Ejercicio 1: Mejorando proyectos

Mejora los proyectos entregados con la teoría:

- P_08_Eventos_01: Si el usuario de la aplicación deja sin teclear alguno de los valores solicitados, la aplicación se rompe

  Ayuda Android: Si queremos que un elemento del layout llamado, por ejemplo, editText2 recobre el foco (el cursor vuelva a ella), basta con editText2. requestFocus();

  Repaso Java: String cadena;

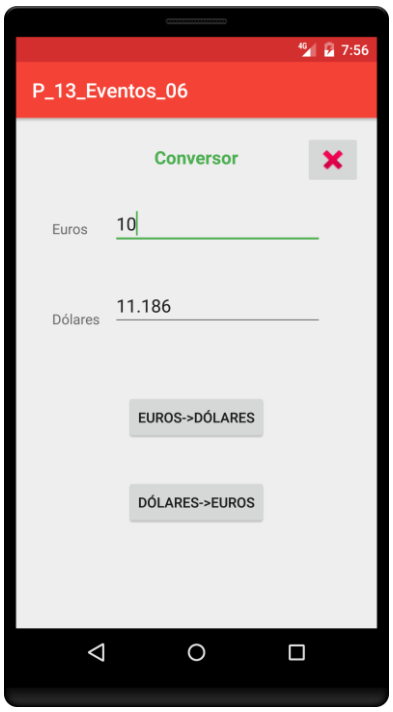
  int num_caract=cadena.trim().length()

- P_09_Eventos_02: Igual que el anterior, pero además no se ha comprobado que el divisor no sea nulo.

Ejercicio 2: Proyecto P_13_Eventos_06

Desarrolla una aplicación que calcule conversiones euros a dólar y viceversa (busca la cotización de hoy).

Nuevo concepto con el que trabajas: la vista ImageButton (recuerda que puedes usar recursos del sistema para la imagen)



Las aplicaciones Android no suelen tener un botón de Finalizar (ya veremos el motivo). Aquí nos interesa para que uses el método finish() para terminar la actividad.

Ejercicio 2: P_14_Eventos_07

Mediante botones queremos poner/quitar transparencia a una imagen.

Lo primero que tendremos que hacer es conseguir los recursos alternativos para nuestro drawable. Con la imagen puede usar el software que prefieras (Gimp, Photoshop,etc.) para editarla y guardar en los respectivos tamaños; pero es muy laborioso, lo más sencillo es que instales el plugin "Android drawable Importer" en AS. Es genial ☺!. Instrucciones de instalación y uso.

Desarrolla una aplicación que aplique transparencia a una imagen (método setImageAlpha(): setImageAlpha(0) transparente, setImageAlpha(255) opaco, un valor intermedio semitransparente).

Nuevo concepto Build.VERSION.SDK_INT: setImageAlpha() fue introducido en API 16, observa las recomendaciones del IDE si has escogido una versión menor como mínima y acéptalas.

```
if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.JELLY_BEAN)
    imageView.setImageAlpha(0);
else
    imageView.setAlpha(255);
```

Ejercicio 3:

Mejora la aplicación anterior para conseguir cambiar el color de fondo. Pista:

```
public void onCheckedChanged(RadioGroup group, int checkedId) {
   RelativeLayout fondo = (RelativeLayout) findViewById(R.id.activity_main);
   int color;
   if (checkedId == R.id.radioButton)
      color = ContextCompat.getColor(getApplicationContext(), R.color.miVerde);
   else
      color = Color.WHITE;
   fondo.setBackgroundColor(color);
}
```

(Conceptos nuevos: métodos setBackgroundColor() y getColor())

Ejercicio 4: P_15_Eventos_08 (Botones personalizados)

1. Crea el fichero boton.xml en la carpeta res/drawable/. Para ello puedes utilizar el asistente *New/Android Resource File* y pon en File: "boton" y selecciona en tipo Drawable. Escoge como elemento raíz selector. Reemplaza el código por el siguiente (aparecen errores porque todavía no tenemos ficheros de imágenes guardados en la carpeta drawable):

```
<?xml  version="1.0"  encoding="utf-8"?>
<selector xmlns:android="http://schemas.android.com/apk/res/android">
   <item android:drawable="@drawable/boton_pulsado"
      android:state_pressed="true"/>
   <item android:drawable="@drawable/boton_con_foco"
      android:state_focused="true"/>
   <item android:drawable="@drawable/boton_normal"/>
</selector>
```

   Este XML define un recurso único gráfico (drawable) que cambiará en función del estado del botón. El primer <item> define la imagen usada cuando se pulsa el botón, el segundo <item> define la imagen usada cuando el botón tiene el foco (cuando el botón está seleccionado con la rueda de desplazamiento o las teclas de dirección), el tercero la imagen en estado normal.

   NOTA: *El orden de los elementos <item> es importante. Cuando se va a dibujar se recorren los ítems en orden hasta que se cumpla una condición. Debido a que "boton_normal" es el último, sólo se aplica cuando las condiciones state_pressed y state_focused* no *se cumplen.*

2. Descarga las tres imágenes que aparecen a continuación. El nombre que ha de tener cada fichero aparece debajo:

   boton_normal.jpg boton_con_foco.jpg boton_pulsado.jpg

3. Arrastra/copia las imágenes a la carpeta res/drawable/ del proyecto.
4. Abre el fichero res/layout/activity_main.xml y elimina el TextView existente.
5. Arrastra una vista de tipo Button dentro del layout. Selecciona el atributo Background del botón y pulsa en el botón selector de recurso (con puntos suspensivos). Selecciona *Drawable/boton*. Modifica el atributo Text para que no tenga ningún valor. Introduce en el atributo onClick el valor sePulsa.
6. Abre el fichero *MainActivity.java* e introduce al final, antes de la última llave, el código:

```
public void sePulsa(View view){
  Toast.makeText(this,"Pulsado", Toast.LENGTH_SHORT).show();
}
```

   Toast es una ventana emergente, lo veremos a fondo más adelante!

7. Ejecuta el proyecto y verifica el resultado.

Si te estás preguntando porque no aparece el "modo foco" es debido a que los botones no pueden recibir el foco, no ocurriría lo mismo con los EditText