

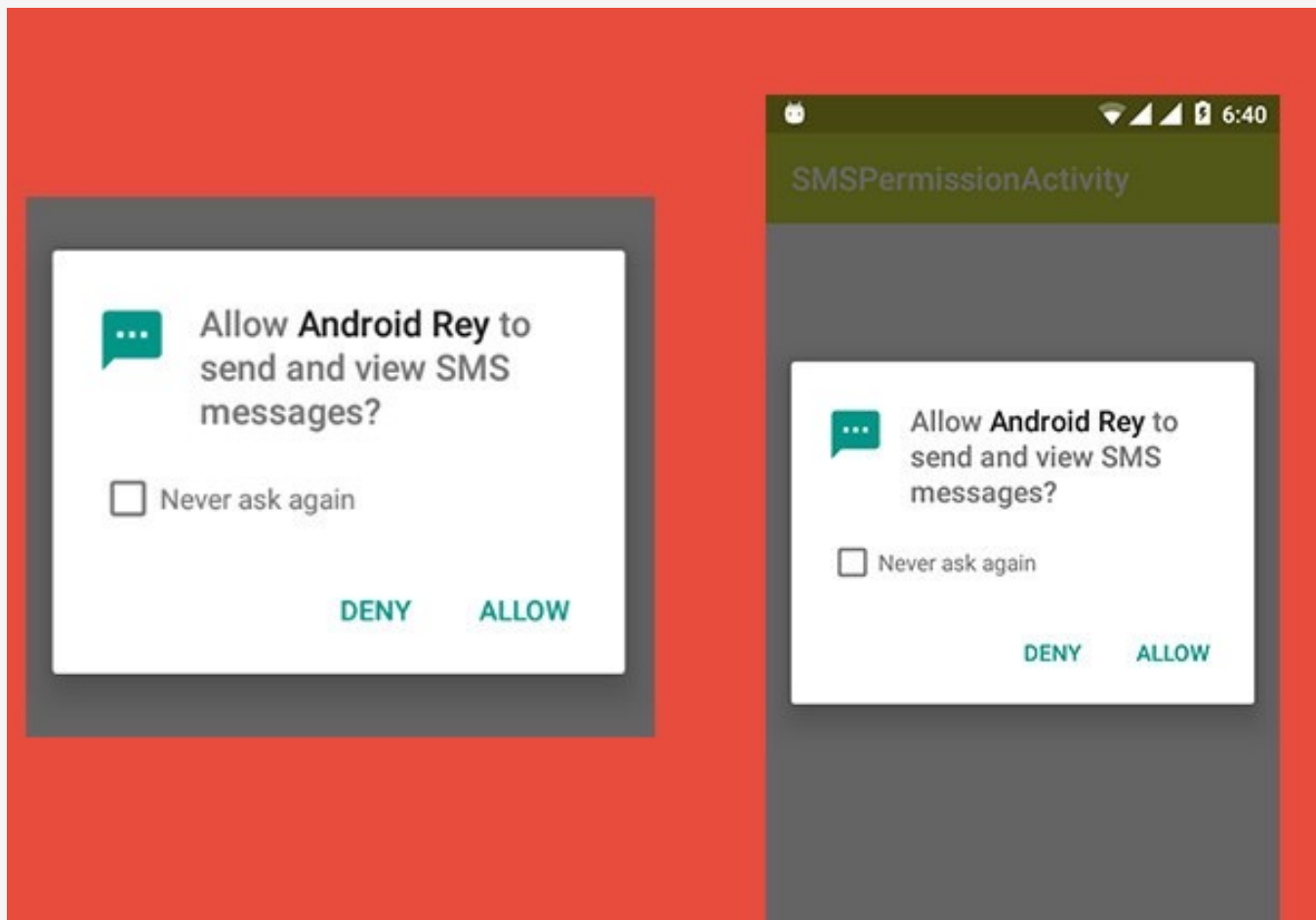
# Interfaz de Usuario y Navegación 1

Clase Nro. 4

Teófilo Copa F.

