Android Developer Fundamentals

Hola Mundo

Lección 1

1.2 Vistas, Diseño, y Recursos

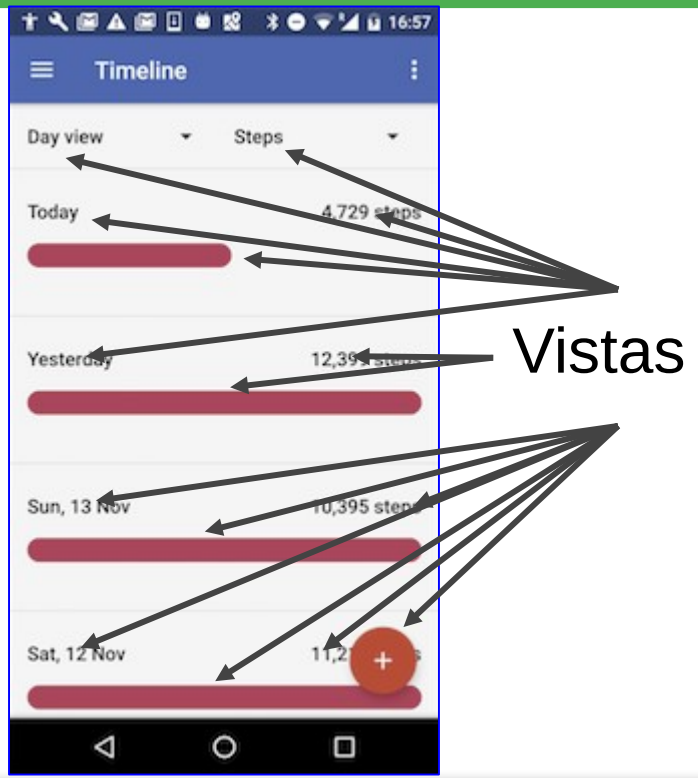
Contenidos

- Vistas, grupos de vistas y jerarquía de vistas
- Diseños en XML y código Java
- Manejo de eventos
- Recursos
- Medidas de pantalla

Vistas

Todo lo que ves es una vista

Si observa su dispositivo móvil, cada elemento de la interfaz de usuario que ve es una **Vista**.



¿Qué es una vista?

Las vistas son los bloques de construcción básicos de la interfaz de usuario de Android

- mostrar texto (clase TextView), editar texto (clase EditText)
- Botones (clase de botones), menús, otros controles
- desplazable (ScrollView, RecyclerView)
- mostrar imágenes (ImageView)
- subclase de clase View

Las vistas tienen propiedades

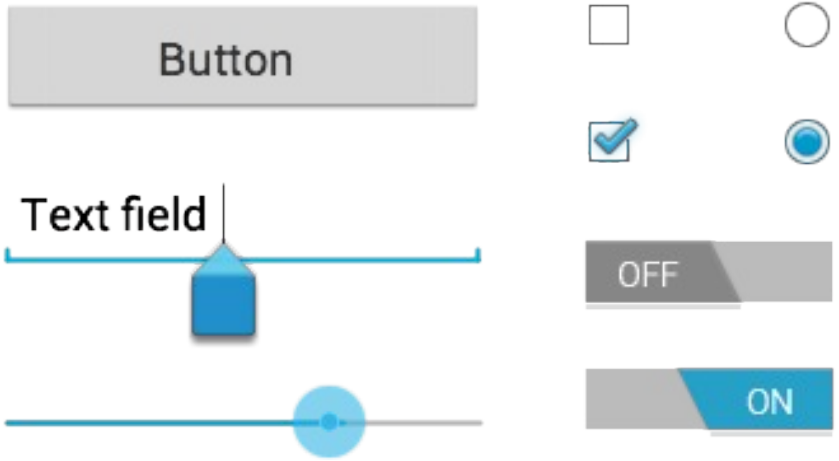
- Tener propiedades (por ejemplo, color, dimensiones, posicionamiento)
- Puede tener foque (por ejemplo, seleccionado para recibir la entrada del usuario)
- Puede ser interactivo (responder a los clicks de los usuarios)
- Puede ser visible o no
- Tener relaciones con otras vistas

