# Lab – User Interface Classes

*Create a complex user interface with Android's user interface classes*

**Objectives:**

Familiarize yourself with Android's User Interface (UI) classes. Create a simple application that uses a variety of UI elements including: Buttons, TextViews and Checkboxes. You will also reinforce the knowledge you've gained in previous lessons by implementing a larger portion of the application from scratch.

**Overview:**

In this Lab, you will create a ToDo Manager application. The application creates and manages a list of ToDo Items (i.e., things that you need "to do.") You will design this application's user interface, including its layout and resource files. You will also implement a bit more of the application's features than you did in previous Labs. Do NOT modify any resource IDs contained in the skeleton layout files as this may break our test cases.

**Exercise A: The Basic ToDo Manager**

The main Activity for this application is called Lab4_UILabs. When the Activity runs, but there are no previously saved ToDo Items, its initial UI will look something like this:

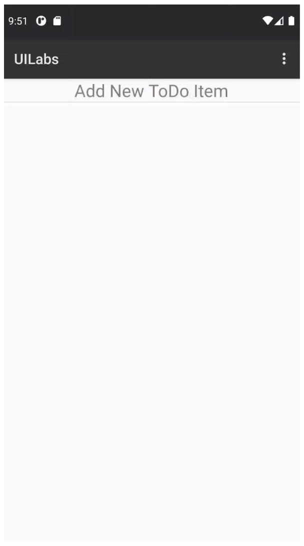


*Figure 1: Initial View*

This user interface contains a single ListView that displays all existing ToDo Items. As shown above, the last row of the ListView always displays a special View, with the words, "Add New ToDo Item." This last position within the ListView is known as the "footer." You can add a View to the footer by using the ListView's addFooterView() method.

When the user clicks on the ListView footer, the application will start a new Activity called AddToDo-Activity that allows the user to create and save a new ToDo Item.

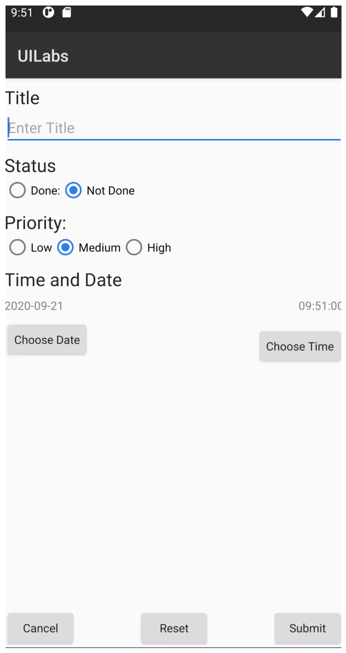


*Figure 2: Adding a New ToDo Item*

ToDo items have at least the following fields. Default values appear in bold:

