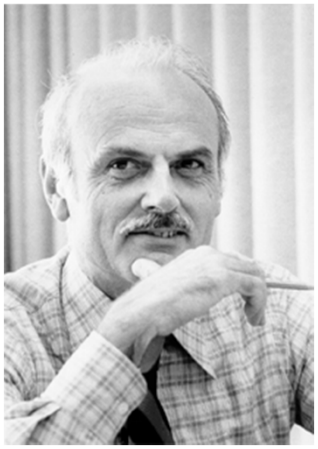
CS378 - Mobile Computing

Persistence - SQLite

Databases

- RDBMS
  - relational data base management system
- Relational databases introduced by E. F. Codd
  - Turing Award Winner



- Relational Database
  - data stored in tables
  - relationships among data stored in tables
  - data can be accessed and view in different ways

SQL and SQLite

- Structured Query Language
- programming language to manage data in a RDBMS
- SQLite implements most,
  but not all of SQL
- SQLite becomes part of application

Database Developer, 2 HR:581

Category :Database Developer Sr
Location :Taylor, TX
Work Status :Full Time

Description

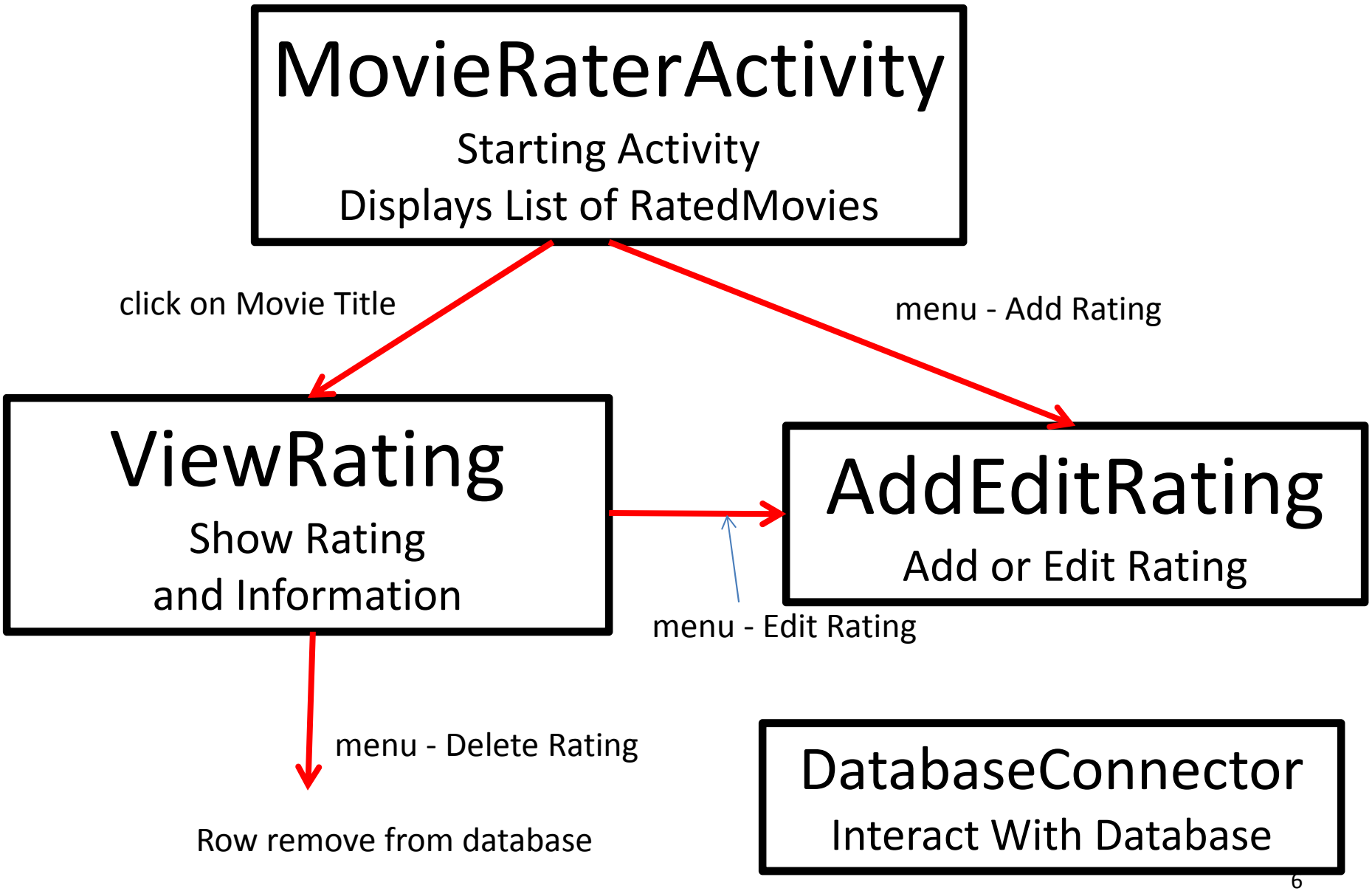
SQLite and Android

- Databases created with or for application accessible by name to all classes in application, but none outside application