Ejemplos de vistas

Button

EditText

SeekBar



CheckBox

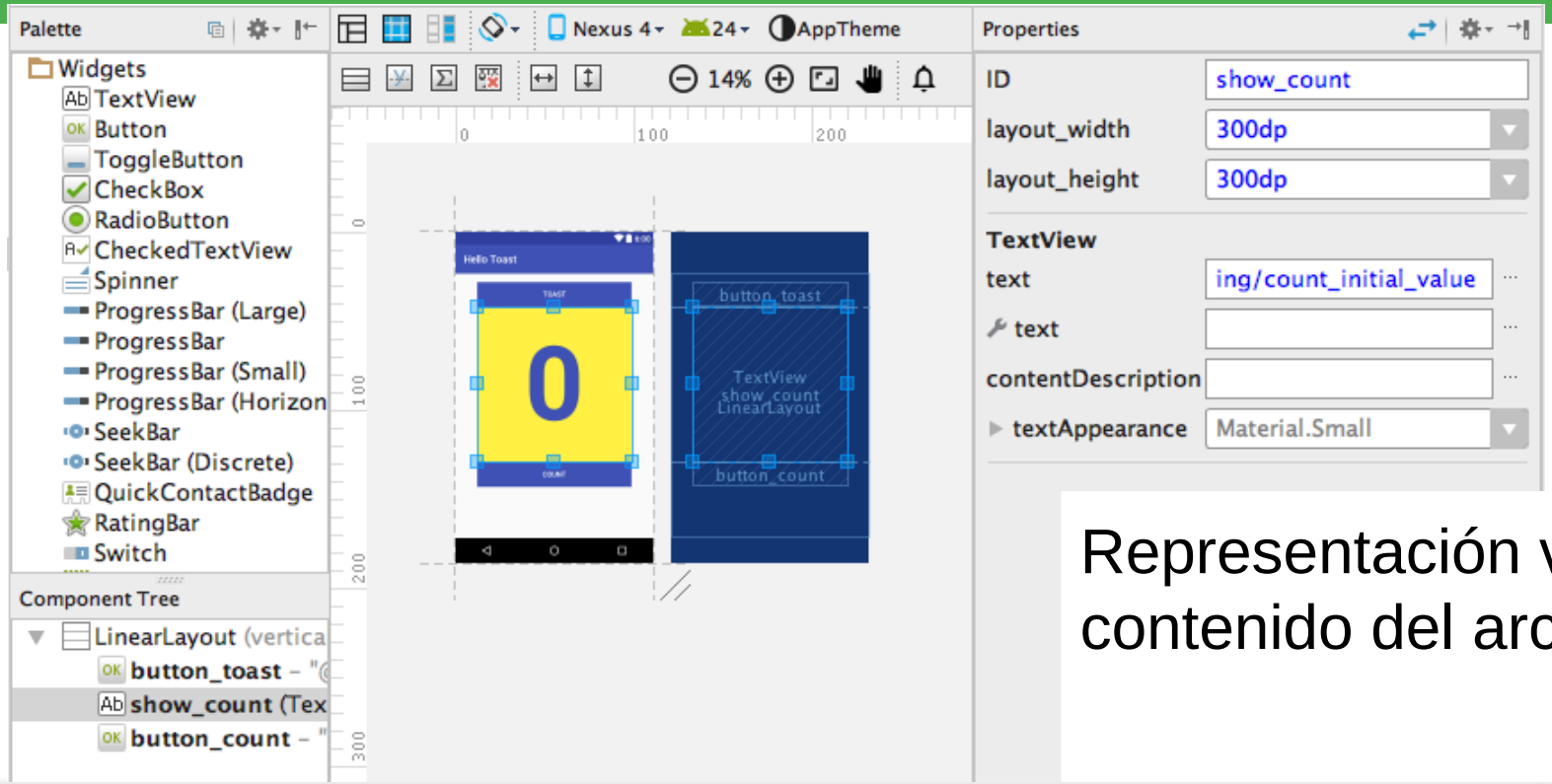
RadioButton

Switch

Creación de vistas y diseño

- Gráficamente dentro de Android Studio
- Editor XML
- De forma codificada en Java