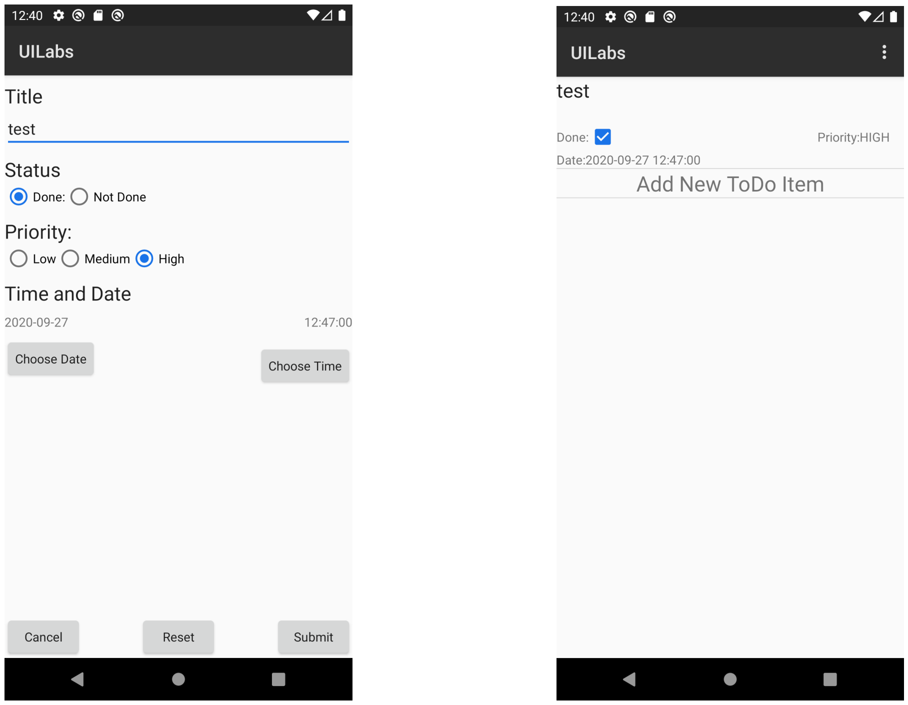
- Title: A user-provided String. The default Title is the empty String ("").
- Status: one of {Done, **Not Done**}
- Priority: one of {Low, **Med**, High}
- Time & Date: A deadline for completing this ToDo Item. The default deadline is **7 days from the current date and time.**

This Activity's user interface includes the following buttons:

- Cancel – finish the Activity without creating a new ToDo Item.
- Reset – reset the fields of the ToDo Item to their default values and update the display to reflect this.

- Submit – create a new ToDo Item containing the user-entered / default data fields and return it to ToDoManagerActivity. When the application returns to ToDoManagerActivity, the new ToDo Item should appear in the ListView.

For example, if the user creates and submits a new ToDo Item to an empty ToDo list, then once the application returns to the ToDoManagerActivity, its ListView will contain the new ToDo Item, as shown below.



Title
test

Status
[✓ Done:] Not Done

Priority:
Low Medium [✓ High]

Time and Date
2020-09-27 12:47:00

test
Done: ✓ Priority:HIGH
Date:2020-09-27 12:47:00
Add New ToDo Item

*Figure 3 (left): The user creates a new ToDo Item. Figure 3 (right) After submitting the new ToDo Item, the application returns to the main Activity, displaying the new ToDo Item.*

Back in the Main Activity, the user will be able to toggle the Done checkbox to indicate that the ToDo Item's status has changed, say from "Not Done" to "Done".

See the UILabs.mp4 screencast to see the app in action.