- Creating database:
  - create subclass of SQLiteOpenHelper and override onCreate() method
  - execute SQLite command to create tables in database

Creating Database

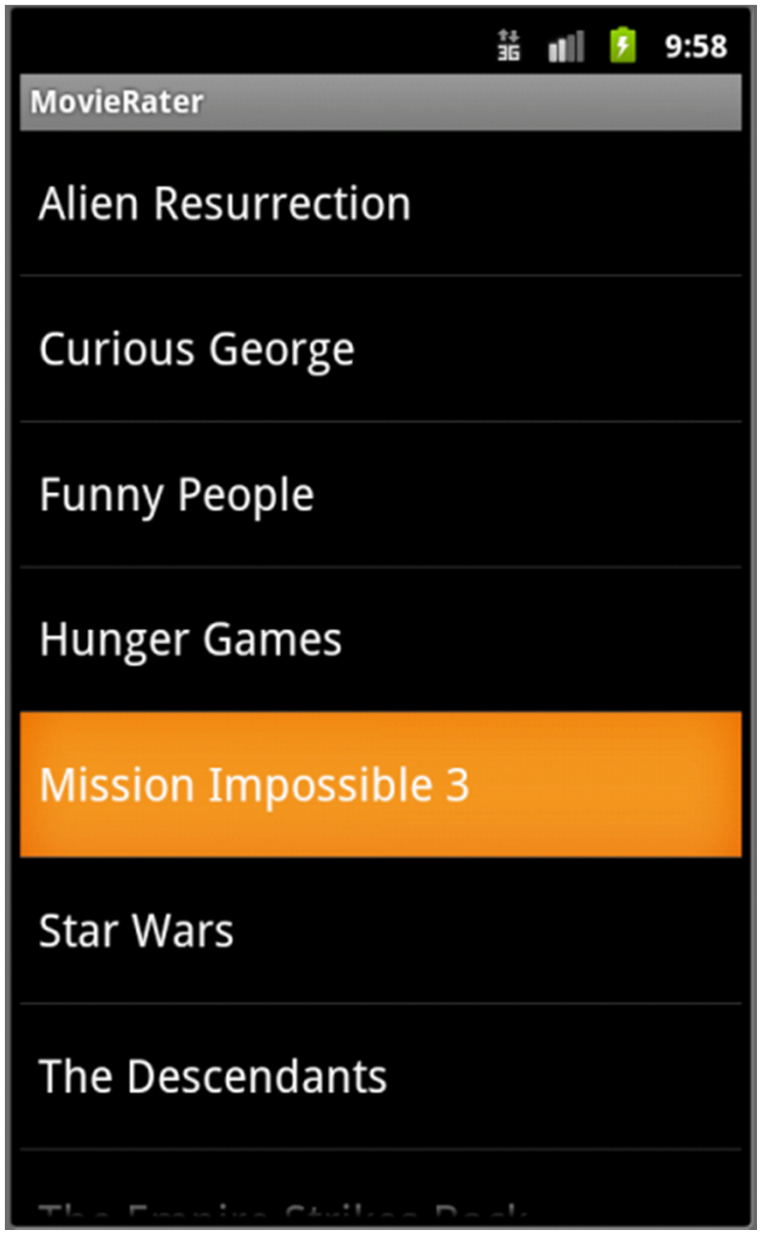
- Example: Movie Rating App
- Stores user ratings
- Not a complex example
- Database only has one table
- Adapted from Deitel Address Book Application
- http://www.deitel.com/Books/Android/AndroidforProgrammers/tabid/3606/Default.aspx

Classes


MovieRaterActivity
Starting Activity
Displays List of RatedMovies
click on Movie Title
menu - Add Rating
ViewRating
Show Rating
and Information
AddEditRating
Add or Edit Rating
menu - Edit Rating
menu - Delete Rating
Row remove from database
DatabaseConnector
Interact With Database