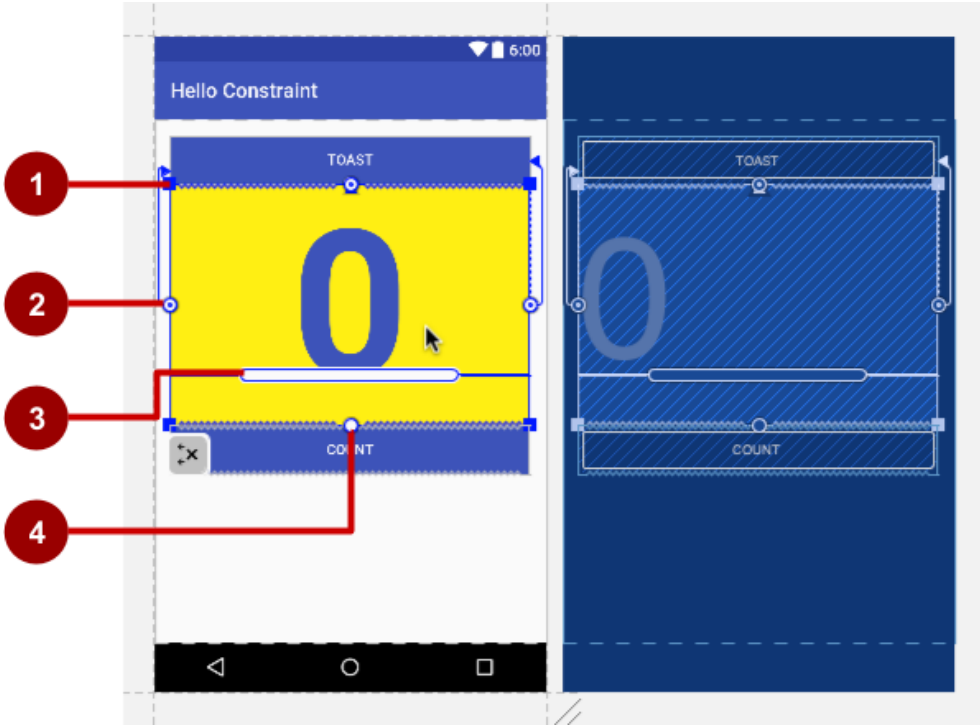
Vistas definidas en el editor de diseño



Representación visual del contenido del archivo XML

Usando el editor de diseño

1. Gestor de redimensión
2. Gestor de línea de restricción
3. Gestor de línea base
4. Gestor de restricción



Vistas definidas en XML

```
<TextView
        android:id="@+id/show_count"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="@color/myBackgroundColor"
        android:text="@string/count_initial_value"
        android:textColor="@color/colorPrimary"
        android:textSize="@dimen/count_text_size"
        android:textStyle="bold"
/>
```

Ver propiedades en XML

android:<property_name>="<property_value>"
Ejemplo: android:layout_width="match_parent"

android:<property_name>="@<resource_type>/resource_id"
Ejemplo: android:text="@string/button_label_next"

android:<property_name>="@+id/view_id"
Ejemplo: android:id="@+id/show_count"

Crear vista en código Java

context

En una actividad:

```
TextView myText = new TextView(this);
myText.setText("Display this text!");
```

¿Cuál es el contexto?

