# Changes

1.Added the scope of Searching
2.Revised some parts of Reminder's introduction

# Reason

According to the comments, we made a clearer description of what we worked on.

# Omni-Notes

# Feature 1: Searching

## 1.Introduction

Searching in Omni-Notes is to use keywords to find particular notes which include the keywords. The keywords can be contained in either notes' content or title. It can only be used in Note List interface to find notes, just as the picture below, click the icon and enter the keywords:
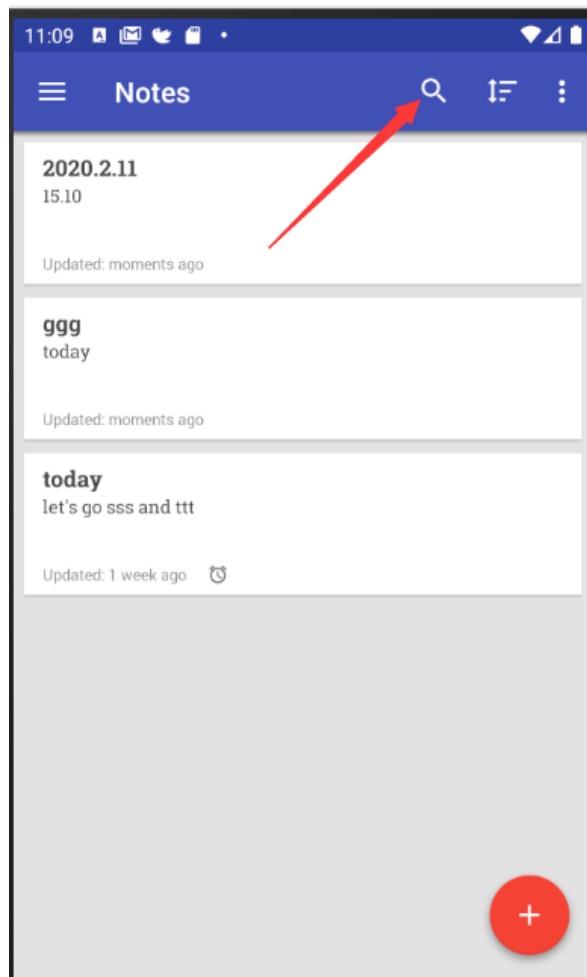
2.Process of finding

1. Use "search*" as a key word to do searching.
   As it shows below, we believe **SearchManager** and **SearchView** may be highly related to searching. Since both **SearchManager** and **SearchView** are in the **ListFragment.java** file. So we decided to go into this file first and try to find more information about how to implement searching.

2. In **ListFragment.java**, search methods with "search" in them. First, we found **initSearchView** method. It is used to initialize the view of search, where associating searchable configuration with the SearchView.



3. In the **initSearchView** method, there are two important callback functions. **onQueryTextSubmit** shows the list saved in SharedPreference with the query text in it, when. And **onQueryTextChange** begins to query in the background and save the data in SharedPreference, when the text changes.

```
MenuItemCompat.setOnActionExpandListener(searchMenuItem, new MenuItemCompat.OnActionExpandListener() {

    boolean searchPerformed = false;


    @Override
    public boolean onMenuItemActionCollapse (MenuItem item) {...}


    @Override
    public boolean onMenuItemActionExpand (MenuItem item) {

        searchView.setOnQueryTextListener(new OnQueryTextListener() {
            @Override
            public boolean onQueryTextSubmit (String arg0) {...}


            @Override
            public boolean onQueryTextChange (String pattern) {...}
        });
        return true;

    }
});
```