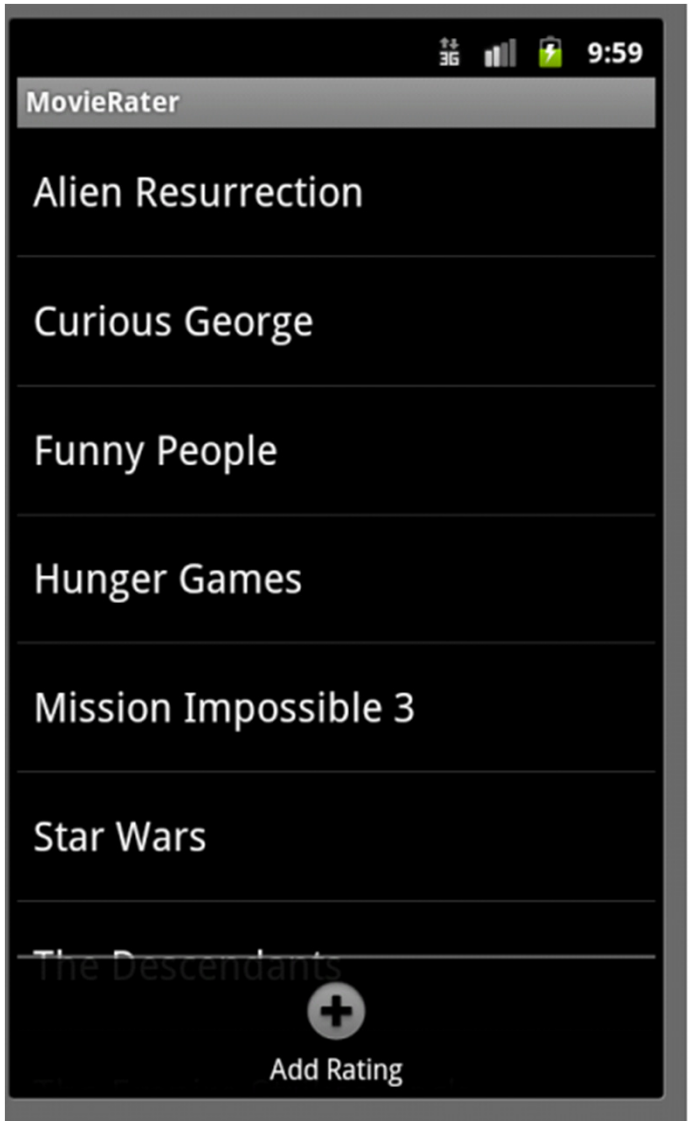
MovieRaterActivity

- ScrollView
- Queries data base for all names / titles
- Clicking on Title brings up that rating in ViewRating