- El contexto es una interfaz para la información global sobre un entorno de aplicación
- Obtener el contexto:

```
Context context = getApplicationContext();
```

- Una actividad es su propio contexto:

```
TextView myText = new TextView(this);
```

Vistas personalizadas

- Más de 100 (!) diferentes tipos de vistas disponibles desde el sistema Android, todos hijos de la clase View
- Si es necesario, cree vistas personalizadas creando subclases de las vistas existentes o la clase View

Jerarquía ViewGroup & View

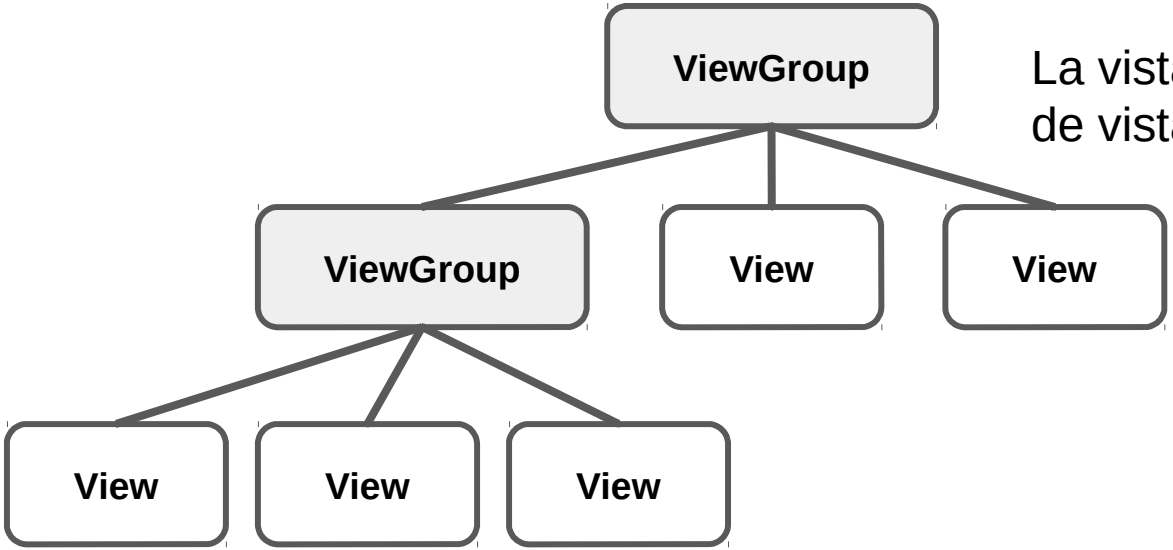
Vistas ViewGroup

Un grupo de vistas (padre) es un tipo de vista que puede contener otras vistas (hijos)

ViewGroup es la clase base para diseños y contenedores

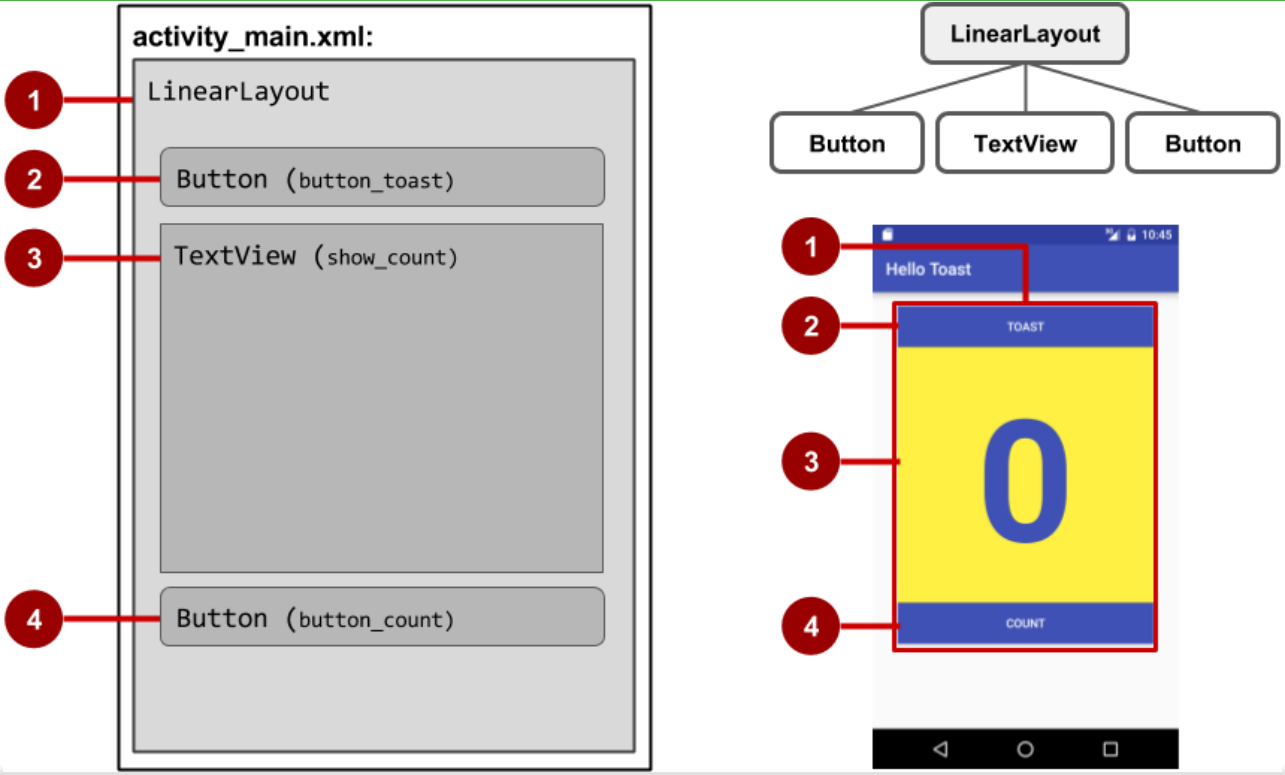
- ScrollView: vista desplazable que contiene una vista secundaria
- LinearLayout: organiza vistas en fila horizontal / vertical
- RecyclerView: "lista" desplazable de vistas o grupos de vistas

