Crear Mensajes Emergentes:

A continuación vamos a ver como crear mensajes emergentes, que nos pueden servir para alertar o informar al los usuarios de la aplicación.

Para crear estos mensajes emergentes debemos ir a la parte lógica de nuestra APP.

La estructura para crear y visualizar los mensajes emergentes es la siguiente:

Toast.makeText(Context, string,Int duracion).show();

Context--Pantalla donde vamos a mostrar nuestro mensaje.

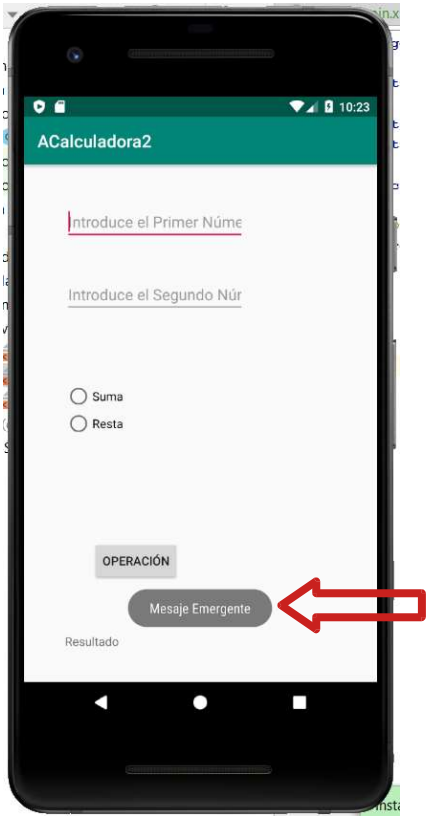
string--Mensaje a mostrar.

Int duracion--Duración del mensaje en pantalla.

show()-- Para mostrar el mensaje.

Ejemplo: `Toast.makeText(this,"Mesaje Emergente",Toast.LENGTH_LONG).show();`

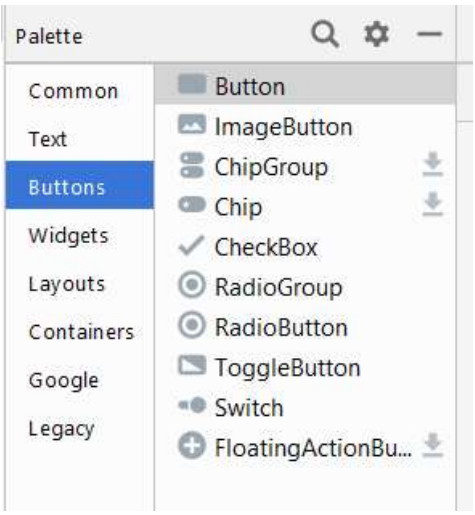
Necesario importar la libreria: `import android.widget.Toast;`