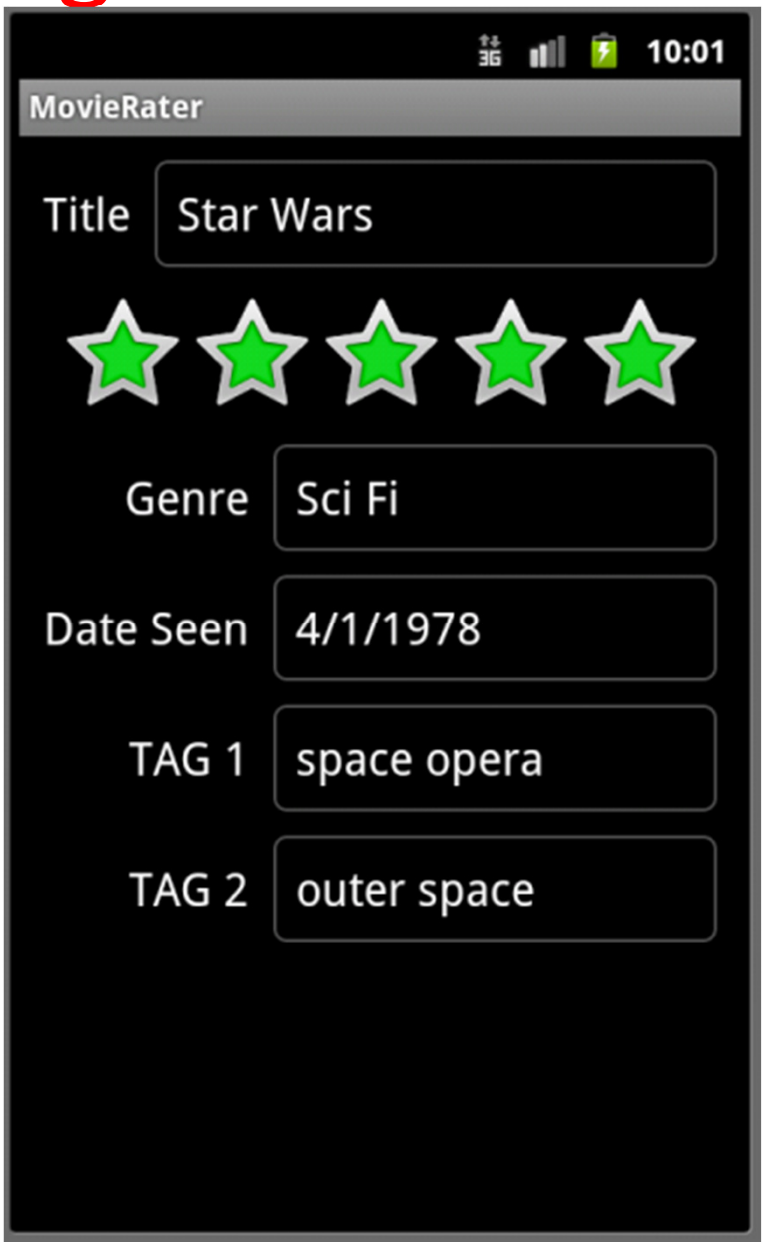
Menu for MovieRaterActivity

- Only one menu option
- button to Add Rating
- Brings up AddEditRating Activity



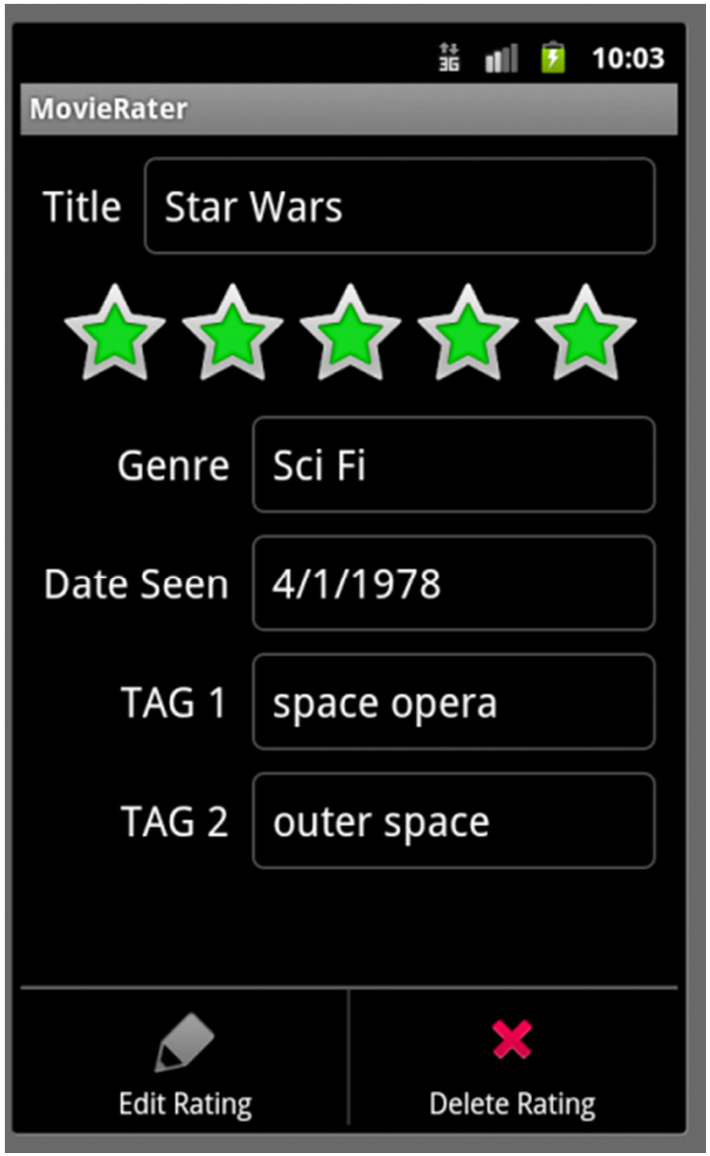
ViewRating

- Pulls all data from database for row based on name / title
- Use of a RatingBar
- ViewRating has its own Menu