# Gestión de Permisos en tiempo de ejecución

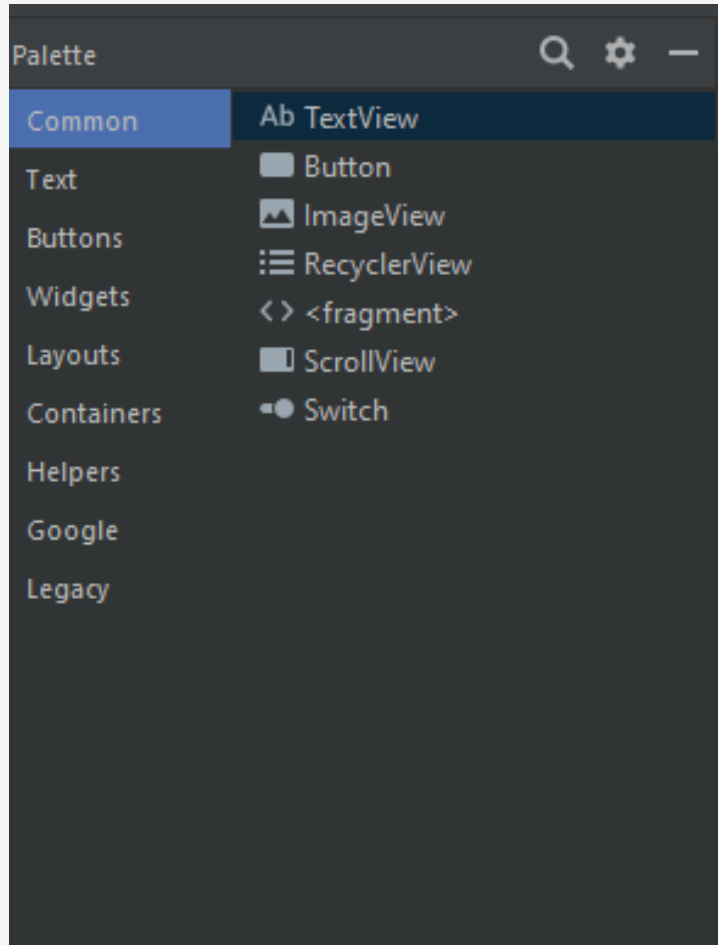
---



Los permisos de tiempo de ejecución, también conocidos como permisos peligrosos, le otorgan a tu app **acceso adicional** a datos restringidos y permiten que tu app realice acciones restringidas que afectan de manera más considerable el sistema y otras apps.

# Widgets

---



- Son elementos diseñados para que el usuario interactúe con ellos.
- Comprende desde: campos de entrada, botones, spinners, Contenedores de imágenes, barras de progreso, etc.

# TextView

---

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:id="@+id/text_view_id"
        android:layout_height="wrap_content"
        android:layout_width="wrap_content"
        android:text="@string/hello" />
</LinearLayout>
```

```
public class MainActivity extends Activity {

    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        final TextView helloTextView = (TextView) findViewById(R.id.text_view_id);
        helloTextView.setText(R.string.user_greeting);
    }
}
```

Es un elemento que se utiliza para mostrar textos.

# ImageView

---

```
<LinearLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent">
  <ImageView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:src="@drawable/my_image"
    android:contentDescription="@string/my_image_description"
  />
</LinearLayout>
```

- Muestra recursos de tipo imagen.
- Para cargar imágenes desde la red de forma optima se recomienda el uso e la librería **Glide**

```
repositories {  
    google()  
    jcenter()  
}  
  
dependencies {  
    implementation 'com.github.bumptech.glide:glide:4.12.0'  
    annotationProcessor 'com.github.bumptech.glide:compiler:4.12.0'  
}
```

```
// For a simple view:  
@Override public void onCreate(Bundle savedInstanceState) {  
    ...  
    ImageView imageView = (ImageView) findViewById(R.id.my_image_view);  
  
    Glide.with(this).load("http://goo.gl/gEgYUd").into(imageView);  
}
```

- Es un framework eficiente para trabajar con de imágenes.
- Permite obtener, decodificar y visualizar imágenes.

# Button

---

```
<Button
    android:id="@+id/button_id"
    android:layout_height="wrap_content"
    android:layout_width="wrap_content"
    android:text="@string/self_destruct" />
```

