420-616-AB Oracle Technologies
Fall 2016

September 19th, 2015
Worth 5%

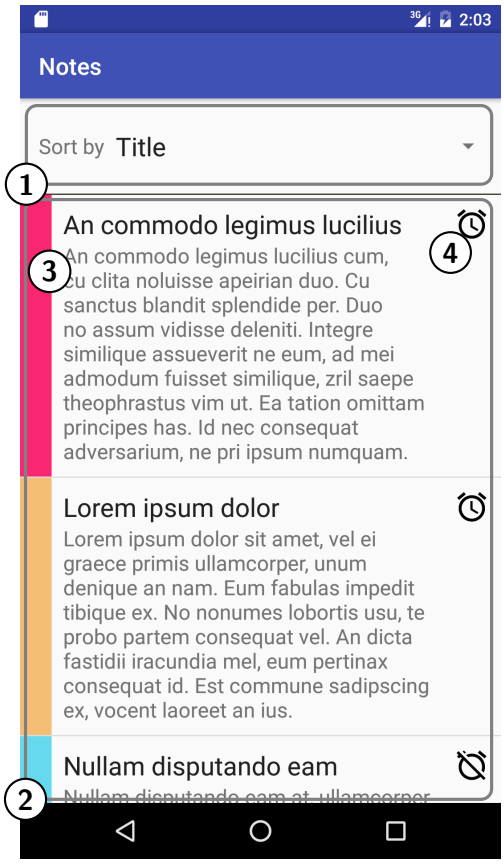
Assignment II

(Due date is on Léa)

Design and implement an Android `Fragment` for a user to browse a list of notes. These notes are stored in an `sqlite` database stored locally on the Android device. Note that these are the same notes as last Assignment, but it currently optional to integrate the two assignments (this is the next assignment!).

The user will interact with a single fragment inside a single activity. It will include UI components for:

1. A spinner (drop-down) for choosing the list sort criteria.
2. A list of notes, containing the title and body of the note.
3. The category of the note as a coloured rectangle on the left-hand side of the list.
4. An icon indicating whether the reminder is set or not. The icon can be clicked to turn off a reminder.