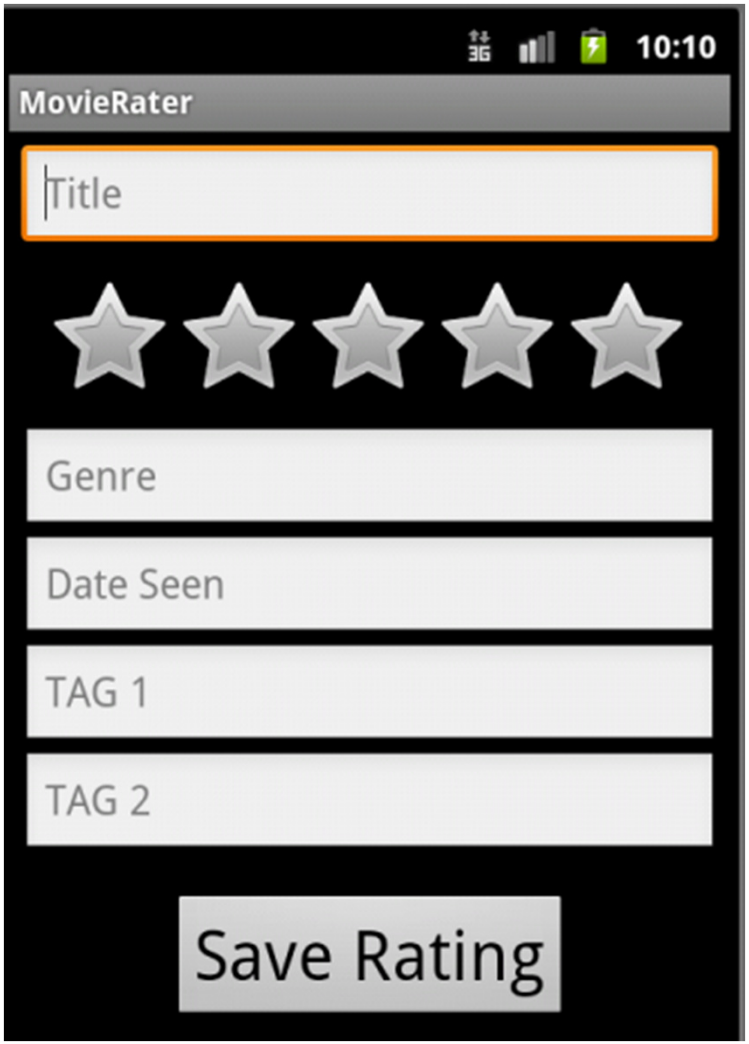
ViewRating Menu

- Edit Rating starts AddEditRating activity and populates fields with these values (place in Extras)
- Delete Rating brings up confirmation Dialog