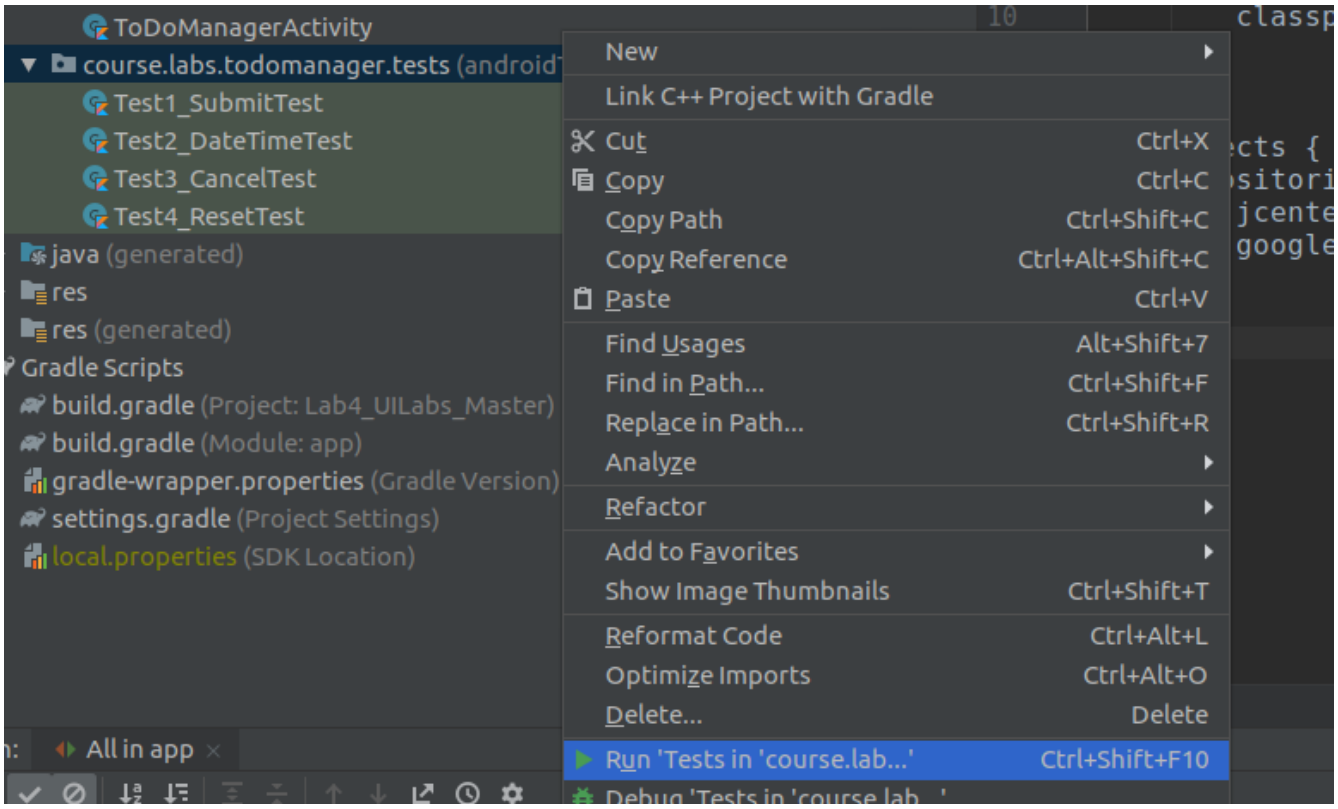
Implementation Notes:

1. Run git pull upstream master branch. Immediately push this to your project on gitlab to make sure you have it there as well.

2. Implement the project according to the specifications described above. To implement the Lab, look for comments in the skeleton files containing the String "//TODO." As with previous Labs, these comments contain hints as to what you need to do to complete the project. However, be aware that from here on out these comments will become increasingly less comprehensive, requiring you to make more decisions about how to implement the entire project and satisfy its requirements.
3. Don't forget that View's can be recycled in the ListAdapter's getView() method. When recycling a View, you should clear out ALL the View's old state.

## Testing

The test cases for this Lab are in the app/java/course/labs/todomanager/tests. You can run the test cases either all at once, by right clicking the folder course.labs.todomanager.tests and then selecting Run 'Tests' in 'course.labs...', or to run them one at a time, by right clicking on an individual test case class and then continuing as before.


ToDoManagerActivity
course.labs.todomanager.tests (androidT
Test1_SubmitTest
Test2_DateTimeTest
Test3_CancelTest
Test4_ResetTest
java (generated)
res
res (generated)
Gradle Scripts
build.gradle (Project: Lab4_UILabs_Master)
build.gradle (Module: app)
gradle-wrapper.properties (Gradle Version)
settings.gradle (Project Settings)
local.properties (SDK Location)
All in app
New
Link C++ Project with Gradle
Cut Ctrl+X
Copy Ctrl+C
Copy Path Ctrl+Shift+C
Copy Reference Ctrl+Alt+Shift+C
Paste Ctrl+V
Find Usages Alt+Shift+7
Find in Path... Ctrl+Shift+F
Replace in Path... Ctrl+Shift+R
Analyze
Refactor
Add to Favorites
Show Image Thumbnails Ctrl+Shift+T
Reformat Code Ctrl+Alt+L
Optimize Imports Ctrl+Alt+O
Delete... Delete
Run 'Tests in 'course.lab...' Ctrl+Shift+F10
Debug 'Tests in 'course.lab...'

