   android:text="3"
   android:textColor="@android:color/white"
   android:textSize="24sp"
   app:cornerRadius="100dp" />
```

El diseño final quedaría así:

2. Operaciones básicas:

```
10 usages
public void agregarNumero(String numeroPulsado){
    if (nuevoNumero) {
        nuevoNumero = false;
        operacion = numeroPulsado.toString();
        textView.setText(numeroPulsado);
    } else {
        operacion = operacion + numeroPulsado;
        textView.setText(operacion);
    }
}

1 usage
public void agregarPunto(){
    if (nuevoNumero){
        textView.setText("0.");
        operacion = "0.";
    } else {
        String operacionExistente = textView.getText().toString();
        if (!operacionExistente.contains(".")) {
            textView.setText(operacionExistente + ".");
            operacion = operacion + ".";
        }
    }
}

4 usages
public void agregarOperador(Character operadorPulsado){
    numero1 = Double.parseDouble(textView.getText().toString());
    operacion = operacion + operadorPulsado;
    textView.setText(operacion);
    operador = operadorPulsado;
    nuevoNumero = true;
}
```

```
1 usage
public void calcular(){
    numero2 = Double.parseDouble(textView.getText().toString());
    double resultado = 0;
    switch (operador){
        case '+': resultado = numero1 + numero2; break;
        case '-': resultado = numero1 - numero2; break;
        case '*': resultado = numero1 * numero2; break;
        case '/': if (numero2 != 0){
                resultado = numero1 / numero2;
            } else {
                textView.setText("Sin definir");
                nuevoNumero = true;
                return;
            } break;
    }

    textView.setText(String.valueOf(resultado));
    nuevoNumero = true;
}

2 usages
public void limpiar(){
    operacion = "";
    nuevoNumero = true;
    textView.setText("0");
}
```

Este es el apartado dedicado a la funcionabilidad de la aplicación donde tenemos principalmente las constantes numero1, char y numero2, esto sirve para realizar la operación que se indique.

Para realizar esto de manera más sencilla utilizamos seudocódigo:

```
int número 1
int número 2
char operación
boolean nuevo numero

agregar numero =
si nuevo numero true
String números = numero pulsado
nuevo numero = falso
imprimir en tv
else
numeros + numero pulsado
imprimir números en tv

agregar char =
numero1 = double del string guardado
nuevo numero = true

resultado
numero2 = double del string guardado
hacer operaciones con condición case
```

Explicación del código simple:
2 Constantes numéricas en las que vamos añadiendo los números que se muestran en la pantalla, en numero1 se guarda tras pulsar el operador y el numero2 al pulsar el espacio.

Antes de realizar esto también podemos añadir decimales con el punto.

Las operaciones case se seleccionan dependiendo del operador pulsado por el usuario lo que da un resultado.

3. Historial de operaciones

```
<Spinner
   android:id="@+id/historialOperaciones"
   android:layout_width="wrap_content"
   android:layout_height="48dp"
   android:padding="8dp"
   tools:ignore="MissingConstraints"
   tools:layout_editor_absoluteX="0dp"
   tools:layout_editor_absoluteY="158dp"
   android:background="#F69B6A"/>
```

Ademas de este codigo para incluir la parte visual, añadimos lo importante; la logica, seria lo sigueinte:

Guardamos en un array las operaciones completas pero si tenemos 3 borramos la primera, asi el historial de operaciones queda maximo de 3.

```
public void agregarHistorial(String ope){
    if (historial.size() == 3){
        historial.remove( index: 0);
    }
    historial.add(ope);
    adapter.notifyDataSetChanged();
}
```

```
public void calcular(){
    numero2 = Double.parseDouble(textView.getText().toString());
    double resultado = 0;
    switch (operador){
        case '+': resultado = numero1 + numero2; break;
        case '-': resultado = numero1 - numero2; break;
        case '*': resultado = numero1 * numero2; break;
        case '/': if (numero2 != 0){
                resultado = numero1 / numero2;
            } else {
                textView.setText("Sin definir");
                nuevoNumero = true;
                return;
            } break;
    }

    textView.setText(String.valueOf(resultado));
    nuevoNumero = true;
    String operacionCompleta = numero1 + "" +operador+""+numero2+ "= " + resultado;

    agregarHistorial(operacionCompleta);
}
```

4. Tema oscuro/claro

Para seleccionar el tema creamos un botón con el logo del sol para poner el modo claro y de la luna para poner el modo oscuro, este cambia tanto el color del texview que muestra la operacion como el icono y fondo.

```
public void cambioModo(){
    if (modo){
        btnModo.setText("🌙 ");
        modo = false;
        mainLayout.setBackgroundColor(getResources().getColor(android.R.color.white));
        textView.setTextColor(getResources().getColor(android.R.color.black));

    } else {
        modo = true;
        btnModo.setText("☀️ ");
        mainLayout.setBackgroundColor(getResources().getColor(android.R.color.black));
        textView.setTextColor(getResources().getColor(android.R.color.white));
    }
}
```

5. Manejo de errores en la división:

Para asegurar el correcto funcionamiento, incluimos un if para comprobar si el número se divide entre 0, a esto responde con Sin definir en el textview.

```
public void calcular(){
    numero2 = Double.parseDouble(textView.getText().toString());
    double resultado = 0;
    switch (operador){
        case '+': resultado = numero1 + numero2; break;
        case '-': resultado = numero1 - numero2; break;
        case '*': resultado = numero1 * numero2; break;
        case '/': if (numero2 != 0){
                resultado = numero1 / numero2;
            } else {
                textView.setText("Sin definir");
                nuevoNumero = true;
                return;
            } break;
    }

    textView.setText(String.valueOf(resultado));
    nuevoNumero = true;
}
```

Desafíos del proceso:

El primer desafió encontrado es la creación de los botones redondos, tras investigar por YouTube encontramos una solución fácil para implementar.

El mayor fallo obtenido en la creación del proyecto seria la función agregarNumeros, al principio no incluía el else, esto hacía que los números se pusieran 2 veces en la pantalla y el cálculo salia doble.

```
else {
    String operacionExistente = textView.getText().toString();
    if (!operacionExistente.contains(".")) {
        textView.setText(operacionExistente + ".");
        operacion = operacion + ".";
    }
```

Para la creación del código (Las funciones de la calculadora) se utilizó primero el seudocódigo para guiarnos de manera más fácil en el desarrollo.