Práctica 2: Creación de tests con Espresso

Alejandro Pérez Ruiz
Grupo de Ingeniería Software y Tiempo Real
perezruiza@unican.es

Después de haber seguido los pasos de configuración de Espresso para Android debéis crear un sencillo test para la práctica 1.

Tareas a realizar:

1) Crear un test con la herramienta “Record Espresso Test”, donde se pruebe sobre la práctica 1 que al pulsar en el botón de limpiar cuando previamente hemos generado una quiniela aleatoria se muestra el texto esperado.

2) Vamos a crear un test para comprobar la correcta ejecución en la generación de una quiniela aleatoria, para ello debemos escribir un test en una nueva clase (ubicarla en androidTest). En esta ocasión no podemos utilizar la herramienta gráfica de Espresso, ya que tendremos que crear algo más elaborado. En nuestra prueba debemos pulsar el botón de “generar” y luego verificar que han aparecido 15 líneas en el TextView del resultado.
   Para que en un test se ejecute la acción de pulsar un botón Espresso nos ofrece los siguientes métodos:
   `onView(withId(R.id.boton)).perform(click())`
   Para contar el número de líneas tenemos que definir un método “matcher”, este sirver para hacer comprobaciones que por defecto no vienen implementadas en Espresso. Debe tener el siguiente aspecto:

```
public static Matcher<View> checkLineas (final int lineas){
      return new TypeSafeMatcher<View>() {
            @Override
            protected boolean matchesSafely(View item){
                  //TODO Aquí se describe nuestra lógica
                  //en este caso deberemos contar el número
                  //de lineas. El item será un textview, haciendo
                  //un cast y sacando el string fácilmente
                  //contaremos cada una de las lineas (‘\n’)
            }

            @Override
            public void describeTo(Description description){
                  description.appendText(“El número de líneas no
                                          coincide);
            }
      };
```

   Una vez completado el “matcher” podemos utilizarlo en nuestro test del siguiente modo:

```
onView(withId(R.id.textViewResultados)).check(ViewAssertions.mat
ches(checkLineas(15)));
```

Fijaos en el código del test generado automáticamente en el apartado 1 para ver cómo se instancia la ActivityTestRule disponible para todos los tests.

3) Parte opcional (1 punto extra): Crear un nuevo matcher que compruebe que en las primeras 14 líneas hay un 1, una X o un 2, y en las dos últimas un 0, 1, 2 o una M.

4) Subir al Moodle de la asignatura el código del test generado automáticamente en el punto 1 junto con un pantallazo (en jpeg) de Android Studio tras haber ejecutado el test. Hacer lo mismo para la tarea del punto 2, subir el código implementado y una captura de pantalla (en jpeg) del resultado de la ejecución. Comprimid los ficheros en un zip.