As you implement various steps of the Lab, run the test cases every so often to see if you are making progress toward completion of the Lab.

**Warnings:**

1. These test cases have been tested on a Pixel 3 AVD emulator with API level 29. To avoid configuration problems, you should test you app against a similarly configured AVD. Also, when testing, make sure that your device is in Portrait mode when the test cases start running.

2. On startup, these test cases delete all existing ToDoItems.

3. All tests will be weighted equally when being graded.

**Submission**

To submit your project just save and commit all your changes locally and push to your gitlab project. In the commit message for your final solution please leave the following message: “completed Lab4 implementation”.

If you got through Exercise A and submitted and passed the tests, and feel that you'd like to do more, here are some suggested additions. This is optional and ungraded, but if you do something cool, please consider posting some screenshots on the forums.

Optional Exercise B: A Prettier ToDo Manager Application

Right now the ToDo manager is quite ugly. Try modifying the layout files to create more attractive and useable layouts. For example, play with the font size of the text, the amount of padding around the elements, background colors and more.

Optional Exercise C: An improved ToDo Manager Application

Modify your application so that ToDoItems whose status is Not Done are displayed in the Main Activity with a different colored background than those whose status are Done. If you do this, then when the user toggles the Done checkbox, the background color should also change as appropriate. You might also consider changing the background color or adding a warning icon as the ToDo Item’s deadline get close. Finally, right now, the user can't modify the ToDoItem's priority from the ToDoItem ListView. Modify this user interface so that it provides a drop down list, allowing users to select a different Priority.

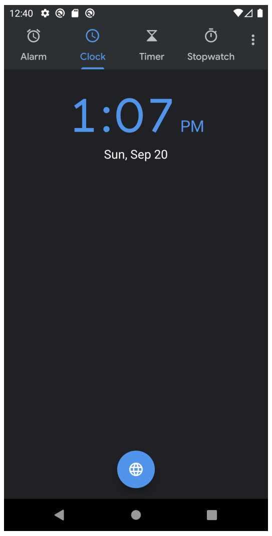
Optional Exercise D: A ToDo Manager Application You Might Actually Use

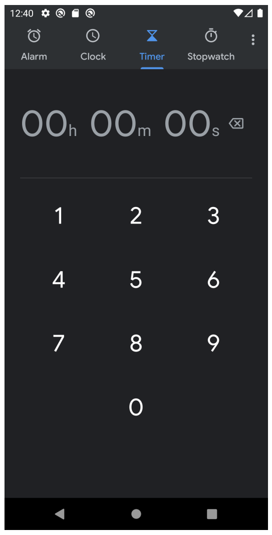
Modify your application so that if the user long presses a ToDo Item in the Main Activity’s ListView, a dialog pops up, allowing the user to delete the selected ToDo Item. If you're really feeling adventurous place the app inside a Tab. Have one Tab display ToDoItems sorted by deadline, and another Tab for ToDoItems first sorted by priority, and within a particular priority, further sorted by deadline.

Optional Exercise E: ViewGroups and Layouts

Open a new project with an empty activity and experiment with how the different ViewGroups can arrange elements on the screen. For example, how are classes like LinearLayout, RelativeLayout, AbsoluteLayout, and FrameLayout different? How might you use them to create a layout like the alarm clock display below? What would you do differently in the timer screen on the right? Try to recreate these in the project you just created.

**Hint**: First focus on understanding how the main content of each screen is laid out (like the time and display in the clock and the buttons in the timer) and then think about how you may incorporate that content in the tabs.


12:40
Alarm
Clock
Timer
Stopwatch
1:07 PM
Sun, Sep 20


12:40
Alarm
Clock
Timer
Stopwatch
00h 00m 00s
1
2
3
4
5
6
7
8
9
0