Documentación

Proyecto agenda

Iván Gallego

Índice de contenidos

Leer datos al inicio

cargarDatos

El método `cargarDatos` hace una consulta a la base de datos y utiliza el método `leerContactos` para extraer los contactos existentes de la base de datos

```
private void cargarDatos() {
    contactos = new ArrayList<>();
    if (dbContactos != null) {
        contactosDatabase = dbContactos.getReadableDatabase();
        Cursor cursor = contactosDatabase.rawQuery(
            "SELECT * FROM contactos", null);
        leerContactos(cursor);
        contactosDatabase.close();
    }
}
```

leerContactos

El método `leerContactos` recorre con el cursor todos los resultados de la consulta hecha por el método `cargarDatos` y construye un objeto `Contactos` a partir de la información obtenida

```
private void leerContactos(Cursor cursor) {
    cursor.moveToFirst();
    Contacto c;
    while (!cursor.isAfterLast()) {
        c = new Contacto();
        c.setId(cursor.getInt(0));
        c.setNombre(cursor.getString(1));
        c.setApellido(cursor.getString(2));
        c.setTelefono(cursor.getString(3));
        c.setCorreo(cursor.getString(4));
```

```
            Bitmap imagen = convertirBytesBitmap(cursor.getBlob(5));
            c.setImagen(imagen);
            c.setAmigo(cursor.getInt(6) == 1);
            c.setFamilia(cursor.getInt(7) == 1);
            c.setTrabajo(cursor.getInt(8) == 1);
            contactos.add(c);
            cursor.moveToNext();
        }
    }
```

Añadir datos

guardarContacto

El método `guardarContacto` lo introduce en la base de datos utilizando `ContentValues`. No se indica ID porque es `AUTOINCREMENT`

```
private void guardarContacto(Contacto c) {
    contactosDatabase = dbContactos.getWritableDatabase();
    SQLiteDatabase readableDatabase = dbContactos.getReadableDatabase();
    ContentValues values = new ContentValues();
    values.put("nombre", c.getNombre());
    values.put("apellido", c.getApellido());
    values.put("telefono", c.getTelefono());
    values.put("correo", c.getCorreo());
    Bitmap b = c.getImagen();
    byte[] bytesImagen = null;
    if (b != null) {
        bytesImagen = convertirImagenBytes(c.getImagen());
    }
    values.put("imagen", bytesImagen);
    values.put("amigo", c.isAmigo());
    values.put("trabajo", c.isTrabajo());
    values.put("familia", c.isFamilia());
    contactosDatabase.insert("contactos", null, values);
    contactosDatabase.close();
    readableDatabase.close();
}
```

Eliminar datos

eliminarContacto

El método `eliminarContacto` elimina un contacto de la base de datos utilizando su id como referencia

```
private void eliminarContacto(Contacto contacto) {
    if (dbContactos != null) {
        contactosDatabase = dbContactos.getWritableDatabase();
```

```
            contactosDatabase.delete(
                "contactos", "id = ?",
                new String[] {String.valueOf(contacto.getId())});
            contactosDatabase.close();
        }
    }
```

Modificar datos

editarContacto

El método `editarContacto` comprueba que exista un contacto con el id anterior, a continuación, edita el contacto anterior por el nuevo en la base de datos

```
private void editarContacto(Contacto editado, Contacto nuevo) {
    contactosDatabase = dbContactos.getWritableDatabase();
    SQLiteDatabase readableDatabase = dbContactos.getReadableDatabase();
    long count = DatabaseUtils.queryNumEntries(
        readableDatabase, "contactos", "id = ?",
        new String[]{String.valueOf(editado.getId())});
    if (count == 1){
        ContentValues values = new ContentValues();
        values.put("nombre", nuevo.getNombre());
        values.put("apellido", nuevo.getApellido());
        values.put("telefono", nuevo.getTelefono());
        values.put("correo", nuevo.getCorreo());
        Bitmap b = nuevo.getImagen();
        byte[] bytesImagen = null;
        if (b != null) {
            bytesImagen = convertirImagenBytes(nuevo.getImagen());
        }
        values.put("imagen", bytesImagen);
        values.put("amigo", nuevo.isAmigo());
        values.put("trabajo", nuevo.isTrabajo());
        values.put("familia", nuevo.isFamilia());
        contactosDatabase.update(
            "contactos", values, "id = ?",
            new String[]{String.valueOf(editado.getId())});
    }
    contactosDatabase.close();
    readableDatabase.close();
}
```

Para poder realizar las operaciones anteriores, debe haber las siguientes variables de clase

```
private BDContactos dbContactos;
private SQLiteDatabase contactosDatabase;
```

```
public ArrayList<Contacto> contactos;
private Contacto contactoTemp;
```

Siendo:

1. `dbContactos` - La base de datos
2. `contactosDatabase` - La base de datos sobre la que se puede escribir o leer dependiendo del case
3. `contactos` - La lista de contactos
4. `contactoTemp` - El contacto que se utiliza de manera auxiliar para saber cuál es el contacto editado, para posteriormente editarlo en la base de datos