4. In the same file (**ListFragment.java**), we found some comments "// Search for a tag" and "// Searching" in **initNotesList** method. In **initNotesList,** there are two ways to do searching by keywords or tags.

```
void initNotesList (Intent intent) {
    LogDelegate.d("initNotesList intent: " + intent.getAction());

    progress_wheel.setAlpha(1);
    list.setAlpha(0);

    // Search for a tag
    // A workaround to simplify it's to simulate normal search
    if (Intent.ACTION_VIEW.equals(intent.getAction()) && intent.getCategories() != null
            && intent.getCategories().contains(Intent.CATEGORY_BROWSABLE)) {...}

    if (ACTION_SHORTCUT_WIDGET.equals(intent.getAction())) {...}

    // Searching
    searchQuery = searchQueryInstant;
    searchQueryInstant = null;
    if (searchTags != null || searchQuery != null || searchUncompleteChecklists
            || IntentChecker.checkAction(intent, Intent.ACTION_SEARCH, ACTION_SEARCH_UNCOMPLETE_CHECKLISTS)) {...} else {
        // Check if is launched from a widget with categories
        if ((ACTION_WIDGET_SHOW_LIST.equals(intent.getAction()) && intent.hasExtra(INTENT_WIDGET))
                || !TextUtils.isEmpty(mainActivity.navigationTmp)) {...} else {
            NoteLoaderTask.getInstance().executeOnExecutor(AsyncTask.THREAD_POOL_EXECUTOR, "getAllNotes", true);
        }
    }
}
```

# Feature 2: Reminder

## 1.Introduction

Reminder in Omni-Notes allows its users to set and modify reminder time for any specific note. Users can open a date & clock view and set/modify reminder time in it by clicking the clock icon or the reminder info text. When it's the reminder time, a notification pops up.
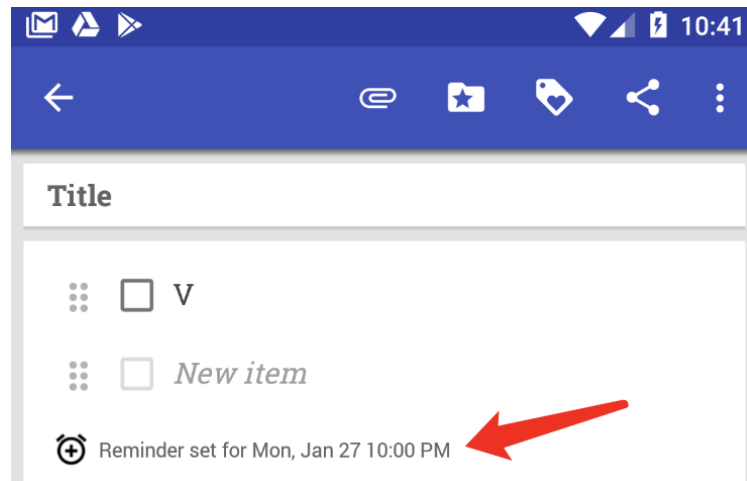


## 2.Process of finding

When users create a new note for a todo item, they usually need a reminder to notify them at some point of time. We first found the implementation of this feature in **DetailFragment**. The basic method is **initViewReminder**. This method is used to initialize the reminder view by binding listeners and set image/text resources to the reminder layout.

```java
private void initViewReminder () {
  reminderLayout.setOnClickListener(v -> {
    ReminderPickers reminderPicker = new ReminderPickers(mainActivity, mFragment);
    reminderPicker.pick(DateUtils.getPresetReminder(noteTmp.getAlarm()), noteTmp
        .getRecurrenceRule());
  });

  reminderLayout.setOnLongClickListener(v -> {
    MaterialDialog dialog = new MaterialDialog.Builder(mainActivity)
        .content(R.string.remove_reminder)
        .positiveText("Ok")
        .onPositive((dialog1, which) -> {
          ReminderHelper.removeReminder(OmniNotes.getAppContext(), noteTmp);
          noteTmp.setAlarm(null);
          reminderIcon.setImageResource(R.drawable.ic_alarm_black_18dp);
          datetime.setText("");
        }).build();
    dialog.show();
    return true;
  });

  // Reminder
  String reminderString = initReminder(noteTmp);
  if (!TextUtils.isEmpty(reminderString)) {
    reminderIcon.setImageResource(R.drawable.ic_alarm_add_black_18dp);
    datetime.setText(reminderString);
  }
}
```

There are two listeners. One is **setOnClickListener**. We found that the listener uses a class called **ReminderPickers**. In this class, we realized this class is highly relevant and all methods in it serve for the feature. A method called **pick** calls another method in the same class called **showDateTimeSelectors**. This method initializes a date time selector for the reminder so users can select a specific time in the graphic view.