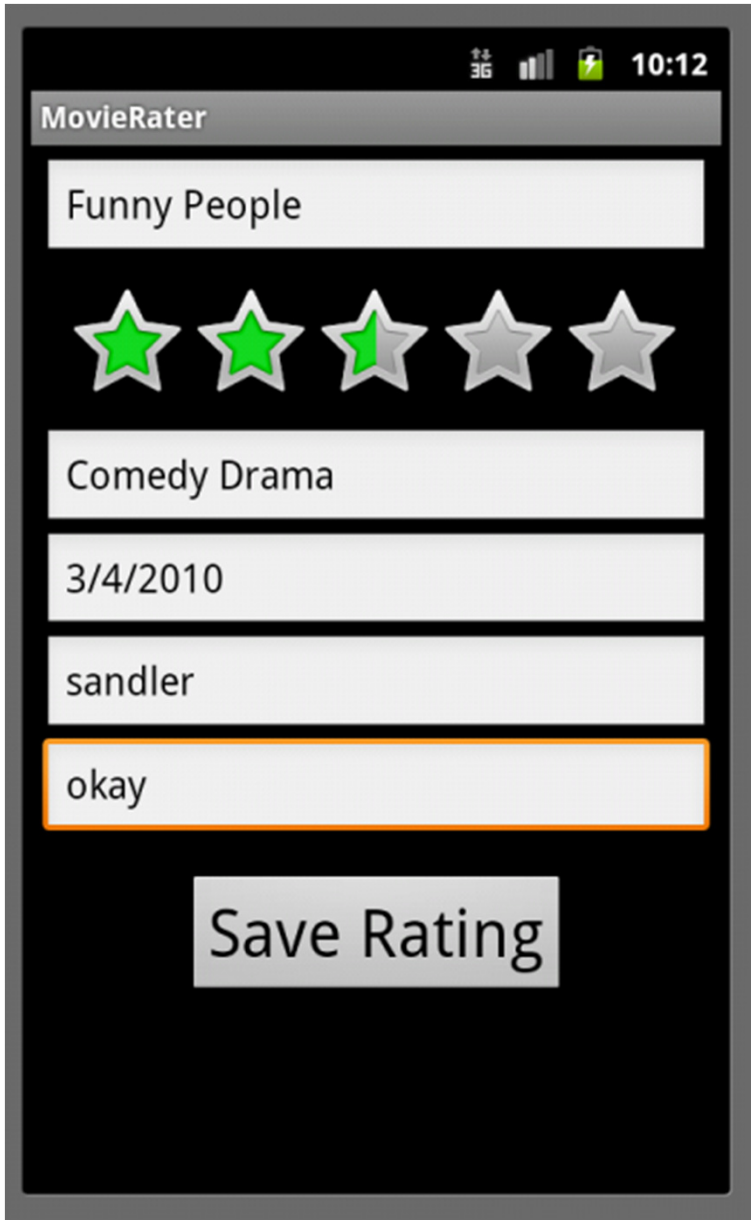
AddEditRating

- Add Rating
  - fields are blank
- Consider adding a button for date picker instead of typing data
- Must enter title / name
- other fields can be blank



AddEditRating

- When title clicked in main Activity, MovieRaterActivity
- Make changes and click save



DatabaseConnector Class

- Start of class

```
public class DatabaseConnector {

    private static final String DATABASE_NAME = "MovieRatings";
    private SQLiteDatabase database;
    private DatabaseOpenHelper databaseOpenHelper;


    public DatabaseConnector(Context context) {
        databaseOpenHelper =
            new DatabaseOpenHelper(context, DATABASE_NAME, null, 1);
    }
```

DatabaseConnector Class

```
public void open() throws SQLException {
    // create or open a database for reading/writing
    database = databaseOpenHelper.getWritableDatabase();
}


public void close() {
    if (database != null)
        database.close();
}
```

Creating Database

- Via an inner class that extends SQLiteOpenHelper

```
private class DatabaseOpenHelper extends SQLiteOpenHelper {

    public DatabaseOpenHelper(Context context, String name,
        CursorFactory factory, int version) {
        super(context, name, factory, version);
    }
```

Creating Database

- Money method

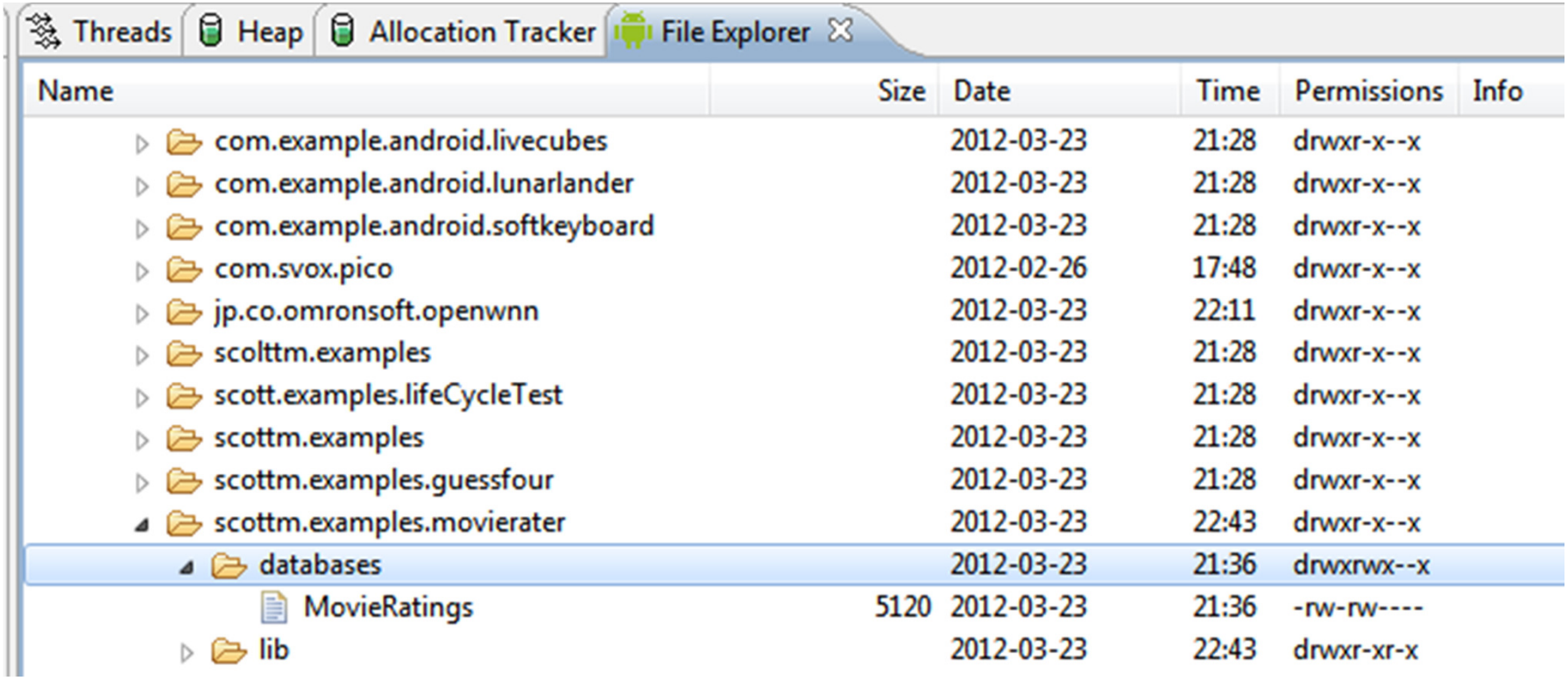
```
// creates the ratings table when the database is create
@Override
public void onCreate(SQLiteDatabase db) {
    // query to create a new table named ratings
    String createQuery = "CREATE TABLE ratings" +
        "(_id INTEGER PRIMARY KEY autoincrement, " +
        "name TEXT, " +
        "genre TEXT, " +
        "dateSeen TEXT, " +
        "tag1 TEXT, " +
        "tag2 TEXT, " +
        "rating INTEGER);";

    db.execSQL(createQuery);
}
```