```
public class MyActivity extends Activity {
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

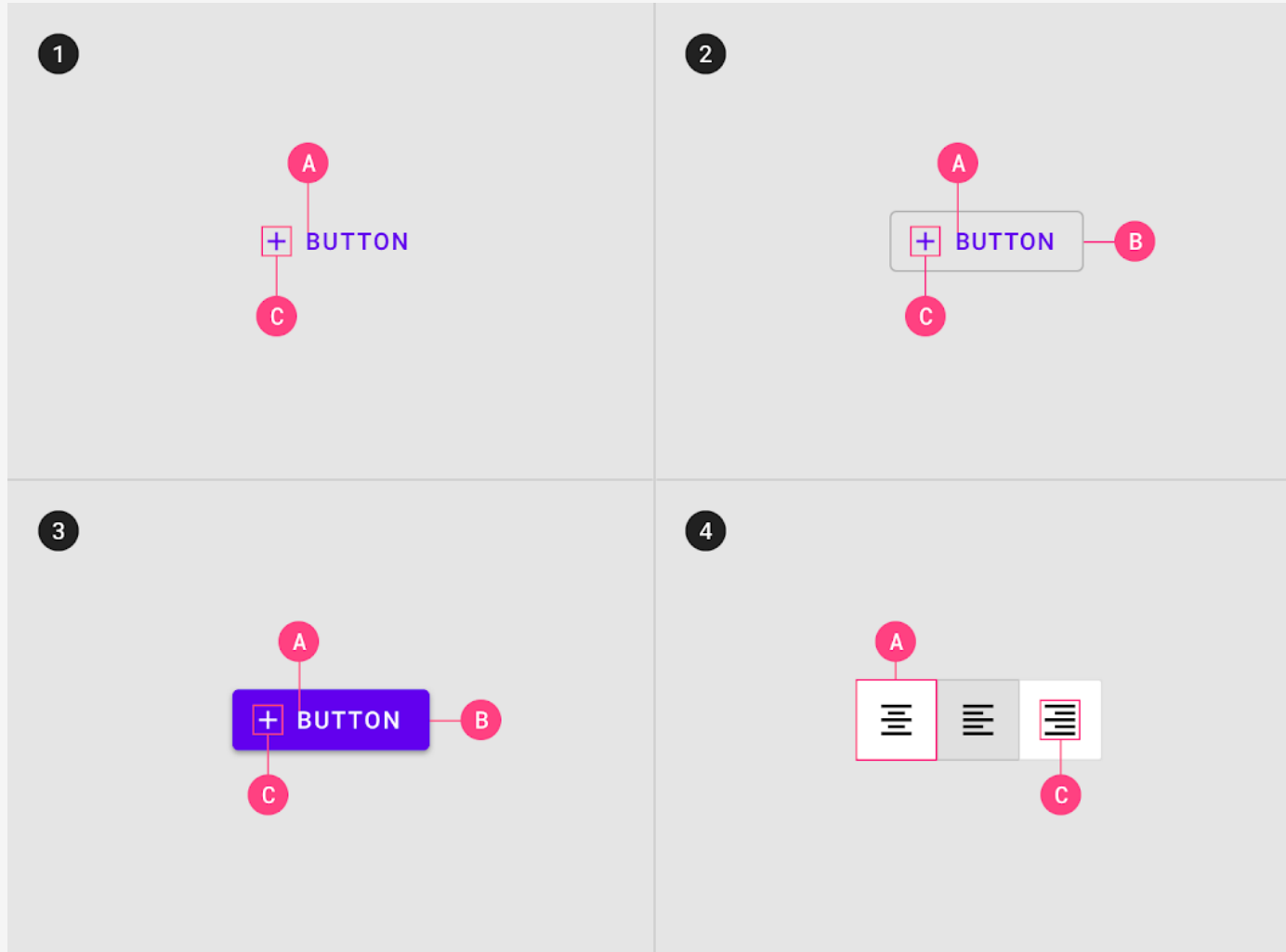
        setContentView(R.layout.content_layout_id);

        final Button button = findViewById(R.id.button_id);
        button.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                // Code here executes on main thread after user presses button
            }
        });
    }
}
```

Es un elemento de la UI en el que el usuario puede tocar o hacer clic para realizar una acción.

# Tipos de botones

---



1. Boton de texto
2. Boton con contorno
3. Botón con contenido
4. Boton de alternancia



# EditText