The other listener is **setOnLongClickListener**. The listener works when users want to remove a reminder and long click on it. It uses a different class called **ReminderHelper**. The class imports an Android built-in class **AlarmManager** and contains many useful methods for modifying a reminder, including **addReminder**, **checkReminder** (it checks if any reminder exists for a given note), **removeReminder** and **showReminderMessage**. These methods are widely called by methods in other classes. Some usages are not highly relevant or real implementation for the reminder feature. For example, a method to delete a note would call the **removeReminder** method. However, other usages add to the implementation. The **addReminder** & **removeReminder** are called in **SnoozeActivity**. This class provides methods such as **setNextRecurrentReminder**, **updateNoteReminder** and **postpone** for postponing the reminder time if users choose to do so when the notification pops up.

```
package it.feio.android.omninotes;

import ...


public class SnoozeActivity extends AppCompatActivity implements OnReminderPickedListener {

  private Note note;
  private Note[] notes;

  public static void setNextRecurrentReminder(Note note) {...}

  private static void updateNoteReminder(long reminder, Note note) {...}

  private static void updateNoteReminder(long reminder, Note noteToUpdate, boolean updateNote) {...}

  @Override
  protected void onCreate(Bundle savedInstanceState) {....}

  private void manageNotification(SharedPreferences prefs) {...}

  private void postpone(SharedPreferences prefs, Long alarm, String recurrenceRule) {...}

  private void removeNotification(Note note) {...}

  @Override
  public void onReminderPicked(long reminder) {...}

  @Override
  public void onRecurrenceReminderPicked(String recurrenceRule) {...}

}
```

Then a reminder string is created according to the information users provide. **initReminder** is the method which forms and sets the reminder state string.

Given that **DetailFragment** is so large in code size, we continued exploring the reminder feature by searching for "reminder" in it. Surprisingly we found almost everything already gets handled by **initViewReminder**.

## Table1 Searching

| Folder | File | Method | Relevant? | Relevant how? | Confidence |
|--------|------|--------|-----------|---------------|------------|
| Omni-Notes\omniNotes\src\main\java\it\feio\android\omninotes | listfragment.java | initSearchView | Yes | Initialize the view of search | High |
| .. | .. | onQueryTextSubmit | Yes | A callback function, when we submit the changed query text, it will show the list saved in SharedPreference with the query text in it | High |
| .. | .. | onQueryTextChange | Yes | A callback function, when the text changes, it begins to query in the background and save the data in SharedPreference | High |
| .. | .. | initNotesList | Yes | When a note list is going to show, it should find all the contents according to current query (searching condition). There are two ways to do searching by keywords or tags. | High |

## Table2 Reminder

| Folder | File | Method | Relevant? | Relevant how? | Confidence |
|---|---|---|---|---|---|
| Omni-Notes\omni Notes\src\main\ja va\it\feio\android\o mninotes | detailFra gment | initViewRemi nder | Yes | When we create or modify a note, we can set a reminder, and this method is to initialize the reminder | High |
| .. | .. | initReminder | Yes | It's used to set & return an actual reminder state when initializing a note | High |
| .. | .. | addReminder | Yes | Add a reminder | High |
| .. | .. | trashNotes | Yes | When a note is deleted, the reminder set in this note will also be deleted | Medium |
| Omni-Notes\omni Notes\src\main\ja va\it\feio\android\o mninotes\utils | Reminde rHelper | addReminder | Yes | Add a reminder | High |
| .. | .. | checkReminder er | Yes | Check if any reminder for a given note exists | High |
| .. | .. | removeRemi nder | Yes | Remove a reminder | High |
| .. | .. | showRemind erMessage | Yes | Show the text of a note reminder in a toast after the reminder is set | High |
| Omni-Notes\omni Notes\src\main\ja va\it\feio\android\o mninotes\helpers\ date\RecurrenceH elper.java | Recurren ceHelper | getNoteRemi nderText | Yes | Form & return the text of a note reminder | High |
| .. | .. | getNoteRecu rrentReminde rText | Yes | Form & return the text of a note reminder | High |
| android.app.Alarm Manager | AlarmMa nager | | Yes | Android built-in class, used to set and show reminder | High |
| Omni-Notes\omni Notes\src\main\ja va\it\feio\android\o mninotes\Snooze Activity.java | SnoozeA ctivity | setNextRecur rentReminder | Yes | Set time for the next reminder according to the recurrence rule | High |
| .. | .. | updateNoteR eminder | Yes | Modify the state of a reminder | High |
| .. | .. | postpone | Yes | Select time based on the recurrence rule | High |