Seminario de Lenguajes Android

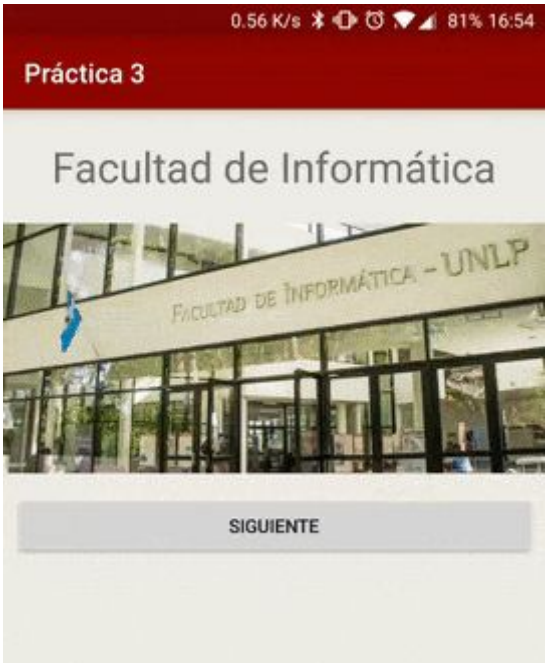
Práctica 3

Nota: *En los ejercicios que contienen código se recomienda copiarlo desde la siguiente URL:* *https://gist.github.com/gcaseres/4d46cff85068a662ee11f41a494959d8*

1. Modificar el ejercicio 12 de la práctica 2 para que, al girar el dispositivo, se mantenga el número mostrado en pantalla. Utilice los métodos `onSaveInstanceState/onRestoreInstanceState`.

2. Implemente la siguiente Activity con cuatro imágenes y un título, usando un ScrollView.

3. Modificar el ejercicio anterior para que muestre una única imagen.
Cada vez que se clickea el botón “Siguiente”, se deberá reemplazar la imagen.
Investigar el uso del mensaje `setImageResource` de la clase ImageView.

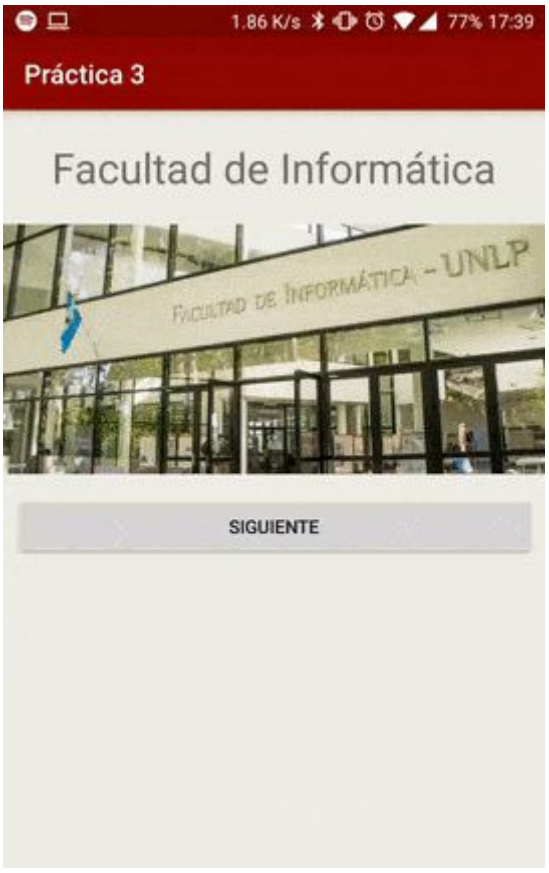


Al girar la pantalla, no debe cambiarse la imagen y el layout debe visualizarse correctamente.

Nota: Puede almacenar las referencias a las imágenes en un arreglo de la siguiente manera:

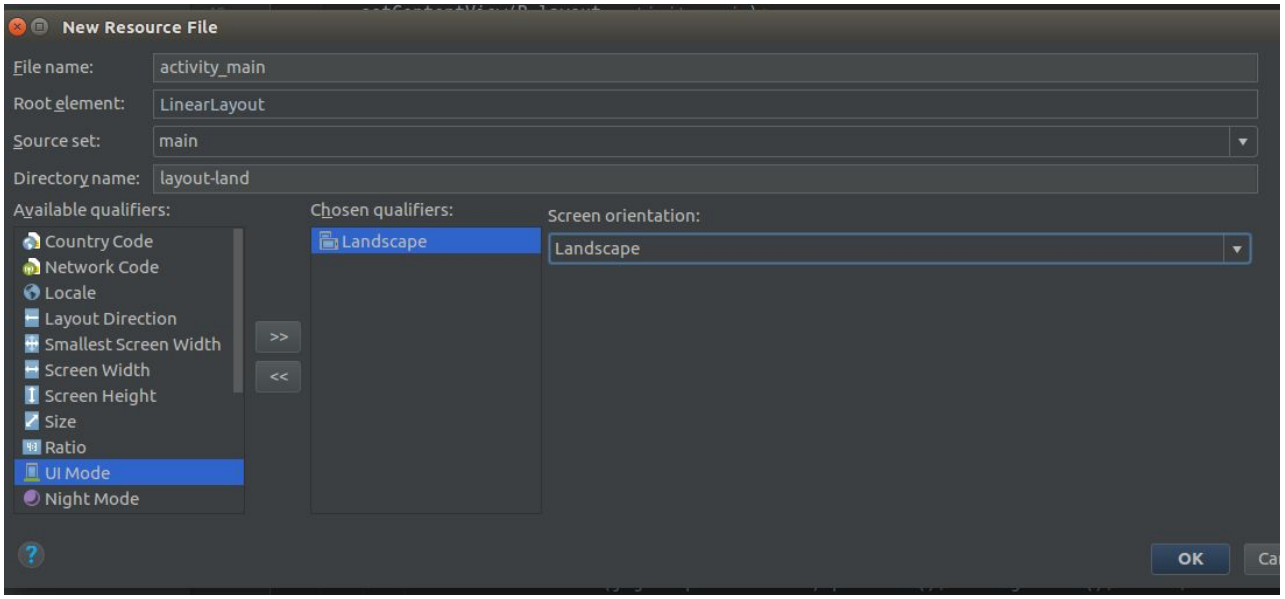
```
private int[] imagenes = {R.drawable.portada, R.drawable.interior, R.drawable.foto3};
```

4. Modifique el ejercicio anterior de manera que al girar la pantalla, se modifique el layout de la siguiente forma.



Implementar la grilla de botones usando GridLayout y un XML diferenciado según la orientación.

Nota: para incluir un layout dependiente de la orientación, desde Android Studio, se debe hacer click derecho en el directorio *layout* → *New* → *Layout resource file*.
Deberá tener elmismo nombre de archivo que el layout que usamos para la versión vertical.
Luego elegimos "*Orientation*" en el cuadro de "*Available qualifiers*", y luego la orientación *Landscape*:



Por defecto, cuando la aplicación esté en orientación *Portrait*, usará el layout que habíamos definido originalmente, y ahora la orientación *Landscape* usará este nuevo layout.

5. Implementar un formulario que permita el ingreso de un nombre y un DNI usando `EditText`, con un botón de guardar. Cuando el usuario hace click en guardar, los datos deberán almacenarse, de manera tal que al salir de la aplicación y volver a entrar, la información guardada deberá estar visible en los campos del formulario.

Nota: Utilizar *SharedPreferences* para almacenar la información.

6. Modificar la aplicación anterior para poder guardar múltiples personas.
Agregar un botón “Buscar”, que abra otro activity, donde se muestre un campo de búsqueda con un botón. Cuando el usuario ingrese un DNI, deberá buscarse entre los DNIs guardados, y mostrar el Nombre en pantalla.

Como SharedPreferences no permite guardar arreglos, utilizaremos la notación JSON.
Para esto, usaremos la clase `JSONArray`, que puede convertirse a String con facilidad para guardarlo.

En el siguiente ejemplo obtenemos un String que cumple el estándar JSON, y lo usamos como un JSONArray.

```
SharedPreferences preferences = getSharedPreferences("seminario",
MODE_PRIVATE);
//Notar que el segundo parámetro es el valor por defecto (array vacío),
// en caso de que "personas" aún no exista
String personasString = preferences.getString(“personas”, ”[]”);
try{
      JSONArray personas = new JSONArray(personasString);
      // acá trabajaríamos con el array de personas…
      // y lo volvemos a guardar en preferences
      preferences.edit().putString(“personas”, personas.toString() ).apply();
} catch (JSONException error){
      // Si se produce un error dentro del try,
      // se ejecuta este código que imprime el error por Logcat
      error.printStackTrace();
}
```