```
<EditText
    android:id="@+id/plain_text_input"
    android:layout_height="wrap_content"
    android:layout_width="match_parent"
    android:inputType="text" />
```

- Elemento de UI que permite ingresar y modificar textos.
- Se puede personalizar el tipo de dato a ingresar

Constante	Descripción
<code>text</code>	Recibe texto plano simple
<code>textPersonName</code>	Texto correspondiente al nombre de una persona
<code>textPassword</code>	Protege los caracteres que se van escribiendo con puntos
<code>numberPassword</code>	Contraseña de solo números enmascarada con puntos
<code>textEmailAddress</code>	Texto que será usado en un campo para emails
<code>phone</code>	Texto asociado a un número de teléfono
<code>textPostalAddress</code>	Para ingresar textos asociados a una dirección postal
<code>textMultiLine</code>	Permite múltiples líneas en el campo de texto
<code>time</code>	Texto para determinar la hora
<code>date</code>	Texto para determinar la fecha
<code>number</code>	Texto con caracteres numéricos
<code>numberSigned</code>	Permite números con signo
<code>numberDecimal</code>	Para ingresar números decimales

# Escuchar cambios de texto

---

```
EditText editText1 = findViewById(R.id.editText1);
editText1.addTextChangedListener(new TextWatcher() {

    @Override
    public void beforeTextChanged(CharSequence s, int start, int count, int after) {

    }

    @Override
    public void onTextChanged(CharSequence s, int start, int before, int count) {
        |
    }