Creating Database

- String is a SQLite command
- ratings is name of table
- table has seven columns
  - _id, name, genre, dateSeen, tag1, tag2, rating
- storage classes for columns:
  - TEXT, INTEGER, REAL
  - also NULL and BLOB
- _id is used as primary key for rows

Database on Device



Name	Size	Date	Time	Permissions	Info
com.example.android.livecubes		2012-03-23	21:28	drwxr-x--x	
com.example.android.lunarlander		2012-03-23	21:28	drwxr-x--x	
com.example.android.softkeyboard		2012-03-23	21:28	drwxr-x--x	
com.svox.pico		2012-02-26	17:48	drwxr-x--x	
jp.co.omronsoft.openwnn		2012-03-23	22:11	drwxr-x--x	
scolttm.examples		2012-03-23	21:28	drwxr-x--x	
scott.examples.lifeCycleTest		2012-03-23	21:28	drwxr-x--x	
scottm.examples		2012-03-23	21:28	drwxr-x--x	
scottm.examples.guessfour		2012-03-23	21:28	drwxr-x--x	
scottm.examples.movierater		2012-03-23	22:43	drwxr-x--x	
databases		2012-03-23	21:36	drwxrwx--x	
MovieRatings	5120	2012-03-23	21:36	-rw-rw----	
lib		2012-03-23	22:43	drwxr-xr-x	

- can pull database and view
- sqlitebrowser is a good tool

Inserting Data

- ContentValues are key/value pairs that are used when inserting/updating databases
- Each ContentValue object corresponds to one row in a table
- _id being added and incremeneted automatically

Inserting Data

- In AddEditRating
- When save button clicked

```
private void saveRating() {
    // get DatabaseConnector to interact with the SQLite da
    DatabaseConnector databaseConnector = new DatabaseConne

    if (getIntent().getExtras() == null) {
        // insert the rating information into the database
        databaseConnector.insertRating(
                title.getText().toString(),
                (int) rating.getRating(),
                genre.getText().toString(),
                dateSeen.getText().toString(),
                tag1.getText().toString(),
                tag2.getText().toString());
    }
    else {
```

Inserting Data

- In DatabaseConnector

```
// inserts a new rating into the database
public void insertRating(String title, int rating,
   String genre, String dateSeen, String tag1, String tag2) {
   ContentValues newRating = new ContentValues();
   newRating.put("name", title);
   newRating.put("rating", rating);
   newRating.put("genre", genre);
   newRating.put("dateSeen", dateSeen);
   newRating.put("tag1", tag1);
   newRating.put("tag2", tag2);

   open();
   database.insert("ratings", null, newRating);
   close();
}
```

nullCoumnHack, for inserting empty row

Updating Data

- In AddEditRating
- When save button clicked
- notice id added

```
else {
    databaseConnector.updateRating(rowID,
            title.getText().toString(),
            (int) rating.getRating(),
            genre.getText().toString(),
            dateSeen.getText().toString(),
            tag1.getText().toString(),
            tag2.getText().toString());
}
```