Jerarquía de grupos de vistas y vistas

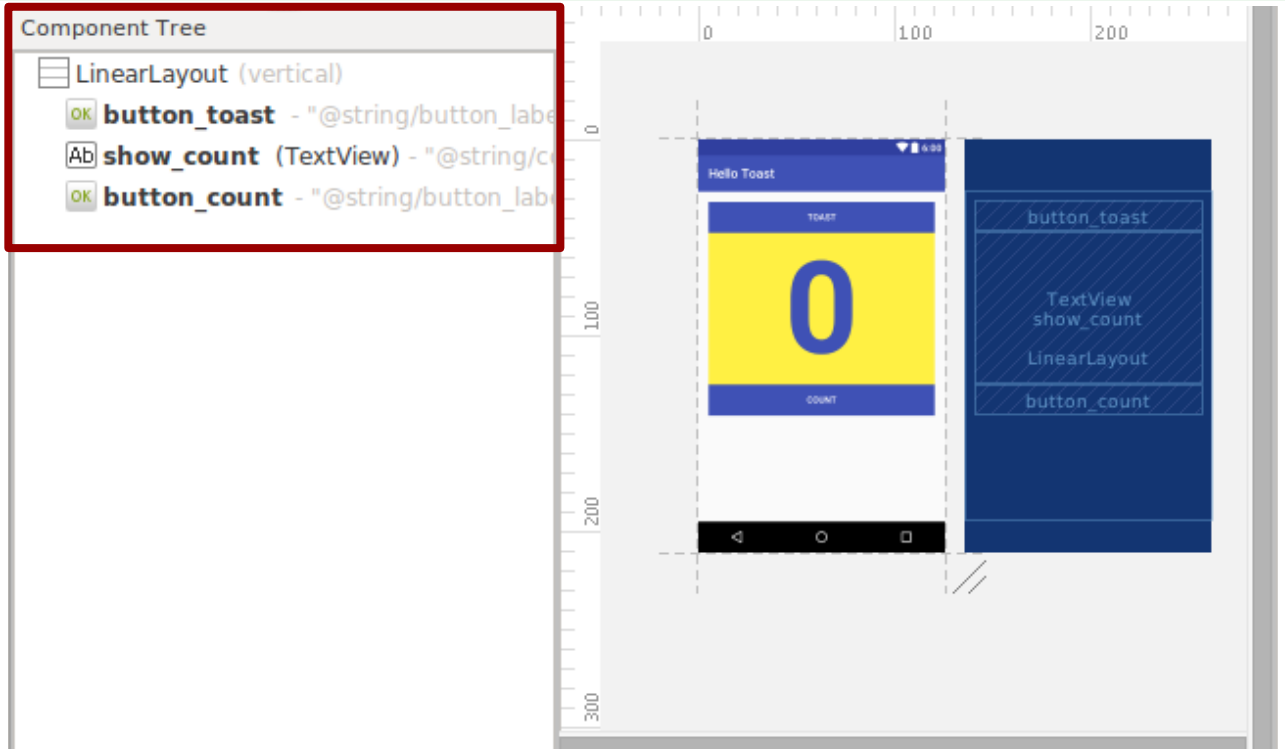


La vista raíz es siempre un grupo de vistas

Ver jerarquía y diseño de pantalla



Ver jerarquía en el árbol de componentes



Mejores prácticas para ver jerarquías

- El arreglo de la jerarquía de vistas afecta el rendimiento de la aplicación
- Utilice el menor número posible de vistas que sean las más simples
- Mantenga la jerarquía plana: limite el agrupamiento de vistas y grupos de vistas

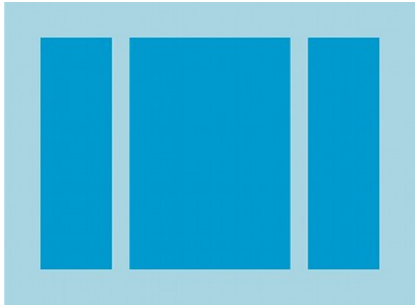
Diseños

Vistas de diseño

Diseños

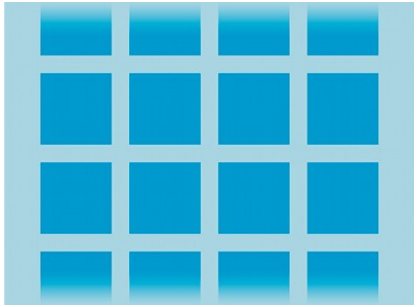
- Son tipos específicos de grupos de vista
- son subclases de ViewGroup
- Contiene hijos que son vistas
- puede estar en una fila, columna, cuadrícula, tabla, de forma absoluta

Clases de diseño común

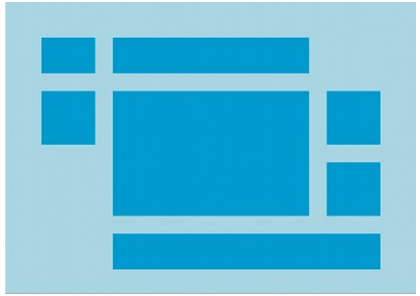


LinearLayout

RelativeLayout



GridLayout



TableLayout

Clases de diseño común

- **ConstraintLayout -** conectar vistas con restricciones
- **LinearLayout** - fila horizontal o vertical
- **RelativeLayout** - Vistas hijo relativas entre sí
- **TableLayout** – filas y columnas
- **FrameLayout** - muestra un hijo de una pila de hijos
- **GridView** - Rejilla desplazable 2D

Jerarquía de clases vs Jerarquía de diseño

- La jerarquía de clases de las Vistas utiliza herencia de clases de la orientación a objetos estándar
  - Por ejemplo, Button es-un TextView es-un View es-un Object
  - Relación superclase-subclase
- La jerarquía de diseño es cómo se organizan las vistas visualmente
  - Por ejemplo, LinearLayout puede contener botones dispuestos en una fila
  - Relación padre-hijo

Diseño creado en XML

```
<LinearLayout
  android:orientation="vertical"
  android:layout_width="match_parent"
  android:layout_height="match_parent">
    <EditText
        ... />
    <Button
        ... />
</LinearLayout
```

Diseño creado en código de actividad Java

```
LinearLayout linearL = new LinearLayout(this);
linearL.setOrientation(LinearLayout.VERTICAL);

TextView myText = new TextView(this);
myText.setText("Display this text!");

linearL.addView(myText);
setContentView(linearL);
```