    @Override
    public void afterTextChanged(Editable s) {

    }

});
```

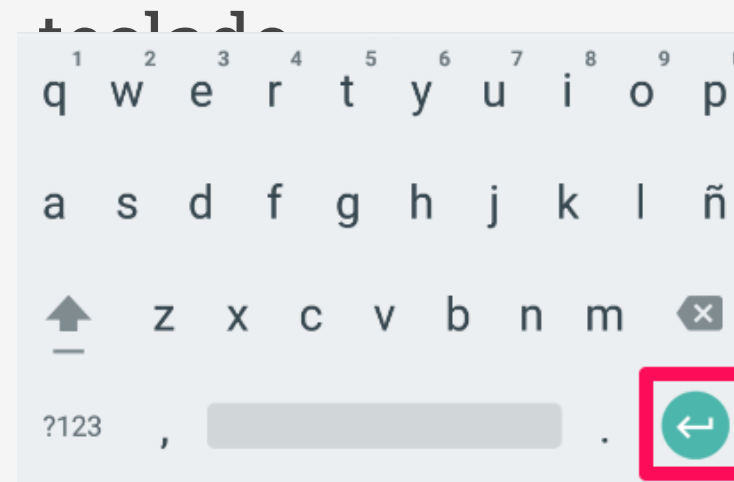
Se debe implementar la  
clase **TextWatcher**

# Métodos de entrada (ImeOptions)

Constante	Descripción	Icono
<code>actionGo</code>	Representa la acción de ejecutar alguna tarea que llevará al usuario a determinados resultados	
<code>actionSearch</code>	Especifica que realizará una búsqueda con el texto que se acaba de agregar al campo de texto	
<code>actionSend</code>	Se usa para indicar que se realizará una operación de envío de un contenido asociado al contenido del campo	
<code>actionNext</code>	Tecla para realizar una operación del tipo «siguiente». Normalmente se usa para asignar el foco al <code>TextField</code> posterior	
<code>actionDone</code>	Determina que se ha llevado a cabo satisfactoriamente la edición cerrando el teclado virtual	
<code>actionPrevious</code>	Acción que lleva al usuario a un campo previamente aceptado.	

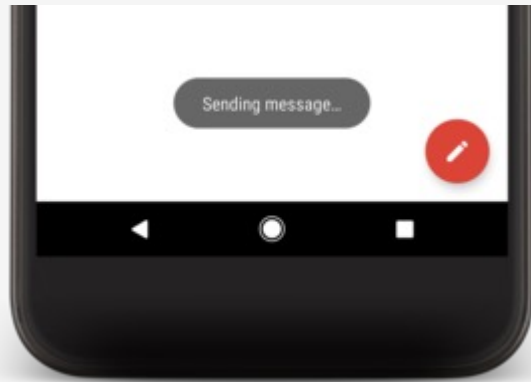
```
<EditText
    android:id="@+id/campo_mensaje"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:imeOptions="actionSend"
    android:imeActionLabel="Enviar"
    android:inputType="text"
    android:singleLine="true" />
```

- Permite configurar el icono de la acción en el



# Avisos (Toast)

---

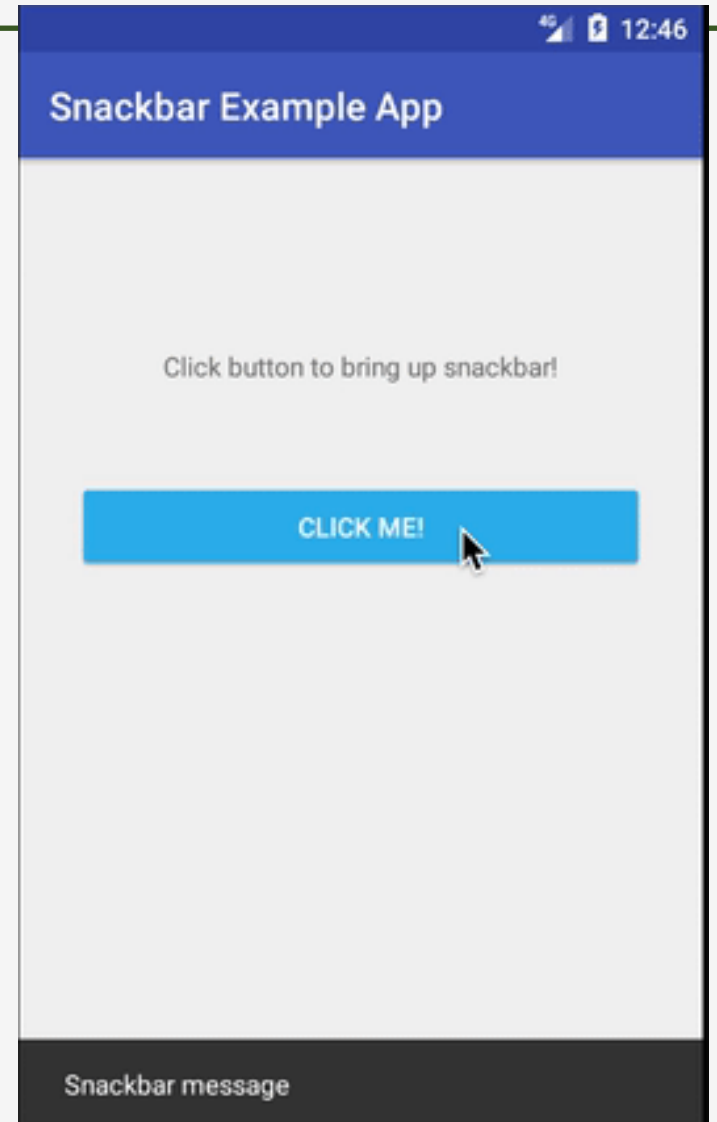


Proporciona  
información simple  
sobre una acción  
en una pequeña  
ventana emergente

```
Context context = getApplicationContext();  
CharSequence text = "Hello toast!";  
int duration = Toast.LENGTH_SHORT;  
  
Toast toast = Toast.makeText(context, text, duration);  
toast.show();
```

# Mensajes Emergentes (SnackBar)

```
public class MainActivity2 extends AppCompatActivity {
    View view;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main2);
        Button btn1 = findViewById(R.id.btn1);
        view = findViewById(android.R.id.content);
        btn1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Snackbar.make(v, text: "Este es un mensaje que aparecerá en el SnackBar", Snackbar.LENGTH_SHORT)
                    .show();
            }
        });
    }
    //txt.setTextColor(0xFF00FF);
}
```



## Mensajes Emergentes (SnackBar) con una acción

