

Mobile Computing

1. Lab Assignment: Detail Activity for Creating / Editing Tasks

11.04.2024

Presentation of results on: 25.04. (A) / 02.05. (B)

Objectives

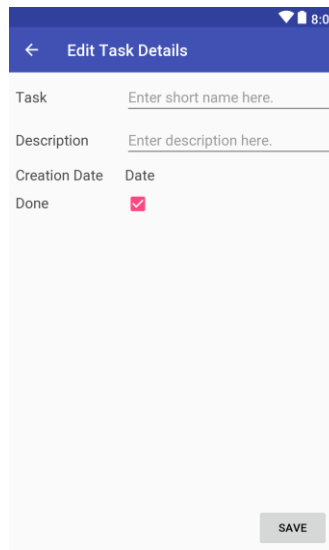
- Install and setup Android Studio
- Create first app
- Work with views and layouts

Your Task

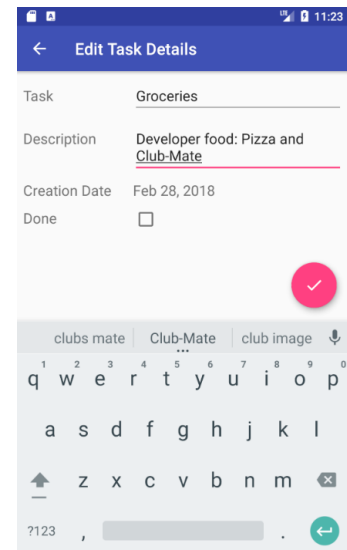
Create an app with a single activity (e.g. „TaskDetailActivity“) for showing task objects and editing task properties.

Use the Task class provided in the AULIS group (folder “Lab Assignments (Standard Track)”). The following properties should be editable in your activity:

- short name
- description
- done



This screenshot shows the 'Edit Task Details' screen on a mobile device. The title bar is blue with a back arrow and the text 'Edit Task Details'. The screen contains four input fields: 'Task' with placeholder text 'Enter short name here.', 'Description' with placeholder text 'Enter description here.', 'Creation Date' with the value 'Date', and 'Done' with a checked checkbox. A 'SAVE' button is located at the bottom right.



This screenshot shows the 'Edit Task Details' screen on a mobile device, similar to the left one but with different data. The title bar is blue with a back arrow and the text 'Edit Task Details'. The screen contains four input fields: 'Task' with the value 'Groceries', 'Description' with the value 'Developer food: Pizza and Club-Mate', 'Creation Date' with the value 'Feb 28, 2018', and 'Done' with an unchecked checkbox. A pink floating action button with a checkmark is visible on the right side. A keyboard is shown at the bottom.

The creation date is initialized during instantiation of the task and not editable.

The layout should be filled with attribute values of a Task object. Values entered by the user should update the Task object when a “Save” button is pressed.

The layout of your view might be as in the example illustrations (left: using classic Button for saving; right: using Floating Action Button). Your app should work on different devices with different screen sizes and in both screen orientations (portrait and landscape):

- Make sure that your layout reacts adequately to these requirements.
- Make sure that your app does not lose progress (i.e. the Task instance or simply text the user entered into the user interface)

See page 2 for some hints.

Hints

- 1.) Alignment of view elements in layouts: In the above layouts, the views showing task data (text fields, date field, checkbox) are left-aligned to a “virtual” vertical line (in the example at about 40% of the screen width). Do not use fixed coordinates to achieve this! Do some research on the (invisible) *guideline* elements provided by Android’s constraint layout instead.
- 2.) Date objects can be converted to a String and formatted for output using, e.g., `DateFormat.getDateInstance().format(mCreationDate)`
- 3.) You may choose whether to use a classic Save button (cf. left image) or a Floating Action Button (FAB; cf. right image). Your activity will already be provided with a Floating Action Button, if you create your activity in Android Studio via *context menu* → *New* → *Activity* → *Basic Views Activity*. I do NOT recommend to use this project template for beginners, since Android Studio generates rather complex Java and layout code. I suggest you rather stick to the *Empty Activity* template and – if you want to add a FAB later - do some research on how to integrate it into your layout and code as soon as you have become familiar with the basic concepts of Android development.

Bonus Tasks

For this exercise sheet, only one bonus task is defined: replace the *creation date* (text label) with a *due date*.

- Can use integrate a (small?) date picker into your layout or do you just rely on a text field?
- How can you recognize, reject, and visually mark invalid values such as dates that are in the past?
- Remember to change the attribute in the Task class!

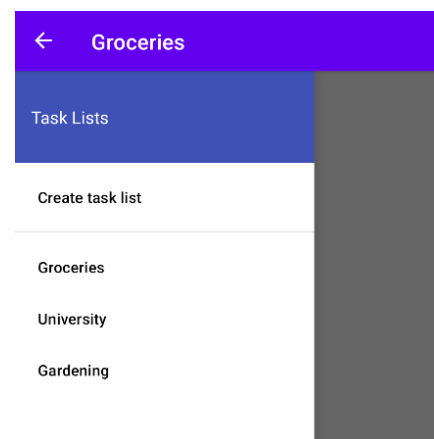
Future exercise sheets will contain smaller bonus tasks that focus on the user experience and larger bonus tasks that increase the overall complexity by additional functionality.

To give you a broad idea of this additional functionality:

The To Do app should support an arbitrary number of task lists, i.e. not only one. This has implications in at least the following aspects:

- Store task lists in a repository / database.
- Make functionality for selecting and creating task lists accessible, e.g. using a *navigation drawer* (cf. figure →) and a dialog for naming new task lists.

These features may seem simple at first sight, but will have major implications on other bonus tasks regarding app architecture.



Note: you are not required to finish defined bonus tasks until the next lab meeting!
Bonus tasks may also be part of your final submission.