Updating Data

- In DatabaseConnector

```
// updates a rating in the database
public void updateRating(long id, String name, int rating,
    String genre, String dateSeen, String tag1, String tag2) {

    ContentValues editRating = new ContentValues();
    editRating.put("name", name);
    editRating.put("rating", rating);
    editRating.put("genre", genre);
    editRating.put("dateSeen", dateSeen);
    editRating.put("tag1", tag1);
    editRating.put("tag2", tag2);

    open();
    database.update("ratings", editRating, "_id=" + id, null);
    close();
}
```

Query Data

- Getting a single row by _id
  - in order to populate ViewRating

```
// get a Cursor containing all information about the movie specifi
// by the given id
public Cursor getOneRating(long id) {
    return database.query(
        "ratings", null, "_id=" + id, null, null, null, null);

    // public Cursor query (String table, String[] columns,
    // String selection, String[] selectionArgs, String groupBy,
    // String having, String orderBy
}
```

Query Data

- Get all rows
- To populate the ListView in the MovieRaterActivity
- only getting _id and name columns

```
public Cursor getAllRatings() {
    return database.query("ratings", new String[] {"_id", "name"},
        null, null, null, null, "name");
    // query(String table,
    // String[] columns, String selection, String[] selectionArgs,
    // String groupBy, String having, String orderBy)
}
```

Deleting Data

- Menu Option in ViewRating

```
// delete the rating specified by the given id
public void deleteRating(long id) {
    open();
    database.delete("ratings", "_id=" + id, null);
    close();
}
```

Database Cursor

- Cursor objects allow random read - write access to the result of a database query
- Ours only used to read the data
- Use a CursorAdapter to map columns from cursor to TextView or ImageViews defined in XML files

Database Connection

- Recall:

```
public Cursor getAllRatings() {
    return database.query("ratings", new String[] {"_id", "name"},
        null, null, null, null, "name");
    // query(String table,
    // String[] columns, String selection, String[] selectionArgs,
    // String groupBy, String having, String orderBy)
}
```

MovieRaterActivity

- Rating Adapter is a CursorAdapter
- from onCreate method

```
// map each ratings's name to a TextView
// in the ListView layout
String[] from = new String[] { "name" };
int[] to = new int[] { R.id.ratingTextView };
ratingAdapter = new SimpleCursorAdapter(
        MovieRaterActivity.this,
        R.layout.rating_list_item, null,
        from, to);
// public SimpleCursorAdapter (Context context,
// int layout, Cursor c,
// String[] from, int[] to)

setListAdapter(ratingAdapter);
```

Updating Cursor

- Cursor initially null
- separate task to create cursor and update adapter

```
@Override
protected void onResume() {
    super.onResume();

    // create new GetRatingsTask and execute it
    new GetRatingsTask().execute((Object[]) null);
}
```

Asynch Task

```
// performs database query outside GUI thread
private class GetRatingsTask extends AsyncTask<Object, Object, 
    DatabaseConnector databaseConnector =
            new DatabaseConnector(MovieRaterActivity.this);

    // perform the database access
    @Override
    protected Cursor doInBackground(Object... params) {
        databaseConnector.open();

        return databaseConnector.getAllRatings();
    }

    // use the Cursor returned from the doInBackground method
    @Override
    protected void onPostExecute(Cursor result) {
        ratingAdapter.changeCursor(result);
        databaseConnector.close();
    }
} // end class GetContactsTask
```

Clicking on Item in List

- _id not displayed but still part of entry in list -> use _id to get back to database row

```
// event listener that responds to the user touching a contact's name
// in the ListView
OnItemClickListener viewRatingListener = new OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int position,
            long id) {

        Log.d("MoiveRater", "postion: " + position + ", id: " + id);
        // create an Intent to launch the ViewRating Activity
        Intent viewContact =
                new Intent(MovieRaterActivity.this, ViewRating.class);

        // pass the selected contact's row ID as an extra with the Intent
        viewContact.putExtra(ROW_ID, id);
        startActivity(viewContact);
    }
};
```

Other Cursor Options

- moveToPrevious
- getCount
- getColumnIndexOrThrow
- getColumnName
- getColumnNames
- moveToPosition
- getPosition

Possible Upgrades

- Add functionality to
  - show all movies that share a particular genre
  - movies from a date range
  - shared tags
- Just more complex data base queries