```
Snackbar s = Snackbar.make(v, text: "El item fue eliminado exitosamente!", Snackbar.LENGTH_SHORT);  
s.setAction( text: "DESHACER", new View.OnClickListener() {  
    @Override  
    public void onClick(View v) {  
        s.dismiss();  
    }  
});  
s.show();
```

My Application

TextView

Name

BUTTON

El item fue eliminado  
exitosamente!

DESHACER

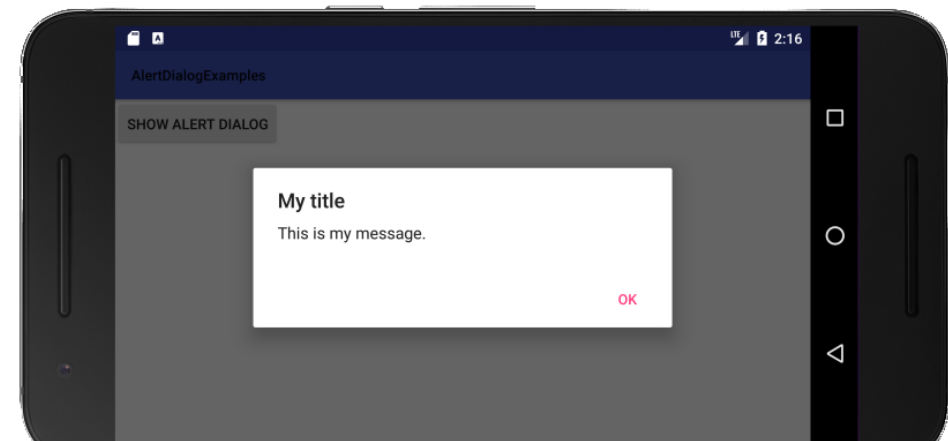
Se puede agregar una acción, para que el usuario pueda responder al mensaje

# Diálogos

---

```
AlertDialog dlg = new AlertDialog.Builder(context: MainActivity2.this).create();
dlg.setTitle("Alerta");
dlg.setMessage("Mensaje de Texto");
dlg.setButton(AlertDialog.BUTTON_NEUTRAL, text: "ACEPTAR",
    new DialogInterface.OnClickListener() {
        public void onClick(DialogInterface dialog, int which) {
            dialog.dismiss();
        }
    });
dlg.show();
```

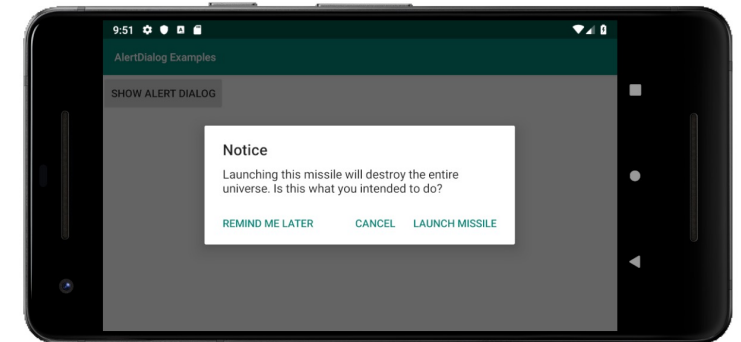
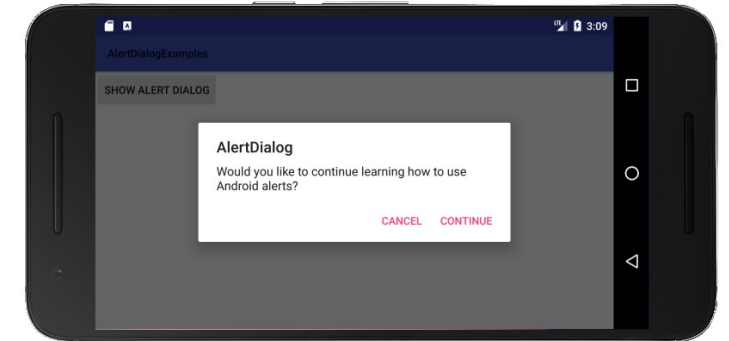
Es una ventana  
pequeña que le indica  
al usuario que debe  
tomar una decisión o  
ingresar información  
adicional



# Diálogos de confirmación