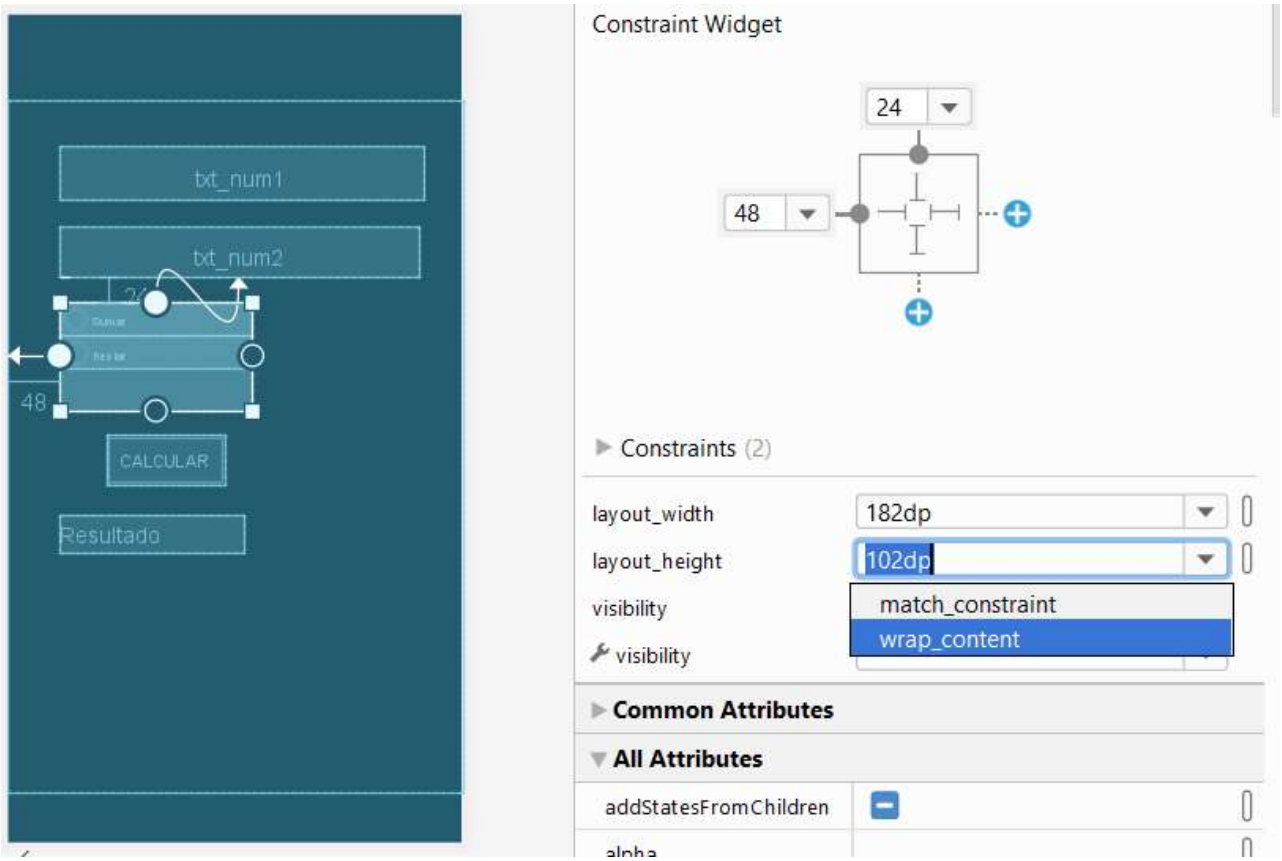
Utilización de RadioButton:

Para hacer uso de los RadioButton primero debemos añadir RadioGrup. El RadioGrup es como una caja que va ha contener los RadioButton que vayamos a utilizar.

Parte grafica:



Una vez arrastrados los RadioButton vemos que en el RadioGrup nos sobra un espacio que en este ejemplo no vamos a utilizar. Para eliminar este espacio vamos a tener que ir al apartado de atributos del RadioGrup.



Parte lógica:

Como ya sabemos cuando trabajamos con entornos gráficos en Java, es necesario que indiquemos que componentes vamos a utilizar en nuestro programa.

En este caso vamos a indicar los componentes de tipo TextView, EditView y los RadioButtons. Los RadioGrup no es necesario que los indiquemos pues solo son una caja que va ha contener los RadioButtons.

```
public class MainActivity extends AppCompatActivity {

    private EditText et1,et2;
    private TextView tv1;
    private RadioButton rb1,rb2;

    @Override
```

Una vez ya tenemos declarados nuestros componentes, ahora toca guardar los valores que vamos a introducir en los EditText, el valor que vamos a mostrar en el TextView y el valor de los RadioButton.

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

    et1 = (EditText) findViewById(R.id.txt_num1);
    et2 = (EditText) findViewById(R.id.txt_num2);
    tv1 = (TextView) findViewById(R.id.txt_Resultado);
    rb1 = (RadioButton)findViewById(R.id.RB_Suma);
    rb2 = (RadioButton)findViewById(R.id.RB_Resta);
}
```

Ahora deberemos implementar el método al que vamos a llamar con el Button **CALCULAR**, para implementar este método vamos hacer uso de la estructura condicional del:

if () {

}else

Se utiliza esta estructura condicional, para que en función del RadioButton elegido se ejecute una parte del código u otra.

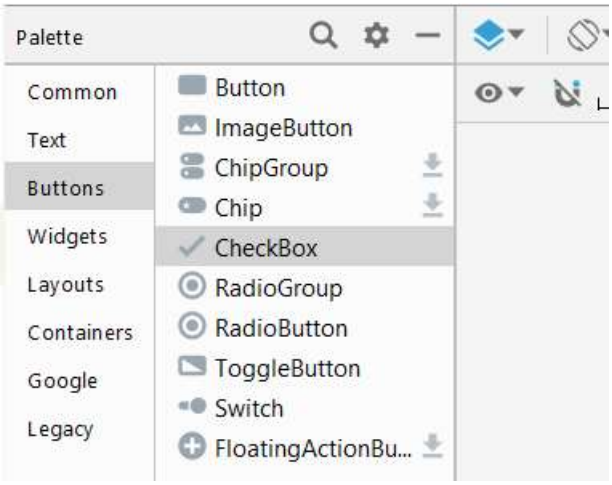
A continuación se muestra el código utilizado para el método Calcular. Para verificar si un RadioButton esta marcado o no vamos a utilizar el método ***isChecked***.

