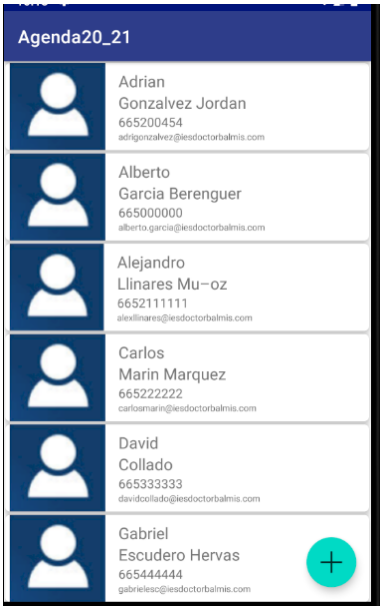
Ejercicio Propuesto Agenda

Descargar estos apuntes

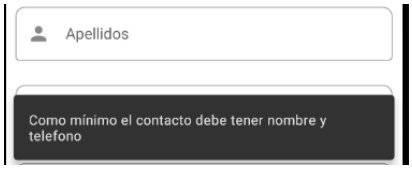
A partir del ejercicio del bloque 7 **Propuesto Agenda**, vamos a añadir más funcionalidad usando los nuevos conocimientos adquiridos en el bloque 8, Comenzaremos añadiendo un nuevo fragments que contendrá un recyclerView. En un principio la agenda no estará ordenada y los datos se guardarán en un ArrayList, las inserciones en la misma serán secuenciales (por el final). El aspecto será como el siguiente, y además podremos interactuar con los elementos del recycler (realizar llamada, eliminar, etc)



Es conveniente mantener la lista de datos en la actividad principal como elemento estático, para no perder la información al navegar entre los fragments.

Funcionalidad del Click en el Botón flotante

Al pulsar sobre el **FAB** se cargará el FragmentNuevoContacto, que ya teníamos construido en el bloque anterior. Ahora se quitará la funcionalidad de deslizamiento. Este fragment permitirá meter los datos del usuario, avisará con un `SnackBar` al pulsar sobre el **FAB** en caso de que no se haya etiquetado con al menos un elemento o si no se ha rellenado como mínimo nombre y tlf.



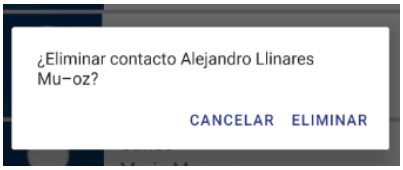
Si se rellenan los datos mínimos y pulsamos sobre FAB, se guardará el nuevo contacto en el array , se cerrará este Fragment volviendo al FragmentRecycler.

Funcionalidad del Click sobre un elemento del recycler

En caso de realizar una pulsación corta **OnClick**, se cargará el FragmentEditarContacto con los datos del contacto pulsado. Para ello podemos usar un `ViewMode`. También controlaremos si se deja algunos de los datos imprescindibles sin rellenar mediante Toast. Si es todo correcto al pulsar el **FAB** se modificará el contacto correspondiente.

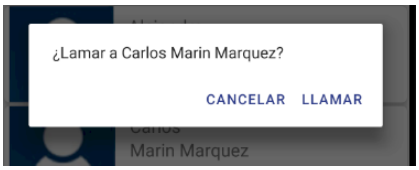
Funcionalidad del Long Click

En caso que el usuario realice una **pulsación larga** sobre algún elemento del recycler, se mostrará un dialogo preguntando si se quiere eliminar el contacto. Solo se eliminará si pulsamos a eliminar, en caso contrario se cerrará el dialogo. Eliminaremos el contacto del array y actualizaremos el adaptador.

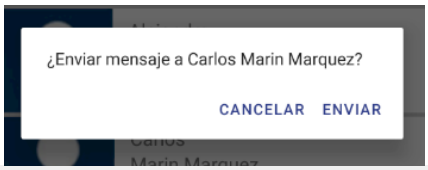


Funcionalidad del desplazamiento izq/derch sobre algún elemento del recycler

Si en la ventana inicial hago un **desplazamiento hacia la derecha** encima de un contacto, entonces lanzo el intent del dial con el teléfono del contacto, mostrando un dialogo que permita aceptar o rechazar la llamada. Si no tiene teléfono se avisará con un mensaje.



Si en la ventana principal hago un **desplazamiento hacia la izquierda**, se mandará un correo al contacto. Si no tiene dirección de correo se presentará un mensaje indicándolo.



> 💡 Para ayudar a la funcionalidad de los movimientos derecha e izquierda, se adjunta la clase **'swipeDetector'** que gestionará el tipo de movimiento. Esta clase implementa un listener para detectar los movimientos sobre la pantalla. En el fragment del recycler definiremos, una variable de esta clase:

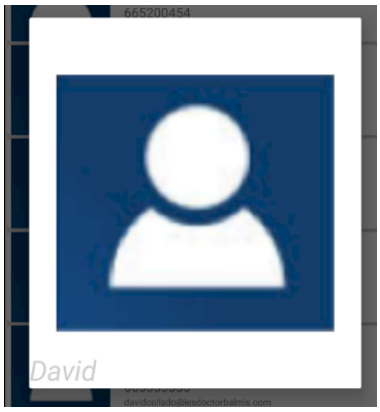
```
var swipeDetector: SwipeDetector= SwipeDetector()
```

Y a continuación escribiremos el `setOnItemClickListener()` de nuestro adaptador usando los métodos de la clase para detectar los movimientos. Importante, tendremos que pasar el objeto `swipeDetector` al listener correspondiente de la clase adaptador.

```
adaptador.setOnTouchListener(swipeDetector)
```

Funcionalidad del Click sobre la imagen del contacto

En caso de pulsar sobre la imagen se mostrará un dialogo personalizado en el que la imagen se mostrará de un tamaño mayor junto con el nombre del contacto, como podemos ver en la siguiente captura



Funcionalidad del Deslizamiento Izq/derch sobre el Fragmet de edición

En caso de realizar un deslizamiento sobre la pantalla edición cargaremos el siguiente elemento de la lista, de forma que podremos recorrer todos los contactos desde esta pantalla.
Lógicamente el deslizamiento a la derecha cargará el elemento siguiente al mostrado en ese momento , mientras que hacia la izquierda se cargará el anterior. Deberemos controlar que al llegar al primer o último elemento, ya no ocurra nada para evitar desbordamientos del array.
Para controlar el deslizamiento utilizaremos la clase que ya conocemos `SwipeDetector`, crearemos un objeto de este tipo que será lanzado con el método seOnTouchListener de la vista sobre la que queramos controlar el movimiento. Después tendremos que controlar el movimiento en el método **OnClick**.