Práctica 1 – Toma de contacto con Android

Introducción

El objetivo de esta práctica es familiarizarse con Android Studio, implementar una primera app sencilla, y aprender técnicas básicas de depuración con Android Studio.

Instrucciones

Siguiendo las presentaciones de clase, crea un nuevo proyecto Android usando la plantilla "Empty Views Activity", el nombre de proyecto "Calculator", `es.ucm.fdi.calculator` como nombre del paquete, Java como "Language".

- Ejecutar en el emulador la aplicación creada por la plantilla, siguiendo las instrucciones de las presentaciones de clase. Si es necesario crear un nuevo dispositivo virtual.
- Crear una nueva clase `es.ucm.fdi.calculator.Calculator` como código de producción (no de test). Para ello, con la vista Android activa en Android Studio, seleccionar el paquete `es.ucm.fdi.calculator` en el project explorer, hacer clic derecho y seleccionar **New > Java Class**. Añadir un método a la clase que implemente la suma de dos números.
- Crear un nuevo test unitario `es.ucm.fdi.calculator.CalculatorUnitTest` como código de test, seleccionando `es.ucm.fdi.calculator (test)` en el project explorer, y haciendo clic derecho y **New > Java Class**.
  - Añadir un test que pruebe que la suma de dos números (poner más de un caso con número enteros y con números reales): mirar un test de ejemplo en la clase existente `ExampleUnitTest` si es necesario.
  - Ejecutar el test unitario.
- Localizar el fichero layout de la actividad principal `MainActivity` en `res/layout`, y asegurarse de que aparece un `ConstraintLayout` en la raíz. Abrir el fichero layout con la vista *Design* del editor de layout de Android Studio.
  - Añadir 2 entradas de texto para introducir un número en cada una de las entradas. Han de admitir decimales.
  - Añadir un botón.
  - Hacer uso de *constraints* para que la actividad presente adecuadamente las entradas de texto y el botón.
    - Añadir un atributo *hint* a las entradas de texto para que aparezca el texto correspondiente.
  - Comprobar que la app se ejecuta correctamente en el emulador.
- Siguiendo las instrucciones, añadir puntos de interrupción (*breakpoints*) en métodos de *MainActivity*, y lanzar la app en el modo de depuración (*debug mode*).
- Para mostrar el resultado de la operación crear una nueva actividad de nombre `CalculatorResultActivity`, y con una única `TextView` en su fichero de layout.
- Añadir una variable miembro `Calculator calculator` a `MainActivity`, y variables miembro `EditText editTextX` y `EditText editTextY` a `MainActivity` e inicializarlas en `onCreate` usando `findViewById`. Añadir también el método `addXandY` que se invocaría al pulsar el botón y que mostraría el resultado en una actividad aparte llamada `CalculatorResultActivity`.

- Añadir el código que recupera y muestra el resultado de la operación a `CalculatorResultActivity`.
- Añadir logs en los métodos de `MainActivity` con diferentes niveles. Utilizar el nombre de la clase como etiqueta de los logs. Lanzar la app en el emulador y visualizar los filtros en el panel *LogCat*, filtrando con la etiqueta de los logs y por sus niveles.

A enseñar durante la corrección

1. Lanzar la app en el emulador:
   - Se valorará la creatividad en el diseño de la interfaz de usuario.
     - Opcional: mostrar el uso de *chains* de ConstraintLayout para agrupar las entradas de texto.
     - Opcional: en vez de entregar el diseño básico explicado (ver ejemplo de diseño de referencia en la Figura 1), se podrá implementar un layout alternativo haciendo uso de un ConstraintLayout para que la entrada de los números en vez de hacerla a través de entradas de texto se realice a través de botones para cada dígito y para el punto decimal (ver ejemplo de diseño de referencia en la Figura 2). El resultado se seguirá mostrando en `CalculatorResultActivity`.
     - Opcional: que se visualice correctamente tanto en vertical como en apaisado.
     - Opcional: que se visualice en varios idiomas.
   - Mostrar que es capaz de sumar dos números correctamente.
2. Con la app ejecutando en el emulador, abrir el panel LogCat y utilizar sus filtros para mostrar los mensajes de log enviados desde la app.
3. Lanzar la app ejecutando en el emulador en modo depuración y con algún breakpoint configurado. Usar el depurador para parar la ejecución en el breakpoint, y mostrar el valor de alguna variable de la actividad.
4. Ejecutar los test unitarios.

Diseños de referencia

Figura 1. Diseño básico

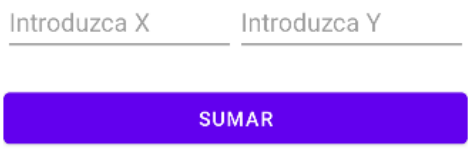

Figura 2. Diseño opcional