Lab 3

Lab 3a

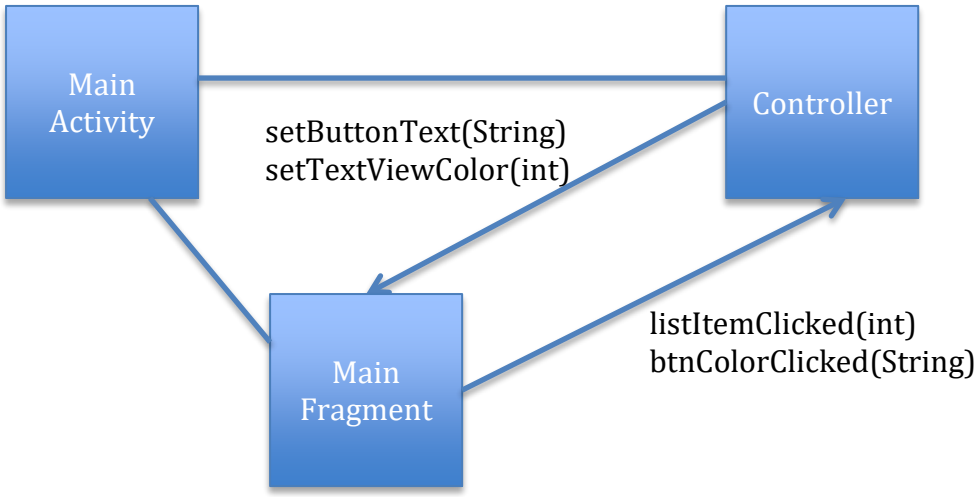
Create an app with a Text View, a Button, and a List View. Your solution should contain at least an Activity, Fragments, and a Controller class.

Handle clicks on the ListView / Button by triggering the necessary actions.

App features:

- At the start, the text of the TextView must be red, the text on the button must be red, and the ListView should contain four lines.
- When the user clicks on a row in the ListView, the text in the clicked row is set in the Button. The figure displays the interface after the user clicked on yellow.
- When the user clicks on the button, the color shown on the button is set as the background color in the TextView. Notice that there are ready-color constants: Color.RED, Color.BLUE, Color.YELLOW, Color.GREEN.
  The figure displays the state of the UI after clicking on the button.

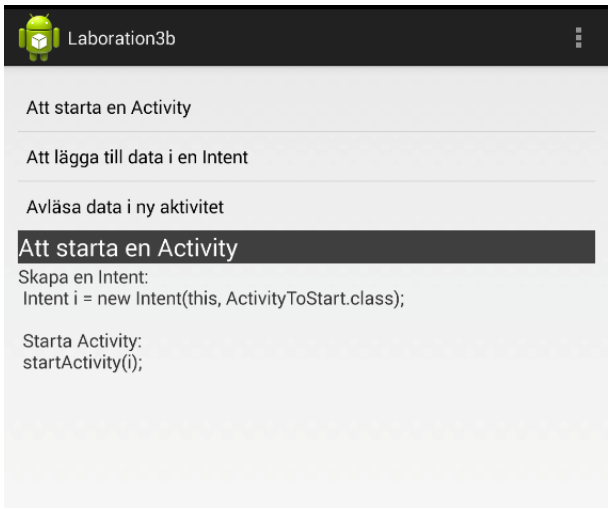
Sketch a valid design. You can start from the one below. You may need to extend it, though.



Lab 3b

Give a new layout to the app HOWTO from the previous lab. WhatToDo strings shall be displayed in a ListView. Instructions shall be displayed after clicking on any item in the list.

There will be a fragment holding the list and another fragment showing the instructions. The fragments should be placed statically, ie by <fragments> tags in activity_main.xml..



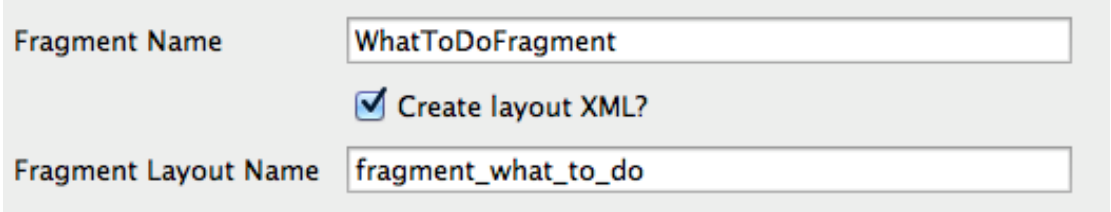
App features:
At the start, an instruction is displayed (see figure on the right)

When the user clicks on an item in the list, the related instructions are displayed.