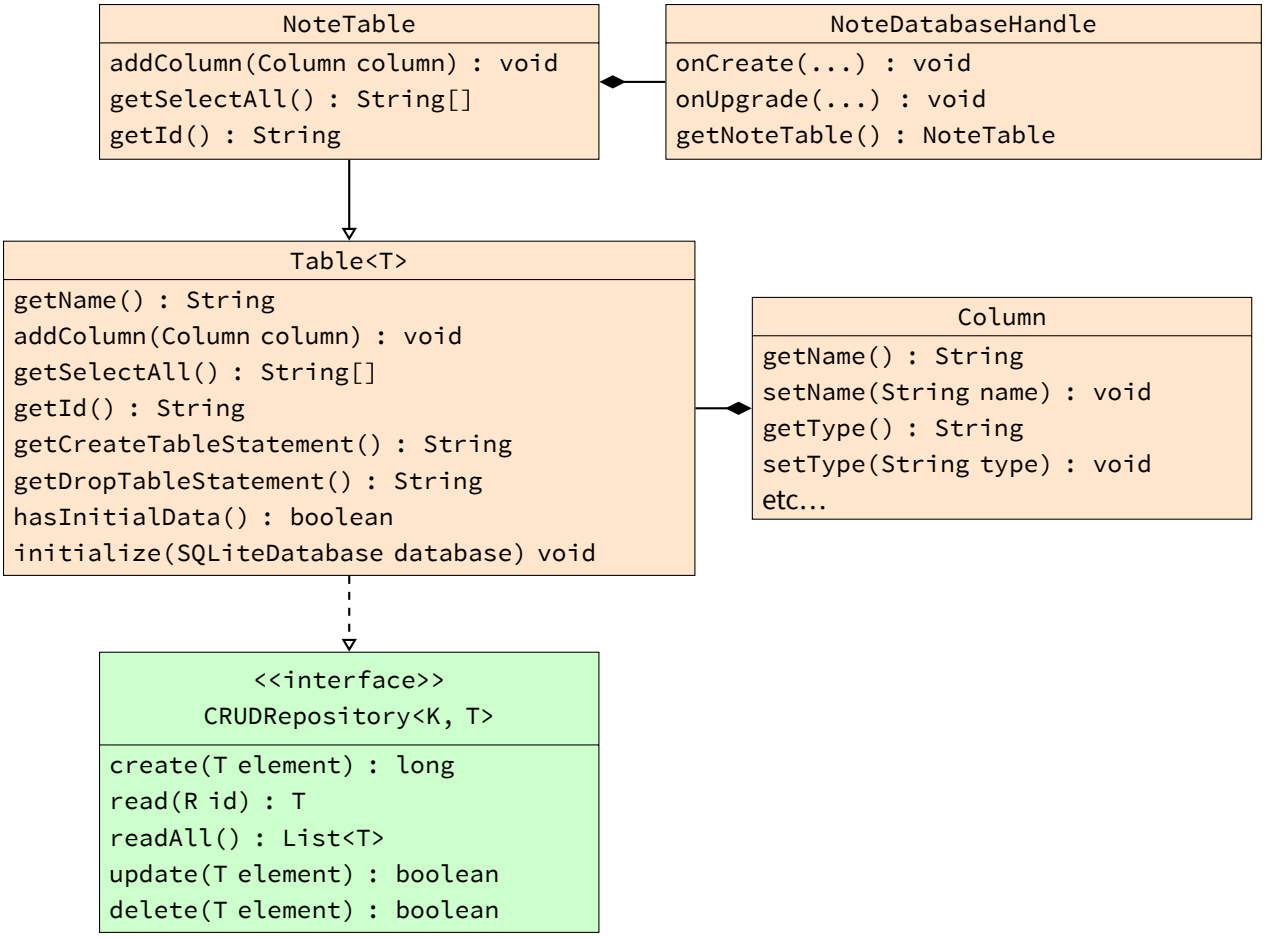
1 Database

Store the notes in an `sqlite` database called `notes.db`. The database consists of a single table called `note`:

Column	Type	Attributes
`_id`	integer	primary key, autoincrement
title	Text	not null, unique
body	Text	
reminder	Text*	
category	Integer	not null
created	Text*	not null

* recall that `sqlite` has no date datatype. Use ISO 8601 standard dates like my example in class.

sqlite. Recall from class that we will use an OOP design to work with sqlite on Android. The following class diagram shows the interactions among the classes involved.



Each class plays a role in the building of a database:

- `CRUDRepository<K,T>` is an interface describing the structure of the CRUD operations.
- `Table<T>` is the super-class of all tables. It stores the structure of the table as a set of `Column`s. SQL statements for creating and deleting the table.. Finally, it implements the CRUD operations in a straightforward way, for which subclasses can override as needed.
- `Column` represents a column in the database with field for column name and datatype, as well as attributes such as `PRIMARY KEY`, `UNIQUE` and `NOT NULL`.
- `NoteTable` a sub-class of `Table<T>` specifically for notes.
- `NoteDatabaseHelper` represents the database. It extends `SQLiteOpenHelper` as was done in class.

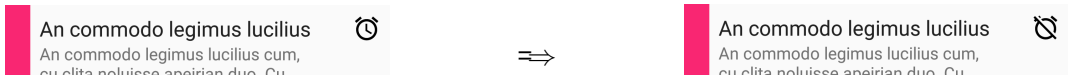
Test Data. The starter files include sample notes data, just as we had in class. My aim was to simplify your development. Use it by overriding the methods `hasInitialData()` and `initialize(SQLiteDatabase database)`

Database version. Recall that the database is recreated for each version increase in the class `NoteDatabase`. So the version of the database on your device is 12, then you can force an update if you change version to 13 (technically, you can increase to any number, but you should probably increment by 1). Use this feature to recreate and repopulate your database with test data.

