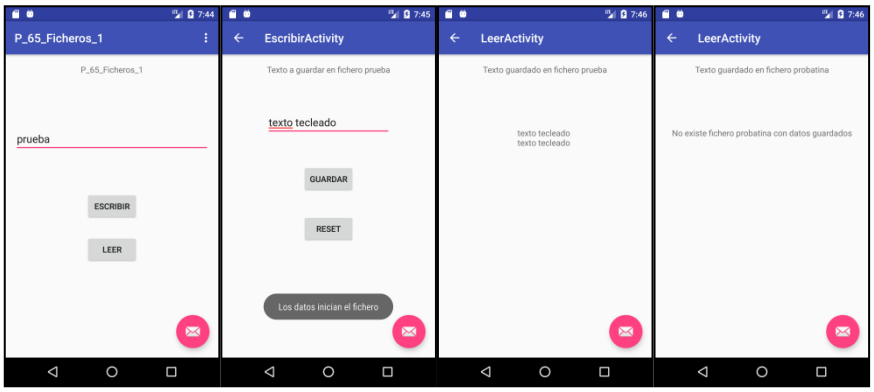
Ejercicio 1: Ficheros en memoria interna

1. El proyecto **P_98_Ficheros_1** (para trabajar con la memoria interna) debe guardar los datos en un fichero en la memoria interna de la siguiente forma:
    a. El layout inicial solo contiene la solicitud de de nombre de fichero y los botones de guardar y leer y sus acciones deben realizarse en actividades distintas.
    b. En la actividad de escribir datos, el texto tecleado para guardar debe añadirse al contenido del fichero en una nueva línea, advirtiendo por "Toast" si el fichero se usa por primera vez o no
    c. En la actividad de leer datos, debe visualizarse todo el contenido del fichero si ya existe o mostrar un mensaje de advertencia en caso contrario.

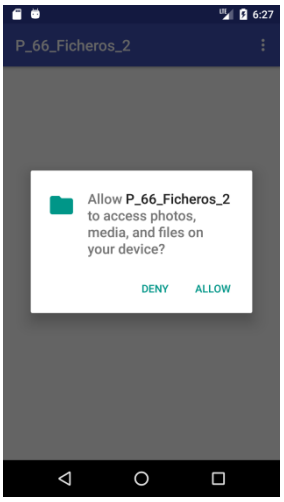
    Pista:

    i. String[] archivos = fileList();
    ii. Dado que tanto en Escribir como en Leer debe comprobarse la existencia del fichero, sería conveniente que para optimizar código te crees una nueva clase (de nombre por ejemplo ComprobarExistencia) que desarrolle un método para comprobar esa existencia de fichero buscado y que sea usada las veces que se necesite.

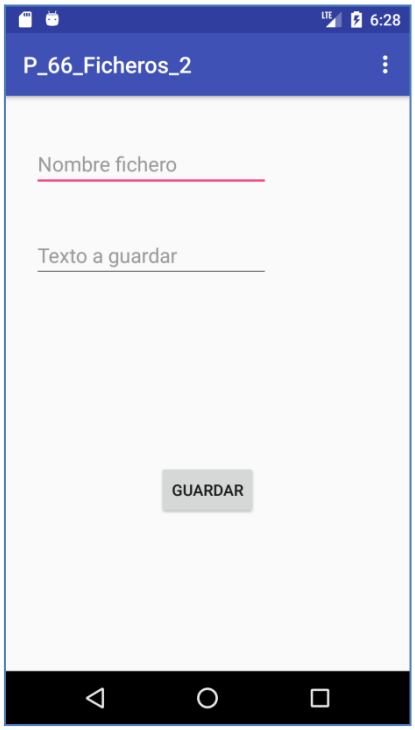


Ejercicios: Ficheros en memoria externa

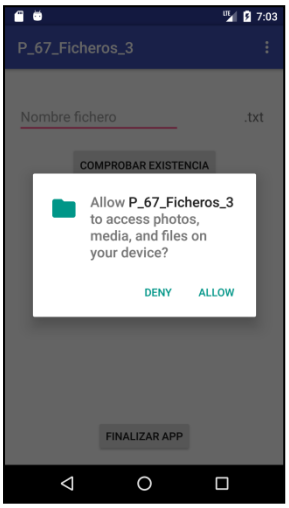
2. El proyecto **P_99_Ficheros_2** (para trabajar con la SD) permite escribir texto en ficheros de la siguiente manera:
    a. Primero comprobar que está asignado el permiso y solicitarlo en caso negativo