```
//Método CALCULAR

public void Calacular (View view)
{
    int num1 = Integer.parseInt(et1.getText().toString());
    int num2 = Integer.parseInt(et2.getText().toString());

    if (rb1.isChecked()==true)
    {
        int suma = num1 + num2;
        tv1.setText(String.valueOf(suma));

    }else if (rb2.isChecked()==true)
    {
        int suma = num1 - num2;
        tv1.setText(String.valueOf(suma));
    }
}
```

Utilización de CheckBox:

Como hemos visto en el anterior apartado, con los RadioButton solo podemos realizar una operación a la vez. En cambio con los CheckBox podemos realizar más de una operación al mismo tiempo.

Parte grafica:



Parte lógica:

Como ya sabemos cuando trabajamos con entornos gráficos en Java, es necesario que indiquemos que componentes vamos a utilizar en nuestro programa.

En este caso vamos a indicar los componentes de tipo TextView, EditView y los CheckBox.

```
public class MainActivity extends AppCompatActivity {

    private EditText et1,et2;
    private TextView tv1;
    private CheckBox cb1,cb2;
```

Una vez ya tenemos declarados nuestros componentes, ahora toca guardar los valores que vamos a introducir en los EditText, el valor que vamos a mostrar en el TextView y el valor de los CheckBox.

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

    et1 = (EditText) findViewById(R.id.txt_num1);
    et2 = (EditText) findViewById(R.id.txt_num2);
    tv1 = (TextView) findViewById(R.id.txt_Resultado);
    cb1 = (CheckBox)findViewById(R.id.CB_Suma);
    cb2 = (CheckBox) findViewById(R.id.CB_Resta);
}
```

Ahora deberemos implementar el método al que vamos a llamar con el Button **CALCULAR.** En este caso vamos a querer que podamos elegir un solo CheckBox o más de uno.

, para implementar este método vamos hacer uso de la estructura condicional del:

if () {

}else

Se utiliza esta estructura condicional, para que en función del CheckBox elegido se ejecute una parte del código u otra. O bien si hemos marcado más de un CheckBox si ejecute otra parte del código.

A continuación se muestra el código utilizado para el método Calcular. Para verificar si un CheckBox esta marcado o no vamos a utilizar el método ***isChecked*** igual que hemos realizado en los RadioButton.

En este caso como vamos a mostrar Texto en nuestro TextView vamos a utilizar una variable de tipo string que la inicializamos a un texto vacío.

En este ejemplo lo que he realizado son todas la operaciones fuera del método y simplemente en el método las voy llamando para mostrarlas en el TextView.

```
//Método Calcular

public void Calcular(View view) {

    int num1 = Integer.parseInt(et1.getText().toString());
    int num2 = Integer.parseInt(et2.getText().toString());
    int suma = num1 + num2;
    int resta = num1 - num2;

    String resultado = "";

    if (cb1.isChecked() == true) {
        if (cb2.isChecked() != true) {
            resultado = "La Suma es: " + suma + "\n";
        } else if (cb2.isChecked() == true) {
            resultado = "La Suma es: " + suma + "\n" + "La Resta es: " + resta + "\n";
        }

    } else if (cb2.isChecked() == true) {
        resultado = "La Resta es: " + resta + "\n";
    }
    tv1.setText(resultado);
}
```

EJERCICIOS:

1-Crear APP con la siguiente apariencia:

En función de la selección, se realizara una operación u otra (Suma, Resta, Multiplicación, División).

Hay que mostrar un mensaje emergente cuando intentemos dividir un número entre cero. El mensaje debe advertir que la operación no es posible.

El Nombre de la aplicación debe ser ***ACalculadora2*** y debéis minimizar al máximo todos los Errores/Warnings que aparezcan en la parte de diseño. Una vez creado subir proyecto al Github.

2-Crear APP con la siguiente apariencia: (EJERCICIO DE AMPLIACIÓN)

En función de la selección, se realizara una operación u otra (Suma, Resta, Multiplicación, División) o más de una de ellas.

Hay que mostrar un mensaje emergente cuando intentemos dividir un número entre cero. El mensaje debe advertir que la operación no es posible.

El Nombre de la aplicación debe ser ***ACalculadora3*** y debéis minimizar al máximo todos los Errores/Warnings que aparezcan en la parte de diseño. Una vez creado subir proyecto al Github.