2 List

In the list section of the fragment, display each note **from the database**. Use a custom adapter with a custom layout to show the title, body, category color and reminder as show on page .

Reminder. The reminder icon is set to ⏰ if the note has a reminder or 🔕 otherwise. These icons correspond to `R.drawable.ic_alarm_black_24dp)` and `R.drawable.ic_alarm_off_black_24dp` respectively.



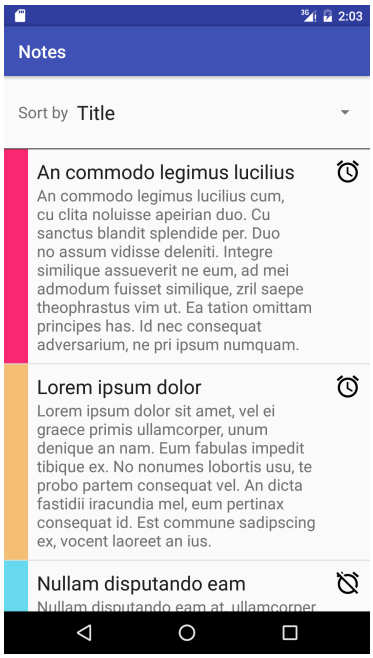
Reminder. *No reminder.*

The reminder can be turned off by clicking the reminder icon. Turning on a reminder is not necessary. *Turning off a reminder updates the note in the database.*

3 Sorting

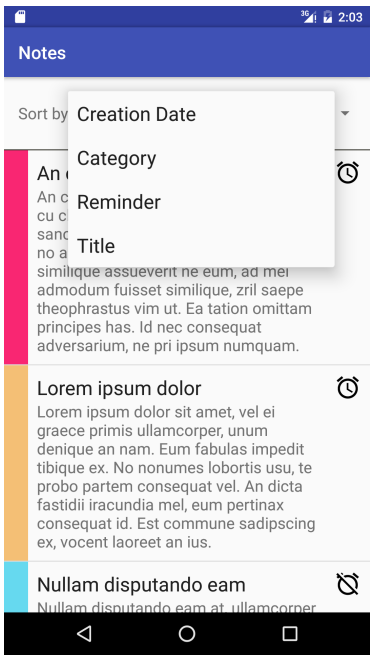
The spinner at the top of the activity indicates the sort order of the notes. Changes to the sort order results in changes to the dataset. Use the technique in class of updating the adapter data and calling `notifyDataSetChanged();`.

Spinners are populated like `ListViews`: they require an adapter to present each spinner item. Since the spinner items are text only, a simple layout and `ArrayAdapter` is all that's needed.



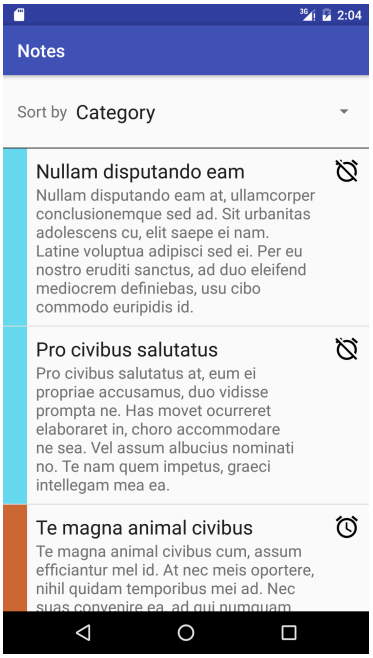
List sorted by title.

⇒



Choose another sort method.

⇒



List sorted by category.