```
new AlertDialog.Builder( context: MainActivity2.this)
    .setTitle("Confirmar")
    .setMessage("Estas seguro de eliminar?")
    .setPositiveButton( text: "SI", new DialogInterface.OnClickListener() {
        public void onClick(DialogInterface dialog, int which) {

        }
    })
    .setNegativeButton( text: "NO", listener: null)
    .setIcon(android.R.drawable.ic_dialog_alert)
    .show();
```

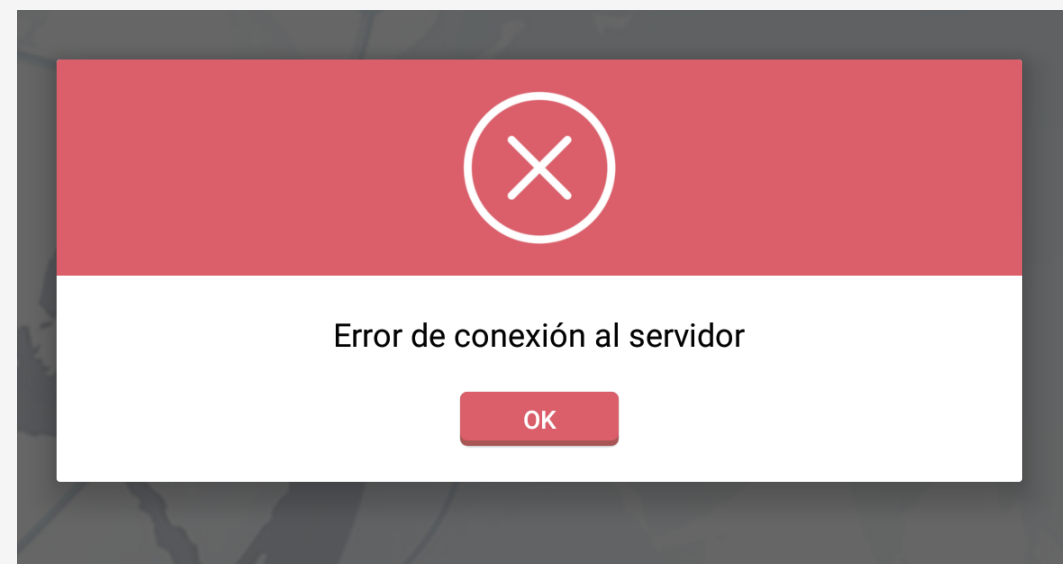
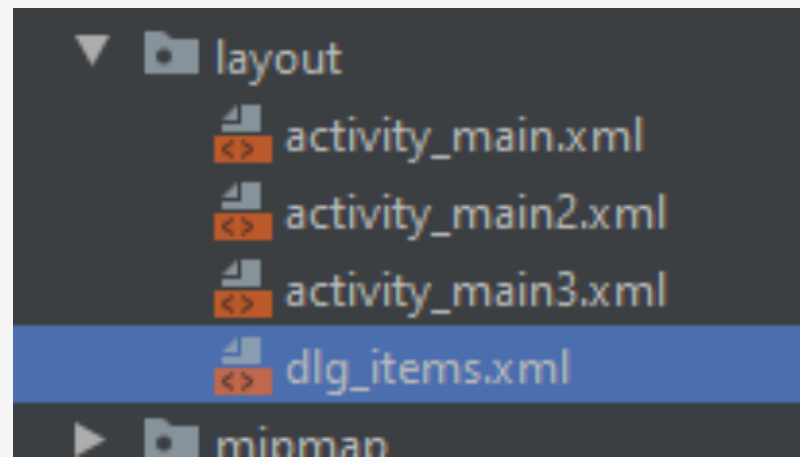




# Diálogos Personalizados

```
Dialog dlg = new Dialog( context: MainActivity2.this);
dlg.setCancelable(false);
dlg.setContentView(R.layout.dlg_items);
Button btnOK = dlg.findViewById(R.id.btnDlg);

btnOK.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        dlg.dismiss();
    }
});
dlg.show();
```



Gracias por su atención