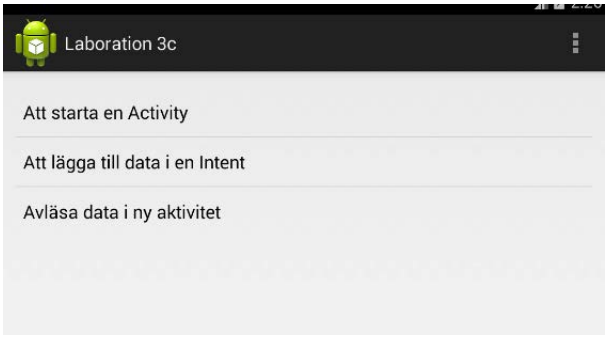
Your solution should include the following classes:

- An Activity
- Two Fragments
- A Controller
- The Instructions
- An InstructionAdapter - the class inflating the contents for the ListView from a string list.

When you create a fragment, right-click Source Package - New - Fragments - Fragments (Blank). Eg



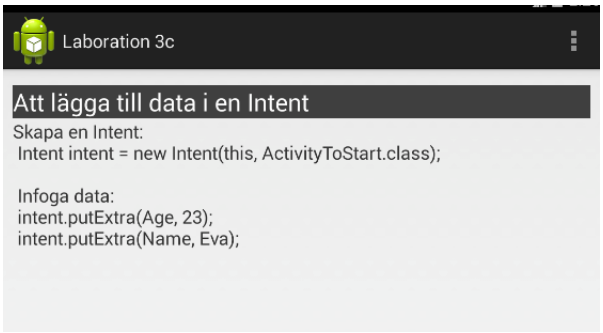
Lab 3c



Improve HOWTO's design even more.

Separate the list from the instructions in two different Fragments. These Fragments must be assigned dynamically. The idea is that the app dynamically shifts fragments.

Hitting the back button on the device when reading instructions should make the UI go back to the instructions list. So this method may look like this (in the Activity class):

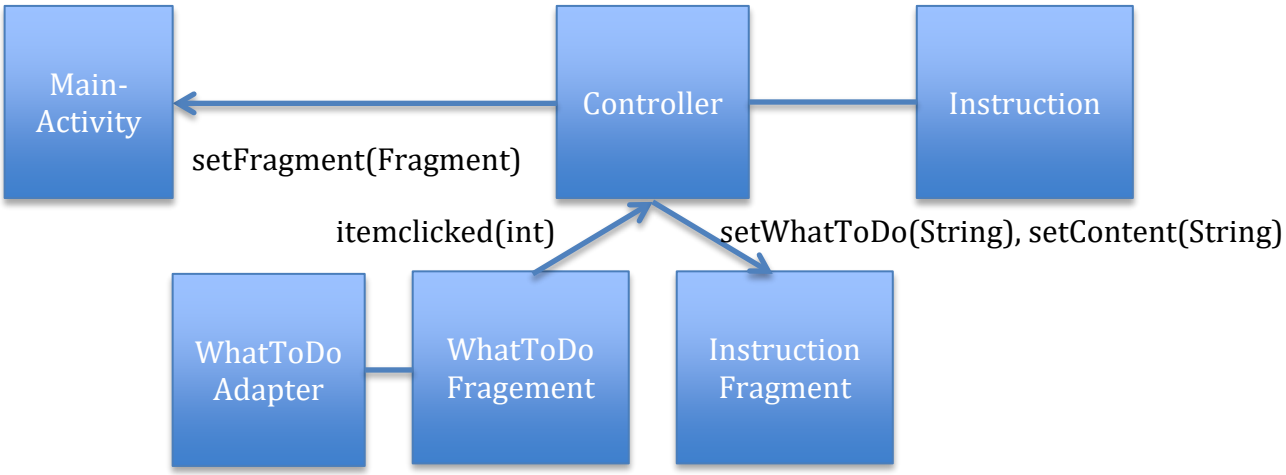



```
public void setFragment(Fragment fragment, boolean backstack) {
    FragmentManager fm = getFragmentManager();
    FragmentTransaction ft = fm.beginTransaction();
    ft.replace(R.id.container, fragment);
    if(backstack) {
        ft.addToBackStack(null);
    }
    ft.commit();
}
```

The WhatToDo fragment should be placed from the beginning, with the second argument set to false. The Instruction fragment should be placed after each click with the second argument set to true.

The same classes in the lab 3b should be part of this solution.

WhatToDoFragment shall report to the Controller every time the user clicks on a list item. The Controller updates the InstructionFragment and then notifies MainActivity to display the InstructionFragment. When the user clicks the Back button, the ListView reappears.


Main-
Activity
Controller
Instruction
setFragment(Fragment)
itemclicked(int)
setWhatToDo(String), setContent(String)
WhatToDo
Adapter
WhatToDo
Fragement
Instruction
Fragment