The following sort orders are supported:

Sort method	Description
Title	Sorts notes by title alphabetically in ascending order (A ◀ Z).
Creation Date	Sorts notes by the date they were created in descending order (present ▶ past).
Category	Groups notes by category, in any order.
Reminder	Sorts notes by reminder in ascending order (present ◀ future). Any note without a reminder will appear at the bottom of the list.

3.1 Option 1: sorting using comparators

This first option of sorting the list uses the `sort` method of the `Collections` class.

```
void sort(List<T> list, Comparator<T> comparator);
```

A comparator is an object that implements the `Comparator<T>` interface:

```java
public interface Comparator<T> {
    int compare(T o1, T o2);
}
```

The `compare()` method works like a `compareTo()` method:

$$\text{compare}(o_1, o_2) = \begin{cases} < 0 & \text{if } o_1 < o_2, \\ 0 & \text{if } o_1 = o_2, \\ > 0 & \text{if } o_1 > o_2. \end{cases}$$

Example: To sort a list of `String`s by their length ascending (recall: their default sorting behaviour is alphabetical), you would code:

```java
String[] data = new String[] {ab, a, , abcdef, abcd};
Collections.sort(data, new Comparator<String>() {
    @Override
    public int compare(String o1, String o2) {
        return o1.size() - o2.size();
    }
});
```

Enums (Optional). A really elegant solution to the spinner involves the advanced features of Java enumerations. Specifically, you can define enum constants with two fields: a `String` to populate the spinner, and a `Comparator<Note>` to sort the data. Take a look at this article or see me about it!

`https://docs.oracle.com/javase/tutorial/java/javaOO/enum.html`

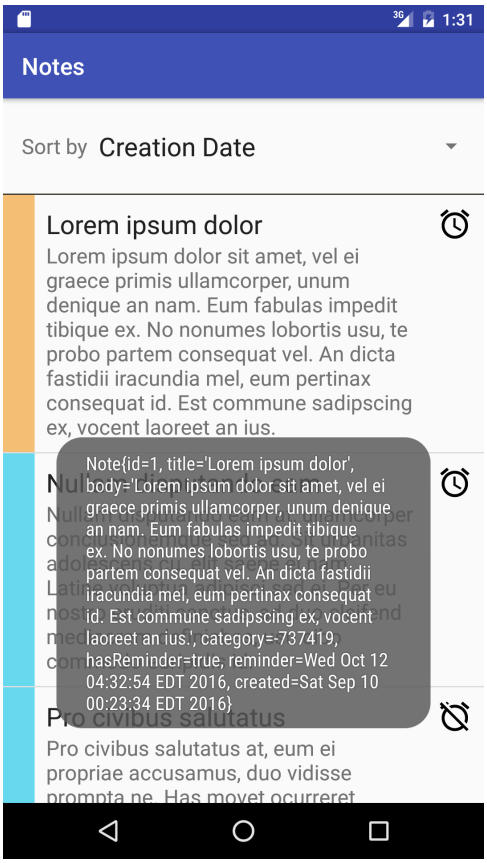
3.2 Option 2: sorting using custom queries

Coming soon…

4 Item Click

When the item is clicked the `String` representation of the note is toasted.

- Retrieve the note directly from the database, not the copy you made while making the adapter.
- Use the ID that is provided to the item click event handler. Recall that the default behaviour of the adapter is to use the position as the ID, which means you need to override the `getId()` method of your adapter subclass.



5 Requirements

- Your program should be clear and well commented. It must follow the "420-616 Style Guidelines" (on Léa).
- Create a second Activity for this Assignment, or create a new Android project with minimum SDK 21 or later.
- Your activity uses a fragment.
- Your `ListView` reads it's data from the `sqlite` database.
- Turning off reminders updates element in the database.
- Changing the sort order updates the adapter data and calls `notifyDataSetChanged();`.
- Clicking an item in the list reads the element from the database. Setup the adapter to have correct IDs for the list elements and use the ID passed in the event handler.
- Submit your project using git. Follow the Git Submission instructions on Léa.