Configuración de ancho y alto en código Java

Establecer el ancho y alto de una vista:

```
LinearLayout.LayoutParams layoutParams =
  new Linear.LayoutParams(
     LayoutParams.MATCH_PARENT,
     LayoutParams.WRAP_CONTENT);

myView.setLayoutParams(layoutParams);
```

Manejo de eventos

Eventos

Cuando sucede algo

- En la interfaz de usuario: Click, toque, arrastre
- Dispositivo: DetectedActivity como caminar, conducir, inclinar
- Los eventos son "detectados" por el sistema Android

Gestores de Eventos

Métodos que hacen algo en respuesta a un clic

- Un método, llamado un **controlador de eventos**, es activado por un evento específico y hace algo en respuesta al evento

Manejo de clics en XML y Java

Adjuntar el gestor para ver en el diseño:

```
android:onClick="showToast"
```

Implementar el gestor en la actividad:

```
public void showToast(View view)
{
  String msg = "Hello Toast!";
  Toast toast = Toast.makeText(
        this, msg, duration);
  toast.show();
  }
}
```

Configuración de gestores de clic en Java

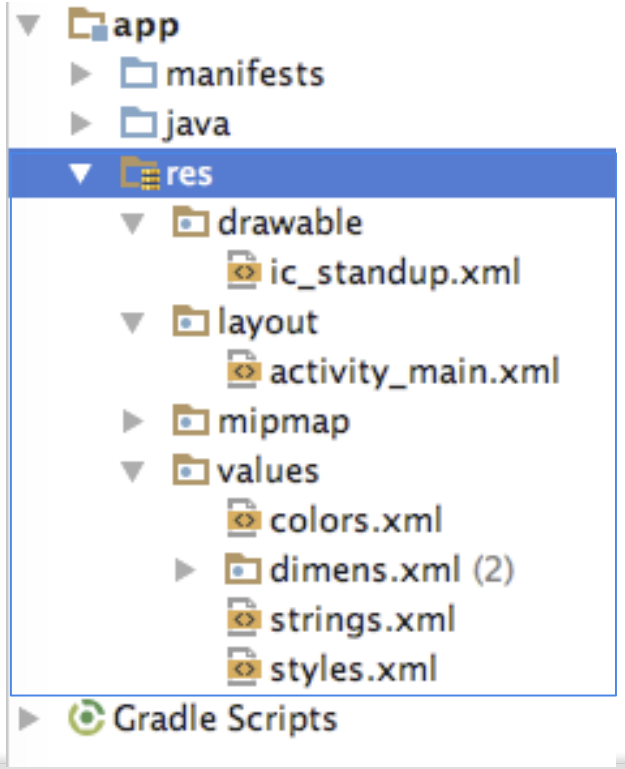
```
final Button button = (Button) findViewById(R.id.button_id);
button.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
        String msg = "Hello Toast!";
        Toast toast = Toast.makeText(this, msg, duration);
        toast.show();
    }
});
```

Recursos

Recursos

- Separe los datos estáticos del código en sus diseños
- Cadenas, dimensiones, imágenes, texto de menú, colores, estilos
- Útil para la localización

¿Dónde están los recursos en tu proyecto?



Recursos y archivos de recursos almacenados en la carpeta **res**

Consulte los recursos en el código

- Diseño:

  `R.layout.activity_main`

  ```
  setContentView(R.layout.activity_main);
  ```

- Vista:

  `R.id.recyclerview`

  ```
  rv = (RecyclerView) findViewById(R.id.recyclerview);
  ```

- String:

  In Java: `R.string.title`

  In XML: `android:text="@string/title"`

Medidas

- Píxeles independientes del dispositivo (dp) - para vistas
- Escala de píxeles independientes (sp) - para texto

No use unidades dependientes del dispositivo:

- Píxeles reales (px)
- Medida real (in, mm)
- Puntos - tipografía 1/72 pulgada (pt)

FIN