Proyecto final – TICMAS

Desarrollo de aplicaciones Móviles

Nahuel Gomez - 43459730

Consigna:

Para la realización de este proyecto, te invitamos a imaginarte que sos un desarrollador recién incorporado en un nuevo emprendimiento tecnológico. Te solicitan que desarrolles una aplicación que tenga un solo activity que cumpla con las siguientes premisas.

Una única pantalla (sin importar el layout elegido) con:

- 2 cuadros de textos (EditText)
- 1 botón con el texto “comparar”
- 1 texto (TextView) que en el que se escriba el resultado de la acción al presionar el botón.

Asegurate de que:

- Utiliza MVVM
- Tiene al menos un test unitario
- Tiene al menos un test de UI

Función de la app: Cuando el usuario hace click en el botón “comparar” debe comparar la entrada de ambos cuadros de texto y escribir en el texto (TextView) si ambas cadenas de caracteres son iguales o no.

El proyecto de github es un fork de https://github.com/eaceto/ticmas-lab-android

Arquitectura del proyecto:

Se utilizan 3 archivos principales para el modelo del problema:

- Se crea un objeto comparador para que guarde el valor de la última comparación realizada.
- Se utiliza un viewModel, el cual es el encargado de la lógica de negocio del proyecto. En este caso, el viewModel posee la capacidad de comparar las cadenas que le llegan y guardar si son iguales o no, en una variable interna.
- Se utiliza un MainActivity, el cual se comunica con la viewModel y con la interacción con la interfaz.

Testing:

El testing se realizó respecto a dos entidades principales:

- Tests integrados para el MainActivity:
  Utiliza el framework espresso, la cual sirve para facilitar la escritura y ejecución de pruebas que simulen interacciones del usuario con la interfaz de la aplicación Android.
  Se utiliza un sleep(1000) y un ViewActions.pressBack() para darle tiempo a la prueba de ejecutarlo correctamente y quitar el teclado luego del primer y segundo tipeo de texto.
  1) compareEqualStrings: Escribe en ambos campos “Texto1” y simula un click hacia el botón para comparar su resultado y se espera que el texto de respuesta sea igual a “las cadenas son iguales”.
  2) CompareNotEqualStrings: Escribe en el primer campo “Texto1” y en el segundo “Texto2”. Posteriormente simula el click en el botón “comparar” y verifica que el texto de respuesta sea igual a “Las cadenas son diferentes”.

- Tests unitarios para el viewModel:

  1) CheckInitialValue: Verifica que el primer texto, el cual aparece cuando aún no se pulsó el botón comparar sea igual a "Pulse 'comparar' para verificar si las cadenas son iguales"
  2) CheckEquality: Se llama al viewModel y le pide que compare 2 textos iguales.
  3) CheckNotEquality: Se llama al viewModel y se le pide que compare 2 textos diferentes.