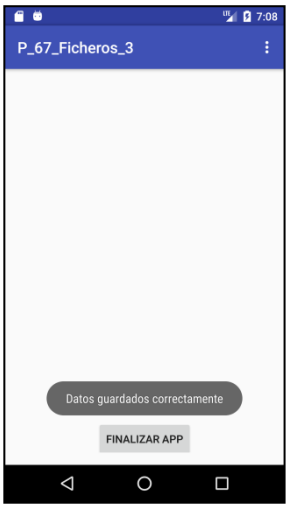


    b. Después debe comprobar la existencia de la tarjeta y su disponibilidad para escribir, avisando mediante Toast si no es posible trabajar con la citada SD.
    c. Y por último, tal y como muestra la imagen, debe solicitar el nombre del fichero y el texto a introducir que se añadirá al final si el fichero ya existía (pista: new FileOutputStream(f, true)) siempre y cuando ninguno de los valores a introducir se hayan dejado vacíos



3. El proyecto **P_100_Ficheros_3** (para trabajar con SD, es decir, hay que comprobar permiso otorgado y disponibilidad de tarjeta) tal y como muestran las imágenes permite escribir texto en ficheros desde una sola pantalla manipulando adecuadamente el atributo de visibilidad y de "focusibilidad" de los controles del layout

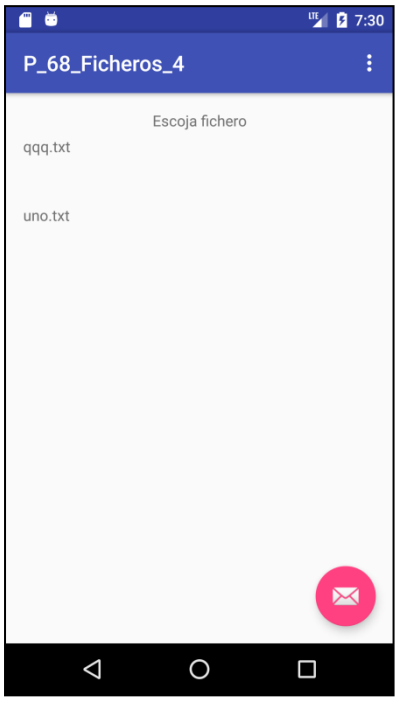




Pista para comprobar si el fichero es nuevo o ya existe y poder mostrar mensaje correcto:

```
private boolean comprobarExistencia(String nombre_fichero) {
   File ruta_sd = Environment.getExternalStoragePublicDirectory(Environment.DIRECTORY_DOWNLOADS);
   String[] listaFicheros = ruta_sd.list();
   for (String archivoLista : listaFicheros) {
      if (nombre_fichero.equals(archivoLista)) {
         return true;
      }
   }
   return false;
}
```

4. El proyecto **P_101_Ficheros_4** (para trabajar con SD) debe mostrar un recyclerView de los ficheros con extensión .txt. Al pulsar sobre uno de ellos debe visualizarse su contenido en otra actividad.



Pistas:

i) Recuerda control de permiso
ii) Recuerda RecyclerView, el método que lee los datos usa la clase Filtro que implementa la interface FilenameFilter (más información):

```
private ArrayList<Item> leerDatos() {
   ArrayList<Item> datos = new ArrayList<>();
   File ruta_sd = Environment.getExternalStoragePublicDirectory(Environment.DIRECTORY_DOWNLOADS);
   String[] listaFicheros = ruta_sd.list(new Filtro(".txt"));
   for (String archivoLista : listaFicheros) {
      datos.add(new Item(archivoLista));
   }
   if (listaFicheros.length==0)
      Toast.makeText(this,"No hay ficheros",Toast.LENGTH_SHORT).show();
   return datos;
}

private class Filtro implements FilenameFilter {
   String extension;
   Filtro(String extension){
      this.extension=extension;
   }
   public boolean accept(File dir, String name){
      return name.endsWith(extension);
   }
}
```